STOCHASTIC GRADIENT DESCENT BASED HATE SPEECH RECOGNITION IN SOCIAL NETWORKS

A PROJECT REPORT

Submitted by

GOKUL S	**513419104015**
HARIPRAKASH M	**513419104017**
KISHORE KUMAR M	**513419104023**

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

UNIVERSITY COLLEGE OF ENGINEERING KANCHEEPURAM

(A Constituent College of Anna University, Chennai)

ANNA UNIVERSITY: CHENNAI – 600 025

MAY 2023

ANNA UNIVERSITY: CHENNAI – 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled **"STOCHASTIC GRADIENT DESCENT BASED HATE SPEECH DETECTION IN SOCIAL NETWORKS"** is the bonafide work of" **GOKUL S (513419104015), HARIPRAKASH M (513419104017) & KISHORE KUMAR M (513419104023) "** of Computer Science & Engineering who carried out this project work under my supervision.

SIGNATURE

Dr. SELVABHUVANESWARI,
M.E., (Ph.D.),

HEAD OF THE DEPARTMENT,
Assistant Professor,
Department of CSE,
University College of Engineering,
Kancheepuram .
Kanchipuram-631 552.

SIGNATURE

Mrs.T.KALA, M.E., (Ph.D)

SUPERVISOR,
Assistant Professor,
Department of CSE,
University College of Engineering,
Kancheepuram .
Kanchipuram-631 552.

Submitted for the Project Viva Voce held on : ……..

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We are always thankful to our college Dean **Dr.V.KAVITHA, M.E, (Ph.D.), PROFESSOR,** who endorsed us throughout this project.

Our heartfelt thanks to HOD **Dr.K.SELVABHUVANESWARI, M.E,(Ph.D.), ASSISTANT PROFESSOR,** Department of Computer Science and Engineering, University College of Engineering, Kancheepuram, for the prompt and limitless help in providing the excellent infrastructure to do the project and to prepare the thesis.

We express our deep sense of gratitude to our guide **Mrs.T.KALA, M.E, (Ph.D.) ASSISTANT PROFESSOR , DEPARTMENT OF COMPUTER SCIENCE &ENGINEERING,** for her invaluable support and guidance and encouragement for successful completion of this project. Her vision and spirit will always inspire and enlighten us.

We express our sincere thanks to the project committee members **Mrs.T.KALA, M.E, (Ph.D.), ASSISTANT PROFESSOR, Mr.G.MANI,M.E, (Ph.D.),ASSISTANT PROFESSOR**, Department of Computer Science and Engineering, Kancheepuram, for their invaluable guidance and technical Support.

We thank all the faculty members of Department of computer Science and Engineering for their valuable help and guidance. We are grateful to our family and friends for their constant support and encouragement. We thank the almighty whose showers of blessings made this project a reality.

GOKUL S	**HARIPRAKASH M**	**KISHORE KUMAR M**
(513419104015)	**(513419104017)**	**(513419104023)**

ABSTRACT

The detection of hate speech in social media is a crucial task. The uncontrolled spread of hate has the potential to gravely damage our society, and severely harm marginalized people or groups. As online content continues to grow, so does the spread of hate speech. It identifies and examines challenges faced by online automatic approaches for hate speech detection in text.

Among these difficulties are subtleties in language, differing definitions on what constitutes hate speech, and limitations of data availability for training and testing of these systems. Furthermore, many recent approaches suffer from an interpretability problem-that is, it can be difficult to understand why the systems make the decisions that they do.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACRONYMS	ABBREVIATIONS
IEEE	INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS
CNN	CONVOLUTIONAL NEURAL NETWORK
GUI	GRAPHICAL USER INTERFACE
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
XGBOOST	EXTREME GRADIENT BOOSTING
ML	MACHINE LEARNING
AI	ARTIFICIAL INTELLIGENCE
P2P	PEER-TO-PEER
SGD	STOCHASTIC GRADIENT DESCENT
LSTM	LONG SHORT-TERM MEMORY

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVE OF THE PROJECT

Hate speech is a big problem on the internet. It can be found on social media, in comment sections, and even in online forums. Machine learning algorithms can be used to detect hate speech. These algorithms can analyze text and identify hate speech. They can also be used to determine the tone of a text. This can be used to identify hate speech that is disguised as jokes or sarcasm. It uses Machine learning algorithm and NLP to detect the hate speech.

1.2 MOTIVATION

The nature of social media means that anyone can post anything they desire, putting forward any position, whether it is enlightening, repugnant or anywhere between. Depending on the forum, such posts can be visible to many millions of people. Different forums have different definitions of inappropriate content and different processes for identifying it, but the scale of the medium means that automated methods are an important part of this task. Hate-speech is an important aspect of this inappropriate content.

1.3 SCOPE OF THE PROJECT

Automated hate speech detection is an important tool in combating the spread of hate speech, particularly in social media. Numerous methods have been developed for the task, including a recent proliferation of deep-learning based approaches. A variety of datasets have also been developed, exemplifying various manifestations of the hate-speech detection problem. Now present here alarge-scale empirical comparison of deep and shallow hate-speech detection methods, mediated through the three most commonly used datasets. The goal is to illuminate progress in the area, and identify strengths and weaknesses in the current state-of-the-art. Particularly focus our analysis on measures of practical performance, including detection accuracy, computational efficiency, capability in using pre-trained models, and domain generalization. Indoing so, aim to provide guidance as to the use of hate-speech detection in practice, quantify the state-of-the-art, and identify future research directions.

1.4 LIMITATIONS OF THE PROJECT

The accuracy and reliability of the predictions generated by the model are highly dependent on the quality of the input data.The data is incomplete or contains errors, the predictions may be less accurate.

There are many layers to the difficulty of automatically detecting hateful and/or offensive speech, particularly in social media. Some of these difficulties are closely related to the shortcomings of keyword-based approaches.

CHAPTER 2

PRELIMINARIES

2.1 SECURITY

Hate speech detection is a critical application of machine learning, with the potential to mitigate the negative impact of hate speech on individuals and society as a whole. However, ensuring the security of hate speech detection models is essential to prevent their misuse or manipulation.

One of the primary security concerns with hate speech detection models is their vulnerability to adversarial attacks. Adversarial attacks involve modifying the input data in a way that causes the model to misclassify the input as a different class than it should. For example, an attacker could modify a hate speech input in a way that causes the model to classify it as non-hate speech. This could be done by adding or removing certain words or characters, or by changing the syntax or structure of the input. To mitigate these attacks, it is important to userobust training techniques and to test the model against a varietyof potential attacks.

Another security concern is the potential for bias in the data used to train the model. If the training data is biased, the model may learn to make incorrect classifications based on certain demographic or social characteristics, leading to unjust outcomes. To prevent bias, it is important to use diverse and representative training data and to monitor the model's performance for any signs of bias.

2.1.1 FEATURES OF HATE SPEECH RECOGNITION

Hate speech detection involves analyzing text to determine whether it contains language that expresses hatred, hostility, or discrimination towards a particular group based on their race, ethnicity, religion, gender, sexual orientation, or other characteristics. Some of the key features that can be used to detect hate speech include:

1. **Profanity and derogatory language**: Hate speech often contains profanity and derogatory language that is intended to insult or demean a particular group.

