

# **Toxic Comment Classification Using Hybrid Deep Learning Model**

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS  
FOR THE AWARD OF THE DEGREE  
OF

**MASTER OF TECHNOLOGY**

IN

**COMPUTER SCIENCE AND ENGINEERING**

Submitted By

**Archna Kumari**

**2K18/CSE/02**

Under the supervision of

**Mr. Rohit Beniwal**

**(Assistant Professor)**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**DELHI TECHNOLOGICAL UNIVERSITY**

(Formerly Delhi College of Engineering)

Main Bawana Road, Delhi-110042

**JUNE, 2020**

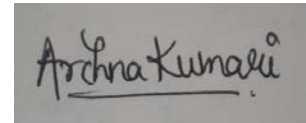
**DELHI TECHNOLOGICAL UNIVERSITY**

(Formerly Delhi College of Engineering)

Main Bawana Road, Delhi-110042

**CANDIDATE'S DECLARATION**

I, **Archna kumari (2K18/CSE/02)** of M.Tech, Computer science and Engineering, hereby declare that the project Dissertation titled "Toxic Comment Classification Using Hybrid Deep Learning Model" which is submitted by me to the Department of Computer Science and Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associated, Fellowship, or other similar title or recognition.



Place: Delhi

Date: 11/07/20

**ARCHNA KUMARI**

**2K18/CSE/02**

**COMPUTER SCIENCE & ENGINEERING**

**DELHI TECHNOLOGICAL UNIVERSITY**

(Formerly Delhi College of Engineering)

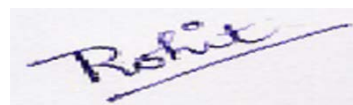
Main Bawana Road, Delhi-110042

**CERTIFICATE**

I hereby certify that the Major Project-II work entitled “**Toxic Comment Classification using Deep Learning Model**” which is submitted by **Archna Kumari (2k18/CSE/02)** to Delhi Technological University, in partial fulfillment of requirements for the award of the degree of Master of Technology (Computer Science and Engineering) is a record of the Major Project-II carried out by the student under my supervision. To the best of my knowledge this work has been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Date: 11/07/20



**Mr. Rohit Beniwal**

**SUPERVISOR**

## ABSTRACT

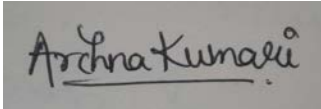
With the increasing availability of affordable data services and social media presence, our life is not untouched with ‘cyber,’ i.e., electronic technology. With it, various challenges and issues are faced, and the most sensitive among them is Cyberbullying. Cyberbullying, in the form of ‘abusive,’ ‘offensive,’ ‘inappropriate,’ and ‘toxic’ comments are present on the platforms. In fear of online abuse and bullying, many people give-up on perceiving different opinions and stop expressing them. Nowadays, various online platforms like Quora, Wikipedia, Twitter, and Facebook have become part and parcel of everybody's life. These stages battle to viably encourage discussions, driving numerous networks to restrict or shutdown client remarks. Unfortunately, online comments with toxicity cause online badgering, bullying, and personal attacks. Therefore, toxic comment classification problem has attracted the attention of many organizations from the past few years. Hence, in this paper, we present a hybrid Deep Learning model that will detect such toxic comments and classify them according to the type of toxicity. As an outcome, we achieved the best results with an accuracy of 98.39% and an f1 score to 79.91%.

## **ACKNOWLEDGMENT**

First of all, I would like to express my deep sense of respect and gratitude to my project supervisor Mr. Rohit Beniwal for providing the opportunity of carrying out this project and being the guiding force behind this work. I am deeply indebted to him for the support, advice, and encouragement he provided without which the project could not have been a success.

Secondly, I am grateful to Dr. Rajni Jindal, HOD, Computer Science & Engineering Department, DTU for her immense support. I would also like to acknowledge the Delhi Technological University library and staff for providing the right academic resources and environment for this work to be carried out.

Last but not least I would like to express sincere gratitude to my parents and friends for constantly encouraging me during the completion of work.



**ARCHNA KUMARI**

**Roll No – 2K18/CSE/02**

# TABLE OF CONTENTS

|   |             |
|---|-------------|
| <b>Candidate’s Declaration</b> .....        | <b>i</b>    |
| <b>Certificate</b> .....                    | <b>ii</b>   |
| <b>Abstract</b> .....                       | <b>iii</b>  |
| <b>Acknowledgement</b> .....                | <b>iv</b>   |
| <b>Contents</b> .....                       | <b>v</b>    |
| <b>List of Figures</b> .....                | <b>viii</b> |
| <b>List of Tables</b> .....                 | <b>ix</b>   |
| <b>List of Symbols, Abbreviations</b> ..... | <b>x</b>    |
| <b>CHAPTER 1 INTRODUCTION</b> .....         | <b>1</b>    |
| 1.1 Motivation .....                        | 1           |
| 1.2 Objective of Study .....                | 2           |
| 1.3 Research Goals.....                     | 3           |
| 1.4 Organization of Thesis.....             | 4           |
| <b>CHAPTER 2 LITERATURE SURVEY</b> .....    | <b>5</b>    |
| 2.1 Theoretical Background.....             | 5           |
| 2.1.1 Internet and Web.....                 | 5           |
| 2.1.2 Sentiment Analysis .....              | 7           |
| 2.1.2.1 Error Analysis .....                | 8           |
| 2.2 Background and Previous work.....       | 10          |
| <b>CHAPTER 3 DEEP LEARNING MODELS</b> ..... | <b>12</b>   |
| 3.1 Deep Neural Network .....               | 12          |
| 3.2 Convolution Neural Network.....         | 13          |
| 3.3 Gated Recurrent Unit .....              | 14          |

|   |           |
|---|-----------|
| <b>CHAPTER 4 METHODOLOGY .....</b>        | <b>17</b> |
| 4.1 Dataset .....                         | 17        |
| 4.2 Data Preprocessing.....               | 17        |
| 4.3 Embedding Layer .....                 | 17        |
| 4.4 Convolution Layer .....               | 18        |
| 4.5 Max Polling Layer .....               | 19        |
| 4.6 Bidirectional-GRU Layer .....         | 19        |
| 4.7 Global Max Polling Layer .....        | 19        |
| 4.8 Dense Layer 1 .....                   | 19        |
| 4.9 Dropout Layer .....                   | 19        |
| 4.10 Dense Layer 2 .....                  | 19        |
| <b>CHAPTER 5 IMPLEMENTATION.....</b>      | <b>20</b> |
| 5.1 Dataset .....                         | 20        |
| 5.2 Data Preprocessing.....               | 21        |
| 5.3 Embedding Layer.....                  | 21        |
| 5.4 Convolution Layer .....               | 22        |
| 5.5 Max Polling Layer .....               | 22        |
| 5.6 Bidirectional-GRU Layer .....         | 23        |
| 5.7 Global Max Polling Layer .....        | 24        |
| 5.8 Dense Layer 1 .....                   | 24        |
| 5.9 Dropout Layer .....                   | 24        |
| 5.10 Dense Layer 2 .....                  | 24        |
| <b>CHAPTER 6 TECHNOLOGICAL STACK.....</b> | <b>25</b> |
| 6.1 Functional Requirements .....         | 25        |
| 6.2 Software Requirements.....            | 25        |

|   |           |
|---|-----------|
| 6.3 Pre-Requisite python libraries to be installed..... | 25        |
| <b>CHAPTER 7 RESULT AND ANALYSIS.....</b>               | <b>26</b> |
| 7.1 Evaluation Metrics.....                             | 26        |
| 7.2 Result.....   | 27        |
| <b>CHAPTER 8 CONCLUSION AND FUTURE SCOPE.....</b>       | <b>30</b> |
| <b>Appendix.....</b>                                    | <b>31</b> |
| <b>References.....</b>                                  | <b>35</b> |
| <b>List of Publications.....</b>                        | <b>37</b> |



