DOI: 10.2478/ias-2024-0012 • JIAS • 19(4) • 2024 • 162-179

# Advancements in Offensive Language Detection: A Comprehensive Review and Experimental Analysis

**C. Nalini, R. Shanthakumari , Y. Agashia Maria, T. Janarthanan and M. Manibharathi**

Department of Information Technology, Kongu Engineering College Erode 638060, TamilNadu, India; nalini@kongu.ac.in (C.N.); rsk_shan@kongu.ac.in (R.S.); agashiamaria@gmail.com (Y.A.M.); janathangavel11@gmail.com (T.J.); manibharathidct@gmail.com (M.M.)

**Abstract:** The proliferation of offensive language in digital communication has become a significant challenge in the internet era, prompting the urgent need for advanced Natural Language Processing (NLP) techniques for its identification and mitigation. With a particular focus on NLP techniques, machine learning, deep learning, and transformer models, this study presents a thorough review of the shifting landscape of offensive language identification from the years 2020 through 2023. The datasets utilized in prior research have been scrutinized, specifically those of Dravidian languages such as Tamil, Malayalam, etc. Preprocessing techniques encompass a range of data cleansing and word embedding methodologies, including TF-IDF and Word2Vec, which are employed to train and optimize the model. We reviewed past work to compare the standard supervised learning models like Support Vector Machine and Naive Bayes to emergent transformer models like BERT, identifying the superior approach that would improve a model's accuracy and effectiveness.

**Keywords:** offensive language; Dravidian language; word embedding; machine learning; deep learning; transformer model

## 1 Introduction

In an era characterized by the escalation of digital communication platforms, the challenge of identifying and mitigating offensive language is crucial. Offensive language, which encompasses hate speech, harassment, and profanity, has the potential to harm individuals, undermine online communities, and perpetuate harmful stereotypes. Consequently, the development of automated techniques to detect and address offensive language has become an essential endeavor, carrying profound implications for online safety, content moderation, and the preservation of healthy digital environments. The identification of offensive language resides at the convergence of natural language processing and ethical content moderation. It is essential to automate the moderation of this content because it is impossible to manually identify within the enormous volume of data generated on social media. As a result, the urge to create computational systems has driven many researchers in the field of natural language processing to mitigate the spread of harmful content or remove it entirely using state-of-the-art techniques. Although there is a notable amount of study based on these processes in languages like English, Dravidian languages lack this level of research. Therefore, it is crucial to expand the research linked with offensive language identification for Dravidian languages like Tamil, Malayalam, Kannada, etc.

As the collection of datasets constitutes the preliminary phase of this undertaking, the comments contained therein, particularly the textual data, may originate from diverse social media platforms like Facebook, YouTube, Twitter, and similar sources. In the context of Dravidian Languages, the establishment of a corpus assumes a pivotal significance in the realm of offensive language detection endeavors. Each Dravidian language itself holds a standard metalinguistic terminology and scholarly vocabulary, necessitating a corpus to be created by proper identification of lexical root words and the removal of affixes and suffixes added to it. The establishment of a corpus does not pose a challenge when working with the English language, as the natural language processing toolkit provides a pre-established wordlist, allowing for direct access through a simple importation process. Subsequently, the data preprocessing stage occurs, involving tasks such as data cleansing, elimination of stop words,

translation, vectorization, and other related procedures, utilizing the accessible dataset and corpus. Data preprocessing can be performed in multiple ways; for example, the word embedding technique includes TF-IDF, Word2Vec, and GloVe. Once the data preprocessing is done, the machine learning and transformer models will act to categorize the data into binary or multi-class labels.

This paper primarily concentrates on a range of methodologies that have been employed in the detection of offensive languages through the utilization of different machine learning algorithms, deep learning techniques, and emerging transformer models. The suggested models are based on supervised learning algorithms such as Logistic Regression, Support Vector Machine, and Decision Tree, as well as algorithms like CNN, RNN, BiLSTM, GCN in deep learning, and transformer models like BERT and m-BERT. These are examined and compared based on their effectiveness and efficiency in identifying offensive language. The objective is to ascertain the most optimal strategy for accomplishing this task.

## 2 Offensive language review

In the contemporary digital landscape, the significance of offensive language detection within the realms of online content moderation, social media, and digital platforms cannot be overstated. As we navigate an era where online communication is a cornerstone of our daily lives, the capability to not only identify but also effectively mitigate offensive content holds a pivotal role. Notably, on social media platforms, it is the first line of defense against issues like hate speech, cyberbullying, and harassment. Furthermore, within comment sections and online communities, this technology fosters an environment where respectful and constructive conversations can thrive. Nonetheless, challenges persist in this domain, primarily stemming from the ever-evolving and context-dependent nature of offensive language. Striking the delicate equilibrium between content moderation and the preservation of free speech remains an ongoing ethical dilemma. Confronting these challenges head-on is imperative to ensure that offensive language detection remains an instrumental tool in constructing safer, inclusive, and respectful digital ecosystems.

### *2.1 Forms of offensive language*

Offensive language takes on various forms, each presenting unique challenges for detection.

**Hate speech.** involves expressions of hatred or prejudice against individuals or collectives, with a focus on characteristics such as race, faith, gender, cultural background, or sexual orientation.

**Profanity. includes** vulgar, obscene, or offensive language and gestures, typically used to insult or degrade others. Its spectrum spans from mild expletives to explicit and profoundly offensive terms.

**Slurs.** are insulting expressions that are directed at racial, ethnic, or social groups. They are intended to injure and denigrate someone based on their identity.

**Harassment.** involves unwanted, persistent, and offensive behavior, including stalking, threats, and personal attacks. It can be directed at individuals or groups.

**Trolling.** is a purposeful attempt to provoke others' emotions through provocative language, offensive statements, and disruptive behavior.

### *2.2 Data Collection*

The acquisition, cleansing, and structuring of data for the purpose of developing efficient models for offensive language detection are fundamental stages. This section delves into the techniques and factors involved for the purpose of this undertaking.

### *2.3 Data sources*

Social media networks, such as Twitter, Facebook, YouTube, and Instagram, serve as abundant repositories of user-generated content, frequently harboring offensive language. The data from these platforms can be extracted utilizing APIs or web scraping tools. Specialized datasets are meticulously curated for research in identifying inappropriate content, including the Twitter dataset focused on offensive language, the Hate Speech and Offensive Language (HASOC) dataset, and the YouTube comments dataset.

Potential challenges may arise during dataset collection, such as imbalanced data due to offensive language being a minority class. Techniques like bagging and SMOTE can be employed to balance the dataset. Ethical concerns regarding privacy and consent must be addressed when collecting data from social media. Additionally, multilingual and Dravidian language datasets are less abundant compared

to English, necessitating researchers to create their own datasets through time-consuming manual annotation.

### *2.4 Text Normalization*

To improve the precision of recognizing offensive content, text data is standardized and cleaned as part of the text normalization process. A few possible techniques include lowercase text conversion, stemming, lemmatization, erasing stopwords, emoticons, URLs, punctuation, and handling unusual characters. Normalization in Dravidian languages like Tamil, Kannada, and Telugu may involve character encoding, script conversion, translation, and managing language-specific diacritics. Models for foul language detection can assess and categorize content effectively by ensuring uniform text representation.

Accordingly, the normalized data will be transformed into numerical form and utilized to train the various ML, DL, and transformer models with several feature engineering techniques. F1-score, precision, accuracy, recall, and AUC-ROC are a few evaluation metrics that can be used to assess the model.