2. **Offensive terms and slurs:** Hate speech often includes offensive terms and slurs that are specifically targeted at a particular group.

3. **Stereotyping and generalization:** Hate speech often includes stereotypingand generalization of a particular group, which can be detected by analyzingthe language used in the text.

4. **Threats and incitement to violence:** Hate speech may include threats or incitement to violence against a particular group, which can be identified through language analysis.

5. **Intensity and frequency of language:** Hate speech may be characterized by an intensity and frequency of language that is more extreme than no hateful language.

2.2 DOMAIN

2.2.1 MACHINE LEARNING TECHNIQUE

2.2.1.1 WHAT IS MACHINE LEARNING?

Machine learning (ML) is a field of artificial intelligence (AI) that provides machines the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning algorithms use historical data as input to predict new output values. The goal of machine learning is to develop algorithms that can learn from data and make predictions or decisions based on that data.

2.2.1.2 HOW IT WORKS

Hate speech detection typically works by using natural language processing (NLP) techniques to analyze text and identify specific linguistic features that are associated with hate speech. Here is a general overview of how it works:

1. **Data collection:** The first step in hate speech detection is to collect a dataset of text that has been labeled as either hateful or non-hateful. This dataset is then used to train the machine learning model.

2. **Preprocessing**: The next step is to preprocess the text data by cleaning and normalizing it. This involves removing any irrelevant or noisy data, such as punctuation or special characters, and transforming the text into a standard format that the machine learning model can understand.

3. **Feature extraction**: In this step, NLP techniques are used to extract specific linguistic features that are associated with hate speech. These features can include things like profanity, derogatory language, offensive terms and slurs, and threats or incitement to violence.

4. **Machine learning**: Once the features have been extracted, they are used to train a machine learning model to classify text as either hateful or non hateful. Various machine learning algorithms can be used for this task, including logistic regression, support vector machines (SVMs), and deep learning neural networks.

5. **Evaluation**: After the machine learning model has been trained, it is evaluated using a test dataset to determine its accuracy and performance. The model is typically evaluated based on metrics such as precision, recall, and F1 score.

6. **Deployment**: Once the machine learning model has been trained and evaluated, it can be deployed to classify new text data as either hateful or non-hateful. This can be done in real-time, such as for online moderation of social media platforms, or as part of a batch process for analyzing large volumes of text data.

Overall, hate speech detection uses a combination of NLP techniques and machine learning algorithms to analyze text and identify specific linguistic features that are associated with hate speech. By accurately detecting hate speech, these models can help promote a safer and more inclusive online environment.

2.2.2 ALGORITHMS OF MACHINE LEARNING

2.2.2.1 Convolutional Neural Network:

In deep learning, a convolutional neural network is a class of artificial neural networks most applied to analyze visual imagery. CNNs use a mathematical operation called convolution in place of general matrix multiplication in at least one of their layers.

2.2.2.2 LSTM:

LSTM stands for long short-term memory networks, used in the field of Deep Learning. It is a variety of recurrent neural networks (RNNs) that are capable of learning long-term dependencies, especially in sequence prediction problems.

2.2.2.3 BI-DIRECTIONAL LSTM:

A Bidirectional LSTM, or bi LSTM, is a sequence processing model that consists of two LSTMs: one taking the input in a forward direction, and theother in a backwards direction.

2.2.2.4 XGBOOST:

XGBoost (extreme Gradient Boosting) is a supervised machine learning algorithm that is used for both classification and regression tasks. It is an ensemble method that combines multiple weak learners, typically

Decision trees, to create a more accurate model. XGBoost is a fast and efficient algorithm and has been used by the winners of many data science competitions. XGBoost works only with numeric variables and have already replaced the categorical variables with numeric variables. Let's havea look at the parameters that going to use in our model.

2.3 APPLICATIONS

There are several applications of hate speech detection, including:

1. **Online moderation**: Social media platforms and online communities can use hate speech detection to moderate user-generated content and remove harmful or offensive messages. This helps to create a safer and moreinclusive online environment.

2. **Law enforcement:** Hate speech detection can be used by law enforcement agencies to monitor online forums and social media platforms for hate speech that could lead to violence or other crimes.

3. **News media:** News outlets can use hate speech detection to monitor comments on their articles or social media platforms to ensure that discussions remain respectful and free from hate speech.

4. **Brand protection:** Companies can use hate speech detection to monitor social media for any negative or offensive comments about their brand or products. This helps companies to identify potential reputational risks and respond appropriately.

CHAPTER 3

LITERATURE SURVEY

3.1 Cross-Lingual Few-Shot Hate Speech and Offensive Language Detection Using Meta Learning

YEAR 2022

AUTHORS : Marzieh Mostafaei, Reza Farah bakhsh, Noel Crespi.

PUBLISHER : IEEE

CONCEPT :

Experiments show that meta learning-based models out perform transfer learning-based models in most cases, and that Proto-MAML is the bestperforming model, as it can quickly generalize and adapt to new languages

DRAWBACKS :

The disadvantage of this approach is that there is a limit to how much information these meta-features can capture, given that all these measures are uni- or bi-lateral measures only (i.e., they capture relationships between two attributes only or one attribute and the class).

3.2 Hate Speech Detection using Convolutional and Bi-directional Capsule Network

YEAR : 2022

AUTHORS : P. Preethy Jemima,Bishop Raj Majumder, Bibek.

PUBLISHER : IEEE

CONCEPT :

Data from social media sites such as Twitter are used to test the effectiveness of these procedures, and they reveal a 6-percentage point improvement in macro-average F1 or a 9 percent improvement for content that has been labeled as hateful, respectively.

DRAWBACKS :

The proposed HCovBi-Caps model detects hateful content with different contextual orientations. However, further improve it in detecting hate content considering different contextual semantics. Further, HCovBi- Caps does not exploit the sentiment and users' profile-related features, which may be effective. Also evaluate HCovBi-Caps over more diverse datasets. The HCovBi-Caps model detects the heat propagated in text only. Therefore, it can be extended to a multi-model approach for hate speech detection.

3.3 Deep Learning Based Fusion Approach for Hate Speech Detection

YEAR : 2022

AUTHORS : Yanling Zhou, Yanyan Yang, Han Liu, Xiu Feng, Liu,

PUBLISHER : IEEE

CONCEPT :

Hate speech is therefore considered as a serious problem worldwide, and many countries and organizations resolutely resist it . The polarity detection of speech on platforms is the first step and is critical to government departments, social security services, law enforcement and social media companies which expect to remove accounts with offensive content from their websites .

DRAWBACKS :

Deep learning works only with large amounts of data. Training it with large and complex data models can be expensive. It also needs extensive hardware to do complex mathematical calculations. Deep learning is a subset of machine learning. It is a field built on self-learning through the examination of computer algorithms.

3.4 PDHS: Pattern-Based Deep Hate Speech Detection

YEAR :2022

AUTHORS : P. Sharmila,Kalaiarasi Sonai ,MuthuAnbananthan,Deisy

PUBISHER : IEEE

CONCEPT :

Pattern-based Deep Hate Speech (PDHS) detection model was proposed to detect the presence of hate speech using a cross-attention encoder with a dual-level attention mechanism features. The first level of Attention is extracting aspect terms using predefined parts-of-speech tagging. The second level of Attention is extracting the sentimentpolarity to form a pattern.