## LIST OF FIGURES

|  |    |
|--|----|
| FIG 1: Simple Neural Network and Deep Learning Neural Network..... | 12 |
| FIG 2: CNN full connected structure.....                           | 14 |
| FIG 3: A GRU Cell.....   | 16 |
| FIG 4: Proposed Model for Toxic Comment Classification.....        | 19 |
| FIG 5: Sample instance of the dataset .....                        | 21 |
| FIG 6: Description of train dataset.....                           | 23 |
| FIG 7: Distribution of Comment Length .....                        | 24 |
| FIG 8: Model Summary of the Hybrid Model.....                      | 25 |
| FIG 9: Result of first ten epochs .....                            | 29 |
| FIG 10: Result of last ten epochs.....                             | 30 |

# LIST OF TABLES

**1. Defines the distribution of labels in the training set..... 22**

# LIST OF ABBREVIATIONS

1. SVM: Support Vector Machine
2. TF-IDF: Term Frequency-Inverse Document Frequency
3. LSTM: Long Short Term Memory Cell
4. CNN: Convolutional Neural Network
5. RNN: Recurrent Neural Network
6. ANN: Artificial Neural Network
7. KNN: K-nearest Neighbors
8. LDA: Linear Discriminated Analysis
9. NB: Naïve Bayes
10. Bi-GRU: Bidirectional-Gated Recurrent Unit
11. Relu: A rectified linear unit
12. SGD: Stochastic Gradient Descent
13. NLP: Natural language processing

# INTRODUCTION

In today's digital era, social media provides a common platform that let users express their opinion in the form of online comments. People consider it their freedom of expression; however, many users use this fundamental right in a negative way, such as disrespecting other users, threatening other users, spreading fake news, cyberbullying, personal comments, toxic comments, etc. on online discussion platforms. Those comments, which are disrespectful and rude, and that force users to leave the conversation or online discussion are called "Toxic Comments." Nowadays, users face issues like abuse, harassment, cyberbullying online threats, and hate speeches, which can be classified as toxic comments. Therefore, such comments need to be recognized as quickly as possible and should be removed from the internet, but it is not that simple. It is a tedious task to filter and ban such comments.

According to the Pew survey (2014) [1] about online harassment, some key findings are that every four in ten, i.e., 40% of internet users are victims of online harassment, purposeful embarrassment, and stalking. Both men and women experience a different kind of online harassment where women face it more frequently. Men experience fewer instances of verbal abuse, embarrassment, and threats, which are "less severe". In contrast, women experience badgering, such as being followed, inappropriate behavior, and threats on a more severe level.

## 1.1 Motivation

The Deep Learning model identifies whether or not a comment is toxic. In the case of toxic, it further categorizes the comment in six different labels, namely "toxic, severe toxic, obscene, threat, insult, and identity hate". All the listed labels are not mutually exclusive. Comment classification problems are generally also known as multi-class classification or multi-label classification [2]. Multi-class classification means data or comment belongs to only one out of the six labels. In contrast, Multi-label classification comment belongs to more than one label simultaneously. For example, a comment can be both insulting and threatening simultaneously.

Various online platforms are taking different initiatives to make their platform free from problems such as toxic comments, online harassment, and provide a safe online environment for their users. A few of the platforms even turn off comments for such posts based on crowdsourcing votes (upvotes/downvotes). Manually identifying such comments and flagging them is a time-consuming and challenging exercise. However, such an exercise is inefficient and not scalable. Comment classification is a classic example of Natural Language Processing (NLP) and a fundamental part of numerous applications such as web search, text mining, and sentiment analysis, etc. Hence, a wide scope of machine learning strategies has also been applied for comment classification.

In the recent past, Google and Jigsaw started a venture called "Perspective", which uses AI to distinguish toxic comments naturally [3]. The perspective API [4] score represents the impact of the comment in the discussion so that platforms can use this score to provide real-time feedback to users. In most of the cases, this model is not reliable, inclined to blunders, and the degree of toxicity is not determined.

## **1.2 Objective of the study**

The objective of the study is to study the current solutions that are available for handling the toxic comments and come up with a better solution than the existing solutions available. Right now, almost all the comments on social media websites are filled with toxic comments. Therefore, in this paper, we are using a hybrid Deep Learning model for improving the performance of toxic comment classification. To be particular, we analyze the dataset to understand how to process the data. Preprocessing and word embedding layer form a matrix, then we feed the matrix to Convolutional Neural Network (CNN) and Bidirectional Gated Recurrent Unit (Bi-GRU) layer respectively. Noise and important features are filtered out through CNN. After this, dense and dropout layers further perform the classification. Hence, we provide a multi-label classification model that is capable of recognizing different types of toxicity, such as "severe toxic, threats, obscenity, insults, and identity-based hate". Moreover, we are providing probability estimates for each sub-type, which is conclusively strong enough to outperform 'Perspective' API's current model.

### **1.3 Research Goals**

The goal of this research will be to utilize deep learning models to recognize harmfulness in comments, which could be utilized to help dissuade clients from posting conceivably toxic messages, craft increasingly polite contentions while taking part in a talk with others and to measure the poisonousness of different user comments.

To viably encourage discussions on online social media platforms, driving numerous platforms to constrain or shut down such user comments. This research centers around building a multi-label model to identify various kinds of toxic comments like, "severe toxic, threats, obscenity, insults, and identity-based hate".

The contributions of this paper are as per the following:

1. Correlations between existing automatic content moderation strategies can assist us with understanding the constraints of existing techniques and identify gaps.
2. This research work will gather and separate toxic and non-toxic comments from the dataset.
3. This paper will build up a multi-label model to identify various kinds of toxic comments.

## **1.4 Organization of thesis**

In the thesis:

### **Chapter One**

Spotlight on giving some presentation on what is the fundamental inspiration for our thesis. It additionally characterizes the problem statement and target of our research.

### **Chapter Two**

Define the Literature Review having Theoretical Background on Sentiment Analysis, Internet, and web. This chapter also organizes the recent work carried out in the field of Toxic comment classification.

### **Chapter Three**

Define the deep learning models that we going to use in the project.

### **Chapter Four**

Define the Methodology which is divided into the ten phases, namely dataset, data preprocessing, embedding layer, convolution layer, max-pooling layer, Bi-GRU layer, global max-pooling layer, dropout layer, and two dense layers.

### **Chapter Five**

Define the implementation of the system along with the system design along with the features of the proposed system.

### **Chapter Six**

Define the execution of the system along with the functional and non- functional requirements technology stack.

### **Chapter Seven**

Defines the result obtained for the execution of the system. It also defines the analysis carried out to further justify our project's performance.

### **Chapter Eight**

Define the corresponding conclusion, limitations, and future work.

At last, the references used in the thesis making are listed.

# CHAPTER 2: LITERATURE SURVEY

## 2.1 Theoretical Background

### 2.1.1 Internet and Web

Internet is known to be an interconnected system that associates various sorts of frameworks (PCs, cell phones, pc, and so on.) with one another which utilizes a standard structure (TCP/IP) for connecting these gadgets. Utilizing the Internet one can get to various pages that are connected with hypertext archives, messages, and document sharing.

The Internet has become a significant bit of our lives. It is changing rapidly so are we. As an ever-increasing number of individuals have begun utilizing it, the web is likewise experiencing a principal convenient. In the previous scarcely years, electronic web records are accomplishing fame as a way that representations singular encounters and assumptions. Web (or "www") is now and then alluded to as equivalent words of the Internet however rather, the web is only a bundle that utilizes the Internet to compose itself.

From the most recent couple of decades, the web is ascending with interlinked hypertext pages, pictures, recordings et al. The progressing web innovations, for example, HTML, CSS, and XML, etc. guarantee that all the electronic substance is bolstered by all the programs. The interface between web advances and programs assists with building intelligent web applications. The idea of the web was given by Tim Berners Lee, otherwise called the dad of Web, in 1989 in his examination. He clarified the significance of the development of web innovations. Some progressive thoughts were given before, for example, decentralization (no authorization required), non-separation (Net Neutrality), base up configuration (coordinated code), accord (straightforwardness), and comprehensiveness (same language).

