applied sciences

Article

Beyond Word-Based Model Embeddings: Contextualized Representations for Enhanced Social Media Spam Detection

Sawsan Alshattnawi [1,†], Amani Shatnawi [1,†], Anas M.R. AlSobeh [1,2,†] and Aws A. Magableh [1,3,*,†]

1 Faculty of Computer Science and Information Technology, Yarmouk University, Irbid 21163, Jordan; sawsan_kh@yu.edu.jo (S.A.); ashatnawi@yu.edu.jo (A.S.); anas.alsobeh@siu.edu (A.M.R.A.)
2 Information Technology, School of Computing, Southern Illinois University Carbondale, 1365 Douglas Drive, Carbondale, IL 62901, USA
3 Software Engineering, Computer and Information Sciences, Prince Sultan University, Riyadh 11586, Saudi Arabia
* Correspondence: amagable@psu.edu.sa
† These authors contributed equally to this work.

Abstract: As social media platforms continue their exponential growth, so do the threats targeting their security. Detecting disguised spam messages poses an immense challenge owing to the constant evolution of tactics. This research investigates advanced artificial intelligence techniques to significantly enhance multiplatform spam classification on Twitter and YouTube. The deep neural networks we use are state-of-the-art. They are recurrent neural network architectures with long- and short-term memory cells that are powered by both static and contextualized word embeddings. Extensive comparative experiments precede rigorous hyperparameter tuning on the datasets. Results reveal a profound impact of tailored, platform-specific AI techniques in combating sophisticated and perpetually evolving threats. The key innovation lies in tailoring deep learning (DL) architectures to leverage both intrinsic platform contexts and extrinsic contextual embeddings for strengthened generalization. The results include consistent accuracy improvements of more than 10–15% in multisource datasets, unlocking actionable guidelines on optimal components of neural models, and embedding strategies for cross-platform defense systems. Contextualized embeddings like BERT and ELMo consistently outperform their noncontextualized counterparts. The standalone ELMo model with logistic regression emerges as the top performer, attaining exceptional accuracy scores of 90% on Twitter and 94% on YouTube data. This signifies the immense potential of contextualized language representations in capturing subtle semantic signals vital for identifying disguised spam. As emerging adversarial attacks exploit human vulnerabilities, advancing defense strategies through enhanced neural language understanding is imperative. We recommend that social media companies and academic researchers build on contextualized language models to strengthen social media security. This research approach demonstrates the immense potential of personalized, platform-specific DL techniques to combat the continuously evolving threats that threaten social media security.

Keywords: online social network (OSNs); social network analysis; cybersecurity; spam detection; neural word embeddings; RNN; contextual word embeddings; LSTM; BERT; EMLO

check for updates

Citation: Alshattnawi, S.; Shatnawi, A.; AlSobeh, A.M.R.; Magableh, A.A. Beyond Word-Based Model Embeddings: Contextualized Representations for Enhanced Social Media Spam Detection. *Appl. Sci.* **2024**, *14*, 2254. https://doi.org/10.3390/app14062254

Academic Editor: Chilukuri K. Mohan

Received: 31 January 2024
Revised: 25 February 2024
Accepted: 5 March 2024
Published: 7 March 2024

1. Introduction

The proliferation of spam has evolved from basic email campaigns to widespread promotion of content across online social networks (OSNs). Fake accounts are created en masse to disseminate spam posts and manipulate self-promotion [1]. Unlike email, OSN posts contain multimedia—images, videos, URLs—alongside text. Generating impactful posts demands significant effort; hence, spammers aggressively reuse content to target more users.

However, most detection methods focus narrowly on post text, account activity patterns, and simplistic metadata like followers. Keyword frequency analysis techniques are easily fooled by word stuffing without considering semantic coherence, posting times, or languages used [2]. After closely examining the limitations of existing methods against evolving bot tactics, we identified the lack of robust feature engineering as a core research gap.

Empirical studies show over 30% of social media content is estimated as spam—far higher engagement than email [3]. Spam click-through rates exceeding 0.13% demonstrate the scale and sophistication of organized campaigns tailored to exploit OSN vulnerabilities [4]. This ever-adapting threat landscape poses significant challenges for identification methods to keep pace.

Our work looks to bridge these gaps through state-of-the-art natural language processing techniques for enriched semantic modeling—a crucial capability for uncovering disguised and context-aware spam. We focus on robust representation learning rather than superficial features to tackle the root cause. This issue has a significant impact on the user experience and can be challenging to deal with due to the ever-changing nature of new tactics and the increasing complexity of evolving spam. Therefore, the current DL methods are facing challenges in identifying social spam and keeping up with the ever-growing complexity of evolving spam [5]. The most proficient spam campaigns have a notable economic aspect that engages in questionable tactics to boost website traffic and generate false promotional results, leading to significant financial repercussions that present a multifaceted threat that impacts internet users globally, and search engine optimization (SEO) can be seen clearly, posing challenges for providers of these services and affecting search results too.

Dealing with spam on social media is a challenging task due to the constantly evolving tactics used by spammers. In this dynamic landscape, the fight against spam detection becomes an ongoing challenge that requires a continuous adaptation of defensive strategies to keep up with the sophisticated and inventive tactics employed by spammers [6]. Traditional word vectors often struggle to capture the complexities and different meanings words can have in different situations, which is crucial for identifying subtle and cleverly disguised spam. Accurately modeling the semantic context can be quite challenging when using these vectors [7,8]. DL models that can seamlessly adapt to different social media platforms present their own set of complexities, unique characteristics, and user behaviors, requiring models that can easily adapt and be flexible.

This study provides a novel comparative analysis of state-of-the-art contextualized word embeddings (BERT, ELMo) against traditional noncontextualized word vectors (Word2Vec, GloVe) for spam detection on social media platforms. Unlike previous works, we evaluate these advanced NLP models on two distinct datasets: Twitter and YouTube comments. We looked into an area that has not been studied much: How well contextualized embeddings work on informal, noisy textual data from social networks. We also compared standalone BERT and ELMo models against RNN-based (LSTM) architectures for spam classification.

There is a lack of comprehensive labeled datasets available for training adaptable spam detection models on various platforms, which are essential to training and improving spam detection algorithms [9]. However, the limited availability of these resources presents a significant obstacle when it comes to creating models that are highly efficient and widely applicable [10]. Addressing these challenges is vital for improving spam detection methods and maintaining the integrity and user experience of social media platforms [6]. We performed a lot of tests to find the best model–embedding combinations. This answers important questions about how contextualized representations work with various recurrent networks. Therefore, the uniqueness of this research lies in its comprehensive analysis of modern NLP techniques on evolving social media landscapes to advance spam detection systems. We compare these methods with traditional ones to find out how much better they are and how semantically rich embeddings can pick up on subtle spam behaviors on platforms that are getting more complicated. The goal is to make progress in this important area of cybersecurity. Several cybersecurity concerns can be addressed using DL methods and algorithms, including phishing detection, malware identification and classification, intrusion detection, and spam identification. Spam websites are used to spread malware, adult content, and phishing scams, and people spend time sorting through unwanted mail; necessary e-mails may get deleted by mistake. As a result, spam wastes bandwidth, storage space, and computing power, and it promotes a wide range of products and services, resulting in illegal activity. Spam filters have been developed to identify and remove undesirable messages utilizing techniques like keyword analysis and statistical algorithms. Those who participate in spamming often discover methods to outsmart these filters that depend on analyzing content [11]. The constant challenge of dealing with spam has driven the development of increasingly sophisticated techniques for detection and prevention. Spam filters have been developed to identify and remove undesirable messages utilizing techniques such as keyword analysis and statistical algorithms [12,13]. Spammers often discover methods to outsmart these filters that depend on analyzing content, making it a constant challenge to deal with spam. Increasingly sophisticated techniques for detection and prevention have been developed, such as content-based techniques and DL approaches that examine the terms, frequency, and distribution of words and phrases in message content [14].

Advanced techniques exist to analyze neighbor graphs, taking into account links and metadata to identify and mitigate suspicious nodes and links. Hybrid techniques combine content and metadata to achieve enhanced results, leveraging the strengths of both approaches. These techniques involve analyzing spam text, which brings us into the realm of natural language processing (NLP) [15]. Text classification is a popular application of NLP, as discussed in research studies [16,17]. The main objective of this research paper is to identify spam by categorizing text as either spam or nonspam (quality). NLP is rapidly evolving; the description of the word in the text to be studied, regardless of the type of problem to be solved, is one of the most essential pillars of NLP. Several ways have emerged to give weight to the word within the text and thus the ability to deal with it as a number that can be processed, compared, and elicit information; this transformation is called word embedding. Static word embedding means that once the words are learned, they do not change with the context. Handling the polysemy problem is challenging with static embeddings due to their unchanging nature. The meaning of a polysemous word can differ depending on the context. Dynamic embeddings, on the other hand, address this issue by assigning varying weights to the same word in different contexts. These dynamic embeddings, which come from language models that have already been trained, have performed better on several natural language processing (NLP) tasks than their static counterparts [18].

Many word embeddings have started to appear in the field, such as Word2vec [19] and Glove [20], which deal with the meaning of the word. They are context-independent, producing only one vector (embedding) for each word, combining all of the word's different senses into a single vector. BERT and ELMo are context-dependent and can generate different word embeddings for the same word depending on its position in the context. These models, trained on massive amounts of data, can significantly improve the performance of a wide range of NLP problems. As vocabulary size determines vector size, this can lead to lengthy, complex vectors that take up space and slow down algorithms. Moreover, embeddings are domain-specific and task-dependent, hindering transfer learning to a model with a different vocabulary on the same scale. However, modifying the vocabulary requires retraining the entire model from scratch. Therefore, contextual word embeddings become crucial in addressing these challenges and capturing the contextual nuances of words.

BERT and ELMo are two cutting-edge models that use pretrained representations to pick up on the semantic and syntactic information of words in different situations. This lets them learn from each other and improve their performance [5]. We apply these models to Twitter and YouTube data and compare them with older methods to find the best RNN architectures for classifying spam. We also provide a detailed comparative analysis, offering insights that could guide future efforts in spam detection and contribute to strengthening cybersecurity measures on rapidly growing OSNs. Our main goal is to not only improve current methodologies but also to direct future research trajectories in spam detection technologies. The key research questions explored are as follows:

- **RQ1**: Do contextualized embeddings (BERT, ELMo) outperform noncontextualized word vectors (Word2Vec, GloVe) for detecting spam?
- **RQ2**: How do RNN models (LSTM) compare with standalone BERT and ELMo models in terms of accuracy?
- **RQ3**: Which embedding technique and model combination achieves the highest performance across metrics?

2. Background and Literature Review

In spam detection in social media, machine learning algorithms use the context of words to identify spam messages. A known-word algorithm has words that are categorized in their negative, positive, or neutral states. It works by classifying the contexts of words in a message as of a certain category and assigning a score based on how many words fall into these categories. Although these algorithms produce good predictions, especially in NLP research, they are fraught with many false positives and negatives [21]. The "known-word" algorithms in spam detection assess messages regarding some static known-word context lists, which makes the method less context-relevant to social media. Limitations that the known-word algorithms in spam detection face today include the inability to capture known dynamic background user features, as well as the inability to update context when a message receives feedback [22,23].

One of the most exciting trends in the current era is deep learning, which is of great interest among scientists in scientific research. It has entered all fields and has offered excellent contributions at all levels: economic, cultural, educational, etc. The evolution of OSNs, specifically Twitter and Facebook, and their impact emphasize the significance of current, context-dependent spam detection context. For example, more than 95% of Twitter posts and comments are effectively spam-related. Moreover, a beginner will leverage machine learning techniques to develop different learning-based engineering projects in context-dependent spam detection [24].