## 3 Related Work

The ubiquity of social media platforms has empowered individuals to express their opinions, yet it has concurrently given alarming rise to hate speech and offensive language resulting in detrimental effects on the psychological well-being of individuals and the deliberate targeting of vulnerable communities. It is imperative to tackle and diminish the prevalence of offensive content on social media to uphold the integrity and inclusiveness of online forums. The following review examines significant contributions and emerging technologies used in this field during the timeframe of 2020 to 2023. The review study approaches show in Figure 1 Related Work Study Plan

### *3.1 Features Extraction Technique*

Feature extraction involves transforming raw data into a more compact and informative representation, or features, which can be used for tasks such as classification, regression, clustering, and information retrieval. Approaches like N-gram, Term Frequency-Inverse Document Frequency, and Bag-of-Words techniques facilitate the transformation of text documents into feature vectors, thereby enabling their application in classification and clustering tasks. Additionally, word embeddings like Word2Vec and GloVe offer compact and semantically meaningful representations for words, thereby enhancing the comprehension of textual context. Feature extraction plays a crucial role in improving model performance, reducing dimensionality, and uncovering valuable insights from complex data.

**N-gram.** The N-gram approach is a powerful and versatile technique for analyzing sequential data in NLP. N-grams are contiguous sequences of ‘n’ items, which can be words, characters, or other linguistic units. It finds extensive application in various tasks, providing valuable insights into word combinations and their probabilities. The researchers in [1] employed a blend of word n-grams and improved syntactic n-grams along with feature selection techniques for the categorization of Twitter data into offensive language, non-offensive content, and hate speech. The proposed model , as seen in [2], used the bigram feature along with the support vector machine technique, producing accurate categorization rate. N-grams possess their own set of constraints; one noteworthy limitation is that words related to each other can have the maximum possible distance within a sentence [3]. In these cases, a concept called “attention” is used for discriminating between offensive and non-offensive contexts.

**Term Frequency-Inverse Document Frequency.** refers to a numerical metric that quantifies the frequency of words in a text, indicating how they characterize the document within a collection. High TF-IDF scores are assigned to words that are common within a document but rare in the overall dataset, making it valuable for feature extraction in NLP. [4] ML classifiers such as Multinominal NB, LightGBM, and SVM employed TF-IDF vector values in detecting multilingual offensive language content. [5] n-grams-based TF-IDF and MuRIL. In other cases, n-grams-based TF-IDF representation is used to effectively capture morphological variations of the words. This paper shows that n-gram-based TF-IDF outperforms MuRIL in Tamil and Kannada. Both BoW and TF-IDF feature representation techniques were implemented in studies [6] with various classifiers, and the superior result is from the SVM along with the TF-IDF.

**Word2Vec.** encodes words as dense, continuous-valued vectors, capturing semantic and syntactic similarities. Words with comparable meanings or usage exhibit vectors that are proximate in this vector space. Word2Vec yielded superior results when integrated with machine learning models such as SVM, ensemble techniques, and deep learning methods like LSTM compared to the performance of the FastText approach when applied to code-mixed text in Tamil, Kannada, and Malayalam [7]. To cope with the drawbacks of the short texts [8], semantic word expansion based on word-embedding vectors,including Word2Vec, has been used as an input in various ensemble classifiers.

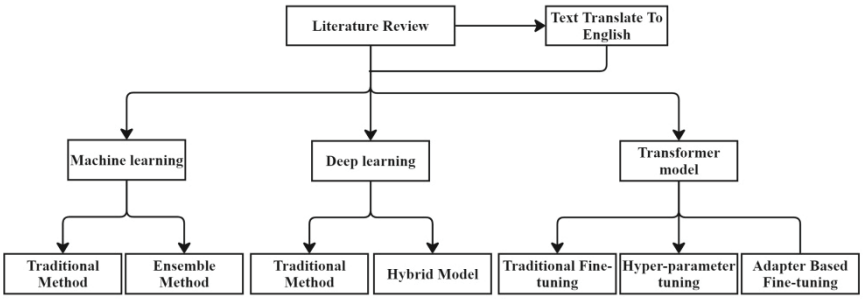


***Figure 1.*** Related Work Study Plan

## *3.2 Machine learning-based approaches*

Numerous ML models are being developed to execute tasks such as clustering, prediction, classification, etc. These models effectively utilize available data, particularly labeled data, to achieve dependable accuracy. The performance of machine learning algorithms relies heavily on the accurate identification or extraction of features. Subsequently, classification algorithms undertake the detection of offensive content in textual data from the use of various algorithms, such as Naïve Bayes, Decision Tree, Support Vector Machine, K-Nearest Neighbor, etc. The linear SVM, NB, and k-NN models are used [6], along with TF-IDF and BoW feature representation techniques using different combinations of n-grams. Linear SVM performs better than the other three classifiers for this corpus. The authors in [9] proposed the finest approach for the HASOC shared task (Task 2) and received 2nd rank for Multinominal Naive Bayes in the classification of Malayalam text, and 3rd rank for SVM on Tamil text in the competition. Among various machine learning models [10], SVM is predominantly employed by researchers, followed by Random Forest and other algorithms for the classification of social media data into hate or non-hate classes. Although their dataset is from South African tweets, the authors of [11] developed an English tweet corpus and implemented a range of algorithms that indicated n-gram features in the SVM, Random Forest, and gradient-boosting multi-tier model consistently demonstrated a well-balanced performance. The ensemble classifier with n-gram features exhibited notably weaker performance. Among the various ML algorithm approaches like LightGBM, Catboost, Random Forest, SVM, and MNB [12], LightGBM exhibited commendable performance when hyperparameter tuning is applied. The authors of [13] stated that conventional ML models such as LR, NB, Random Forest, and SVM are unable to grasp the context present in Dravidian code-mixed posts. Therefore, employing an ensemble framework can address this issue. The findings in [14] reveal that the machine learning algorithm will certainly stop learning if the data continues to grow through a learning rate graph. This calls for the use of deep learning to solve the problem.

## *3.3 Deep learning-based approaches*

Deep learning models are distinguished by their capacity to autonomously acquire and depict intricate patterns and correlations within data via numerous layers of interconnected neurons. They encompass fundamental principles such as Artificial Neural Networks, hidden layers, activation functions, and backpropagation. The techniques employed in deep learning encompass convolutional neural networks, RNN, long short-term memory, and others. The authors of [14] built an ensemble model that utilizes the advantages of the BiLSTM, the LSTM+Convolution, and the CNN models, securing rank in the detection of Malayalam-English data. [15] made use of GCN to capture high-order neighborhood information through word co-occurrence and frequency-based edges, preserving global structure information. This method automatically learns interpretable word and document node embeddings, achieving strong classification performance with a small proportion of labeled

documents,highlighting the significance of graph embeddings in handling non-consecutive and long-distance semantics. Three Deep Learning architectures along with grid search for choosing optimal hyperparameters were implemented for identifying Malayalam, Tamil, and Kannada datasets [16]. The first model is the combination of CNN, Bi-LSTM and a fully connected layer, the second model consists of a Bi-LSTM layer, and the third model has a Bi-RNN layer. [17] introduces GTG, which combines Transformer and Graph Convolutional Network by incorporating part-of-speech (POS) information in word-document graphs. The GTG model focuses on spatial feature information and text order, utilizing a Graph Convolution Layer-Transformer-Graph Convolution Layer architecture to mitigate issues like gradient disappearance in RNN. The author also discusses related works involving TextGCN and introduces another method, IMGCN, which leverages GCN and BiLSTM for text classification for enhanced accuracy. [18] presents a novel text-level GNN, addressing issues of fixed corpus-level graph structures and high memory consumption. The proposed model constructs individual graphs for each input text with shared global parameters, facilitating online testing while preserving global information. By connecting word nodes within a small window and employing a message passing mechanism, the model extracts more local features, which in turn enhance the accuracy. The authors of [19] proposed a document-relational GCN and evaluated on five benchmark databases. Their method explores different hidden nodes and proportions of document-document features, demonstrating superior accuracy. This study also emphasizes the effectiveness of link relations between documents in improving topic-sensitive text classification through feature engineering. The inductive graph-based framework has been implemented by the authors of [20], which distinguishes itself by presenting a novel way to make transductive GCN-based models inductive and outperforming state-of-the-art methods on five benchmarks, showcasing its effectiveness without relying on transductive approaches or pre-trained embeddings. It improves scalability by reducing time and space complexity as data size increases. The hybrid model of CNN and BiLSTM, followed by Graph convolution network and document-document relation, outperforms all other methods.