The Internet has gotten an amalgamated, perfect, and an essential piece of our lives. It is changing quickly so are we. As an ever-increasing number of individuals have begun utilizing it, the Web is additionally experiencing a vital catalyst. In previous years, electronic archives are accomplishing fame as a way that representations different experiences and opinions. Addition of Web 2.0 offered to ascend to applications, for example, smaller scale blogging, gatherings, person to person communication locales, wikis, and so on.



- **Blogs**

The blog is an expressive online diary that lets clients post various updates. Post can be as text, pictures, sound, or a video record. It is the focal point of some commonly talked about subjects around the users.

- **Wikipedia**

Wikis are the cooperative pages of various creators that permit them to post pertinent data and connections to some popular explores. Clients can discover everything about that particular topic. Creators on wikis are given a different secret key to change or to modify the data gave by them.

- **Social Networking sites**

Social Networking sites like Twitter, Facebook, Instagram, Reddit, and so on are currently a day's utilization to post ordinary updates and status of clients by setting up an individual client id. They permit clients to post photographs, recordings, criticism, remarks, and so on.

- **Discussion forums**

Discussion forums are the platforms user or client can connect with one another on a specific inquiry or question. Like one post a problem or inquiry then different clients will offer answers to this inquiry and express their conclusion on what they think.

With the development of Web 2.0, which accentuation client produced content, the manner in which individuals used to communicate their perspectives and sentiments has additionally changed noticeably. Thoughts, remarks, sees, recommendations, criticisms are shared by the clients. The comments that are disrespectful, rude, and force users to leave the conversation or online discussion are called “Toxic Comments”. Nowadays user face issues like abuse, harassment, cyberbullying online threats and hate speech that all cover in the toxic comments Such content need to identify as soon as possible and remove from the internet but it is not that easy task to identify such comments and flags them that are such a difficult and time-consuming method. Sentiment analysis is also used for toxic comment classification.

### **2.1.2 Sentiment Analysis**

Sentiment Analysis now and then referred to as "opinion mining", used to determine how users are responding to a particular issue. It is the study that determines the attitude of the writer on some documents. It is a kind of text classification that determines the opinions of the writer. The polarity of opinions is classified into three types as positive, negative, and neutral. To calculate the percentage of emotions in any comment or view we first need to differentiate essential features from it and then classify it. For selecting a feature, the subset is maintained but due to hundreds of thousands of datasets the size of the search subset is difficult to maintain which leads to redundant data and ambiguity of features leading to a reduced amount of accuracy and precision.

To moderate the toxic comments different filters are used but users are so smart they constantly find a new way to pass the comments like these filters detect toxic comments based on abusive words used as the keywords in the sentence but users use the wrong spelling of these words and pass the filter still sentiment of comments are so difficult to hide therefore with the help of sentiment information we can still find the toxic comments. Hence, there is an important relation between sentiment and toxicity in social network messages.

Two basic strategies are used by online platforms to moderate the messages and try to stop toxic comments in online discussions. The first strategy is known as human surveillance. In this, each comment is checked by human surveillance before posting online on the platform. This is such a time-consuming strategy and a lead to a severe slowdown in the conversation because checking each comment one by one is not practically possible. The second strategy is keyword detection and this one is faster than the previous strategy. In this strategy, toxic comments are filtered based on the set of denial words. If comments have denial words then such comments are removed or moderated by the platform. But this method has a limitation such that different internet users introduce minor misspelling of denial words for example: fuck off can be written as fuuuck off. These minor changes help to skip the toxic comment detection system. Therefore, we know that high-risk keywords are easily disguised but the negative sentiment tone of the message is not changed that's why the correlation between sentiment and toxicity is very important.

### **2.1.2.1 Error Analysis**

Based on the analysis by Van Aken[11] et al, we study about the basic reasons for misclassification

- **Toxicity without offending words**

Toxicity can be passed on without referring bad words. The harmful significance can be uncovered with an understanding of the full sentence. For example “she resembles a buffalo” here “buffalo” is not offending in general. To comprehend the harmfulness of the remark, a model needs to comprehend that “she” alludes to an individual and “resemble a buffalo” is commonly viewed as insulting. Whenever coordinated to an individual insult is not uncovered by taking words individually.

- **Sarcasm**

Sarcasm is hard to detect because the purpose of the remark is different from its actual significance. Example: “I love going to work on holidays” normally this sentence is not toxic but it implies the inverse. “What a great online shopping site, can’t find anything useful”. “Great” is used but the whole meaning of the sentence is different and is a negative one. This type of problem is a part of sentiment analysis subjectivity.

- **Mislabeled comments**

Most comments in the dataset are labeled by human raters. There is no standard definition of toxicity that depends upon the context of comments. Human rater labels the comments according to their understanding so there are more chances of mislabeled comments.

- **Slag, abbreviation, and rare words**

Slangs, typo, abbreviations, and rare words are a challenge for toxic comment classification datasets. Slangs are not standard language term that are terms that generally used by people in their daily life. Rare words we understand such words that are new to the dictionary and not in the training dataset. Abbreviations are mostly used on twitter like platforms where limited number words are used for comments and post that’s why they are also called microblogging sites and the measure of slang is stage-specific constantly misclassify because of uncommon words is twice as high for tweets than for Wikipedia talk page dataset.

- **Comparison**

Comparative belief recognition is a field of study in Sentiment Analysis, for example, "Brand A telephone is better than Brand B telephone", this survey has no belief in the positive or negative sense. It just shows Brand A is better Brand B.

- **Spam Detection**

Identifying spam is likewise troublesome in light of the fact that occasionally delegates of that item copy such huge numbers of phony surveys or their rivals give negative audits then it turns out to be difficult to structure algorithms for such issues.

## 2.2 Background and Previous work

Toxic comments have a deep influence on a user's health online as well as offline. There have been several research papers on detecting toxic comments in online discussions. Most of the work is based on machine learning, text classification, sentiment analysis, and deep learning neural network.

Abusive comment classification work started with Yin et al. [5] paper in which they used Support Vector Machine (SVM) and Term Frequency-Inverse Document Frequency (TF-IDF) features and compared the performance with a simple TF-IDF model on a chat-style database. Nguyen [6] proposed a model for sentiment label distribution using a hybrid model of bidirectional Long Short Term Memory cell (LSTM) model with word-level embedding and Convolutional Neural Network (CNN) model with character embedding technique on Stanford Twitter sentiment corpus. This hybrid model achieved an accuracy of 86.63%.

Chu and Jue [7] compared Deep Learning models such as Recurrent Neural Network (RNN) models as LSTM with word embedding, CNN model with character embedding, and CNN model with word embedding. CNN, with the character embedding model, performed best between them with an accuracy of 94%. This paper also specified that character level embedding has improved performance than word-level embedding for CNN. In the real-life for the practical applications such as automatic comment moderation, CNN with word embedding was suggested.

Georgeakopoulos [3] proposed a Deep Learning approach using CNN for toxicity classification in the text classification and compare the performance with SVM, K-nearest neighbors (KNN), Naïve Bayes (NB) and Linear discriminated analysis (LDA). NB and KNN had the lowest precision and recall scores. It means that they classify some non-toxic comments to toxic comments and vice versa. CNN had the best precision and recall score. CNN also attained the best performance with an accuracy score of 92.7%.

Khieu and Narwal [8] used different Deep Learning models for toxic comment classification. They used SVM, LSTM, CNN, multilayer perceptron in combination with word and character-level embedding models for toxicity detection.

They evaluated their model on the Kaggle toxic comment classification challenge dataset. LSTM model achieved the best performance with an accuracy of 92.7% and an f1 score of 70.6%.

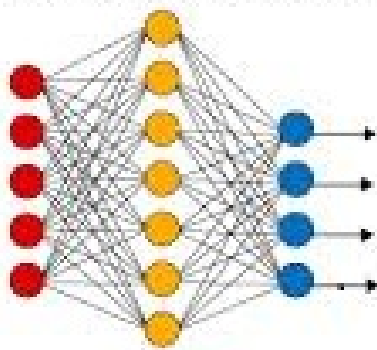
As far as the above models or approaches are concerned, we provide a model in this paper, combining both CNN and Bi-LSTM Deep Learning models with word embedding that increases the accuracy of toxic comments classification along with the F1 score. The next section contains the system architecture of the proposed classification model.