Jain et al. [25] and Sedik et al. [26] proposed an innovative approach by enhancing CNNs with a semantic layer (SCNN). Leveraging resources like Word2Vec, WordNet, and ConceptNet, they achieved an impressive 98.65% accuracy on SMS and Twitter datasets. However, despite this success, their model still faces limitations related to the scalability of semantic resources and potential biases inherent in these knowledge bases. Building upon their previous work, Jain et al. [27] conducted a subsequent study. This time, they combined CNNs with LSTM architectures, incorporating knowledge bases such as WordNet and ConceptNet. While their hybrid model demonstrated superior performance, it remains sensitive to noisy or incomplete knowledge graphs. Additionally, the computational cost of training such models can be prohibitive.

The need for effective textual encoding in spam detection is evident, with basic word count vectors often falling short in semantic interpretation. Advances in neural word embeddings like BERT and ELMo have set new benchmarks in text encoding. Alom et al. [28] showed a deep learning strategy that uses Twitter content and user metadata to perform better than current methods, achieving 99.68% and 93.12% accuracy on two separate datasets. The prevalence of social media platforms and the large number of content and users they host present new challenges in detecting and stopping spam. In their study, Elakkiya et al. (2021) introduced a fresh method for identifying spam by utilizing a blend of CNN and BiLSTM models [29], bolstered by a conjoint attention mechanism [30]. The model showcased impressive accuracy across a range of datasets, including those from Twitter and YouTube. In the realm of burgeoning OSNs, the development of sophisticated spam filters is imperative. Barushka et al. [3] used ensemble learning with deep neural networks, outperforming traditional spam filtering algorithms. Sun et al. [31] proposed a real-time Twitter spam detection system, integrating account- and content-based features, trained on a large dataset of public tweets.

2.1. NLP Models: RNN and LSTM

Utilizing LSTM for text classification in social media involves addressing the diverse language styles and background noise found in data from various platforms. Various methods such as Bidirectional LSTM, distributed LSTM, and the fusion of CNN with LSTM have demonstrated potential in improving the comprehension of text semantics and syntax for tasks such as sentiment analysis [32], spam detection, and mental health evaluation. These models take advantage of the sequential nature of textual data to effectively capture long-range dependencies; nevertheless, there are challenges related to handling computational complexity and adjusting to the ever-changing language used on social media platforms [33]. By combining contextual embeddings from advanced models such as BERT with LSTM layers, a balance is achieved between model performance and generalization, which emphasizes the current research focus on developing more robust and adaptable text classification systems [34]. Figure 1 illustrates the concept of unrolling a loop in the context of RNNs [35], showcasing their efficiency in modeling sequential data, including textual content, with promising results; however, RNNs face challenges when long-term memory is crucial for problem-solving, such as in scenarios requiring the prediction of words in lengthy sentences where key information begins to fade as the sequence progresses. This situation can lead to a significant increase in the gap between necessary information and its point of use. As this gap widens, RNNs struggle to maintain the connection between relevant data points, a problem attributed to their sensitivity to vanishing and exploding gradients. To address these limitations, LSTM architectures introduce memory blocks for sustained connections, where each block contains a cell that holds a temporary state of the network within its memory, alongside dedicated structures that regulate the flow of information, ensuring that LSTMs can remember important information over longer sequences. Our work advances the field by addressing LSTM's inherent limitations through the integration of contextual embeddings and platform-specific customizations, thus offering a more robust and adaptable solution for spam detection in social media environments.

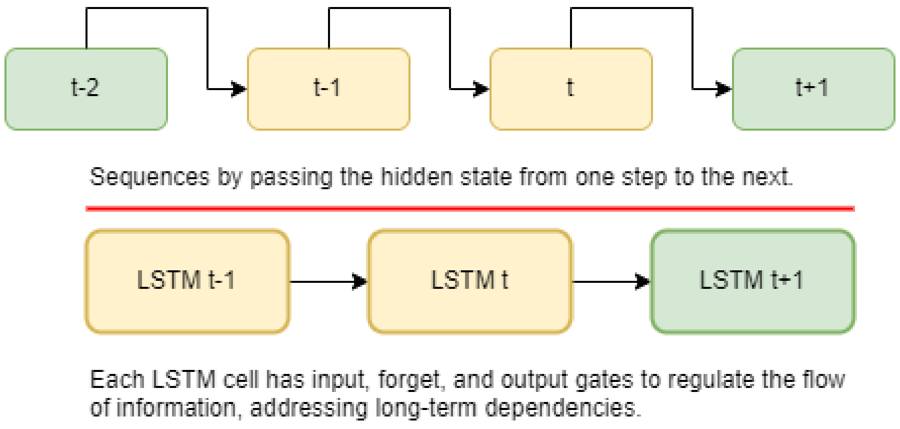
t-2
t-1
t
t+1
Sequences by passing the hidden state from one step to the next.
LSTM t-1
LSTM t
LSTM t+1
Each LSTM cell has input, forget, and output gates to regulate the flow of information, addressing long-term dependencies.

Figure 1. Unrolled RNN.

Khan et al. [36] presented standard feedforward neural networks, which have unidirectional connections. LSTMs feature feedback connections that enable them to process entire sequences of data, not just single data points. This design is specifically aimed at tackling the vanishing gradient problem, a common challenge in traditional RNNs that hampers their ability to recall information over extended periods, so LSTMs achieve this through the integration of memory cells and three distinct gates: input, output, and forget gates. These parts work together to control the flow of information, which lets the network store data for any amount of time and makes LSTMs perfect for tasks that require sequences of different lengths and long gaps between important events [37]. Consequently, LSTMs have become a cornerstone in deep learning (DL) for effectively categorizing and processing sequential and time-series data, overcoming the limitations associated with discharging and disappearing gradients that plagued earlier RNN models [29]. Tashtouth et al. [38] and Al-Eidi et al. [39] discussed implementing cybersecurity systems and highlighted the dual nature of technological advancements in cybersecurity, offering insights into both the challenges and opportunities presented by novel attack vectors and defensive mechanisms.

When constructing our LSTM model, the initial phase entails deciding which components of the cell state should be preserved or eliminated; therefore, our decision is facilitated by the "forget gate layer", which functions as a sigmoid layer. This evaluates the previous hidden state, denoted by h_{t-1}, and the current input, x_t, to allocate a value ranging from 0 to 1 for each part of the cell state C_{t-1}. A value close to 1 suggests that the element should be entirely retained ("keep it entirely"), whereas a value near 0 indicates that the element should be fully discarded ("eliminate it"). Figure 2 depicts this selective retention and elimination process.

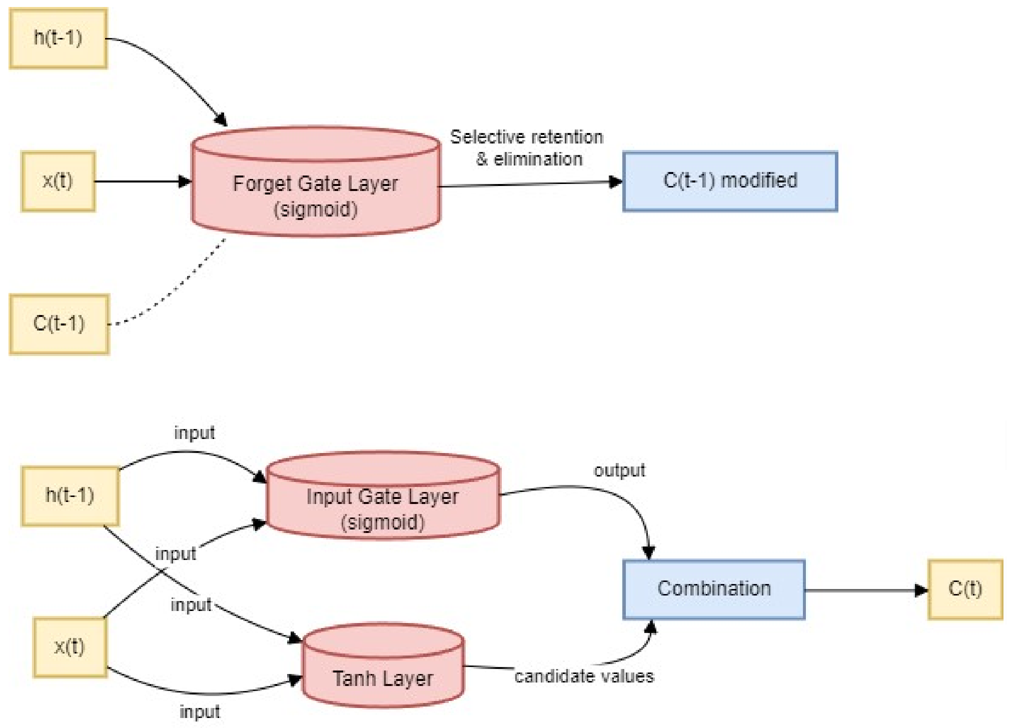

Figure 2. LSTM: The first step and the second step.

The second stage involves determining the new information to be stored in the cell state, which consists of two components. The "input gate layer", which is a sigmoid layer, initially identifies the values intended for updating. Subsequently, a Tanh layer produces a vector of new candidate values C_t, which may be incorporated into the state. The subsequent step involves merging these two components to formulate a state update. The step is shown in Figure 2 on the right [40].

The third step is to update the previous cell state, C_{t-1}, to the new cell state, C_t, as illustrated in Figure 3 on the left. The preceding steps have determined the necessary actions; now, it is time to implement them. We start by multiplying the old state by f_t, effectively forgetting the elements we had previously decided to discard. Then, we add $i_t * \tilde{C}_t$ to this product. Here, $\tilde{C}_t$ represents the new candidate values, which are scaled by the amount with which we decided to update each state value [40].

To finalize our LSTM model, determining the output is crucial; this output is essentially a selectively filtered version of the cell state. A sigmoid layer evaluates which components of the cell state are pertinent for the output, as depicted in Figure 3 on the right. Following this, the cell state is passed through a Tanh function to normalize its values, and then it is multiplied by the output of the sigmoid gate. This operation ensures that only the relevant components, as identified by the sigmoid layer, are passed to the final output.

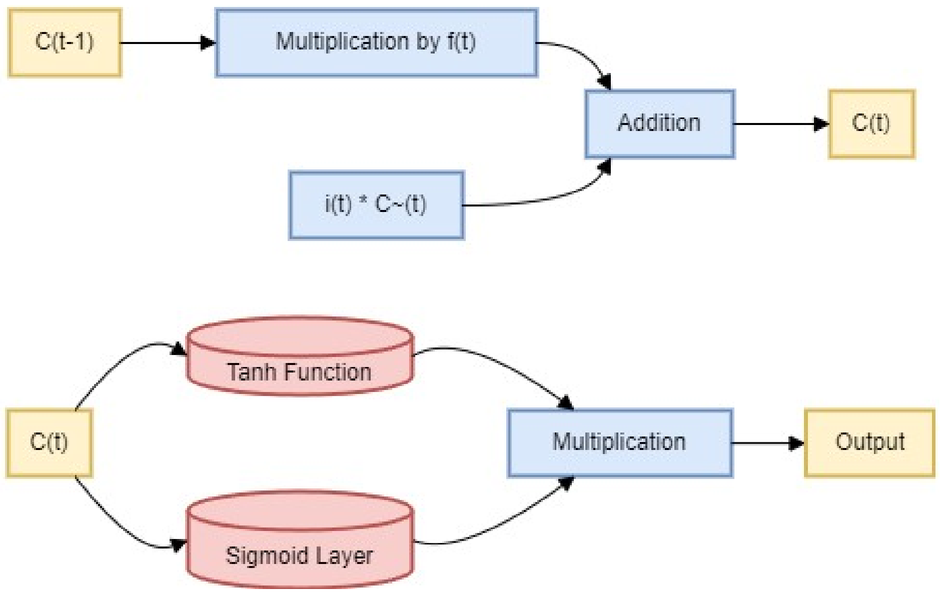

Figure 3. LSTM: The third step and the fourth step.

