

Article

ML-Based SMS Messaging Spam Detection: Impacts of Text Feature Extraction Techniques

Ahmad Ababneh ¹, Maram Bani Younes ^{2,*}¹ Computer Science, Information Technology Faculty, American University of Madaba, Amman 11821, Jordan² Cybersecurity, Information Technology Faculty, American University of Madaba, Amman 11821, Jordan

* Correspondence: m.baniyounes@aum.edu.jo

Abstract

Spam detection on SMS messaging has not received as much attention from researchers recently as the spam detection studies on emails or social media platforms. However, spam SMS messaging can be more intrusive, annoying, and harmful. Thus, detecting and filtering spam SMS messages is becoming a priority that saves human productivity. This work aims to introduce a dynamic, accurate, and efficient machine learning-based spam detection technique for SMS messaging. It aims at protecting users and businesses from spam SMS attacks. It aims to detect and identify suspicious messages that contain promotional, misleading, irrelevant, or harmful content. It primarily aims to test and evaluate the impact of feature extraction methods on the performance of machine-learning-based spam detection. Several text feature extraction techniques have been used and tested, including classical, statistical, contextual, and advanced embedding techniques. An extensive set of experiments has been presented on benchmark datasets in this field. From the comparative study, we can infer that all investigated feature extraction techniques have achieved high accuracy (90%+) on the in-domain dataset. However, their performance decreased when they were tested on the out-of-domain dataset (70%+). The advanced embedding techniques achieved the best performance across both datasets compared to the other tested feature extraction models.

Keywords: SPAM; SMS messaging; feature extraction; TF-IDF; GloVe; BERT; LSTM

1. Introduction

Spam detection has been identified as a challenging topic that motivates researchers across several communication platforms. Spam texts have appeared as an annoying and threatening context that disrupts users and carries malware and phishing links to their personal devices. Spam emails have received the closest attention in this field, and all email providers have introduced a junk or spam folder to separate spam emails from legitimate ones [1]. Thus, it enhances the user's productivity and the safety conditions of their devices.

On open social media platforms, spam content has been very popular as well. Several posts and comments on Twitter, YouTube, Facebook, or Instagram have been set as commercial advertisements, irrelevant content, or even harmful URLs that lead to malware content and phishing attempts [2–4]. Spam detection research over social media platforms has examined the context of the shared text according to a set of standard criteria (i.e., number of characters, URL existence, punctuation, emojis, keywords, etc.) [2,3]. Other studies have focused on the reputation of the user; they aim to detect suspicious accounts based on the number of friends, frequency of posts, and other indicating parameters [5–7].



Academic Editor: Martin Gilje Jaatun

Received: 13 June 2026

Revised: 10 July 2026

Accepted: 16 July 2026

Published: 18 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.

On the other hand, research studies that target spam SMS messaging are less common. Spam SMS messaging is any unsolicited text message sent to a mobile phone with the intention to advertise, scam, or collect personal information. Recently, it has grown aggressively because sending SMS at scale is cheap, and people often trust messages on their phones more than email [8]. Spam SMS messaging can be highly intrusive, annoying, and harmful. They reach directly to personal mobile phones that users hold and use all the time [9]. In addition, several users forward the SMS messages to their smartwatches and wearable devices. SMS messages have different characteristics compared to email and comments on social media platforms. This is summarized by being short, informal, and mobile-oriented messages. They contain abbreviations, slang, emojis, symbols, links, and numeric patterns such as OTPs or promotional codes. In general, SMS messages lack rich formatting, detailed context, threaded discussions, and user profile information. This introduces a more challenging issue to investigate. This motivates us to tackle this platform of messages to detect spam ones.

At the same time, artificial intelligence (AI) introduces various techniques and mechanisms to replace tedious human activities [2]. Spam detection is one of the main topics that can efficiently utilize AI and machine learning techniques. Training machine learning models to distinguish between spam and legitimate texts sounds promising for different communication platforms [3]. This requires selecting the proper model, tuning this model, preparing suitable datasets, and extracting the most effective features to obtain the best performance.

In this work, we aim to investigate the previous studies that have mainly considered spam SMS messaging. Study the ability of machine learning models to classify English spam SMS messages after investigating the benchmark datasets in this field. Investigating the role of the standard text feature extraction techniques, such as TF-IDF, GloVe, BERT, and LSTM, in SMS spam detection mechanisms and comparing the performance on standard unseen datasets. Figure 1 illustrates the main methodology of this study; the right side of the figure shows how to prepare the dataset and extract the main features to train the models. The left side illustrates how to test the trained models with the unseen merged dataset. The performance of these models has been evaluated through four main metrics: accuracy, precision, recall, and F1-score. From the experimental study, we can see that the transformer-based representation (BERTs) achieved the highest accuracy, recall, and F1-score across both datasets. This indicates good performance on unseen spam patterns. The performance of the other classical, static, and even sequence models is high for the in-domain dataset. However, a drastic decrease appears in the out-of-domain dataset in all measured metrics. Based on these results, the paper recommends the advanced embedding techniques to extract features of text before developing spam detection techniques. Most classification models can be used to develop spam detection techniques for SMS messages with an accuracy higher than 70% for unseen datasets.

The rest of this paper is organized as follows: Section 2 investigates the previous studies that have considered SMS messaging spam detection. It summarizes the main features, limitations, and considered parameters of each study. Section 3 identifies the main benchmark datasets that have been gathered and prepared for SMS messaging spam detection and how to prepare them for the study. Section 4 discusses the main standard text feature extraction techniques, illustrating the considered features and how to extract them. Section 5 introduces the experimental study that investigates and discusses the role of the selected feature extraction techniques in ML-based spam detection mechanisms. Section 6 presents a detailed discussion of the presented results and recommendations. Finally, Section 7 concludes the entire paper with some recommendations for future study directions.

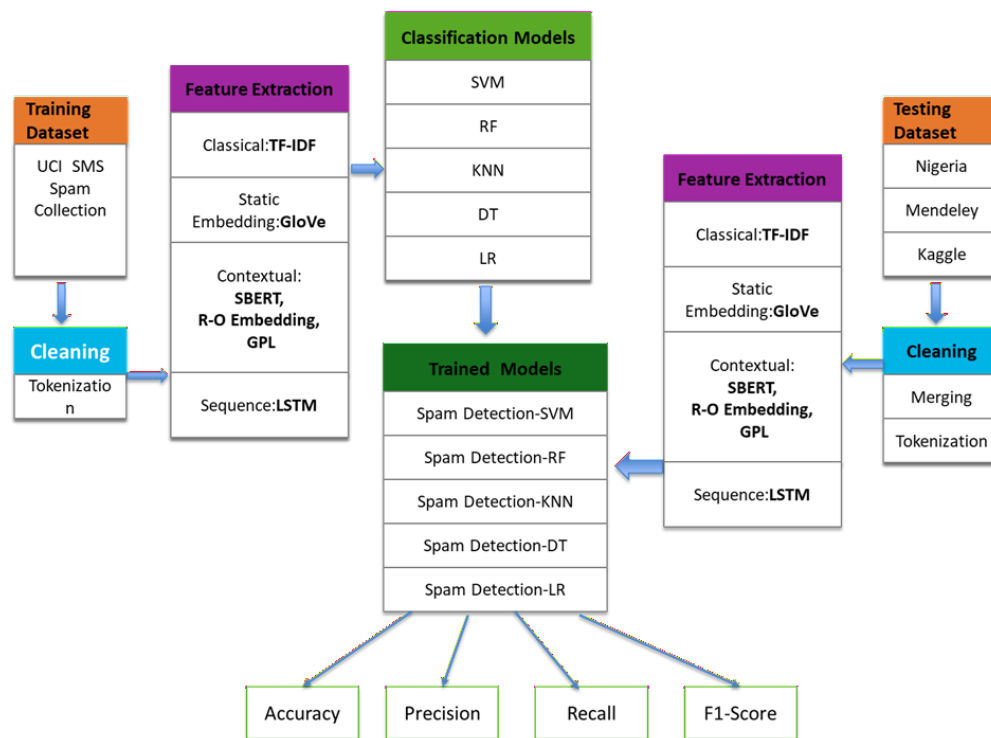


Figure 1. The methodology.

2. Related Work

Spam detection mechanisms have been widely investigated in the literature. Spammers, scammers, and criminals use spam messages over communication platforms to deceive users and achieve their targets. Declaring spam messages can be a challenging process that requires careful consideration of the target, mechanism, and consequences of the message. The message that is identified as spam for an individual can seem helpful and interesting to another, especially messages with advertising content. In addition, spammers evolve their tactics and continuously enhance their skills to spread spam messages [5,10]. Most spam detection studies have targeted email [1] and social media platforms [4]. However, few research studies have considered the spam detection for Short Message Service (SMS) [2,3].

Recently, mobile phones have experienced a high percentage of spam through the SMS messaging platform. The limited number of allowed characters, restrictions on attachment files, costs and bills of SMS messages, and enforced encoding strategy have reduced the speed of progress in spam and scam techniques on this platform. However, recently, due to the development of this service, which now allows including URLs in SMS messages. Moreover, the decreasing cost of SMS packages, spam SMS messages are more popular and highly spread [2,3]. Public websites have been developed for mobile customers to report received spam SMSs, such as: Grumbletext website (UK forum) [11] and National University of Singapore (NUS) SMS Copus [12]. Other common spam SMS messages have been shared as screenshots on Twitter [13], this in addition to the ability to report spam SMS messages directly to the mobile carrier.

Some research studies have been introduced in the literature, considering SMS spam detection. These studies have considered advances in machine learning (ML), deep learning (DL), and natural language processing (NLP). They vary widely in dataset selection, pre-processing techniques, feature extraction methods, and classifier architectures. Several comparative surveys [14,15] provided general overviews of spam detection pipelines, investigating the importance of pre-processing and feature extraction phases. Several

studies highlight the effective roles of each phase in shaping classifier performance. Table 1 illustrates a simple comparative study among previous studies that identifies the selected options in the spam detection pipeline for each work.

As we can see from Table 1, some studies have focused on investigating the importance of pre-processing and feature extraction mechanisms for general datasets. For instance, Maqsood et al. [16] evaluated the effects of using Bag of words (BoW) and Term Frequency–Inverse Document Frequency (TF-IDF) feature extraction techniques on Russian SMS and email datasets while using traditional machine learning (ML) techniques as classifiers. This study [16] reports that the support vector mechanism (SVM) consistently outperforms naive Bayes (NB), random forest (RF), and convolutional neural network (CNN) classifiers in terms of detecting spam SMS messages. Moreover, Jian et al. [17] and Ahmadi et al. [18] have also confirmed the effectiveness of the extracted features by BoW and TF-IDF from SMS text messages. Classical machine learning models such as NB, SVM, and deep neural network (DNN) have reported strong accuracy across multiple datasets.

More recent studies have utilized more advanced contextual and neural feature extraction [19]. In these studies, transformer-based architectures have shown substantial improvements over classical machine learning (ML) pipelines [8,20,21]. Liu et al. [20] proposed a modified transformer model that surpassed traditional machine learning baselines across two datasets (i.e., SMS Spam Collection and Twitter Spam Dataset). The proposed model leverages positional encoding and self-attention mechanisms for enriched semantic representation. On the other hand, Oswald et al. [8] and Airlangga [21] explored BERT, RoBERTa, and DistilBERT embeddings, demonstrating that contextualized feature extraction significantly boosts classification performance. First, DistilBERT has been joined with the SVM [8] to detect spam messages. Then, RoBERTa [21] has also been used with traditional ML models. Both studies have achieved better accuracy results, which confirms the strength of hybrid and deep contextual models.