DRAWBACKS :

- The normal network traffic pattern must be profiled first.
- It cannot detect unknown attacks.
- It is difficult to deploy in a large network.
- Its configuration is complex.

CHAPTER 4

PROPOSED ARCHITECTURE

4.1 PROPOSED ARCHITECTURE:

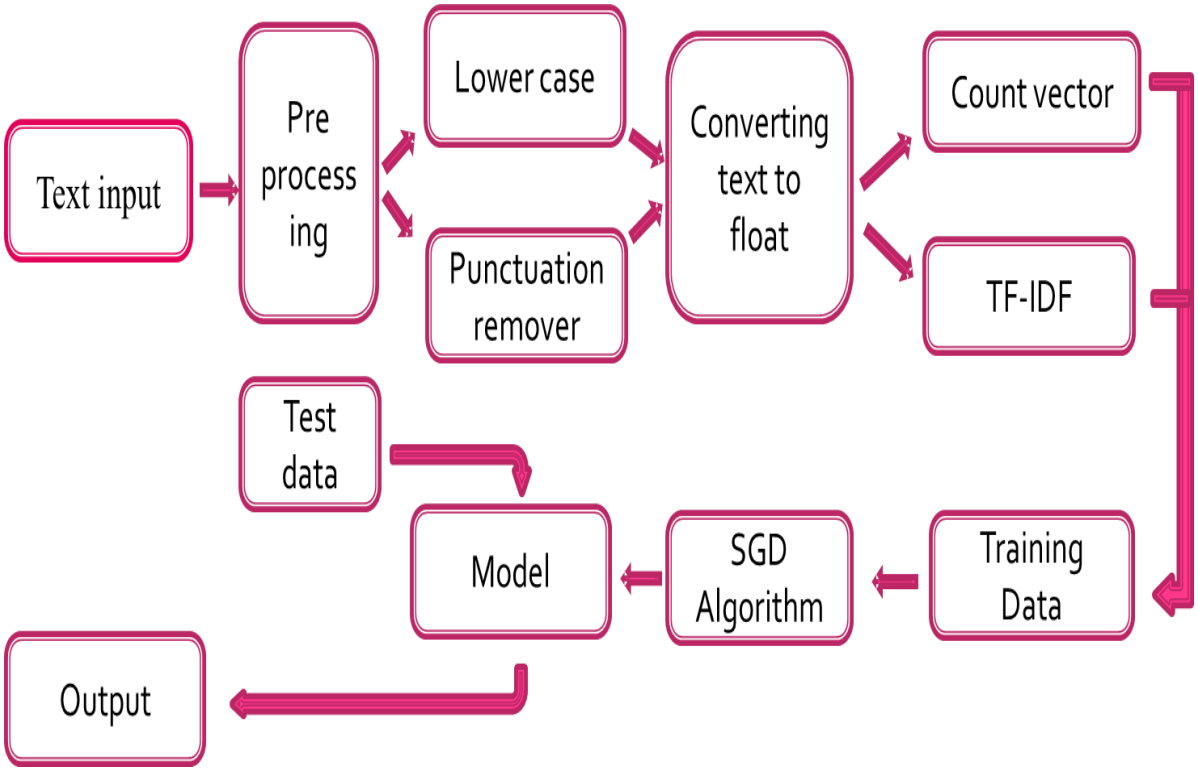

Fig.4.1 Proposed Architecture

PREDICTION MODEL PHASE:

The prediction model phase in machine learning involves using a trained machine learning model to make predictions on new, unseen data. This phase typically involves the following steps:

1. **Preprocessing the data**: The new data is preprocessed in the same way as the training data, including any feature scaling, normalization, encoding, or missing data handling.

2. **Applying the trained model**: The trained machine learning model is applied to the preprocessed data to make predictions. The model takes in the input features and produces output predictions based on its learned parameters.

3. **Evaluating the model performance**: The predictions made by the model are evaluated against the ground truth labels to measure its accuracy, precision, recall, F1 score, or other metrics, depending on the task and domain.

4. **Iterating and refining the model**: If the model performance is not satisfactory, the model parameters can be fine-tuned, or the model architecture can be modified and retrained on the available data until the desired performance is achieved.

GHAPTER 5

IMPLEMENTATION MODULES

4.1 LIST OF MODULES

- Data gathering
- Data Preprocessing
- Precision, recall
- Bot creating, prediction

DATA GATHERING

The Quant Connect Data Explorer is a portal to directly manipulate, validate and explore the Quant Connect back testing data source taken a radically open approach. Let's work together to provide the world's first crowd-curated data library for the community.

KAGGLE:

Kaggle is a popular online platform for data scientists and machine learning practitioners to find and participate in data science competitions, as well as to access datasets and explore data-related resources. Here are some general steps to follow when gathering data from Kaggle.

1. Create an account on Kaggle: You will need to sign up for a Kaggle account before you can access the datasets.

2. Browse the Kaggle datasets: Kaggle offers a variety of datasets, including datasets related to machine learning, natural language processing, computer vision, and more. You can browse the datasets by topic or search for specific datasets using keywords.

3. Choose a dataset: Once you find a dataset that interests you, read the description and any accompanying documentation to ensure that it meets your needs. Check the license and terms of use to make sure you can use the data for your intended purposes.

4. Download the dataset: Kaggle provides different formats to download the datasets, you can choose between CSV, JSON, SQLite, etc.

5. Understand the data: Before you start working with the data, it's important to understand its structure and any quirks it might have. This includes identifying missing values, outliers, and other issues that could affect your analysis.

6. Prepare the data: Depending on the data and your analysis goals, you may need to preprocess the data before you can use it. This can include cleaning the data, performing feature engineering, and normalizing or scaling the data.

Overall, Kaggle can be a great resource for gathering datasets, but it's important to carefully evaluate the quality and suitability of the data before using it for any analysis or modeling.

DATA PREPROCESSING

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always the case that we come across clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put it in a formatted way. So for this, we use data preprocessing tasks.

TF-IDF:

TF-IDF stands for Term Frequency-Inverse Document Frequency. It is a numerical statistic that is used to represent the importance of a term in a document or a collection of documents.

The TF-IDF score of a term in a document is calculated by multiplying the term frequency (TF) and the inverse document frequency (IDF). The term frequency is simply the number of times a term appears in a document, while the inverse document frequency is a measure of how rare or common a term is across all documents in a collection. The higher the TF-IDF score of a term in a document, the more important the term is to that document. Terms with higher TF-IDF scores are more likely to be useful for information retrieval, text classification.

Count Vectorization

Count Vectorization is a technique used to transform text data into a numerical representation that can be used in machine learning algorithms. It involves converting a collection of text documents into a matrix of word counts.

In count vectorization, each document is represented as a vector of word counts, where each element in the vector represents the number of times a particular word occurs in the document. The resulting matrix, called a count matrix, can be used as input to machine learning algorithms.

The process of count vectorization involves several steps:

1. Tokenization: The text data is first divided into individual words or tokens.

2. Vocabulary creation: A vocabulary is created from the unique tokens in the text data. This vocabulary serves as the columns of the resulting count matrix.

