machine learning & knowledge extraction

Article

Cyberattack Detection in Social Network Messages Based on Convolutional Neural Networks and NLP Techniques

Jorge E. Coyac-Torres †, Grigori Sidorov †, Eleazar Aguirre-Anaya *,† and Gerardo Hernández-Oregón †

Centro de Investigación en Computación (CIC), Instituto Politécnico Nacional (IPN), Av. Juan de Dios Batiz, s/n, Mexico City 07320, Mexico; jcoyact2019@cic.ipn.mx (J.E.C.-T.); sidorov@cic.ipn.mx (G.S.); ghernandezo1101@alumno.ipn.mx (G.H.-O.)

* Correspondence: eaguirre@cic.ipn.mx

† These authors contributed equally to this work.

Abstract: Social networks have captured the attention of many people worldwide. However, these services have also attracted a considerable number of malicious users whose aim is to compromise the digital assets of other users by using messages as an attack vector to execute different types of cyberattacks against them. This work presents an approach based on natural language processing tools and a convolutional neural network architecture to detect and classify four types of cyberattacks in social network messages, including malware, phishing, spam, and even one whose aim is to deceive a user into spreading malicious messages to other users, which, in this work, is identified as a bot attack. One notable feature of this work is that it analyzes textual content without depending on any characteristics from a specific social network, making its analysis independent of particular data sources. Finally, this work was tested on real data, demonstrating its results in two stages. The first stage detected the existence of any of the four types of cyberattacks within the message, achieving an accuracy value of 0.91. After detecting a message as a cyberattack, the next stage was to classify it as one of the four types of cyberattack, achieving an accuracy value of 0.82.

Keywords: bot; CNN; cyberattack; deep learning; malware; NLP; phishing; social networks; spam

Citation: Coyac-Torres, J.E.; Sidorov, G.; Aguirre-Anaya, E.; Hernández-Oregón, G. Cyberattack Detection in Social Network Messages Based on Convolutional Neural Networks and NLP Techniques. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 1132–1148. https://doi.org/10.3390/make5030058

Academic Editor: Simon Tjoa

Received: 1 August 2023
Revised: 24 August 2023
Accepted: 28 August 2023
Published: 1 September 2023

1. Introduction

One of the services people utilize most on the Internet is social networks. The popularity of these sites derives from the advances in information technology, which provide users with the practicality and simplicity to use these digital media, allowing them to access these services from anywhere with an Internet connection through an electronic device. Nowadays, social networks have become an excellent medium for sharing information, reaching places humanity could hardly have imagined. In these interaction spaces [1], users are the main actors, who involuntarily participate in the generation of data, which happens from the moment they post a comment about an event that attracts their attention.

The amount of information that circulates on social networks minute by minute is enormous because the data are generated and published not only by member users but also by public and private institutions, government entities, and even electronic devices such as sensors and actuators [2], which integrate intelligent systems as processes within the Internet of Things (IoT) technology [3].

With events such as COVID-19, people were forced to work remotely, which caused many individuals to engage in digital interactions to conduct various activities such as homework, reports, meetings, and other activities on the Internet. Traditional communications changed to digital platforms such as social networks, and being the most used service, they became a broader panorama to carry out attacks on different users and institutions, facts that have been worrisome and recurring for victims of these threats within these platforms. Furthermore, just as social networks have captured the attention of many people, in the same way, these services have attracted various malicious users whose purpose is to compromise the assets of other members through different cyberattacks, which represent a threat within these communities [4].

The practicality, security controls, and mechanisms implemented on current web platforms [5] to prevent different types of cyberattacks demonstrate that threats grow along with advances and technological developments. These facts have concerned many researchers, institutions, and companies, in such a way that it has prompted the development of different works, mechanisms, and software tools as a countermeasure against different threats from malicious users or groups who have contemplated using social networks as an attack vector to compromise the assets of their victims.

