RESEARCH PAPER

Enhancing Hate Speech Detection Through TF-IDF and Optimized Gradient Boosting

Pragya Goswami[1*], Kapil Sharma[2], Devendra Kumar Mishra[3]

[1,2,3] *Department of Computer Science and Engineering, Amity University Madhya Pradesh, Gwalior, MP, India*

Emails: pragyagoswami2729@gmail.com, ksharma@gwa.amity.edu, dkmishra@gwa.amity.edu

Corresponding author: Pragya Goswami, pragyagoswami2729@gmail.com

ABSTRACT

The fast expansion of user-created material on social networking sites made it much easy for hate speech and harsh language to spread, which makes it much harder to keep people safe online, be polite, and moderate information. Conventional manual moderation methods lack scalability and efficiency, hence requiring the creation of resilient automated solutions. In this research, we introduce an innovative and a proposed machine learning framework that combines the representation of features in terms of Term Frequency Inverse Document Frequency (TFIDF) and optimized Gradient Boosting Classifier (GBC) proposed model with its accuracy of 94.9%, precision 94.9%, recall 92.9%, and the F1 score of 93.9% delivers competitive performance relative to the previously reported deep learning methods, and needs lower computing resources. The model under discussion is evaluated on the hate speech dataset which consists of thousands of. annotated tweets classified as either hate(0), offensive(1), or neither(2). These findings indicate that the TF-IDF and Gradient Boosting combination is a method that is lightweight, intuitively explainable and appr approachable and can be incorporated into the production of social networking sites in real time as a way to detect hate speech.

Keywords: Hate speech detection, machine learning, TF-IDF, gradient boosting classifier, grid search, cross-validation, deep learning, LSTM, CNN

How to cite this article: Goswami P, Sharma K, Mishra DK., Enhancing Hate Speech Detection Through TF-IDF and Optimized Gradient Boosting. Int J Drug Deliv Technol. 2026;16(78s): 862-872; DOI: 10.25258/ijddt.16.78s.91

1. Introduction

Hate speech messages in are humiliating or intimidating to people because of such identity traits as ethnic group, religion, sex, or sexual orientation. Its spreading via media, such as social media, forums and news media creates a sense of division, violence and creates an atmosphere of fear and marginalization(Vijay & Verma, 2023).

Hate speech are significant challenges in the digital age. It can undermine the foundation of democracy by distorting public opinion and undermining trust in institutions. Moreover, it can cause harm to individuals and groups by promoting hate and discrimination. Effectively detecting and reducing hate speech is difficult a prominent task that requires a multidisciplinary approach, including ML techniques uses, Natural Language Processing (NLP), and social science research. Hate speech detection on the Internet is a very significant yet challenging endeavor to industry and academia. The area of hate speech, which can be detected with the help of ML, is extensive, and the possible use cases are numerous. With the issue of hate speech only expanding, ML will probably become more significant in the context of assisting to detect and prevent the proliferation of hate speech. (Mercan et al., 2021).

The document is structured as follows Section 2 provides literature survey, Section 3 outlines the proposed approach, involving data pre-processing, TF-IDF feature extraction and development of the tuned Gradient Boosting Classifier (GBC) model with Grid Search Cross Validation(CV) for hyperparameter tuning, Section 4 describes the results and discussion, Section 5 presents the conclusion of the proposed GBC and Section 6 provides future work.

2. Related Work

The review methods of detecting hate speech on the online platforms and it will focus more on Twitter. A common approach that is touted takes the approach of unsupervised learning wherein the context of an image as well as its visual properties can be used to evaluate sentiments in large scales of data.

(Goswami & Daniel, 2026a)proposed a hybrid model integrating DistilBERT with a Bidirectional Long Short-Term Memory (BiLSTM) network to enhance contextual understanding and sequential dependency learning. The model employed Bayesian optimization for hyperparameter tuning, achieving an accuracy of 95%, precision of 93%, recall of 95%, and an F1-score of 94% on the Jigsaw Toxic Comment Classification Dataset.

In another study, (Goswami & Daniel, 2026b) developed an optimized BiLSTM-based approach utilizing FastText for feature extraction. To improvise the data quality, their method included important preprocessing steps like tokenization, normalizing and stopword elimination. The model is resistant to misspellings and colloquial language frequently encountered in social media since the FastText embeddings successfully gathered subword-level information. text. (Pitenis et al., 2020a) compared linear classifiers (SVM and SGDC) with LSTM and GRU models, finding that while linear classifiers show improved performance compared to others, the best results were achieved with LSTM and GRU with attention models. They recommended further evaluation of computational models on this dataset.

Lastly,(Que et al., 2020) employed BERT-Ger with bi-GRU and Bi-LSTM Although the direction succeeded in finding hate speech and offensive materials in the German language and showed better extraction of semantic features, the BERT-Ger model failed to reach the promise of deep semantic information retrieval completely.

*Author for Correspondence: pragyagoswami2729@gmail.com; ksharma@gwa.amity.edu; dkmishra@gwa.amity.edu

(de Gibert et al., 2018), implemented CNNs to analyze a dataset from a white supremacy forum, revealing the distribution of classes in the test set. They proposed that subsequent research could investigate the influence of relationally labeled sentences on classification.

(Ribeiro et al., 2018), used a node embedding method to find and describe hateful users on Twitter, and they suggested that a user-centered approach would make detection tools better. But they stressed the need for better ways to find these kinds of users.

(Davidson et al., 2017), applied LR and linear SVM to hate speech detection, noting that these models significantly achieve higher performance than others, especially in identifying racially or homophobic tweets. Nonetheless, the challenge of classifying tweets lacking explicit hate keywords remained.

(Gao & Huang, 2017) Used together with logistic regression using neural models of prediction with ensemble models combining these approaches achieving the best performance. Their work underscored the complexity of addressing hateful speech within full discussion threads.

(Jha & Mamidi, 2017)focused on ambivalent sexism, analyzing Twitter data using SVM and achieving the best F1-score with a fast text classifier. They called for a deeper understanding of ambivalent sexism prevalent on social media.

Overall, these studies show that hate speech detection has come a long way thanks to a variety of methods and the use of socio-linguistic factors. However, they also show that there are still problems that need to be solved, such as the need for more detailed language feature classification and stronger detection systems. Table 1 dipicts a literature review.