3. Count matrix creation: The count matrix is created by countingthe number of times each token appears in each document.

Count vectorization is a simple yet powerful technique for processing text data for machine learning. It is often used as a baseline approach for natural language processing tasks and can be improved upon using more advanced techniques such as TF-IDF or word embeddings.

Punctuation Remover:

Punctuation remover is a simple text preprocessing technique that involves removing all punctuation marks from a text document. Punctuation marks are characters such as periods, commas, question marks, and exclamationpoints that are used to separate sentences or clarify the meaning of a sentence.

Punctuation removal is often performed as a preprocessing step before performing natural language processing tasks such as text classification, sentiment analysis, or text summarization. Removing punctuation can help simplify the text and reduce the size of the vocabulary, which can improve the performance of the machine learning algorithms used in these tasks.

5.2 ALGORITHM

CHOOSE THE MODEL

5.2.1 STOCHASTIC GRADIENT DESCENTALGORITHM

Stochastic Gradient Descent (SGD) is a variant of the Gradient Descent algorithm used for optimizing machine learning models. In this variant, only one random training example is used to calculate the gradient and update the parameters at each iteration. Here are some of the advantages and disadvantages of using SGD.

Suppose, you have a million samples in your dataset, so if you use a typical Gradient Descent optimization technique, you will have to use all of the one million samples for completing one iteration while performing the Gradient Descent, and it has to be done for every iteration until the minima are reached. Hence, it becomes computationally very expensive to perform.

This problem is solved by Stochastic Gradient Descent. In SGD, it uses only a single sample, i.e., a batch size of one, to perform each iteration. The sample is randomly shuffled and selected for performing the iteration.

Stochastic gradient descent (SGD) is an iterative optimization algorithm commonly used in machine learning to minimize the cost function of a model. It is a variant of gradient descent that updates the model's parameters based on a randomly selected subset of training examples, known as a mini-batch, instead of the entire dataset.

The algorithm works by first initializing the model's parameters with random values. Then, for each iteration, the algorithm randomly selects a mini-batch of training examples from the dataset and computes the gradient of the cost function with respect to the parameters using only the examples in the mini-batch. The algorithm then updates the parameters in the opposite direction of the gradient, scaled by a learning rate parameter. The learning rate determines the step size of the update and must be chosen carefully to balance between convergence speed and stability.

The process is repeated for a fixed number of iterations or until convergence, which is typically determined by monitoring the change in the cost function or the model's performance on a validation set. The SGD algorithm is widely used in deep learning, where the datasets are often large and computationally expensive to process, making it impractical to compute the gradient over the entire dataset in each iteration.

5.2.2 LSTM:

LSTM stands for Long Short-Term Memory, and it is a type of artificial recurrent neural network (RNN) architecture that is commonly used in natural language processing (NLP) tasks such as text classification, language translation and speech recognition.

LSTM networks are designed to address the vanishing gradient problem that is often encountered in standard RNNs. The vanishing gradient problem occurs when gradients propagated back through the network during training become too small to update the weights effectively. This problem can make it difficult for RNNs to learn long-term dependencies in sequential data.

LSTM networks use a more complex architecture than standard RNNs, with additional memory cells and gates that control the flow of information through the network. The three key components of an LSTM network are:

1. Cell state: This is the memory of the LSTM network. The cell state is updated by adding or removing information from it via gates.

2. Gates: Gates are used to control the flow of information through the network. They are made up of a sigmoid neural net layer and a pointwise multiplication operation. The three types of gates in an LSTM network are the input gate, the forget gate, and the output gate.

3. Hidden state: This is the output of the LSTM network, which is based on the cell state and the input at the current time step.

LSTM networks are particularly useful in tasks that require processing of long sequences of data, where traditional RNNs may struggle. They have been successfully applied to a wide range of NLP tasks, such as sentiment analysis, text generation, and machine translation.

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) that is designed to address the problem of vanishing gradients in traditional RNNs, which can occur when the gradient signal becomes too small as it is propagated through many time steps. LSTMs are commonly used in deep learning for sequential data analysis such as speech recognition, natural language processing, and video analysis.

The basic building block of an LSTM is the LSTM cell, which contains three gates: the input gate, the forget gate, and the output gate. The input gate controls the flow of new input into the cell, the forget gate controls the amount of information that is discarded from the cell, and the output gate controls the flow of output from the cell. Each gate is implemented as a sigmoid function that outputs a value between 0 and 1, indicating how much of the input should be allowed to pass through.

The LSTM cell also contains a memory cell that stores the internal state of the cell and is updated by the input and forget gates. The output of the cell is determined by the output gate and is also used to update the memory cell.

During training, the LSTM is fed a sequence of inputs, and at each time step, the LSTM cell updates its internal state based on the input, previous output, and previous memory cell state. The LSTM outputs a prediction at each time step, and the parameters of the LSTM are updated using backpropagation through time.

LSTMs have been shown to be effective in capturing long-term dependencies in sequential data and have been widely used in various applications.

5.2.3 CNN:

CNN stands for Convolutional Neural Network, which is a type of artificial neural network commonly used for image and video recognition, object detection and classification tasks.

The basic building blocks of a CNN are convolutional layers, pooling layers, and fully connected layers. Convolutional layers apply a set of filters to

the input image, which extracts features from the image by convolving the filters with the image data. Pooling layers down sample the output of the convolutional layers to reduce the spatial dimensions of the data, which helps to reduce the computational complexity of the network. Finally, fully connected layers perform the classification or regression task by mapping the output of the previous layers to the output label or value.

Convolutional Neural Networks (CNNs) are a type of deep neural network that are commonly used in image recognition and computer vision tasks. They are designed to automatically learn and extract relevant features from images, allowing for high accuracy in image classification and object detection.

CNNs are composed of multiple layers, each of which performs a specific operation on the input data. The most important layers in a CNN are the convolutional layers, which apply a set of filters to the input image to extract features such as edges, corners, and textures.

During training, the filters are learned through backpropagation, which adjusts the weights of the filters to minimize the difference between the predicted output and the actual output. The output of the convolutional layer is then passed through a non-linear activation function, such as ReLU, to introduce non-linearity into the model.

After several convolutional and activation layers, the output is flattened into a vector and fed into a fully connected layer, which performs the final classification decision based on the extracted features.

CNNs can also include other types of layers such as pooling layers, which reduce the spatial dimensions of the input by applying a function such as max or average pooling, and dropout layers, which randomly drop out nodes during training to prevent overfitting.

Overall, CNNs have proven to be highly effective in image recognition tasks, achieving state-of-the-art performance on a wide range of benchmarks.

5.2.4 Bi-directional LSTM:

A Bidirectional LSTM (Bi-LSTM) is a type of Long Short-Term Memory (LSTM) network that processes the input data in two directions: from the beginning to the end, and from the end to the beginning. This means that the input sequence is read both forward and backward, allowing the network to capture dependencies and patterns in the data that may be missed by a unidirectional LSTM.

In a Bi-LSTM, the input sequence is first passed through two separate LSTM layers, one in the forward direction and one in the backward direction. The output of each layer is then concatenated to produce the final output sequence.

