International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

Toxic Comment Classification using Deep Learning

[1]B.Ramesh Naidu,[2]Naresh Tangudu, [3]Ch. Chandra Sekhar, [4]K. Kavitha,[5]B.V.Ramana,[6]P.Venkateswarlu Reddy, [7]Jayavardhanarao Sahukaru, [8] Raj Ganesh Lopinti

[1,2,3,4,5,8]Dept. of IT, Aditya Institute of Technology and Management, Tekkali, Sriakkulam, A.P., India -532201
[6]School of Computing, Mohan Babu University, (Erstwhile Sree Vidyanikethan Engg. College(Autonomous), Tirupati, A.P. , India-517102
[7]Dept. of CSE, Aditya Institute of Technology and Management, Tekkali, Sriakkulam, A.P., India -532201.

Abstract— Online Conversation media serves as a means for individuals to engage, cooperate, and exchange ideas; however, it is also considered a platform that facilitates the spread of hateful and offensive comments, which could significantly impact one's emotional and mental health. The rapid growth of online communication makes it impractical to manually identify and filter out hateful tweets. Consequently, there is a pressing need for a method or strategy to eliminate toxic and abusive comments and ensure the safety and cleanliness of social media platforms. Utilizing LSTM, Character-level CNN, Word-level CNN, and Hybrid model (LSTM + CNN) in this toxicity analysis is to classify comments and identify the different types of toxic classes by means of a comparative analysis of various models. The neural network models utilized for this analysis take in comments extracted from online platforms, including both toxic and non-toxic comments. The results of this study can contribute towards the development of a web interface that enables the identification of toxic and hateful comments within a given sentence or phrase, and categorizes them into their respective toxicity classes.

Keywords- abusive comments, toxic classes, LSTM, CNN, Hybrid model, toxic classes, word-level and character-level, online interface.

I. INTRODUCTION

In recent years, online platforms and social networking website communities have become increasingly pervasive and vital for facilitating social interaction and data sharing. Undoubtedly, social networking website represents the most significant milestones of the 21st century. This podium provides a gigantic environment for their users to communicate ideas. The Internet is an open communication and multifaceted mass medium. However, the issues of harassment and cyberbullying have emerged as serious concerns that deter a vast majority of users from expressing their thoughts and opinions. In light of this challenge, our research aims to develop technology that utilizes deep learning models to detect the abusive language in online conversations, which will define as anything that is disrespectful, rude, or abusive. These toxic comments are then categorized into different classes such as toxic, severe-toxic, threat, insult, identity, obscene. It's noteworthy that in online conversations, it's possible for a single comment to contain multiple types of abuse and toxicity simultaneously. To build a deep learning model capable of detecting multiple types of abusive language in a given comment, this article utilized the multi-label jigsaw-toxic-comment-classification-challenge dataset provided by the Kaggle competition. The dataset used in our research comprises a significant quantity of comments and it has data imbalance. This problem is solved using random under-sampling and random over-sampling techniques. We trained various robotic models: long-short term memory (LSTM), character level, word level Convolutional neural network (CNN), and Hybrid model, which consists of the LSTM layer and CNN layer. then we performed a comparative analysis in terms of the performance of these trained models. we create an online web interface using Gradio app. this online interface takes the real-time comment as input in the string section and after submission of the comment it predicts the toxicity and classifies the comment into various toxic levels and represents the classification in the output section. The structure of this paper is in this way. In segment 3 presents the intricacies of the literature survey while in segment 4 we narrate the text data pre-processing, details of design and various methodologies involved in this paper. Segment 5 is devoted to results, which contain detailed information about the performance of the trained models. Finally, Segment 6 contains the outcome and potential directions for further research.

II. LITERATURE SURVEY

Correlated studies have examined inappropriate language, harassment, abusive remarks, cyber bullying, and inciting hatred. Toxic comment detection has become a study area, and researchers have developed numerous strategies to eliminate biases in Toxic Comment Detection and Classification. Detecting hate and abusive comments is a supervised classification problem that may be accomplished using neural networks [22] or manual feature engineering [26]. Aminu Tukur et al (2020) worked on Multi-label Binary Classification of toxic comments using Ensemble Deep learning. Ensemble learning integrates the single-model outputs to enhance generalization and predictions. researchers stated that Ensemble learning improves upon three crucial aspects of learning, statistics, and computation. Zaheri et al (2020) used

