Toxic Comment Classification In Discord

Zaed Magzoub
EEMCS
University of Twente
Enschede, The Netherlands
z.magzoub@student.utwente.nl

***Abstract*—In March 2020, the COVID-19 pandemic forced educational institutions to close their campuses and move all educational activities to online education. In addition, social media platforms were the leading platforms for communication between people. While it was easier to shift to online lectures, one aspect that took more work to deal with was student-student communication. In offline education, students interact in a real-life environment where students can ask questions and receive feedback more efficiently. In order to achieve better online education, educational institutions have decided to use the platforms the students are familiar with, Discord being one of these popular platforms. However, sharing toxic comments is hard to control in Discord since there is no feature in Discord to detect toxicity in shared messages. In this paper, various machine learning models were trained to classify toxic messages shared on University's discord servers. Machine learning-based solutions were used because they can achieve better results than traditional rule-based approaches.**

***Index Terms*—CNN, LSTM, FastText, SVM, TF-IDF, Word Embeddings, Deep Learning**

I. Introduction

After March 2020, people had a massive fear of the COVID-19 pandemic and feared being infected when they came into contact with someone with COVID-19. In addition, lockdowns and restrictions pushed people to use social media more because social media platforms were the best place for people to interact and be entertained without having any infection risk. July 2020 saw a rise of 10.5% in social media usage, compared to July 2019, according to a GlobalWebIndex survey[1]. Furthermore, the pandemic affected education, and to minimize the losses of students, educational institutions had to shift from offline education to online education.

In order to achieve better online education, many universities have used common communication platforms among students. Such a famous platform is Discord. The issue with Discord is that it is difficult to control what the students share on it, a deluge of contributions is added in many channels, and it is hard for the teachers and teaching assistants to control what is being shared. Some students share comments that contain sarcasm or toxicity. However, sharing toxic comments on the Discord server intended for educational purposes is inconvenient.

The drawback of Discord is that it has no built-in feature to control what is being shared in text channels. However, developers can build bots and add them to servers. Discord bots are AI-driven tools that can help automate tasks on Discord servers. By integrating a trained model with a Discord bot, it will be able to classify shared messages on text channels. The bot's main functionality is to hide messages with potential toxic probability and add them to a queue. Afterward, the teacher can review and accept/reject the messages.

Three different approaches were used to classify toxic comments. First is Convolution Neural Network (CNN) with FastText word embeddings. Second, Long Short Term Memory (LSTM) with FastText word embeddings. Lastly, Support Vector Machine (SVM) with TF-IDF embeddings. There are two types of classification used in the research. The first type is multi-class classification, where the goal is to classify a comment into different categories. The second type of classification is binary, meaning that the comment will be classified as toxic or non-toxic.

The dataset used is Jigsaw's data set hosted on Kaggle for the Toxic Comment Classification Challenge competition[2]. Jigsaw's dataset contains many labeled comments by humans for toxic behaviors. The types of toxicity are toxic, severe_toxic, obscene, threat, insult, and identity hate. The models were ultimately tested and compared using multiple evaluation metrics to select the best one for classification.

II. Research questions

The research problem has led to the following research questions:
1- What are the most effective machine learning methods for classifying toxic comments in discord-based education environments?
2- How will the performance of the models differ when testing them on educational data?
3- How to integrate a machine learning model in the Discord bot?

[1] GlobalWebIndex

[2] Kaggle Toxic Comment Classification Dataset

III. Related work

Natural language processing (NLP) has been an attractive research field for many years due to the spreading of the world wide web and digital libraries [12]. Therefore, much research has been found related to toxic comment classification. There are multiple machine learning methods used to classify toxic comments. Most of the research has used more than one approach to classify toxic comments and compare the performances in the research results. Most papers used deep neural networks, and the rest used more straightforward traditional methods. Table I below shows the methods used in the selected papers.

Approach	Paper
Convolutional neural network (CNN)	[2],[4],[6] ,[7],[10]
Logistic regression classifier	[3],[9],[10]
Long Short Term Memory (LSTM)	[1],[2],[10]
Bidirectional long short-term memory (BiLSTM)	[4],[11]
Bidirectional Gated Recurrent Unit Networks (Bidirectional GRU)	[4],[8]
Recurrent Neural Network (RNN)	[4],[5],[6]
Bi-GRU-LSTM	[8]
Bidirectional Encoder Representations from Transformers (BERT)	[8]
Random Forest	[9]
Support Vector Machine (SVM)	[9]
Naive Bayes	[9]
Decision tree	[9]
KNN classification	[9]

TABLE I: Approaches used in previous research

The most used dataset is Jigsaw's data set hosted on Kaggle for the Toxic Comment Classification Challenge competition. Jigsaw's dataset contains many labeled comments by humans for toxic behaviors. The dataset includes around 150K comments for training, and for testing the model, around 60K comments. The types of toxicity are toxic, severe toxic, obscene, threat, insult, and identity hate.