Therefore, this paper presents a novel approach based on a convolutional neural network (CNN) architecture to detect and classify four different cyberattacks such as malware, phishing, spam, and even one whose aim is to deceive a user into spreading malicious messages to other users, which, in this work, is defined as a bot attack. An essential feature of this approach involves utilizing natural language processing (NLP) techniques such as filtering, Part-of-Speech (POS) tagging, lemmatization, tokenization, and stopword removal. These techniques extract specific text attributes from each message to represent and create the subsequent inputs to the deep learning (DL) architecture.

In addition to the vector that represents the content of the message, two vectors complement the information utilized for the detection and classification activities. The first vector comprises the features of the plaintext structure with a size equal to 16, which is the total number of extracted values that describe how the user typed the message.

The second vector is created based on the properties of each Uniform Resource Locator (URL) mentioned in the message. Even if a URL is not present in the content, a vector with specific attributes is utilized to indicate the absence of a URL. In general, the characteristics extracted from each URL integrate this vector with a total of 25 properties that define its size. It is essential to highlight that the analysis is applied to textual content and not to multimedia such as images, audio, video, or other formats.

On the other hand, the idea of implementing an architecture based on a CNN aims to identify significant words, which are represented sequentially in each hashing vector, for the detection and classification activities of the four threats mentioned above. This activity is the extraction of characteristics necessary for the entry to the next stage of this architecture. Unlike traditional CNN architectures, this work proposes to implement an additional layer within this architecture to avoid extrapolating information from the message content with characteristics related to the morphology of the text and the properties of the existing URLs. Hence, this additional layer concatenates the three vectors proposed before entering the final stage of the architecture.

One remarkable feature of this work is that it detects the previously mentioned four types of cyberattacks using just the proposed approach without needing to employ different models for each type of threat, allowing users to expand the detection landscape of these threats in social network messages, even if the message does not have a URL. This CNN architecture was tested on real data, demonstrating its results in two stages. The first stage detects the existence of any of the four types of cyberattacks within the message, achieving an accuracy value of 0.91. On the other hand, after detecting a message as a cyberattack, the next stage is to classify it as one of the four types of cyberattack, achieving an accuracy value of 0.82.

This paper is organized as follows. First, a review of the related works in the literature is presented in Section 2. The methodology for analyzing messages is described in Section 3. The cyberattack detection model is explained in Section 4. Section 5 presents the relevant numerical results, evaluations, and a discussion of the results. Finally, Section 6 presents the conclusions.

2. Related Works

The diversity of topics within social networks has allowed for the consideration of the content of messages as an information source. Research and studies on different topics analyze the posted messages on these sites, such as the acceptance and trends of a product in the market [6,7], the study of preferences and reactions to a new law [8,9], and movements in the stock market [10,11], among other topics. These demonstrate that users perform a function similar to observers of social events or preference indicators, and all this information is helpful for further analysis.

Similarly, the information from these sites is analyzed to identify malicious content within the messages shared among users on social networks to prevent diverse forms of cyberattacks from affecting the assets of users or institutions using this route as an attack vector [12]. Kunwar et al. [13] mentioned that social networks have become a new attack vector, and various malicious users are looking for opportunities to take advantage of them. Meanwhile, Saidi et al. [14] examined the importance of analyzing the communities and information in social networks, concluding that modeling and semantic analysis can extract information from these sites to provide a profound vision of the operations of clandestine groups of cybernetic terrorists.

It is important to mention that activities involving the analysis of information contained in social network messages for the detection of cyberattacks cannot be compared to the performance of software tools and mechanisms used in cybersecurity, such as antivirus software, firewalls, Intrusion Detection Systems (IDS), and Intrusion Prevention Systems (IPS), among others. However, due to the popularity of social networks, cybercriminals have found these sites to be a way of accessing the assets of victims and avoiding security controls, turning these services into spaces for the propagation of cyberattacks.