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

the RNN approach to identify the toxic comments. The magnificent parameter might be a series of terms tagged as belonging to a particular class. RNN-LSTM recognizes the comment as a group of pointed words identical to a time series, attempting to learn how the words in a time series closely related to a certain label are aligned. The models' presentation was compared to the benchmark model. Nayan Banik et al (2019) developed a method for detecting hateful and abusive comments that employ two common deep learning-based design known as Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) and proposed performance statistics of the various trained models. B. Vidgen et al. (2019) proposed the intrinsic unconsidered obstacles of toxic comment discern and potential clarification to them in a systematic manner and the researchers in [5] researched with Convolutional neural networks and stated that toxicity can be reduced over time and intrinsic intelligence can be obtained. According to Spiros V. Georgakopoulos et al. (2018), for text categorization issues, the information reflects the previously indicated analytical features based on the reality that neighbouring terms in a sentence have dependence, but their interpretation is not uncomplicated. The word embedding method was trained on a huge volume text of terms, generating a dense vector with a defined aspect and constant values for each word, and the values of this dense vector do not alter during the process of training a neural network model. Aken et al (2018) worked on categorization impediments of abusive comments and compare various neural network models and superficial techniques on a new, huge dataset of the comments, proposing an ensemble that exceeds the classification performance of all individual classifiers. Further, the researchers corroborate their experimental results on the alternative dataset. The ensemble results allow the researchers to conduct a comprehensive error analysis, which exposes the obstacles for further research. These difficulties include a lack of paradigmatic context and inconsistent dataset labeling. Mujahed A. Saif et al (2018) performed an analysis of LSTM and CNN models in terms of performance statistics. Among the two Long Short-Term Memory layers, four convolutional neural network layers, logistic regression, RNN and LSTM were performed. K.Kavitha et al(2022), Waseem et al (2017) stated that hate speech can be expressed in various forms. Implicit abusive or hateful comments can be expressed with a touch of satire and mockery [27][28]. Explicit abusive or hateful comments consist of disrespectful terms for example ‘shit’, ‘dumbasses, and ‘shithole’. Implicit abusive or toxic comments are frequently challenging to detect and require analysis of the semantics of comments. Explicit abusive or hateful comments can be recognized by using the lexicons of that comment and the automated identification and classification of abusive and hateful comments are challenging obstacles in NLP (Natural Language Processing). K.Kavitha et al. (2022)[8] proposed neural network models for automated abusive and toxic comment detection and classification are based on the numerical representation of the words and the features of classifiers on these numerical format representations (Nobata et al., 2016). And the vector values are tuned through the training process of convolution neural networks and support vector machines (SVM). Another common approach considered here is to utilize constant dense vectors for terms, which have been generated depending on word embedding ways such as word2vec [29] and GloVe [30]. These algorithms were trained on a huge term of terms, yielding a dense vector with a particular aspect and constant values for each word. All these papers perform the detection and classification of hateful and abusive comments using the deep learning models this paper considers. This report performed the observation and classification of hate speech and abusive comments using Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN). This research attempted to identify and categorize the obscene, insult, toxic, threat, severe-toxic, and racial-hate comments. Fig 2. describes the timeline diagram of the literature survey.

Figure 2. Literature Survey Timeline diagram

III. SYSTEM DESIGN & IMPLEMENTATION

System Design & Implementation is mainly categorized into three sections the first section is System Design and this section deals with the overall classifier design. The second category deals with text pre-processing, which consists of various data balancing techniques to balance the dataset and techniques that encodeuncleaned raw data into a computable format. The third category mainly deals with the selection of neural network models for training purposes. The combination

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

of these three approaches for a better outcome while using neural networks.

A. Design Analysis

The toxic comment classification model performs as illustrated in the following flowchart diagram. Moreover, the huge text dataset that is gathered from Kaggle consists of two datasets named the training dataset and the test dataset. The training dataset which includes 159752 comments and tweets has been utilized for model training and the test dataset consisting of 153165 tweets comments and tweets has been utilized for model testing. The neural network models can be trained and tested on real-world communication comments and tweets. The imbalanced text dataset can be balanced by using various balancing techniques These comments are pre-processed in such a way that the stop words are eliminated to obtain the most meaningful words and Lemmatization has been applied in order to recover the essence word from its multiple forms, Regular expressions are employed to eliminate hash tags, hyperlinks, punctuations, convert words from uppercase to lowercase to ignore distinctness in consideration of the identical term sowing to case sensitiveness. Most often used abbreviated variants of words are modified into their native English forms, resulting in a significantly cleaner dataset for the models to train on. In order to divide sentences into an array of terms, Tokenization is performed on train data. This process is essential for word vectorization. For the most common 50000 words in the text, utilized pre-trained word embeddings and fig 3.1. demonstrates the entire Workflow of the implementation.

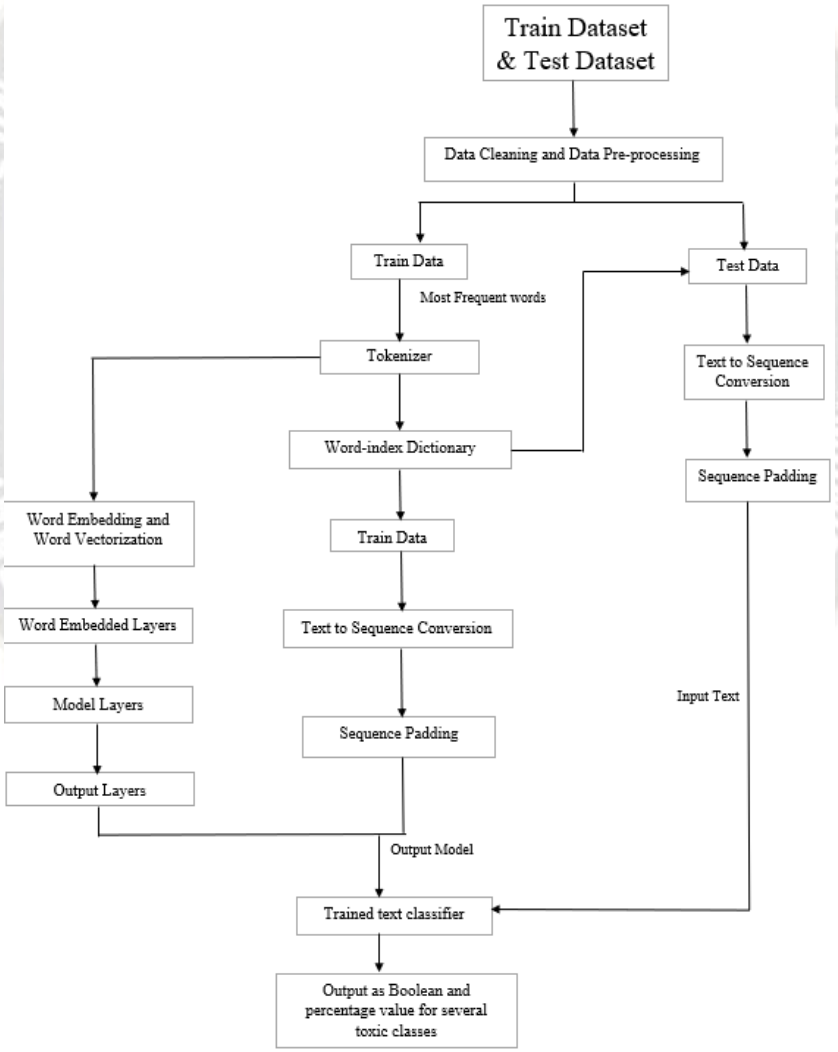

Figure 3.1. Workflow Diagram

Data pre-processing: Analyzing & Balancing Dataset: Two individual datasets were downloaded from Kaggle, the first dataset is for testing and the second dataset is for training purposes. These two datasets have a massive number of comments and tweets, and their unique IDs. it has six classification categories those are a insult, threat, toxic,

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

obscene, severe-toxics identity and moreover, each category of classification called "class" consists of two binary values "1" which indicates toxic or abusive comments, and "0" which represents the non-toxic comments or tweets and toxicity of their respective classification category. The second dataset is for testing purposes and it also has a large number of comments and unique IDs for the comments and tweets, Fig 3.2. Visualizes the total occurrences of toxicity classes.

