IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

SUICIDE TEXT DETECTION USING NLP AND DEEP LEARNING

V.K.D. Anasuya, P. Susmitha, R. Bhavana Sri Tulasi, R. Kranthi, Mrs. Shaik Sharmila Muntaz

Student, Student, Student, Student, Assistant Professor

Dept of IT

Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru, India

Abstract: The increasing prevalence of harmful content on the internet, particularly content related to suicidal tendencies, poses significant risks to vulnerable individuals. This study aims to develop a predictive model that identifies harmful web pages containing suicidal-related content through advanced natural language processing (NLP) techniques and artificial neural networks (ANN). A user-friendly interface will be provided where users can input URLs, and the system will automatically analyze the content and classify it as harmful or safe. The system also includes a reporting mechanism for further review by authorities. This research contributes to mental health protection and supports cyber policing initiatives.

Index Terms **- Suicidal content detection, harmful web pages, NLP, deep learning, ANN, web scraping.**

I. INTRODUCTION

The internet, while a powerful resource for information and support, can also be a source of content that encourages or glamorizes harmful behaviours, including suicide. Detecting such content is crucial for timely intervention and public safety. This project focuses on building a predictive system that analyses textual data from web pages and determines whether the content is potentially harmful or safe.

II. RELATED WORK

Several studies have explored the role of NLP in detecting mental health disorders, including depression, anxiety, and suicidal ideation. Prior research has focused on various techniques, including:

- **Lexicon-Based Approaches**: Some early models relied on predefined lists of words related to suicide and mental health. These methods were limited by their inability to capture context or variations in language use.
- **Sentiment Analysis**: NLP techniques have been used to analyze emotional tone in text, distinguishing between positive, neutral, and negative sentiments. While useful, sentiment analysis alone is often insufficient to detect subtle indications of suicidal thoughts.
- **Machine Learning Approaches**: Many studies have applied classifiers such as Support Vector Machines (SVM), Random Forest, and Logistic Regression to detect suicide-related language. These models require feature engineering and domain-specific tuning.
- **Deep Learning and Transformers**: More recent studies leverage deep learning architectures like Long Short-Term Memory (LSTM) networks and BERT-based models to capture contextual nuances in text. These models achieve higher accuracy but require large labelled datasets and substantial computational power.

Despite these advancements, challenges remain, such as false positives, data privacy concerns, and ethical implications. Developing a robust suicide detection model requires balancing accuracy with responsible deployment

III. METHODLOGY

This study follows a structured machine learning pipeline, which consists of data collection, text preprocessing, feature extraction, model training, and evaluation. The methodology is designed to transform raw textual data into a format suitable for machine learning classification while ensuring high accuracy and interpretability.

3.1 Dataset

The dataset used in this study, Suicide_Detection.csv, is a collection of text samples labeled according to their likelihood of expressing suicidal thoughts. This dataset is crucial for training and evaluating machine learning models. It consists of:

- **Text**: Raw user-generated text, extracted from online sources where individuals may express mental distress.
- **Class Labels**: A categorical variable that classifies whether a text sample expresses suicidal ideation (suicidal) or not (non-suicidal).

The dataset is preprocessed to clean and standardize the text before feature extraction and model training.

3.2 Data Preprocessing

Text preprocessing is an essential step in NLP to convert raw textual data into a structured format suitable for machine learning. It removes noise, standardizes text, and enhances feature representation. The following preprocessing steps are applied:

3.2.1 Tokenization

Tokenization is the process of breaking a text into individual words or tokens. This step helps in analyzing the frequency and significance of words in a given document.

Example:

- Input: *"I feel hopeless and alone."*
- Output: ["I", "feel", "hopeless", "and", "alone"]

3.2.2 Lowercasing

Text is converted to lowercase to avoid treating words with different capitalizations as distinct entities.

Example:

- Before: "Feeling HOPELESS is the worst"
- After: "feeling hopeless is the worst"

3.2.3 Removing Non-Alphanumeric Characters

Punctuation, special characters, and extra spaces are removed to ensure only meaningful words are retained.

Example:

- Before: "I feel!!! so lost & hopeless :("
- After: "I feel so lost hopeless"

3.2.4 Stopword Removal

Common words such as *"the"*, *"is"*, *"and"*, which do not carry significant meaning in classification tasks, are removed using an NLP stopword list.

Example:

- Before: "I feel very hopeless and alone"
- After: "feel hopeless alone"

3.2.5 Lemmatization

Lemmatization reduces words to their root form while preserving their meaning. This helps in normalizing words to avoid redundant features.