Lippmann et al. [15] contributed to the creation of an automated process for extracting cybersecurity discussions from online forums, arguing that this would reduce the amount of time an analyst needs to remove irrelevant content in their investigation in this area. Similarly, information sources such as blogs and discussion forums have been analyzed to find malicious content in shared messages. In this sense, Grishman et al. [16] proposed a method for identifying messages containing information or distribution characteristics of malware applications to determine the users that spread threats in deep/dark web forums and blogs. They used a recurrent neural network (RNN) architecture and analyzed connections among users using graphs.

The following works focus on detecting the existence of some variant of a cyberattack in messages from social networks. Unlike traditional security controls, these works do not contain, mitigate, or eliminate threats. However, these report findings to the user to evaluate the actions to take.

Liao et al. [17] presented “iACE”, a framework for extracting Indicators of Compromise (IoC) from unstructured texts. The IoC [18] concept describes malicious activities or artifacts relevant to a cybersecurity incident by analyzing their behavior patterns for identification. Liao et al. [17] proposed using NLP NER/ER techniques to gather information from public sources such as blogs, articles, and other publicly accessible written media, creating a series of tokens and terms that allow defining unique characteristics of different cyberattacks.

A study aimed at detecting phishing was carried out by Liew et al. [19], who developed a real-time security alert mechanism using a classification model derived from a supervised machine learning technique called Random Forest (RF). They identified 11 classification features that yielded a 94.75% accuracy value after analyzing 200 phishing URLs collected from Twitter and PhishTank to determine the effectiveness of their model.

Erkal et al. [20] proposed an alert system for detecting cybersecurity threats such as spam and malware spread through Twitter messages, utilizing a Naive Bayes classifier and vectorization based on the weights of the Term Frequency/Inverse Document Frequency (TF/IDF) values. They achieved a success rate of 70.03% in detecting phishing and spam content.

Ashour et al. [21] evaluated the performance of their classifier based on n-grams for spam detection in Twitter messages and obtained better results than those achieved using TF/IDF techniques. They achieved an F1 value of 0.794, according to their reported results.

Wu et al. [22] proposed a model based on deep learning techniques to address spam detection, where the syntax of each tweet was learned using the WordVector technique for preprocessing and converting them into high-dimension vectors, achieving an F-measure value of up to 0.942.

Meanwhile, Feng et al. [23] proposed a framework based on DL for spam detection, which sets up a detection system at the mobile terminal and the server. This model utilizes the Word2vec tool to convert words into vectors, achieving an accuracy value of 0.9136 in the classification model on the server side.

Madisetty et al. [24] presented a method that combines five CNN models and one feature-based model through a neural network for spam detection at the tweet level. Each CNN was trained with word embeddings of different dimensions, and the feature-based model utilized three types of features (user-based, content-based, and n-gram features), achieving an F-measure value of 0.894.

Chen et al. [25] proposed an analysis to detect low-quality content in real time in Twitter messages, which involves identifying valueless content of different types from the perspective of the user, including spam and phishing content. The method requires direct and indirect Tweet features, as well as word-level analysis, to capture the content characteristics of the text. The results of their research show that this method achieved an F1 score of 0.8379 through an RF classifier.

Djaballah et al. [26] presented an approach to detect phishing in Twitter messages that combines a verification method in a database or "Blacklist" with a classification model based on URLs and web page content analysis. They achieved an accuracy that exceeded 95% with the RF classifier.

The previous works analyzed messages from social networks through different models to detect from one to three specific cyberattacks. Nevertheless, this work proposes using a detection model that combines NLP tools and a CNN architecture to detect four types of cyberattacks in social network messages: phishing, spam, malware, and bot cyberattacks (which involve deceiving users into spreading malicious content). Furthermore, this approach analyzes the messages even if there is no URL in the content. Similarly, when the message contains more than one URL, this model examines each one to detect the possible existence of more than one cyberattack in a single message.