Table 1: Literature Review

Ref.	Methodology	Findings	Limitations
(Umansky & others, 2026)	An integrated approach to detecting hate speech across text, speech and visual data.	The fine-tuned large language models and the expert labeled training	High computational complexity in multimodal processing.
(Kevin et al., 2025)	Proposed CNN-RNN with attention for cross-platform hate speech detection.	Automated content moderation of correctly identifying harmful content online.	Latency in real-time applications
(Ramos et al., 2024)	Compared BERT, RoBERTa, and XLNet	RoBERTa showed better performance than BERT and XLNet, with consistent results	Resource-intensive models
(Feng et al., 2022)	Trained models with expert-annotated data.	Performance comparable to previous efforts; recommended weighted F1-score for evaluation.	Calls for deeper exploration of socio-linguistic features like gender and location.
(Ayo et al., 2021)	Compared linear models (SVM, SGD Classifier) with LSTM & GRU attention.	Best results from LSTM and GRU with attention; linear models also performed well.	Suggests deeper evaluation of computational models on given datasets.
(Pitenis et al., 2020b)	Used SVM and Fast Text classifiers on Twitter data.	Achieved best F1-score with Fast Text; highlighted ambivalent sexism prevalence.	Needs more research into ambivalent sexism.
(Que et al., 2020)	Utilized CNNs with a single input channel for text.	Baseline studies revealed skewed class distribution in the dataset.	Recommends studying the impact of relationally labeled sentences.
(Ayo et al., 2020)	Introduced node embedding techniques to detect hateful users.	Advocated for user-centric approaches in hate speech detection.	Needs more robust user-level detection mechanisms.
(Ribeiro et al., 2018)	Employed LR and neural network models; created ensemble approaches.	Ensemble methods combining both models yielded best results.	Addressing hate speech across complete threads remains a challenge.
(Davidson et al., 2017)	Used character n-grams (up to 4); included gender as a factor.	Established standards for identifying racist and sexist insults using critical race theory.	Requires improvement in gender and location classification in future datasets.

3. Data Collection and Preprocessing:

The sample used in the research in question consists of 24,783 tweets which can be classified as hate speech (1,430 instances), offensive language (19,190 instances), and neither (4,163 instances). It was done by employing the class-conscious splitting strategy to obtain the same distribution of classes in subsets. It divided the data into test, validation, and training sets 70:15:15.

Some training of the model was done after preprocessing steps were done. such as lowercasing, URL removal, mentions, special characters, and stop words, and tokenization was done. In order to achieve reproducibility of the experiments, data splitting and model initiation were done with a fixed random seed .The research utilizes a evaluation database of Hate Speech and Offensive Language, comprising thousands of annotated tweets sourced from Twitter. Every tweet is systematically categorized into one of the following classifications:

HATE (0): Includes hate speech directed at individuals or groups.

OFFENSIVE(1): Contains derogatory yet non-hateful language.

NEITHER(2): Content that is neutral or non-offensive.

The dataset provides a balanced distribution of genuine textual content for model training and assessment, accompanied by a comparative analysis of the previously discussed models.

3.1 Feature Representation and Analysis using TF-IDF

The text data is preprocessed by means of a TF-IDF vectorization process. It converts it into a numerical format that can be used for analysis. TF-IDF is used to assign lower importance to frequently occurring words in the text such as "the" and "is" and increase the importance of words that are significant. This involves a few stages like text cleaning, making the text lowercase, word breaking and removing punctuation and stop words. Once the text is preprocessed, the TF-IDF values for every word in the corpus are calculated. This process results in a sparse matrix of TF-IDF vectors. It is an effective method of encoding text data (such as tweets) so that they can be used in machine learning.

$$TF\text{-}IDF(r,n)=TF(r,n).\log(N/DF(r))....(1)$$

TF-IDF converts text to vectors by increasing the importance of significant words and reducing the influence of common words.

Where:

TF(r,n): Term of frequency

DF(r): Document of frequency

N: Total documents

This section presents the analysis and interpretation of the extracted features using the TF-IDF representation. The visualization demonstrates how textual data can be converted into numerical feature space, enabling the identification of important terms contributing to classification. Fig. 1 illustrates the TF-IDF

feature extraction and representation workflow, providing insights into how textual patterns are represented and utilized by the proposed model.