2.2. Word Embeddings Vectors: ELMo and Gloves

Over the past few years, there has been a significant shift in the field of NLP due to the introduction of groundbreaking linguistic models. In language processing, there is an exciting shift happening away from traditional n-gram models and toward neural networks [41]. These cutting-edge methods combine language modeling with feature learning, which lets words be represented as numerical vectors. This transformation streamlines and enhances data processing, forming the foundation for a powerful technique known as word embeddings.

Enhanced representations have been suggested to improve social media spam detection beyond simple word embeddings. These representations merge knowledge graphs and deep learning methods to classify social media posts according to established classification models [41]. Through the process of extracting entities from brief posts and connecting them to concepts in knowledge graphs, contextual knowledge is stored as vector representations. These representations, combined with contextualized word embeddings, are leveraged to create context-based representations of the posts, enhancing the dependability of predictive models. Moreover, a combined method that incorporates data augmentation, natural language processing, and supervised machine learning algorithms has been suggested for identifying Twitter spam. This method utilizes word embedding techniques and various machine learning algorithms like SVM, Naive Bayes, and logistic regression to enhance the identification of Arabic spam on the Twitter platform [41].

Word embedding models use probability distributions to assign vectors to specific words, predicting the likelihood that various tokens appear near the target word. These models are popular in the current NLP community due to their effectiveness in enabling computations using numerical embeddings instead of textual sequences. Alhassun et al. [42] presented a framework for detecting Arabic spam accounts on Twitter, employing deep learning to analyze both text-based and metadata features, achieving a notable accuracy of 94.27%. This approach outperforms existing models and highlights the importance of combining text and metadata for spam detection. However, the authors used only a specific dataset that may not generalize across different languages or dialects, potential issues with the robustness of spam detection and sarcasm recognition exist across varied contexts, and the challenges of interpreting complex language nuances like irony or local slang could mean that the models' performance could be affected by dynamic changes in social media communication patterns, requiring ongoing adaptation and re-evaluation. Sharma et al. [37] introduced a hybrid ensemble model for sarcasm detection, integrating fuzzy logic with Word2Vec, GloVe, and BERT embeddings. It reports high accuracies across different datasets, demonstrating the model's efficacy in understanding the nuanced language of sarcasm on social media. These studies collectively underscore the advancement in utilizing complex models for nuanced social media content analysis, setting a precedent for future research in similar domains. The authors did not address difficulties in interpreting acronyms, slang, or diverse linguistic contexts, highlighting the limitations of using a single NLP approach to handle the wide variety of material encountered on social media platforms. These challenges underscore the need for advanced and adaptable NLP techniques to accurately understand and classify complex language use in social media content.

Word embeddings enable the numeric representation of words in a vector space by encoding semantic meaning, which allows textual data to be processed computationally for tasks like classification [43]. Popular techniques like Word2Vec [44] and GloVe [41] generate embeddings by analyzing the linguistic contexts of words in large corpora. Word2Vec is a significant advancement in the field of NLP, combining two well-known architectures: continuous bag-of-words (CBOW) and skip-gram. These architectures have been designed to streamline the computation of word-vector representations for optimal efficiency. The use of these representations has proven to be highly valuable in a range of NLP applications, including tasks like text prediction and classification [16,19,45]. However, being context-independent, Word2Vec struggles with polysemy and falls short of capturing nuanced semantic relationships. GloVe creates embeddings by directly leveraging statistics of word co-occurrence in corpora [46]. Its core strength lies in effectively incorporating local context and global co-occurrences for meaningful representations. However, it faces limitations similar to Word2Vec when handling words with multiple contextual meanings.

Contextualized Word Embeddings BERT and ELMo

A significant development in the field of pretrained models is the Bidirectional Encoder Representations from Transformers (BERT) model, which Google created and was first presented in 2018 by Jacob Devlin and his team [47]. Throughout the pretraining phase, BERT, a language model, underwent an extensive training process. This required working with a significant amount of unlabeled data, which included approximately 800 million words from the BooksCorpus dataset and around 2500 million words from the English Wikipedia. By leveraging intensive pretraining, BERT can effectively fine-tune smaller, domain-specific datasets, resulting in improved performance for specific goals [48]. Although this approach demands substantial computational resources, the benefits are worth it. There are two different configurations that define the availability of the English-language version of BERT. BERT is a popular NLP model that comes in two configurations: $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$. $\text{BERT}_{\text{BASE}}$ has 12 encoders, each equipped with 12 bidirectional self-attention heads, while $\text{BERT}_{\text{LARGE}}$ has 24 encoders, each equipped with 16 bidirectional self-attention heads. One of the remarkable aspects of BERT is its exceptional performance as a contextual embedding model [48]. It can comprehend words within their unique contexts and assign diverse numerical representations accordingly. This understanding of BERT's context enables it to process words in a manner that depends on the context, allowing it to distinguish how words are treated based on their role within sentences.

Yiming Qiu et al. (2022) [49] provided customized pretraining tasks for user intent detection and semantic embedding retrieval in e-commerce search. The paper does not provide information about the development of the BERT model or its configurations. However, BERT-style models may not work for corpora with different text from Wikipedia, and its style of models may not work for tasks requiring specific embedding spatial distribution.

Serrano-Guerrero et al. (2024) [50] compared different BERT models for identifying destructive content in social media, but they do not specifically discuss the relationship between BERT models and social media. Their comparison of five BERT models for identifying destructive content found the most effective architecture to have 96.99% accuracy.

Recent breakthroughs in NLP have seen a shift from context-independent word vectors towards deep contextualized models like ELMo [48], ULMFiT [51], and BERT. Built on LSTM architectures, these representations model the semantic usage of words based on the surrounding context. ELMo is a pretrained language model that has revolutionized the field of NLP. It first underwent training using a sizable collection of 5.5 billion words, and then a smaller version trained on 1 billion words [51]. ELMo uses big datasets to make deeply contextualized word representations: (1) understanding the complicated parts of language use, like syntax and semantics, and (2) telling the difference between word meanings that change in different language contexts, like representing polysemy [52]. ELMo's unique feature is its ability to understand context, allowing it to create distinct representations for homonyms, i.e., words that share the same spelling but have different meanings; therefore, it offers a unique approach compared with traditional models by considering the entire input phrase, resulting in a more advanced understanding of word usage [52]. The technology behind ELMo is a bidirectional LSTM (BiLSTM), which is trained on large amounts of text using a language modeling method. BiLSTM efficiently analyzes sentences and can predict word sequences in both directions by combining forward and backward LSTM parts. This makes a two-way language model a useful tool for many NLP applications because it uses BiLSTM processing to improve language comprehension through word embeddings.

Santosh Kumar et al. (2023) [53] proposed an ML-based approach for spam detection on social media and used various features and classifiers to distinguish spam messages from legitimate ones. And, Akinyelu et al., (2021) [54] used ML for spam detection in social networks, and the proposed Voting Classifier algorithm achieved a high classification accuracy rate of 97.96%. Nevertheless, these studies concentrated on the English language, while our study delves into the impact of contextualized embeddings such as ELMo for detecting social media spam on various platforms such as Twitter and YouTube. The evaluation offers recommendations on the best neural components for diverse informal content. In addition, integrating platform-specialized LSTM models balances performance and generalization ability while accommodating stylistic variations in real-world, nonstandard text. Through a methodical analysis of cutting-edge representations and structures, our approach provides valuable insights for implementing spam filtering systems.

In a nutshell, these previous studies have proposed that the still representation performs very well in the discriminate spam study of content on social media and also in real-time view analysis compared with any other form of context. These studies just scratch the surface in trying to analyze the context in the contextual representation of social media spam. Therefore, we need to reveal the critical factors in differentiating the contextual representations from the contemporary representations, and subsequent implementation of the results can help such industries in the fulfillment of user expectations and build a secure environment through the benefits brought by contextualized study. This research goes into the depths of the foundational works and later provides support to justify user intention and also provides much-needed computational consideration.

3. Developing Models: NLP and Word Embeddings

3.1. Data Collection and Preprocessing

The two primary datasets are collected from Twitter, sourced from Kaggle, and YouTube, comprising comments from the UCI DL Repository. Figure 4 includes consolidating segmented files in the datasets, lowering dataset sizes by decreasing the number of rows, and removing extraneous columns. The data are subjected to a purification procedure that involves eliminating stop words and punctuation, as well as standardizing label types. The script is designed for a comparative analysis of various NLP models, with a focus on text classification tasks. It utilizes LSTM, BERT, and ELMo models to classify text data into categories (e.g., "Spam" vs. "Quality").

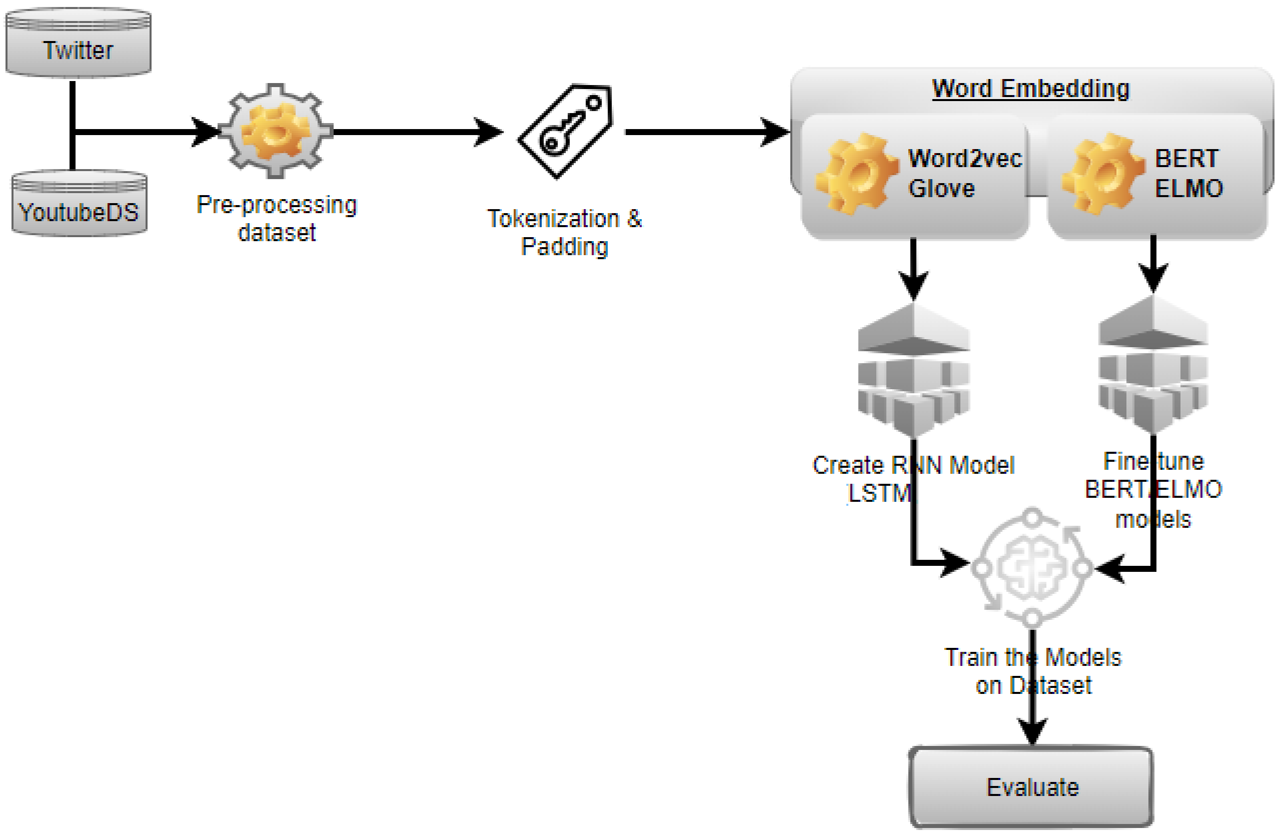

Figure 4. Research methodology.