### *3.4 Transformer model-based approaches*

Transformer models have significantly advanced the field of natural language processing and provide context-aware, self-attention mechanisms and pre-trained model representations of text. Their versatility, transfer learning capabilities, and ability to handle complex linguistic constructs make them essential tools to combat offensive language on digital platforms. Transformer models include BERT, IndicBERT, Multilingual-BERT, RoBERTa, GPT, etc. The BERT-based, multilingual-cased model in [21] performed well for low-resource Dravidian offensive language detection as it used an additional input mask, dropout, and a pooled output layer to enhance the performance rate. Rajalakshmi et. al [22] proposed a MuRIL model and a majority voting ensemble as downstream classifier to analyze offensive content that aimed to harass, anger, or annoy an individual or community over a stemmed data with the combination of oversampling and downsampling techniques, achieving an accuracy of 86%. The authors also addressed that removal of stop words performs less compared to the stemming process. [23] used a majority voting ensemble of 6 models, 3 of XLM-RoBERTa and 3 of mBERT, based on the task adaptive pretraining technique along with Adam optimizer and fusion-architecture of subword-level, character-level, and word-level embedding. This ensemble improved the model performance and topped in EACL-2021 Shared Task. The authors of [24] have put forth three distinct iterations of BERT, namely IndicBERT, mBERT, and MuRIL. They assert that MuRIL, a transfer learning model, surpasses other models due to its training in both traditional scripts and transliterated corpus in Roman script, unlike IndicBERT which solely relies on native Indian scripts. The distillery-base-uncased and multilingual distilBERT-base case models were used in [25] to identify the offensive language for three different languages. The distilBERT-base-uncased contains 6 layers and 12 heads with 66 million parameters and 768 dimensions, whereas the multilingual distilBERT-base case is preferable as it contains 6 layers and 12 heads with 134 million parameters and 768 dimensions, and is trained on 104 different languages in Wikipedia text. The BERT-BiLSTM, XLMRoBERTa, CNN-BiLSTM, and Universal Language Model Tuning (ULMFit) models are trained on datasets of Malayalam, Tamil, and Kannada according to the authors of [26], who statee that the ULMFit performance got elevated both in minority and majority class. It has a slanted triangular learning rate by freezing a few of the layers in the model, whereas the other models' performances stand out only to the majority class.

The ensemble model of IndicBERT and generic BERT using a multi-label classification approach is implemented in [27], and the final score for each class is the combination of the confidence scores obtained from the IndicBERT model and the regular BERT model. Some researchers have also tried the ensemble of deep learning model and transformer models: [28] used the XLM-RoBERT, a pre-training model, to extract semantic information features of the text, and Deep Pyramid Convolutional Neural Networks (DPCNN) was

used to process the output features further. To improve the training effect, hierarchical cross-validation has been used, and the model performed better for the Malayalam dataset than Tamil and Kanada. In contrast to the ML and DL model, which has a high rate of misclassification for Tamil and Malayalam tasks, the authors of [13] proposed an ensemble model combining BERT, DNN, and MuRIL as a preferable method. Another author employed a voting soft ensemble model [29] that used the multilingual DistilBERT model, having 6 transformer layers with 12 attention heads and 134 million parameters in total, and the performance is better for the Malayalam dataset compared to Tamil. The base models in the transformer layers identified different patterns from the text which is captured and used by the ensemble model to produce the final predictions. [12] shows that MuRIL outperformed mBERT and all other ML models when trained on both translated and transliterated data as it is fine-tuned over various parameters.

## 4 Discussion

Detecting offensive content has become a pressing concern, especially in the ever-evolving landscape of social media. This paper systematically organizes state-of-the-art research conducted between 2020 and 2023 in response to this challenge. Notably, researchers have widely relied on the Twitter Hate Speech dataset and the Hate Speech and Offensive Language (HASOC) dataset to train and evaluate their models. To enhance feature extraction, methods such as TF-IDF, word2vec, and n-grams have been favored over traditional distance metrics and multimodal information extraction techniques.

In the process of identifying offensive language, a combination of directional models like LSTM and RNN have been instrumental, as well as non-directional models like Transformers and BERT. While traditional ML algorithms like Multinomial Naive Bayes, Support Vector Machines, and ensemble models have yielded promising results, the integration of advanced models like BERT and Transformers has catalyzed the rapid advancement of Natural Language Processing in this domain.

## 5 Proposed Methodology

### *5.1 Dataset Description*

In our research, we employ a binary classification approach to develop a detection model that assesses the offensiveness of user comments shared on social media. We have carefully curated a specialized dataset for this purpose, drawing from the Hate Speech and Offensive Language (HASOC) FIRE 2021 task as show in Figure 2. Datase. This dataset holds a feature vector with a total of 38214 rows containing 9381 Offensive comments and 28833 Non-Offensive comments.

| index | text | category |
|---|---|---|
| 0 | வணக்கம் டா மாப்ள வேதாரண்யத்துலேருந்து.. தரமான சம்பவம் காத்துட்டு இருக்கு | NOT |
| 1 | அப்ப 96 to 2016; வரைக்கும் ரஜினி அரசியலுக்கு வருவேன்னு தான் சொல்லிட்டு இருக்காரா | OFF |
| 2 | அருமை எவதொட்டாலும் வெட்டுடா அடிச்சா திருப்பி அடிப்பான் டா சத்திரியன் | OFF |
| 3 | எங்கள் சாதியின் குல தெய்வம் திரௌபதி அம்மன் அருள் பெற்று படம் வெற்றி அடைய வாழ்த்துக்கள்.... | NOT |
| 4 | trailer பார்த்துட்டு ஆய்த எழுத்து சூர்யா ஞாபகம் வந்தா மட்டும் like போடுங்க | NOT |
| 5 | நான் சார்ந்த devendira குல வேளாளர் சமுதாயம் சார்பாக படம் வெற்றி பெற என் மனமார்ந்த வாழ்த்துக்கள் | NOT |

***Figure 2.*** Dataset

### *5.2 Dataset Pre-processing*

In the context of text data for Natural Language Processing (NLP), preprocessing involves a series of steps to clean and prepare the text for analysis, classification, or any NLP-related task. Showed in Figure 3 Dataset Preprocess.

**Removing Duplicate Text.** The removal of duplicates ensures that the same text does not skew the analysis or training of NLP models by artificially inflating the importance of particular phrases or sentences.

**Removing Emoticons and Special Characters.** Symbols, punctuation, special characters, and emoticons often do not carry meaningful information so removing them can help streamline the text for further analysis and make it more readable.

**Tokenization.** Tokenization is the process of breaking text into individual units, such as words or subword units, so that algorithms can work effectively in that format. (e.g., “tokenization” into “token” and “ization”).

**Removing Stop Words.** Removing stop words (e.g., “ஒரு”, “இது”, “அது”) helps reduce the dimensionality of the data and focuses the analysis on more meaningful content words.

**Translation.** Translation is useful in multilingual NLP tasks. As the dataset contains both Tamil and English text, translating it into a common language helps in making the data consistent and accessible for analysis.

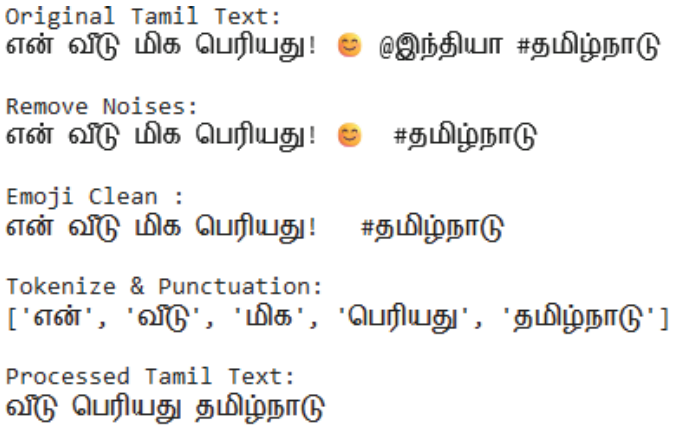


***Figure 3.*** Dataset Preprocess

The proposed methodology presents a comprehensive approach to categorizing comments into offensive and non-offensive classes. This methodology comprises three distinct modules: Machine Learning, Deep Learning, and Transform models, each strategically designed to leverage different computational paradigms for optimal performance.