One advantage of the proposed work is that it does not require consideration of specific characteristics of social networks, such as the number of followers, likes, people the user follows, and other details that different social networks may share, but that others lack to analyze the content of a message.

Therefore, this work can be used to examine the content of diverse blogs, discussion forums, and social networks as sources of information for cyberattack detection tasks. After detecting a message with characteristics of any of the four types of cyberattacks, this model classifies according to the specific type of cyberattack associated with the malicious content. The following section presents the proposed methodology of this work, which outlines the phases for analyzing each message.

3. Methodology of Social Network Message Analysis

This work aims to detect four types of cyberattacks: spam, phishing, malware distribution, and bot attacks (which refers to the spreading of harmful messages among users, whether or not the attacker interacts with the victim). Therefore, the problem is defined as follows: given a message m, consider four subsets of a set C – C_1 for spam, C_2 for phishing, C_3 for malware, and C_4 for a bot attack. The task is to detect whether m belongs to the C set. If m is in C, classify m according to which subset of C it belongs to.

To address this problem, this work presents the following methodology, which involves analyzing the content of texts from diverse social networks, discussion forums, and blogs to detect and classify the content according to the previously mentioned problem. This methodology utilizes tools and techniques such as NLP and a DL model based on a CNN to detect and classify each message. Therefore, it only considers the text and disregards multimedia elements like images, videos, animations, or audio. Moreover, the importance of creating a model capable of detecting a more significant number of threats and being utilized regardless of the data source is seen as an opportunity for extending the landscape of cyberattack detection tasks in this kind of content.

Figure 1 illustrates the proposed pipeline for cyberattack detection in social network messages, depicting the stages from gathering the message to detecting and classifying it using the CNN model.

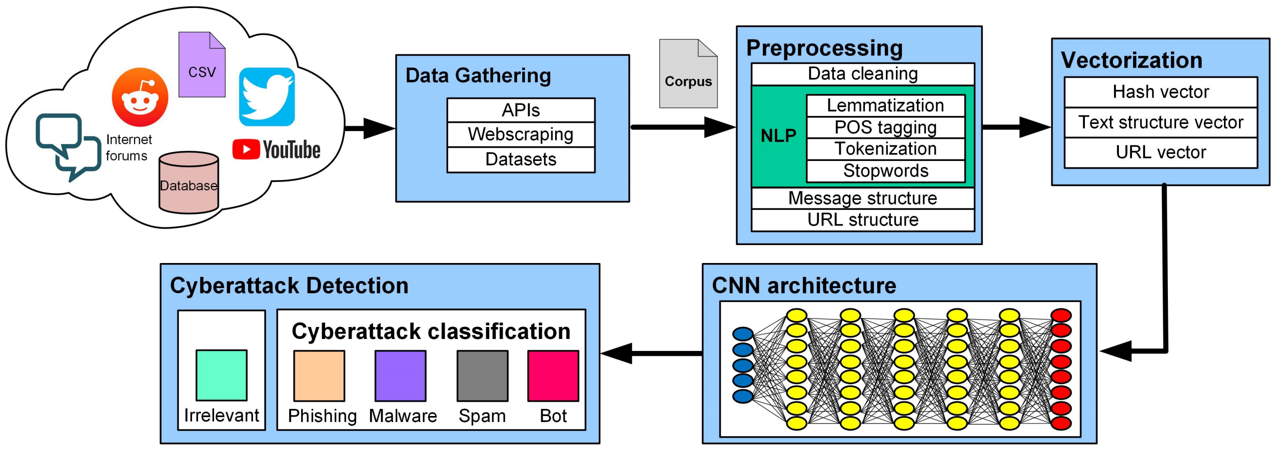

Figure 1. Pipeline of stages of cyberattack detection in social network messages.