# CHAPTER 3: DEEP LEARNING MODELS

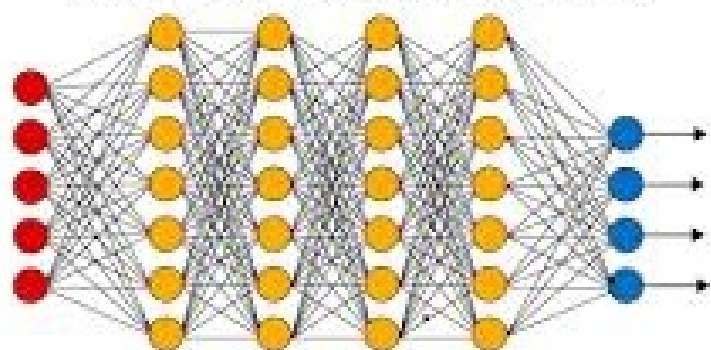
## 3.1 Deep Neural Network

Deep learning (DL) is a subfield of machine learning. Deep learning is enlivened by the human mind and how it sees data through the collaboration of neurons. "A standard Neural network (NN) comprises of numerous basic, associated processors called neurons". Input neurons get enacted through natural sensors; remaining neurons get activated through already activated neurons. The "Deep" likewise alludes to the many concealed layers of Artificial neural network (ANN) utilized in deep learning.

### Simple Neural Network



### Deep Learning Neural Network



● Input Layer      ● Hidden Layer      ● Output Layer

Fig 3.1: Simple Neural Network and Deep Learning Neural Network

Numerous specialists describe profound neural systems as systems that have an input layer, an output layer, and at least one hidden layer in the middle. Each layer performs explicit kinds of arranging and requesting in a procedure that some allude as "feature hierarchy". One of the key applications of these refined neural systems is managing unlabeled or unstructured information. DNNs are increasingly appropriate for mapping highlights into a progressively discrete space. A completely associated DNN included top of the LSTM arrange, can give better arrangement by mapping among yield and concealed loads by changing highlights into a yield space.

For DNN, a few NN layers are associated in input style to pass data to other layers and the associations don't frame a cycle. The input is legitimately taken care of output through a progression of the load. They are widely utilized in pattern recognition. NN are further classified in CNN, RNN, and LSTM.

### 3.2 Convolutional Neural Network

CNN is mainly used for the classification. It takes the input data and extracts the various features of input that help in differentiating different classes from each other. The Preprocessing require by CNN is less compared to other classification algorithms. CNN mainly has three layers: "The convolution layer", "the pooling layer", and "the fully connected layer".

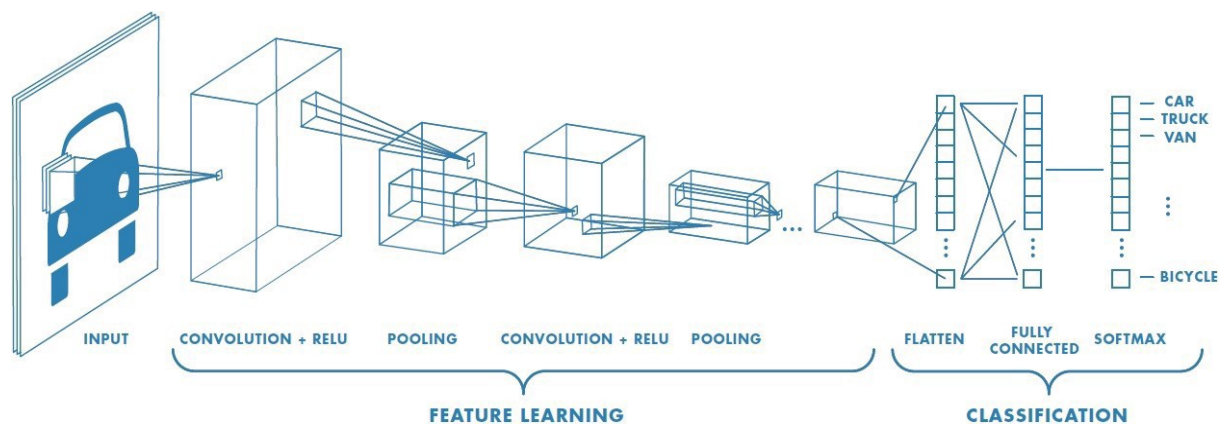


Fig 3.2 CNN full connected structure

The Convolution layer is the center structure of CNN. It conveys the principal part of the system's computational burden. The fundamental target of convolution is to extract features, for example, "color", "edge", and "corner" of the data. As we go further inside the system, the system begins distinguishing progressively complex features like "shape", "digit", "face" etc. At the end of the convolution layer, we have features matrix that will be feed into the next pooling layer.



The pooling layer exclusively eliminates the computation required to process the data. It is finished by diminishing the feature matrix significantly more. In this layer, we attempt to remove the predominant aspect from a limited measure of the neighborhood. At the end of the pooling layer, we have a matrix with only important features and feed into the fully connected layer. This layer will train the model with back-propagation over a series of epochs then finally get the classified result.

As per figure 2, Input is a collection of information regarding different types of vehicles. This dataset is passing through the CNN model for the classification. The first layer of CNN is the convolution layer that will extract the feature from the dataset with the help of the Relu activation function. The output of this layer is feed into the Pooling layer. This layer will extract important features that help in the classification in a fully connected layer. Convolution and pooling are applied over a series of epochs then extracted feature matrix is feed into a fully connected layer that will finally classify the dataset and give the results.

### **3.3 Gated Recurrent Unit**

RNN has limitations like short-term memory. If the data is sufficient long then it will be difficult to transfer data from prior time steps to later ones. If we try to process long text paragraphs so RNN may forget about significant data from the earliest starting point.

Another limitation of RNN is the "vanishing gradient problem". The gradient is mainly used for updating the weight in the neural network. The vanishing gradient is the point at which the inclination shrivels as it back spreads through time. If angle esteem turns out to be incredibly little, it doesn't contribute a lot of learning.

An LSTM has a comparative control stream as an intermittent neural system. It forms information passing on data as it proliferates onward. The distinctions are the tasks inside the LSTM's cells. These tasks are utilized to permit the LSTM to keep or overlook data. Presently taking a gander at these activities can get a touch of overpowering so we'll go over this bit by bit.

Presented by Cho, et al. in 2014, GRU (Gated Recurrent Unit) intends to take care of the vanishing gradient issue which accompanies a standard RNN. GRU can likewise be considered as a minor departure from the LSTM because both are planned comparably and, now and again, produce similarly incredible outcomes.

The GRU is the more up to date age of Recurrent Neural systems and is truly like the LSTM. GRU has freed of the cell state and utilized the shrouded state to move data. It additionally just has two gates, "a reset gate" and "update gate".

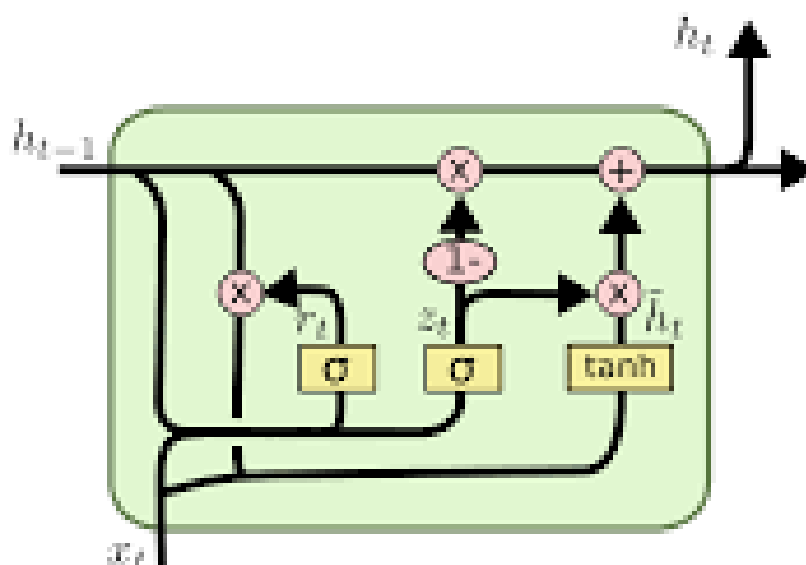


Fig 3.3: A GRU Cell

Update Gate (u): It decides the amount of the past information should be passed along into what's to come. It is undifferentiated from the Output Gate in the LSTM.