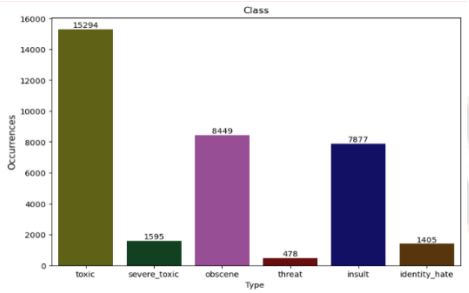

Figure 3.2. Toxicity classes occurrences

The Kaggle dataset has a huge number of tweets and comments, this dataset exhibits a class inequality and greater than 75% of the records in the dataset are non-toxic comments. This article overcame this obstacle by utilizing random sampling methods. This article uses both random under-sampling and random over-

sampling approaches to create new comments for the minority classes shown in fig 3.5. the header part of the training dataset and header part of the testing dataset was engraved in fig 3.3 and fig 3.4 respectively.

	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
5	00025465d4725e87	"\n\nCongratulations from me as well, use the ...	0	0	0	0	0	0
6	0002bcb3da6cb337	COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK	1	1	1	0	1	0
7	00031b1e95af7921	Your vandalism to the Matt Shirvington article...	0	0	0	0	0	0
8	00037261f536c51d	Sorry if the word 'nonsense' was offensive to ...	0	0	0	0	0	0
9	00040093b2687caa	alignment on this subject and which are contra...	0	0	0	0	0	0

Figure 3.3. Training Dataset

	id	comment_text
0	00001cee341fdb12	Yo bitch Ja Rule is more succesful then you'll...
1	0000247867823ef7	== From RfC == \n\n The title is fine as it is...
2	00013b17ad220c46	" \n\n == Sources == \n\n * Zawe Ashton on Lap...
3	00017563c3f7919a	:If you have a look back at the source, the in...
4	00017695ad8997eb	I don't anonymously edit articles at all.

Figure 3.4. Testing Dataset

Toxic classes	Before balancing		After balancing		Balancing techniques
	0	1	0	1	
Toxic	144277	15294	15294	15294	Random-under sampling
Severe toxic	157976	1595	1595	1595	Random-under sampling
Obscene	151122	8449	8449	8449	Random-under sampling
Threat	159093	478	159093	159093	Random-over sampling
Insult	151694	7877	151694	151694	Random-over sampling
Identity hate	158166	1405	158166	158166	Random-over sampling

Figure 3.5. Training Dataset Before and After Balancing

- **Data Cleaning:** The data which is gathered from the real world is typically in an unstructured and unorganized format, in order to eliminate extraneous values, the text data have to clean, fill in the missing value, eliminate data inconsistency, and by applying data cleaning techniques, can obtain quality and clean data.Data-cleaning operations performed by using the NLTK library. The cleaning operations are as follows:
  - ***Stemming:*** This technique is typically used to transform all the prefixes, suffixes, circum fixes, or infixes into

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

their stem word. This process is required to make all the words uniform

- ***Lemmatization:*** Lemmatization and stemming are two popular aproaches used in natural language processing to simplify words to their base or root forms. While stemming involves obtaining the stem of a word, lemmatization captures the canonical form of a word based on its lemma. For instance, if the word "Worst" as an example, stemming may not be able to obtain the precise form of the word. However, using the lemmatization technique, it is possible to convert the word to "Bad,"

- **Tokenization and conversion of text to sequence:**

Tokenization is the most commonly used text preprocessing technique and used that technique to segment an entire text into small units such as paragraphs, sentences, words, and alphabets. Training a neural network model with entire paragraph sentences and alphabets as a token for textual analysis and pre-processing is complicated as it requires a fundamental approach. So, in this process, there a need to instruct a neural network model on words as well as their significance. Hence, by utilizing the Keras tokenizer class, tokenized textual content into words.

To reduce the size of the embedding layer, the keras tokenizer was trained on the 50,000 most frequently occurring unique words in the training data, with less commonly used words being eliminated from consideration. A unique and specific integer is assigned to each word in the form of a word-index dictionary, allowing for the conversion of text-based data into a numeric format that can be digested by a neural network. This conversion process is crucial, as neural networks exclusively work with numerical data and necessitate this methodology for successful model training. The words of text in train data need to be converted to a sequence of numbers respectively as shown in fig 3.6.

```
Vocab size: 210438
Longest comment size: 1403
Average comment size: 68.22057892724868
Stdev of comment size: 101.07408115901785
Max comment size: 371

Sequence 1
  Input:  Explanation
Why the edits made under my username Hardcore Metallica Fan were reverted? They weren't vandalisms, just closure on some GAs af
ter I voted at New York Dolls FAC. And please don't remove the template from the talk page since I'm retired now.89.205.38.27
  Output: [688, 75, 1, 126, 130, 177, 29, 672, 4511, 12052, 1116, 86, 331, 51, 2278, 11448, 50, 6864, 15, 60, 2756, 148, 7, 293
7, 34, 117, 1221, 15190, 2825, 4, 45, 59, 244, 1, 365, 31, 1, 38, 27, 143, 73, 3462, 89, 3085, 4583, 2273, 987]
Sequence 2
  Input:  D'aww! He matches this background colour I'm seemingly stuck with. Thanks.  (talk) 21:51, January 11, 2016 (UTC)
  Output: [96148, 52, 2635, 13, 555, 3809, 73, 4556, 2706, 21, 94, 38, 806, 2682, 1008, 589, 8377, 182]
```