3.1. Data Gathering

The gathered information integrates the dataset utilized for the training and testing activities of the CNN model, making these data the corpus for the model. The messages obtained through the following mechanisms and resources serve as inputs for the preprocessing stage of this methodology:

- **APIs**
  This software mechanism enables the execution of activities to request samples of publications through functionalities specific to each social network. These functionalities include searching by keywords, specific accounts, retrieval of messages by ID, publication dates, and other properties. For example, to collect Tweets and Retweets from Twitter, an endpoint connection from their API was used to download available public messages.
- **Scraping**
  Websites such as blogs and discussion forums operate based on browsing through the HTTP protocol. To extract messages from these interaction spaces, this work utilized web-scraping and -crawling techniques to simulate recursive browsing through different websites.
- **Datasets**
  To complement the corpus of this work, a subset of messages from different datasets was used from works such as Lampu [27], Behzadan [28], and Chen [25], in which the content refers to messages labeled as spam, phishing, and spam, respectively.

3.2. Preprocessing

The next step is preprocessing, which involves using NLP techniques to create three arrays. These arrays represent the properties of each message, such as the grammar, lexical, and semantics through tokens, the structural composition of the message, and the representation of features based on the URL structure.

3.2.1. Content of Message

The details of the corpus and NLP tools utilized for content extraction from each message in this stage are shown below:

1. **Corpus**
   This corpus comprises two values for each message: the raw text content and the label that categorizes the message into one of the subsets related to the initial problem. In addition, this corpus includes a selection of messages labeled "irrelevant" due to their non-malicious content, according to the data sources used for this work. Hence, there are five labels in this corpus: malware, phishing, spam, bot, and irrelevant.
2. **Data Cleaning**
   This activity removes unnecessary characters from the text without altering the main idea of the message. To filter each message, different tasks are performed, such as eliminating unrecognized characters, replacing repetitive line breaks and blank spaces with only one, and removing emoticons and emojis by using regular expression rules.
3. **Lemmatization**
   The lemmatization procedure reduces the morphological variants of a word to their roots or lexemes. Thus, the number of words that appear in this corpus is reduced, making it possible to minimize the diversity of the words needed to represent the content of a message.
4. **POS Tagging**
   The purpose of the Part-of-Speech (POS) tags in this work is to accurately define the intended lemma based on the context and grammar of a message, rather than assuming a generic lemma transformation of a word. By applying POS tagging analysis to correctly convert words into lemmas, the number of words in this corpus is reduced compared to considering the text of a message in its raw form.
5. **Tokenization**
   Tokenization involves breaking down the text of each message from this corpus at the word level to produce an array of individual elements called tokens.
6. **Stopwords**
   Stopwords play a crucial role in the text by serving as the connecting point for syntax and grammar. However, they do not have meaning when written alone. Thus, to represent the main idea of each message with a reduced number of tokens, the stopwords in each token array were removed.

3.2.2. Message Structure

The text structure of a message shows how the content was posted and shared on social networks. In this sense, the use of features based on the structure of the text derives from the following hypothesis: extracting morphological properties from the structure of the text of a message could reveal characteristics that harmful content may share with written messages from malicious users. Through experimental analysis of diverse characteristics from messages in this corpus, the 16 properties that yielded the most satisfactory results during the subsequent stages of this work were determined, as shown in Table 1.

Table 1. Features extracted from the message structure.

(1) Length of message	(9) # Sentences
(2) # Tokens	(10) # Mentioned users
(3) # Hashtags	(11) # Uppercase
(4) # Emails	(12) # Question marks
(5) # URLs	(13) # Exclamation marks
(6) # Periods	(14) # Emojis and emoticons
(7) # Commas	(15) # Dollar symbols
(8) # Digits	(16) # Other symbols

Once information about the structure of the text has been extracted, specific text components, such as email addresses and mentioned users, are replaced with predefined tokens like "email_nlp" and "at_user_nlp" to simplify the message and reduce the number of words that exist in the corpus. Furthermore, the hash symbol "#" is removed from the aforementioned tags to be used as tokens.