Furthermore, feature extraction techniques play another pivotal research focus. Srinivasarao and Sharaff [22] have investigated the most effective features and optimized them. They combined the word-to-vector (Word2Vec) embedding technique with six evolutionary optimization algorithms (i.e., PRFS, PCC, LASSO, GSS, MRDC, CBFS) for feature selection. The selected features have then been fed into a hybrid KNN-SVM classifier. This model has outperformed conventional models across a broad set of metrics (i.e., accuracy, precision, recall, and F1-score). Metadata regarding the collected text messages have also been utilized to enrich the investigation of spam messages. Oyeyemi et al. [9], integrate text-based and metadata-based features using a BERT embedding system. Then, the NB classification model has been utilized to detect spam messages. This hybrid NB and BERT model achieved 97.3% accuracy with minimal computation time. Other studies [23–25] have discussed the added value of optimized feature extraction and ensemble methods—particularly TF-IDF paired with sophisticated voting systems or majority classifiers, where the Relevance Vector Machine (RVM) [24] and ensemble approaches [25] achieved the best performance.

In parallel, several research studies have investigated data balancing and augmentation to enhance the spam detection mechanisms [19,21]. This should reduce the percentage of overfitting problems. In order to remove any imbalance in the investigated data, Uddin et al. [21] employed back-translation and resampling mechanisms before training transformer-based classifiers. The synthetic minority over-sampling technique (SMOTE) has also been used [25] to improve a balanced dataset before training the models. On the other hand, Al-Kabbi et al. [19] proposed a more sophisticated fusion model that combines different feature extraction streams: CNN, LSTM, and TF-IDF. The latter model has used the multi-

branch principle to capture the semantic and statistical attributes of the investigated text. This model has positive effects on the accuracy of identifying and detecting spam messages.

Furthermore, several other studies have addressed the manipulation challenges in real-world spam detection. This includes the evasive spelling and data poisoning tactics [26,27]. For instance, Salman et al. [26] have evaluated the robustness feature across several datasets sourced from SMS, Twitter, and scam observatories. Their work emphasizes the need and importance of considering the adversarial tactics: evasive spelling and readability manipulation. In addition, Li et al. [28] highlighted the role of modern classifiers and transformers (i.e., BERT and GPT-based) to in identifying adversarial attacks and detecting data poisoning. Yerima and Bashar [29] and Ozoh et al. [27] have also discussed the need for continual refinement learning using BoW and N-gram systems to extract the features of text.

Table 1. Summary of existing studies on SMS spam detection.

Study	Dataset	Preprocessing	Feature Extraction	Classification Models	Metrics	Results
[16]	Spam-based SMS and emails from Russia	tokenization, lowercasing, normalization, stemming	BoW, TF-IDF	NB, RF, SVM, CNN	Accuracy, Precision, Recall, F1, Confusion Matrix	SVM outperforms all ML classifiers
[20]	SMS Spam Collection v.1; Twitter Spam Dataset	embeddings + positional encoding	Transformer encoder–decoder self-attention	Modified Transformer	Accuracy, Precision, Recall, F1	Transformer outperforms traditional ML
[8]	SMS Spam Collection	contextual embeddings (BERT, SpanBERT, RoBERTa, DistilBERT)	BERT-based embeddings	RF, SVM, DT	Accuracy, Precision, Recall, F-measure, AUC	DistilBERT+SVM achieves best contextual extraction
[22]	SMS Spam Collection	stemming, stop-word removal, tokenization	Word2vec + six EO (PRFS, PCC, LASSO, GSS, MRDC, CBFS)	hybrid KNN-SVM	Accuracy, Precision, Recall, F1, Kappa, MAE, RMSE	Outperforms current classifiers
[9]	Kaggle, DSN, Self-Data	tokenization, special tokens, masking, padding	text-based + metadata features	LR, NB, RF, GB, SVM + BERT	Accuracy, Precision, Recall, F-measure, Time	NB+BERT hybrid achieves 97.3% accuracy
[23]	Kaggle Repository	cleaning + tokenization	TF-IDF, Word Vector	TF-IDF directory classifier	Accuracy	Accuracy > 95%
[17]	SMS Spam Collection	–	BoW with TF and TF-IDF, chi-square selection	Multinomial NB	Accuracy	Highest accuracy vs classical models
[24]	SMS Spam Collection	cleaning + tokenization	TF-IDF	Majority voting (NB, SVM, RVM, KNN)	Accuracy, Recall, Precision, F-score, ROC	RVM best performance
[26]	“Super Dataset” (SMS + Twitter + volunteers + observatories)	readability, spelling, evasion detection	BoW / TF-IDF	Shallow ML + DL models	Accuracy, Precision, Recall, F-score, Confusion Matrix, ROC	Robust evaluation across diverse datasets
[18]	Kaggle	removing stop words	BoW, TF-IDF	NB, SVM, KNN, DNN, DT, LDA	Accuracy, Precision, Recall, F-score, Confusion Matrix, ROC	TF-IDF + NB/SVM/DNN most effective
[21]	SMS Spam Collection	back-translation augmentation, resampling	DistilBERT, RoBERTa embeddings	XGB, SVM, KNN, RF, DistilBERT, RoBERTa	–	RoBERTa performs best
[25]	SMS Spam Collection	SMOTE	TF-IDF	Voting (RF, XGBoost, SVM, LR)	Accuracy, Precision, Recall, F-score	Voting achieves best performance
[19]	SMS Spam Collection v1	punctuation removal, lowercasing, stop-word removal, Tokenization	Fusion: CNN + LSTM + TF-IDF	Dense layer w / attention	Accuracy, Precision, Recall, F-score	Fusion model outperforms other classifiers
[29]	SMS Spam Collection	Tokenization, removing stop words	BoW	ET, KNN, DT, SVM, RF, XGBoost	–	100% TPR, 98% accuracy
[27]	Kaggle	cleaning, Tokenization	N-grams	NB, SVM, K-means, TF-IDF, BERT	Accuracy, Precision, Recall, F-score	Highlights need for continuous model improvement
[28]	SpamHunter	cleaning, tokenization	Binary SSDs, Multi-label SSDs, FL-trained SSD	CNN, BERT, GPT-4	Accuracy, Precision, Recall, F-score	Models vulnerable to adversarial attacks

In general, by investigating the details and selected options of the previous studies, we can find several gaps. First, while classical feature extraction approaches such as BoW and TF-IDF remain strong baselines, contextual embeddings and hybrid architectures provide substantial performance gains. Second, robustness, interoperability, and resilience to adversarial manipulation are emerging as critical considerations. These findings motivate the need to systematically examine how different text feature extraction techniques influence ML-based spam detection performance. Thus, in this work, we aim to investigate

the performance of different text feature extraction classes. This includes classical, static embedding, contextual embedding, and sequence models. One or more representative models have been selected from each class, and extensive sets of experiments have been applied to compare their performance on benchmark datasets.

3. Processing SMS Messaging Datasets

Short Message Service (SMS) is short messages to be transferred among mobile phones. They represent means of personal communication that are important communicative artifacts in the current digital era. Moreover, SMS is used to advertise certain products or broadcast notifications or instructions to employees in each company. Several benchmark datasets have been utilized and prepared in the literature [30,31]. In this section, we first explore the utilized SMS spam datasets. Then, we explain how to prepare them for the experiments.

3.1. SMS Messaging Datasets

Several datasets are gathered and prepared by capturing real transferred SMS messages among mobile phones to train and prepare intelligent spam detection mechanisms. Here, we investigate some benchmark datasets that have been prepared and used to detect spam SMS messages in the literature. Table 2 summarizes the main characteristics of the utilized SMS messaging spam detection datasets. In this work, the first dataset (i.e., UCI) has been used to train the standard machine learning models. The other three investigated datasets have been merged and used as an unseen dataset to test the trained models. Training the models on a dataset of 5000 records (UCI) and testing them on 15,000 records (merged dataset) aims to test the general ability of these models to detect spam SMS messages in real-world scenarios on unseen messages.

3.1.1. UCI Machine Learning Repository

The UCI Machine Learning Repository is a benchmark dataset for SMS spam detection [32]. It is the most famous dataset that is used for research and teaching purposes. It includes the SMS Spam Collection Dataset, which is a set of SMS tagged messages that have been collected for SMS spam detection research. It contains 5574 SMS messages in English are labeled as ham (legitimate) or spam. This dataset is available in a simple CSV file that contains 5,574 records. Each record contains two main fields. The first one contains the ham or spam label, and the second field contains the message plain text. These records have been collected from three main free sources available on the Internet, and they have been developed and adapted for research purposes. First, a collection of records was manually extracted from the Grumbletext [11] website. It is the UK forum where cell phone users publicly report the received SMS spam messages. Second, some records were extracted directly from NUS SMS Corpus (NSC) [12]. This is a dataset of about 10,000 legitimate SMS messages collected from volunteer students attending the National University of Singapore. These students were informed that their contributions would be made publicly available for research. Third, several records had been utilized from the SMS Spam Corpus v.0.1 Big [33]. This is a benchmark SMS spam dataset that has 1002 SMS ham messages and 322 spam messages.

3.1.2. Nigeria

This is an SMS spam collected dataset (*ExAIS-SMS*) considering the African-English context [34]. It was conducted at the Federal University of Agriculture, Abeokuta, Nigeria [35]. Records of the dataset were collected from 20 different mobile devices of users within the university community, involving students and staff within the university community. The final CSV file contains three main columns. The third column contains the text

message, the second column shows the label for each text as spam or ham, and the first column determines if this message was a sent or received message. In this dataset, 2176 messages are labeled as spam, and the remaining 2805 are labeled as Ham (legitimate).

3.1.3. Mendeley

This dataset is a subset of the original Mendeley dataset [36]. It is extracted as related only to English text and scam attempts. The subset used in this work is represented by a CSV file that contains 412 different records. Two columns appear in this dataset: the first one determines if the text is legitimate or if it is a scam (a dedicated type of spam to deceive the receivers), so it is the label column. The second column contains the text messages that are gathered from the mobile phones. The scam messages represent 59% (i.e., 243 messages) of the dataset records, while the legitimate messages represent 41% (i.e., 169 messages). The second column contains the SMS text message that has been gathered from mobile phones.

3.1.4. Kaggle Dataset

This dataset contains a curated collection of SMS messages labeled as spam or ham (legitimate) [37]. The dataset consists of over 10,000 SMS messages, each labeled as either ‘spam’ or ‘ham’. These messages have been collected from various undeclared sources, so we consider them a random dataset. Each message is represented as a text string. It is represented in a CSV file that contains two main columns: text and label. The first column represents the content of each SMS message. The second column indicates whether each SMS message is a spam or a ham message. In this dataset, the spam messages represent 22% of the entire records. There are around 2200 messages in total. This dataset can represent a good unseen dataset to test the trained models since it contains a relatively large number of records. In addition, its imbalanced shape simulates real-life scenarios.

Table 2. Comparison of the main characteristics of SMS spam datasets.

Dataset	Size / Records	Label Distribution	Key Characteristics
UCI SMS Spam Collection [32]	5574 SMS messages	4827 ham and 747 spam	Most widely used benchmark dataset; collected from Grumbletext + NUS Corpus; English only; publicly available.
Nigara ExAISS Dataset [34]	4981 SMS messages	2805 ham and 2176 spam	Collected from 20 volunteer users in Nigeria; includes metadata (contacts, timestamps, sender info); African-English linguistic style.
Mendeley SMS Scam Subset [36]	412 SMS messages	243 ham and 169 spam	Subset of the original Mendeley dataset; English-only scam detection; highly imbalanced; real-world mobile phone collection.
Kaggle SMS Spam Dataset [37]	10,000 SMS messages	7800 ham and 2200 spam	Collected from mixed/unknown sources; larger dataset; good for evaluating generalization on unseen, imbalanced data.
Merged Dataset	15,393 SMS messages	10,848 ham and 4545 spam	It merges Nigara, Mendeley, and the Kaggle datasets.