Much research has been done on toxic comment classification using different approaches. However, applying these approaches in Discord-based educational environments has lacked. This research used multiple machine learning approaches used in previous research using different implementations to achieve higher performance and integrate the best approach into Discord by building a bot that will automate detecting toxic messages.

IV. Data analysis

A. Dataset

The dataset used for training and testing the models was created by the Conversation AI team, a research group founded by Jigsaw and Google. Their objective is to help improve online conversation. The comments presented in the dataset were collected from Wikipedia's talk page edits. The dataset is used for Toxic Comment Classification Challenge launched by Kaggle[3].

In the Toxic Comment Classification Challenge dataset, there are 159571 samples for training and 63978 samples for testing. Each sample has a text comment labeled by human raters. Each comment has one or more labels from six different types of toxicity. In more detail, the six labels are toxic, severe toxic, obscene, threat, insult, and identity hate. If a comment contains any of the six types, it has a label of 1 for one or more types and 0 otherwise.

B. Classes Of Toxicity

Toxicity can have many different types. In the adopted dataset, there are six types of toxicity presented. Each of the six types represents a different toxicity level; based on that, the comment is classified. In the following, definitions are given according to Perspective API (a product of a collaborative research effort by Jigsaw and Google's Counter Abuse Technology team.)[14]

a) ***Toxic****:* is a "rude, disrespectful, or unreasonable comment that is likely to make people leave a discussion."

b) ***Serve Toxic****:* is a "very hateful, aggressive, disrespectful comment or otherwise very likely to make a user leave a discussion or give up on sharing their perspective."

c) ***Obscene****:* "Obscene or vulgar language such as cursing"

d) ***Threat****:* "describes an intention to inflict pain, injury, or violence against an individual or group."

e) ***Insult****:* is an "insulting, inflammatory, or negative comment towards a person or a group of people."

f) ***Identity Hate****:* are "negative or hateful comments targeting someone because of their identity."

C. Classification Types

a) ***Multi-class classification****:* uses all the six labels mentioned in the dataset to train the models. The model will predict a probability for each of the six classes to show whether the comment has potential toxicity for each class. Figure 1 shows how many comments are labeled with 1 for each class.
b) ***Binary classification****:* a new label is created called "toxic". The value of the label will be 0 or 1. The label's value will be 1 if the comment is from any of the six classes and 0 otherwise. 2 shows how many comments are toxic or clean.

[3]Kaggle is a web platform known for hosting competitions based on machine learning tasks