Reset Gate (r): It decides the amount of the past information to overlook. It is similar to the blend of the Input Gate and the Forget Gate in the LSTM.

The essential work-stream of a Gated Recurrent Unit Network is like that of a fundamental Recurrent Neural Network when represented, the primary distinction between the two is in the inner working inside each intermittent unit as Gated Recurrent Unit systems comprise of entryways which balance the current input and the past hidden state.

$$\tilde{c}_t = \tanh (W_c [G_r * c_{t-1}, x_t] + b_c) \quad (3.1)$$

$$G_u = \sigma (W_u [c_{t-1}, x_t] + b_u) \quad (3.2)$$

$$G_r = \sigma (W_r [c_{t-1}, x_t] + b_r) \quad (3.3)$$

$$C_t = G_u * c_{\tilde{t}} + (1 - G_u) * c_{t-1} \quad (3.4)$$

The GRU controls the progression of data like the LSTM unit, however without utilizing a memory unit. It just uncovered the full hidden content with no control. GRU is generally new, the exhibition is comparable to LSTM, however computationally progressively productive (less perplexing structure as called attention to). So we are seeing it being utilized to an ever-increasing extent.

## CHAPTER 4: METHODOLOGY

Our methodology for detecting toxic comments and classifying them according to the type of toxicity is divided into the following ten phases, namely dataset used, data pre-processing, embedding layer, convolution layer, max-pooling layer, Bi-GRU layer, global max-pooling layer, dropout layer, and two dense layers. Fig. 4 represents the proposed methodology.

### 4.1 Dataset

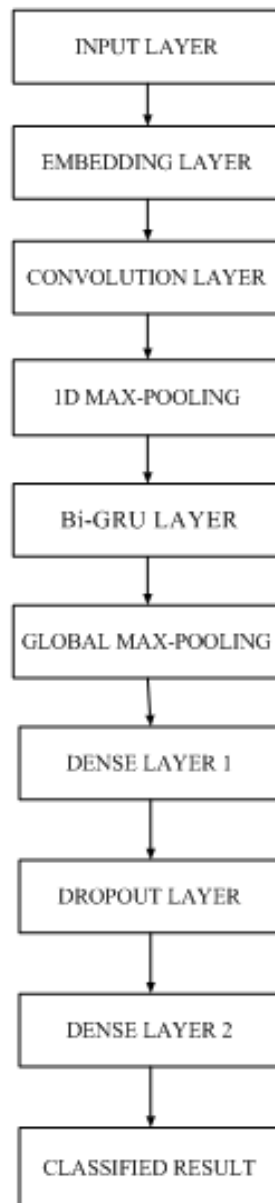
In this research paper, we will be using the dataset available from the Kaggle Competition [9] known as the “Toxic Comment Classification Challenge”. This dataset is a collection of comments from "Wikipedia's talk page edit". The dataset contains 159,571 comments that have been rated by humans for six sorts of toxicity labels such as "toxic, severe toxic, obscene, threat, insult, and identity hate".

### 4.2 Data Preprocessing

Dataset is a collection of real-world data; however, such data is generally inconsistent and incomplete that requires data preprocessing. Data preprocessing helps us to clean, format, and organize the raw data. To achieve the same, firstly, we will remove Stopwords from the dataset. “Stop words are common English words such as, the, am, there; which do not influence the semantic of the review and removing them can reduce noise” [10]. Secondly, Tokenization will be performed. "Tokenization is the process of splitting the input into meaningful pieces" [11]. These pieces are called tokens of words. At last, the padding sequence will be used to make each comment of the dataset into the same length.

### 4.3 Embedding Layer

In the third phase, the Embedding layer will be used for mapping the words of comment on a vector of real numbers. For each unique word, the corresponding vector will be assigned in the space. There are various methods for creating word embeddings such as Glove, Word2vec, and FastText.



**Fig 4.4: Proposed Hybrid Deep Learning Model for Toxic Comment Classification**

#### **4.4 Convolution Layer**

In the fourth phase, the embedding layer output will feed into the 1D convolution layer. The Convolution layer is the center structure of the Convolutional Neural Network (CNN). The primary target of convolution will be to extract features from the input and pass its outcomes to the next layer.

#### **4.5 Max-Pooling Layer**

In the fifth phase, the yield of convolution will be transferred to the 1D Max pooling layers. The pooling layer will be used for reducing the dimension of processed data and only keeps important information. This layer will reduce the computation cost of the network. The pooling layer will diminish the features that decline the likelihood of overfitting.

#### **4.6 Bidirectional-GRU Layer**

The yield of Max pooling will be transferred into the bidirectional GRU layer. Bi-GRU is a kind of bidirectional recurrent neural network. It is almost similar to the bi-LSTM model. "GRU is faster than LSTM because it requires less calculation to refresh its concealed state" [12] [13]. GRU also overcomes the problem of vanishing gradient.

#### **4.7 Global Max-Pooling Layer**

In this phase, we will be using the 1D global max-pooling layer. The global constraint will yield the absolute most significant feature of the feature map rather than a feature window. The global max-pooling layer will reduce the dimension of input to one.

#### **4.8 Dense Layer 1**

In this phase, the dense layer will utilize the Relu (Rectified linear unit) activation function. A dense is only a normal layer of neurons in a neural system. This dense layer will receive the output from all the neurons of previous layers and help in refining the flow of gradient.

#### **4.9 Dropout Layer**

In this phase, we will use the dropout layer to dodge the issue of overfitting in the system. This layer will remove extra neurons from the neural network during the training phase and reduce the complexity of the model.

#### **4.10 Dense Layer 2**

Lastly, in the tenth phase, the last dense layer will utilize the sigmoid activation function for multi-label classification. The number of neurons in the last layer will be the number of classes in our dataset.

# CHAPTER 5: IMPLEMENTATION

The following sub-sections elaborate phase-wise implementation details of the proposed methodology. We used the Python programming language for the implementation of our model. TensorFlow and Keras libraries were used for building the neural network. To provide GPU support, we implemented our model on the Kaggle platform. After data preprocessing, we started building our model. We set up our input layer. As mentioned in the Keras documentation, we have to include the shape for the very first layer, and then Keras will automatically derive the shape for the rest of the layers.

## 5.1 Dataset

Exploratory Data Analysis (EDA) was performed on the dataset that gives us important information regarding the dataset and provides a way to handle the data for the model. Dataset was split into preparation and validation set into the 90:10 ratios. Table 1 defines the distribution of labels in the training set. Fig. 5 shows sample instance of the dataset schema and database schema was represented as <id, comment, toxic, severe toxic, obscene, threat, insult, identity hate> that helps in understanding the dataset for further use in the model. Fig.6 describes the training set. Fig. 7 shows the distribution of the number of comments vs. the length of comments.

|   | id               | comment_text                                      | toxic | severe_toxic | obscene | threat | insult | identity_hate |
|---|------------------|---|-------|--------------|---------|--------|--------|---------------|
| 0 | 0000997932d777bf | Explanation\nWhy the edits made under my usern... | 0     | 0            | 0       | 0      | 0      | 0             |
| 1 | 000103f0d9cfb60f | D'aww! He matches this background colour I'm s... | 0     | 0            | 0       | 0      | 0      | 0             |
| 2 | 000113f07ec002fd | Hey man, I'm really not trying to edit war. It... | 0     | 0            | 0       | 0      | 0      | 0             |
| 3 | 0001b41b1c6bb37e | "\nMore\nI can't make any real suggestions on ... | 0     | 0            | 0       | 0      | 0      | 0             |
| 4 | 0001d958c54c6e35 | You, sir, are my hero. Any chance you remember... | 0     | 0            | 0       | 0      | 0      | 0             |

Fig 5.5: Sample instance of the dataset

## 5.2 Data Preprocessing

Data preprocessing was used to increase the quality of the dataset and make it ready for model implementation. Firstly, we removed stopwords using Python built-in dictionary of stopwords `Nltk.corpus`. Secondly, we performed Tokenization using `keras.preprocessing.text()` function. At last, we performed the padding sequence using `keras.preprocessing.sequence()` function. Padding sequence in Deep Learning was used to make each comment of the dataset into the same length. We assume the maximum length of a comment to be 500 and then add padding sequences at the end of shorter comments to make their length equal to 500. Beautiful soup Python library was utilized for hauling information out of HTML and XML records that exist in the dataset. This phase was mainly used for converting the dataset into the standard form for further implementation.

