

Article

A Secure Lightweight SMS Spam Detection Framework with Robustness to Text Obfuscation Attacks

Baraa Tareq Hammad ^{1,*}, Ismail Taha Ahmed ^{1,*}, Mohamed A. Hafez ^{2,3} and Betty Wan Niu Voon ⁴

¹ College of Computer Sciences and Information Technology, University of Anbar, Anbar 55431, Iraq

² Faculty of Engineering and Quantity Surveying, INTI International University, Nilai 71800, Malaysia; mohdahmed.hafez@newinti.edu.my

³ Faculty of Management, Shinawatra University, Pathum Thani 12160, Thailand

⁴ College of Engineering, Universiti Tenaga Nasional, Kajang 43000, Malaysia; betty@uniten.edu.my

* Correspondence: baraa.tareq@uoanbar.edu.iq (B.T.H.); ismail.taha@uoanbar.edu.iq (I.T.A.)

Abstract

The proliferation of mobile communications has led to a significant increase in SMS spam, posing challenges related to security, privacy, and user experience. Although numerous machine-learning-based spam detection approaches have been proposed, developing systems that are simultaneously lightweight and resilient to adversarial manipulation remains an open problem. This paper proposes an SMS spam detection framework that incorporates multiple feature extraction methods, including bag-of-words (BoW), Term Frequency–Inverse Document Frequency (TF-IDF), and N-gram models with dimensionality reduction using principal component analysis (PCA), followed by classification using decision tree (DT) and Logistic Regression (LogReg) models. Experimental evaluations on the UCI SMS Spam Collection dataset demonstrate that the TF-IDF-PCA-DT pipeline achieves a detection accuracy of 99% while reducing model size by 77% and inference time by 75%. Robustness evaluation under adversarial text perturbations indicates minimal performance degradation, maintaining an accuracy of 96.5%. These findings demonstrate the practicality of the proposed framework for real-world deployment in resource-constrained environments.

Keywords: SMS spam detection; lightweight security framework; cybersecurity; principal component analysis (PCA); text obfuscation perturbations; Kaggle SMS dataset

1. Introduction

Technology has a huge impact on our daily lives, significantly improving our actions. This process of development is ongoing, and for this reason, we have come to rely on technology to perform tasks in all fields. Great technological developments have enabled us to solve problems with great efficiency compared to past decades, with several improvements to existing infrastructures and services in all areas of life, especially communications [1,2].

This development has led people to rely more on computers and adopt them as reliable resources for carrying out tasks. Previously, communication was carried out via telephones, sometimes over long distances, which exposed it to external factors such as noise, interference, and resistance, affecting sound quality and transmission [3,4]. Afterwards, cell phone technology was introduced as a wireless alternative to a wired service, which contributed significantly and effectively to reducing distance. Short Message Service (SMS) is one of the most important means of wireless communication [5], prompting advertising companies to take advantage of this feature to expose users to their ads in the fastest way to reach



Academic Editors: Suo Gao and Aditya Kumar Sahu

Received: 19 June 2026

Revised: 14 July 2026

Accepted: 14 July 2026

Published: 16 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

the largest number of people [6]. In this context, SMS spam has emerged as a threat and disturbance in the field of cybersecurity, as shown in Figure 1.

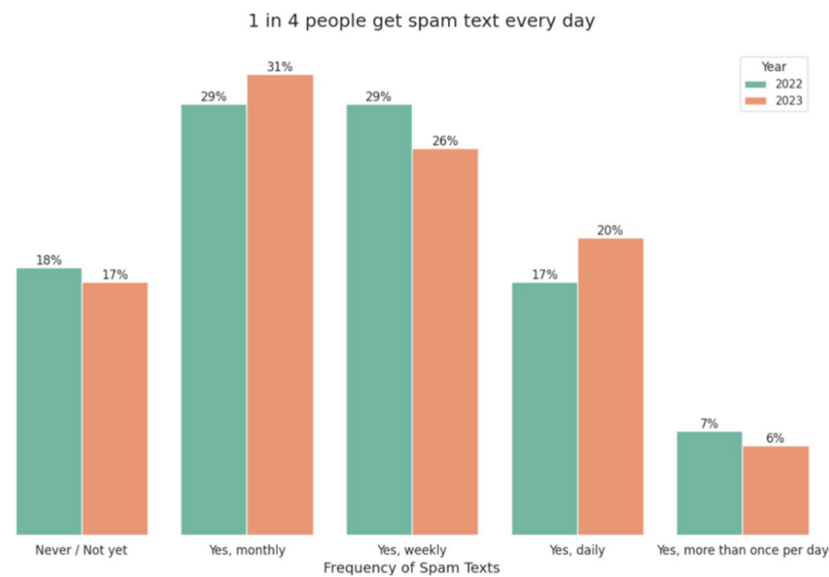


Figure 1. The growth of SMS phishing attacks [7].

Because this field is continuing to develop. Inappropriate messages sent through SMS have a greater effect on consumers than unwelcome emails since SMS customers feel safer using the service in order to confirm payments, transmit secret data, and do other everyday activities. The small letter size of SMS makes it extremely difficult to classify them well, which impacts the accessibility of a large enough SMS dataset for testing and training. Additionally, the effectiveness of current classifiers is impacted by the frequent usage of nonstandard abbreviations (e.g., emotes and punctuation) and unique languages. There are many proposed works that deal with predicting spam messages, but there is still room to develop it and find new ways to distinguish between spam and ham SMS [8,9]. Figure 2 shows the taxonomy of SMS spam detection.

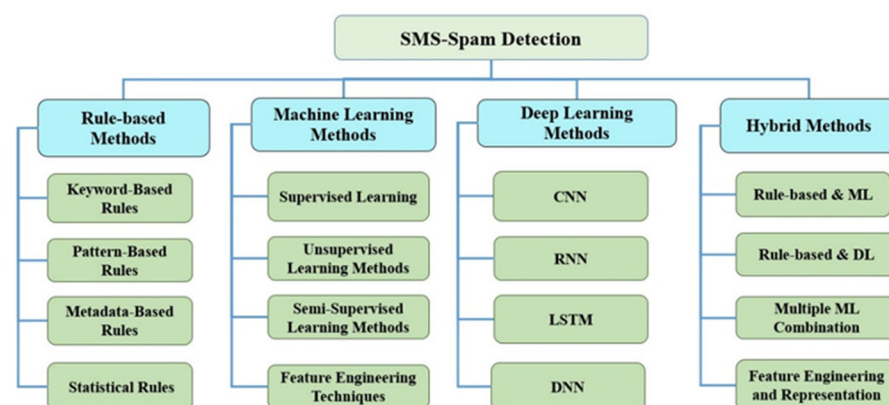


Figure 2. SMS spam detection taxonomy.

It can be challenging to differentiate spam from authentic messages since standard spam filters frequently have trouble identifying recent or changing spam, which is growing progressively more complex. Therefore, automated systems that consistently identify and filter unwanted content are becoming increasingly necessary. To address these challenges, this paper proposes a lightweight and generalizable SMS spam detection framework. First, we took a database, which was pre-processed to clean it using stop-word, punctuation,

and numeric removal, as well as conversion to lowercase. The database was then balanced using oversampling techniques, including the following methods for feature extraction: Term Frequency–Inverse Document Frequency (TF-IDF) [10], bag-of-words (BoW) [11], and N-gram. Then, PCA was used to decrease the dimension of the feature-extracted vector and two classifiers—Decision Tree (DT) and Logistic Regression—were used to classify spam messages as spam or ham.