3.1.5. Merged Dataset

The benchmark datasets in this field of research are stable and representative. They have been gathered from different servers and reported environments. Thus, we merged “Nigeria”, “Mendeley”, and “Kaggle Dataset” into a new “Merged Dataset” to form an unseen dataset after training the classification model using the “UCI” dataset. In the new merged dataset, the text field has been gathered and prepared as a set of sequential words. The label has been unified into numerical values, where “0” indicates a ham message and “1” indicates a spam message. In the merged dataset, 15,393 SMS messages appear, where 10,848 are ham and 4545 are spam. This dataset has been prepared as a big unseen

(out-of-domain) dataset to measure the performance of the trained classification models. The behavioral pattern of static text features [38] is evaluated in these experiments.

3.2. Text Preprocessing and Normalization

Datasets were preprocessed using a unified cleaning and normalization pipeline appear in Figure 2. Datasets were collected from different sources. The raw messages contained various inconsistencies, such as hyperlinks, email addresses, punctuation symbols, numerical values, and irregular formatting.

First, all text messages were converted to lowercase in order to ensure consistency and avoid treating words with different capitalizations as separate terms. Next, URLs, web links, and email addresses were removed using regular-expression-based filtering. These elements generally do not contribute meaningful semantic information for spam classification purposes. Numerical values, punctuation marks, special characters, and extra whitespace characters were also removed. This reduces unnecessary noise in the dataset and standardizes the formatting of the text.

After that, the cleaned messages were tokenized into individual words using whitespace-based tokenization. Common English stop words such as “the”, “is”, and “and” were removed using the stop-word corpus. These words occur frequently in text but typically carry limited discriminative value. Subsequently, lemmatization was applied to transform words into their base forms. For example, words such as “playing”, “plays”, and “played” are normalized to the root form “play”. This step helps reduce vocabulary size while preserving the semantic meaning of the text. The investigated datasets represent SMS text messages, which are mainly characterized by their short length. The number of tokens extracted from each message is around 15 tokens. The minimum number of tokens of a single message was 10 tokens. However, the maximum number of tokens was 27. The average length of ham messages (i.e., 13) was shorter than the average length of spam messages (i.e., 23).



Figure 2. Text pre-processing.

Finally, the processed messages are stored as clean text for subsequent feature extraction and machine learning experiments. This preprocessing pipeline improved textual consistency, reduced noise, and produced a more compact and meaningful representation of the datasets [39].

4. Text Feature Extraction

Text feature extraction is a foundational step in any SMS spam detection pipeline. It determines how simple text is transformed into numerical representations with strong indications that are suitable for machine learning (ML) and deep learning (DL) models. The quality of these representations directly affects the classifier’s performance, robustness, generalization, and resilience against adversarial manipulation. Numerous approaches have been proposed in the literature, ranging from classical statistical representations to modern transformer-based contextual embeddings. Table 3 summarizes the primary categories of feature extraction of text and their

typical output representations. In this work, we have selected one technique from each category and then compared the performance of the ML-based spam detection mechanisms while using these different feature extraction techniques.

Table 3. The primary categories of feature extraction of text.

Category	Examples	Output
Classical	BoW, TF-IDF , n-grams	Sparse vector
Static Embeddings	Word2Vec, GloVe	Dense word vectors
Contextual	BERT, RoBERTa, SBERT , R-O , GPL	Dense sentence vectors
Sequence Models	LSTM , GRU, CNN	Learned sentence vectors

4.1. Classical

Classical feature extraction techniques such as bag-of-words (BoW), Term Frequency–Inverse Document Frequency (TF-IDF), and n-gram models have long been used as baseline representations in text classification tasks. These methods transform text into high-dimensional, sparse vectors, where features correspond to word occurrences or co-occurrence patterns [40]. SMS spam often relies on predictable keyword-based triggers (e.g., “free”, “win”, “urgent”), so classical methods remain highly competitive, since they can detect these words efficiently and their frequency in the text.

However, classical approaches fail to capture semantic meaning, word order, or contextual differences. They are vulnerable to simple adversarial manipulations such as intentional misspellings (“fr33”, “w1n”), word obfuscation, or paraphrasing—limitations increasingly exploited in modern spam campaigns [41]. In this work, we consider the TF-IDF mechanism from this category for the comparison study among the different text feature extraction categories.

Term Frequency–Inverse Document Frequency (TF-IDF)

This is a statistical measure that evaluates the importance of each word in a document relative to a dataset. It consists of two primary components: Term Frequency (TF) and Inverse Document Frequency (IDF) [42]. They are defined in the following:

- Term Frequency quantifies how often a term appears in a specific record. Equation (1) illustrates how to compute the TF value for each word, where $f_{w,r}$ denotes the number of times word w appears in the record r . N_r represents the total number of words in the considered record (r). Higher TF values indicate that the word is more significant within that text record.

$$TF(w, r) = \frac{f_{w,r}}{N_r} \quad (1)$$

- Inverse Document Frequency measures the uniqueness of each word within the entire dataset. It aims to mitigate the effect of common words across different records. It allows for a focus on words that are more exclusive to individual records. The least frequent words are the most important words to classify the text. Equation (2) is used to find these values for each word in the dataset. N is the total number of records in the dataset, and f_w is the number of records that contain the word w .

$$IDF(w) = \log\left(\frac{N}{f_w}\right) \quad (2)$$

- TF-IDF Score combines both previous factors to provide a weighted word representation as appears in Equation (3). The resulting score signifies a word’s significance in

each record against the broader dataset. This facilitates the effective feature representation for subsequent classification tasks.

$$\text{TF-IDF}(w, r) = \text{TF}(w, r) \times \text{IDF}(w) \quad (3)$$

The maximum number of extracted features is one of the main parameters that have to be adjusted in the TF-IDF technique. It is set based on the language and the number of possible words. This parameter affects the accuracy, generalization, and computational cost of the classification model. In this work, we investigate four different numbers of extracted features (i.e., 500, 1000, 2500, 5000) and their effects on the performance of the ML-based spam detection mechanisms.

4.2. Static Embeddings

Static word embeddings represent a shift from sparse lexical features to dense semantic vectors. Word to vector (Word2Vec) [43] and Global Vectors (GloVe) [44] are the most popular static embedding techniques. These embeddings map each word into a continuous vector space where semantic similarity is encoded through geometric proximity. By capturing distributional semantics, static embeddings overcome several limitations of classical feature extraction techniques, especially in handling synonyms and informal texting patterns.

However, a key limitation is that static embeddings assign one fixed vector per word regardless of context. This results in poor handling of polysemous terms, an important issue where meaning can differ drastically depending on intent. Furthermore, aggregating word-level embeddings into sentence-level features (e.g., through averaging or pooling) may cause loss of contextual information, particularly in the extremely short structure of text. In this work, we focus on the Global Vectors (GloVe) technique for comparison purposes.

Global Vectors for Word Representation (GloVe)

This is an unsupervised learning algorithm developed to create word embeddings by aggregating global word–word co-occurrence statistics from a corpus. It transforms each word into a continuous vector space where words with similar meanings are closer together. GloVe is based on the co-occurrence matrix. If two words occur together frequently in a specific context, their meanings are likely similar. The co-occurrence matrix (X) for a corpus is defined in Equation (4), where X_{ij} represents how often word w_j occurs in the context of word w_i .

$$X_{ij} = \text{frequency}(w_i, w_j) \quad (4)$$

The GloVe model aims to find word vectors $\mathbf{w}_i$ and $\mathbf{w}_j$ such that their dot product can approximate the logarithm of the probability of co-occurrence. The objective function is defined in Equation (5). Where V is the vocabulary size. $f(X_{ij})$ is a weighting function that reduces the influence of rare events. b_i and b_j are bias terms for words w_i and w_j .

$$J = \sum_{i=1}^V \sum_{j=1}^V f(X_{ij}) \left(\mathbf{w}_i^T \mathbf{w}_j + b_i + b_j - \log(X_{ij}) \right)^2 \quad (5)$$

The weighting function $f(X_{ij})$ is introduced to ensure that the model focuses more on significant co-occurrences while down-weighting the impact of rare occurrences, typically defined in the following Equation, where $x_{\max}$ is a cutoff threshold.

$$f(x) = \begin{cases} (x/x_{\max})^\alpha & \text{for } x < x_{\max} \\ 1 & \text{for } x \geq x_{\max} \end{cases}$$

GloVe embeddings are available in multiple pretrained versions differing in corpus size, vocabulary coverage, and embedding dimensionality, allowing practitioners to select representations optimized for general text, web-scale data, or social media language. In this study, we employ the GloVe Wikipedia 2014 + Gigaword 5 pretrained word embeddings [45]. It constitutes one of the most widely adopted and well-established GloVe variants in natural language processing research. These embeddings were trained on a combined corpus consisting of the Wikipedia 2014 dump and the English Gigaword Fifth Edition, totaling approximately 6 billion tokens. This model provides vector representations for more than 400,000 words. These vectors capture the syntactic and semantic relationships through global word statistics.

The training dataset in this version is high-quality, well-selected, and rich diverse. It uses encyclopedic text and carefully edited news content to reduce any noise ensures wide lexical and semantic coverage. In this work, we consider four embedding dimensions (i.e., 50, 100, 200, and 300) that offer a balance between representational capacity and computational efficiency.

4.3. Contextual

Contextual embeddings are introduced by several encoder transformer architectures such as BERT, RoBERTa, and Sentence-BERT. These models provide vector representations of the text context that is dynamically adjusted based on the surrounding words (i.e., full sentences) [46–48]. These vectors capture deep semantic and syntactic relationships, which is better than the capability of classical or static approaches.

Contextual models excel in capturing intent, subtle cues, and linguistic irregularities often present in sophisticated phishing or scam messages. Sentence-level models such as SBERT are particularly beneficial for short text contexts because they compress the entire text into a compact vector, enabling efficient downstream classification even with lightweight ML models. Despite their strengths, contextual models require greater computational resources, may demand careful finetuning, and are sensitive to domain shift if trained solely on general-purpose corpora. Accordingly, this study investigates three major categories of contextual embedding models, namely, Sentence-Transformer models, retrieval-oriented embedding models, and general-purpose transformer language models, represented by six state-of-the-art architectures: `all-mpnet-base-v2` (BERT-1), `all-MiniLM-L12-v2` (BERT-2), `all-MiniLM-L6-v2` (BERT-3), `bge-large-en-v1.5` (BERT-4), `e5-base` (BERT-5), and `DeBERTa-v3-base` (BERT-6).

4.3.1. Sentence-BERT (SBERT)

This is a modification of BERT designed specifically to generate semantic sentence embeddings that can be compared efficiently [48]. Standard BERT models are great options for downstream NLP tasks. However, they cannot produce fixed-size embeddings for scalable similarity computation, retrieval, clustering, or semantic search. Thus, running BERT as a cross-encoder for each sentence pair is not possible for large datasets.

SBERT addresses this limitation by using a Siamese/bi-encoder structure. It generates dense, fixed-dimensional embeddings that also preserve the semantic meanings. These embeddings enable fast comparison using cosine similarity or vector indexing. They have become essential in modern NLP pipelines. SBERT employs a Siamese/bi-encoder architecture in which each sentence is encoded independently, as shown in Equation (6) where e is the fixed-size embedding vector.

$$e = \text{BERT}(s) \quad (6)$$

Each BERT encoder outputs contextualized token representations $H = [h_1, h_2, \dots, h_n]$, which are aggregated via mean pooling to form the final embedding, as illustrated in Equation (7)

$$e = \frac{1}{n} \sum_{i=1}^n h_i. \quad (7)$$

SBERT is trained using semantic similarity-oriented objectives such as Natural Language Inference (NLI), triplet loss, or cosine similarity loss, enabling embeddings to reflect semantic relationships (i.e., similar sentences are mapped closer in vector space). Similarity between two sentences s_1 and s_2 is computed using cosine similarity, which is computed in Equation (8).

$$\cos(e_1, e_2) = \frac{e_1 \cdot e_2}{\|e_1\| \|e_2\|}. \quad (8)$$

By encoding each sentence only once and storing embeddings for reuse, SBERT enables efficient large-scale semantic search, clustering, and retrieval while maintaining the contextual understanding of Transformer-based models. In this work, three pre-trained SBERT models are employed in the comparison study. These models aim to reflect different accuracy–efficiency trade-offs: “all-mpnet-base-v2” (BERT-1) for the highest semantic accuracy, “all-MiniLM-L12-v2” (BERT-2) for balanced performance, and “all-MiniLM-L6-v2” (BERT-3) for large-scale or real-time feature extraction [49].