**Table 1 defines the distribution of labels in the training set.**

| Comment Label   | Number of comments |
|-----------------|--------------------|
| Toxic           | 15294              |
| Severe _toxic   | 1595               |
| Obscene         | 8449               |
| Threat          | 478                |
| Insult          | 7877               |
| Identity _ hate | 1405               |

## 5.3 Embedding Layer

In our model, embedding will be made dependent on the tokenized text. TF-IDF tokenization will make a lattice of features which would be utilized to develop the embedding. Each comment of the dataset was changed over to a one-dimensional vector of numerical components paying little mind to the tokenization strategy. To handle the comment of variable length padding was used and the vector was filled with zeros at the end so that all the comments have equal length. We assume the size of the embedding word vector to be 240d.



## 5.4 Convolution Layer

In the 1D convolution layer, we used 100 filters with length 4. By 1D convolution, we understand that “the kernel used here for convolution was a one-dimensional vector ” [13]. Adding CNN on the top of the GRU helps in the sense that CNN combines with the polling layer brings out the important temporal features devoid of any noise which bidirectional GRU can use more effectively. The yield of this layer is conveyed to the 1D Max pooling layers.

## 5.5 Max-Pooling layer

Various types of pooling techniques could use, e.g., 1D max pooling, global, max, average, and sum that depends on the architecture of the model.

|       | toxic         | severe_toxic  | obscene       | threat        | insult        | identity_hate |
|-------|---------------|---------------|---------------|---------------|---------------|---------------|
| count | 159571.000000 | 159571.000000 | 159571.000000 | 159571.000000 | 159571.000000 | 159571.000000 |
| mean  | 0.095844      | 0.009996      | 0.052948      | 0.002996      | 0.049364      | 0.008805      |
| std   | 0.294379      | 0.099477      | 0.223931      | 0.054650      | 0.216627      | 0.093420      |
| min   | 0.000000      | 0.000000      | 0.000000      | 0.000000      | 0.000000      | 0.000000      |
| 25%   | 0.000000      | 0.000000      | 0.000000      | 0.000000      | 0.000000      | 0.000000      |
| 50%   | 0.000000      | 0.000000      | 0.000000      | 0.000000      | 0.000000      | 0.000000      |
| 75%   | 0.000000      | 0.000000      | 0.000000      | 0.000000      | 0.000000      | 0.000000      |
| max   | 1.000000      | 1.000000      | 1.000000      | 1.000000      | 1.000000      | 1.000000      |

**Fig 5.6: Description of train dataset**

We passed the output of convolution to the 1D max-pooling layer that applied the max pool operation on the window of every four characters. As the output, we get a matrix of size = number of sentences \*125\*100, which was also called Extracted Features. These extracted features were then transferred into the bidirectional GRU layer.

## 5.6 Bidirectional-GRU Layer

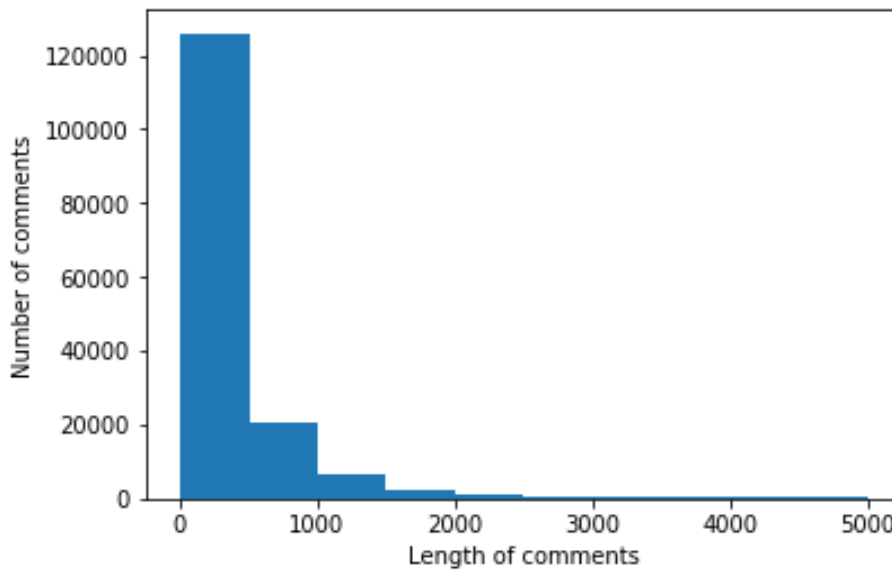
Bidirectional GRU has only two gates, the reset and update gate. "The reset gate( $r$ ) was utilized to choose how much past state data was required to keep and to overlook" [12].

$$r^{(t)} = \sigma(W^{(r)} x^{(t)} + U^{(r)} h^{(t-1)}) \text{ (Reset gate)} \quad (5.1)$$

The update gate ( $z$ ) worked similarly to the forget gate and input gate of the LSTM model. "The update signal  $z^{(t)}$  is responsible for determining how much of the hidden state should be carried forward to the next state" [12].

$$z^{(t)} = \sigma(W^{(z)} x^{(t)} + U^{(z)} h^{(t-1)}) \text{ (Update gate)} \quad (5.2)$$

Bi-GRU had two units of the recurrent network, one unit to move the data in a forward way, and the second unit moved the data in a backward way with the help of reset and update gate.



**Fig.5.7.** Distribution of Comment length

## **5.7 Global Max-Pooling Layer**

The next global max-pooling layer reduced the dimension of input to one. For example, if we had data [2,3,4,5,6,6,7] with pool length 3 yield was 4,5,6,6,7 respectively; however, if 1D global max-pooling was used, then yield equal to 7. We performed 1D global max-pooling using `Keras.layers()` function.

## **5.8 Dense Layer 1**

This layer produced the yield of dimension 50. We performed this dense layer using `Keras.layers()` function and utilized the Relu (Rectified linear unit) activation function.

## **5.9 Dropout layer**

The yield of the dense layer passed to the Dropout layer, which impaired a few neurons in the following layer so that the entire system could conclude better. The dropout rate was set at 20% and performed using `Keras.layers()` function.

## **5.10 Dense layer 2**

The last dense layer utilized the sigmoid activation function for multi-label classification using `Keras.layers()` function that created six-dimensional vectors, defined as the labels of toxicity. The final output file contained the probability of labels occurring on the dataset.

We used binary cross-entropy loss function because this function is more effective on classification tasks compared to other loss functions. Adam optimizer is designed to improve the classic Stochastic Gradient Descent (SGD) optimizer. This model minimized the log-loss function using the SGD optimization algorithm. The model was trained using batch size 32 and run for 20 epochs. A 10% size validation dataset was likewise passed on along.

# Chapter 6: TECHNOLOGICAL STACK

## 6.1 Functional requirements

These are the functions that the framework must convey to meet the client's prerequisites.

The functional prerequisites of this framework include:

1. The system will allow an effective and reliable way of gathering user opinion to formulate a user-dependent model.
2. The system should take care of all the dependencies required and have the required software installed.
3. User(s) should have a fair knowledge of the computer system in order to effectively make use of the system.