In the Machine Learning module, four well-established algorithms are employed: Bernoulli Naïve Bayes, Logistic Regression, Support Vector Machine, and K-Nearest Neighbor. These algorithms, known for their efficiency and effectiveness in classification tasks, are deployed to process and analyze textual data, discerning patterns indicative of offensive language. Contrastingly, the Deep Learning module harnesses the power of neural networks, specifically employing a Graph Convolutional Network (GCN). GCNs excel in capturing intricate relationships within data represented in graph structures, making them particularly suitable for modeling complex interactions present in linguistic contexts. Lastly, the Transform module adopts a state-of-the-art language representation model, the Multilingual Bidirectional Encoder Representation from Transformers (BERT). BERT, renowned for its ability to capture contextual nuances and semantic relationships in text across multiple languages, offers unparalleled performance in natural language understanding tasks.

Each module is meticulously evaluated for its predictive prowess, with performance metrics such as accuracy, precision, recall, and F1-score meticulously recorded. Leveraging insights from previous studies and drawing upon the richness of the HASOC dataset, these models undergo rigorous experimentation to assess their efficacy in classifying comments accurately and efficiently.

## *5.3 Data Visualization*

The generated class label and its percentage count are visualized Figure 4. Category count . The pie chart indicates an imbalance in the dataset with a significantly higher proportion of non-offensive data than offensive data.

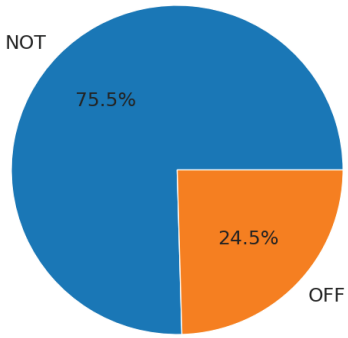


***Figure 4.*** Category count

The dataset's records are translated to Tamil because it contains Tamil terms mixed with English words. Use of this English method determines whether the data contain English words. Prior to and following translation charts are displayed in Figure 5 Dataset Translation rate. This demonstrates how many English words were present before translation and how important English words were changed to Tamil words after translation.

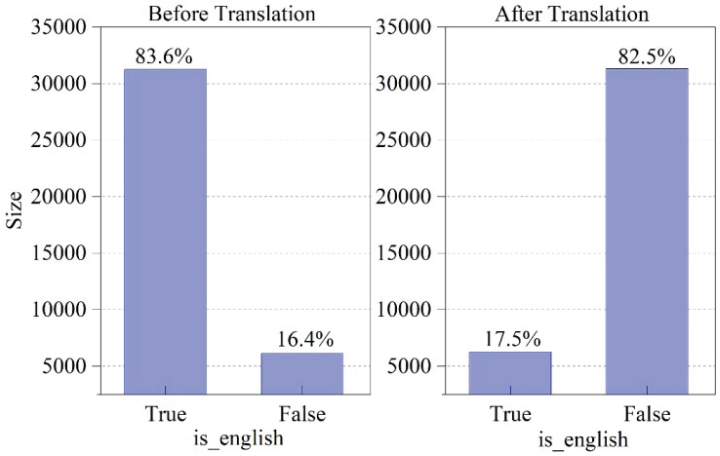


***Figure 5.*** Dataset Translation rate

## *5.4 Feature Extraction*

For feature extraction, there are various techniques available. In this case, Bag-of-Words, TF-IDF, and Word2Cec have been used. These techniques are chosen because the vectorization is based on the context of the comments. The parameter values are fixed only for Word2Vec based on the dataset. For example, vector_size is fixed as 100, window_size as 5, and min_count as 1 for better vectorization.

## *5.5 Dataset Balancing*

Balancing the dataset addresses the issue of imbalance to ensure equitable and effective model training and evaluation. Given the dataset's skewed nature, where the non-offensive class significantly outnumbers the offensive class, an approach known as SMOTE (Synthetic Minority Over-sampling Technique) has been introduced. SMOTE identifies the minority class and, for each data point within it, selects a reference point and a specified number of nearest neighbors based on user-defined parameters. It then generates synthetic data points by combining the reference point and the neighbor using random weights between 0 and 1. These synthetic data points are created along the line segment connecting the reference point and the neighbor in the feature space. This process is repeated for multiple data points in the minority class, resulting in a set of synthetic data points. For the HASOC dataset, random state is fixed as 12, and the targeted value is over-sampled based on the class offensive or non-offensive.

## *5.6 Machine learning Methodologies*

**Bernoulli Naïve Bayes.** The Naive Bayes classifier is a probabilistic classification method that estimates the likelihood of an input belonging to various classes. It's also referred to as a conditional probability approach. A variant of Naive Bayes is the Bernoulli Naive Bayes, tailored for discrete data and based on the Bernoulli distribution. What sets Bernoulli Naive Bayes apart is its exclusive acceptance of binary feature values, such as true or false, yes or no, success or failure, or 0 or 1, with a binary representation here indicating offensive or non-offensive.

$$P(c \mid x) = \frac{P(x \mid c) * P(c)}{P(x)}$$

Where c represents the class label (offensive or non-offensive),

x denotes the features extracted from the comment (e.g., words or n-grams).

$P(c)$ can be estimated from the training data as the proportion of comments belonging to class.

$P(x \mid c)$ is calculated based on the assumption that features are conditionally independent given the class, leading to:

$$P(x \mid c) = \pi_{i=1}^{n} P(x_i \mid c)$$

Where $x_i$ represents the i[th] feature. For binary features, such as the presence/absence of specific words, the probability $P(x \mid c)$ can be estimated using the relative frequency of feature $x_i$ in class c. After computing the posterior probabilities for each class, the comment is assigned to the class with the highest probability.

**Logistic Regression.** Logistic Regression is a widely-used statistical method for binary classification tasks, making it suitable for distinguishing offensive and non-offensive comments. The logistic regression model predicts the probability of a binary outcome (e.g., offensive or non-offensive) based on one or more independent variables (features). The logistic function (sigmoid function) is utilized to map the output of the linear combination of features to the range [0, 1], representing the probability of belonging to the positive class (offensive). The logistic regression model is trained by maximizing the likelihood of observing the actual class labels given the features. This is typically achieved through gradient-based optimization techniques such as gradient descent is show in Figure 6 Logistic Regression.

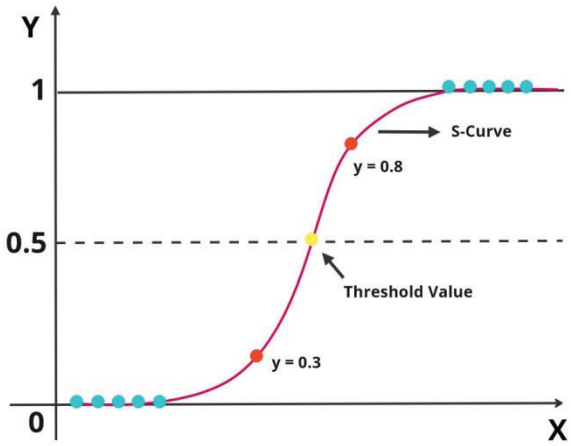


***Figure 6.*** Logistic Regression

The logistic function is defined as:

$$P(y=1 \mid x) = \frac{1}{1+e^{-(w^T X+b)}}$$

Where:

$\boldsymbol{P(y=1 \mid x)}$ is the probability of the positive class (offensive comment) given the features x.

$\boldsymbol{w}$ is the weight vector.

**x** is the feature vector.

$\boldsymbol{b}$ is the bias term.