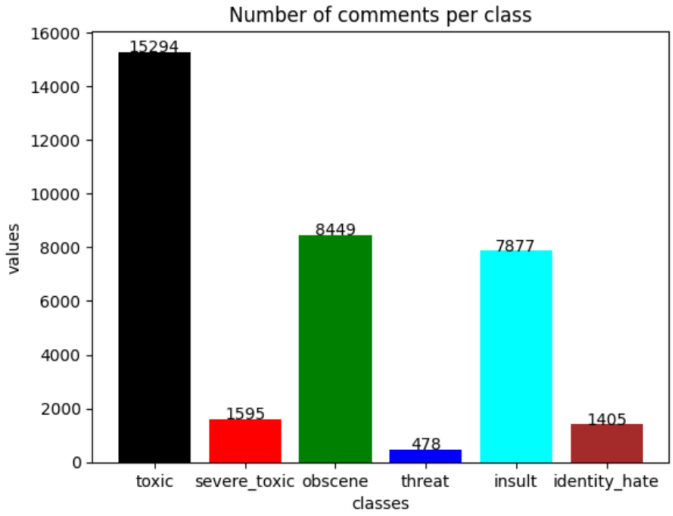

Fig. 1: Number of comments per class

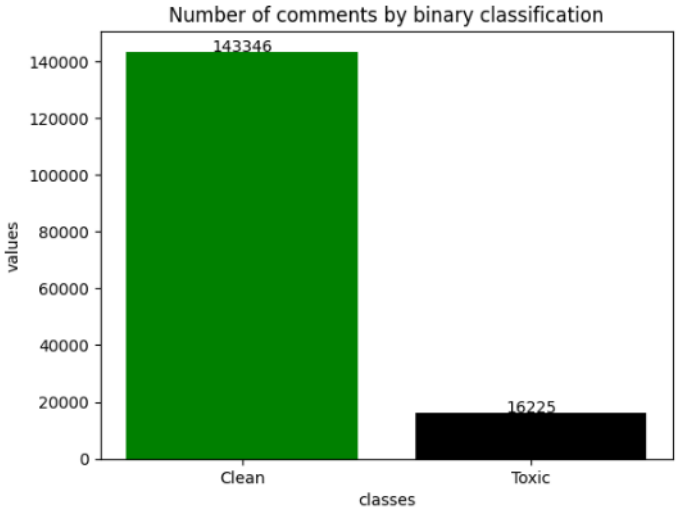

Fig. 2: Number of clean and toxic comments

D. *Data Preprocessing*

Data pre-processing plays a crucial role in NLP classification problems. The reason for that is that data pre-processing has an essential effect on the performance of the models [4]. In pre-processing, the intention is to clean the data and convert it into a valuable and efficient format. In addition, computers do their operations and understand only numbers, so data should be represented in vectors instead of text. Scientists commonly use pre-processing techniques that should not be applied in toxic classification, such as stemming or stopword removal, because that can lead to the loss of beneficial information that can help the model perform better [8].

Before starting pre-processing, the data should be understood first to know how to pre-process it. After doing some data analysis, it appeared that the data contained non-English words, emojis, numbers, extra spaces, HTTP/HTTPS links, and decorated English words, both upper/lower letters. According to the observations, the pre-processing procedure has been carried out:

1) **Converting emojis to text:** removing emojis is not good because emojis have meaning. Therefore, emojis are replaced with the corresponding words. For example, ☺ will be replaced by ”slightly smiling face”.

2) **Lower casing:** converting all upper case letters to lower case. For example, ”HeLLo” will be converted to ”hello”.

3) **Correction of toxic words:** if a comment contains a toxic word and some letters of the word were replaced by a * or #, then it will be converted back to its original word. For example, ”id*ot” will be converted to ”idiot”

4) **Decontraction of word:** A contraction is a word made smaller by combining two words. Example for that is ”I will” will be ”I’ll”. In decontraction we mean the opposite so ”you’re” will be ”you are”.

5) **Removing http/https links:** if a comment contain an URL link then it will be removed.

6) **Removing numbers, tabs (\n) and newline (\n).**

7) **Removing non-English characters/symbols:** that includes punctuation, mathematical symbols and non-English letters.

8) **Removing extra spaces:** after finishing all the previous steps, there might be extra spaces present so they were removed.

Figure 3 shows an example of the complete pre-processing procedure.

E. *Word Embeddings*

Representation of words is vital in NLP. The default approach of representing words as discrete symbols is deficient for many NLP tasks. For example, the words ”cat” and ”dog” are entirely unrelated, considering the letters represent them. However, we can, as humans, know that ”cat” and ”dog” are both ”pets”[15]. Therefore, we use word embeddings to represent the meaning of words in fixed-dimensional vectors. These word vectors capture semantic and syntactic similarities between words. Word embeddings are essential in various Natural Language Processing (NLP) tasks[17]. We used different types of word embeddings depending on the model uses:

*a) **FastText**:* The deep learning models implemented in this research use FastText word embeddings to represent the

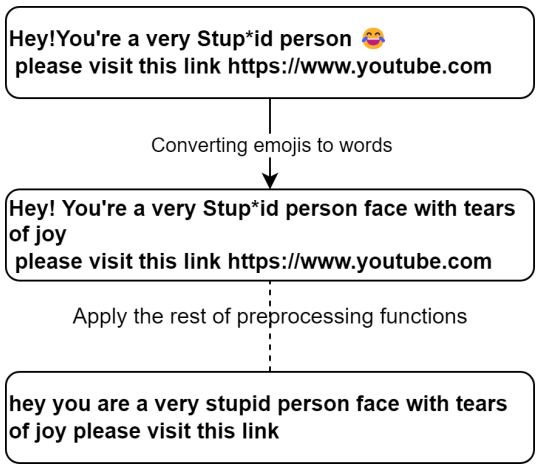

Fig. 3: Example of preprocessing a toxic comment

input words as vectors of 300 dimensions. We need word embeddings for our models because they do not accept the text as input. FastText is a word embeddings open-sourced project. FastText main goal is to consider not the word's representations but the internal structure of words [16]. FastText is similar Word2Vec. The main difference between Word2Vec and FastText is that FastText also learns vectors for subparts of words, and this will guarantee that, for example, the words "love", "loved", and "beloved" will all have the same vector representations [18].

*b) **Term Frequency-Inverse Document Frequency (TF-IDF)**:* TF-IDF is another approach for converting text into vectors. TF-IDF is a mathematical measure used to determine how important a word is to a comment in the dataset [20]. The term frequency (TF) measures the frequency of a word in a comment and is calculated as the number of times a word appears in a comment divided by the total number of words in the comment. The inverse document frequency (IDF) is a measure of how important a word is in the whole dataset, and it is calculated using the following equation:

$$IDF(word) = log(\frac{N}{documentsContain(word)}) \quad (1)$$

Where N is the total number of comments.