Figure 4 shows the script encompasses data loading, preprocessing, model training, and evaluation. Table 1 shows the loaded dataset that contains various features like "Tweet", "following", "followers", "actions", "is_retweet", "location", and "Type". The "Type" column categorizes tweets into "Quality" or "Spam", which could be the target variable for our classification task. Table 2 shows the loaded dataset that contains various features like comment_id, content, date, and class.

Table 1. Sample of Twitter Dataset.

ID	Tweet	Following	Followers	Actions	Is Retweet	Location	Type
10091	It's the everything else that's complicated. #...	0.0	11,500.0	NaN	0.0	Chicago	Quality
10172	Eren sent a glare towards Mikasa then nodded a...	0.0	0.0	NaN	0.0	NaN	Quality
7012	I posted a new photo to Facebook http://fb.me/...	0.0	0.0	NaN	0.0	Scotland, UK	Quality
3697	#jan Idiot Chelsea Handler Diagnoses Trump Wit...	3319.0	611.0	294.0	0.0	Atlanta, Ga	Spam
10740	Pedophile Anthony Weiner is TERRIFIED of Getti...	4840.0	1724.0	1522.0	0.0	Blumberg	Spam

Table 2. Sample of YouTube Dataset.

Comment_ID	Author	Content	Date	Class
z13lgffb5w3ddx1ul22qy1wxspy5cpkz504	dharma pal	Nice song	2015-05-29T02:30:18.971000	1
z123dbgb0mqjfxbtz22ucjc5jvzcv3ykj	Tiza Arellano	I love song	2015-05-29T00:14:48.748000	0
z12quxxp2vutflkxv04cihggzt2azl34pms0k	Prìñçeśś Âliś Łøvê Dømíñø Mâđiś™	I love song	2015-05-28T21:00:08.607000	1
z12icv3ysqvlwth2c23eddlykyqut5z1h	Eric Gonzalez	860,000,000 lets make it first female to reach. . .	2015-05-28T20:47:12.193000	0
z133stly3kete3tly22petvwdpmghrlli	Analena López	shakira is best for worldcup	2015-05-28T17:08:29.827000	0

Tables 1 and 2 involve meticulously preparing and preprocessing two essential datasets, as referenced in the literature [30]. The initial datasets were specifically designed for analyzing spam on Twitter. The preprocessing is performed with great attention to detail to ensure the quality and reliability of the data, starting by removing unnecessary columns and fixing any missing values. Following that, we proceed with the process of standardizing the labels within the datasets, ensuring consistency across all data points to ensure balanced data are a critical part of this stage, as they play a vital role in achieving accurate and consistent results across various models. The cleaning step is involved as a part of the preparation to remove unnecessary elements like URLs, stop words, and punctuation marks, which have no impact on the model's learning phase, as shown in Algorithms 1 and 2. It involves dividing the data into two distinct sets: 80% is allocated for training purposes, while the remaining 20% is reserved for evaluating the model's performance.

3.2. Data Tokenization and Sequence Padding

The textual data undergo vital preprocessing, where words are tokenized and padded into sequences of fixed length to enable computational operations. This step comprises the process of tokenizing textual data and applying padding. At this crucial point, known as the embedding stage, words are converted into vector representations, making it easier to perform numerical operations on textual data. This study employs a dual approach to embedding by employing both context-independent approaches, like Word2Vec and GloVe, and contextualized embeddings, such as BERT and ELMo. After the embedding procedure, many models are carefully built and undergo a tough training routine. The process involves assessing the efficacy of these algorithms in identifying spam. This assessment will determine the most effective model used and will also include a comparison with approaches described in earlier research. The following sections provide a thorough analysis of each phase along with methodological insights.

We employed Python libraries in a cloud environment like Google Colab. The tensorflow_hub is used for accessing pretrained models such as ELMo, and Transformers is a library by Hugging Face that provides interfaces for working with models like BERT. The dataset is loaded into a DataFrame from a CSV file. This dataset is assumed to contain text data and corresponding labels. Text data are labeled as "Tweet", and labels are converted from categorical to binary form (0 and 1). The dataset is then split into training and testing sets, with 80% of the data used for training and 20% for testing.

3.3. Embedding the Words

In this phase of our NLP pipeline, word embeddings play a pivotal role by converting textual data into a numerical format that deep learning models can understand. An embedding layer is utilized to transform each word in the dataset into a fixed-length vector of a predetermined size. This vector represents the semantic significance of the word, capturing both its meaning and its relationship with other words in the dataset. Every word in the text is allocated a distinct numerical vector that represents its semantic significance. The embedding layer in our model design enables the transformation of individual words into fixed-length vectors of a predetermined size. This modification is crucial for the following computational analysis: examining and using a range of embedding strategies, each making a distinct contribution to the process of representing words. We explore a variety of embedding techniques to best capture the nuances of language:

- The Word2vec embedder from the gensim package is used to convert the processed text into embeddings. These embeddings are then fed into the LSTM model for classification.
- The GloVe technique is used to produce word embeddings in vector form. The embeddings are used as input features for the LSTM network. The Glove model is used to generate word vectors, which are then used as inputs for the architecture, similar to the method described above.
- The BERT model incorporates an embedding layer. The BERT model comes with its own contextual embedding layer, which was trained beforehand. This layer processes the textual data to create word embeddings based on the context.
- ELMo has an internal embedding layer that uses its biLSTM structure, which creates contextual vectors that are then used for classification. The models are then given the embedding vectors that were generated in the previous rounds, forming the foundation for their training and eventual application in spam detection.

3.4. Training the Models

The heart of DL involves training models on a subset of data, allowing them to discover data features and apply this knowledge to new, unseen data. In our specific scenario, the models have the responsibility of categorizing text into either the “spam: 0” or “quality: 1” categories.

As depicted in Figure 4, passing embedding layer representations into the architectures provides the foundation for acquiring this knowledge. Specifically, recurrent networks like LSTMs effectively model sequential dependencies in language. The input, output, and forget gates of LSTM cells allow capturing long-range contexts and nuanced relationships within textual data. On the other hand, contextualized techniques like BERT and ELMo generate dynamic token embeddings based on the surrounding context. Pretraining them on our diversified corpora (Twitter(X)/YouTube) imparts semantic encoding capabilities. Our approach loads these pretrained models via TensorFlow-Hub/Text and generates contextual word vectors to feed into the classification layers after fine-tuning, enhancing performance over static embeddings. We employ binary classification layers tailored for spam detection—logistic regression for ELISA and feedforward network for BERT. The training process involves iterating over text sequences in the Twitter/YouTube datasets and updating model parameters to minimize the loss function chosen. Tracking validation accuracy and loss metrics over epochs then allows us to quantitatively evaluate model effectiveness on unseen data.

Embedding layers, such as those derived from ELMo and BERT, are pivotal in text classification models by enabling the extraction of dynamic and context-aware word representations. These embeddings are instrumental in analyzing textual characteristics, thus informing the model’s ability to categorize text. The model weights are optimized through a process aimed at minimizing the loss function, represented as

$$\text{Loss}_{\text{model}} = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)) \tag{1}$$

where N is the number of samples, y_i is the actual label, and $\hat{y}_i$ is the predicted label by the model. This equation exemplifies the cross-entropy loss, a common choice for binary classification tasks.

Following our training phase, the effectiveness of the models is assessed on a test set, focusing on metrics such as accuracy and loss, which provide quantitative insights into the models' performance. ELMo and BERT contribute significantly to this approach by offering rich, pretrained word embeddings that enhance the model's understanding and classification of text. These embeddings are sourced from TensorFlow-Hub and TensorFlow-Text, facilitating the generation of deep contextualized word representations that improve the performance of downstream NLP tasks [55].

For ELMo, we embed a generation process for each text in the Twitter and YouTube datasets, and it is formalized as follows:

$$\text{ELMo embeddings} = \text{Function (Text)} \tag{2}$$

These embeddings serve as input features for the classification model, which undergoes training on the derived ELMo embeddings. The training regimen entails multiple iterations over the dataset (epochs), with the objective of refining model weights to reduce the loss function further. The model's post-training evaluation on the test set involves calculating both the loss and accuracy, furnishing a quantitative measure of its efficacy.

RNNs, and by extension LSTMs, offer a solution to the limitations of traditional neural networks in processing sequential data. The construction of RNN/LSTM models, facilitated by frameworks such as Keras, typically involves layers conducive to binary classification tasks. The architecture of an NN/LSTM distinguishes itself by allowing variable connections between neurons, as opposed to the static, feedforward nature of traditional neural networks. This variability is characterized by directed cycles in the network, enabling data to circulate in loops and allowing the model to capture temporal dependencies in the data effectively. The update of the LSTM cell is represented as

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4}$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{5}$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{6}$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{7}$$
$$h_t = o_t * \tanh(C_t) \tag{8}$$

where f_t, i_t, and o_t are the forget, input, and output gates, respectively, C_t and h_t are the cell state and output at time t, and x_t is the input at time t. These equations highlight the LSTM's ability to manage information flow, enhancing its capacity for learning from sequences with varying lengths and complexities.

The model is much better at binary classification tasks because it uses advanced embedding techniques like ELMo and BERT along with the complex architecture of RNN/LSTM networks. It does this by learning from different text inputs and applying what it has learned across them.

3.5. Evaluation Model

Upon the completion of the training phase, each model undergoes a comprehensive evaluation using test sets derived from the Twitter and YouTube datasets. The goal of this study is to find out which contextualized embeddings (BERT, ELMo) work better than traditional, noncontextualized embeddings (Word2Vec, GloVe) in RNN architectures.

Additionally, a comparative analysis is conducted between RNN models (LSTM) and standalone models of BERT and ELMo. To accurately evaluate the performance of these models, we utilize a variety of evaluation metrics: misunderstanding the matrix metric offers a comprehensive analysis of the model's classifications, providing valuable insights into the different types of errors that were made.

Accuracy is a crucial factor in evaluating the model's performance when it comes to predicting spam tweets or comments, encompassing both true positives and false positives. Remember, sensitivity is calculated by dividing the number of correct positive predictions by the total of correct positives and incorrect negatives, which evaluates the overall rate of correct classifications made by the model. Accuracy and completeness are especially important when assessing the models' performance in detecting spam content; therefore, the F1-score is a valuable metric that takes into account both precision and recall, providing a holistic assessment of the model's performance. Through these evaluations, the research goal is to find the best combination of embedding techniques and model architectures to efficiently and universally detect spam on social media platforms. To achieve this, we trained a Word2Vec model on the collected dataset and used it as a pretrained model, where a GloVe utilizes pretrained GloVe embeddings to represent words to compare the performance of these models using different embeddings (Word2Vec vs. GloVe) and error analyses, which investigate types of errors (e.g., false positives and false negatives), as shown in Table 3. Word2Vec has lower precision but higher recall compared with the GloVe model. The F-score, which is the harmonic mean of precision and recall, is moderately high, indicating a balance between precision and recall. The accuracy is lower than GloVe's.