The output of the logistic regression model is the probability $\boldsymbol{P(y=1 \mid x)}$, which is then used to make predictions. If the probability exceeds a certain threshold (commonly 0.5), the comment is classified as offensive; otherwise, it's classified as non-offensive.

**Support Vector Machine.** Support Vector Machines (SVM) represent a category of supervised machine learning algorithms employed for tasks related to classification and regression is show in Figure 7 Support Vector Machine. SVMs excel in binary classification challenges, and their core concept involves locating a hyperplane within a high-dimensional feature space that effectively segregates data points belonging to different classes. This hyperplane is known as the decision boundary, with its dimension contingent on the number of input features. For instance, when there are two input features, the hyperplane takes the form of a line, while with three input features, it transforms into a 2-D plane. The SVM's primary objective is to maximize the margin, which is the gap between

the decision boundary and the nearest data points from each class. This margin-maximization enhances the model's generalization and resilience.

Mathematically, the decision function for SVM can be represented as:

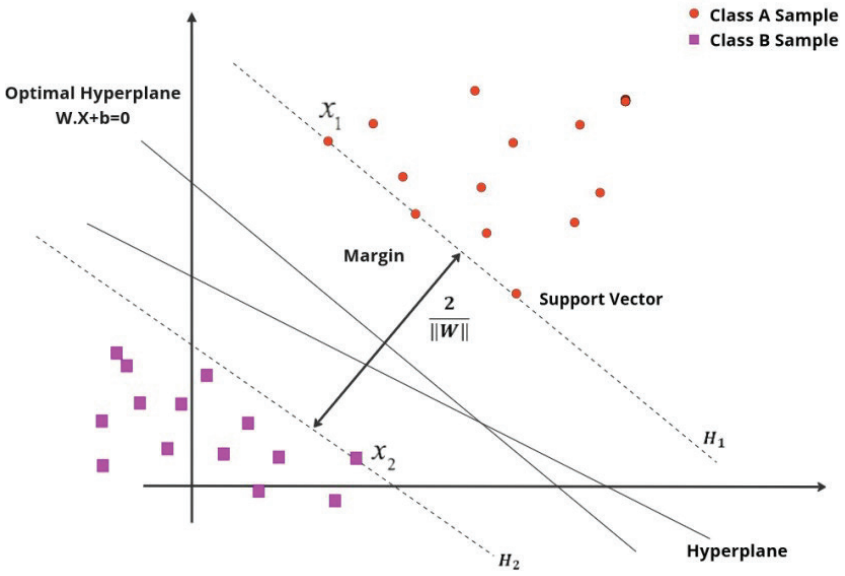


***Figure 7.*** Support Vector Machine

$$f(X) = sign\left(W^T X + b\right)$$

Where:

$f(X)$ represents the decision function.

$W$ is the weight vector.

$X$ is the input feature vector.

$b$ is the bias term.

The goal of SVM is to find the optimal values of w and b that maximize the margin between the support vectors, which are the data points closest to the decision boundary.

This can be formulated as an optimization problem:

$$minimize \frac{1}{2}\|W\|^2$$

Subject to the constraints:

$$y_i\left(W^T X_i + b\right) \geq 1 \, for \; i = 1,...,N$$

Where:

N is the number of training samples.

$X_i$ is the $i^{th}$ training sample.

$y_i$ is the corresponding class label (+1 for offensive, -1 for non-offensive).

SVM can handle linearly separable data, as well as non-linearly separable, by using different kernel functions. For the proposed methodology, linear function is used to map the input features into a higher-dimensional space where separation is possible. Additional parameter probability has been set as true to instruct the SVC to enable probability estimates, allowing it to output probabilities for each class prediction rather than just the predicted class labels.

**K-Nearest Neighbor.** K-NN is a non-parametric method used for classification and regression tasks. It works based on the similarity of data points, where the class of a data point is determined by the classes of its nearest neighbors in the feature space. The principle behind K-NN involves calculating the distance between the query instance (the comment to be classified) and all other instances in the training dataset. The class label of the query instance is then determined by the majority class among its K-NN is show in Figure 8 K-Nearest Neighbor.

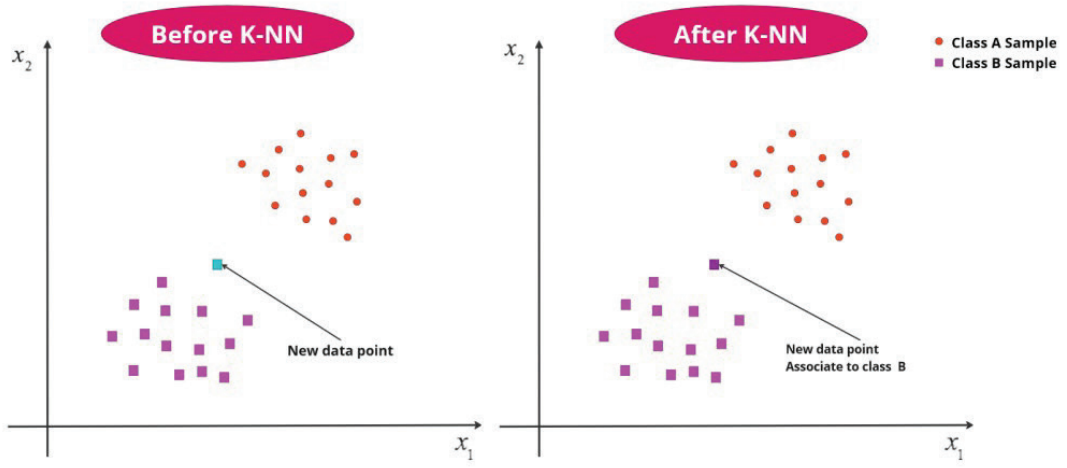


***Figure 8.*** K-Nearest Neighbor

Let $x_q$ represent the query comment, and $x_i$ represent the i[th] comment in the training dataset. Let $y_i$ denote the class label of $x_i$.

Calculate the distance $d\left(x_q, x_i\right)$ between the query comment and each comment $x_i$ in the training dataset.

Select the K comments with the smallest distances to $x_q$, denoted as $\left\{x_{i1}, x_{i2}, ..., x_{iK}\right\}$ The K value is used as 1000 due to the scale of dataset size.

Determine the class of the query comment based on the majority class among its K nearest neighbors: $\hat{y}_q = \arg max_y \sum_{j=1}^{K} I\left(y_i = y\right)$, where $\hat{y}_q$ is the predicted class label for the query comment, $I(\cdot)$ is the indicator function returning 1 if its argument is true and 0 otherwise, and y iterates over all possible class labels.

## *5.7 Deep Learning Methodology*

**Graph Convolutional Network.** Graph Convolutional Networks are employed as a powerful tool for modeling complex relationships within textual data that operate on graph-structured data, where comments can be represented as nodes in a graph, and relationships between comments (e.g., based on co-occurrence or semantic similarity) are represented as edges.

Initially, graph building is the prior step before building the GCN model. The graph can be constructed through various methods, and in this case three techniques are included that are Point-wise Mutual Information, TF-IDF, and Jaccard value.

The propagation rule in GCNs can be expressed as:

$$H^{(l+1)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}D^{-\frac{1}{2}}H^{(l)}W^{(l)}\right)$$

Where:

$H^{(l)}$ is the feature matrix at layer l, where each row represents the feature vector of a node.

$\hat{A}$ is the adjacency matrix of the graph, potentially normalized.

$\hat{D}$ is the degree matrix of $\hat{A}$.

$W^{(l)}$ is the weight matrix at layer l.

$\sigma$ is the activation function, typically ReLU.

The normalization by $\hat{D}^{-\frac{1}{2}}$ ensures that each node's features are scaled by the inverse of its degree, accounting for differences in node connectivity.

In the context of comment classification:

Each comment is represented as a node in the graph.

The adjacency matrix $\hat{A}$ encodes relationships between comments, which could be based on co-occurrence of words, semantic similarity, or other linguistic features.

$H^{(0)}$ represents the initial feature matrix, typically encoding word embeddings or other text representations for each comment.