V. Models

After preprocessing our dataset, the train data set is used to train our models. We are using deep learning because deep learning algorithms have improved accuracy regarding text classification problems [23]. The deep learning neural network approaches used are Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) with long-term, short-term memory (LSTM). Both approaches were used for multi-class classification and binary classification. We also tried a more straightforward traditional method for binary classification: the Support Vector Machine (SVM). For our deep learning approaches, we used FastText to represent the vocabulary as vectors of 300d. Conversely, the Term Frequency-Inverse Document Frequency (TF-IDF) represents our vocabulary.

A. *Convolutional Neural Network (CNN)*

Convolutional Neural Network (CNN) is a type of deep-learning neural network [10]. CNN is widely used for image classification tasks and was proposed by Yann Lecun when he applied CNN for handwritten character recognition [21]. CNN was first proposed for text categorization tasks by Yoon Kim in 2014 and was applied using a single-layer convolution neural network and achieved perfect classification results [22]. CNN is a multilayer neural network consisting of multistage trainable Neural Networks architectures. Each stage consists of Embedding Layer, Convolutional Layers, Pooling Layers, and a Fully Connected Layer [7]. Figure A.1 shows the architecture of the CNN model used. The architecture in figure A.1 is used for multi-class classification. The only difference for binary classification is that the output is 1 for the last Dense layer.

B. *Long Short Term Memory (LSTM)*

Recurrent Neural Network (RNN) is a type of deep learning neural network structure that consists of a loop [23]. The architecture of RNN offers a tool to search for hidden patterns in textual data [24]. RNN can maintain information transferred through multiple layers within the recurrent network module. In other words, for every input, the output of the hidden layer depends on the past computation [23]. As a result, long-term dependency problem arises. Long-term dependency refers to the difficulty that recurrent neural networks (RNNs) have in remembering information from a long sequence of inputs. Sepp Hochreiter and Juergen Schmidhuber have proposed Long Short Term Memory (LSTM) neural networks to avoid long-term dependence on recurrent neural networks [19]. The architecture in A.2 is used for multi-class classification. It differs only with the last Dense layer, where in binary classification, it has only one output.

C. *Support Vector Machine (SVM)*

Support Vector Machine (SVM) is a supervised machine learning algorithm for classification and regression problems [13]. Standard SVM is used for binary classification tasks, which means that for each input, SVM predicts which of the two classes the input might be [12]. There are two types of SVM, linear and non-linear. In this research, we used linear kernel SVM because we applied SVM to linearly separable data.

The main idea behind linear SVM is to find the best hyperplane that separates the data points into two classes [25]. The hyperplane should have the maximum margin, which is the distance between the hyperplane and the closest data points of each class. After finding the best hyperplane, classification is done by checking which side the new data fall on [25]. Figure A.3 shows an example of an SVM model with linearly separable data.

VI. Models Evaluation

We have evaluated our models by making use of multiple evaluation metrics. Since we are working with an imbalanced dataset for our problem, we are using different evaluation metrics to compare the performances of our models. The evaluation metrics are calculated using the following acronyms:

TP = true positive, the number of toxic comments correctly identified as toxic.
FN = false negative, the number of toxic comments predicted as clean comments.
FP = false positive, the number of clean comments incorrectly identified as toxic comments.
TN = true negative, the number of clean comments correctly identified as clean comments.

Evaluation metrics:

a) ***Accuracy****:*

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

b) ***Precision****:*

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

c) ***Recall****:*

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

d) ***F1 Score****:*

$$F1Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5}$$

We used micro averaging for multi-class classification to calculate precision, recall, and F1-score. Micro averaging is a method of evaluating the performance of multi-class classification models. The model's overall precision, recall, and F1-score are obtained by summing the true positives, false positives, and false negatives across all classes and then computing the metrics using equations 3, 4 and 5.

Evaluating the models is done in two phases. First, all the models are tested and compared using Jigsaw's dataset. Second, the best models are chosen and tested on educational data from Discord.

A. *Evaluate Models On Jigsaw's Dataset*

The models are evaluated using the 63978 testing samples of Jigsaw's dataset. Table II shows the results of the metrics used. From the results, we can conclude that according to the F1 score, CNN for multi-classification was slightly better than LSTM. Regarding binary classification, SVM performed the best compared to CNN and LSTM.

Model	Accuracy	Precision	Recall	F1-score
CNN Multi	0.880	0.611	0.704	0.654
CNN Binary	0.904	0.506	0.891	0.645
LSTM Multi	0.874	0.589	0.733	0.653
LSTM Binary	0.910	0.523	0.847	0.646
SVM Binary	0.928	0.596	0.812	0.687

TABLE II: Comparison of models performances

B. *Evaluate Best Models On Educational Data*

In this phase, we tested CNN for multi-class classification model and SVM for a binary classification model on educational data from two different Discord servers of the University of Twente. We tested the models on 2865 samples from different channels. Figure 4 and 5 indicate the confusion matrix and the evaluation metrics of CNN and SVM, respectively.