Bidirectional Long Short-Term Memory (BiLSTM) is a type of recurrent neural network (RNN) that consists of two LSTMs, one processing the input sequence in a forward direction and the other processing the input sequence in a backward direction. The outputs of the two LSTMs are then concatenated to form the final output.

Bi –Dir LSTMs are commonly used in natural language processing (NLP) tasks such as sentiment analysis, named entity recognition, and machine translation, where the context in both directions is important for understanding the meaning of a sentence.

During training, the forward LSTM processes the input sequence from the first to the last element, while the backward LSTM processes the input sequence from the last to the first element. The output at each time step is a concatenation of the forward and backward LSTM outputs, which provides a more complete representation of the input sequence.

At each time step, the forward LSTM updates its internal state based on the input, previous output, and previous memory cell state, while the backward LSTM updates its internal state based on the input, subsequent output, and subsequent memory cell state.

The final output of the Bi LSTM is a concatenation of the forward and backward LSTM outputs at each time step. This output can be fed into a fully connected layer for classification or further processing.

Bi-Dir LSTMs have been shown to be effective in capturing the contextual information in both directions, leading to improved performance in NLP tasks compared to traditional unidirectional LSTMs.

5.3 WORKING OF SGD ALGORITHM:

The basic steps involved in the SGD algorithm are as follows:

1. Initialize the model parameters: The algorithm starts with initializing the model parameters with some initial values.

2. Choose a mini-batch: The algorithm randomly selects a small subset (mini-batch) of the training data.

3. Compute the cost function: The cost function is calculated on the mini-batch using the current model parameters.

4. Compute the gradients: The gradient of the cost function with respect to each parameter is calculated on the mini-batch.

5. Update the model parameters: The parameters are updated by moving in the direction of the negative gradient of the cost function. The learning rate determines the step size in the update equation.

6. Repeat steps 2-5: The algorithm repeats the above steps on different mini-batches of the training data until it reaches convergence.

SGD is called stochastic because it updates the model parameters based on the gradient calculated on a randomly selected mini-batch, as opposed to the whole dataset. This randomness helps the algorithm to escape from local minima and improve the model's generalization performance.

5.4 EVALUATE MODEL

PRECISION:

Precision is a performance metric used in machine learning and statistics to evaluate the accuracy of a model's positive predictions. It is defined as the ratio of true positives to the sum of true positives and false positives. In other words, precision measures how many of the positive predictions made by the model are correct.

Precision is particularly useful when the cost of a false positive prediction is high. For example, in a medical diagnosis task, a false positive result could lead to unnecessary medical procedures or treatments, which can be costly and potentially harmful to the patient. In this case, a model with high precision would be preferred, as it would minimize the number of false positives.

RECALL:

Recall is a performance metric used in machine learning and statistics to evaluate the ability of a model to correctly identify positive instances in the data. It is defined as the ratio of true positives to the sum of true positives and false negatives. In other words, recall measures how many of the actual positive instances in the data were correctly identified by the model.

Recall is particularly useful when the cost of a false negative prediction is high. For example, in a disease detection task, a false negative result could mean that a patient with the disease is not identified and does not receive thenecessary treatment.

F-1 SCORE:

The F1 score is a harmonic mean of precision and recall, and is often used as a summary metric to evaluate the performance of a binary classification model. TheF1 score combines both precision and recall into a single metric that provides a balanced evaluation of the model's performance.

Here's the formula for F1 score:

F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

The F1 score ranges from 0 to 1, with a higher score indicating better performance. A perfect F1 score of 1 means that the model has perfect precision and recall, and is correctly identifying all the positive instances in the data.

The F1 score is a useful metric when both precision and recall are important for a given task. For example, in a fraud detection task, both precision and recall are important to minimize the number of false positives and false negatives, respectively. In this case, a high F1 score would indicate that the model is effectively balancing both precision and recall.

It's worth noting that the F1 score can be biased towards precision or recall, depending on the distribution of the data and the threshold used to make predictions. In some cases, other metrics such as the area under the ROC curve (AUC) or the average precision (AP) may be more appropriate for evaluating model performance.

METHODOLOGY

The methodology of our proposed project has three major steps that involves.

- Dataset collection,
- Data preprocessing,
- Features extraction,
- Classification,
- Model training and NLP.

5.5 DATASET COLLECTION

Data collection is the process of gathering and measuring in formation on variables of interest, in an established systematic fashion that enables one to answer stated research questions, test hypotheses, and evaluate outcomes. The data collection component of research is common to all fields of study including physical and social sciences, humanities, business, etc. While methods vary by discipline, the emphasis on ensuring accurate and honest collection remains the same. The platform called KAGGLE for dataset collection.

```
In [2]: df = pd.read_csv('labeled_data.csv')
        df.head()
```

Out[2]:

	Unnamed: 0	count	hate_speech	offensive_language	neither	class	tweet
0	0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	4	6	0	6	0	1	!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...

Fig.5.2 Data set

5.6 DATASET PREPROCESSING

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always the case that we come across clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put it in a formatted way. So for use data preprocessing tasks. The dataset is cleaned by assigning the mean and median values to the missing null values or undefined values.

5.7 FEATURES EXTRACTION

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. A characteristic of these large data sets is a large number of variables that require a lot of computing resources to process. Feature extraction is the name for methods that select and or combine variables into features, effectively reducing the amount of data that must be processed, while still accurately and completely describing the original data set. In the proposed system, the dataset is splitter into Categorical and Numerical variables.

5.8 CLASSIFICATION

The Processed and extracted variable is further given as input to the classification. The proposed system using the combination of supervised algorithm .In which the STOCHASTIC GRADIENT DESCENT ALGORITHM has achieved up to 96% accuracy.

5.9 PLATFORM OF IMPLEMENTATION

SOFTWARE REQUIREMENTS

The Software is a set of instructions that are used to command any systems to perform any operations .The software has the advantage to make decisions and to hardware sensible results and is useful in handling complex situations.

OS : Windows 10, Linux

LANGUAGE : Python

LIBRARIES : Pandas, NumPy, Matplotlib, Seaborn.

5.9.1 VISUAL STUDIO CODE IDE

Visual Studio Code is a freeware source-code editor made by Microsoft for Windows, Linux and MacOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality. In the Stack Overflow 2019 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 50.7% of 87,317 respondents reporting that they use it. VS Code can be extended via extensions, available through a central repository. This includes additions to the editor and language support. A notable feature is the ability to create extensions that add support for new languages, themes, and debuggers, perform static code analysis, and add code linters using the Language Server Protocol. Visual Studio Code includes multiple extensions for FTP, allowing the software to be used as a free alternative for web development.

5.9.2 PYTHON (PROGRAMMING LANGUAGE)

Python is an interpreted, high-level, and general-purpose programming language.Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically-typed and garbage-collected.

It supports multiple programming paradigms, including structured (particularly, procedural), object- oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. Python was designed to be highly extensible (with modules). This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Libraries such as NumPy matplotlib allow the effective use of Python in scientific computing, with specialized libraries such as Bio python and Atrophy providing domain-specific functionality.

Sage Math is a mathematical software with a notebook interface programmable in Python: its library covers many aspects of mathematics, including algebra, combinatory, numerical mathematics, number theory, and calculus. Open CV has python bindings with a rich set of features for computer vision and image processing.