$W^{(l)}$ are learnable parameters that control the transformation of features at each layer.

After several layers of convolution, the final feature matrix $H^{(l)}$ encodes high-level representations of the comments, capturing complex relationships and semantic structures present in the data. Finally, these learned features can be fed into a classifier (e.g., a softmax layer) for comment classification into offensive and non-offensive categories is show in Figure 9 GCN Work Flow.

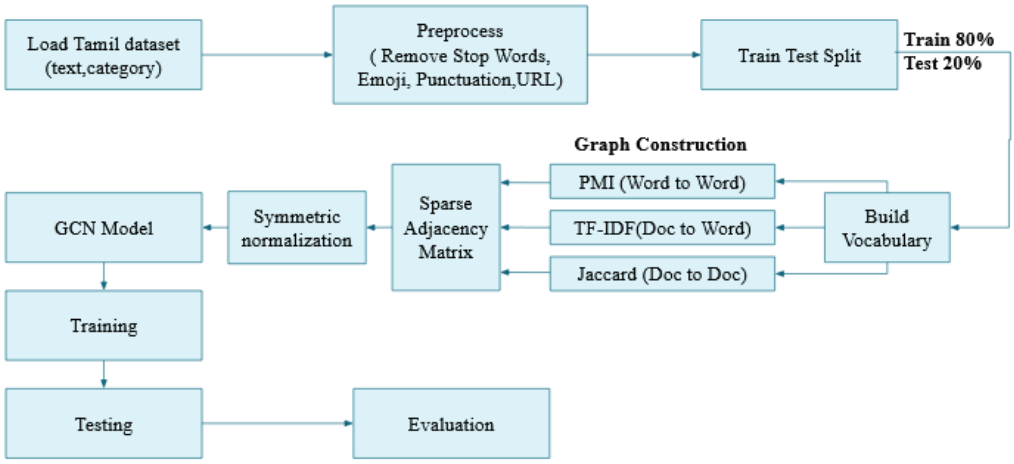


***Figure 9.*** GCN Work Flow

## *5.8 Transformer model*

**mBERT.** Multilingual Bidirectional Encoder Representation for Transformers, an extension of the BERT model, stands out for its ability to comprehend and represent text in multiple languages simultaneously. Its architecture show in Figure 10 mBERT, built upon bidirectional transformers, excels in capturing intricate contextual relationships within textual data. During pre-training, mBERT undergoes two key tasks: masked language modeling (MLM) and next sentence prediction (NSP). The MLM task involves predicting masked tokens within the input sequence, guided by the context provided by the surrounding tokens. Mathematically, the MLM loss is defined as $L_{MLM} = -\sum_{i=1}^{n} \log P(x_i \mid context)$, where $P(x_i \mid context)$ represents the probability of the masked token given its context. Concurrently, the NSP task trains mBERT to understand the relationship between pairs of sentences. Once pre-trained, mBERT is fine-tuned on labeled data specific to our comment classification task. Fine-tuning involves adjusting the model's parameters through backpropagation, optimizing its performance for accurately categorizing comments into offensive and non-offensive classes. The utilization of mBERT in our methodology offers several advantages. Its multilingual capabilities make it well-suited for datasets containing comments in diverse languages, ensuring robust performance across linguistic variations. Additionally, mBERT's proficiency in capturing contextual information enables a nuanced understanding of comments, leading to more accurate classification outcomes. By incorporating mBERT into our methodology, we aim to enhance the effectiveness and efficiency of comment classification, contributing to advancements in natural language processing research and facilitating more effective moderation of online discourse.

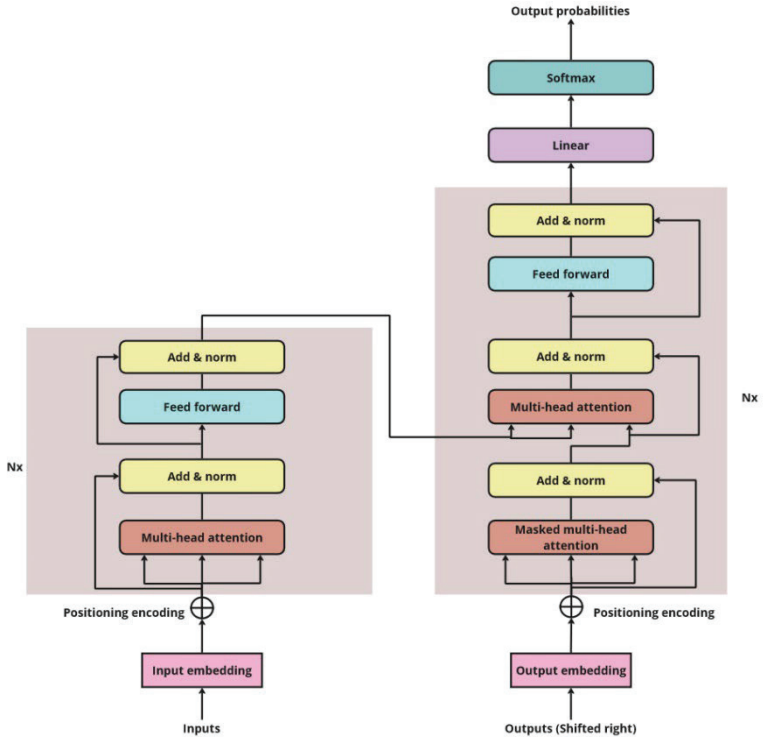


***Figure 10.*** mBERT

## 6 Experimental Results

The performance of different machine learning, GCN and m-BERT models, was evaluated by employing Bag-of-Words (BOW), Term Frequency-Inverse Document Frequency (TF-IDF), and Word2Vec (W2V) embeddings.

In our analysis, we considered several performance metrics, primarily the F1 score and accuracy, to assess the effectiveness of different scenarios. The scenarios we examined encompassed both balanced and imbalanced datasets, with and without translation. The choice to employ the F1 score is motivated by its capacity to balance precision and recall. This is crucial as it aids in minimizing both false positives and false negatives, ensuring a robust evaluation of the models' performance. In the context of both balanced and imbalanced datasets, the F1 score provides a comprehensive measure of how well the models are capable of making accurate predictions. While the F1 score is an informative metric for both balanced and imbalanced datasets, we selectively used accuracy as an additional performance indicator, but exclusively for the balanced dataset. In cases involving imbalanced datasets, accuracy tends to yield misleadingly high performance scores, primarily because it results from systematically predicting only the majority class. Furthermore, in our performance evaluation, we incorporated a crucial constraint: time. The time taken to build the model is a significant aspect of practical usability. We found that SVM consistently consumed more time to construct models across all scenarios. This extended time requirement might be attributed to the inherent complexity of SVM-based modelling techniques. Following SVM, LR and BNB were the next in terms of time consumption. In contrast, the K-NN model stood out as significantly slower, as the time it takes to build is directly influenced by the size of the dataset, which can potentially lead to long processing times in cases involving extensive data. We have also included the concept of ROC, as they are used in performance evaluation, particularly in the context of binary classification problems similar to our goal. They provide valuable insights into the performance of a classifier by illustrating the trade-off between sensitivity (true positive rate) and specificity (true negative rate) across different decision thresholds.

In the evaluation of models for imbalanced datasets, two distinct scenarios emerge. In datasets without translation Table 1 Imbalanced Dataset without Translation, LR, BNB and SVM consistently achieve high F1 scores, peaking at 89%. This underscores their proficiency in aligning with the project's core objective of accurately categorizing offensive and non-offensive comments. The use of Word2Vec embeddings further enriches their nuanced understanding of the Tamil language. However,

the absence of translation presents challenges in achieving robust Kappa values, indicating moderate agreement between predictions and actual classifications. The omission of accuracy as a primary metric is deliberate, as it can overestimate model performance in imbalanced datasets. Turning to datasets with translation Table 2 Imbalanced Dataset with Translation, a similar trend prevails with LR, BNB and SVM. Their performance reflects their linguistic grasp and their capacity to achieve a harmonious trade-off. However, the challenges related to Kappa values and accuracy in imbalanced datasets still persist. TF-IDF and BOW models, while capable of achieving competitive F1 scores, exhibit limitations in terms of Kappa values. This underscores their limitations in precisely capturing the nuances of offensive and non-offensive comments in the Tamil language. In conclusion, the choice of Logistic Regression, Bernoulli Naive Bayes, and Support Vector Machine remains robust, driven by their linguistic expertise and their ability to effectively ensuring accurate comment classification in imbalanced datasets, with or without translation.