4.3.2. Retrieval-Oriented Embedding Models (R-O)

Retrieval-oriented embedding models are trained to maximize semantic similarity between related texts and separate unrelated texts in the embedding space. Such models often exhibit excellent transferability across downstream tasks. In this work, we consider two main models: BAAI/bge-large-en-v1.5 (BERT-4) and ntfloat/e5-base (BERT-5).

First, the BAAI/bge-large-en-v1.5 (BERT-4) model belongs to the BGE (BAAI General Embedding) family and is designed primarily for semantic retrieval and text matching tasks. It generates high-dimensional embeddings (1024 dimensions) and consistently achieves state-of-the-art results on retrieval benchmarks. The model’s enhanced semantic discrimination capabilities make it particularly suitable for detecting sophisticated phishing attempts that employ obfuscated or paraphrased language.

On the other hand, the e5-base (BERT-5) model is part of the E5 embedding family, which is trained using large-scale contrastive learning. It generates 768-dimensional embeddings and has demonstrated strong performance across retrieval, clustering, and classification tasks. The contrastive training strategy encourages semantically similar texts to occupy nearby regions of the embedding space, potentially improving the separability between legitimate and malicious SMS messages.

4.3.3. General-Purpose Language Models (GPL)

Unlike Sentence-Transformer and retrieval-oriented models, general-purpose language models are primarily trained for language understanding rather than sentence embedding generation. Nevertheless, they can be utilized as powerful feature extractors by applying suitable pooling strategies to their hidden representations. In this work, we consider microsoft/deberta-v3-base (BERT-6) model.

This model is based on the DeBERTa (Decoding-enhanced BERT with Disentangled Attention) architecture developed by Microsoft. Its disentangled attention mechanism separately models content and positional information, resulting in improved contextual understanding. Although it was not originally designed as a sentence embedding model, its contextual representations have shown excellent performance in many NLP tasks.

Consequently, it provides an interesting comparison between dedicated embedding models and general-purpose transformer architectures.

4.4. Sequence Models

Sequence models represent an alternative family of feature extraction techniques that learn sentence representations directly from ordered word sequences. Long Short-Term Memories (LSTMs) [50], Gated Recurrent Units (GRUs) [51], and Convolutional Neural Networks (CNNs) [52] encoders derived from these bases are the main sequence models that can extract the features from texts. Unlike static embeddings, sequence models capture temporal or local patterns in the data, such as repeated structures, positional clues, or ellipsis-like phrasing common.

LSTMs and GRUs effectively model contextual dependencies over short messages [51], while CNNs extract informative n-gram features through convolutional filters [52]. In this work, we investigate the performance of LSTM in terms of extracting features of text messages and compare its performance to other feature extraction mechanisms.

Long Short-Term Memory (LSTM):

These networks [50], are a specialized form of Recurrent Neural Network (RNN) designed to capture long-range dependencies in sequential data. Unlike traditional RNNs, LSTMs effectively mitigate the vanishing and exploding gradient problems, allowing them to learn contextual representations from long text sequences. In Natural Language Processing (NLP), LSTMs are widely used for feature extraction, transforming raw textual sequences into dense, information-rich embedding vectors. These vectors capture temporal and semantic relationships between words in the documents. This makes them suitable for classification and sentiment analysis purposes. To extract features from text, the LSTM network takes the input sequences word by word. These words are inserted into the network, and the LSTM generates hidden states that capture local and long-term dependencies among words of the text.

5. Experimental Results and Discussions

This set of experiments investigates the impact of different text feature extraction models on the performance of ML-based SMS spam detection mechanisms. Different sets of experiments have been used to test and compare the performance of classical lexical features, static word-embedding, contextualized embeddings, R-O embedding techniques, and sequence-based modeling. To ensure a fair and comprehensive comparison, these models were evaluated under identical experimental conditions. This starts by selecting the same set of basic machine learning classification models for all experiments. These classification models are Support Vector Machines (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), Decision Trees (DT), and Logistic Regression (LR). Then, using the same dataset (i.e., UCI repository) to train all models. Finally, test all trained models on the same unseen merged dataset (i.e., out-of-domain dataset). This aims to check the generality and scalability of the trained models in order to detect unknown spam patterns in the SMS messages.

First, classical lexical features were examined using the Term Frequency–Inverse Document Frequency (TF-IDF) model. Second, static word-embedding models were evaluated using pre-trained Global Vectors for word representation (GloVe). Third, contextualized embeddings were investigated using three pre-trained Sentence-BERT (SBERT) models: all-mpnet-base-v2 (BERT-1), all-MiniLM-L12-v2 (BERT 2), and all-MiniLM-L6-v2 (BERT-3). In addition, retrieval-oriented embedding and general-purpose language models were included in the comparative study using three transformer models available through HuggingFace: BAAI/bge-large-en-v1.5 (BERT-4), intfloat/e5-base (BERT-5), and microsoft/deberta-v3-base (BERT-6). Finally, sequence-

based modeling was explored using a Long Short-Term Memory (LSTM) network to capture word-order dependencies and temporal patterns within SMS messages. The performance of each model was evaluated using four main metrics: accuracy, precision, recall, and F1-score. The results are presented and analyzed in the rest of this Section, with particular attention paid to robustness, generalization ability, and the relative strengths and limitations of each model.

5.1. Classical Lexical Features: Impact of TF-IDF Feature Dimensionality

These experiments evaluate the performance of a classical ML-based spam detection model when the TF-IDF mechanism is used to extract the features from the text messages. Four TF-IDF configurations were considered: the number of maximum features set to 500, 1000, 2500, and 5000, respectively. The performance of these models is evaluated on two different datasets: the UCI SMS Spam Collection dataset (in-domain) and a merged unseen SMS dataset (out-of-domain). Figure 3 graphically illustrates the results of these experiments. The effects of increasing the number of considered features for both considered datasets are shown in Figure 3 as well.

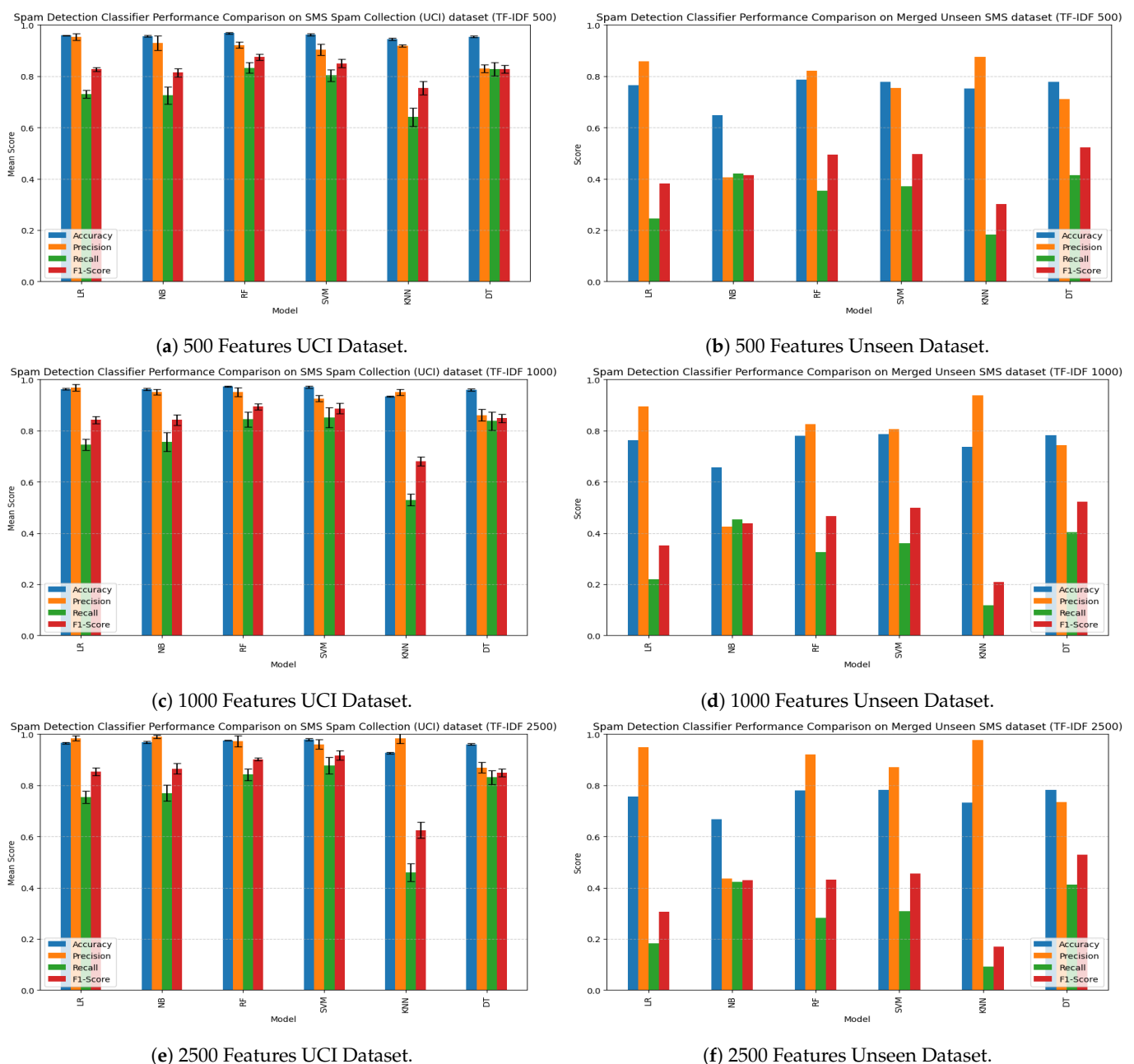


Figure 3. Cont.

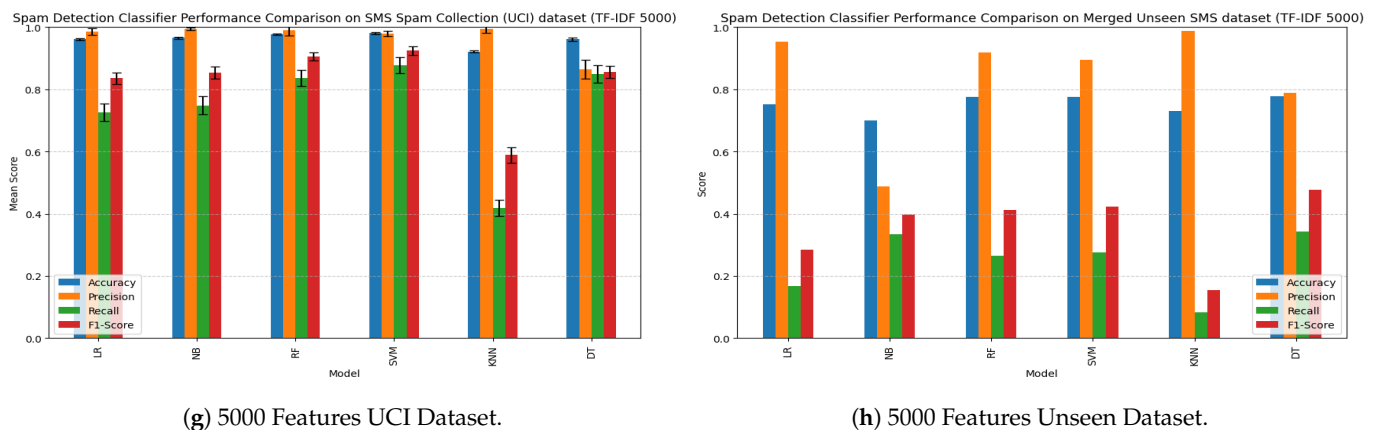


Figure 3. The effect of different numbers of TF-IDF features on the performance of ML-based Spam Detection.