This work contributes to advancing innovation in cybersecurity for mobile communications by developing lightweight and generalizable SMS spam detection models. It supports the enhancement of secure, resilient infrastructures and inclusive digital communication, helping to improve the reliability and trustworthiness of mobile messaging services.

Our research contributions can be illustrated by the following points:

1. As the results could have been skewed toward the greater class because of the imbalanced dataset, oversampling approaches were employed;
2. Using TF-IDF, bag-of-words, and N-gram techniques created rich, weighted feature sets to feed into machine learning models, enhancing their capacity to differentiate between spam and non-spam messages;
3. We applied PCA to significantly reduce the feature space extracted by TF-IDF, BoW, and N-gram methods, resulting in a compact model without sacrificing accuracy.
4. The combination of TF-IDF feature extraction and PCA enhanced the model's discriminative ability while reducing overfitting, and integrating optimized features with a Decision Tree classifier yielded remarkable performance;
5. We assessed our model's robustness to typical adversarial text disturbances, proving it can continue to operate at a high level even when spam attempts are masked.

The remainder of this paper is organized as follows. Section 2 reviews related works in SMS spam detection. Section 3 details the proposed methodology, including feature extraction, dimensionality reduction, and classification. Section 4 presents the experimental setup and datasets used. Section 5 discusses the results. Finally, Section 6 concludes the paper and outlines future directions.

2. Related Works

Email spam and SMS spam continue to have distinctive features: for instance, since typical text messages are limited to 140 bytes (characters), they have fewer characters than emails [12]. Most recognition techniques concentrate on multi-domain or multi-scale analysis (semantic for text and spatial for images) and are investigated in academia to improve automatic detection models' effectiveness [13]. Many machine learning techniques have been used to automatically detect SMS spam. In this section, we will review some of these studies.

Safie et al. (2018) [14] proposed an SMS spam classification system that employs the vector space model (VSM) and an artificial neural network (ANN). Applying the VSM, which displays text data according to word frequency, SMS messages are initially prepared and converted into numeric feature vectors. In order to differentiate between spam and authentic messages, these vectors are subsequently passed into an ANN classifier. Abid et al. (2022) [1] presented SMS spam detection that makes use of text-based features (TF-IDF) and bag-of-words) as well as machine learning techniques (Naïve Bayes, Support Vector Machines (SVM), and Random Forest). According to the experimental results, the proposed approach successfully raises the accuracy of spam identification, making it a workable alternative for SMS spam filtering. De Luna et al. [15] proposed an SMS spam detection method based on TF-IDF, word embedding, and machine learning. After evaluating several classifiers, the researchers found that the Bernoulli Naive Bayes algorithm performed the best, with an accuracy of 96.63%. The research highlighted constraints relating to dataset

unbalance and the concentration of conventional machine learning models, even if the method showed great accuracy. Dare Azeez Oyeyemi [16] proposed an SMS spam detection method based on some feature extraction techniques, including tokenization, lemmatization, and TF-IDF vectorization. For the best results, a number of machine learning classifiers were trained and evaluated, such as Random Forest, Support Vector Machine (SVM), and Logistic Regression. The Naïve Bayes classifier with the BERT model obtained the greatest accuracy at 97.31%, according to evaluation results. Ballı and Karasoy [17] proposed an SMS spam detection method based on Word2Vec-based feature extraction. They examined a number of machine learning techniques for classification, and Support Vector Machines (SVMs) performed most effectively. The method's successful spam identification by meaningful feature extraction was demonstrated when it was tested on the popular SMS Spam Collection dataset. Baaqeel and Zagrouba [8] proposed an integrated SMS spam filtering method that incorporates many machine learning algorithms and TF-IDF feature extraction. They used the RF and Naïve Bayes algorithms for classification, merging their results in a hybrid way to improve performance. The UCI SMS Spam Collection dataset, which includes labelled SMS messages classified as spam or ham, was used to assess the method. Saeed and Vaman Ashqi [18] proposed an SMS spam detection based on lexical and content-based features. The UCI SMS Spam Collection dataset, which includes labelled SMS messages classified as spam or ham, was used to assess the method. They examined a number of machine learning techniques for classification, and Decision Tree performed most effectively with 98.40% accuracy. Abayo-mi-Alli et al. [2] proposed an SMS spam detection method based on Term Frequency–Inverse Document Frequency (TF-IDF) and word embedding feature extraction methods. The Ex-AIS_SMS dataset, which includes labelled SMS messages classified as spam or ham, was used to assess the method. They examined a number of machine learning techniques for classification, and RNNs performed most effectively. The research emphasizes the success that deep learning has, especially when applied to datasets that are specific to a given location. Gupta et al. [19] proposed an SMS spam detection method based on Term Frequency–Inverse Document Frequency (TF-IDF). The UCI SMS Spam Collection dataset, which includes labelled SMS messages classified as spam or ham, was used to assess the method. They examined a number of machine learning techniques for classification, and SVM performed most effectively in separating authentic texts from spam. The research demonstrates how well TF-IDF characteristics and SVM work together to detect SMS spam. Dharrao et al. [20] proposed an SMS spam detection method based on Term Frequency–Inverse Document Frequency (TF-IDF) and bag-of-words feature extraction. The UCI SMS Spam Collection dataset, which includes labelled SMS messages classified as spam or ham, was used to assess the method. They examined a number of machine learning techniques for classification, and SVM achieved the highest accuracy with 97.31%. Ahmadi et al. [21] proposed an SMS spam detection method based on bag-of-words and TF-IDF feature extraction methods. They tested six multiple machine learning algorithms for categorization in order to find which was the most efficient mix for detecting spam information. Using standard SMS Spam datasets, such as the popular UCI SMS Spam Collection, the authors assessed their methodology. Taylor and Robert [22] proposed an SMS spam detection based on lexical and syntactic features, such as word N-grams and character-level patterns. After evaluating several classifiers, they found that Random Forest and Support Vector Machine (SVM) performed well. A recently assembled Chichewa SMS dataset was used to train and test the method. The research emphasizes the value of native-language datasets while pointing out issues with the absence of language-specific instruments, the short dataset size, and the quality of translation. Table 1 displays a summary of all of these works.

Table 1. Summary of various research publications.

Year/Ref.	Feature Extraction Method	Classification Methods	Accuracy (%)	Dataset	Limitations
2019, [17]	Word2Vec	Random Forest (RF)	99.64	UCI	Limits of the model generalizability
2020, [8]	Word Tokenization	SVM	98.8	UCI	Limit adaptability and scalability due to manual pre-processing used
2021, [19]	TF-IDF	Naive Bayes (NB)	96.5	Spam Assassin	Limits to independence assumption inherent in the Naive Bayes algorithm
2022, [2]	Word2Vec	Long Short-Term Memory (LSTM)	98.6	Indigenous	Limits of the model scalability and transferability
2022, [1]	TF-IDF and bag-of-words	SVM, RF, NB, and Logistic Regression	99	UCI	Weak contextual comprehension
2023, [18]	Word Embedding	J48, KNN, and DT	98.40	UCI	Limits the generalizability of the model
2023, [15]	Bag-of-Words	Bernoulli NB	96.63	Self-Acquired SMS Messages	Limit of cross-dataset evaluation
2024, [16]	Contextual Sentence Embedding	NB and BERT	97.3	Kaggle's and DSN	Limits of the model generality
2024, [20]	TF-IDF	SVM, NB, RF, Logistic Regression	97.31	UCI	Limits the generalizability of the model
2025, [22]	TF-IDF	RF and Logistic Regression	97	Chichewa	Limits the generalizability of the model
2025, [21]	Bag-of-Words and TF-IDF	NB, KNN, SVM, DT, LDA	96.2	UCI	Performance of a biased classifier

3. Proposed Methods

This section details the overall architecture of the proposed SMS spam detection framework, emphasizing its compact and efficient design and robustness to adversarial modifications. The system follows a modular approach comprising pre-processing, feature extraction, dimensionality reduction, classification, and evaluation, as illustrated in Figure 3.