Figure 3.6. Conversion of Text input into Sequence of numbers

- **Word Vectorization:**

Word Vectorization, or Word Embedding, is an innovative method for extracting and retaining the semantic connections between disparate words. Although one-hot encoding can represent the words numerically and uniquely, it lacks the ability to identify the intricate relationships between words and their contextual meanings, which is a major drawback of this approach. Word embeddings function under the premise that a word's context and significance can be inferred from the words surrounding it. In comparison to one-hot encoding, which maps each word to a vector with a single 1 at its respective location, word embeddings can capture more nuanced information about the relationships between words. For instance, if our word vector consists of [hi, how, are, you], and we're examining the word "you," its corresponding input vector would be [0,0,0,1]. However, when working with larger vocabularies such as 210,000, one-hot encoding becomes less efficient and produces word vectors that are predominantly filled with zeroes and Word embedding involves generating dense vectors for each word that contain an array of numeric values. that reflects the word's interconnectivity with other words. Typically, word embeddings are created through training on vast amounts of data, although pre-existing word embeddings such as Google's word2vec and Face book’s Fast Text embedding can also be employed. In order to create a 300-dimension vector for each term in our terminology, utilized pre-rained Fast Text embeddings from Facebook. The benefit of this continuous embedding is that words with similar predictive power will appear closer together on our word vector and in this word embedding were found 999995-word vectors. The downside is that this creates more of a black box where the words with the most predictive power get lost in the numbers.

B. Models Implementation:

Several models were trained to classify various types of toxicity. Among these models, the LSTM classifier was chosen as the baseline model, owing to its prevalent use in the literature. Moreover, a CNN classifier was trained using both

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

Word and Character embeddings. Furthermore, a Hybrid model was developed by combining the LSTM and CNN classifiers. The Hybrid model comprises two layers, namely the CNN and the LSTM layer.

- ***Long Short-Term Memory (LSTM)***

The detection of toxic tweets and comments is carried out using the Long Short-Term Memory (LSTM) Recurrent Neural Network. The design of the LSTM is analogous to that of the conventional RNN, and the overall format is depicted in Figure 3.7.

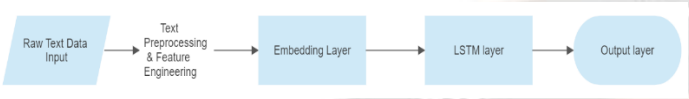

Figure 3.7. Structure of LSTM model

In the specific context of toxicity detection in textual content, the LSTM architecture (as illustrated in Figure 3.8) has emerged as a promising approach due to its unique cellular design. In this architecture, each cell operates autonomously and can selectively add or remove information to the cell state through the gate layers, enabling the model to capture and retain meaningful insights conveyed by individual words throughout the comment. Furthermore, the LSTM model comprises a dense layer with a predetermined number of units equivalent to the concatenated word vectors of every term in the comment. To construct the embedding layer, the Fast Text method, a proven technique in natural language processing, will be utilized. At the core of the LSTM model is the memory cell, referred to as the 'cell state,' which plays a vital role in maintaining the model's state over time. The cell state is depicted as a horizontal line in the diagram below and can be envisioned as a conveyor belt that allows information to flow smoothly and remain unaltered. Figure 3.9 provides review of the LSTM method, highlighting the significance of the memory cell in facilitating its functionality

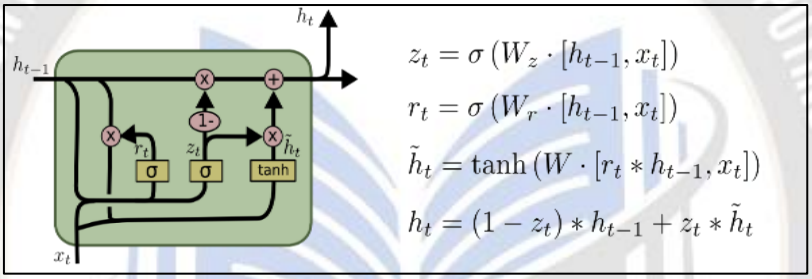

Figure 3.8. Architecture of LSTM

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 371, 300)          63131700

 lstm_layer (LSTM)           (None, 371, 60)           86640

 max_pooling1d (MaxPooling1D  (None, 123, 60)          0
 )

 global_max_pooling1d (Globa  (None, 60)               0
 lMaxPooling1D)

 batch_normalization (BatchN  (None, 60)               240
 ormalization)