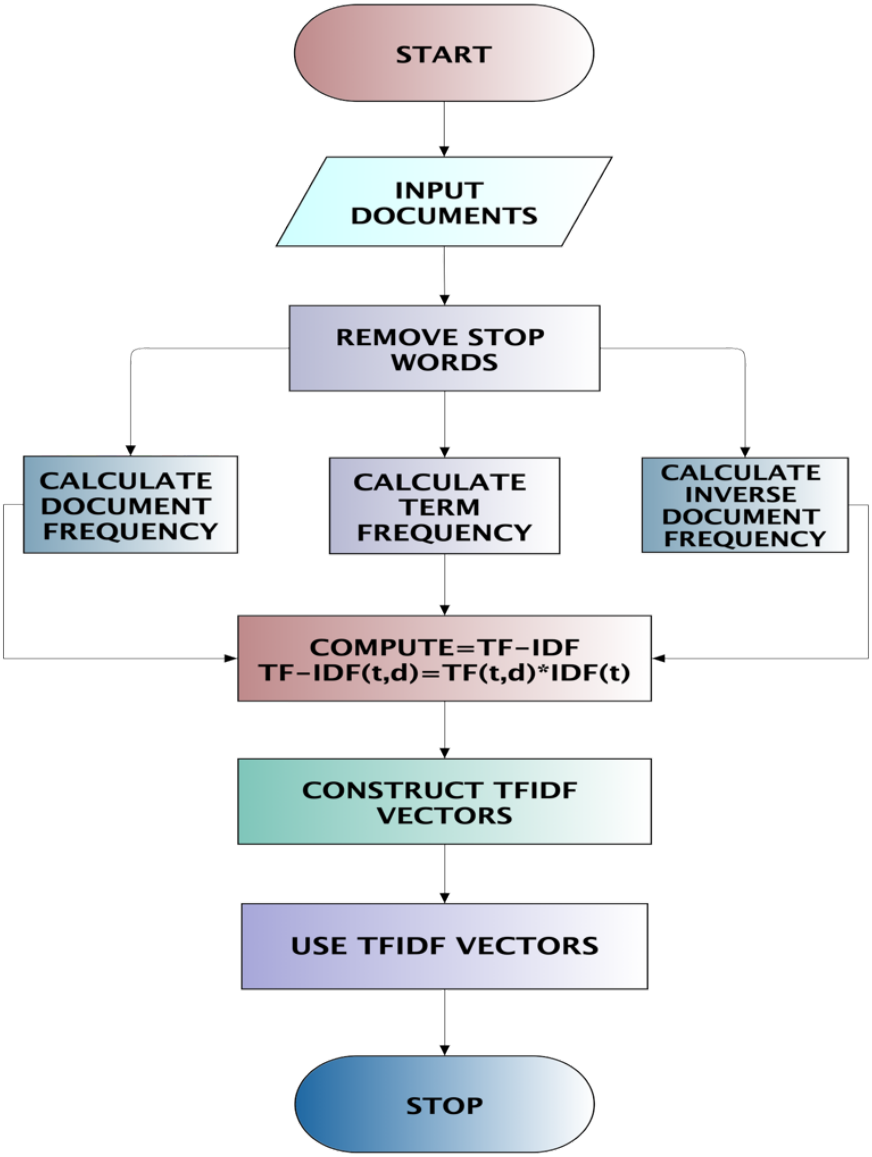

Figure 1: TF-IDF feature extraction and representation workflow

3.2 Hyperparameter Tuning (Selected 10 out of 21 Parameters)

Several hyperparameters that affect several parts of the learning process, including model complexity, optimization, and regularization, are provided by the ensemble module. Ten important hyperparameters that significantly affect the model's model performance and the capacity for generalization well were the focus of this study. Both experimental findings and insights from earlier research were used to choose these parameters. Due to their relatively minor the overall the model's performance is much improved. impacted by these other hyperparameters were left at their default settings. Table 2 illustrates Analytical Representation of Gradient Boosting Hyperparameters

Selected hyper parameters and descriptions:

1. n_estimators - It is decided how many boosting iterations, or decision trees, to use. The model can be more accurate if this value is higher., however, it can also contribute to the longer training time and over-fitting.
2. learning_rate- A parameter that reduces a tree output before it is added to a model. Lower values need more trees, but they are likely to have better generalization of the model.
3. max_depth - Each decision tree's maximum depth is set by it. Trees with deeper roots will be able to capture intricate patterns, but it will be at the risk of overfitting.
4. min_samples_split- Sets the least value sample required to divide any internal node. It plays a part such as a regularization parameter, hindering the growth of trees.
5. min_samples_leaf - Using this, it is possible to specify how many samples should appear in one of the leaf nodes. It leaves the leaf nodes well-populated so they would generalize.
6. subsample - The size of the chunk of training data selected randomly at random per tree. This can be set to less than 1.0 to create randomness hence minimizing on over fitting.
7. max_features - A limit to the number of features that are used in deciding the split that is best. The values that are lower foster tree diversity and less computational complexity.
8. loss - Refers to the loss that is being minimized when training a model (e.g. log_loss in a classification problem). This has a direct effect on the way the error of prediction is treated.
9. criterion- The operation that is applied to assess the quality of a split (e.g. fried-man_mse). It is important in the optimization of decision tree, particularly those tasks that involve both regression and classification.
10. random_state - Ensures consistency by establishing a set random seed at the beginning of each algorithmic stochastic process.

Table 2: Analytical Representation of Gradient Boosting Hyperparameters

S.No.	Parameter	Description	Mathematical Representation
1	n_estimators	No. of boosting stages (trees)	(F_M(a) = \sum_{m=1}^{M} \gamma_m h_m(a)), where (M = n_estimators)
2	learning_rate	Step size shrinkage for each tree	(F_m(a) = F_{m-1}(a) + \eta \cdot h_m(a)), where (\eta = learning_rate)
3	max_depth	Max depth of the tree	(depth(T) \leq d), where (d = max_depth)
4	min_samples_split	The minimal quantity of samples needed to execute a node split	(N_{node} \geq min_samples_split)
5	min_samples_leaf	Min samples required at leaf node	(N_{leaf} \geq min_samples_leaf)
6	subsample	proportion of samples utilized to train each tree	(N_{sample} = subsample \times N)
7	max_features	Number of features considered for best split	(F_{subset} \subseteq F, \
8	loss	Loss function to minimize	(L(y, F(x))) (e.g., Logistic Loss: (\log(1 + e^{-yF(x)})))
9	criterion	Function to measure split quality	(MSE = \frac{1}{n} \sum (y_j - \hat{y}_j)^2)
10	random_state	Controls randomness	(RandomSeed = k \Rightarrow Same\ Model\ Output)

3.3 Hyperparameter Optimization via Grid Search CV

This research attempts to increase the efficiency of the classifier, to determine the optimal hyperparameter values on the basis of validation set performance, this method tuning the hyperparameters.

In this investigation, K Fold CV is utilized to search the hyperparameters: n_estimators, learning_rate and max_depth. This approach explores the combinations of hyperparameters and finds the optimal set of hyperparameters for the validation set.

In this research, the hyperparameters: n_estimators, learning_rate and max_depth are optimised with K-Fold Cross-Validation. This provides better generalization to unseen data. Figure 2 demonstrates Hyperparameter Optimization via Grid Search Cross Validation.