3.1. Dataset Collection Phase

The dataset, consisting of English-language text messages, is used in the experimental attempts. The dataset can be downloaded for free through the UCI library. The source of these text messages was an open forum in the United Kingdom. A full set of 5574 SMS messages is included in the sample; 747 of them are categorized as spam, and 4827 are categorized as ham.

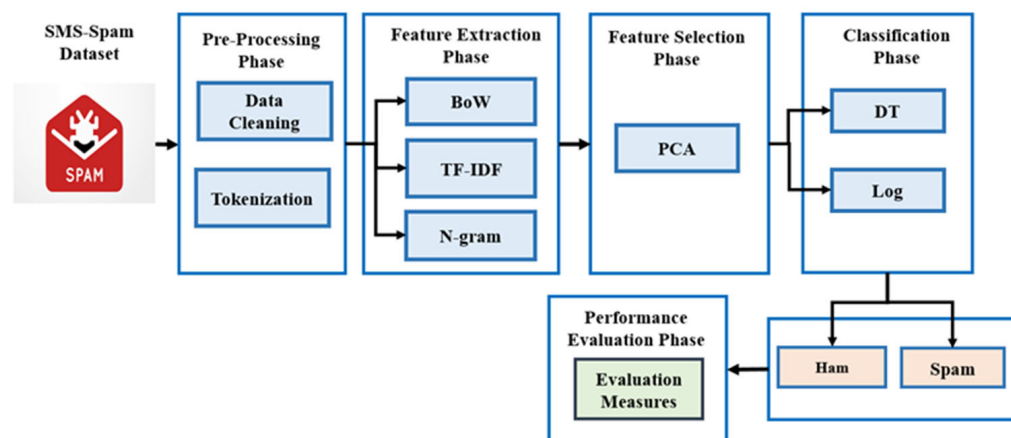


Figure 3. Overview of the proposed SMS spam detection framework.

3.2. Pre-Processing Phase

In order to guarantee that the information being collected is clean, consistent, and prepared for modelling, pre-processing is a crucial phase in SMS spam detection. In the end, this improves the performance of any model trained on the collected information by guaranteeing that the information is in an accessible form with no signs of noise [23]. It enhances the classification models' performance. The standard pre-processing procedures applied to SMS spam detection are broken down as follows:

3.2.1. Data Cleaning

We used common text pre-processing methods in the data cleaning step, such as truncation, lemmatization, lower-casing, punctuation removal, and the elimination of empty words, URLs, and unnecessary symbols. Figure 4 illustrates how to remove unwanted parts from text input, including unique characters, HTML tags, URLs, spaces, and numbers.



Figure 4. Example of raw SMS vs. cleaned SMS.

3.2.2. Tokenization

The procedure of dividing a text message into smaller segments called tokens is known as tokenization. These tokens usually stand for particular words, phrases, or characters in SMS spam detection. In order to prepare text for feature extraction and machine learning, tokenization is essential. By collecting important text properties, efficient tokenization guarantees that machine learning models can correctly detect spam patterns. It helps detect spam by bridging the gap between raw SMS data and insightful information. Figure 5 illustrates the various tokenization techniques: Word Tokenization, Character Tokenization, Subword Tokenization, and N-gram Tokenization. These techniques partition the text into words, characters, subwords, and sequences of n consecutive words, respectively.

Following that, the cleaned and tokenized text is converted into numerical features using bag-of-words (BoW), Term Frequency–Inverse Document Frequency (TF-IDF), and N-gram.

3.3. Feature Extraction Phase

A crucial stage in SMS spam detection is feature extraction, which converts processed text into numerical representations that contain appropriate information for models to be

classified. Robust spam identification is made possible by these features, which can record the lexical, semantic, syntactic, and statistical characteristics of messages sent through SMS. Popular feature extraction approaches that convert tokens into numerical feature vectors that can be applied to classification models include bag-of-words, TF-IDF, word embeddings, and N-gram features.

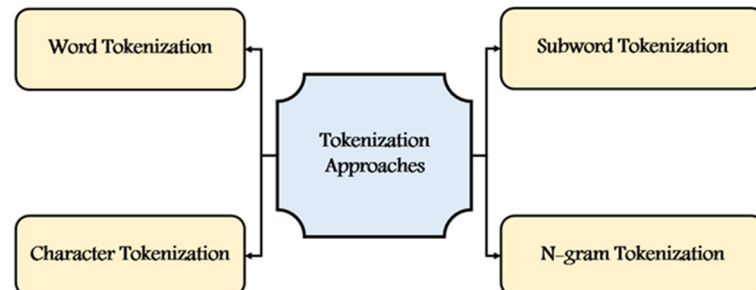


Figure 5. Common tokenization techniques.

3.3.1. Bag of Words (BoW)

By calculating the number of times a word appears in a document, bag-of-words (BoW) converts text into its numerical form while retaining vocabulary and ignoring syntax and word sequence. Algorithm 1 provides a summary of the phases in the BoW approach. Large vocabulary sizes might result in feature vectors that are sparse and computationally costly. The arrangement and content of words or features are not taken into account by BoW.

Algorithm 1: Bag-of-Words (BoW) Feature Extraction

Input: **Pre-processed text documents** $D = \{d_1, d_2, \dots, d_n\}$.

Output: **Bag-of-Words feature matrix** B .

Begin

For

1. Import the pre-processed text.
2. Vocabulary Development:
 - 2.1 Construct an individual corpus out of all the documents.
 - 2.2 To build vocabulary $V = \{v_1, v_2, \dots, v_m\}$, one must take special terms out of the corpus.
3. Initialize the Feature Matrix:
 - 3.1 Construct an $n \times m$ matrix.
 - 3.2 Set all the matrix's elements to zero.
4. Generate BoW Features:
 - 4.1 For each document d_i in D :
 - 4.1.1 For each word w in d_i :
 - a. Find the index j of w in the vocabulary V .
 - b. Increment $B(i, j)$ by 1.
5. Construct the Final Representation:
 - 5.1 Each row of B represents a document.
 - 5.2 Each column of B represents a vocabulary term.
 - 5.3 Each matrix value indicates the frequency of a term in a document.

End for

6. Return the Bag-of-Words feature matrix B .

End

3.3.2. Term Frequency–Inverse Document Frequency (TF-IDF)

A popular method for text feature extraction that finds keywords or phrases in a text corpus is the TF-IDF [24]. The Term Frequency (TF) multiplied by the Inverse Document Frequency (IDF) produces the TF-IDF value. IDF assesses the word's rarity throughout the SMS corpus, whereas TF gauges the word's repetition inside the text of the message. To make things more accurate, the logarithm of the proportion between the overall number of SMS messages in the corpus and the overall number of messages that include the word is used to calculate IDF. The following is the TF-IDF Equation (1). Algorithm 2 provides a summary of the phases in the TF-IDF approach.

$$W(d, t) = tf(d, t) * \log(N / (df(t))) \quad (1)$$

The TF-IDF value of a token t in sentence d is denoted by a variable called $W(d, t)$. The frequently occurring value at which a token t appears in sentence d is denoted by the variable $tf(d, t)$. The number of sentences in the dataset is denoted by the variable N . The number of sentences in the dataset that contained the token t is denoted by the variable $df(t)$.