 dense (Dense)               (None, 50)                3050

 dropout (Dropout)           (None, 50)                0

 dense_1 (Dense)             (None, 6)                 306

=================================================================
Total params: 63,221,936
Trainable params: 63,221,816
Non-trainable params: 120
```

Figure 3.9. LSTM model Summary

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
Article Received: 24 April 2023 Revised: 18 June 2023 Accepted: 04 July 2023

- **Character-level Convolutional Neural Network (CNN)**

This study employed a 1-Dimensional convolutional layer on the concatenated character embedding layer for character-level analysis of input

comments. This approach offers distinct advantages in

handling misspelled words, word permutations, and contextual word conjugation in languages. The model reads each character, including spaces, and generates a one-hot embedding of the comment. Detailed visualization of the model structure and summary is afforded in Figure 3.10 and Figure 3.11, respectively.

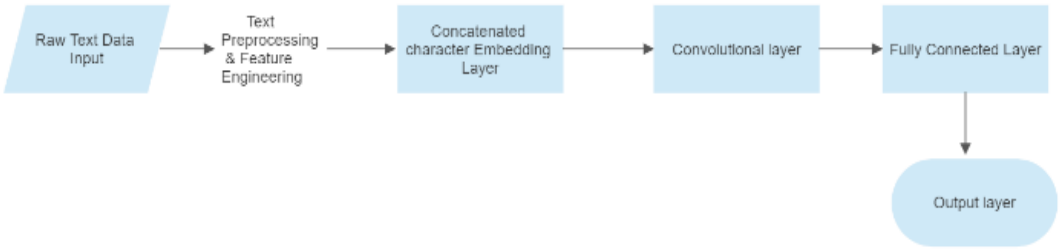

Figure 3.10. Character-level CNN model structure

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 2166, 300)         634500

 conv1d (Conv1D)             (None, 2166, 128)         192128

 max_pooling1d (MaxPooling1D  (None, 722, 128)         0
 )

 global_max_pooling1d (Globa  (None, 128)              0
 lMaxPooling1D)

 batch_normalization (BatchN  (None, 128)              512
 ormalization)

 dense (Dense)               (None, 50)                6450

 dropout (Dropout)           (None, 50)                0

 dense_1 (Dense)             (None, 6)                 306

=================================================================
Total params: 833,896
Trainable params: 833,640
Non-trainable params: 256
```

Figure 3.11: Character level CNN model summary

- ***Word-level Convolutional Neural Network (CNN)***

Our research methodology involved implementing a model based on the standard CNN approach. To create a concatenated word embedding layer, and utilized FastText word embeddings to align the words of the input text with a fixed dense vector.

This research study utilizes a CNN model that incorporates a 1-Dimensional convolutional layer applied to the concatenated word embeddings of each input comment. The convolutional layer encompasses 128 filters with a kernel size of 5, enabling it to consider a window of 5-word embeddings in each convolution operation. Following this, a Fully connected layer with 50 units is included, followed by the output layer, as depicted in Fig. 3.12. Fig. 3.13 provides a detailed overview of the model architecture.

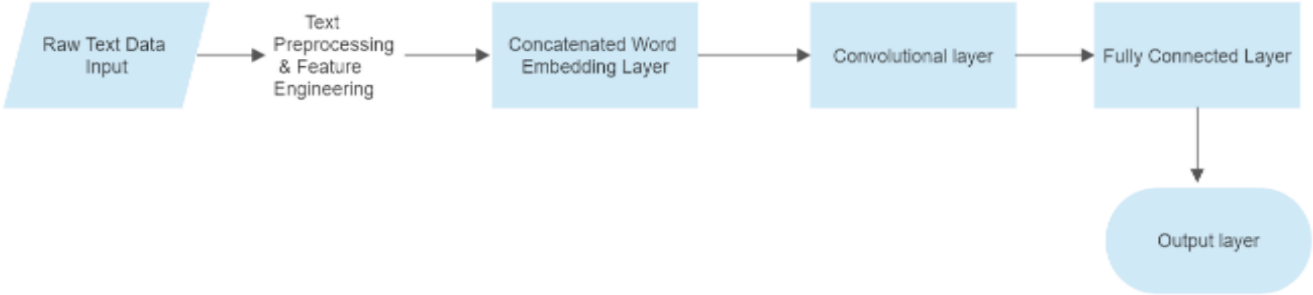

Figure 3.12. Word-level CNN structure

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

Embedding layer – Word vector representations.

Convolutional layer – run multiple filters over the data.

Fully connected layer – classify input based on filters

If, take a bag of words $q_{i, \ldots,} q_{i+K}$ the resulting vector is the concatenation of these words, as follows:

$$X_i = [q_i, q_{i+1}, \ldots, q_{i+k}] \in R^{k \times d} \quad (1)$$

After applying the convolution filter to each bag, obtain scalar values r_i, one for each i^{th}bag.:

$$r_i = g\,(x_i(u)) \in R \quad (2)$$

Usually, multiple filters$u_{1, \ldots,} u_L$ are utilized in practice, which can be represented as a vector multiplied by a matrix U, and then added to a bias term *b*:

$$r_i = g\,(x_i\,(U+b)) \quad (3)$$

with $r_i \in R^l$, $x_i \in R^{k \times d}$, $U \in R^{k \cdot d \times l}$ *and* $b \in R^l$

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 371, 300)          63131700

 conv1d (Conv1D)             (None, 371, 128)          192128

 max_pooling1d (MaxPooling1D  (None, 123, 128)         0
 )

 global_max_pooling1d (Globa  (None, 128)              0
 lMaxPooling1D)

 batch_normalization (BatchN  (None, 128)              512
 ormalization)