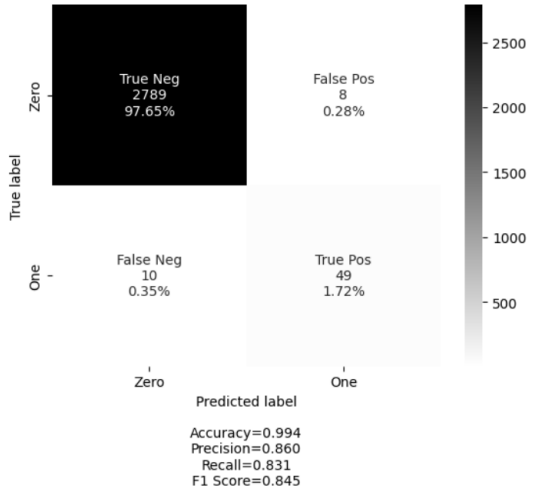

Fig. 4: Evaluation metrics on educational data using CNN-multi

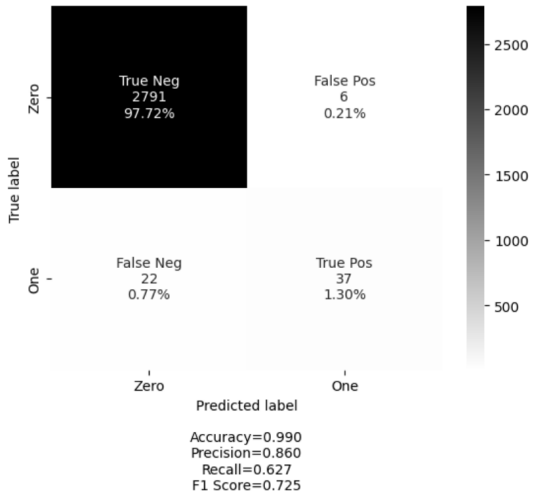

Fig. 5: Evaluation metrics on educational data using SVM

VII. Discord Bot

Discord is a popular platform used for voice and text communication. A community can create a server on Discord consisting of text and voice channels. In many courses at the University of Twente, it has been decided to use Discord for communication between students and teachers because almost all students are familiar with it. In order to automate functionalities in Discord servers, Discord allows server owners to build what are called Discord Bots. Discord bots are automated programs that can act as users and do various tasks. Discord Bots can be programmed using many programming languages. We used Python to program our bot by using the Discord.py[3] library.

After evaluating our models, CNN multi-class classification is chosen to integrate into the bot. Multi-class classification is used because it allows for more distinctions between different types of toxicity. Furthermore, it gives detailed reporting and analysis, as it can determine which type of toxic behavior is present in a given comment. Finally, when the models were tested on educational data, the performance of multi-class classification was better than binary classification.

ToxiClean is the name we defined to name our bot. ToxiClean can be invited to any server using the invite link generated from the Discord developer portal. When a message is sent it will be preprocessed according to IV-D. Afterwards, the model will predict whether the message contains toxic language. If the message is toxic, the bot will hide the message and send it to a queue channel to be reviewed by the server owners.

A. Features

In this section, we explain the features of ToxiClean and how it controls messages shared on text channels:

*a) **Classification**:* When the bot is run, all pre-processing functions and the trained machine learning model will be loaded depending on which classification is selected. When a message is sent on any text channel, it will be pre-processed and used as input to the trained model. The model will print the toxicity scores on the console. Figure 6 shows an example of a message shared on a text channel called general and the result of pre-processing.

```
ZZ#3525 said: 'Hey! How are you idi*ot? 😂' (general)
Original message: Hey! How are you idi*ot? 😂
Pre-processed message: hey how are you idiot face with tears of joy
```

Fig. 6: Example of message preprocessing

Depending on the type of classification, the model will return per class the probability as a value between 0 and 1 that the message is from that class. We convert the output to percentage to make the prediction results more readable than probability values between 0 and 1. Figure 7 shows an example of prediction based on six classification classes. If

```
Toxicity levels for 'Hey! How are you idi*ot? 😂':
Toxic:          98%
Severe Toxic:   1%
Obscene:        27%
Threat:         0%
Insult:         89%
Identity Hate:  1%
```

Fig. 7: Example of multi-classification using CNN

any of the classes has a prediction result greater than 50%, then the message will be hidden and posted in the toxic queue.

*b) **Toxic queue channel**:* is a hidden channel accessible only by users with the teacher role in Discord (teachers and teaching assistants). The main idea of this channel is to act as a queue for toxic messages. The model may classify messages incorrectly by considering them toxic, but they are not. Therefore, if a message is classified as toxic, then the bot will post the message in the toxic queue channel, where it can be reviewed later by teachers or teaching assistants.