Algorithm 2: TF-IDF Feature Extraction

Input: Pre-processed Text $D = \{d_1, d_2, \dots, d_n\}$.

Output: TF-IDF feature matrix F .

Begin

For

1. Import the pre-processed text.

2. Construct the vocabulary set V from all unique terms in the dataset.

3. For each document $d \in D$, perform the following steps:

3.1 Calculate the Term Frequency (TF) for each term t in document d :

$TF(t, d) = \text{Number of times term } t \text{ appears in document } d / \text{Total number of terms in document } d$

3.2 Calculate Document Frequency (DF) for each term t in document d :

Count the number of documents containing term t .

3.3 Calculate Inverse Document Frequency (IDF) using the formula: $I(t, D) = \log(\text{Total number of documents in } D / \text{Number of documents with term } t \text{ in } D)$

3.4 Combine TF and IDF to calculate the final TF-IDF using the formula:

$TF-IDF(t, d, D) = TF(t, d) \times IDF(t, D)$

3.5 Represent document d as a numerical feature vector using the TF-IDF weights of all terms in V .

4. Construct the document-term matrix F , where

4.1 Show every document as a vector, with each dimension representing the TF-IDF score of a phrase.

4.2 A document-term matrix is the end product, where words are represented by columns and documents by rows.

End for

5. Return the set of TF-IDF feature matrix F .

End

3.3.3. N-Gram

A popular method in text analysis for representing word or character sequences in a dataset is N-gram feature extraction. It entails segmenting text into consecutive groups of n elements (words, letters, etc.). Because it may be used at the word or character level, it

works for a variety of languages and jobs. High-dimensional feature spaces may result from the extraction and management of huge N-grams, which leads to three issues: (i) Feature redundancy arises from the fact that numerous N-grams are classified as distinct features despite representing overlapping or strongly correlated context-specific information. (ii) Overfitting is the result of machine learning algorithms memorizing uncommon and unique dataset-specific N-grams rather than picking up reliable linguistic patterns. (iii) Higher memory consumption, longer training, and more sophisticated models are all examples of higher costs associated with computation. Algorithm 3 provides a summary of the phases in the N-gram approach.

Algorithm 3: N-gram Feature Extraction

Input: **Pre-processed Text D** = {d1, d2, ..., dn}

Output: **Numerical Feature Vector F.**

Begin

For

1. Import the pre-processed and tokenized text documents.

2. Initialize:

NGram_List ← empty list

Vocabulary ← empty set

3. For each document $d \in D$ do

3.1 Generate N-grams using a sliding window approach.

3.2 For each generated N-gram g do

3.2.1 Add g to NGram_List.

3.2.2 Update the frequency count of g .

3.3 Store all extracted N-grams for document d .

End For

4. Vocabulary Construction:

4.1 Combine all extracted N-grams from all documents.

4.2 Extract unique N-grams to form the vocabulary V .

5. Feature Representation:

5.1 For each document $d \in D$ do

5.1.1 Count the occurrences of each N-gram in V .

5.1.2 Represent d as a numerical feature vector.

End for

5.2 Store all feature vectors in F .

6. Return the set of N-gram feature vectors F .

End

3.4. Feature Selection Phase

Feature selection is critical to reducing the dataset's total dimension by retaining valuable information and removing redundant and superfluous information. Principal component analysis (PCA) is employed since it is a popular and practical method for feature selection. For representations of data with spaces that are highly dimensional, PCA is more appropriate. The initial process in PCA is to convert the source data into another set of parameters that are linearly mixed with the initial variables, allowing only the variables with the greatest importance to be chosen [25]. PCA operates in four stages. After the data has been normalized, a covariance matrix is created, the eigenvectors and eigenvalues are calculated, and the eigenvectors are arranged in a decreasing sequence [26]. By distilling data to its most valuable elements, noise is removed. Additionally, it shrinks the feature space, which speeds up models and lessens their susceptibility to overfitting.

Algorithm 4 provides a summary of the phases in PCA-Based Feature Selection. Following the selection of the best characteristics, the selected extracted features were used in the classification stage.

Algorithm 4: PCA-Based Feature Selection

Input: Feature vector $F = \{f_1, f_2, \dots, f_n\}$

Number of principal components k

Output: Reduced feature set Y

Begin

1. Normalize the Data:
Standardize each feature using: $Z = (X - \mu) / \sigma$
2. Compute the Covariance Matrix using: $C = (1 / (n - 1)) \times Z^T Z$
3. Compute Eigenvalues and Eigenvectors using $Cv = \lambda v$
4. Sort eigenvalues and corresponding eigenvectors: $\lambda_1 > \lambda_2 > \lambda_3 > \dots > \lambda_m$
5. Form the projection matrix: $P = [v_1, v_2, v_3, \dots, v_k]$
6. Project data onto the reduced feature space: $Y = ZP$
7. Obtain Reduced Features: Y represents the reduced-dimensional feature set.

End

3.5. Classification Phase

These features are used to train various classifiers to recognize SMS spam. According to the literature analysis, many experiments with different classifiers produced better accuracy outcomes. In order to differentiate between spam and ham in the current paper, we used Logistic Regression and DT as effective techniques for SMS spam detection.

3.5.1. Logistic (LogReg)

A popular classification approach called Logistic Regression uses a linear decision boundary to distinguish between spam and ham SMS. It uses the sigmoid function, which returns numbers between 0 and 1, to model the probability of an SMS that defines a class. Classification is based on the decision threshold: SMS with a probability higher than 0.5 is categorized as spam, whereas those with a probability less than 0.5 are classed as ham. Because of the technique's simplicity, interpretability, and suitability for binary classification tasks, Logistic Regression was selected as the fundamental classification technique [27].

3.5.2. Decision Tree (DT)

An effective and comprehensible machine learning model is a Decision Tree (DT) classifier [28]. In the end, it creates a structure similar to a tree with every node inside representing a decision rule, every link representing the result of the rule, and every leaf node representing a class label (spam or ham) by iteratively dividing the data according to feature values. Because of its ease of use and capacity to represent non-linear relationships, it is frequently employed for classification problems, such as SMS spam detection. It manages both numerical and categorical feature types and operates with unscaled data. Even on quite big datasets, they may be trained rapidly and with high computational efficiency.

4. Experimental Setup

The following four main elements of the findings from experiments will be thoroughly covered in the following section: the experimental environment, the datasets employed, the evaluation parameters, and the validation strategy.

4.1. Experimental Environment

As shown in Figure 6, a number of specific instructions were closely adhered to in order to carry out the experiment.

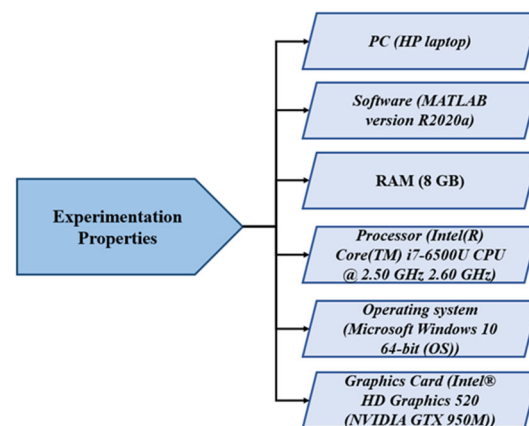


Figure 6. Summary of experimental parameters.