 dense (Dense)               (None, 50)                6450

 dropout (Dropout)           (None, 50)                0

 dense_1 (Dense)             (None, 6)                 306

=================================================================
Total params: 63,331,096
Trainable params: 63,330,840
Non-trainable params: 256
```

Figure 3.13. Word-level CNN model summary

- **Hybrid Model (LSTM + CNN)**

This research paper, present a novel hybrid model that combines two popular deep learning design, LSTM and CNN, CNN model will be constructed on top of the LSTM model, CNN structure extracts the text feature vector output from LSTM as shown in fig 3.14. This model's performance is compared with the previous models.

The LSTM structure presents a significant improvement over traditional RNN models. When it comes to extracting semantic information and features from lengthy text sequences. In this model, Specifically, the CNN component takes the output vector generated by the multi-layer LSTM as its input vector, thereby allowing for the extraction of additional characteristics from the input text sequences and improving the accuracy of the categorization process. fig 3.15 illustrates the summary of the hybrid model.

$$H = [h_1, h_2, \ldots, h_T]^T \quad (4)$$

$h_T \rightarrow$ the feature vector of the t-th word within a given text sequence, consisting of m dimensions.

T$\rightarrow$ the number of expansion steps within the LSTM architecture, equivalent to the length of the text sequences.

The quantity of hidden layer nodes in the LSTM architecture corresponds to the length of the vector.H $\in R^{m \times T}$is the matrix of input for the convolutional neural network. F $\in R^{j \times k}$is the convolutional filter, and j signifies the characterstic information derived from j neighbouring words during the convolution operation, while k denotes the aspect of the word embedding vector.

F = [$F_0, \ldots F_{m-1}$], where F is the convolution filter.At time step t, a singular value will be obtained as follows

$$O_{F_t} = ReLU\left[\left(\sum_{i=0}^{m-1} h_{t+i}{}^T F_i\right) + b\right] \quad (5)$$

b $\rightarrow$ bias

F, b$\rightarrow$ The parameters associated with this particular filter

The activation function utilized in this instance is RELU, and its expression is given by the following formula:

$$F(n) = \max(0, n) \quad (6)$$

To generate the probability distribution pertaining to the labels, softmax function is used. The equation governing the probability of classifying n as category **m** using the softmax function is as follows

$$P\left(y^{(i)} = m \,|n^{(i)};\, \theta\right) = \frac{e^{\theta_m^T n^{(i)}}}{\sum_{k=1}^{k} e^{\theta_m^T n^{(i)}}} \quad (7)$$

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

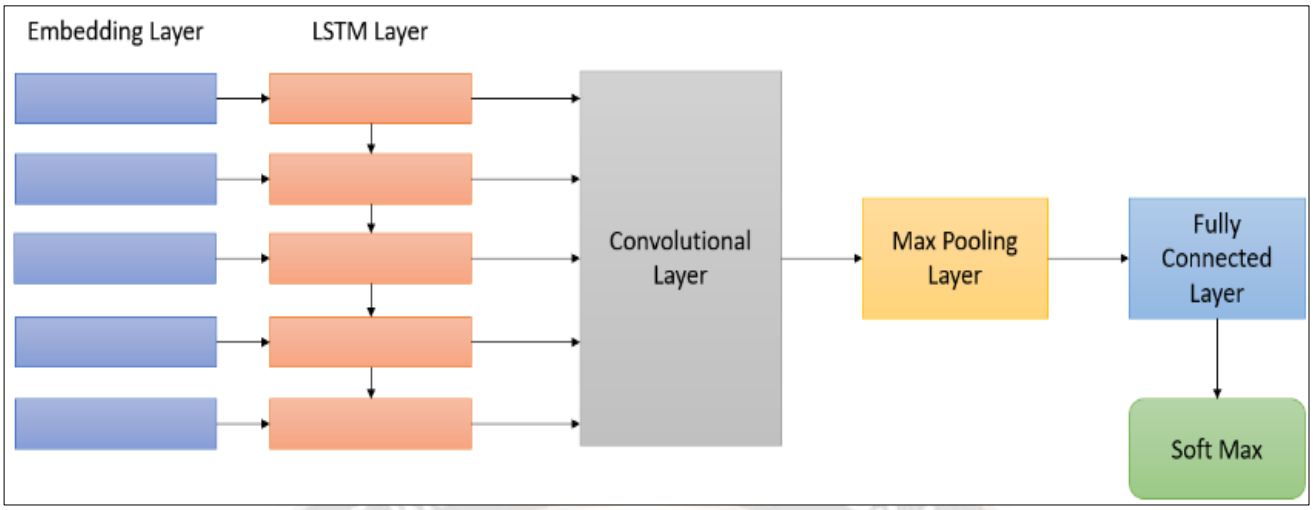

Figure 3.14. Hybrid model structure

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 371, 300)          63131700

 lstm_layer (LSTM)           (None, 371, 60)           86640

 conv1d (Conv1D)             (None, 371, 128)          38528

 max_pooling1d (MaxPooling1D  (None, 123, 128)         0
 )

 global_max_pooling1d (Globa  (None, 128)              0
 lMaxPooling1D)

 batch_normalization (BatchN  (None, 128)              512
 ormalization)

 dense (Dense)               (None, 50)                6450

 dropout (Dropout)           (None, 50)                0

 dense_1 (Dense)             (None, 6)                 306

=================================================================
Total params: 63,264,136
Trainable params: 63,263,880
Non-trainable params: 256
```

Figure 3.15. Hybrid model summary

IV. RESULTS

In this article, The AUC ROC score (area under the receiver operating characteristics curve) was initially utilized to conduct a comparative analysis of the model's performance. Character-level CNN has least AUC ROC score and Hybrid model has the highest AUC ROC score. Table 1 illustrates the AUC ROC score of the various model**.**

Model	LSTM	Word-level CNN	Character-level CNN	Hybrid model (CNN+LSTM)
AUC ROC score	0.9841	0.9747	0.9813	0.9842

Table 1: AUC ROC score of four models

Another comparative analysis performed by utilizing the accuracy score. Remarkably, all four methodologies demonstrated impressive performance, with a success rate exceeding 88%. The classification accuracy pertaining to the six toxic categories is illustrated in the subsequent table.

Clearly, the Hybrid model outperforms the remaining three models in terms of accuracy and Character-level CNN achieve the less accuracy than LSTM, Word-level CNN and Hybrid model, but the accuracy difference between these advanced methods and below Table 2 illustrates the classification accuracy of various models.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

Model	LSTM	Character-level CNN	Word-level CNN	Hybrid model (LSTM+CNN)
Accuracy	88.75%	83.54%	90.34%	92.63%

Table 2: Classification Accuracy

Below graph illustrates the comparison of classification accuracy and AUC ROC score of the four models .