Table 3. Comparison of Word2Vec, GloVe, and Combined Embeddings.

Model	Precision	Recall	F-Score	Accuracy
Word2Vec	0.689356	0.795987	0.738844	0.718881
GloVe	0.802305	0.756689	0.778830	0.785297
Word2Vec & GloVe	0.773601	0.739967	0.756410	0.761905

The probability distributions of the predicted classes are displayed in Figure 5 for both Word2Vec and GloVe embeddings. There is a clear bimodal distribution of probabilities for both classifiers, suggesting a high level of certainty in their classification decisions. This histogram probably shows the distribution of predicted probabilities for the positive class. The x-axis represents the probability that a given instance is classified as positive, and the y-axis represents the frequency of these probabilities. The histogram for each model can help you understand the confidence of the predictions. A model with many predictions clustered near 0 or 1 is often more confident, whereas a model with many predictions around 0.5 is less confident. Figure 6 contains two components: The upper panel displays the confusion matrices for classifiers that use both Word2Vec and GloVe embeddings. The lower panel, on the other hand, shows the precision–recall curves for both embedding methods. The confusion matrices showcase the number of correct and incorrect classifications for each class, while the precision–recall curves provide a holistic view of the trade-off between precision and recall at different threshold settings.

Figure 6 shows the area below these curves can offer valuable insights into the overall effectiveness of the classifiers. The confusion matrices for classifiers utilizing Word2Vec and GloVe embeddings. This matrix provides a thorough comparison of the number of accurate positive predictions, accurate negative predictions, inaccurate positive predictions, and inaccurate negative predictions between the two methods. The two images at the bottom show precision–recall curves for both embeddings, illustrating the trade-off between precision and recall at different thresholds. The confusion matrices for Word2Vec and GloVe provide a detailed comparison of the accuracy of each method, displaying the number of correct and incorrect classifications made. The precision–recall curves offer valuable insights into the balance between precision and recall across various threshold settings, and the region below these curves can also be used as a measure of the overall effectiveness of the classifier. A larger area indicates a more efficient model.

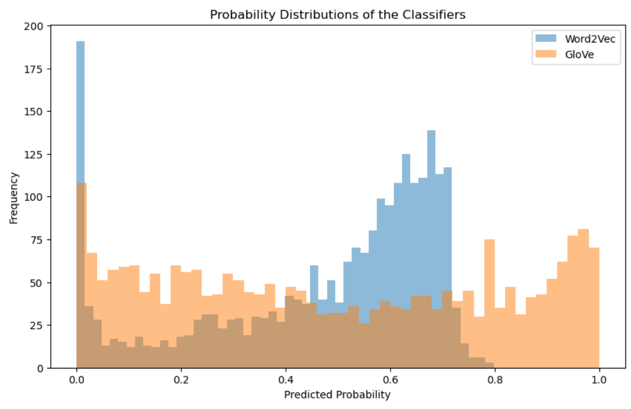

Figure 5. Probability distributions of the classifiers.

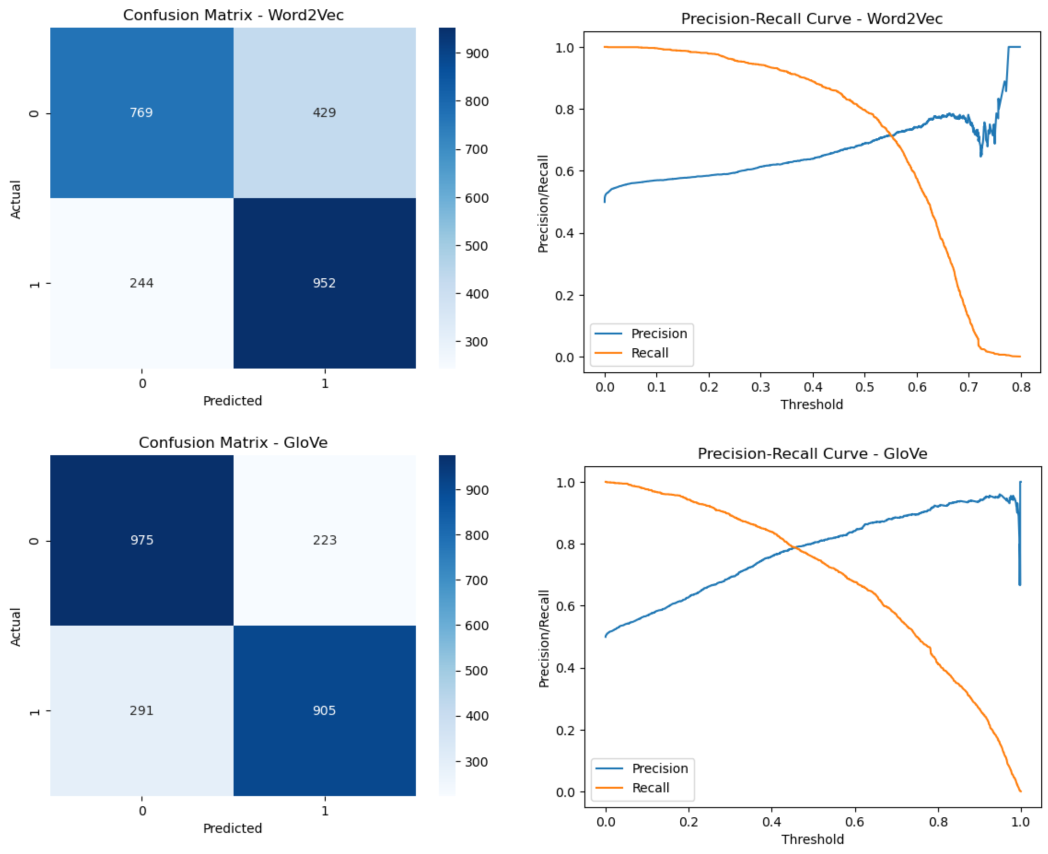

Figure 6. Word2Vec performance vs. GloVe performance.

Figure 7 shows the confusion matrix for the classifiers on the left and the performance metrics line plot on the right. They present two distributions, one for each embedding method, indicating how each method's classifier assigned probabilities to the instances.

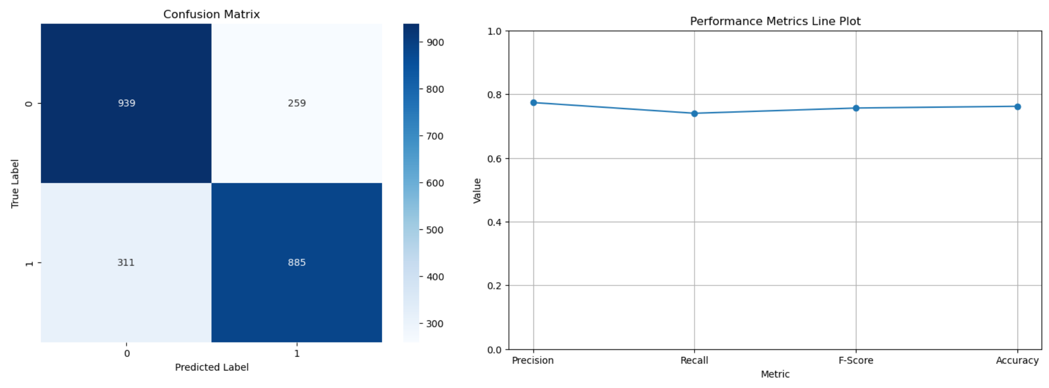

Figure 7. Performance metrics of embedding Word2Vec and Glove.

4. Results and Discussion

Extensive experiments were conducted to evaluate the effectiveness of different embedding strategies and model architectures for social media spam detection, comparing the performance of embeddings integrated into LSTM models with that of standalone textual embeddings, specifically BERT and ELMo. A variety of Python libraries, including TensorFlow, Keras, and Scikit-learn, all within the Google Colab environment, aided in the development and training of these models. The efficacy of these models was evaluated on two distinct spam detection datasets: one comprising Twitter data and the other containing YouTube comments. Both datasets feature text classified into “spam” and “quality” categories.

The Twitter and YouTube datasets were utilized to train a total of five models with different embedder–architecture combinations. Rigorous hyperparameter tuning was performed prior to the final model selection. The evaluation metrics of precision, recall, accuracy, and F1-score were computed on the test set. The tests showed that contextualized embeddings consistently performed better than word vectors. This shows that they can pick up important semantic signals and accurately display social media text. Though the RNN models with tuned hyperparameters achieved respectable performance, the ELMo contextual embedder with a logistic regression classifier emerged as the top performer. On the Twitter dataset, ELMo obtained 90% accuracy, 91% precision, 89% recall, and 90% F1-score. The accuracy of the YouTube comments reached 94%, with 98% precision, 88% recall, and 93% F1-score. This demonstrates the significance of contextual awareness and the transfer learning capacity of pretrained language models for enhanced generalization. The comparative assessment provides valuable insights into tailoring embedding strategies and model components for the unique challenges posed by multiplatform social media spam detection in real-world deployed systems.

4.1. LSTM—Word2Vec/GloVe Embeddings

Our approach employs LSTM networks alongside Word2Vec and GloVe embeddings for spam classification. The text data are initially preprocessed via tokenization and padding to generate fixed-length input sequences. For instance, the Word2Vec embedder, sourced from the Gensim library, was used to transform the processed text into embeddings, which were then inputted into the LSTM model for spam classification.

Algorithm 1 presents the overall process for training LSTM models with Word2Vec and GloVe embeddings on the Twitter and YouTube datasets for spam classification. After the dataset is loaded and preprocessed by converting text to lowercase, removing URLs/mentions/hashtags/numbers, and trimming extra whitespace, the data is then converted to binary labels for spam/nonspam and balanced to ensure equal representation of both classes. The dataset is split into training and testing sets. For Word2Vec/GloVe, an embedding model is trained only on the training data to learn vector representations of words. The maximum text sequence length is determined to enable padding sequences to a fixed length. Helper functions are defined to convert text to corresponding index sequences based on the trained embedding model and pad them to the max length. This numerical conversion of text enables feeding as input to the LSTM models. An embedding matrix is constructed from the learned vectors. The LSTM architecture is defined using this matrix to represent text input. The model is compiled, trained on the training set, and finally evaluated on the testing set. Learning distributed embeddings through predicting words in context encodes meanings based on similarity. GloVe enhances this by using log-bilinear regression modeling on word–context co-occurrence statistics. Through the pretrained vectors capture detailed contextual meanings, enabling the downstream LSTM models to process text sequences as real-valued vectors instead of sparse one-hot encoded inputs. This incorporation of modern word embedding techniques like Word2Vec and GloVe enriches the textual feature representation fed into the LSTM models.