4.2. Dataset

The dataset, consisting of English-language text messages, was used in the experimental attempts. The dataset can be obtained for download from the repository owned by UCI [29,30]. The source for the above text messages was a public message board in the United Kingdom. Out of the 5574 SMS messages in the publicly available dataset, 747 are categorized as spam and 4827 are categorized as ham (not spam).

There is an imbalance in the UCI SMS Spam dataset. In order to overcome this problem, we used oversampling techniques to alter the initial data and produce a more evenly distributed allocation of “ham” and “spam” occurrences. Ultimately, there was going to be a rise in the minority class proportion. The primary advantages of oversampling are increasing the sample size, producing more features for model training, and improving the accuracy of the model. The primary method of oversampling involves selecting smaller groups at arbitrary points and determining each smaller class’s K-close neighbour. To create a new minority class at that specific time, the chosen samples were evaluated using the K-nearest neighbour.

4.3. Evaluation Parameters

The proposed approach was tested using a variety of assessment factors, including accuracy, precision, recall, and F1-score approaches [31,32], to ensure the reliability of the findings. The mathematical formulas and descriptions for the above measurements are shown in Table 2. To identify the most effective feature extraction methods and classification methods for our situation, we evaluated the results. Figure 7 describes all the terms that can be found using the previously provided equations.

4.4. Validation Strategy

To be more accurate, we used the 10-fold cross-validation method [33], making sure that every fold preserved the distribution of classes. This approach provided a robust assessment of the proposed SMS spam detection model using all samples for both training and validation across different iterations. The mean results derived from a 10-fold cross-validation serve as a basis for the findings reported in this research. Exclusively inside the training data of every cross-validation fold, oversampling and PCA pre-processing processes were implemented. Oversampling was conducted on the training set to address class imbalance, and PCA was trained exclusively on the training data and subsequently

applied to the validation set without refitting. This ensured there was no data leakage and retained the validity of the evaluation. This method offers a more thorough evaluation of the suggested SMS spam detection strategy and enables consistent validation. To evaluate robustness of the proposed method, text obfuscation perturbations were made to the test set by randomly swapping out characters and adding special symbols in spam messages. The performance of the model was then evaluated with regard to these changes. Performance metrics were calculated to evaluate the performance when the model trained on the UCI dataset was deployed straight to the Kaggle dataset without retraining.

Table 2. An overview of the assessment metrics.

Metric	Description	Formula	Equation No.
Accuracy	It represents the proportion of accurately recognized items to all items entered.	$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$	(2)
Precision	The percent of all given images that the classification algorithm properly identified is represented by this number.	$Precision = \frac{TP}{TP+FP}$	(3)
Error	This is the proportion of every image to the overall number of images that have been mislabelled.	$Error = \frac{FP+FN}{TP+TN+FP+FN}$	(4)
Recall	It represents the proportion of all images belonging to that category that have been accurately classified.	$Recall = \frac{TP}{TP+FN}$	(5)
F1-Score	A common way to think of the F1-score is to consider it a weighted average of recall and precision.	$F1-Score = 2 * \frac{Precision * Recall}{Precision + Recall}$	(6)

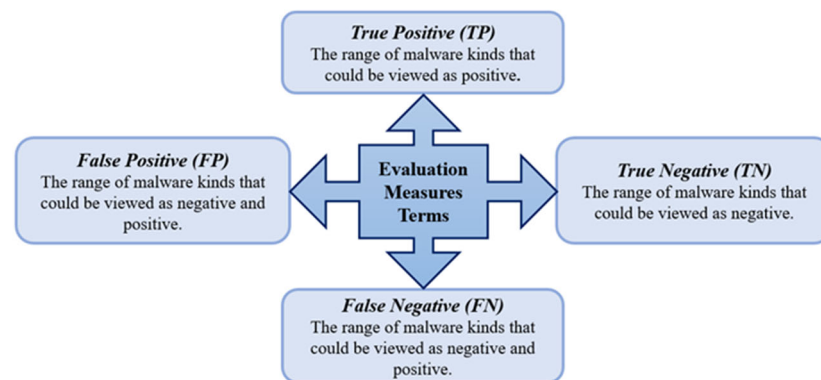


Figure 7. A detailed description of every term.

5. Results and Discussion

The experimental findings of the proposed SMS spam detection framework are shown in this section. Accuracy, precision, recall, F1-Score, model size, inference time, and robustness to text obfuscation perturbations are the metrics used to assess performance. To illustrate the trade-offs between various feature extraction methods and classifiers, comparison assessments are offered. The presented results (included in Tables 3 and 4) examine how well different classifiers and feature extraction techniques perform on a balanced sample of SMS spam. Accuracy, error rate (ER), F1-Score, precision, and recall are among the assessment metrics that are used to break down the observations using feature extraction techniques and classifiers.

Table 3. Evaluation measure results for different feature extraction methods and classifiers over the balanced SMS Spam dataset.

Classifier	BoW_7275 Features Evaluation Measures (%)					Feature Extracted Dimension TF-IDF_7275 Features Evaluation Measures (%)					N-Gram_27369 Features Evaluation Measures (%)				
	Accuracy	Error	F1-Score	Precision	Recall	Accuracy	Error	F1-Score	Precision	Recall	Accuracy	Error	F1-Score	Precision	Recall
DT	97	3	97	96	100	99	1	99	99	99	97	3	89	94	84
Log	95	5	95	98	92	98	2	98	96	100	90	10	94	100	90

Table 4. Evaluation measure results for different feature extraction methods and classifiers over the balanced SMS Spam dataset using feature selection.

Classifier	BoW_PCA_1248 Features Evaluation Measures (%)					Feature Extracted Dimension TF-IDF_PCA_1976 Features Evaluation Measures (%)					N-Gram_PCA_2702 Features Evaluation Measures (%)				
	Accuracy	Error	F1-Score	Precision	Recall	Accuracy	Error	F1-Score	Precision	Recall	Accuracy	Error	F1-Score	Precision	Recall
DT	98	2	98	97	100	99	1	99	98	100	97	3	97	96	100
Log	96	4	96	99	94	98	2	98	99	97	93	7	93	99	87

A number of observations can be seen in Table 3: The Decision Tree (DT) classifier's accuracy, which ranges from 97% to 99%, is generally excellent among all feature extraction techniques, as shown in Figure 8a. The accuracy for the BoW and TF-IDF approaches (97% and 99%, respectively) is matched by the F1-Score, as shown in Figure 8a,b. However, recall (84%) and precision (94%) significantly decrease when N-gram features are used, as shown in Figure 8c,d. With BoW as the feature extraction method, the Logistic Regression (Log) classifier performs slightly less well than DT (accuracy 95%), while with the N-gram feature extraction method, it significantly declines (accuracy 90%), as shown in Figure 8a. For Log, the TF-IDF feature extraction method yielded the best results, with 98% accuracy and F1-Score.