5.1.1. Performance on the UCI SMS Spam Collection Dataset

For all TF-IDF configurations, the ML-based spam classifiers achieved strong performance on the UCI dataset (on-domain). This ensures that lexical representations are highly effective when training and testing data from the same distribution. Figure 3a,c,e,g, graphically show that. We can also infer from these figures, by increasing the number of TF-IDF features, the classification performance is generally improved. Among all the evaluated classifiers, the RF and SVM models consistently achieved the highest overall accuracy, reaching values around 98% when using 2500 and 5000 TF-IDF features. They also demonstrated competitive performance, especially in terms of the recall metric. This indicates strong spam detection capability with low false-negative rates.

From Figure 3e,g, we can infer that increasing the number of TF-IDF features achieved minor improvements for most classifiers. However, increasing the number of TF-IDF features will consume more resources and time with no tangible gain.

5.1.2. Performance on the Merged Unseen SMS Dataset

For all ML-based classifiers and TF-IDF configurations, a significant reduction in performance is observed when evaluated on the unseen dataset. This is due to the limited generalization ability of lexical classical lexical features. Accuracy values drop to 65–79%, while recall and F1-score experience more reductions. This indicates difficulty in correctly identifying spam messages originating from different distributions. Figure 3b,d,f,h illustrate these results graphically.

An additional observation is that increasing the TF-IDF vocabulary size from 500 to 5000 features does not improve generalization performance. In fact, recall generally decreases as the feature dimension grows, suggesting that larger vocabularies introduce dataset-specific terms that lead to overfitting. The best generalization results are consistently obtained using the more compact TF-IDF representations (i.e., 500–1000 features).

Overall, the unseen-dataset experiments demonstrate that strong performance on a benchmark dataset does not necessarily translate to effective real-world spam detection. The results emphasize the importance of evaluating classifiers on heterogeneous external datasets and indicate that SVM, RF, and DT provide the most reliable generalization performance under domain shift conditions.

5.2. Static Word-Embedding: Impact of Static Word Embeddings (GloVe)

This set of experiments evaluates the effectiveness of static word-embedding representations for SMS spam detection by using pre-trained GloVe vectors. Four embedding dimensions: 50, 100, 200, and 300 were investigated to analyze the influence of embedding

size on classification performance. Results are also reported on the UCI SMS Spam Collection dataset (in-domain) and a merged unseen SMS dataset (out-of-domain). Figure 4 graphically illustrates the comparative results for different dimensions and different classification models.

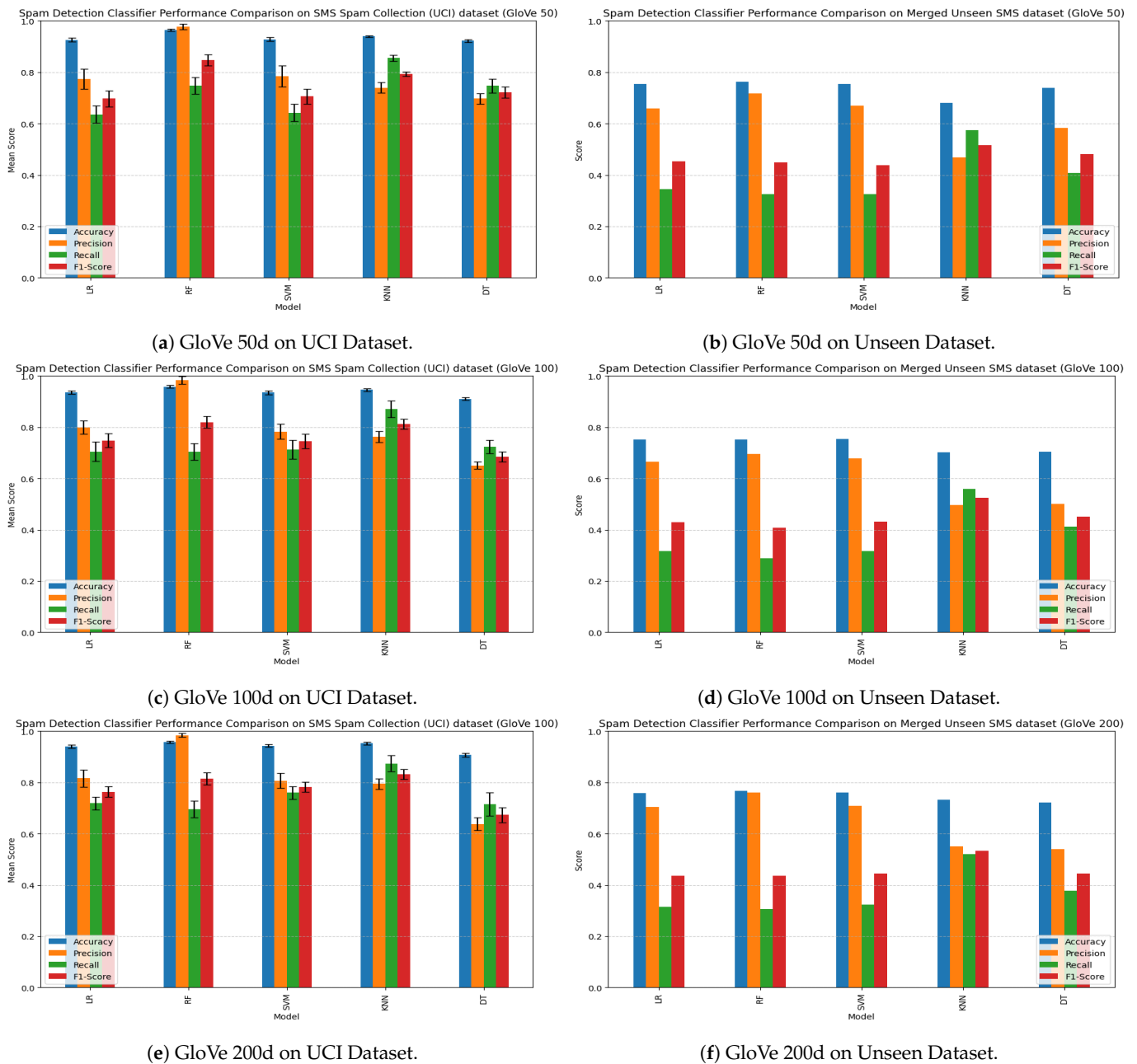


Figure 4. Cont.

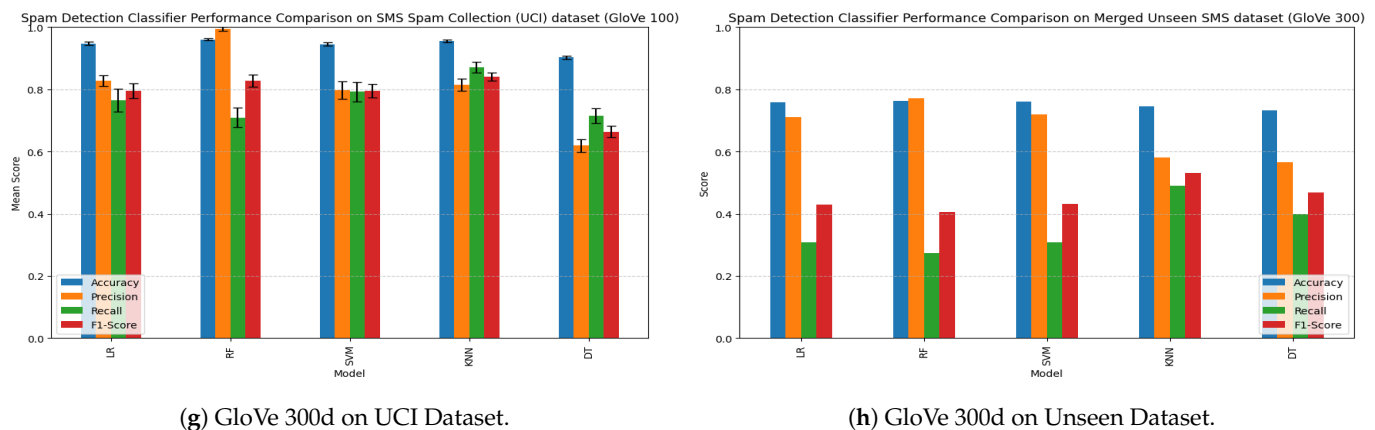


Figure 4. The effect of different dimensions of GloVe features on the performance of ML-based Spam Detection.

5.2.1. Performance on the UCI SMS Spam Collection Dataset

Overall, GloVe-based representations achieve strong performance on the UCI dataset (on-domain), although they generally underperform compared to high-dimensional TF-IDF configurations. Figure 4a,c,e,g show that by increasing the embedding dimension from 50 to 300, small improvements in accuracy and F1-score are achieved for most of the considered classifiers. This shows that higher-dimensional embeddings capture richer semantic information. In more detail, among the evaluated classifiers, the RF model achieves the highest accuracy across all embedding dimensions, reaching 98–99% for 200 and 300-dimensional embeddings. The SVM and LR classifiers also achieve good performance with high recall values, ensuring that static embeddings are effective for identifying spam-related keywords and patterns. In contrast, the KNN model achieves relatively high recall but lower precision, particularly at smaller embedding sizes, which negatively affects its F1-score.

5.2.2. Performance on the Merged Unseen SMS Dataset

These experiments evaluate the trained models on the out-of-domain dataset to measure the generality of the developed models. GloVe-based models achieve an improved robustness compared to TF-IDF; this is mainly in the recall and F1-score metrics. However, the accuracy is still low when the domain shift occurs; the decline is less than that observed with lexical features extraction. This shows that static embeddings offer better generalization by capturing semantic similarity beyond exact word matching. The LR and SVM models show the best performance in all tested dimensions. The accuracy values are 70%–77%. The KNN model also benefits from increased embedding dimensions, with higher recall values compared to TF-IDF-based KNN. This reflects an improved neighborhood structure in dense embedding spaces. However, the precision remains relatively low, limiting overall F1-score gains. As we can infer from Figure 4f,h, the embedding dimensions of 200 and 300 outperform smaller configurations, especially in recall. Higher-dimensional embeddings are more effective at capturing diverse spam expressions present in the unseen dataset.

Compared to TF-IDF, GloVe representations provide improved generalization to unseen datasets due to their ability to encode semantic similarity. However, these representations are unable to model contextual meaning, which restricts their effectiveness when the text contains subtle linguistic variations or word sense ambiguity. These findings also motivate the exploration of contextualized embedding models, which are expected to further enhance robustness and generalization. Accordingly, the next section investigates transformer-based contextual representations using SBERT models.

5.3. Contextualized Embeddings: The Performance of Different SBERT Models

These experiments mainly investigate the effectiveness of contextualized sentence embeddings for ML-based SMS spam detection models. Three pre-trained Sentence-BERT (SBERT) models: all-mpnet-base-v2 (BERT-1), all-MiniLM-L12-v2 (BERT-2), and all-MiniLM-L6-v2 (BERT-3) [49] have been used in these experiments. These models differ in architecture size and represent a trade-off between semantic accuracy and computational efficiency. The performance of each model was evaluated on both the UCI SMS Spam Collection dataset (in-domain) and the merged unseen SMS dataset (out-of-domain). Figure 5 graphically illustrates the performance of these models on the traditional ML-based classifiers.