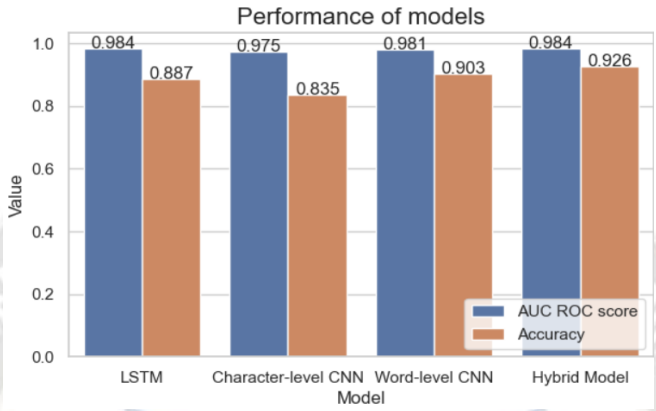

Figure 4.1. Performance Comparison

The fig 4.2 illustrates the classification of each comment into various toxic classes and the trained models provide the percentage of toxicity of various classes of given comment.

```
1/1 [==============================] - 0s 93ms/step
Toxicity levels for 'go jump off a bridge jerk':
Toxic:          99%
Severe Toxic:   13%
Obscene:        93%
Threat:         0%
Insult:         86%
Identity Hate:  1%

1/1 [==============================] - 0s 47ms/step
Toxicity levels for 'i will kill you':
Toxic:          96%
Severe Toxic:   0%
Obscene:        0%
Threat:         92%
Insult:         1%
Identity Hate:  0%

1/1 [==============================] - 0s 21ms/step
Toxicity levels for 'have a nice day':
Toxic:          0%
Severe Toxic:   0%
Obscene:        0%
Threat:         0%
Insult:         0%
Identity Hate:  0%
```

Figure 4.2. Classification of comments

The fig 4.3visualizes responsive web interface that created using gradio. It takes comment as input and classifies the comment into various toxic classes and visualizes the toxicity of given comment as shown in the below fig.

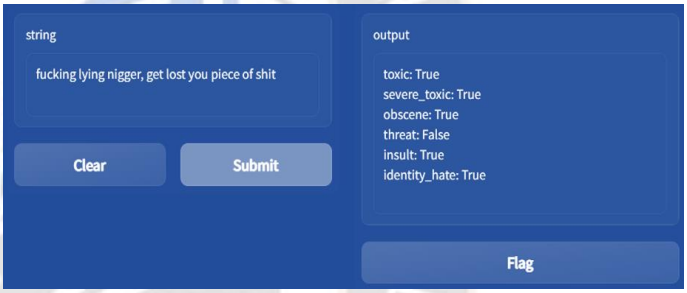

Fig 4.3: Gradio web interface

V. CONCLUSION & FUTURE SCOPE

In recent years, both the industrial and research communities have devoted significant efforts toward the development of an effectual model for the detection and classification of toxic comments in online communication channels. Given the criticality of this issue in the context of user engagement, our study was dedicated to the exploration of a novel methodology for text classification, which incorporates the use of Long Short-Term Memory and Convolutional Neural Networks. Through our investigation, we have established the potential of this hybrid model in effectively addressing the challenges associated with online toxic comment identification and classification. Our study involved the implementation of four distinct models, each aimed at detecting the levels of toxicity associated with online comments. Impressively, all four models demonstrated exceptional performance, highlighting the effectiveness of the various approaches in identifying and classifying toxic comments accurately. As per our study, the combination of LSTM neural network and Convolutional neural networks is an

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
***Article Received**: 24 April 2023 **Revised**: 18 June 2023 **Accepted**: 04 July 2023*

optimal approach for text classification tasks. Our results indicate that with sufficient and appropriate training data, the proposed model demonstrates remarkable performance on testing data, accurately classifying sentences into various toxicity categories while also providing the corresponding toxicity percentage for each sentence. This highlights the capability of the proposed model as a reliable and efficient gadget for detecting and classifying toxic comments in online communication platforms. Additionally, we created a web interface using Gradio. This interface allows us to give real-time comments as input to the model and classify the comment and visualizes the toxicity of that comment.

In future studies, we aim to investigate alternative methods for data augmentation and data pre-processing techniques to address the issue of class instability in the Kaggle jigsaw dataset. Additionally, we plan to enhance the performance of the model by exploring the possibility of enlarging the number of neural network layers to handle more intricate long-text classification tasks. Another intriguing approach we intend to explore is combining bidirectional LSTM and bidirectional GRUs in a single model. Such explorations have the potential to improve the performance of our proposed hybrid model for text classification tasks.

6. Acknowledgement

We have great pleasure to acknowledge our sincere gratitude to our research guide Dr. Naresh Tangudu, Asst Professor, Department of IT, AITAM, Tekkali for his help and Guidance during the research. His valuable suggestions and encouragement helped us a lot in carrying out this paper work. We are also very much thankful to Dr. Y Ramesh, Head of Department IT, for his help and valuable support in completing the paper.

References

[1] Ayush kumar, Pratik Kumar (2021). “Investigating Bias in Automatic Toxic Comment Detection: An Empirical Study”. arXiv:2108.06487 cs.CL]

[2] Georgios Patoulidid, Jonas Bokstaller (2021). “MAygual Zagidullinaodel Bias in NLP Application to Hate Speech classification using transfer learning techniques”. arXiv:2109.09725v4 cs.CL]

[3] Zhixue Zhao, Ziqi Zhang, Frank Hopfgartner (2021). “A comparative Study of using Pre-trained Language Models for Toxic Comment Classification”. IW3C2.

[4] Ashwin Geet, Irina Illin, Dominique Fohr (2021). “Classification of Hate Speech Using Deep Neural Networks”. CERIST 25(01). Hal-03101938

[5] Mihir Gada, Kaustubh Damania, Smita Sankhe (2021). “Cyberbullying Detection Using LSTM-CNN architecture and its applications”. IEEE 978-7281-5875-4.

[6] Kunfu Wang, Pengyi Zhang and Jian Su (2020). “A Text Classification Method Based on the Merge-LSTM-CNN Model”. ICNISC journal of Physics: conference Series.

[7] Muhammad Abubakar, Aminu Tukur, Usman Bukar usman (2020). “An Improved Multi-labeled LSTM Toxic Comment Classification”. Journal of Applied Sciences, Information, and computing Vo1ume 1, Number 2.