Some of the conclusions are shown in Table 4: When incorporating PCA, the Decision Tree (DT) classifier's accuracy increases marginally, hitting 99% for the TF-IDF feature extraction method and 98% for the BoW feature extraction method, as shown in Figure 9a. The N-gram feature extraction method performs comparably to the unoptimized model (97% accuracy). All feature extraction methods have consistently high precision and recall (96–100%), as shown in Figure 9c,d. Regarding the classifier known as Logistic Regression (Log), it reaches 98% accuracy and F1-Score with PCA, demonstrating a steady increase in performance, especially for the TF-IDF feature extraction method, as shown in Figure 9a,b. Once more, the N-gram feature extraction method performs worse than other approaches, with a noticeable decline in recall (87%), as shown in Figure 9d.

For both classifiers, it is evident that the TF-IDF features continuously perform better than the BoW and N-gram approaches with respect to accuracy, F1-Score, and precision. This leads to the conclusion that the optimum choice is the TF-IDF feature extraction approach. However, as demonstrated by declines in recall and precision, a large number of N-gram features may result in overfitting and inconsistency in results, particularly when used with Logistic Regression.

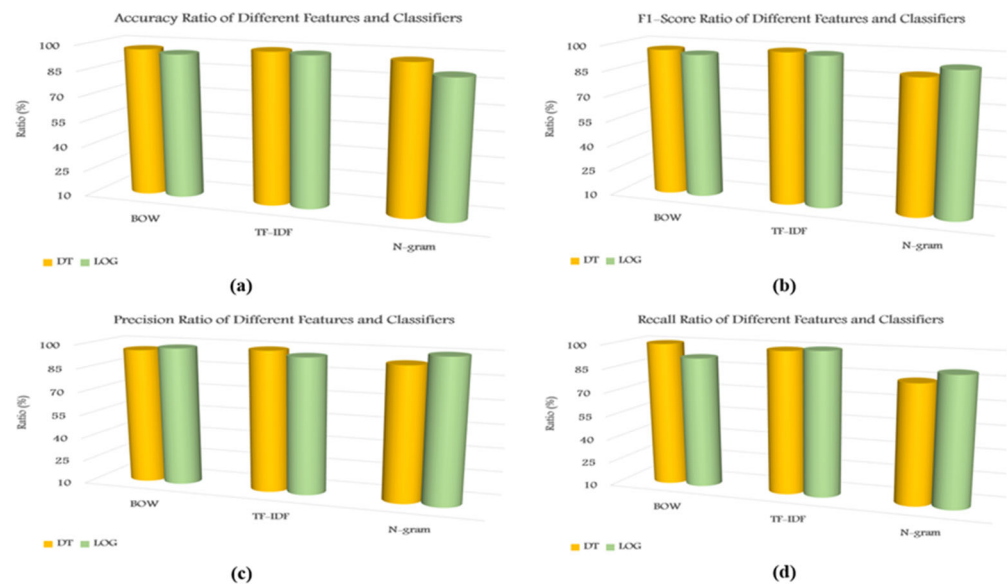


Figure 8. Evaluation measure results for different feature extraction methods and classifiers over the balanced SMS Spam dataset: (a) accuracy measure; (b) F1-Score measure; (c) precision measure; (d) recall measure.

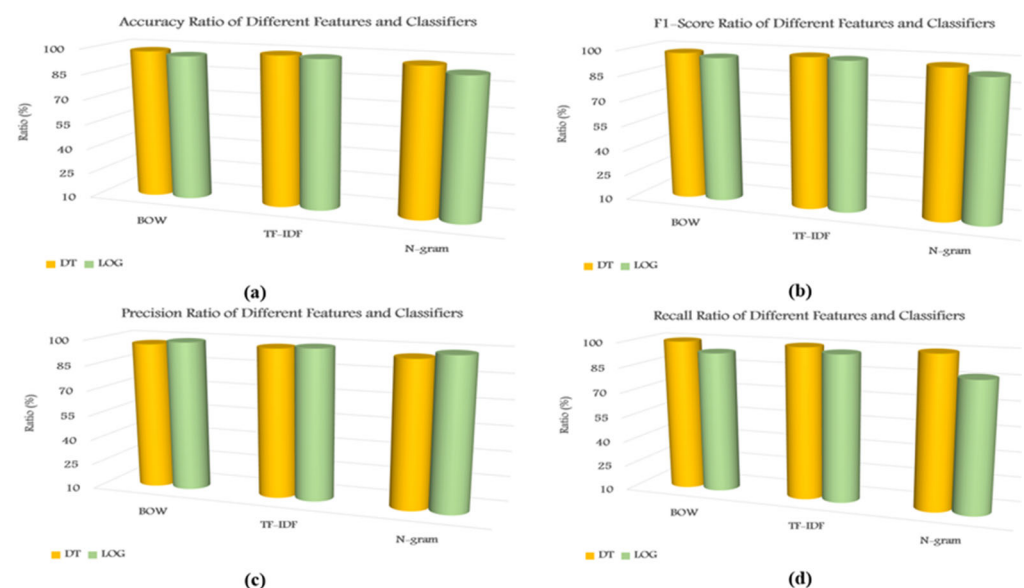


Figure 9. Evaluation measure results for different feature extraction methods and classifiers over the balanced SMS Spam dataset using feature selection: (a) accuracy measure; (b) F1-Score measure; (c) precision measure; (d) recall measure.

Using PCA for feature selection significantly improved model performance by lowering feature dimensions and raising evaluation measures such as accuracy, precision, and recall among the two classifiers. This trend concludes that the TF-IDF_PCA emerges as the most robust feature set, especially with Decision Tree, and regularly outperforms the others.

5.1. Compact and Efficient Design Analysis

We verify that the framework's lightweight properties are true. This is accomplished by comparing model size and inference time before and after using the technique of principal component analysis (PCA). The results are shown in Table 5. By using PCA, the average inference time per message is reduced by 75% and the model size is reduced by 77% without sacrificing classification accuracy. These outcomes show that the architecture is

appropriate for implementation on devices with limited resources, like embedded systems and cellphones.

Table 5. Lightweight performance analysis of the proposed (TF-IDF + DT) method.

Property	Without PCA	With PCA
Model size (MB)	18.5	4.2
Inference Time (ms/message)	8.4	2.1

5.2. Robustness Against Adversarial Text Perturbations

The proposed method is evaluated for its robustness against common spam evasion strategies (e.g., random character substitutions and the insertion of special symbols), rather than against optimization-based adversarial attacks. A comparison will be made between clean and adversarial-altered datasets. The performance under both clean and adversarial perturbed settings is shown in Table 6. The experimental results remain unchanged and demonstrate that the proposed model maintains strong performance under these realistic spam obfuscation scenarios, with accuracy decreasing only from 99% to 96.5% and an F1-Score of 95.5%.

Table 6. Robustness performance under adversarial text perturbations.

Measure (%)	Clean Data	Adversarial Data
Accuracy	99	96.5
Precision	99	95
Recall	99	96
F1-Score	99	95.5

5.3. Additional Evaluation Results

The model trained on the UCI SMS Spam Collection dataset was also evaluated using the Kaggle-hosted version of the same SMS Spam Collection dataset without retraining. The corresponding performance results are presented in Table 7. The proposed approach achieved an accuracy of **95.2%** and an **F1-Score of 95.0%**, indicating strong performance in SMS spam classification. These results are consistent with the findings from the other experimental evaluations presented in this study and further demonstrate the effectiveness of the proposed approach under the reported experimental settings.

Table 7. Supplementary evaluation results.