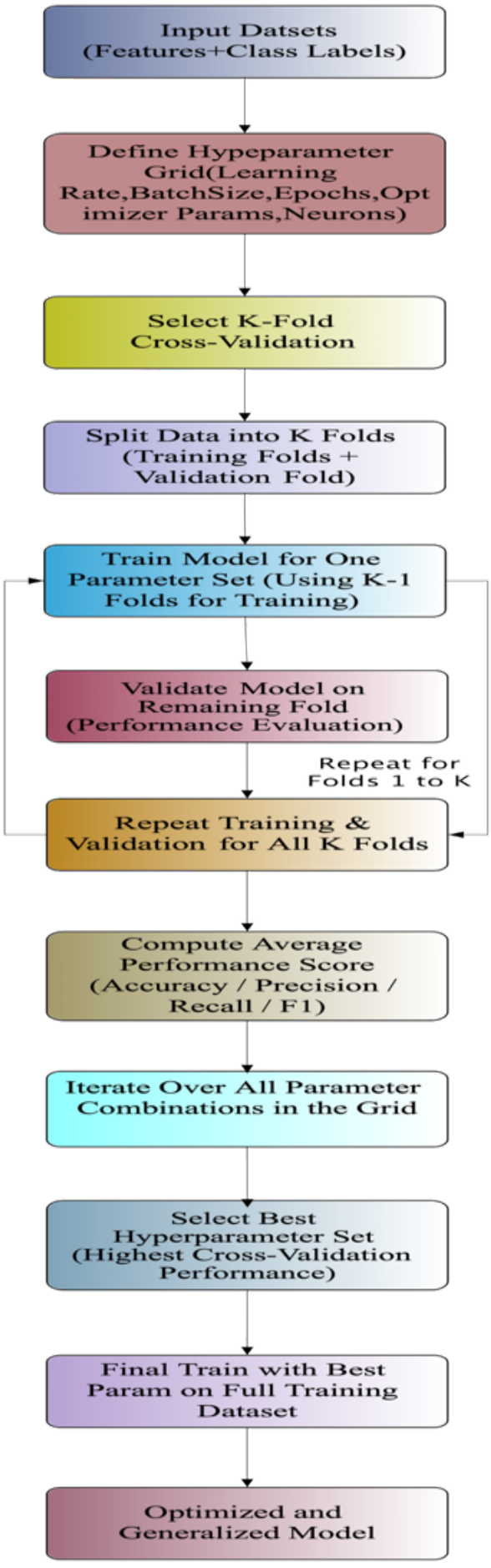

Fig.2 Hyperparameter Optimization via Grid Search Cross-Validation

In order to proceed ML models' presentation and capacity for extrapolation, hyperparameter tweaking is essential. Grid Search CV is utilized in this research to choose the ideal collection of hyperparameters to use with the GBC.

Define the training dataset as:

$$D = \{(x_i, y_i)\}_{i=1}^{N} \quad (2)$$

x i is the input features and y i is the class labels.

Hyperparameter Space

The set of hyperparameters is defined as:

$$\Theta = \{\theta_1, \theta_2, \dots, \theta_m\} \quad (3)$$

where each configuration is given by

$$\theta = \{n_{est}, \eta, d_{max}, p, f_{max}\} \quad (4)$$

Here, nest denotes the number of estimators, η is the learning rate, dmax represents the maximum depth of trees, p is the subsample ratio, and fmax indicates the number of features considered for splitting.

3.4 Cross-Validation Strategy

The dataset is split into K disjoint subsets:

$$D = \bigcup_{k=1}^{K} D_k, D_i \cap D_j = \emptyset \text{ for } i \neq j \quad (5)$$

For each fold k, the training and validation sets are defined as:

$$D_{train}^{(k)} = D \setminus D_k, D_{val}^{(k)} = D_k \quad (6)$$

Model Evaluation

For a given hyperparameter configuration θ, the model prediction is expressed as:

$$\hat{y}_i^{(k)} = f(x_i; \theta, D_{train}^{(k)} \quad (7)$$

The accuracy for each fold is computed as:

$$Accuracy_k(\theta) = \frac{1}{|D_k|} \sum_{(x_i, y_i) \in D_k} 1(y_i = \hat{y}_i^{(k)}) \quad (8)$$

3.5 Optimal Hyperparameter Selection

The overall performance is calculated as the average across all folds:

$$Score(\theta) = \frac{1}{K} \sum_{k=1}^{K} Accuracy_k(\theta) \quad (9)$$

The optimal hyperparameter set is obtained by maximizing the validation score:

$$\theta^* = \arg\max_{\theta \in \Theta} Score(\theta) \quad (10)$$

Integration with Gradient Boosting

The selected optimal parameters are used to train the final model:

$$F_m(x; \theta^*) = F_{m-1}(x) + \eta \cdot h_m(x) \quad (11)$$

3.6 Proposed Optimized Gradient Boosting Classifier

The suggested study methodology is a systematic framework that integrates traditional Natural Language Processing (NLP) techniques with an ensemble learning model to categorize textual data into three classifications: HATE (0), OFFENSIVE (1), and NEITHER (2) (mrmorj, 2022) The framework comprises the subsequent successive components, as depicted in fig.1.To avoid data leakage during model development, hyperparameter tuning was performed using the training and validation sets, while the test set remained completely unseen until the final evaluation stage. The mean performance and its corresponding standard deviation are displayed. Fig.3 demonstrates Proposed Research Methodology.

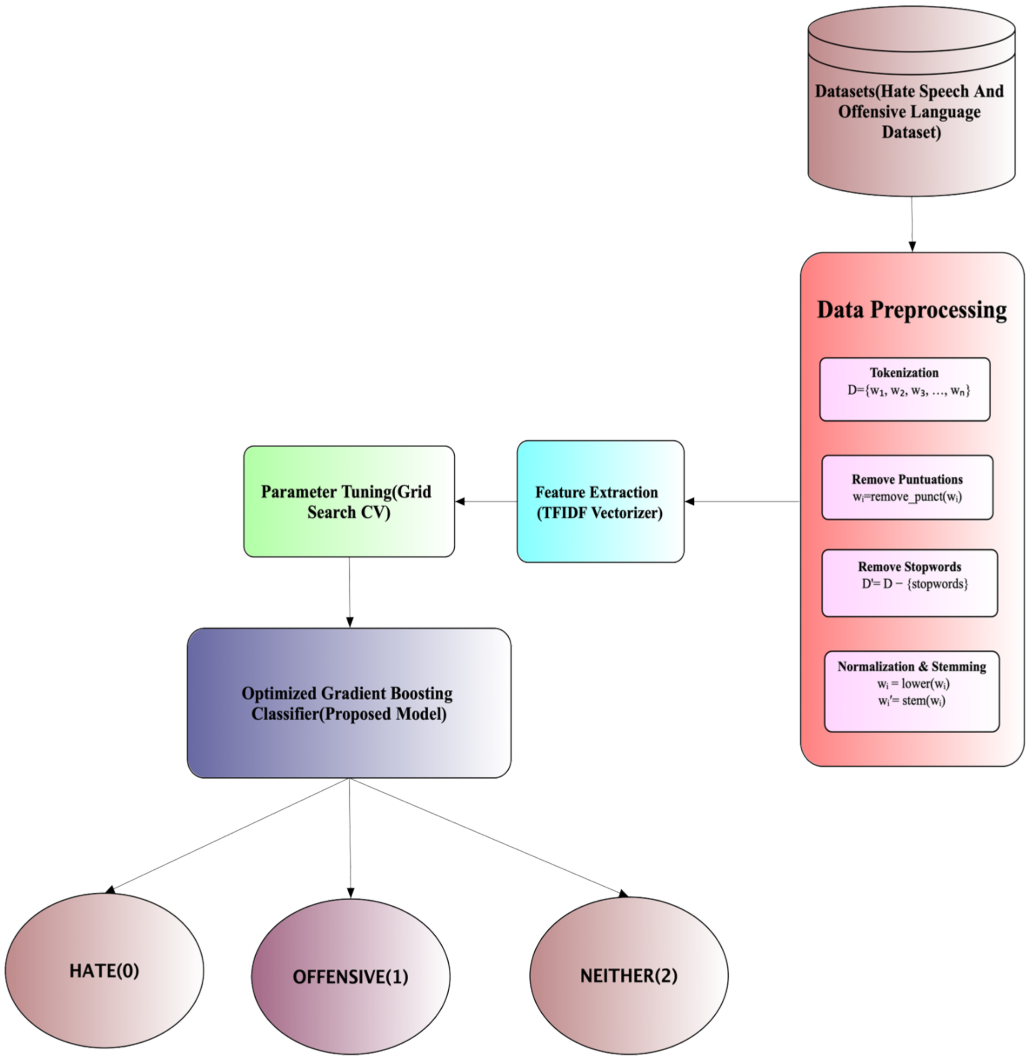

Figure 3: Proposed Research Methodology

3.7 Classification using Optimized Gradient Boosting

The study utilizes the GBC as the primary classification model, which operates as an ensemble learning approach using stage-