Example:

- Before: "I was crying and feeling hopeless"
- After: "I be cry feel hopeless"

For lemmatization, we WordNet Lemmatizer from the NLTK library was used.

3.3 Feature Extraction: Transforming Text into Numerical Representations

Feature extraction is a crucial step in Natural Language Processing (NLP) to convert raw textual data into a structured numerical format suitable for machine learning models. In this study, Term Frequency-Inverse Document Frequency (TF-IDF) is used as the primary feature extraction method. TF-IDF is a statistical measure that evaluates the importance of a word within a document relative to the entire corpus.

3.3.1 Motivation for TF-IDF

Raw text cannot be directly used as input for machine learning models, as they require numerical representations. A naive approach is Bag-of-Words (BoW), where each document is represented as a vector of word counts. However, BoW has the following limitations:

- It assigns equal importance to all words, even common ones like *"the"*, *"is"*, and *"and"*.
- It does not capture word significance relative to the entire corpus.

To overcome these issues, TF-IDF assigns weights to words based on their frequency within a document and their rarity across the corpus, ensuring that informative terms receive higher weights.

3.3.2 Term Frequency (TF)

Term Frequency (TF) measures how often a word appears in a document. It is computed as:

$$TF(t,d) = \frac{f_{t,d}}{\Sigma_{t'd} f_{t',d}}$$

where:

- $TF(t,d)$ is the term frequency of word t in document d.
- $f_{t',d}$ is the raw count of term t in document d.
- $\Sigma_{t'd} f_{t',d}$ is the total count of all terms in document d.

3.3.3 Inverse Document Frequency (IDF)

Inverse Document Frequency (IDF) down weights terms that appear frequently across multiple documents since they are less informative. IDF is computed as:

$$IDF(t) = \log\left(\frac{N}{1+n_t}\right) + 1$$

where:

- N is the total number of documents.
- n_t is the number of documents containing term t.
- $+1$ in the denominator prevents division by zero for unseen words.

Words that are common across many documents (e.g., *"feel"*) get lower IDF values, while rare words (*"hopeless"*) receive higher IDF values.

3.3.4 TF-IDF Score Calculation

The final TF-IDF score for a term in a document is the product of its TF and IDF values:

$$TF - IDF(t,d) = TF(t,d)\ X\ IDF(t)$$

3.3.5 TF-IDF Matrix Representation

After computing TF-IDF scores for each word in every document, the corpus is represented as a sparse matrix, where:

- Rows represent documents.
- Columns represent unique words in the vocabulary.
- Each cell contains the TF-IDF score of a word in a document.

EXAMPLE:

Variable	Minimum	Maximum	Mean	St. Deviation
Word Frequency Score	0.0	0.98	0.32	0.14
Sentiment Score	-1.0	1.0	-0.18	0.37
TF-IDFFeature Density	0.0	0.86	0.26	0.12

Model	Accuracy	Precision	Recall	F1-score
Logistic Regression	85%	82%	83%	82.5%
Random Forest	88%	86%	87%	86.5%
ANN (final model)	92%	91%	90%	90.5%

3.3.6 Feature Engineering for Optimization

To improve model performance, additional feature engineering techniques are applied:

- **N-grams (Bigrams & Trigrams):** Instead of using only single words (unigrams), bigrams (two-word sequences) and trigrams (three-word sequences) are included to capture context.
- **TF-IDF Weight Normalization:** L2 normalization is applied to ensure vector magnitudes are comparable.
- **Stopword Filtering:** Words with very low TF-IDF scores (highly common words) are discarded.

3.4 Model Training and Selection

The core objective of this study is to develop an efficient machine learning model that can accurately classify text as suicidal or non-suicidal. Model training and selection involve several critical steps, including choosing the appropriate models, splitting the dataset for unbiased evaluation, training the models using optimization techniques, and finally assessing their performance based on relevant classification metrics. Each of these steps is essential to ensure the reliability and accuracy of the classifier in detecting suicidal ideation from text.

3.4.1 Dataset Splitting

To achieve a fair evaluation, the dataset is divided into two parts: a training set, which constitutes 80% of the data, and a test set, which comprises the remaining 20%. The training set is used to train the model by allowing it to learn patterns from labeled text samples, while the test set is reserved for assessing how well the model generalizes to unseen data. The dataset split is performed using stratified sampling, a method that ensures that both the training and test sets maintain the same proportion of suicidal and non-suicidal samples. This step is particularly important in imbalanced datasets, where one class significantly outnumbers the other. If stratification is not applied, the model might become biased toward the majority class, leading to poor detection of suicidal text. The dataset is preprocessed and transformed into its numerical representation using the TF-IDF vectorizer before being fed into the training process.