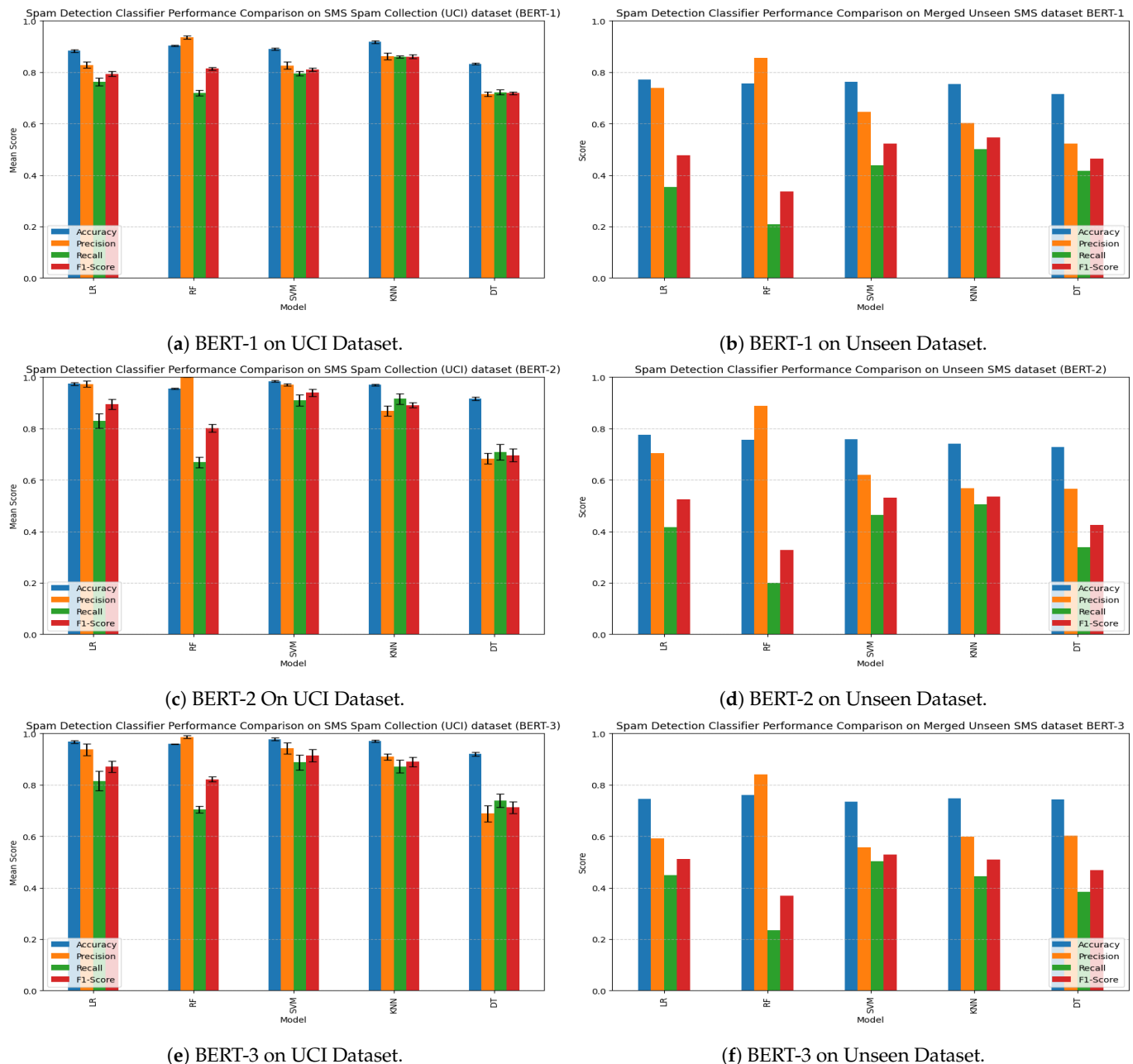


Figure 5. The effect of different pre-trained BERT models on the performance of ML-based Spam Detection.

5.3.1. Performance on the UCI SMS Spam Collection Dataset

All three SBERT variants achieved excellent performance on the UCI dataset. The accuracy values consistently exceeded 97% for most classifiers. This confirms the strong

representational capacity of contextual embeddings in capturing the semantics and the syntax-level word patterns. The RF and LR models achieve the highest accuracy and F1-scores across all SBERT variants. The SVM classifier also shows strong and stable performance, particularly in recall, indicating robust spam detection with low false-negative rates. The DT model achieves slightly lower performance, suggesting limited capacity to exploit dense contextual embeddings fully.

The KNN model demonstrates a very high recall score for all SBERT models, but it suffers from a low precision score. This behavior results in lower F1-scores and indicates that while KNN effectively captures semantic similarity in embedding space, it is more prone to false positives. Overall, as we can see from Figure 5a,c,e, performance differences among BERT-1, BERT-2, and BERT-3 on in-domain data are marginal. This means even lightweight SBERT models can adequately model the semantics present in the UCI dataset (in-domain).

5.3.2. Performance on the Merged Unseen SMS Dataset

When evaluating the SBERT models on the merged unseen dataset, we can see that SBERT-based representations achieve substantially improved generalization compared to both TF-IDF and GloVe features. From Figure 5b,d,f, although performance reduction due to domain shift is still there, the decline is notably smaller, particularly in recall and F1-score.

Among the three variants of the tested SBERT, BERT-1 (all-mpnet-base-v2) consistently achieves the strongest generalization performance, with the highest recall and F1-scores across most classifiers. BERT-2 and BERT-3 follow closely, demonstrating that compact transformer models can still preserve robust semantic representations for spam detection. LR and SVM models maintain the most stable performance on the unseen dataset. They achieve accuracy values in the range of approximately 74–78%.

These results indicate that contextual embeddings reduce the reliance on exact lexical overlap. Thus, they improve resilience to evolving spam patterns and unseen message structures. The results of the SBERT models show the clear advantage of contextualized embeddings over classical and static feature representations, especially for domain shift scenarios. By modeling word meaning in its context, the SBERT models capture the semantics and its phrasing strategies. This leads to better recall results and then more overall robustness.

5.4. The Performance of Different Retrieval-Oriented (R-O) Embedding and General-Purpose Language Models

Here, the performance of large transformer-based sentence embedding models for SMS spam detection is experimentally evaluated. Three pre-trained models available through HuggingFace were considered: BAAI/bge-large-en-v1.5 (BERT-4), intfloat/e5-base (BERT-5), and microsoft/deberta-v3-base (BERT-6). These models employ deep transformer architectures trained using contrastive or discriminative objectives to produce high-quality semantic sentence representations. The performance was assessed on both the UCI SMS Spam Collection (in-domain) dataset and the merged unseen SMS dataset (out-of-domain). Figure 6 graphically illustrates the results of these experiments.

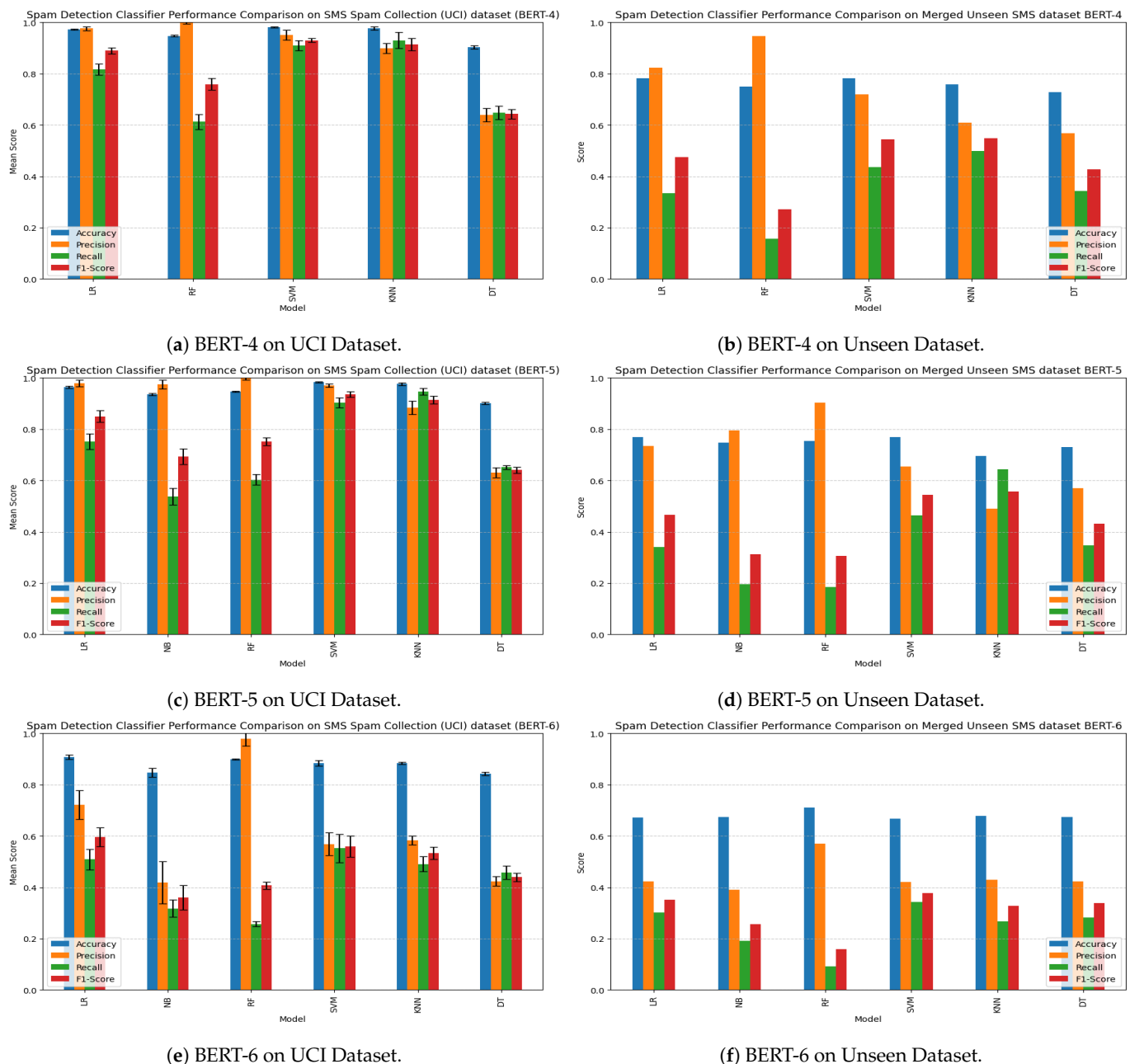


Figure 6. The effect of retrieval-oriented embedding and general-purpose language models on the performance of ML-based spam detection.

5.4.1. Performance on the UCI SMS Spam Collection Dataset

All three embedding models achieve excellent performance on the UCI dataset, with accuracy values more than 90% for all classifiers, as we can see from Figure 6a,c,e. RF and LR achieve the highest accuracy and F1-score values. This ensures strong compatibility between dense semantic embeddings and linear or ensemble machine learning classifiers.

The SVM model shows robust performance in the recall metric. This confirms the ability of R-O embeddings to capture the semantics of the text. Among the three tested models, BERT-4 (BGE-large) and BERT-5 (E5-base) achieve the best balance between precision and recall values. The BERT-6 (DeBERTa-v3) shows slightly higher recall at the expense of precision in certain classifier configurations.

5.4.2. Performance on the Merged Unseen SMS Dataset

On the merged unseen dataset, these embedding models achieve the best generalization performance compared to all other evaluated feature-extraction models tested in this work. Some performance reduction is noticed when the domain is shifted to the unseen dataset. However, this reduction is significantly smaller compared to TF-IDF, GloVe, and even SBERT models. As we can see from Figure 6b,d,f, the LR model achieves the highest accuracy and F1-score in all three models, and the accuracy values are in the range of 75–80%. The SVM also shows strong stability, particularly in recall, indicating reliable spam detection for an unseen dataset. In contrast, the RF model witnesses high variability in recall values, showing sensitivity to embedding distribution differences between training and test domains. In general, BERT-4 (BGE-large) achieves the highest performance and the three tested models.