Measure (%)	Kaggle Dataset
Accuracy	95.2
Precision	94
Recall	96
F1-Score	95

5.4. Comparison

This section evaluates the effectiveness of the proposed approach by comparing it with various other methods. The majority of the techniques that employ both bag-of-words (BoW) and TF-IDF feature extraction techniques, together with the UCI dataset employed during the testing process, are chosen for a fair comparison. The findings of the different spam detection techniques that will be compared are shown in Table 8.

Table 8. Comparing performance with previous approaches.

Reference	Techniques	Feature Selection	Classifier	Dataset	Accuracy (%)
[34]	TF-IDF	N/A	RF	UCI	97
[35]	BoW	N/A	Hidden Markov Model (HMM)	UCI	96.9
[20]	BoW	N/A	DT	UCI	96
[36]	TF-IDF	N/A	RF	KaiDMML	77.8
[37]	TF-IDF & BoW	N/A	Logistic	SMS	91.8
<i>Proposed</i>	TF-IDF	PCA	DT	UCI	99

Considering a variety of feature extraction methods, classifiers, and datasets, we compared multiple spam detection techniques based on the data presented in Table 8. The majority of earlier studies may have retained the extracted features because they did not use a feature selection method. Nonetheless, the proposed method makes use of PCA, which probably aids in dimensionality reduction, enhancing classification performance and processing efficiency.

Two mentioned approaches [34,36] employ Random Forest (RF) as their classifier, with accuracy ratings of 97%, 97.5%, and 77.8%, respectively. Random Forest is appropriate for SMS spam detection since it is capable of handling a variety of features and is resistant to overfitting. The fluctuations in RF's performance, particularly the decline to 77.8% on KaiDMML, show how crucial dataset appropriateness is for a classifier. In the proposed approach, the Decision Tree (DT) classifier performs better than the DT in [20] (96% versus 99%), indicating that the combination of design with PCA (TF-IDF with PCA) greatly improves DT's performance. Although it performs rather well (91.8% via SMS), the accuracy of Logistic Regression is not as high as that of the presented approach.

When it comes to feature extraction techniques, TF-IDF often produces better results (97% to 99%) [34] on the UCI dataset than bag-of-words (BoW) (86.45% to 96.9%) [20,35], demonstrating that the scoring process of TF-IDF collects better discriminating features. On the SMS dataset, the combination of TF-IDF with BoW [37] has an accuracy rating of 91.8%, demonstrating that while combined techniques can be advantageous, they may not constantly outperform more straightforward approaches like the one presented.

In comparison to KaiDMML (77.8%) and SMS (91.8%), the UCI dataset regularly produces greater accuracy levels (96% to 99%), indicating that UCI may be cleaner or more appropriate for these methods. The poor 77.8% accuracy for TF-IDF with RF on KaiDMML shows that the dataset may have issues with noise, unbalanced classes, or domain-specific complexity.

SMS Spam datasets frequently produce poor results in approach [37] because they are highly unbalanced (e.g., 90% valid messages vs. 10% spam). The most common class (ham) may be preferred, which could result in subpar spam detection. Because datasets are unbalanced, accuracy might be deceptive.

The proposed technique (TF-IDF with PCA and Decision Tree) outperforms every other related approach, achieving the greatest accuracy of 99% on the UCI dataset. According to this, discriminative ability is probably increased and overfitting is decreased when TF-IDF feature extraction and PCA are combined. When combined with optimized features, Decision Tree exhibits remarkable performance.

6. Conclusions

To develop an efficient SMS spam detection method, this research focused on the significance of data preparation, feature extraction and selection, and machine learning models.

Bag-of-words (BoW), Term Frequency–Inverse Document Frequency (TF-IDF), and N-gram techniques were essential to transform unprocessed text into useful features. Nonetheless, we faced difficulties with unbalanced classes, informal language, and acronyms. By resolving these issues, further advancements could be achieved, with outstanding results in spam message detection. We evaluated two machine learning models and three popular feature extraction approaches and discovered that the TF-IDF with the DT classification algorithm worked better than the other two with regard to the F1-Score, accuracy, precision, and recall.

By using PCA, the average inference time per message and the model size were reduced without sacrificing classification accuracy. These outcomes show that the architecture is appropriate for implementation on devices with limited resources, like embedded systems and cellphones.

Experimental evaluations conducted on the UCI SMS Spam Collection dataset indicated that the proposed TF-IDF-PCA-DT approach had a 99% accuracy. The findings demonstrate that the proposed method works better than other advanced techniques, particularly when it comes to accuracy, which is believed to be the most crucial element influencing the performance of detection methods. In future research, to improve the resilience of the recommended spam detection method, we are going to look into semantic-level adversarial attacks, various text obfuscation methods, and active defence tactics. In addition, it should examine whether the presented method offers comparable advantages on challenging datasets, such as IMDB and KaiDMML. In order to further optimize the balance between classification performance and processing efficiency, it is interesting to investigate light deep learning models like TinyBERT or MobileBERT.

Author Contributions: Conceptualization, I.T.A. and B.T.H.; Writing—original draft preparation, I.T.A. and B.T.H.; Review and editing, B.W.N.V.; Validation, B.T.H. and M.A.H.; Software, I.T.A.; Validation, B.W.N.V.; Funding acquisition, M.A.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Department of Civil Engineering, FEQS, INTI-IU University, Malaysia, the research grant INTI IU-2025: INTI-FEQS-01-03-2025. This support was crucial to the publication of this study.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in the study are openly available in [29,30].

Acknowledgments: The authors are grateful to the Department of Civil Engineering, FEQS, INTI-IU University, Malaysia, for their generous support through the research grant INTI IU-2025: INTI-FEQS-01-03-2025. This support was crucial to the publication of this study.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Abid, M.A.; Ullah, S.; Siddique, M.A.; Mushtaq, M.F.; Aljedaani, W.; Rustam, F. Spam SMS filtering based on text features and supervised machine learning techniques. *Multimed. Tools Appl.* **2022**, *81*, 39853–39871. [\[CrossRef\]](#)
2. Abayomi-Alli, O.; Misra, S.; Abayomi-Alli, A. A deep learning method for automatic SMS spam classification: Performance of learning algorithms on indigenous dataset. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6989. [\[CrossRef\]](#)
3. Durlik, I.; Miller, T.; Kostecka, E.; Zwierzewicz, Z.; Łobodzińska, A. Cybersecurity in autonomous vehicles—Are we ready for the challenge? *Electronics* **2024**, *13*, 2654. [\[CrossRef\]](#)
4. Jasim, A.A.; Ali Alheeti, K.M. The role of intrusion detection systems and machine in protecting industrial control systems ICS environments. In *AIP Conference Proceedings*; AIP Publishing: Melville, NY, USA, 2024; Volume 3207.