***Table 1.*** Imbalanced Dataset without Translation

| Type | Algorithm | F1 Score (%) | Time (s) | Kappa (%) |
|---|---|---|---|---|
| BOW | Bernoulli Naive Bayes | 88 | 0.08 | 27 |
| | Support Vector Machine | 88 | 1281.27 | 45 |
| | Logistic Regression | 89 | 2.35 | 45 |
| | K-Nearest Neighbor | 86 | 0.03 | 0 |
| TF-IDF | Bernoulli Naive Bayes | 88 | 0.07 | 27 |
| | Support Vector Machine | 89 | 469.69 | 47 |
| | Logistic Regression | 88 | 1.4 | 42 |
| | K-Nearest Neighbor | 86 | 0.03 | 1 |
| Word2Vec | Bernoulli Naive Bayes | 74 | 0.13 | 26 |
| | Support Vector Machine | 86 | 747.66 | 0 |
| | Logistic Regression | 86 | 2.57 | 14 |
| | K-Nearest Neighbor | 86 | 0.05 | 4 |

***Table 2.*** Imbalanced Dataset with Translation

| Type | Algorithm | F1 Score (%) | Time (s) | Kappa (%) |
|---|---|---|---|---|
| BOW | Bernoulli Naive Bayes | 86 | 0.35 | 25 |
| | Support Vector Machine | 86 | 4290.71 | 25 |
| | Logistic Regression | 87 | 5.2 | 29 |
| | K-Nearest Neighbor | 86 | 0.06 | 0 |
| TF-IDF | Bernoulli Naive Bayes | 86 | 0.17 | 25 |
| | Support Vector Machine | 87 | 1698.51 | 22 |
| | Logistic Regression | 87 | 2.46 | 28 |
| | K-Nearest Neighbor | 86 | 0.06 | 0 |
| Word2Vec | Bernoulli Naive Bayes | 73 | 0.12 | 26 |
| | Support Vector Machine | 86 | 444.38 | 0 |
| | Logistic Regression | 86 | 4.3 | 6 |
| | K-Nearest Neighbor | 86 | 0.05 | 10 |

Considering time constraints and how word embedding techniques significantly impact model performance, it can be seen that SVM with Word2Vec consistently excels in delivering the highest F1 scores and accuracy Table 3 Balanced Dataset without Translation, aligning well with the objective

and while considering factors like computational resources, training time, and real-world applicability. The Kappa value, which measures agreement between predictions and actual values, corroborated the performance trends observed in F1 scores and accuracy, reinforcing the effectiveness of Word2Vec in delivering robust results. While ROC curves indicate strong discrimination for TF-IDF combined with SVM and Logistic Regression, Word2Vec offers a deeper linguistic understanding. ROC curves primarily emphasize binary classification, which may not fully encapsulate the nuances and linguistic subtleties that Word2Vec can. Even though K-NN took less time in training a model, its performance rate was poor in comparison.

***Table 3.*** Balanced Dataset without Translation

| Type | Algorithm | F1 Score (%) | Time (s) | Kappa (%) |
|---|---|---|---|---|
| BOW | Bernoulli Naive Bayes | 80 | 82 | 0.35 |
| | Support Vector Machine | 82 | 83 | 4290.71 |
| | Logistic Regression | 82 | 83 | 5.2 |
| | K-Nearest Neighbor | 16 | 54 | 0.06 |
| TF-IDF | Bernoulli Naive Bayes | 86 | 85 | 0.17 |
| | Support Vector Machine | 85 | 85 | 1698.51 |
| | Logistic Regression | 82 | 83 | 2.46 |
| | K-Nearest Neighbor | 7 | 52 | 0.06 |
| Word2Vec | Bernoulli Naive Bayes | 62 | 66 | 0.27 |
| | Support Vector Machine | 68 | 72 | 1645.81 |
| | Logistic Regression | 70 | 73 | 7.39 |
| | K-Nearest Neighbor | 68 | 72 | 0.07 |

Here, we observe a diverse range of models for classification Table 4 Balanced Dataset with Translation. SVM with Word2Vec consistently emerges as a standout performer, showcasing profound linguistic understanding. However, it's essential to consider that SVM exhibits a moderately lower Kappa value, indicating a moderate level of agreement. On the other hand, BNB with TF-IDF delivers competitive F1 scores and accuracy, showcasing its ability to capture nuances in the Tamil language. Additionally, the TF-IDF models, including SVM and LR, yield competitive results, underlining the significance of term frequency and linguistic grasp. SVM with TF-IDF demonstrates robust performance, accompanied by a strong Kappa value. BOW models offer promising F1 scores and accuracy, albeit with slightly lower Kappa values, potentially affecting their suitability for precise classification. In contrast, K-NN consistently underperforms, attributed to its reliance on local information and its sensitivity to outliers. While SVM with Word2Vec excels in multiple aspects, BNB with TF-IDF remains a strong contender, providing diverse options for classification.

***Table 4.*** Balanced Dataset with Translation

| Type | Algorithm | F1 Score (%) | Time (s) | Kappa (%) |
|---|---|---|---|---|
| BOW | Bernoulli Naive Bayes | 71 | 73 | 0.17 |
| | Support Vector Machine | 75 | 78 | 2845.36 |
| | Logistic Regression | 75 | 76 | 3.2 |
| | K-Nearest Neighbor | 18 | 54 | 0.06 |
| TF-IDF | Bernoulli Naive Bayes | 76 | 74 | 0.32 |
| | Support Vector Machine | 75 | 76 | 1207.85 |
| | Logistic Regression | 74 | 75 | 1.53 |
| | K-Nearest Neighbor | 26 | 52 | 0.07 |
| Word2Vec | Bernoulli Naive Bayes | 65 | 69 | 0.27 |
| | Support Vector Machine | 67 | 72 | 1912.06 |
| | Logistic Regression | 69 | 72 | 5.97 |
| | K-Nearest Neighbor | 65 | 71 | 0.07 |

The integration of Graph Convolutional Networks (GCN) into our analysis has yielded remarkable outcomes. GCN consistently achieves an F1 score and accuracy of 80% across various dataset configurations, including those with balanced and imbalanced distributions, with and without graph-based feature construction methods such as PMI, TF-IDF, and Jaccard Similarity. This notable performance can be attributed to GCN's unique ability to capture intricate relationships within graph structures, enabling it to discern nuanced patterns indicative of offensive language or sentiment effectively. GCN's iterative feature learning and propagation mechanisms facilitate the extraction of meaningful representations from the input graph, enhancing its understanding of the comment data. Moreover, GCN's interpretability provides valuable insights into influential nodes and relationships within the graph, enhancing transparency and trust in the model's decision-making process. The consistent performance across diverse dataset scenarios underscores GCN's adaptability and effectiveness in comment classification tasks, positioning it as a robust solution for analyzing textual data in a graph-based framework.

The inclusion of mBERT as a model in our analysis has yielded noteworthy results. mBERT consistently achieves an F1 score and accuracy of 89% across various dataset scenarios, including balanced with and without translation, as well as imbalanced with and without translation. This robust performance can be attributed to mBERT's intrinsic capability to comprehend and process multiple languages, including Tamil, with a high degree of accuracy. mBERT's multilingual pre-trained embeddings facilitate a more comprehensive understanding of the Tamil language, allowing it to capture nuances and linguistic subtleties effectively. Furthermore, mBERT's contextual embeddings provide a contextual understanding of the text, which is particularly valuable for our objective where context plays a pivotal role in accurate categorization. The consistent performance across diverse dataset scenarios underscores mBERT's adaptability and effectiveness, making it a strong contender for comment classification in a multilingual context.