The general results show that retrieval-oriented embedding and general-purpose language models provide the strongest semantic representations for spam detection, particularly in out-of-domain scenarios. These findings position these models as a powerful and robust alternative to both static and SBERT-based representations.

5.5. Performance of Sequence-Based Modeling (LSTM)

These experiments evaluate the effectiveness of sequence-based modeling for SMS spam detection using a Long Short-Term Memory (LSTM) network. A single-layer Long Short-Term Memory (LSTM) network has been simulated here. Messages were tokenized using a vocabulary of 20,000 words and padded to a fixed length of 50 tokens. Each token was represented by a 100-dimensional embedding vector, after which the sequences were processed by an LSTM layer with 128 hidden units. The final hidden state of the LSTM was used as a 128-dimensional feature vector for each message. Padding tokens were masked during processing, while the default Keras LSTM configuration was adopted, with no dropout or recurrent dropout. The performance in these experiments was evaluated on both the UCI SMS Spam Collection dataset (in-domain) and the merged unseen SMS dataset (out-of-domain). The results are illustrated graphically in Figure 7.

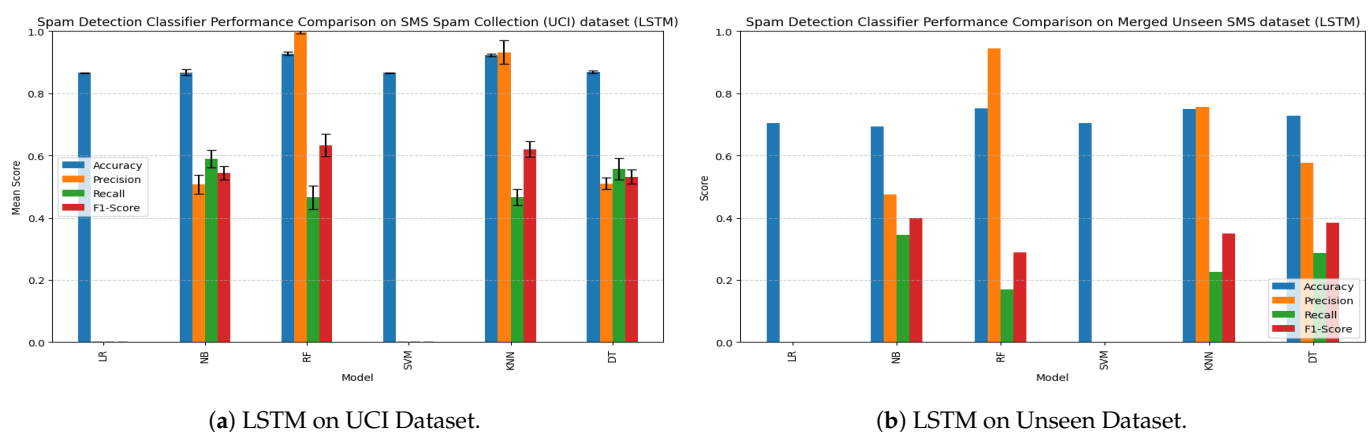


Figure 7. The effect of LSTM feature extraction model on the performance of ML-based spam detection.

5.5.1. Performance on the UCI SMS Spam Collection Dataset

On the UCI dataset, the LSTM model has lower performance compared to the transformer-based embedding methods. Among the evaluated classifiers, the RF model achieves the highest accuracy, exceeding 93%, while KNN and DT achieve lower performance, as we can see from Figure 7a.

Although LSTM models are capable of capturing sequential patterns, their precision and F1-scores remain lower than those achieved by SBERT and R-O embeddings. The SVM

and LR models achieve relatively low precision, indicating a higher false-positive rate. These results conclude that while LSTM can model local word order effectively, it cannot capture the broader semantic variations characteristic of spam messages when trained on limited datasets.

5.5.2. Performance on the Merged Unseen SMS Dataset

When evaluating the trained models on the merged unseen dataset, LSTM-based models show lower performance, especially in recall and F1-score. Accuracy values are reduced to 66–75% across classifiers, as we can see from Figure 7b. The recall value decreases sharply in several configurations, particularly for RF and SVM models. The KNN model demonstrates relatively balanced performance on unseen dataset. It achieves comparable precision and recall at modest levels. This behavior suggests that LSTM-derived embeddings capture some degree of semantic similarity but lack the robustness of transformer-based representations.

Overall, the generalization performance of LSTM is lower than that of SBERT and R-O embeddings. It is sensitive to domain shift and limited vocabulary coverage.

6. Discussion and Recommendations

This section aims to consolidate the experimental results across all feature-extraction paradigms and provide a high-level view of classification behavior under both in-domain and out-of-domain evaluation settings. It reveals how different modeling choices affect accuracy, precision, recall, and F1-score, and it explains why certain paradigms are more suitable for real-world SMS spam detection. Table 4 summarizes the best feature extraction models for SMS spam detection models. As we can infer from the table, the SVM model achieves the best accuracy performance when it is joined with R-O embedding for the in-domain dataset. It maintains the best score among all other feature extractions and spam detection mechanisms when it experiences a shift in the tested dataset. The contextual embedding techniques achieve comparative results in this context as well. They highly outperform the performance of other feature paradigms.

Table 4. Summary of best feature-extraction mechanisms for SMS spam detection.

Feature Extraction Mechanism	Best Feature Model	Best Classifier	In-Domain Performance (UCI)	Out-of-Domain Performance (Unseen)
Lexical (Classical)	TF-IDF (500–1000 Features)	SVM / RF	97–98% Accuracy	≈78% Accuracy
Static Embeddings	GloVe (200–300 Dimensions)	KNN	95–96% Accuracy	74–76% Accuracy
Contextual Embeddings	SBERT (all-mpnet-base-v2)	SVM	97–98% Accuracy	75–77% Accuracy
R-O Embedding	BAAI/bge-large-en-v1.5 ntfloat/e5-base	SVM / KNN	98% Accuracy	76–78% Accuracy
GPL Embedding	microsoft/deberta-v3-base	SVM	88–91% Accuracy	67–71% Accuracy
Sequence Models	LSTM	NB	93% Accuracy	69–75% Accuracy

Most feature extraction mechanisms achieve high accuracy on the in-domain dataset. Table 5 summarizes the best in-domain performance per feature extraction mechanism. TF-IDF, SBERT, and R-O embeddings all reach accuracy values close to 98%. This indicates

that the dataset is relatively well structured and separable when training and testing data follow the same distribution.

However, despite similar accuracy values, the underlying error characteristics differ. TF-IDF achieves high precision but slightly lower recall, which means it correctly classifies most predicted spam messages but fails to detect a portion of spam messages whose vocabulary deviates from the training data. On the other hand, SBERT and R-O embeddings achieve higher recall, reflecting their ability to identify spam messages even when messages use varied phrasing or implicit intent.

Table 5. Best in-domain (UCI dataset) performance per feature-extraction mechanism.

Paradigm	Best Feature Model	Best Classifier	Accuracy	Precision	Recall	F1-Score
TF-IDF	5000 features	SVM	98%	98%	88%	92%
GloVe	300-dim	KNN	95%	81%	87%	84%
Contextual Embeddings	all-mpnet-base-v2	SVM	98%	97%	91%	94%
R-O Embedding	BAAI/bge-large-en-v1.5	SVM	98%	95%	91%	93%
GPL Embedding	microsoft/deberta-v3-base	SVM	89%	57%	56%	56%
LSTM	LSTM	RF	93%	100%	47%	63%

The out-of-domain results provide the most critical insight into model robustness. All paradigms experience a reduction in all the measured metrics of the performance evaluation process when evaluated on unseen data. However, the degree of this decline varies substantially. R-O embeddings achieve the highest overall out-of-domain performance. This is mainly in recall and F1-score values. Their superior performance suggests that large-scale pre-training and sentence-level semantic alignment enable more reliable detection of previously unseen spam patterns. Table 6 summarizes the best out-of-domain (unseen dataset) performance per feature-extraction mechanism. As we can infer from the Table, a substantial drop in the recall of the merged dataset for all models. This implies that the systems miss some spam messages coming from new sources. This is simply justified by the African slang in the Nigerian dataset and the different specifics of the Mendeley scam that can be detected easily by natural language processing.

Table 6. Best out-of-domain (unseen dataset) performance per feature-extraction mechanism.

Paradigm	Best Feature Model	Best Classifier	Accuracy	Precision	Recall	F1-Score
TF-IDF	500–1000 features	SVM	78%	81%	36%	50%
GloVe	300-dim	KNN	75%	58%	49%	53%
SBERT	all-mpnet-base-v2	SVM	76%	65%	50%	53%
R-O Embedding	BAAI/bge-large-en-v1.5	SVM	78%	72%	44%	55%
GPL Embedding	microsoft/deberta-v3-base	SVM	67%	42%	35%	38%
LSTM	LSTM	NB	69%	48%	35%	40%

Table 7 summarizes the relative robustness of each paradigm on unseen data. R-O embeddings rank first, followed closely by SBERT, reflecting the dominant role of semantic and contextual understanding in modern spam detection. Static embeddings and LSTM models occupy the middle range, while TF-IDF ranks lowest due to its limited ability to generalize beyond the training vocabulary. An important observation from this ranking is that high in-domain accuracy does not guarantee strong generalization. Models with similar accuracy on the UCI dataset differ significantly in their ability to maintain recall and F1-score under domain shift. This reinforces the necessity of evaluating spam detection systems on unseen data and prioritizing metrics such as recall and F1-score over accuracy alone.

Table 7. Generalization ranking based on unseen dataset performance.

Rank	Feature Paradigm	Best F1-Score
1	R-O Embeddings (BGE / E5)	56%
2	SBERT Contextual Embeddings	53%
2	GloVe Static Embeddings	53%
4	TF-IDF Lexical Features	50%
5	LSTM Sequence Modeling	40%
6	GPL Embedding (DeBERTa-v3)	38%

Taken together, the three tables demonstrate that robust spam detection requires semantic-rich representations capable of generalizing beyond surface-level lexical patterns. Transformer-based embeddings, particularly R-O models, consistently provide the most balanced and reliable performance across all evaluation settings. These findings strongly support the transition from traditional lexical and sequence-based approaches toward large-scale pre-trained semantic models for real-world spam detection applications.

7. Conclusions and Future Work

This paper presented a comprehensive evaluation of multiple feature representation strategies for ML-based SMS spam detection, including TF-IDF, GLoVe, BERT, GPT, and LSTM-based representations. Extensive experiments conducted on the UCI SMS Spam Collection dataset and a merged unseen SMS dataset demonstrate that feature representation plays a critical role in both in-domain classification performance and cross-dataset generalization. The results show that TF-IDF representations achieve strong performance on in-domain data but suffer from limited robustness under domain shift. GLoVe embeddings improve generalization by incorporating semantic information, while LSTM-based feature extraction further decrease the performance by modeling sequential dependencies. Among all evaluated approaches, R-O embedding representations consistently achieve the highest accuracy, recall, and F1-score across both datasets, indicating superior robustness to unseen spam patterns. BERT-based representations also demonstrate competitive performance, particularly in recall-sensitive scenarios. Future work will focus on end-to-end fine-tuning of transformer models, hybrid representations combining contextual and task-aware features, and model compression techniques to enable deployment in resource-constrained environments. Additionally, extending the framework to multilingual and evolving spam datasets remains an important direction for further investigation.

In summary, this work establishes a strong empirical foundation for understanding the role of feature extraction in spam detection and highlights transformer-based semantic embeddings as the most effective approach for modern and evolving spam environments. Continued research along the outlined directions will further advance the development of robust, scalable, and trustworthy spam detection systems.