## 6.2 Software requirements

Following are the software requirements, modules, and components that would be needed to develop the proposed application:

- Any Unix/Linux/windows based Operating System, with command-line functionality.
- Kaggle Notebook should be installed.
- Any standard Text Editor software such as Sublime Text, Visual studio code, ATOM to properly view the code.
- A web browser so that the Kaggle notebook can be launched.
- Python2 and Python3 must be installed.

## 6.3 Pre-Requisite python libraries to be installed

- NLTK
- Pandas
- Keras
- Numpy
- Tensorflow
- BeautifulSoup

# CHAPTER 7: RESULT AND ANALYSIS

## 7.1 Evaluation Metrics

A confusion matrix is utilized to calculate the performance of the characterized model. The confusion matrix is also called the error matrix. We report the result of our model using standard Precision, Recall, Accuracy, and F1 measure [14].

"Precision is defined as the ratio of correctly predicted positive observations of the total predicted positive observations".

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

"The recall is defined as the proportion of correctly identified positives and all known as Sensitivity".

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

"F1-score is defined as the harmonic mean of precision and recall score".

$$\text{F1-score} = 2 (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$

"Accuracy is defined as the ratio of exactly matched instances to total instances".

$$\text{Accuracy} = \text{TP} + \text{TN} / (\text{TP} + \text{FP} + \text{FN} + \text{TN})$$

Where TP= True Positive

FP= False Positive

TN= True Negative

FN= False Negative

## 7.2 Result

We partitioned the dataset into training and testing sets. The training set contains 143613 comments and the testing set contains 15958 comments. After training and testing, we got the best accuracy of 98.39%, a precision score of 86.05%, and a recall score of 74.59%. The F1 score turns out to be 79.91%.

| Layer (type)                                | Output Shape     | Param # |
|---|------------------|---------|
| input_1 (InputLayer)                        | (None, 500)      | 0       |
| embedding_1 (Embedding)                     | (None, 500, 240) | 507600  |
| conv1d_1 (Conv1D)                           | (None, 500, 100) | 96100   |
| max_pooling1d_1 (MaxPooling1D)              | (None, 125, 100) | 0       |
| bidirectional_1 (Bidirectional)             | (None, 125, 120) | 57960   |
| global_max_pooling1d_1 (GlobalMaxPooling1D) | (None, 120)      | 0       |
| dense_1 (Dense)                             | (None, 50)       | 6050    |
| dropout_1 (Dropout)                         | (None, 50)       | 0       |
| dense_2 (Dense)                             | (None, 6)        | 306     |
| Total params: 668,016                       |                  |         |
| Trainable params: 668,016                   |                  |         |
| Non-trainable params: 0                     |                  |         |

Figure 7.8: Model Summary of the hybrid model

```
Train on 143613 samples, validate on 15958 samples
Epoch 1/20
- 1368s - loss: 0.0836 - acc: 0.9757 - precision: 0.8666 - recall: 0.4411 - val_loss: 0.0590 -
val_acc: 0.9810 - val_precision: 0.9204 - val_recall: 0.5784
Epoch 2/20
- 1381s - loss: 0.0595 - acc: 0.9803 - precision: 0.8748 - recall: 0.6086 - val_loss: 0.0520 -
val_acc: 0.9822 - val_precision: 0.8300 - val_recall: 0.7257
Epoch 3/20
- 1394s - loss: 0.0544 - acc: 0.9812 - precision: 0.8750 - recall: 0.6525 - val_loss: 0.0486 -
val_acc: 0.9828 - val_precision: 0.8515 - val_recall: 0.7088
Epoch 4/20
- 1398s - loss: 0.0515 - acc: 0.9818 - precision: 0.8774 - recall: 0.6708 - val_loss: 0.0475 -
val_acc: 0.9830 - val_precision: 0.8637 - val_recall: 0.7020
Epoch 5/20
- 1372s - loss: 0.0499 - acc: 0.9821 - precision: 0.8738 - recall: 0.6873 - val_loss: 0.0472 -
val_acc: 0.9829 - val_precision: 0.8862 - val_recall: 0.6838
Epoch 6/20
- 1379s - loss: 0.0485 - acc: 0.9824 - precision: 0.8751 - recall: 0.6926 - val_loss: 0.0467 -
val_acc: 0.9830 - val_precision: 0.8724 - val_recall: 0.6885
Epoch 7/20
- 1386s - loss: 0.0478 - acc: 0.9828 - precision: 0.8760 - recall: 0.7015 - val_loss: 0.0451 -
val_acc: 0.9837 - val_precision: 0.8801 - val_recall: 0.7041
Epoch 8/20
- 1371s - loss: 0.0470 - acc: 0.9830 - precision: 0.8767 - recall: 0.7062 - val_loss: 0.0454 -
val_acc: 0.9836 - val_precision: 0.8276 - val_recall: 0.7493
Epoch 9/20
- 1398s - loss: 0.0463 - acc: 0.9830 - precision: 0.8801 - recall: 0.7022 - val_loss: 0.0447 -
val_acc: 0.9838 - val_precision: 0.8746 - val_recall: 0.7162
Epoch 10/20
- 1380s - loss: 0.0460 - acc: 0.9832 - precision: 0.8830 - recall: 0.7094 - val_loss: 0.0455 -
val_acc: 0.9835 - val_precision: 0.8694 - val_recall: 0.7149
```

Figure7.9: Result of first 10 epochs

```
Epoch 11/20
- 1395s - loss: 0.0456 - acc: 0.9833 - precision: 0.8783 - recall: 0.7136 - val_loss:
0.0451 - val_acc: 0.9837 - val_precision: 0.8617 - val_recall: 0.7196
Epoch 12/20
- 1385s - loss: 0.0453 - acc: 0.9833 - precision: 0.8773 - recall: 0.7100 - val_loss:
0.0454 - val_acc: 0.9834 - val_precision: 0.8418 - val_recall: 0.7372
Epoch 13/20
- 1398s - loss: 0.0450 - acc: 0.9835 - precision: 0.8772 - recall: 0.7173 - val_loss:
0.0448 - val_acc: 0.9836 - val_precision: 0.8366 - val_recall: 0.7439
Epoch 14/20
- 1397s - loss: 0.0451 - acc: 0.9833 - precision: 0.8795 - recall: 0.7167 - val_loss:
0.0445 - val_acc: 0.9839 - val_precision: 0.8508 - val_recall: 0.7514
Epoch 15/20
- 1418s - loss: 0.0447 - acc: 0.9835 - precision: 0.8809 - recall: 0.7231 - val_loss:
0.0454 - val_acc: 0.9834 - val_precision: 0.8286 - val_recall: 0.7514
Epoch 16/20
- 1431s - loss: 0.0445 - acc: 0.9835 - precision: 0.8762 - recall: 0.7219 - val_loss:
0.0444 - val_acc: 0.9838 - val_precision: 0.8622 - val_recall: 0.7189
Epoch 17/20
- 1409s - loss: 0.0445 - acc: 0.9835 - precision: 0.8754 - recall: 0.7214 - val_loss:
0.0439 - val_acc: 0.9838 - val_precision: 0.8633 - val_recall: 0.7209
Epoch 18/20
- 1397s - loss: 0.0446 - acc: 0.9835 - precision: 0.8795 - recall: 0.7234 - val_loss:
0.0458 - val_acc: 0.9834 - val_precision: 0.8605 - val_recall: 0.7169
Epoch 19/20
- 1404s - loss: 0.0442 - acc: 0.9837 - precision: 0.8777 - recall: 0.7243 - val_loss:
0.0466 - val_acc: 0.9836 - val_precision: 0.8799 - val_recall: 0.7027
Epoch 20/20
- 1412s - loss: 0.0444 - acc: 0.9835 - precision: 0.8765 - recall: 0.7198 - val_loss:
0.0444 - val_acc: 0.9835 - val_precision: 0.8604 - val_recall: 0.7162

<keras.callbacks.History at 0x7f3fdd0fa4a8>
```

Figure 7.10: Result of last 10 epochs



## CHAPTER 8: CONCLUSION AND FUTURE SCOPE

There have been ceaseless trials of experiments to detect the presence of toxic comments of various kinds on online platforms. This holds importance in the research field due to the tremendously growing online interactive communication among users. Toxic comment classification is used for detecting the toxicity in social media platforms. Our work is dedicated to finding the best possible solution for toxic comment classification. In this paper, we have implemented a Deep Learning based model using convolution and bidirectional gated recurrent units that successfully performs the multi-label classification of different sorts of toxic comments. As an outcome, we achieved the best results with an accuracy of 98.39%, a precision score of 86.05%, and a recall score of 74.59%, and the computed F1 score of 79.91%.