wise additive optimization. It sequentially integrates weak learners, commonly decision trees, by fitting them to the residual errors derived from a differentiable loss function. The model may gradually improve prediction accuracy and successfully capture intricate data patterns while retaining a strong generalization capability thanks to its iterative error-correction method.

The TFIDF method, which transforms textual data into a numeric representation appropriate for machine learning(ML) it is applied to derive features for training purposes, while Grid Search with CV is utilized to select the optimal set of hyperparameters. model configuration and further improve performance.

Thanks to its exceptional performance, the GBC model is well suited to this task. It's a reliable choice for accurate and stable classification as it can handle class imbalances well and is less prone to overfitting. GBC is a collaborative approach that integrates several weak learners in a sequential manner to produce a strong learner. The model begins by making an initial prediction, on the data using a base model. The residuals are then calculated as the residual representing the variation between the performances of different models predicted output and the actual value.

These residuals are used to train each successive weak learner, with the goal of fixing the errors committed by the prior model. With each additional learner in the process, the model becomes more accurate. The overall prediction is then produced by grouping of the weighted predictions obtained from all weak classifiers

$$\hat{y} = \hat{y}_0 + a_1\hat{y}_1 + a_2\hat{y}_2 + \quad \ldots\ldots\ldots\ldots \quad (12)$$

This iterative error-correction process allows the model to progressively improve its predictions, resulting in higher accuracy and more robust classification performance. Fig. 4 illustrates the overall Gradient Boosting framework used for the classification.

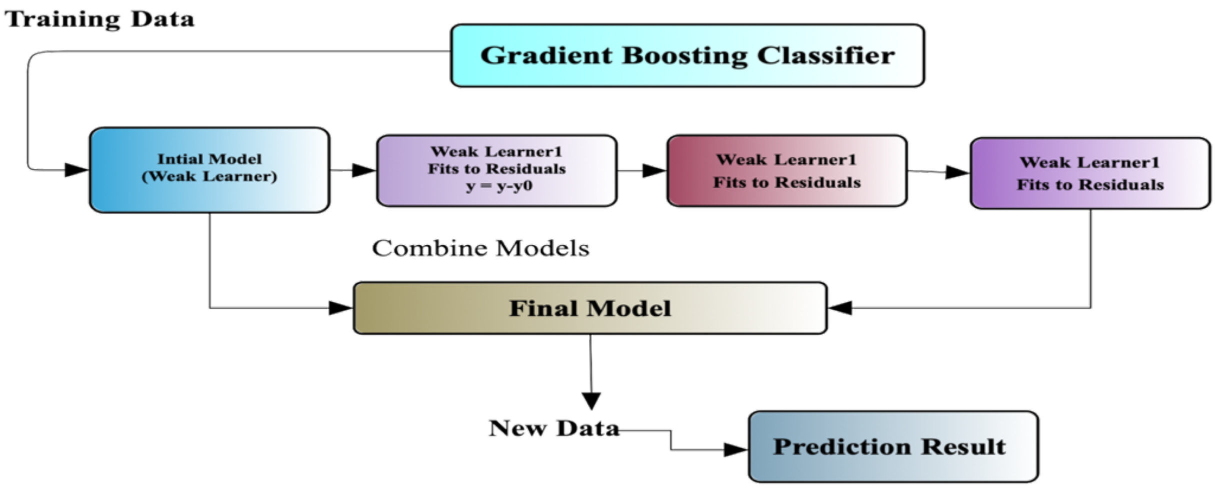

Figure :4 Gradient Boosting Framework for Classification

By including Grid Search Cross-Validation for hyperparameter adjustment, the enhanced Gradient Boosting model improves performance. First, a collection of hyperparameters is established, including the tree depth, earning rate and the number of estimators; to determine the optimal configuration, Grid Search methodically assesses various combinations of these factors.Figure5 illustrates Optimized Gradient Boosting Classifier using Grid Search CV

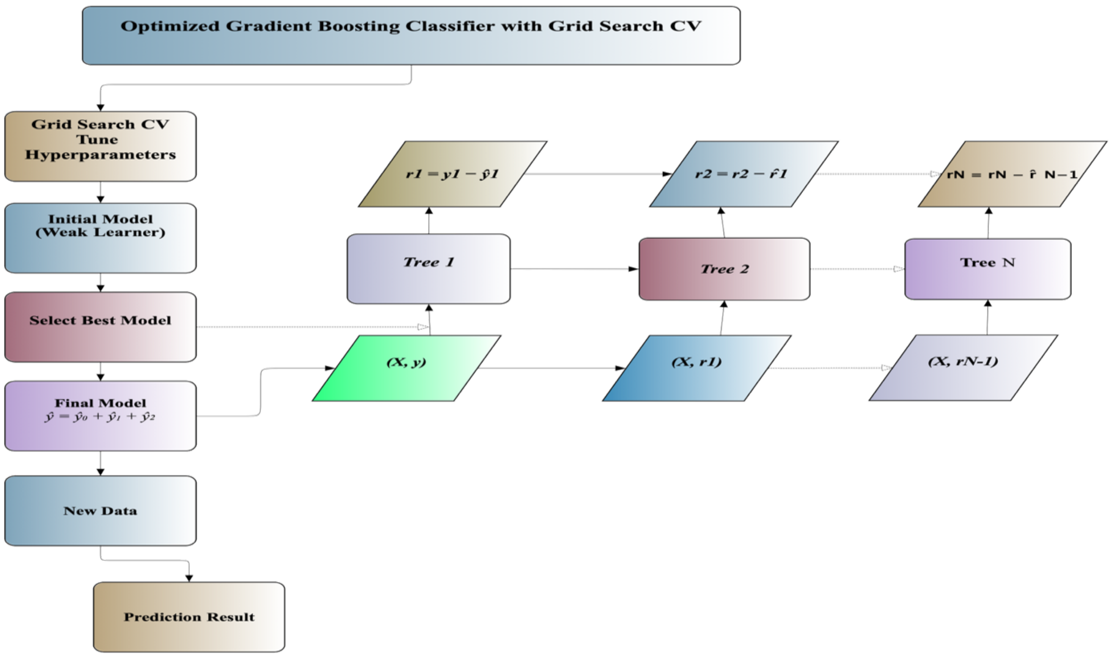

Figure :5 Optimized Gradient Boosting Classifier using Grid Search CV

The model begins with an initial weak learner that produces a prediction. Residual errors are computed as:

$$r_1 = y_1 - \hat{y}_1 \ldots\ldots\ldots\ldots (13)$$

Subsequent trees are trained iteratively on updated residuals:

$$r_2 = r_1 - \hat{r}_1, \ldots, r_N = r_{N-1} - \hat{r}_{N-1} \ldots\ldots\ldots\ldots (14)$$

In order to increase prediction accuracy, each tree tries to reduce the residual error. The chosen hyper parameters are guaranteed to generalize well across various data splits thanks to cross-validation. The final optimized model is built and applied to make estimates on new data once the best-performing model is identified. This approach depicts Optimized GBC using Grid Search CV.

4. Result and Discussion

The proposed model integrates TF-IDF feature extraction and a GBC that utilizes Grid Search CV. The model demonstrates excellent performance with high metrics in precision, recall, and F1 from train and test sets. The model also achieved a efficiency of 94.9% accuracy.

LSTM+RNN, CNN, and LSTM+GBDT, which all demonstrate lower performance levels, the model achieves a competive predictive accuracy and a lower cost in comparison to the aforementioned. Although promising, the results do signal a need for improvement. This improvement may be achieved through the use of varied datasets to rigorously validate the model's effectiveness. In light of aging datasets, to achieve the best results, introduced datasets have a sparse presence. In such an established field, a greater need for improvement persists. The model was assessed against comprehensive standards, including precision, recall, accuracy, and F1, in addition to k-fold cross-validation and confusion matrix analysis, in order to verify its reliability and robustness.

CONFUSION MATRIX-BASED COMPUTATION

For a given class i, the elements that make up the confusion matrix include defined as: True Positive (TP),False Positive (FP),False Negative (FN),True Negative (TN)

Confusion Matrix Values

TP = 10,736

FP = 443

FN = 821

TN = 12,783

Total= 24,783

Precision

Precision it represents how many selected objects are relevant.

Precision=TP/(TP+FP)…..……..……..……..….(15)

Precision=10736/(10736+443)=10736/11179=96.04

Recall

Recall is a measurement of the model's ability to pick up on true positive cases.

Recall=TP/(TP+FN)…….……..…..…(16)
Recall= 10736/(10736+ 821)=10736/11557=92.9

F Score

Harmonic mean in between Precision & Recall.

F1=2×(Precision×Recall)/(Precision+Recall)….…….(17)
F1 = 2 × (96.04 × 92.9) / (96.04 + 92.9) = 94.40%

Accuracy

It shows the percentage of all forecasts that were successfully classified.

Accuracy=(TP+TN)/Total ……..……..……..…….(18)

Accuracy=(10736+12783)/24783=23519/24783=94.9

Where

TP(Total = 10,736): Correctly predicted the positive occurrences.

FP (FP = 443): Incorrectly predicted positive occurrences.

FN(FN = 821): Actual positive instances incorrectly predicted as negative.

TN (TN = 12,783): Correctly predicted negative instances.

4.1 Comparative Analysis of Proposed and Existing Models

The Gradient Boosting Classifier in the proposed work achieves accuracy 94.9%, precision 96.0%, recall 92.9%, and F1-score 94.4%. Additionally, the findings are contrasted with earlier research. Figure 6 displays Table 7 graphical depiction.

Table 7: Evaluation Metrics Comparison for Existing and Proposed Models

S.No.	Metrics	Accuracy(%)	Precision(%)	Recall(%)	F1score(%)
1	LSTM+RNN	85	84	83	81
2	CNN	86	82	70	75
3	LSTM+GBDT	93	92	89	90
6	Gradient Boosting Classifier (Proposed Work)	94.9	96.0	92.9	94.4

Here's your comparison written clearly with calculated percentage improvements:

The proposed model has an accuracy of 94.9%, outperforming LSTM+RNN by 9.9%, CNN by 8.9%, and LSTM+GBDT by 1.9%.For precision, the proposed model achieves 96%, which is 12% higher than LSTM+RNN, 144% higher than CNN, and 4% higher than LSTM+GBDT. Likewise, for recall, the proposed model attains 92.9%, which is 9.9% higher than LSTM+RNN, 22.9% higher than CNN, and 3.9% higher than LSTM+GBDT.

The optimized F1-score of the proposed model is 94.4%, which is 13.4% better than LSTM+RNN, 19.4% better than CNN, and 4.4 % better than LSTM+GBDT. In conclusion, the findings demonstrate that the proposed Gradient Boosting Classifier (GBC) significantly outperforms traditional and hybrid deep learning models across all evaluation metrics, with especially strong improvements over CNN and notable gains even against high-performing hybrid models like LSTM+GBDT.