The message will be posted in the toxic queue associated with multiple attributes. The attributes are essential for the teacher to know who shared the message and where it was posted.

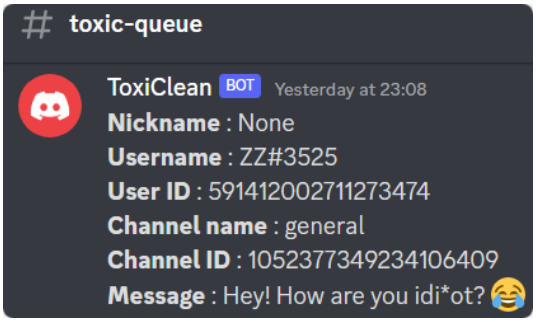

Fig. 8: Example of a message posted in the toxic queue channel

*c) **Accept/reject message**:* After the bot posts a message to the toxic queue channel, it will be reviewed by the teacher to decide whether the message is classified correctly or not. Accepting and rejecting messages can be done by reacting to the message with specific emojis. The teacher must react with ☑ to accept a message and with ✖ to reject a message using the built-in emojis of Discord. Figures 9 and 10 shows both cases.

Depending on the teacher's reaction, the message will be either deleted from the queue and re-posted in the channel where the original message was posted or deleted from the queue only.

VIII. Conclusion And Future Works

Toxic comment harms individuals. In this paper, we have proposed five approaches for detecting toxic comments. We

[3]Discord.py is a Python library that allows you to control a Discord bot and create applications that utilize the functionality of the Discord platform

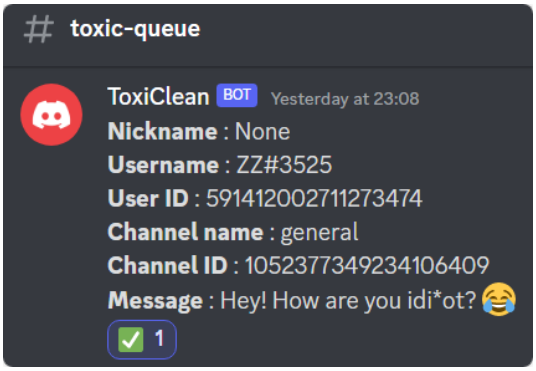

Fig. 9: Example of accepting a message

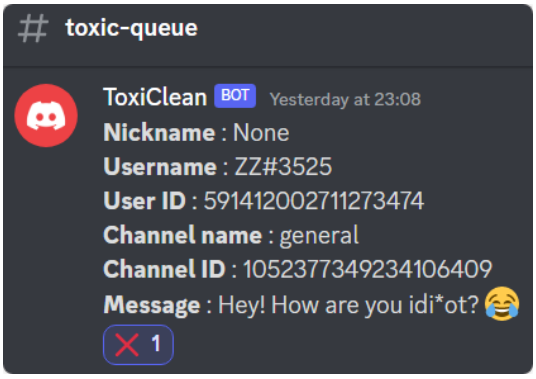

Fig. 10: Example of rejecting a message

used Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) with Long Short Term Memory (LSTM) as deep learning approaches and Support Vector Machine as a simpler traditional classification method. We can conclude from the results that CNN is more effective than LSTM for multi-class classification. SVM performed the best among CNN and LSTM for binary classification. In Discord, we integrated CNN with multi-class classification to classify messages into six categories. Massages classified as toxic will be hidden and queued to be approved by the teachers. If the teacher approves a message, it will be re-posted in the channel where it was initially shared.

For future work, we suggest working with a more balanced dataset to achieve higher performance [27]. Training a classifier with a balanced dataset will reduce biases because imbalanced data sets will cause the classifier to be biased towards the majority class [26]. The concept of bias is related to toxic comment classification because when the Conversation AI team first built machine learning models to detect toxicity, they discovered biases and incorrectness in learning [8]. Comments that contain terms of frequently attacked identities were considered toxic. The models predicted a high toxicity likelihood for comments containing those identities, even when the comments were not toxic [8]. We also suggest extending the features of the ToxiClean bot by, for example, warning or muting users who repeatedly make toxic comments.

REFERENCES

[1] H. Almerekhi, H. Kwak, J. Salminen, B. J. Jansen, Are These Comments Triggering? Predicting Triggers of Toxicity in Online Discussions, Proceedings of The Web Conference 2020, Taipei, Taiwan, Apr. 2020, pp. 3033-3040.

[2] M. Anand, R. Eswari, Classification of Abusive Comments in Social Media using Deep Learning, 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, Mar. 2019, pp. 974-977.

[3] S. Carta, A. Corriga, R. Mulas, D. R. Recupero, R. Saia, A supervised multiclass multi-label word embeddings approach for toxic comment classification, IC3K 2019 - Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, Vienna, Austria, 2019, pp. 105-112.