3.2.3. URL Structure

URLs in a message sometimes may not show the true website domain to the user because of URL shortening services, causing users to be redirected to a different website than expected when clicking the URL. By utilizing software tools to reveal the final website of shortened URLs, this stage analyzed the authentic domain and extracted characteristics from the resulting URL. Table 2 shows 27 characteristics used to represent the properties of a URL present in a message, which delivered the most satisfactory results during the subsequent stages of this work.

Table 2. Features extracted from the URL structure.

(1) Length of URL	(10) # Words	(19) # Equal symbols
(2) Has security protocol (Y/N)	(11) # IPs	(20) # Question marks
(3) Creation date (Days)	(12) # Digits	(21) # Wave symbols
(4) Last update date (Days)	(13) # Hyphens	(22) # Plus signs
(5) Shortened URL	(14) # Periods	(23) # Colon symbols
(6) # Underscores	(15) # Slashes	(24) # Other characters
(7) # Strings divided by periods	(16) # Uppercase	(25) Has extension (Y/N)
(8) # Strings divided by hyphens	(17) # Lowercase	(26) Domain suffix (Tokens)
(9) # Strings divided by slashes	(18) # Ampersand symbols	(27) Registrant (Tokens)

Furthermore, if a message does not contain a URL, the structure uses specific values called "URL_ZERO" to indicate the absence of a URL in the message. Sometimes, messages may have more than one URL. In such cases, each URL is analyzed separately, resulting in a structure for every URL included in the message.

3.3. Vectorization

This step involves encoding the arrays containing the tokens, text structure, and URL structure into numeric values, which will serve as the input, as vectors, to the CNN architecture. The procedures to obtain these vectors are explained below:

- **Hashing Vector**
  Each word in the dictionary of this corpus is assigned a unique integer hash using the hashing model. The hash acts as an identifier for each token in the array, resulting in a vector of integers representing the initial token array. In addition to the token array, two characteristics from the URL structure are appended as tokens, such as the "domain suffix" and "registrant," which complement the content using information referring to each URL that appears. Likewise, if the message does not have a URL, the tokens "no_domain" and "no_suffix" are appended to the URL_ZERO values.
  Finally, unlike other models such as the bag-of-words model, the size of the vectors used in this model does not depend on the number of words in the vocabulary. Therefore, the vector size was determined based on the length of the message with the maximum number of tokens after the content analysis from the training dataset. This value defines the size of these vectors.
- **Text Structure**
  This vector represents the structural and typographic characteristics of the raw text in a message. The vector consists of the extracted values, as shown in Table 1, from the message structure array. Thus, the size of this vector is equal to the number of extracted features, with a total of 16 values.

- **URL Structure**
  This vector is created for each URL present in the message. The extracted values from the URL structure array, as shown in Table 2, integrate this vector. Likewise, if the message does not have a URL, this vector is created using predefined values, defined as "URL_ZERO" values, that symbolize the lack of a URL in the message.
  Although the quantity of elements in the URL structure is 27, this vector utilizes 25 because two of them (the "domain suffix" and "registrant" values) are utilized as tokens for the hashing vector process. Therefore, the final number of elements represents the size of the resulting URL vector.

3.4. Deep Learning Model

Kim [29] stated that a CNN model can utilize documents as a matrix of words, similar to how it handles pixels for image recognition and computer vision tasks. For this reason, and according to our earlier review of the works in the literature that employ this model, this part presents a CNN architecture that utilizes the three resulting vectors from the previous stages to detect and classify the cyberattacks defined in the initial problem.