Python is commonly used in artificial intelligence projects and machine learning projects with the help of libraries like TensorFlow, Keras, Pytorch and Sickler.As a scripting language with modular architecture, simple syntax and rich text processing tools, Python is often used for natural language processing

.

5.9.3 PANDAS

Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures andoperations for manipulating numerical data and time series.

Pandas is an open-source Python library that is widely used for data analysis and manipulation. It provides powerful data structures for working with structured data, such as tabular data, time series, and matrix data. Pandas is built on top of the NumPy library and is designed to be fast and efficient.

The main data structures provided by Pandas are the Series and DataFrame. A Series is a one-dimensional array-like object that can hold any data type, while a DataFrame is a two-dimensional table of data with rows and columns. DataFrames are particularly useful for working with tabular data, as they allow for easy indexing, grouping, and aggregation.

Some of the key features of Pandas include:

1. Data reading and writing: Pandas can read and write data from a wide range of formats, including CSV, Excel, SQL databases, and JSON.

2. Data manipulation: Pandas provides a wide range of functions for manipulating and cleaning data, including filtering, grouping, merging, and reshaping.

3. Data visualization: Pandas integrates with popular visualization libraries like Matplotlib and Seaborn to provide powerful visualization tools for exploring and understanding data.

4. Missing data handling: Pandas provides tools for handling missing data, such as filling in missing values or dropping rows or columns with missing data.

5. Time series analysis: Pandas provides tools for working with time series data, including date and time parsing, resampling, and shifting.

Overall, Pandas is a versatile and powerful library that is widely used in data analysis and machine learning workflows.

5.9.4 PICKLE

Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script.

Pickle is a Python module that is used for serializing and de-serializing Python objects. It allows Python objects to be easily saved to a file and later loaded back into memory, preserving their original state. Pickle can be used to store any Python object in a file, such as lists, dictionaries, functions, classes, and instances of classes.

Pickle works by converting a Python object hierarchy into a byte stream, which can then be written to a file or transmitted over a network. The byte stream can be reassembled into an equivalent copy of the original object hierarchy by using the de-serialization process.

Pickle has two main methods: `dump()` and `load()`. The `dump()` method serializes a Python object hierarchy and writes it to a file, while the `load()` method reads a byte stream from a file and de-serializes it into a Python object hierarchy

PYTHON FLASK

Flask Tutorial provides the basic and advanced concepts of the Python Flask framework. Our Flask tutorial is designed for beginners and professionals. Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by Armin Ronacher who leads an international group of python enthusiasts (POCCO).Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by Armin Ronacher who leads an international group of python enthusiasts .

Flask is a lightweight and flexible Python web framework that allows developers to quickly and easily build web applications. Flask is a micro-framework, meaning that it provides only the essential features required for building web applications, while allowing developers to choose their own tools and libraries to add additional functionality as needed.

Some of the key features of Flask include:

1. Routing: Flask provides a simple way to map URLs to view functions, allowing developers to easily define the routes for their web application.

2. Templating: Flask includes a templating engine, Jinja2, which allows developers to separate the presentation of their web application from the application logic.

3. Request handling: Flask provides a Request object that contains all of the information about an incoming HTTP request.

4. Integration with third-party libraries: Flask can easily be integrated with a wide range of third-party libraries and tools, allowing developers to add additional functionality to their web application as needed.

5. Debugging and testing: Flask provides tools for debugging and testing web applications, making it easy to identify and fix issues during development.

Flask is often used for building small to medium-sized web applications, Restful APIs, and prototyping. It is a popular choice for developers who value simplicity, flexibility, and the ability to choose their own tools and libraries.

5.9.5 NUMPY

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high- level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant createdNumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

5.9.6 MATPLOTLIB

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, python, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

5.9.7 SEABORN

Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and color palettes to make statisticalplots more attractive. It is built on the top of matplotlib library and also closelyintegrated to the data structures from pandas. Seaborn aims to make visualization the central part of exploring and understanding data. It provides dataset-oriented APIs, so that switch between different visual representations for same variables for better understanding of dataset.

Seaborn is a popular Python data visualization library based on Matplotlib. It provides a high-level interface for creating visually appealing and informative statistical graphics. Seaborn is built on top of Matplotlib and offers additional functionality and aesthetic enhancements to create visually stunning plots.

5.9.8 HARDWARE REQUIREMENTS

PERSONAL COMPUTER / LAPTOP

Fig.5.9 Personal computer

The minimum requirement of personal computer or laptop that have Intel i3 or AMD or better processor and virtualization is supported. That have either windows or linux operating system. Minimum 4GB of RAM required. And storage space needed for this face mask detector is minimum 2GB required.

CHAPTER 6

RESULTS AND DISCUSSIONS

6.1 RESULT

SGD based methods for such tasks, particularly designed to capture implicit features that are potentially useful for classification. Finally, methods were thoroughly evaluated on the largest collection of Twitter datasets for hate speech, to show that they can be particularly effective ondetecting and classifying hateful content (as opposed to non-hate), which have shown to be more challenging and arguably more important in practice. The results set a new benchmarking reference in this area of research.

The very challenging nature of identifying hate speech from short text such as tweets is due to the fact that hate tweets are found in the long tail of a dataset due to their lack of unique, discriminative features.

Further experiments showed in experiments that for this very reason, the practice of 'micro-averaging' over both hate and non-hate classes in a dataset adopted for reporting results by most previous work can be questionable. Itcan significantly bias the evaluation towards the dominant non-hate class in a dataset, overshadowing a method's ability to identify real hate speech.

6.2 OUTPUT OF AN APPLICATION

HATE SPEECH : -

```
text="kill you, idiot"
```

```
prediction = model.predict([text])
if prediction[0] == 1:
    print("Hate Speech")
else:
    print("Not hate speech")
```

```
Hate Speech
```

Fig.6.2.1 OUTPUT FOR HATE SPEECH

NON HATE SPEECH : -

```
In [16]: text = "kill you buddy"
```

```
In [17]: prediction = model.predict([text])
         if prediction[0] == 1:
             print("Hate Speech")
         else:
             print("Not hate speech")
```

```
Not hate speech
```