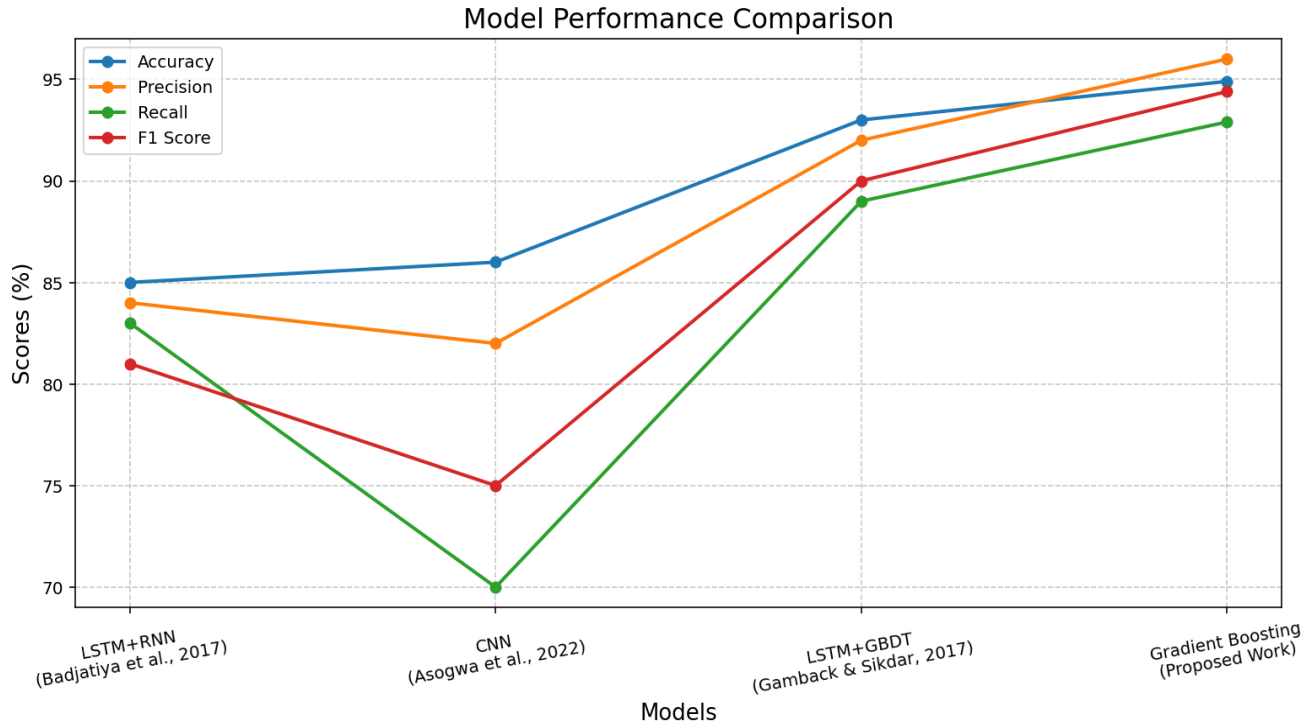

Figure :6 Performance for F1score, Recall, Precision and Accuracy

Fig. 6 shows that the proposed model got accuracy 94.9%, exceeding LSTM+RNN (85%), CNN (86%), LSTM+GBDT (93%), thus achieving better classification performance among all compared models.

4.2 Confusion matrix

The matrix has four components: TP, TN, FP and FN. The greater the number of predictions that are true, be they TP or TN, and the fewer the number of false predictions, the greater the overall performance. The results show that the model could accurately classify 9416 positive predictions (37.99%) and 14,104 negative predictions (56.91%). These components fall on the diagonal f the matrix, showing that the model is capable of classifying predictions accurately.

The fewest instances of error fell. 506 instances (2.04%) were incorrectly classified as negative, and 757 instances (3.05%) were incorrectly classified as positive. Therefore ,it achieves higher accuracy Fig.7 depicts Confusion Matrix for Model Performance Evaluation

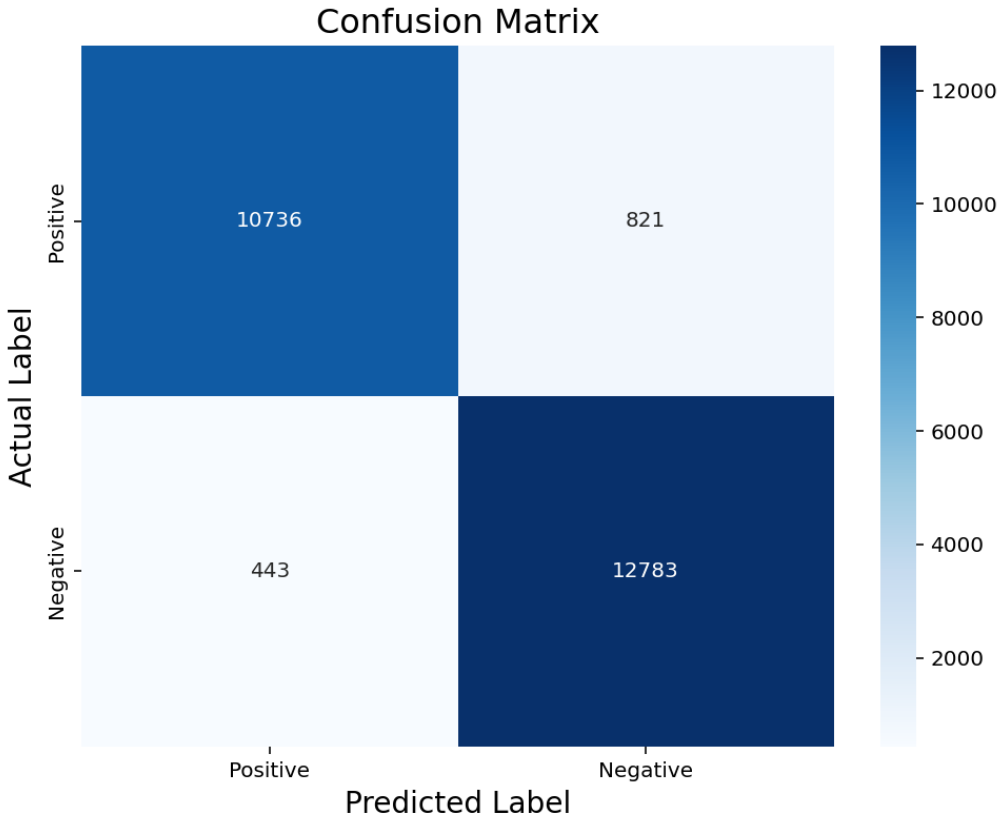

Figure: 7 Confusion Matrix for Model Performance Evaluation

Fig.7 depicts Confusion Matrix for Model Performance Evaluation which is compared with actual and projected class labels.

4. Conclusion

The use of TF-IDF feature extraction with a GBC and hyperparameter optimization by Grid Search CV yields satisfactory results in multi-class classification of hate speech. It is the model with the competitive performance in the work with accuracy 94.9%, precision 96%, recall 92.9% and F1-score 94.4 %.

The suggested classifier has consistently consistent performance and the competitive accuracy among other work. This can be used in order to compare the suggested model with existing advanced techniques but may vary due to different text processing and feature extraction techniques, model architecture and tuning of hyperparameters.

The proposed Gradient Boosting Classifier is an effective model for real-time monitoring of social media as it can deliver these results with less computational resources than other deep learning models. This model provides a suitable trade-off between computational efficiency, scalability and accuracy, and is an appropriate approach for large-scale, practical hate speech recognition.