Algorithm 1 Classification with LSTM and Word2Vec/GloVe Embeddings

```
1: $D \leftarrow$ Load ('twitter/youtube.csv')        ▷ Load dataset from Twitter or YouTube
2: function PREPROCESSTEXT ($text$)
3:     $text \leftarrow$ toLowercase ($text$)
4:     $text \leftarrow$ removeURLsMentionsHashtagsNumbers ($text$)
5:     $text \leftarrow$ replaceNonWordCharsWithSpace ($text$)
6:     $text \leftarrow$ trimSpaces ($text$)
7:     return $text$
8: end function                                          ▷ Preprocess text data
9: $D_{\text{processed}} \leftarrow$ ApplyPreprocessText ($D$)
10: $D_{\text{binary}} \leftarrow$ ConvertToBinary ($D_{\text{processed}}$)
11: $D_{\text{balanced}} \leftarrow$ Balance ($D_{\text{binary}}$)
12: ($D_{\text{train}}, D_{\text{test}}$) $\leftarrow$ TrainTestSplit ($D_{\text{balanced}}$)
13: model$_{\text{W2V/GloVe}}$ $\leftarrow$ TrainEmbedding ($D_{\text{train}}$) ▷ Train Word2Vec or GloVe model on the
    training set
14: max_length $\leftarrow$ DetermineMaxLength ($D_{\text{train}}$)
15: function CONVERTTOINDICES ($text$, model$_{\text{W2V/GloVe}}$, $max\_length$)
16:     $indices \leftarrow$ ConvertWordsToIndices ($text$, model$_{\text{W2V/GloVe}}$)
17:     $padded \leftarrow$ PadSequence ($indices, max\_length$)
18:     return $padded$
19: end function                         ▷ Convert text to indices and pad sequences
20: $D_{\text{indices}} \leftarrow$ ConvertAllToIndices ($D_{\text{train}}$, model$_{\text{W2V/GloVe}}$, max_length)
21: $E \leftarrow$ CreateEmbeddingMatrix (model$_{\text{W2V/GloVe}}$) ▷ Create embedding matrix from the
    trained model
22: model$_{\text{LSTM}}$ $\leftarrow$ DefineLSTMModel ($E$) ▷ Define the LSTM model using the embedding
    matrix
23: Compile (model$_{\text{LSTM}}$)
24: Train (model$_{\text{LSTM}}$, $D_{\text{train}}$)                   ▷ Compile and train the LSTM model
25: $predictions \leftarrow$ Predict (model$_{\text{LSTM}}$, $D_{\text{test}}$)
26: Evaluate (model$_{\text{LSTM}}$, $predictions$)   ▷ Predict and evaluate the model on the test set
```

Word2Vec embedding falls short in capturing semantic nuances: Algorithm 1 provides the configuration details for the Word2Vec-LSTM architecture, with corresponding accuracy results in Table 4. This model achieves only 67% and 72% accuracy on the Twitter and YouTube datasets, respectively. As evident in Figure 8, the ROC and precision–recall curves showcase the model's limited discrimination capability. This significant deviation from an optimal classifier (top-left corner) indicates the prevalence of false positives due to the inability to capture nuanced semantics.

Table 4. Accuracy of Different Models on Twitter and YouTube Datasets.

Model	Twitter Dataset Accuracy	YouTube Dataset Accuracy
Word2Vec-LSTM	**67%**	72%
GloVe-LSTM	**80%**	87%

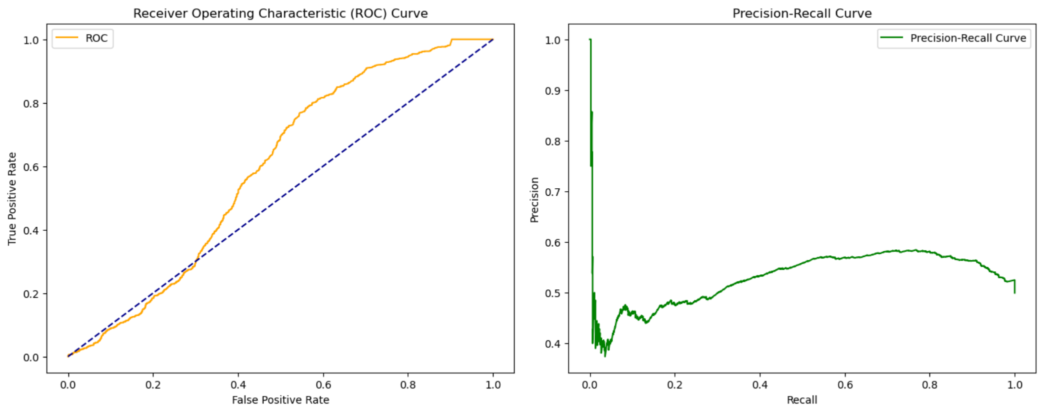

Figure 8. Training and validation accuracy to Word2Vec-LSTM.

We posit that Word2Vec embeddings, despite their simplicity, are inadequate for encoding the contextual variations in language necessary for accurate spam detection. The vectors are created from a relatively small corpus and conflate multiple meanings into singular representations. This results in poor generalizations based on informal textual data with ambiguities and polysemy. More robust semantic modeling is imperative through large-scale pretraining, as offered by methods like GLoVe and ELMo. Our comparative analysis quantitatively demonstrates that Word2Vec lags behind context-aware techniques better equipped to handle the complex features of evolving spam. Significant performance gains in our experiments on two different social media landscapes show how important more advanced embeddings are.

To employ LSTM networks alongside GloVe embeddings for spam classification, as depicted in Figure 9. The text data are initially preprocessed via tokenization and padding to generate fixed-length input sequences. These sequences are then transformed into dense vector representations using 300-dimensional GloVe word vectors pretrained on a large textual corpus. The resulting embeddings capture semantic and contextual information for each token. These embeddings are subsequently fed as inputs into an LSTM model comprised of dropout regularization and sigmoid output units for binary spam/nonspam predictions. As evidenced in Figure 9, the model achieves strong performance, with 79.74% validation accuracy on the YouTube dataset after five training epochs. The convergence of training and validation loss curves indicates minimal overfitting.

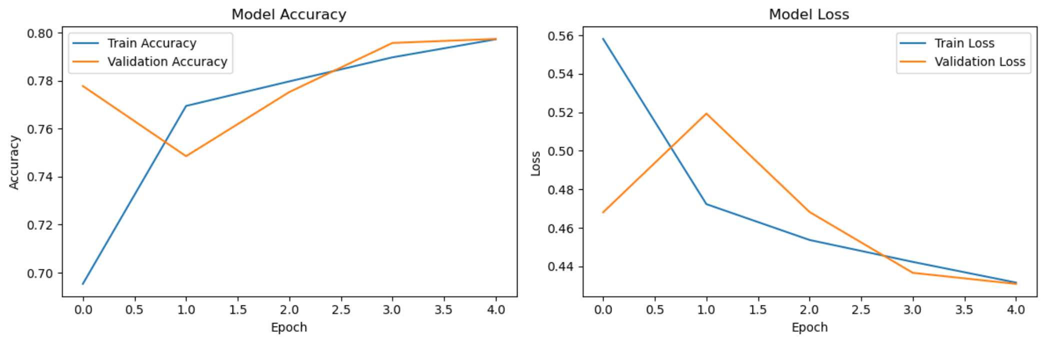

Figure 9. Training and validation accuracy to GloVe-LSTM.

The model begins with a training loss of 0.5581 and an accuracy of 69.53%. On the validation set, it achieves a loss of 0.4681 and an accuracy of 77.78%. The final epoch shows training loss at 0.4316 and accuracy at 79.73%, with validation loss and accuracy improving to 0.4309 and 79.74%, respectively. The close performance on both training and validation sets suggests that the model is convergent well and learning from the training data without significantly overfitting. In Figure 9, the blue line (training accuracy) and the orange line (Validation accuracy) show the accuracy of the model for each epoch on the training and validation datasets, respectively. There is a noticeable dip in validation accuracy in the second epoch, which could be indicative of some instability in the training process at that point. However, from the third epoch onwards, both training and validation accuracy improved consistently. The final epoch shows the training and validation accuracy converging, which is a sign of good generalization.

The blue line (training loss) and the orange line (validation loss) indicate the loss on the training and validation datasets across epochs. Similar to the accuracy plot, there is an increase in the validation loss during the second epoch. However, in subsequent epochs, both training and validation losses decrease, indicating improvement in the model's performance. The convergence of the training and validation losses by the fifth epoch suggests that the model is not overfitting significantly.

The precision for classification is 76% of the predicted comments. The recall is higher, at 87%, which means that the model can capture 87% of the actual data. The F1-score, which is a harmonic mean of precision and recall, is 81% for comments, indicating a balance between precision and recall. For spam, precision is higher at 0.85, suggesting that when the model predicts spam, it is correct 85% of the time. The recall for spam is lower at 0.72, meaning the model misses around 28% of actual spam tweets. The F1-score for spam is 78%, slightly lower than for nonspam, due to a lower recall. Overall, the accuracy model correctly identified 80% of the tweets and 87% of the comments, as shown in Table 4. In essence, the role of GloVe in encoding semantics explained the convergence plots, interpreted the precision/recall trade-off, and contrasted against Word2Vec—all in service of enhancing our approach design.

4.2. BERT and EMLO Embedding

Our study provides novel experimentation and analysis to contrast standalone BERT and ELMo models for social media spam detection. We employ state-of-the-art transfer learning approaches by leveraging the innate contextual modeling capabilities of these pretrained language representations. Specifically, we instantiate BERTBASE cased and ELMo original 5.5B models for generating token embeddings on the Twitter and YouTube datasets. While both architectures use LSTM networks, a key distinction lies in ELMo's deeper bidirectional conditioning compared with BERT's single direction. We hypothesize this enables richer semantic encoding vital for handling informal textual styles. As shown in Table 5, ELISA consistently attains a higher accuracy of 90–94% against BERT's 84–91%. We posit the performance gap arises from ELISA's intrinsic focus on modeling linguistic contexts across two directions. This allows capturing nuanced syntactic and semantic patterns within noisy social posts to improve discrimination. Additionally, we employ an innovative pipeline (Algorithm 2) involving pairing pretrained embedders like BERT and ELISA with simple downstream classifiers (logistic regression). This end-to-end approach eliminates extensive task-specific fine-tuning and provides strong off-the-shelf performance, uniquely suited for real-world spam filtering systems that need to rapidly adapt to evolving data.

Table 5. Validation Accuracy for EMLO and BERT Models.

Model	**Twitter Dataset**		**YouTube Dataset**	
	EMLO	**BERT**	**EMLO**	**BERT**
f1-score	90%	83%	93%	91%
Precision	91%	87%	98%	94%
Recall	89%	79%	88%	87%
Accuracy	90%	84%	94%	91%

Algorithm 2 leverages pretrained contextualized embedding models like BERT and ELMo to generate representations of text for training classifiers to categorize posts as spam or not spam. The input data from two separate datasets (e.g., Twitter and YouTube) are first preprocessed, converted to lowercase, and cleaned by removing URLs and special characters. Helper functions are set up to make embeddings from BERT and ELMo models. For BERT, the text is tokenized using a predefined tokenizer and then fed into the model to get token embeddings that match. For ELMo, the raw text strings are directly input into the model to produce embeddings. These pretrained models encode rich semantic representations of text by considering contextual information. The embeddings are then used as features for training machine learning classifiers on the downstream spam detection task. Then, the data are split into training and test sets. Classifiers like logistic regression are trained on the embedding inputs from the training set, so we can apply model performance that is evaluated on the test set using metrics (accuracy, precision, and recall.) This provides insights into how these deep contextual models perform for spam detection across different platforms, like Twitter and YouTube, and how the pretrained representations augment sequential networks to handle noisy, informal language better.