Fig.6.2.2 OUTPUT FOR NON-HATE SPEECH

6.3 PERFORMANCE METRICS

EXISTING SYSTEM

ALGORITHM	PRECISION	RECALL	F1 SCORE
LSTM	0.93	0.95	0.94
CNN	0.90	0.92	0.91
BI-DIRECTIONAL LSTM	0.95	0.96	0.95

Table 6.3.1 EXISTING SYSTEM

PROPOSED SYSTEM

ALGORITHM	PRECISION	RECALL	F1 SCORE
STOCHASTIC GRADIENT DESCENT	0.98	0.99	0.98

Table 6.3.2 PROPOSED SYSTEM

6.4 PERFORMANCE ANALYSIS

6.4.1 TRAINING ACCURACY VS TEST ACCURACY

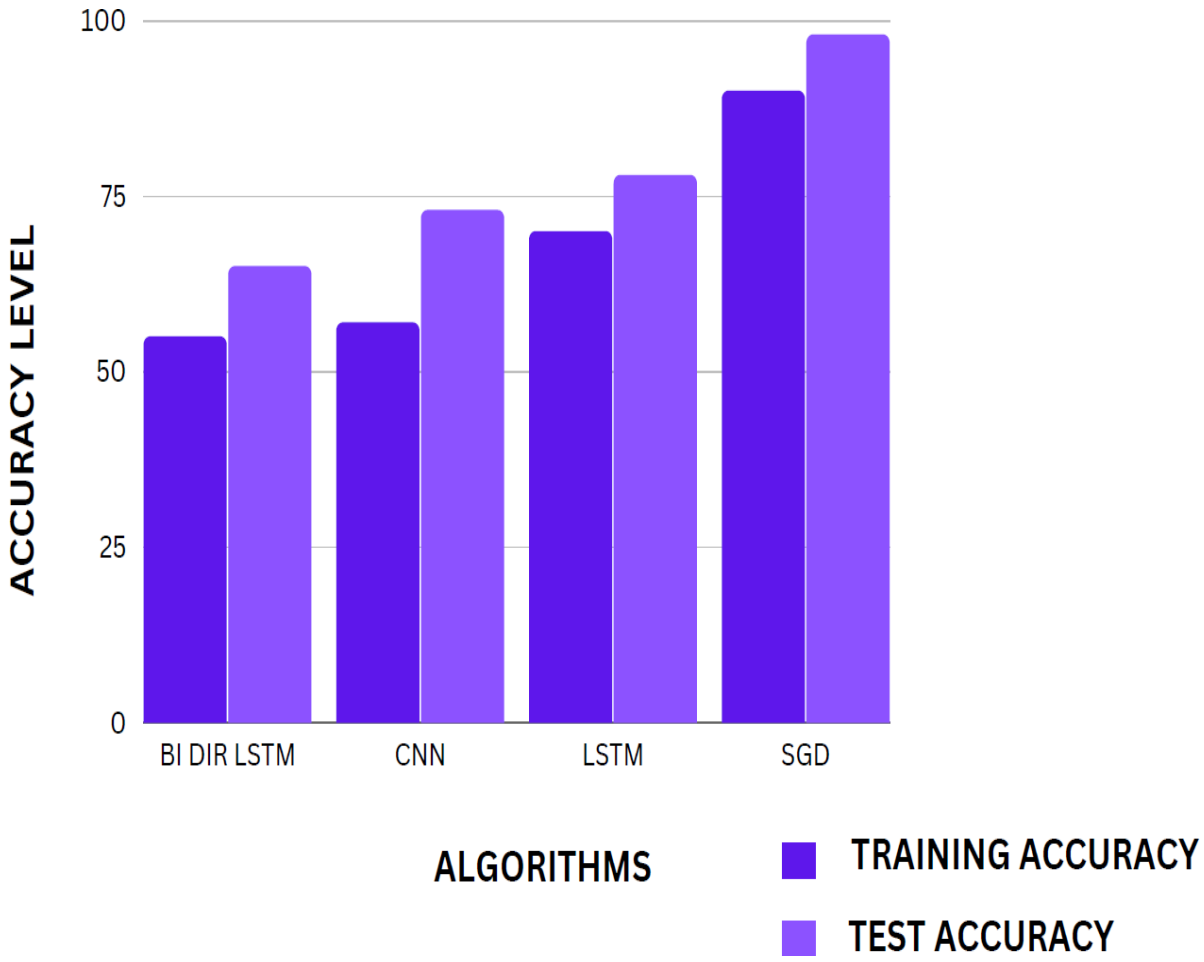

Fig 6.4.1 TRAINING ACCURACY VS TEST ACCURACY

6.4.2 COMPARATIVE ANALYSIS ON ALGORITHMS

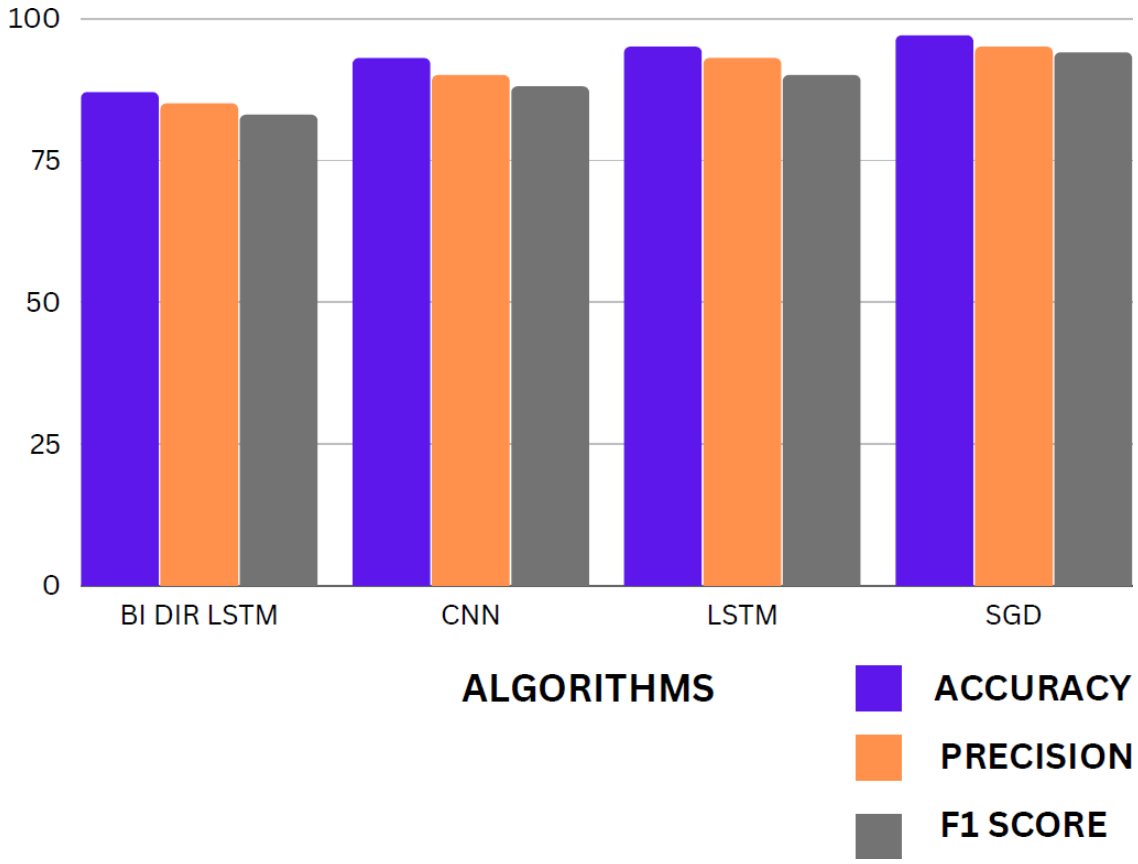

Fig. 6.4.2 COMPARATIVE ANALYSIS ON ALGORITHMS

6.4.3 ACCURACY AND LOSS ANALYSIS

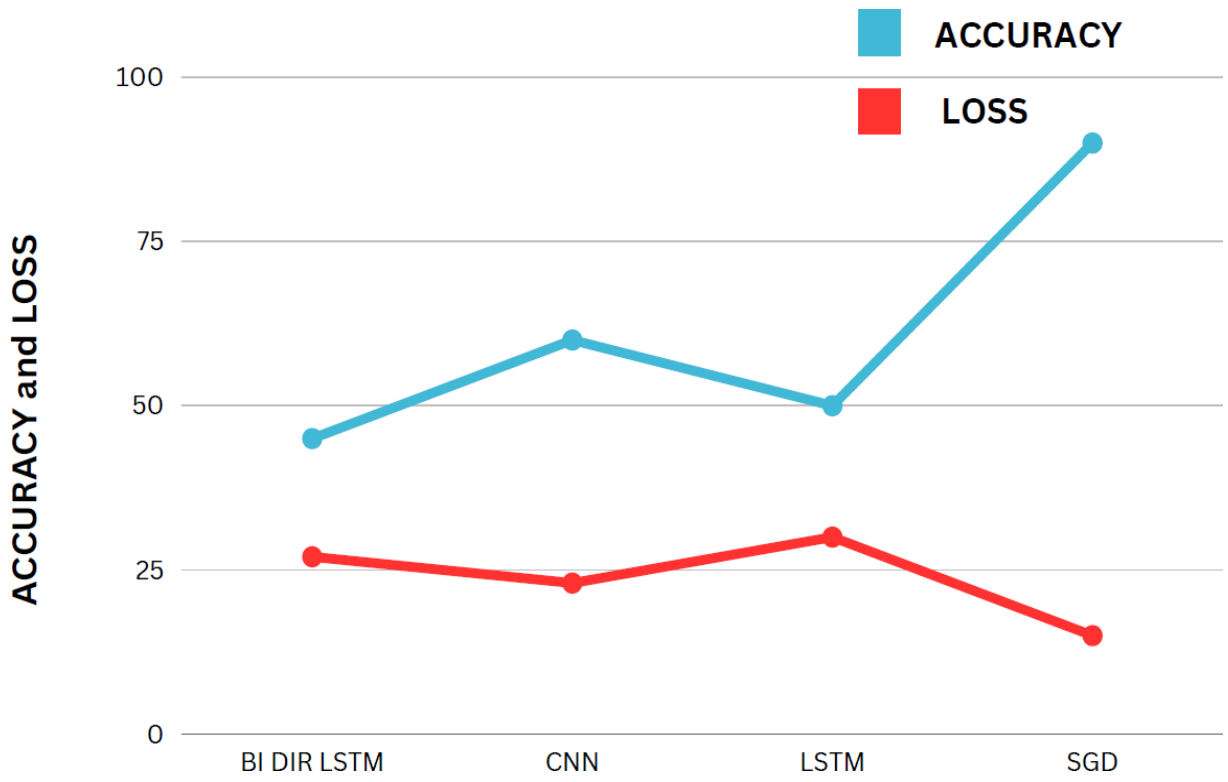

Fig 6.4.3 ACCURACY AND LOSS ANALYSIS

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

As hate speech continues to be a societal problem, the need for automatic hate speech detection systems becomes more apparent. The current approaches for this task as well as a new system that achieves reasonable accuracy. A new approach that can outperform existing systems at this task, with the added benefit of improved interpretability.

7.2 FUTURE WORK

The dataset is collected from Kaggle and using those dataset, train the models. The work accuracy of the model is increased usingstochastic gradient descent. The existing system has an accuracy of 93% and the proposed system provides an accuracy of 98% accuracy.

The future work is to make real time predictions of hate speech in social media. The model could be trained with more dataset and the model can have even higher accuracy.

REFERENCES

1.P. K. Jain, V. Saravanan, and R. Pamula, ‘‘A hybrid CNN-LSTM: A deep learning approach for consumer sentiment analysis using qualitative user generated contents,’’ ACM Trans. Asian Low-Resource Lang. Inf. Process., vol. 20, no. 5, pp. 1–15, Sep. 2022.

2.A. Kamaal and M. Abulaish, ‘‘CAT-BiGRU: Convolution and attention with bidirectional gated recurrent units for self-deprecating sarcasm detection,’’ Cognit. Compute., vol. 14, pp. 91–109, Jan. 2022.

3.P. K. Jain, R. Pamula, and E. A. Yeung, ‘‘A multi-label ensemble predicting model to service recommendation from social media contents,’’ J. Supercomputer., E. W. Pamungkas, V. Basile, and V. Patti, ‘‘A joint learning approach with knowledge injection for zero-shot cross-lingual hate speech detection,’’ Inf.Process. Manage., vol. 58, no. 4, pp. 1–19, 2021.