Author Contributions: Conceptualization, A.A. and M.B.Y.; methodology, M.B.Y.; software, A.A.; validation, A.A. and M.B.Y.; formal analysis, M.B.Y.; investigation, A.A.; resources, A.A.; data curation, A.A. and M.B.Y.; writing—original draft preparation, A.A.; writing—review and editing, M.B.Y.; visualization, A.A.; project administration, M.B.Y.; funding acquisition, M.B.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Datasets are available on reasonable request.

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

References

1. Ahmed, N.; Amin, R.; Aldabbas, H.; Koundal, D.; Alouffi, B.; Shah, T. Machine learning techniques for spam detection in email and IoT platforms: Analysis and research challenges. *Secur. Commun. Netw.* **2022**, *2022*, 1862888.
2. Crawford, M.; Khoshgoftaar, T.M.; Prusa, J.D.; Richter, A.N.; Najada, H.A. Survey of review spam detection using machine learning techniques. *J. Big Data* **2015**, *2*, 23.
3. Kontsewaya, Y.; Antonov, E.; Artamonov, A. Evaluating the effectiveness of machine learning methods for spam detection. *Procedia Comput. Sci.* **2021**, *190*, 479–486.
4. Rao, S.; Verma, A.K.; Bhatia, T. A review on social spam detection: Challenges, open issues, and future directions. *Expert Syst. Appl.* **2021**, *186*, 115742.
5. Jeong, S.; Noh, G.; Oh, H.; Kim, C.-k. Follow spam detection based on cascaded social information. *Inf. Sci.* **2016**, *369*, 481–499.
6. Song, J.; Lee, S.; Kim, J. Spam filtering in Twitter using sender-receiver relationship. In *Recent Advances in Intrusion Detection, 14th International Symposium, RAID 2011, Menlo Park, CA, USA, 20–21 September 2011*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 301–317.
7. Sinha, P.; Maini, O.; Malik, G.; Kaushal, R. Ecosystem of spamming on Twitter: Analysis of spam reporters and spam reportees. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, India, 21–24 September 2016*; IEEE: New York, NY, USA, 2016; pp. 1705–1710.
8. Oswald, C.; Simon, S.E.; Bhattacharya, A. Spotsam: Intention analysis-driven SMS spam detection using bert embeddings. *ACM Trans. Web (TWEB)* **2022**, *16*, 1–27.
9. Oyeyemi, A.D.; Ojo, A.K. SMS Spam Detection and Classification to Combat Abuse in Telephone Networks Using Natural Language Processing. *arXiv* **2024**, arXiv:2406.06578.
10. Bani, Y.M.; Ababneh, A. Machine Learning Based Spam Detection in Digital Communication Systems: A Comparative Analysis. *Systems* **2026**, *14*, 229.
11. Grumbletext. Available online: <https://www.grumbletext.co.uk/> (accessed on 10 April 2026).
12. Chen, T.; Kan, M. Creating a Live, Public Short Message Service Corpus: The NUS SMS Corpus. *Lang. Resour. Eval.* **2013**, *47*, 299–335. Available online: <https://www.nus.edu.sg/> (accessed on 10 April 2026).
13. Tang, S.; Mi, X.; Li, Y.; Wang, X.; Chen, K. Clues in tweets: Twitter-guided discovery and analysis of SMS spam. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, Los Angeles, CA, USA, 7–11 November 2022*; Association for Computing Machinery: New York, NY, USA, 2022; pp. 2751–2764.
14. Al Saidat, M.R.; Yerima, S.Y.; Shaalan, K. Advancements of SMS spam detection: A comprehensive survey of NLP and ML techniques. *Procedia Comput. Sci.* **2024**, *244*, 248–259.
15. Johari, M.F.; Chiew, K.L.; Hosen, A.R.; Yong, K.S.C.; Khan, A.S.; Abbasi, I.A.; Grzonka, D. Key insights into recommended SMS spam detection datasets. *Sci. Rep.* **2025**, *15*, 8162.
16. Maqsood, U.; Rehman, S.U.; Ali, T.; Mahmood, K.; Alsaedi, T.; Kundi, M. An intelligent framework based on deep learning for SMS and e-mail spam detection. *Appl. Comput. Intell. Soft Comput.* **2023**, *2023*, 6648970.
17. Jain, H.; Mahadev, M. An Analysis of SMS Spam Detection using Machine Learning Model. In *Proceedings of the 2022 Fifth International Conference on Computational Intelligence and Communication Technologies (CCICT), Sonapat, India, 8–9 July 2022*; IEEE: New York, NY, USA, 2022; pp. 151–156.
18. 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. In *Proceedings of the Cyber-Physical Systems, Emden, Germany, 12–15 May 2025*; pp. 1–27.
19. Al-Kabbi, A.H.; Feizi-Derakhshi, M.; Pashazadeh, S. Multi-type feature extraction and early fusion framework for SMS spam detection. *IEEE Access* **2023**, *11*, 123756–123765.
20. Liu, X.; Lu, H.; Nayak, A. A spam transformer model for SMS spam detection. *IEEE Access* **2021**, *9*, 80253–80263.
21. Uddin, M.A.; Islam, M.N.; Maglaras, L.; Janicke, H.; Sarker, I.H. ExplainableDetector: Exploring transformer-based language modeling approach for SMS spam detection with explainability analysis. *Digit. Commun. Netw.* **2025**, *11*, 1504–1518.
22. Srinivasarao, U.; Sharaff, A. Machine intelligence based hybrid classifier for spam detection and sentiment analysis of SMS messages. *Multimed. Tools Appl.* **2023**, *82*, 31069–31099.
23. Gupta, S.D.; Saha, S.; Das, S.K. SMS spam detection using machine learning. *J. Physics Conf. Ser.* **2021**, *1797*, 012017.
24. Pudasaini, S.; Shakya, A.; Pandey, S.P.; Paudel, P.; Ghimire, S.; Ale, P. SMS spam detection using relevance vector machine. *Procedia Comput. Sci.* **2023**, *230*, 337–346.
25. Airlangga, G. Optimizing SMS spam detection using machine learning: A comparative analysis of ensemble and traditional classifiers. *J. Comput. Networks Archit. High Perform. Comput.* **2024**, *6*, 1942–1951.

26. Salman, M.; Ikram, M.; Kaafar, M.A. Investigating evasive techniques in SMS spam filtering: A comparative analysis of machine learning models. *IEEE Access* **2024**, *12*, 24306–24324.
27. Ozoh, P.; Ibrahim, M.; Ojo, R.; Sunmade, A.; Oyetayo, T. SMS Spam Detection Using Machine Learning Approach. *Int. STEM J.* **2025**, *6*, 10–27.
28. Li, Y.; Zhang, R.; Rong, W.; Mi, X. Spamdram: Towards privacy-preserving and adversary-resistant SMS spam detection. *arXiv* **2024**, arXiv:2404.09481.
29. Yerima, S.Y.; Bashar, A. Semi-supervised novelty detection with one class SVM for SMS spam detection. In *Proceedings of the 2022 29th International Conference on Systems, Signals and Image Processing (IWSSIP), Sofia, Bulgaria, 1–3 June 2022*; IEEE: New York, NY, USA, 2022; pp. 1–4.
30. Younes, M.B. Towards Preparing a Representative Phishing Dataset. In *Proceedings of the 2026 International Conference on Smart Multidomain Integrated Learning Environments (ICSMILE), Irbid, Jordan, 30–31 March 2026*; IEEE: New York, NY, USA, 2026; pp. 1–6.
31. Ahmad A.; Younes, M.B.; Gaashan, K., Dataset Preparation for Spam and Phishing Attacks Detection. In *Proceedings of the 17th International Conference on Information and Communication Systems (ICICS 2026), Irbid, Jordan, 18–21 May 2026*; ACM: New York, NY, USA, 2026; 6p. <https://doi.org/10.1145/3812734.3813413>.
32. Almeida, T.; Hidalgo, J. *SMS Spam Collection*; UCI Machine Learning Repository: Irvine, CA, USA, 2011. <https://doi.org/10.24432/C5CC84>.
33. Gómez, J.M.; Cajigas, G.; Puertas Sanz, E.; Carrero García, F. Content Based SMS Spam Filtering. In *Proceedings of the 2006 ACM Symposium on Document Engineering, Amsterdam, The Netherlands, 10–13 October 2026*; Association for Computing Machinery: New York, NY, USA, 2006.
34. Onashoga, A.S.; Abayomi-Alli, O.O.; Sodiya, A.S.; Ojo, D.A. An Adaptive and Collaborative Server-Side SMS Spam Filtering Scheme Using Artificial Immune System. *Inf. Secur. J. Glob. Perspect.* **2015**, *24*, 133–145.
35. Federal University of Agriculture, Abeokuta. Available online: <https://funaab.edu.ng/> (accessed on 10 April 2026).
36. Mishra, S.; Soni, D. *SMS Phishing Dataset for Machine Learning and Pattern Recognition*; Mendeley Data, V1; Mendeley: Amsterdam, The Netherlands, 2022. <https://doi.org/10.17632/f45bkkt8pr.1>
37. Kaggle. SMS Spam Collection Dataset. Available online: <https://www.kaggle.com/datasets/tinu10kumar/sms-spam-dataset> (accessed on 10 April 2026).
38. Tumma, C.; Reddy, R.A.; Yadav, R.; Salati, A.; Surendarkumar, S.; Karthikayen, A. Robust Jailbreak Detection in Large Language Models Using Behavioral Fingerprinting. In *Proceedings of the 2025 International Conference on Modeling, Simulation & Intelligent Computing (MoSiCom), Dubai, United Arab Emirates, 10–12 December 2025*; IEEE: New York, NY, USA; pp. 284–289. <https://doi.org/10.1109/MoSiCom67153.2025.11398297>.
39. Shukla, P.K.; Veerasamy, B.D.; Alduaiji, N.; Addula, S.R.; Pandey, A.; Shukla, P.K. Fraudulent account detection in social media using hybrid deep transformer model and hyperparameter optimization. *Sci. Rep.* **2025**, *15*, 38447.
40. Wan, C.; Wang, Y.; Liu, Y.; Ji, J.; Feng, G. Composite feature extraction and selection for text classification. *IEEE Access* **2019**, *7*, 35208–35219.
41. Sasaki, M.; Shinnou, H. Spam detection using text clustering. In *Proceedings of the 2005 International Conference on Cyberworlds (CW'05), Singapore, 23–25 November 2005*; IEEE: New York, NY, USA, 2005; pp. 319–323.
42. Karo, I.M.K.; Karo, J.A.K.; Djasmayena, S.; Wahyudi, R. Sentiment Analysis of Tourist Reviews at Waterfront City Pangururan Using Naive Bayes and TF-IDF Algorithm. *J. Softw. Eng. Inf. Commun. Technol.* **2025**, *6*, pp. 13–20. <https://doi.org/10.17509/seict.v6i1.86210>
43. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, pp. 1–9. .
44. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2014; pp. 1532–1543.
45. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global Vectors for Word Representation. Available online: <https://nlp.stanford.edu/projects/glove/> (accessed on 10 April 2026).
46. Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; Volume 1 (Long and Short Papers), pp. 4171–4186.
47. Liu, X.; He, P.; Chen, W.; Gao, J. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. *arXiv* **2019**, arXiv:1904.09482.

48. Reimers, N.; Gurevych, I. The curse of dense low-dimensional information retrieval for large index sizes. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Virtual, 1–6 August 2021*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; Volume 2: Short Papers, pp. 605–611.
49. Galli, C.; Donos, N.; Calciolari, E. Performance of 4 pre-trained sentence transformer models in the semantic query of a systematic review dataset on peri-implantitis. *Information* **2024**, *15*, 68.
50. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780.
51. Cho, K.; Merriënboer, B.V.; Bahdanau, D.; Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv* **2014**, arXiv:1409.1259.
52. Kim, S.; Kim, D.; Cho, S.W.; Kim, J.; Kim, J. Highly efficient RNA-guided genome editing in human cells via delivery of purified Cas9 ribonucleoproteins. *Genome Res.* **2014**, *24*, 1012–1019.

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.