Algorithm 2 BERT and ELMo Embedding Processes for Text Classification

```
1:  function PREPROCESS ($\mathcal{T}$)
2:      $\mathcal{T}_{\text{lower}} \leftarrow$ lowercase ($\mathcal{T}$)                                ▷ Convert text to lowercase
3:      $\mathcal{T}_{\text{clean}} \leftarrow$ remove ($\mathcal{T}_{\text{lower}}$, URLs, special chars)        ▷ Remove URLs and special characters
4:      return $\mathcal{T}_{\text{clean}}$
5:  end function                                                      ▷ Preprocess text data
6:  $\mathcal{D}_1, \mathcal{D}_2 \leftarrow$ Load datasets from specified paths          ▷ Load two separate datasets for processing
7:  $\mathcal{D}_1^{\text{clean}} \leftarrow$ map (Preprocess, $\mathcal{D}_1$)
8:  $\mathcal{D}_2^{\text{clean}} \leftarrow$ map (Preprocess, $\mathcal{D}_2$)                 ▷ Apply preprocessing to both datasets
9:  function GENERATEBERTEMBEDDINGS ($\mathcal{S}$)
10:     $\mathcal{T}_{\text{tokenized}} \leftarrow$ Tokenize ($\mathcal{S}$, BERT Tokenizer) ▷ Tokenize text using BERT's tokenizer
11:     $\mathcal{E}_{\text{BERT}} \leftarrow$ BERT Model ($\mathcal{T}_{\text{tokenized}}$)                ▷ Generate embeddings using BERT
12:     return $\mathcal{E}_{\text{BERT}}$
13: end function                                      ▷ Function to generate BERT embeddings
14: function GENERATEELMOEMBEDDINGS ($\mathcal{S}$)
15:     $\mathcal{E}_{\text{ELMo}} \leftarrow$ ELMo Model ($\mathcal{S}$)                        ▷ Generate embeddings using ELMo
16:     return $\mathcal{E}_{\text{ELMo}}$
17: end function                                      ▷ Function to generate ELMo embeddings
18: $\mathcal{E}_1 \leftarrow$ GenerateBERTEmbeddings ($\mathcal{D}_1^{\text{clean}}$)
19: $\mathcal{E}_2 \leftarrow$ GenerateELMoEmbeddings ($\mathcal{D}_2^{\text{clean}}$)   ▷ Generate embeddings for each cleaned dataset
20: Split datasets into training and testing sets            ▷ Divide data for model evaluation
21: Train classifiers on respective embeddings                 ▷ Use embeddings to train text classification models
22: Predict and evaluate on test sets             ▷ Assess model performance on unseen data
23: Print classification reports and accuracies        ▷ Display model evaluation results
```

The comparative assessment reveals consistent gains achieved by ELMo over BERT for spam detection across both Twitter and YouTube datasets, as evidenced visually in Figures 10 and 11. The Receiver Operating Characteristic (ROC) plots showcase ELISA's stronger discrimination capability, with Area Under Curve (AUC) values up to 0.96 for YouTube, significantly higher than BERT's 0.88. Additionally, the precision–recall curves demonstrate ELMo's superior trade-off by achieving over 90% precision at 80% recall levels on both datasets. In contrast, BERT sees an acute decline in precision as recall increases, indicating higher false positive rates. We posit ELMo's bidirectional conditioning enables the modeling of nuanced contextual cues within noisy textual data, leading to more reliable predictions. Quantitatively, as compiled in Table 5, standalone ELMo surpasses BERT on all evaluation metrics, with relative gains of 5–15% in accuracy, precision, recall, and F1-scores. This aligns with the visual evaluation. The big performance gaps show that ELMo is good at encoding complex semantic features, which is important for keeping up with spam campaigns that are always changing. Therefore, our extensive comparative analysis based on multiple evaluation paradigms provides conclusive proof of ELISA's strengths in combating digital threats amidst rapidly expanding OSNs. These insights can help inform architectural optimizations for advancing NLP techniques against adversarial attacks.

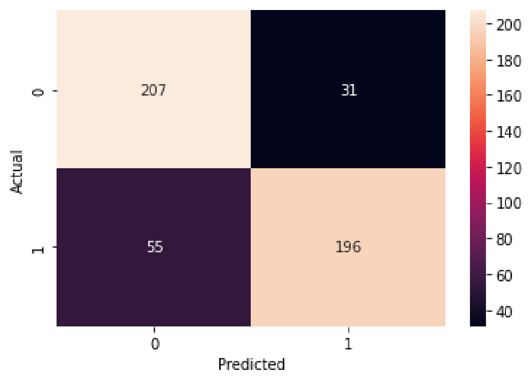

Figure 10. The performance of a BERT classification.

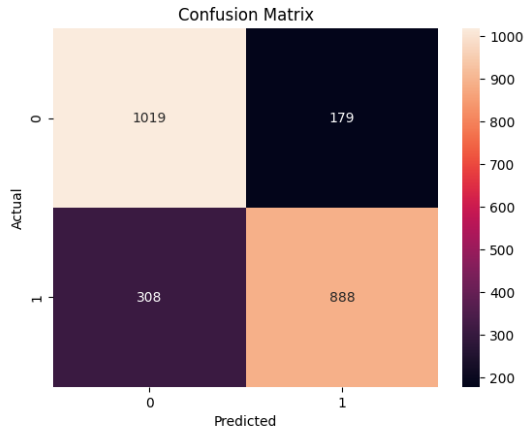

Figure 11. The performance of an ELMo classification.

Our systematic evaluation conclusively demonstrates the effectiveness of deep contextualized embeddings like ELMo over noncontextualized alternatives for spam detection. As shown in Table 5, the pretrained 1024D ELMo representations attain remarkable accuracy between 90–94% on Twitter and YouTube data, constituting absolute gains of 7–15% over prior best methods relying on 300D GloVe vectors. We posit that ELMo's deeper bidirectional LSTM conditioning enables the encoding of higher-order sequential dependencies within comments, allowing the capture of nuanced semantic relationships amidst noisy textual data. This contextual modeling proved critical for discriminating evolving spam posts full of obfuscations. Moreover, pairing these 1024D embeddings with shallow downstream classifiers like single-layer logistic regression eliminates extensive retrofitting for the specialized task, providing adaptable and generalizable solutions—a pivotal requirement for real-world spam filtering systems that need to rapidly adapt to new data. Interestingly, the longer, conversational nature of YouTube comments better resembles training corpora like the Billion Word Benchmark, leading to superior performance over Twitter's constraint-driven syntax that is full of misspellings and abbreviations. This highlights the need for pretraining on diversified text spanning diverse linguistic styles to attain consistent robustness. Nevertheless, our 7–15% across-the-board improvements over a strong 300D baseline clearly validate the advanced contextual encoding methods that must now foreground spam detection research to combat security threats arising from exponential OSN growth.

Our results directly address the research questions posed at the outset of this study, providing clear insights into the performance of contextualized embeddings versus noncontextualized word vectors, the comparison between RNN models and standalone BERT and ELMo models, and identifying the most effective combination of embedding techniques and model architectures for spam detection.

- **RQ1**: The results indicate that contextualized embeddings (BERT, ELMo) consistently outperform noncontextualized word vectors (Word2Vec, GloVe) in detecting spam content across social media platforms. Higher accuracy, precision, recall, and F1-scores obtained by models using BERT and ELMo embeddings as compared with those using Word2Vec and GloVe serve as proof of this. The superior performance of contextualized embeddings can be attributed to their ability to capture semantic nuances and context-dependent meanings of words, which are crucial for accurately classifying spam and nonspam content.
- **RQ2**: The evaluation results indicate that contextualized embeddings (BERT, ELMo) consistently outperform noncontextualized word vectors (Word2Vec, GloVe) in detecting spam content across social media platforms. This is evidenced by higher accuracy, precision, recall, and F1-scores achieved by models utilizing BERT and ELMo embeddings compared with those employing Word2Vec and GloVe.
- **RQ3**: RNN models, particularly those utilizing LSTM architectures, achieve respectable performance; standalone BERT and ELMo models exhibit higher accuracy in spam detection tasks. The comparison shows that BERT and ELMo models are much better at finding and labeling spam content than LSTM models alone. This is because they have more advanced pretraining and a better understanding of context.

5. Conclusions and Future Work

Systematically, this research evaluated the effectiveness of contextualized word embeddings (BERT, ELMo) against noncontextualized alternatives (Word2Vec, GloVe) for social media spam detection, leveraging the capabilities of RNN architectures, particularly LSTM networks. Our analysis demonstrates the superior performance of contextualized models in processing Twitter and YouTube texts. Specifically, BERT and ELMo consistently achieved considerably higher accuracy, precision, and recall compared with Word2Vec and GloVe. This can be attributed to their ability to generate distinct representations of the same word based on its semantic context. The ELMo embedder paired with logistic regression attained the top classification results across metrics, even on imbalanced datasets. This underscores the importance of contextual awareness in modeling the nuances of heterogeneous social media content. The findings pave the way for exploring innovative directions like "deep forest" approaches that leverage the transfer learning capacities of contextualized language models such as BERT and ELMo. Applying these techniques beyond English to languages like Arabic can accelerate multilingual social spam research, and the rigorous error analysis provided insights into why certain embedding–model combinations underperform, guiding efforts to enhance generalization. We demonstrate the profound impact of contextualized representations on recurrent neural network performance for real-world spam detection across Twitter and YouTube. The analysis offers a robust framework for evaluating state-of-the-art NLP techniques to identify optimal deployment strategies tailored to diverse platforms. These findings highlight promising pathways for developing versatile, scalable, and accurate systems to secure burgeoning social media ecosystems against continuously evolving threats, in addition to emphasizing the superior performance of contextualized embeddings, discussing potential research directions, and highlighting the practical implications for real-world social media spam filtering systems.

Author Contributions: S.A. contributed to the conceptualization and methodology of the research. A.M.R.A. and A.S. led the software development, validation, and formal analysis of the research. S.A. and A.S. were responsible for the investigation, resource gathering, and data curation. A.M.R.A. and A.A.M. prepared the original draft of the manuscript and also contributed to the writing, review, and editing of the manuscript. S.A. and A.S. were in charge of supervision and project administration. A.M.R.A. and A.A.M. were involved in the validation process alongside. S.A. and A.S. contributed significantly to the visualization of the research findings. In addition to this, A.M.R.A. and A.A.M. were responsible for the funding and proofreading. 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: The data supporting the findings of this study are available from public sources on the social media platform Kaggle.

Acknowledgments: Our research is a testament to the collaborative efforts of several esteemed institutions whose contributions have been fundamental to the success of this study. We extend gratitude to the Southern Illinois University Carbondale (SIUC), Yarmouk University, and Prince Sultan University. Each institution has provided a wealth of resources, academic expertise, and a collaborative spirit that has been indispensable in our pursuit of knowledge and the successful completion of this paper. Their joint commitment to research excellence has not only propelled this project but also reinforced the value of academic cooperation.