5. Limitations of the Study

The effectiveness of the proposed model is good, with a number of limitations should be considered. The model is tested on a single dataset, which might not be very generalizable in a variety of real-life situations. The TF-IDF feature extraction algorithm is limited in capturing to reveal deep contextual/ semantic relationships as effectively as transformer-based algorithms. Future studies are aimed at overcoming these limitations through the use of various datasets and the most modern deep learning methods.

6. Future work

Future research can be directed towards improving the hate speech detection systems by improving the feature extraction techniques by using advanced DL models. To improve the system's precision, resilience, and generalizability, ensemble learning (a combination of multiple models) can be also used. In addition, larger and more varied data sets, as well as improved detection of complex and minority classes can result in more accurate and balanced detection.

This work can be enhanced by considering Internet of Things (IoT) methods for hate speech detection IoT can help capture real-time, multimodal information, including text, pictures, and videos from IoT devices and social media sites (currently only text data), which can helps enhance the effectiveness of hate speech detection. This will enable the creation of more robust systems to identify hate speech in the real dynamic world from diverse data sources. In short, these advances will assist in the growth of more efficient, scalable and context-sensitive hate speech detection systems for a safe and healthy online ecosystem.

Data Availability: The datasets used in this study are available for free from the Kaggle repository:

https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset

REFERENCES

Asogwa, D. C., Chukwuneke, C. I., Ngene, C., & Anigbogu, G. (2022). Hate speech classification using SVM and naive BAYES. *ArXiv Preprint ArXiv:2204.07057*.

Ayo, F. E., Folorunso, O., Ibharalu, F. T., & Osinuga, I. A. (2020). Machine learning techniques for hate speech classification of twitter data: State-of-the-art, future challenges and research directions. *Computer Science Review*, *38*, 100311.

Ayo, F. E., Folorunso, O., Ibharalu, F. T., Osinuga, I. A., & Abayomi-Alli, A. (2021). A probabilistic clustering model for hate speech classification in twitter. *Expert Systems with Applications*, *173*(February 2020). https://doi.org/10.1016/j.eswa.2021.114762

Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017). Deep learning for hate speech detection in tweets. *Proceedings of the 26th International Conference on World Wide Web Companion*, 759–760.

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. *Proceedings of the International AAAI Conference on Web and Social Media*, *11*(1), 512–515. https://doi.org/10.1609/icwsm.v11i1.14955

de Gibert, O., Perez, N., García-Pablos, A., & Cuadros, M. (2018). *Hate Speech Dataset from a White Supremacy Forum*. https://doi.org/10.48550/arXiv.1809.04444

Feng, C., Jiang, M., Huang, Q., Zeng, L., Zhang, C., & Fan, Y. (2022). A Lightweight Real-Time Rice Blast Disease Segmentation Method Based on DFFANet. *Agriculture (Switzerland)*, *12*(10), 1–12. https://doi.org/10.3390/agriculture12101543

Gambck, B., & Sikdar, U. K. (2017). Using convolutional neural networks to classify hate-speech. *Proceedings of the First Workshop on Abusive Language Online*, 85–90.

Gao, L., & Huang, R. (2017). *Detecting Online Hate Speech Using Context Aware Models*.

Goswami, P., & Daniel, A. (2026a). Enhancing Hate Speech Detection with a Distil BERT and BiLSTM Hybrid Mode. *SN Computer Science*, *7*(4), 353. https://doi.org/10.1007/s42979-026-04913-w

Goswami, P., & Daniel, A. (2026b). Optimized BiLSTM for Multi-Class Hate Speech Detection with Improved Performance. *SN Computer Science*, *7*(4), 336. https://doi.org/10.1007/s42979-026-04945-2

Jha, A., & Mamidi, R. (2017). When does a compliment become sexist? Analysis and classification of ambivalent sexism using twitter data. *Proceedings of the Second Workshop on NLP and Computational Social Science*, 7–16. https://doi.org/10.18653/v1/W17-2902

Kevin, L., Yee, C., Liang, C., Yong, S., Han, Y., Mukred, M., & Mohammed, F. (2025). *Hybrid Deep Learning for Detecting Hate Speech Across Social Media Platforms* (pp. 289–304). https://doi.org/10.1007/978-3-031-75091-5_16

Mercan, V., Jamil, A., Hameed, A. A., Magsi, I. A., Bazai, S., & Shah, S. A. (2021). Hate Speech and Offensive Language Detection from Social Media. *2021 International Conference on Computing, Electronic and Electrical Engineering (ICE Cube)*. https://doi.org/10.1109/ICECube53880.2021.9628255

mrmorj. (2022). *Hate Speech and Offensive Language Dataset*.

Pitenis, Z., Zampieri, M., & Ranasinghe, T. (2020a). Offensive Language Identification in Greek. *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 5113–5119.

Pitenis, Z., Zampieri, M., & Ranasinghe, T. (2020b). Offensive Language Identification in Greek. *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 5113–5119.

Que, Q., Sun, R., & Xie, S. (2020). Simon@HASOC 2020: Detecting Hate Speech and Offensive Content in German Language with BERT and Ensembles. *FIRE 20, Forum for Information Retrieval Evaluation*, 283–289.

Ramos, G., Batista, F., Ribeiro, R., Fialho, P., Moro, S., Fonseca, A., Guerra, R., Carvalho, P., Marques, C., & Silva, C. (2024). A comprehensive review on automatic hate speech detection in the age of the transformer. In *Social Network Analysis and Mining* (Vol. 14, Number 1). Springer. https://doi.org/10.1007/s13278-024-01361-3

Ribeiro, M., Calais, P., Santos, Y., Almeida, V., & Jr., W. M. (2018). Characterizing and Detecting Hateful Users on Twitter. *Proceedings of the International AAAI Conference on Web and Social Media*, *12*(1). https://doi.org/10.1609/icwsm.v12i1.15057

Umansky, N., & others. (2026). Improving Hate Speech Detection with Large Language Models. *EPJ Data Science*.

Vijay, V., & Verma, P. (2023). Variants of Na□ve Bayes Algorithm for Hate Speech Detection in Text Documents. *2023 International Conference on Artificial Intelligence and Smart*

Communication (AISC), 18–21. https://doi.org/10.1109/AISC56616.2023.10085511

Waseem, Z. (2016). Are You a Racist or Am I Seeing Things? Annotator Influence on Hate Speech Detection on Twitter. *Proceedings of the First Workshop on NLP and Computational Social Science*, 138–142. https://doi.org/10.18653/v1/W16-5618