5. Ali, N.J.; Hamzah, N.A.; Radhi, A.D.; Niu, Y.; JosephNg, P.S.; Tawfeq, J.F. 5G-backed resilience and quality enhancement in internet of medical things infrastructure for resilient infrastructure. *TELKOM-NIKA (Telecommun. Comput. Electron. Control)* **2024**, *22*, 372–379. [\[CrossRef\]](#)
6. Hama, D.K.; Mubarek, F.S.; Abdullatif, F.A. Enhanced Security Taxonomy for Fog-Enabled VANETs: A Comprehensive Survey on Attacks, Challenges, Applications and Architectures. *Passer J. Basic Appl. Sci.* **2025**, *7*, 37–61.
7. Johari, M.F.; Chiew, K.L.; Hosen, A.R.; Yong, K.S.; Khan, A.S.; Abbasi, I.A.; Grzonka, D. Key insights into recommended SMS spam detection datasets. *Sci. Rep.* **2025**, *15*, 8162. [\[CrossRef\]](#) [\[PubMed\]](#)
8. Baaqeel, H.; Zagrouba, R. Hybrid SMS spam filtering system using machine learning techniques. In *Proceedings of the 2020 21st International Arab Conference on Information Technology (ACIT)*; IEEE: New York, NY, USA, 2020; pp. 1–8.
9. Chrismanto, A.R.; Sari, A.K.; Suyanto, Y. Enhancing spam comment detection on social media with emoji feature and post-comment pairs approach using ensemble methods of machine learning. *IEEE Access* **2023**, *11*, 80246–80265. [\[CrossRef\]](#)
10. Kim, S.N.; Baldwin, T.; Kan, M.-Y. An unsupervised approach to domain-specific term extraction. In *Proceedings of the Australasian Language Technology Association Workshop 2009*, Sydney, Australia, 3–4 December 2009; p. 94.
11. Angeli, A.; Filliat, D.; Doncieux, S.; Meyer, J.-A. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Trans. Robot.* **2008**, *24*, 1027–1037. [\[CrossRef\]](#)
12. Nagwani, N.K. A Bi-Level Text Classification Approach for SMS Spam Filtering and Identifying Priority Messages. *Int. Arab. J. Inf. Technol.* **2017**, *14*, 61–68.
13. Ahmed, I.T.; Der, C.S.; Hammad, B.T. Recent Approaches on No-Reference Image Quality Assessment for Contrast Distortion Images with Multiscale Geometric Analysis Transforms: A Survey. *J. Theor. Appl. Inf. Technol.* **2017**, *95*, 561–569.
14. Safie, W.; Sjarif, N.N.A.; Azmi, N.F.M.; Yuhaniz, S.S.; Mohd, R.C.; Yusof, S.Y. Sms spam classification using vector space model and artificial neural network. *Int. J. Adv. Soft Compu. Appl.* **2018**, *10*, 130–140.
15. de Luna, R.G.; Magnaye, V.C.; Reaño, R.A.L.; Enriquez, K.L.; Astorga, D.; Celestial, T.; Española, A.M.; Lanting, B.A.; Mugar, D.; Ramos, M. A Machine Learning Approach for Efficient Spam Detection in Short Messaging System (SMS). In *Proceedings of the TENCON 2023-2023 IEEE Region 10 Conference (TENCON)*; IEEE: New York, NY, USA, 2023; pp. 53–58.
16. Oyeyemi, D.A.; Ojo, A.K. SMS Spam Detection and Classification to Combat Abuse in Telephone Networks Using Natural Language Processing. *arXiv* **2024**, arXiv:2406.06578.
17. Balli, S.; Karasoy, O. Development of content-based SMS classification application by using Word2Vec-based feature extraction. *IET Softw.* **2019**, *13*, 295–304. [\[CrossRef\]](#)
18. Saeed, V.A. A Method for SMS Spam Message Detection Using Machine Learning. *Artif. Intell. Robot. Dev. J.* **2023**, *3*, 214–228. [\[CrossRef\]](#)
19. Das Gupta, S.; Saha, S.; Das, S.K. SMS spam detection using machine learning. *J. Phys. Conf. Ser.* **2021**, *1797*, 12017. [\[CrossRef\]](#)
20. Dharrao, D.; Gaikwad, P.; Gawai, S.V.; Bongale, A.M.; Patel, K.; Singh, A. Classifying SMS as Spam or Ham: Leveraging NLP and Machine Learning Techniques. *Int. J. Saf. Secur. Eng.* **2024**, *14*, 289–296. [\[CrossRef\]](#)
21. Ahmadi, M.; Khajavi, M.; Varmaghani, A.; Ala, A.; Danesh, K.; Javaheri, D. Leveraging Large Language Models for Cybersecurity: Enhancing SMS Spam Detection with Robust and Context-Aware Text Classification. *arXiv* **2025**, arXiv:2502.11014.
22. Taylor, A.; Robert, A. Using Machine Learning to Detect Fraudulent SMSs in Chichewa. *arXiv* **2025**, arXiv:2502.16947.
23. Ahmed, I.T.; Der, C.S.; Jamil, N.; Hammad, B.T. Analysis of Probability Density Functions in Existing No-Reference Image Quality Assessment Algorithm for Contrast-Distorted Images. In *Proceedings of the 2019 IEEE 10th Control and System Graduate Research Colloquium (ICSGRC)*; IEEE: New York, NY, USA, 2019; pp. 133–137.
24. Robertson, S. Understanding inverse document frequency: On theoretical arguments for IDF. *J. Doc.* **2004**, *60*, 503–520. [\[CrossRef\]](#)
25. Omuya, E.O.; Okeyo, G.O.; Kimwele, M.W. Feature selection for classification using principal component analysis and information gain. *Expert Syst. Appl.* **2021**, *174*, 114765. [\[CrossRef\]](#)
26. Wang, S.; Yu, X.; Jia, W. A new population initialization of particle swarm optimization method based on pca for feature selection. *J. Big Data* **2021**, *3*, 1–9. [\[CrossRef\]](#)
27. Jayapandian, N. Machine Learning Based Spam E-Mail Detection Using Logistic Regression Algorithm. In *Proceedings of the 2023 IEEE International Conference on ICT in Business Industry & Government (ICTBIG)*; IEEE: New York, NY, USA, 2023; pp. 1–6.
28. Priyanka; Kumar, D. Decision tree classifier: A detailed survey. *Int. J. Inf. Decis. Sci.* **2020**, *12*, 246–269. [\[CrossRef\]](#)
29. Kaggle. Sms Spam Collection Dataset. 2016. Available online: <https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset> (accessed on 1 January 2026).
30. SMS Spam Collection. Available online: <https://archive.ics.uci.edu/dataset/228/sms+spam+collection> (accessed on 1 January 2026).
31. Ahmed, I.T.; Hammad, B.T.; Jamil, N. A comparative analysis of image copy-move forgery detection algorithms based on hand and machine-crafted features. *Indones. J. Electr. Eng. Comput. Sci.* **2021**, *22*, 1177–1190. [\[CrossRef\]](#)
32. Heckler, C.E. *Applied Multivariate Statistical Analysis*; Prentice Hall: Saddle River, NJ, USA, 2005; p. 517.

33. Ahmed, I.T.; Der, C.S.; Hammad, B.T.; Jamil, N. Contrast-distorted image quality assessment based on curvelet domain features. *Int. J. Electr. Comput. Eng.* **2021**, *11*, 2595–2603. [[CrossRef](#)]
34. Sjarif, N.N.A.; Azmi, N.F.M.; Chuprat, S.; Sarkan, H.M.; Yahya, Y.; Sam, S.M. SMS spam message detection using term frequency-inverse document frequency and random forest algorithm. *Procedia Comput. Sci.* **2019**, *161*, 509–515. [[CrossRef](#)]
35. Xia, T.; Chen, X. A weighted feature enhanced Hidden Markov Model for spam SMS filtering. *Neurocomputing* **2021**, *444*, 48–58. [[CrossRef](#)]
36. Szabó Nagy, K.; Kapusta, J. Twidw—A novel method for feature extraction from unstructured texts. *Appl. Sci.* **2023**, *13*, 6438. [[CrossRef](#)]
37. Subhalakshmi, R.T. SMS Spam Detection using Machine Learning. *J. Sci. Technol. Res. (JSTAR)* **2025**, *6*, 1. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.