After the flattening layer, an additional layer is added to the architecture. This layer concatenates the resulting convolutional vector using the "text" and "URL" structure vectors before continuing with the fully connected layers. The details of each stage in this CNN architecture are shown below:

1. **Convolution Layer**
   The initial layer of this architecture is a convolution layer, which extracts the features from the hash tokens to preserve the relationship between each value. This layer utilizes a *ReLU* activation function to break the linearity and increase the non-linearity during this process.
2. **Dropout Layer**
   This layer utilizes a downsampling operation [30] to make this model more robust to variation, absence, and distortion.
3. **Pooling Layer**
   The pooling operation involves sliding a two-dimensional filter to summarize the features within a region covered by a filter and reduce the number of parameters to learn during the computation performed in the DL model.
4. **Flattening Layer**
   This layer converts the pooling output into a one-dimensional matrix.
5. **Concatenation Layer**
   This layer concatenates the resulting flattened vector with the text and URL structure vectors, creating an input vector for the fully connected layers.
6. **Fully Connected Layers**
   These layers utilize the resulting concatenated vector as input. It is composed of input, hidden, and output layers, where the output layer utilizes the *Softmax* activation as an activation function to generate the final results.

4. Cyberattack Detection Model

Nowadays, social networks continue to demonstrate that they are a widely used option for spreading diverse types of cyberattacks [13]. For this reason, the following model focuses on detecting and classifying four different types of cyberattacks (spam, phishing, malware, and bot attacks), which are related to the term social engineering attacks. The malicious content associated with these attacks can be distributed among the community, sometimes without any interaction between the attacker and the victim.

This section presents information about the amount of data utilized for testing and training, the utilization of the NLP tools to analyze each message through an algorithm, and details about the proposed CNN architecture using the TensorFlow [31] framework to detect cyberattacks in social network messages.

4.1. Dataset

The messages utilized in this work come from diverse sources, including social networks, blogs, and other referred sources. Some of these messages were obtained by sampling from Twitter using its API to create a compilation of available messages, extracting different types of content, and integrating them into a corpus.

Meanwhile, other messages were gathered from websites such as Reddit and from software product announcements on YouTube. Moreover, this work complemented the corpus with specific messages collected from sources such as Kaggle, including SMS messages [27] marked as spam, and subsets of Tweets from related studies, such as those by Behzadan et al. [28] and Chen et al. [25].

Figure 2 shows the number of messages utilized in this work, with a total of 39,670 posts, where each item contains only the raw text and a label that identifies the type of content. In addition, the content of the messages was reviewed to exclude details such as empty content and duplicated items from the corpus.

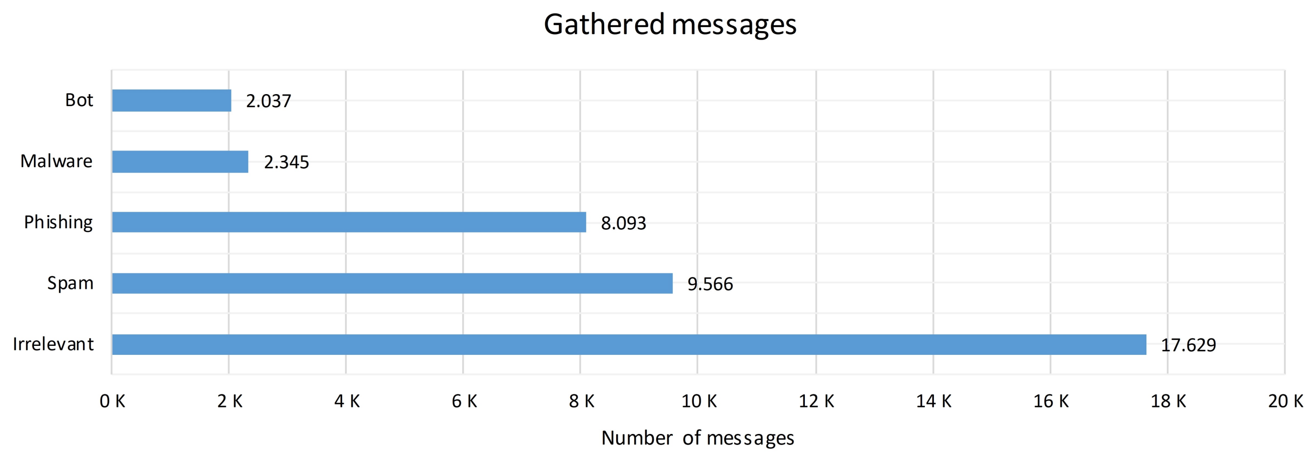

Figure 2. Distribution of messages in the dataset.