“Common challenges for toxic comment classification among different datasets contain out-of-jargon words long-range dependencies, and multi-word phrases” [15]. The limitation of this model is that it is trained on Google Jigsaw’s toxic comment dataset and we achieved high accuracy on our test set. However, our model may not be able to achieve the same level of performance on other datasets like twitter dataset. Another limitation is for handling the out of vocabulary words and our dataset is mostly a collection of English language comments, so our model works on the English language only. Google Jigsaw’s toxic comment dataset is a collection of comments from "Wikipedia's talk page" and comments have been labeled by human raters [3] [7] because there is no standard definition of toxic labels, human raters rate the comments on their personal beliefs and therefore this dataset is skewed.

For future work, we would like to experiment with our model with the pre-trained word embedding techniques like Glove, Word2vec, and FastText, trained on toxic comment dataset. Many enhancements may be possible to our model by adding consideration based instruments for better detection of toxic comments. Different users use the different number of online platforms for discussions, so developing different models for each platform is not an efficient way to handle this problem; therefore, we need to build a solitary framework that works over various platforms. All such work can be done in the future.

# APPENDIX A

## PROGRAM CODE

```
import sys, os, re, csv, codecs, numpy as np, pandas as pd

import matplotlib.pyplot as plt

!pip install keras-metrics

import keras

import keras_metrics

%matplotlib inline

from bs4 import BeautifulSoup

from nltk.corpus import stopwords # Import the stop word list

from keras.preprocessing.text import Tokenizer

from keras.preprocessing.sequence import pad_sequences

from keras.layers import Dense, Input, LSTM, Embedding, Dropout, Activation,
GRU, Conv1D, MaxPooling1D

from keras.layers import Bidirectional, GlobalMaxPool1D, Bidirectional

from keras.models import Model

from keras import initializers, regularizers, constraints, optimizers, layers

from keras.callbacks import EarlyStopping, ModelCheckpoint
```

```

import gc

train = pd.read_csv('../input/jigsaw-toxic-comment-classification-
challenge/train.csv.zip')

test = pd.read_csv('../input/jigsaw-toxic-comment-classification-
challenge/train.csv.zip')

submit_template = pd.read_csv('../input/jigsaw-toxic-comment-classification-
challenge/sample_submission.csv.zip',header = 0)

train.head()

train.describe()

list_sentences = train["comment_text"]

list_sentences_test = test["comment_text"]

max_features = 20000

tokenizer = Tokenizer(num_words=max_features,char_level=True)

tokenizer.fit_on_texts(list(list_sentences))

list_tokenized = tokenizer.texts_to_sequences(list_sentences)

list_tokenized_test = tokenizer.texts_to_sequences(list_sentences_test)

maxlen = 500

X_t = pad_sequences(list_tokenized, maxlen=maxlen)

X_te = pad_sequences(list_tokenized_test, maxlen=maxlen)

```

```

totalNumWords = [len(one_comment) for one_comment in list_tokenized]

plt.hist(totalNumWords)

plt.xlabel('Length of comments')

plt.ylabel('Number of comments')

plt.show()

inp = Input(shape=(maxlen, ))inp

embed_size = 240

x = Embedding(len(tokenizer.word_index)+1, embed_size)(inp)

x = Conv1D(filters=100,kernel_size=4,padding='same', activation='relu')(x)

x=MaxPooling1D(pool_size=4)(x)

x = Bidirectional(GRU(60, return_sequences=True,name='lstm_layer',dropout
=0.2,recurrent_dropout=0.2))(x)

x = GlobalMaxPool1D()(x)

x = Dense(50, activation="relu")(x)

x = Dropout(0.2)(x)

x = Dense(6, activation="sigmoid")(x)

model = Model(inputs=inp, outputs=x)

model.compile(loss='binary_crossentropy',

optimizer='adam',

```

```
model.summary()

X_train, X_test, y_train, y_test = train_test_split(X_t, train[["toxic", "severe_toxic", "obscene", "threat", "insult", "identity_hate"]], test_size = 0.10, random_state = 42)

batch_size = 32

epochs = 8

model.fit(X_train,y_train, batch_size=batch_size, epochs=epochs,validation_data=(X_test,y_test),verbose=2)

y_submit = model.predict(X_te,batch_size=batch_size,verbose=1)

y_submit[np.isnan(y_submit)]=0

sample_submission=pd.DataFrame(y_submit,columns=["toxic", "severe_toxic", "obscene", "threat", "insult", "identity_hate"])

sample_submission.to_csv('submission.csv', index=False)
```

## REFERENCES

1. Duggan, M.: Online harassment (2017).
2. Risch, J., & Krestel, R.: Toxic Comment Detection in Online Discussions. In *Deep Learning-Based Approaches for Sentiment Analysis*, pp. 85-109. Springer, Singapore (2020).
3. Georgakopoulos, S. V., Tasoulis, S. K., Vrahatis, A. G., & Plagianakos, V. P.: Convolutional neural networks for toxic comment classification. In *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, pp. 1-6. (2018).
4. Perspective API, <https://perspectiveapi.com//>
5. Yin, D., Xue, Z., Hong, L., Davison, B. D., Kontostathis, A., & Edwards, L.: Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2, pp.1-7. (2009).
6. Nguyen, H., & Nguyen, M. L.: A deep neural architecture for sentence-level sentiment classification in twitter social networking. In *International Conference of the Pacific Association for Computational Linguistics* pp. 15-27. Springer, Singapore (2017).
7. Chu, T., Jue, K., & Wang, M.: Comment abuse classification with deep learning. *Von <https://web.stanford.edu/class/cs224n/reports/2762092.pdf> abgerufen* (2016).
8. Khieu, K., & Narwal, N. Detecting and classifying toxic comments. *Web: <https://web.stanford.edu/class/archive/cs/cs224n/cs224n,1184>*.
9. Toxic Comment Classification Challenge | Kaggle, <https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data>.
10. Maalej, W., & Nabil, H.: Bug report, feature request, or simply praise? on automatically classifying app reviews. *IEEE 23rd international requirements engineering conference (RE)*, pp. 116-125. (2015).

11. Mullen, L. A., Benoit, K., Keyes, O., Selivanov, D., & Arnold, J.: Fast, consistent tokenization of natural language text. *Journal of Open Source Software*, 3(23), 655 (2018).
12. Dey, R., & Salemt, F. M.: Gate-variants of gated recurrent unit (GRU) neural networks. *IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, pp. 1597-1600. (2017).
13. Saeed, H. H., Shahzad, K., & Kamiran, F.: Overlapping toxic sentiment classification using deep neural architectures. *IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 1361-1366. (2018).
14. "Precision and recall", *En.wikipedia*. [https://en.wikipedia.org/wiki/Precision\\_and\\_recall](https://en.wikipedia.org/wiki/Precision_and_recall)
15. Van Aken, B., Risch, J., Krestel, R., & Löser, A.: Challenges for toxic comment classification: An in-depth error analysis. *arXiv preprint arXiv:1809.07572* (2018).

## LIST OF PUBLICATIONS

1. R. Beniwal, A. Maurya. “Toxic Comment Classification using Hybrid Deep Learning”  
2020 International Conference on Sustainable Communication Networks and Application  
(ICSCN 2020), Tamil Nadu, India, 2020. Springer. [**Accepted**]

2. R. Beniwal, and A. Maurya. “Toxic Comment Classification: A Perspective on its Past,  
Present, and Future” 2020. [**Under Progress**]