## 7 Conclusion

This comprehensive survey explores the crucial domain of offensive language detection within Dravidian languages, underscoring its growing significance in the digital landscape. Innovative methodologies have emerged in response to this pressing issue, fuelled by benchmark datasets like HASOC and Twitter Hate Speech, which have played pivotal roles in shaping research endeavors. Noteworthy advancements in feature extraction techniques, including TF-IDF for contextual awareness, n-grams, and word2vec for nuanced language pattern recognition, have garnered considerable attention. In the realm of machine learning, SVM and K-NN have demonstrated superior predictive performance, while ensemble models such as Bi-LSTM have shown promise in deep learning methodologies. Fine-tuning and optimization strategies, often leveraging grid search techniques, are deemed essential for optimizing deep learning models. Among transformative models, various BERT variants like mBERT, XLM-RoBERTa, and IndicBERT have been explored, with MuRIL and multilingual DistilBERT emerging as standout choices for Dravidian languages, offering a single multilingual model trained on Indian languages for effective offensive content recognition. Additionally, experimental results highlight SVM with Word2Vec, GCN, and mBERT as yielding the best performance across all models tested, emphasizing their efficacy in real-world applications. Further avenues for research could include exploring ensemble techniques incorporating multiple models, investigating domain adaptation methods for improved model generalization, and exploring interpretability techniques to enhance transparency in offensive language detection systems. Aa recommended approach entails leveraging mBERT and MuRIL in conjunction with TF-IDF, Adam optimization, and grid search techniques to bolster the safety of digital spaces, paving the way for more secure online environment.

**References**

[1] Aziz, N. A. A., Maarof, M. A. and Zainal, A. (2021). 'Hate speech and offensive language detection: a new feature set with filter-embedded combining feature selection', in *2021 3rd International Cyber Resilience Conference (CRC)*.

[2] William, P., et al. (2022). 'Machine learning based automatic hate speech recognition system', in *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*.

[3] Jarquín-Vásquez, H. J., Montes-y-Gómez, M. and Villaseñor-Pineda, L. (2020). 'Not all swear words are used equal: Attention over word n-grams for abusive language identification', in *Mexican Conference on Pattern Recognition*.

[4] Chakravarthi, B. R., et al. (2023). 'Offensive language identification in Dravidian languages using MPNet and CNN', *International Journal of Information Management Data Insights*, 3(1), p. 100140.

[5] Dave, B., Bhat, S. and Majumder, P. (2021). 'IRNLP_DAIICT@DravidianLangTech-EACL2021: offensive language identification in Dravidian languages using TF-IDF char n-grams and MuRIL', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*.

[6] Sivalingam, D. and Thavareesan, S. (2021). 'OffTamil@DravideanLangTech-EASL2021: Offensive language identification in Tamil text', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*.

[7] Sharif, O., Hossain, E. and Hoque, M. M. (2021). 'NLP-CUET@dravidianlangtech-eacl2021: Offensive language detection from multilingual code-mixed text using transformers', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, arXiv preprint arXiv:2103.00455.

[8] Ekinci, E., Omurca, S. I. and Sevim, S. (2020). 'Improve offensive language detection with ensemble classifiers', *International Journal of Intelligent Systems and Applications in Engineering*, 8(2), pp. 109–115.

[9] Pathak, V., et al. (2020). 'KBCNMUJAL@HASOC-Dravidian-CodeMix-FIRE20: Using machine learning for detection of hate speech and offensive code-mixed social media', in *The 12th Meeting of the Forum for Information Retrieval Evaluation*.

[10] Mullah, N. S. and Wan Zainon, W. M. N. Z. (2020). 'Advances in machine learning algorithms for hate speech detection in social media: a review', *IEEE Access*, 9, pp. 88364-88376.

[11] Oriola, O. and Kotzé, E. (2020). 'Evaluating machine learning techniques for detecting offensive and hate speech in South African tweets', *IEEE Access*, 8, pp. 21496–21509.

[12] Palanikumar, V., et al. (2022). 'DE-ABUSE@TamilNLP-ACL 2022: Transliteration as data augmentation for abuse detection in Tamil', in *Proceedings of the Second Workshop on Speech and Language Technologies for Dravidian Languages*.

[13] Kumar, R. P., Bhawal, S. and Chinnaudayar, S. C. (2022). 'Hate speech and offensive language detection in Dravidian languages using deep ensemble framework', *Computer Speech & Language*, 75, p. 101386.

[14] Zhu, Y. and Zhou, X. (2020). 'Zyy1510@ HASOC-Dravidian-CodeMix-FIRE2020: An Ensemble Model for Offensive Language Identification', *FIRE (Working Notes)*.

[15] Yao, L., Mao, C. and Luo, Y. (2019). 'Graph convolutional networks for text classification', in *Proceedings of the AAAI Conference on Artificial Intelligence*.

[16] Sreelakshmi, K., Premjith, B. and Kp, S. (2021). 'Amrita_CEN_NLP@DravidianLangTech-EACL2021: Deep learning-based offensive language identification in Malayalam, Tamil and Kannada', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*.

[17] Liu, B., Guan, W., Yang, C., Fang, Z. and Lu, Z. (2023). 'Transformer and Graph Convolutional Network for Text Classification', *International Journal of Computational Intelligence Systems*, 16(1), 161.

[18] Huang, L., Ma, D., Li, S., Zhang, X. and Wang, H. (2019). 'Text level graph neural network for text classification', in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, arXiv preprint arXiv:1910.02356.

[19] Liu, C., Wang, X. and Xu, H. (2022). 'Text Classification Using Document-Relational Graph Convolutional Networks', *IEEE Access*, 10, 123205–123211.

[20] Wang, K., Han, S. C. and Poon, J. (2022). 'Induct-gcn: Inductive graph convolutional networks for text classification', in *2022 26th International Conference on Pattern Recognition (ICPR)*, 1243-1249. IEEE.

[21] Vasantharajan, C. and Thayasivam, U. (2021). 'Hypers@DravidianLang-Tech-EACL2021: Offensive language identification in Dravidian code-mixed YouTube comments and posts', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*.

[22] Ratnavel, R., Selvaraj, S., Vasudevan, P. and Kumar, A. M. (2023). 'Hottest: Hate and offensive content identification in Tamil using transformers and enhanced stemming', *Computer Speech & Language*, 78.

[23] Jayanthi, S. M. and Gupta, A. (2021). 'Sj_aj@dravidianlangtech-eacl2021: Task-adaptive pre-training of multilingual BERT models for offensive language identification', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, arXiv preprint arXiv:2102.01051.

[24] Bhawal, S., Roy, R. P. and Kumar, A. (2021). 'Hate speech and offensive language identification on multilingual code mixed text using BERT', *Working Notes of FIRE 2021-Forum for Information Retrieval Evaluation (Online)*, CEUR.

[25] Kumar, A., Saumya, S. and Singh, J. P. (2020). 'NITP-AI-NLP@HASOC-FIRE2020: Fine Tuned BERT for the Hate Speech and Offensive Content Identification from Social Media', *FIRE (Working Notes)*.

[26] Yasaswini, K., et al. (2021). 'IIITT@DravidianLangTech-EACL2021: Transfer learning for offensive language detection in Dravidian languages', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*.

[27] Garain, A., Mandal, A. and Naskar, S. K. (2021). 'JUNLP@Dravidian-LangTech-EACL2021: Offensive language identification in Dravidian languages', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*.

[28] Zhao, Y. and Tao, X. (2021). 'ZYJ123@DravidianLangTech-EACL2021: Offensive language identification based on XLM-RoBERTa with DPCNN', in *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*.

[29] Chaitanya, B. S. N. V. and Karri, A. P. (2021). 'Transformer Ensemble System for Detection of Offensive Content in Dravidian Languages', *Working Notes of FIRE 2021-Forum for Information Retrieval Evaluation (Online)*, CEUR.