[4] A. G. D'Sa, I. Illina, D. Fohr, Towards non-toxic landscapes: Automatic toxic comment detection using DNN, ArXiv191108395 Cs Stat, Nov. 2019, Accessed: Jul. 03, 2020. http://arxiv.org/abs/1911.08395.

[5] S. Deshmukh, R. Rade, Tackling Toxic Online Communication with Recurrent Capsule Networks, 2018 Conference on Information and Communication Technology (CICT), Jabalpur, India, 2018.

[6] A. Elnaggar, B. Waltl, I. Glaser, J. Landthaler, E. Scepankova, F. Matthes, Stop Illegal Comments: A Multi-Task Deep Learning Approach, ACM International Conference Proceeding Series, 2018, pp. 41-47.

[7] S. V. Georgakopoulos, S. K. Tasoulis, A. G. Vrahatis, V. P. Plagianakos, Convolutional Neural Networks for Toxic Comment Classification, Proceedings of the 10th Hellenic Conference on Artificial Intelligence, Patras, Greece, Jul. 2018, pp. 1-6.

[8] S. Morzhov, Avoiding Unintended Bias in Toxicity Classification with Neural Networks, 2020 26th Conference of Open Innovations Association (FRUCT), Yaroslavl, Russia, Apr. 2020, pp. 314-320.

[9] Rahul, H. Kajla, J. Hooda, G. Saini, Classification of Online Toxic Comments Using Machine Learning Algorithms, 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, May 2020, pp. 1119-1123.

[10] M. A. Saif, A. N. Medvedev, M. A. Medvedev, T. Atanasova, Classification of Online Toxic Comments Using the Logistic Regression and Neural Networks Models, Proceedings of the 44th International Conference Applications of Mathematics in Engineering and Economics, Sozopol, Bulgaria, 2018.

[11] S. Srivastava, P. Khurana, Detecting Aggression and Toxicity using a Multi Dimension Capsule Network. Stroudsburg: Assoc Computational Linguistics-Acl, 2019, pp. 157-162.

[12] Murty, M.. (2008). Text Document Classification basedon Least Square Support Vector Machines with Singular Value Decomposition. International Journal of Computer Applications. Vol 27. 21-26.

[13] Hsu, C. W., Chang, C. C., & Lin, C. J. (2003). A practical guide to support vector classification.

[14] Perspective Developers. (2023). Perspectiveapi.com. https://developers.perspectiveapi.com/s/about-the-api-attributes-and-languages?language=en_US

[15] Levy, O., & Goldberg, Y. (2014, June). Dependency-based word embeddings. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (pp. 302-308).

[16] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the association for computational linguistics, 5, 135-146.

[17] Anke, L. E., & Schockaert, S. (2018, August). SeVeN: Augmenting word embeddings with unsupervised relation vectors. In Proceedings of the 27th International Conference on Computational Linguistics (pp. 2653-2665).

[18] I. Santos, N. Nedjah and L. de Macedo Mourelle, "Sentiment analysis using convolutional neural network with fastText embeddings," 2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI), Arequipa, Peru, 2017, pp. 1-5, doi: 10.1109/LA-CCI.2017.8285683.

[19] Hochreiter, S., Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[20] Aizawa, A. (2003). An information-theoretic perspective of tf–idf measures. Information Processing & Management, 39(1), 45-65.

[21] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[22] Yoon Kim. (2014). Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

[23] Luan, Y., Lin, S. (2019, March). Research on text classification based on CNN and LSTM. In 2019 IEEE international conference on artificial intelligence and computer applications (ICAICA) (pp. 352-355). IEEE.

[24] Ullah, A., Ahmad, J., Muhammad, K., Sajjad, M., & Baik, S. W. (2017). Action recognition in video sequences using deep bi-directional LSTM with CNN features. IEEE access, 6, 1155-1166.

[25] Pradhan, A. (2012). Support vector machine-a survey. International Journal of Emerging Technology and Advanced Engineering, 2(8), 82-85.

[26] Afzal, Z., Schuemie, M. J., van Blijderveen, J. C., Sen, E. F., Sturkenboom, M. C., & Kors, J. A. (2013). Improving sensitivity of machine learning methods for automated case identification from free-text electronic medical records. BMC medical informatics and decision making, 13(1), 1-11.

[27] Laradji, I. H., Alshayeb, M., & Ghouti, L. (2015). Software defect prediction using ensemble learning on selected features. Information and Software Technology, 58, 388-402.

APPENDIX

A. Appendix A

Models architecture of CNN and and LSTM used.