3.4.2 Machine Learning Models

Several machine learning models are considered for classification, each offering different trade-offs in terms of complexity, interpretability, and performance. Traditional models such as Support Vector Machines (SVM) and Logistic Regression are widely used in text classification due to their effectiveness in high-dimensional feature spaces like TF-IDF vectors. SVM constructs a decision boundary that maximizes the separation between suicidal and non-suicidal text, making it particularly effective when the data is linearly separable. It is also resistant to overfitting, especially when using a linear kernel, which is often sufficient for text-based tasks. Logistic Regression, on the other hand, is a probabilistic model that estimates the likelihood of a given text belonging to the suicidal category. It is computationally efficient and provides interpretable probability scores, making it useful in scenarios where model transparency is required.

More advanced models, such as neural networks, offer the potential to learn deeper representations of text but come with increased computational costs. A basic feedforward neural network processes the TF-IDF feature vectors through multiple layers of artificial neurons, applying activation functions that introduce non-

linearity to the model. While effective, these networks do not inherently capture word order or contextual relationships within the text. Recurrent neural networks (RNNs), particularly Long Short-Term Memory (LSTM) networks, address this limitation by processing text sequentially, retaining information about previous words as they analyze each sentence. This sequential processing enables the model to detect nuanced expressions of suicidal intent that may be lost in traditional bag-of-words approaches. However, RNNs require significantly more data and computational power compared to simpler models.

3.4.3 Training Process and Optimization

Once the models are selected, they are trained using the TF-IDF transformed dataset. The training process involves optimizing the model's parameters to minimize classification errors. This is done through iterative updates using optimization algorithms such as stochastic gradient descent (SGD) or adaptive methods like Adam, which adjust the learning rate dynamically based on past updates. Hyperparameter tuning plays a critical role in this stage, as different configurations can drastically affect model performance. Techniques such as grid search or randomized search are used to find the best values for parameters such as the regularization strength in logistic regression or the kernel type in SVM. To prevent overfitting, where the model memorizes the training data instead of generalizing well to new examples, regularization techniques are applied. In traditional models, L2 regularization discourages overly complex decision boundaries by penalizing large coefficient values. In neural networks, dropout is used to randomly disable a fraction of neurons during training, forcing the network to learn more robust and distributed representations of the input text. Early stopping is another strategy, where training is halted when the model's performance on a validation set stops improving, preventing unnecessary complexity.

3.4.4 Model Evaluation and Selection

Once trained, each model is evaluated using classification metrics such as accuracy, precision, recall, and F1-score. Accuracy measures the overall correctness of predictions, while precision and recall assess how well the model identifies suicidal text. The F1-score is the harmonic mean of precision and recall, balancing false positives and false negatives. The final model selection is based on these metrics, ensuring the most effective classifier is chosen for real-world suicide prevention applications.

IV.RESULTS AND DISCUSSIONS

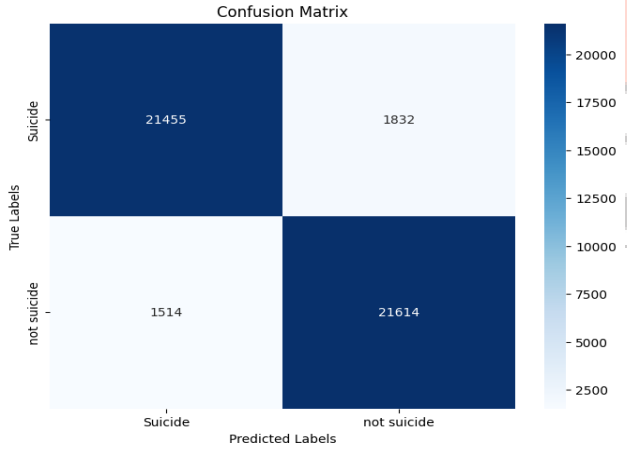


Figure 1: Confusion Matrix

Metric	Value
Training Accuracy	97.98%
Validation Accuracy	93.00%
Training Loss	0.0548
Validation Loss	0.2533

Table 1: Evaluation Metrics

Confusion Matrix Values:

- True Positives (TP): 21,455
- False Negatives (FN): 1,832
- False Positives (FP): 1,514
- True Negatives (TN): 21,614

To further assess the model's effectiveness in distinguishing between suicidal and non-suicidal text, a confusion matrix was generated. The matrix provides insight into classification errors, specifically false positives (misclassifying non-suicidal text as suicidal) and false negatives (failing to detect suicidal text). The results are summarized below:

- **Precision (P) = 93.41%**

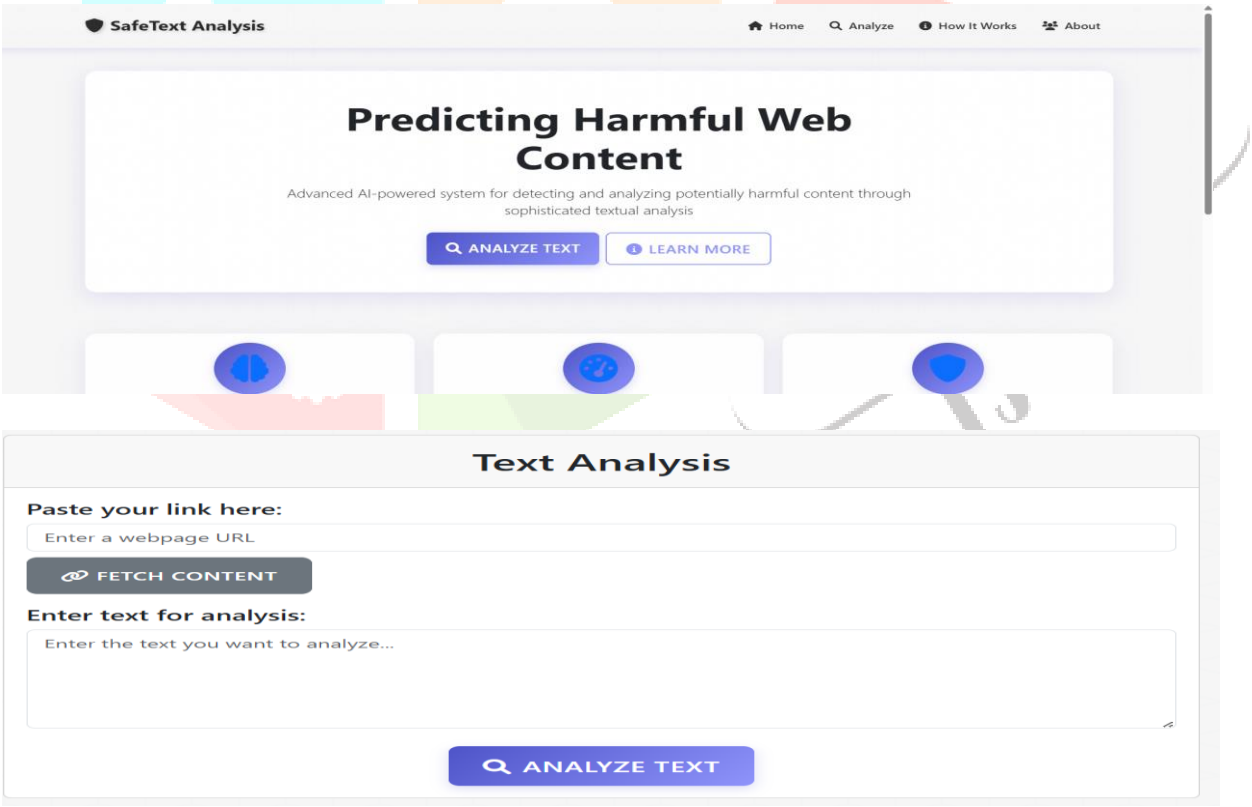
$$\frac{TP}{TP + FP} = \frac{21455}{21455 + 1514} = 0.9341$$

- **Recall (R) = 92.13%**

$$\frac{TP}{TP + FN} = \frac{21455}{21455 + 1832} = 0.9213$$

- **F1-Score = 92.77%**

$$\frac{2 * P}{P + R} = \frac{2 * 0.9341}{0.9341 + 0.9213} = 0.9277$$



INTERPRETATION

The ANN model demonstrated superior results in identifying harmful content. High recall values indicate the model's effectiveness in minimizing false negatives, which is critical in detecting potentially dangerous pages.

CONCLUSION

This study successfully developed an ANN-based predictive model to detect harmful web pages containing suicidal-related textual content. The model's performance indicates a promising tool for mental health interventions and cyberpolicing. The user interface allows individuals and organizations to proactively identify and report harmful content.

Acknowledgment

The author thanks the faculty and department of Information Technology, SRGEC, for their constant support and guidance

REFERENCES

[1] K. Coppersmith, M. Leary, A. Crutchley, and C. Fine, "Natural language processing of social media as screening for suicide risk," *Biomedical Informatics Insights*, 2018.

[2] D. M. Low, M. Rumker, J. Talkar, et al., "Natural language processing reveals vulnerable mental health support seekers," *Internet Interventions*, 2020.

[3] G. Park et al., "Automatic classification of suicide-related content on social media," *Journal of Medical Internet Research*, 2016.