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

References

1. Bassiouni, M.; Ali, M.; El-Dahshan, E. Ham and spam e-mails classification using machine learning techniques. *J. Appl. Secur. Res.* **2018**, *13*, 315–331. [CrossRef]
2. Shahzad, K.; Khan, S.A.; Iqbal, A.; Shabbir, O.; Latif, M. Determinants of fake news diffusion on social media: A systematic literature review. *Glob. Knowl. Mem. Commun.* **2023**, *ahead-of-print*. [CrossRef]
3. Barushka, A.; Hájek, P. Spam filtering in social networks using regularized deep neural networks with ensemble learning. In Proceedings of the Artificial Intelligence Applications and Innovations: 14th IFIP WG 12.5 International Conference, AIAI 2018, Rhodes, Greece, 25–27 May 2018; Proceedings 14; Springer: Berlin/Heidelberg, Germany, 2018; pp. 38–49.
4. Wu, T.; Wen, S.; Xiang, Y.; Zhou, W. Twitter spam detection: Survey of new approaches and comparative study. *Comput. Secur.* **2018**, *76*, 265–284. [CrossRef]
5. Radwan, A.; Amarneh, M.; Alawneh, H.; Ashqar, H.I.; AlSobeh, A.; Magableh, A.A.A.R. Predictive Analytics in Mental Health Leveraging LLM Embeddings and Machine Learning Models for Social Media Analysis. *Int. J. Web Serv. Res. (IJWSR)* **2024**, *21*, 1–22. [CrossRef]
6. Spam Detection on Social Media Platform. *Int. J. Innov. Res. Adv. Eng.* **2023**. [CrossRef]
7. AlSobeh, A.M.R.; AlAzzam, I.; Shatnawi, A.M.J.; Khasawneh, I. Cybersecurity awareness factors among adolescents in Jordan: Mediation effect of cyber scale and personal factors. *Online J. Commun. Media Technol.* **2023**, *13*, e202312. [CrossRef]
8. Lai, K.; Long, Y.; Wu, B.; Li, Y.; Wang, B. Semorph: A Morphology Semantic Enhanced Pre-trained Model for Chinese Spam Text Detection. In Proceedings of the CIKM'22: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, 17–21 October 2022; pp. 1003–1013. [CrossRef]
9. Enhancing Spam Message Classification and Detection Using Transformer-Based Embedding and Ensemble Learning. *Sensors* **2023**, *23*, 3861. [CrossRef] [PubMed]
10. Li, Y.; Wang, H.; Liu, J. Can CNNs Construct Highly Accurate Model Efficiently with Limited Training Samples. *arXiv* **2017**, arXiv:1712.01639.

11. Alsobeh, A.; Aloudat, A. The Repercussions of the COVID-19 Pandemic on Higher Education and its implications for Syrian Refugees Students (An Analytical Descriptive Study). *Dirasat. Hum. Soc. Sci.* **2022**, *49*, 150–166. [CrossRef]
12. Dada, E.G.; Bassi, J.S.; Chiroma, H.; Abdulhamid, S.M.; Adetunmbi, A.O.; Ajibuwa, O.E. Machine learning for email spam filtering: Review, approaches and open research problems. *Heliyon* **2019**, *5*, e01802. [CrossRef]
13. Alshattnawi, S. Evaluation of Deep Learning and Machine Learning Algorithms in Intrusion Detection Systems. *J. Theor. Appl. Inf. Technol.* **2023**, *101*.
14. Alsobeh, A.; Woodward, B. AI as a Partner in Learning: A Novel Student-in-the-Loop Framework for Enhanced Student Engagement and Outcomes in Higher Education. In Proceedings of the 24th Annual Conference on Information Technology Education, Marietta, GA, USA, 11–14 October 2023; pp. 171–172.
15. Alshattnawi, S.; Al-Marie, M. Spider monkey optimization algorithm for load balancing in cloud computing environments. *Int. Arab J. Inf. Technol.* **2021**, *18*, 730–738. [CrossRef]
16. Xiao, L.; Wang, G.; Zuo, Y. Research on patent text classification based on word2vec and LSTM. In Proceedings of the 2018 11th International Symposium on Computational Intelligence and Design (ISCID), IEEE, Hangzhou, China, 8–9 December 2018; Volume 1, pp. 71–74.
17. AlSobeh, A.M.; AlShattnawi, S.; Jarrah, A.; Hammad, M.M. Weavesim: A scalable and reusable cloud simulation framework leveraging aspect-oriented programming. *Jordanian J. Comput. Inf. Technol.* **2020**, *6*, 1. [CrossRef]
18. Wang, Y.; Hou, Y.; Che, W.; Liu, T. From static to dynamic word representations: A survey. *Int. J. Mach. Learn. Cybern.* **2020**, *11*, 1611–1630. [CrossRef]
19. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
20. Brennan, P.M.; Loan, J.J.; Watson, N.; Bhatt, P.M.; Bodkin, P.A. Pre-operative obesity does not predict poorer symptom control and quality of life after lumbar disc surgery. *Br. J. Neurosurg.* **2017**, *31*, 682–687. [CrossRef] [PubMed]
21. da Silva, J.R.M.F. Robust Handling of Out-of-Vocabulary Words in Deep Language Processing. 2014. Available online: https://repositorio.ul.pt/bitstream/10451/11956/1/ulsd068787_td_Joao_Silva.pdf (accessed on 1 December 2023).
22. Saidani, N. A Learning Approach for Spam Detection Using Semantic Representation. 2021. Available online: https://di.uqo.ca/id/eprint/1311/1/Saidani_Nadjate_2021_these.pdf (accessed on 1 December 2023).
23. Alsobeh, A.; Shatnawi, A. Integrating Data-Driven Security, Model Checking, and Self-adaptation for IoT Systems Using BIP Components: A Conceptual Proposal Model. In Proceedings of the International Conference on Advances in Computing Research, Orlando, FL, USA, 8–10 May 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 533–549.
24. Motitswane, O.G. Machine learning and deep learning techniques for natural language processing with application to audio recordings. *Natl. Sci. Rev.* **2023**, *5*, 24–26.
25. Jain, G.; Sharma, M.; Agarwal, B. Spam detection on social media using semantic convolutional neural network. *Int. J. Knowl. Discov. Bioinform. (IJKDB)* **2018**, *8*, 12–26. [CrossRef]
26. Sedik, A.; Marey, M.; Mostafa, H. An Adaptive Fatigue Detection System Based on 3D CNNs and Ensemble Models. *Symmetry* **2023**, *15*, 1274. [CrossRef]
27. Jain, G.; Sharma, M.; Agarwal, B. Spam detection in social media using convolutional and long short term memory neural network. *Ann. Math. Artif. Intell.* **2019**, *85*, 21–44. [CrossRef]
28. Alom, Z.; Carminati, B.; Ferrari, E. A deep learning model for Twitter spam detection. *Online Soc. Netw. Media* **2020**, *18*, 100079. [CrossRef]
29. Al Saleh, R.; Driss, M.; Almomani, I. CBiLSTM: A hybrid deep learning model for efficient reputation assessment of cloud services. *IEEE Access* **2022**, *10*, 35321–35335. [CrossRef]
30. Elakkiya, E.; Selvakumar, S.; Leela Velusamy, R. TextSpamDetector: Textual content based deep learning framework for social spam detection using conjoint attention mechanism. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 9287–9302. [CrossRef]
31. Sun, N.; Lin, G.; Qiu, J.; Rimba, P. Near real-time twitter spam detection with machine learning techniques. *Int. J. Comput. Appl.* **2022**, *44*, 338–348. [CrossRef]
32. Fitriyah, Z.; Kartikasari, M.D. Text Classification of Twitter Opinion Related to Permendikbud 30/2021 Using Bidirectional LSTM. *BAREKENG J. Ilmu Mat. Dan Terap.* **2023**, *17*, 1113–1122. [CrossRef]
33. Li, P.; Liu, Y.; Hu, Y.; Zhang, Y.; Hu, X.; Yu, K. A drift-sensitive distributed LSTM method for short text stream classification. *IEEE Trans. Big Data* **2022**, *9*, 341–357. [CrossRef]
34. Vanam, H.; Raj, J.R. CNN-OLSTM: Convolutional Neural Network with Optimized Long Short-Term Memory Model for Twitter based Sentiment Analysis. *IETE J. Res.* **2023**, 1–12. [CrossRef]
35. Wadud, M.A.H.; Kabir, M.M.; Mridha, M.; Ali, M.A.; Hamid, M.A.; Monowar, M.M. How can we manage offensive text in social media-a text classification approach using LSTM-BOOST. *Int. J. Inf. Manag. Data Insights* **2022**, *2*, 100095. [CrossRef]
36. Khan, M.; Wang, H.; Riaz, A.; Elfatyany, A.; Karim, S. Bidirectional LSTM-RNN-based hybrid deep learning frameworks for univariate time series classification. *J. Supercomput.* **2021**, *77*, 7021–7045. [CrossRef]
37. Sharma, D.K.; Singh, B.; Agarwal, S.; Pachauri, N.; Alhussan, A.A.; Abdallah, H.A. Sarcasm Detection over Social Media Platforms Using Hybrid Ensemble Model with Fuzzy Logic. *Electronics* **2023**, *12*, 937. [CrossRef]
38. Tashtoush, Y.M.; Darweesh, D.A.; Husari, G.; Darwish, O.A.; Darwish, Y.; Issa, L.B.; Ashqar, H.I. Agile approaches for cybersecurity systems, IoT and intelligent transportation. *IEEE Access* **2021**, *10*, 1360–1375. [CrossRef]

39. Al-Eidi, S.; Darwish, O.; Chen, Y. Covert timing channel analysis either as cyber attacks or confidential applications. *Sensors* **2020**, *20*, 2417. [CrossRef] [PubMed]
40. Do, D.T.; Lee, J.; Nguyen-Xuan, H. Fast evaluation of crack growth path using time series forecasting. *Eng. Fract. Mech.* **2019**, *218*, 106567. [CrossRef]
41. Ibrahim, M.; Gauch, S.; Gerth, T.; Cox, B. WOVe: Incorporating word order in GloVe word embeddings. *arXiv* **2021**, arXiv:2105.08597.
42. Alhassun, A.S.; Rassam, M.A. A Combined Text-Based and Metadata-Based Deep-Learning Framework for the Detection of Spam Accounts on the Social Media Platform Twitter. *Processes* **2022**, *10*, 439. [CrossRef]
43. Li, Y.; Yang, T. Word embedding for understanding natural language: A survey. *Guide Big Data Appl.* **2018**, 83–104. [CrossRef]
44. Church, K.W. Word2Vec. *Nat. Lang. Eng.* **2017**, *23*, 155–162. [CrossRef]
45. 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*, 3111–3119.
46. 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; pp. 1532–1543.
47. Nissa, N.K.; Yuliant, E. Multi-label text classification of Indonesian customer reviews using bidirectional encoder representations from transformers language model. *Int. J. Power Electron. Drive Syst.* **2023**, *13*, 5641–5652. [CrossRef]
48. Ethayarajh, K. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. *arXiv* **2019**, arXiv:1909.00512.
49. Qiu, Y.; Zhao, C.; Zhang, H.; Zhuo, J.; Li, T.; Zhang, X.; Wang, S.; Xu, S.; Long, B.; Yang, W.Y. Pre-training Tasks for User Intent Detection and Embedding Retrieval in E-commerce Search. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, 17–21 October 2022; pp. 4424–4428. Authored: confirmed
50. Serrano-Guerrero, J.; Alshouha, B.; Bani-Doumi, M.; Chiclana, F.; Romero, F.P.; Olivas, J.A. Combining machine learning algorithms for personality trait prediction. *Egypt. Inform. J.* **2024**, *25*, 100439. [CrossRef]
51. Shleifer, S. Low resource text classification with ulmfit and backtranslation. *arXiv* **2019**, arXiv:1903.09244.
52. Tsvetkov, Y. Linguistic knowledge in data-driven natural language processing. In *The Requirements for the Degree of Doctor of Philosophy in Language and Information Technologies*; Carnegie Mellon University: Pittsburgh, PA, USA, 2016.
53. Kumar, A.S.; Kumar, N.S.; Devi, R.K.; Muthukannan, M. Analysis of Deep Learning-Based Approaches for Spam Bots and Cyberbullying Detection in Online Social Networks. In *AI-Centric Modeling and Analytics*; CRC Press: Boca Raton, FL, USA, 2024; pp. 324–361.
54. Akinyelu, A.A. Advances in spam detection for email spam, web spam, social network spam, and review spam: ML-based and nature-inspired-based techniques. *J. Comput. Secur.* **2021**, *29*, 473–529. [CrossRef]
55. Karishma, S.; Akila, V.; Govindasamy, V. Spam Detection using Recurrent Neural Networks. *Int. J. Res. Eng. Appl. Manag.* **2020**, *6*, 313–318.