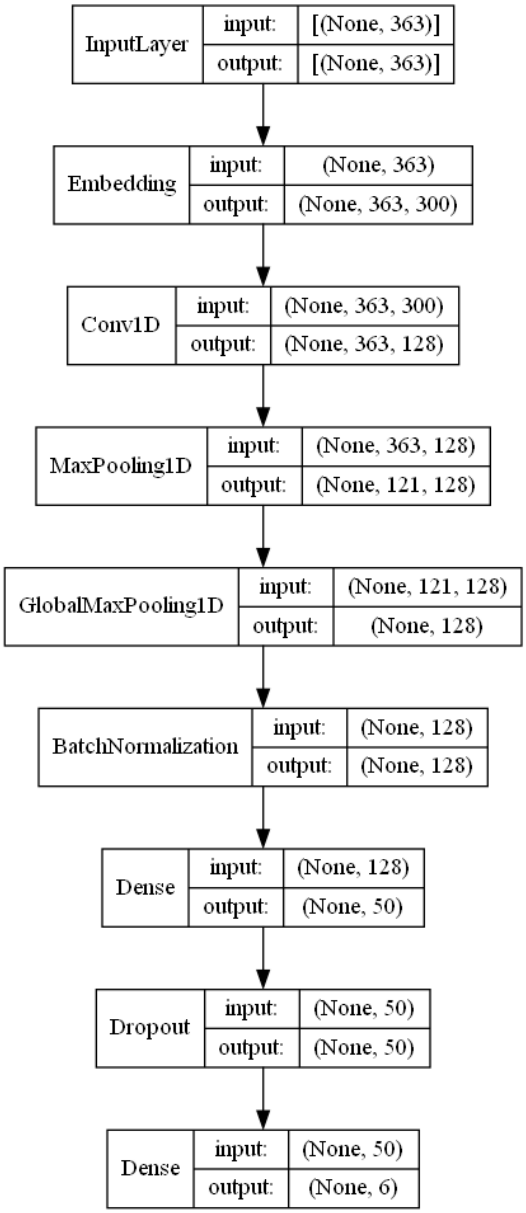

Fig. A.1: Architecture of CNN model

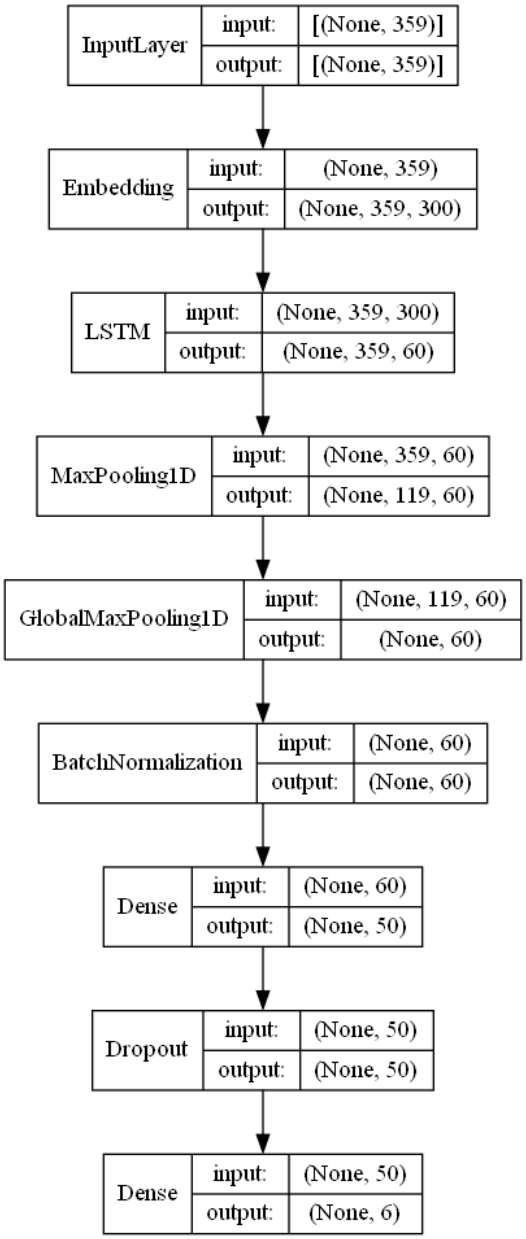

Fig. A.2: Architecture of LSTM model

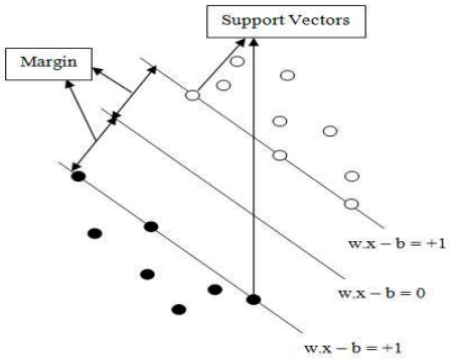

Fig. A.3: SVM model[25]