4. Ksss. Miok, B. Skrlj, D. Zaharie, and M. Robnik-Sikonja, ‘‘To ban or not to ban: Bayesian attention networks for reliable hate speech detection,’’ Cognit. Comput., vol. 21, no. 1, pp. 1–19, 2021.

5.S. Kravchenko, I. Negaholism, and K. C. Fraser, ‘‘Confronting abusive language online: A survey from the ethical and human rights perspective,’’ J. Arif. Intel.Res., vol. 71, pp. 431–478, Jul. 2021.

6. Z. Mossie and J.-H. Wang, ``Vulnerable community identi_cation using hate speech detection on social media," Inf. Process. Manage., vol. 57,no. 3, pp. 1_16, 2020.

7. P. Fortuna, J. Soler-Company, and L. Wanner, ``How well do hate speech, toxicity, abusive and offensive language classi_cation models generalize across datasets?" vol. 58, no. 3, pp. 1_17, Oct. 2021.

8.G. E. Hinton, S. Sabour, and N. Frosst, ``Matrix capsules with em routing," in Proc. Int. Conf. Learn. Represent. (ICLR), Vancouver, BC Canada, Apr/May 2022, pp. 44_51.

9. D. K. Jain, R. Jain, Y. Upadhyay, A. Kathuria, and X. Lan, ``Deep re_nement: Capsule network with attention mechanism-based system fortext classi_cation," Neural Comput. Appl., vol. 32, no. 7, pp. 1839_1856,Apr. 2022.

10. U. Bhattacharjee, ``Capsule network on social media text: An application to automatic detection of clickbaits," in Proc. 11th Int. Conf. Commun. Syst. Netw. (COMSNETS), Bengaluru, India, Jan. 2022, pp. 473_476.

11.Y. Du, X. Zhao, M. He, and W. Guo, ``A novel capsule based hybridneural network for sentiment classi_cation," IEEE Access, vol. 7,pp. 39321_39328, 2022.

12. Y. Ding, X. Zhou, and X. Zhang, ``YNU_DYX at SemEval-2019 task 5:A stacked BiGRU model based on capsule network in detection of hate," in Proc. 13th Int.Workshop Semantic Eval., Minneapolis, MN, USA, 2022, pp. 535_539.

13. W. Warner and J. Hirschberg, ``Detecting hate speech on the world wide web," in Proc. Workshop Lang. Social Media. Montreal, QC, Canada,2021, pp. 19_26.

14. I. Kwok and Y.Wang, ``Locate the hate: Detecting tweets against blacks," in Proc. 27th AAAI Conf. Artif. Intell., Bellevue, WA, USA, 2022pp. 1621_1622.

15. Z. Waseem and D. Hovy, ``Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter," in Proc. NAACL-HLT, Sacramento, CA, USA, 2022. pp. 88_93.