[8] An Intelligent Metaheuristic Optimization with Deep Convolutional Recurrent Neural Network Enabled Sarcasm Detection and Classification Model, International Journal of Advanced Computer Science and Applications(IJACSA), Vol. 13, No. 2, 2022 ..

[9] Gunjal, M. B. ., & Sonawane, V. R. . (2023). Multi Authority Access Control Mechanism for Role Based Access Control for Data Security in the Cloud Environment. International Journal of Intelligent Systems and Applications in Engineering, 11(2s), 250 –. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2623

[10] Vidyullatha, Satya Narayan Padhy, Javvaji Geetha Priya (2020). “Identification and Classification of Toxic Comment Using Machine Learning Methods”. Turkish Journal of Computer and Mathematics Education. Volume 12, Number 9.

[11] Sara Zaheri, Jeff Leath, David Stroud (2020). “Toxic Comment Classification”. SMU Data science Review. Volume 3, Number 1.

[12] Sergey Morzhov (2020) “Avoiding Unintended Bias in Toxicity Classification with neural networks”. 26th Conference of fruct association.

[13] Krishna Dubey, Rahul Nair, Mohd. Usman Khan (2020). “Toxic Comment Detection using LSTM”. Third International conference on ICAECC.

[14] Hao Li, Weiquan Mao, Hanyuan Liu (2019). “Toxic Comment Detection and Classification”. LREC.

[15] Nayan Banik, Md. Hasan Hanfizur Rahman (2019). “Toxicity Detection on Bengali Social Media Comments using Supervised Models.”. ICIET. Volume 23, Number 24.

[16] Mladen Karan and Jan Snadjer (2019). “Preemptive Toxic Language Detection in Wikipedia Comments Using Thread-Level Context”. Association for Computational Linguistics.

[17] B. Vidgen, A. Harris, D. Nguyen, R. Tromble, S. Hale, and H. Margetts (2019). “Challenges and frontiers in abusive content detection”. Association for Computational Linguistics.

[18] S. V. Georgakopoulos, S. K. Tasoulis, A. G. Vrahatis, and V. P. Plagianakos (2019), “Convolutional neural networks for twitter text toxicity analysis”.

[19] Alejandro Garcia, Machine Learning for Customer Segmentation and Targeted Marketing , Machine Learning Applications Conference Proceedings, Vol 3 2023.

[20] Mai Ibrahim, Marwan Torki and Nagwa El-Makky (2018). “Imbalanced Toxic Comments Classification using Data Augmentation and Deep Learning”. 17th IEEE International Conference on Machine Learning and Applications. In INNS Big Data and Deep Learning conference. Springer, 2019, pp. 370–379.

[21] Spiros V. Georgakopoulos, Sotiris K. Tasoulis, Aristidis G. Vrahatis, Vassilis P. Plagianakos (2018). “Convolutional Neural

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7834
Article Received: 24 April 2023 Revised: 18 June 2023 Accepted: 04 July 2023

Network for Toxic Comment Classification". Association for Computing Machinery. SETN.

[22] Kehn Wang, Jiaxi, Hongjun Wu (2018). "A survey of Toxic Comment Classification Methods".

[23] Jiarui Zhang, Yingxiang Li, Juan Tian, Tongyan Li (2018). "LSTM-CNN Hybrid Model for Text Classification". Third Conference, IAEAC.

[24] Z. Z. David Robinson and J. Tepper (2018). "Hate speech detection on twitter: Feature engineering vs feature selection". The Semantic Web: ESWC 2018 Satellite Events, pages 46–49.

[25] van Aken, B., Risch, J., Krestel, R., & Löser, A. (2018). "Challenges for toxic comment classification: An in-depth error analysis". arXiv preprint arXiv:1809.07572.

[26] Mujahed A. Saif, Alexander N. Medvedev, Maxim A. Medvedev, TodorkaAtanasova (2018). "Classification of Online Toxic Comments Using the Logistic Regression and Neural Networks Models".

[27] Mr. B. Naga Rajesh. (2019). Effective Morphological Transformation and Sub-pixel Classification of Clustered Images. International Journal of New Practices in Management and Engineering, 8(01), 08 - 14. https://doi.org/10.17762/ijnpme.v8i01.74

[28] Zhihao Guo, Shangquan Sun, Yicheng Zhou (2017). "Classification of Comment Toxicity". 17th Conference, Association for Computing Machinery.

[29] Moore, B., Clark, R., Martinez, J., Rodriguez, A., & Rodriguez, L. Anomaly Detection in Internet of Things (IoT) Data Streams. Kuwait Journal of Machine Learning, 1(4). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/151

[30] M. G. V. V. Pinkesh Badjatiya, Shashank Gupta (2017). "Deep learning for hate speech detection in tweets". WWW'17', pages 759– 760.

[31] K..Kavitha, Ch.Suneetha, Efficient Sentimental Analysis Using Hybrid Deep Transfer Learning Neural Network, International Journal of Trends and Technology(IJETT),ISSN: 2231-5381, Vol. 70 .Issue 10, 155-165, October-2022

[32] Waseem, Z., Davidson, T., Warmsley, D. and Weber (2017). "Understanding Abuse: A Typology of Abusive Language Detection Subtasks". In Proceedings of the First Workshop on Abusive Language Online (pp. 78-84).

[33] Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y. and Chang, Y. (2016). Abusive language detection in online user content. In Proceedings of the 25th international conference on world wide web (pp. 145-153).

[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. (2013). Distributed Representations of Words and Phrases and their Compositionality. In Advances in Neural Information Processing Systems 26, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3111–3119.

[35] M Ramakrishna Murty, JVR Murthy, and Prasad Reddy PVGD. (2011). Text Document Classification based-on Least Square Support Vector Machines with Singular Value Decomposition. International Journal of Computer Applications