This corpus contains five labels (spam, phishing, malware, bot, and irrelevant) for messages that do not fit into the cyberattack categories. Therefore, these samples are considered a representative portion within a scenario where the information is labeled for the testing and training purposes of the CNN architecture.

4.2. Analysis

Once the corpus has the gathered messages, they are saved in a CSV file for the training and testing stages of the model. The next step is to analyze each message in the CSV file through the stages explained earlier. Algorithm 1 presents this novel approach to analyzing each message before implementing the methodology, returning a list of three-tuple objects that contain the tokens, text structure, and URL structure, which are then prepared for the next stage: vectorization.

Algorithm 1 Preprocessing.

Require: Text from message
Ensure: List of 3-tuple objects

```
1: function ANALYZE_MESSAGE(message_text)
2:     STOP_WORDS ← load_stopWords()
3:     structure_vector ← structure_features(message_text)
4:     Tokens ← empty list
5:     text ← clean_data(message_text)
6:     Sentences ← text_to_sentences(text)
7:     $N$ ← len(Sentences)
```

```
 8:     for $i \leftarrow 1$ to $N$ do
 9:         sentence ← Sentences[$i$]
10:         tagged_tokens ← POS_tags(sentence)
11:         $M \leftarrow$ len(tagged_tokens)
12:         for $j \leftarrow 1$ to $M$ do
13:             append lemmatize(tagged_tokens[$j$]) to Tokens
14:         end for
15:     end for
16:     tokens_message ← empty list
17:     $L \leftarrow$ len(Tokens)
18:     for $k \leftarrow 1$ to $L$ do
19:         if Tokens[$k$] not in STOP_WORDS then
20:             append Tokens[$k$] to tokens_message
21:         end if
22:     end for
23:     Vectors ← empty list
24:     URLs ← find_URLs(message_text)
25:     $K \leftarrow$ len(URLs)
26:     if $K = 0$ then
27:         URLs[1] ← URL_zero
28:     end if
29:     for $w \leftarrow 1$ to $K$ do
30:         URL_features ← extract_URLfeatures(URLs[$w$])
31:         append URL_features.domain_suffix to tokens_message
32:         append URL_features.registrant to tokens_message
33:         URL_vector ← URL_features[$1, \cdots, 25$]
34:         v ← (tokens_message, structure_vector, URL_vector)
35:         append v to Vectors
36:     end for
37:     return Vectors
38: end function
```

The vectorization process involves taking each element from the three-tuple list and converting it into a one-dimensional array using the corresponding vectorization model. For the vector concerning the tokens of the message, the hashing model replaces each token with a hash value in a vector, and the size of this vector depends on the maximum number of tokens in the most extended analyzed message in the corpus.

Meanwhile, for the two remaining vectors, the size of the vector representing the structural text is 16 features, and the URL vector has a size of 25 extracted features. After the convolution process, the text and URL structure vectors are combined with the flattened vector to create a final vector. This final vector is then fed into the input layer for the fully connected layer of the CNN architecture.

4.3. CNN Architecture

CNNs are not standalone models; they are designed to be incorporated into a more extensive architecture to work collaboratively and generate an outcome. In this way, the role of the CNN layers is to extract significant data components at the level of the hash value, where each hash represents a token, and each element corresponding to the input of the convolution stage corresponds to a region of the text of the message.

In this work, the implementation of the convolution and pooling operations is based on applying a non-linear function through a sliding window of k-values on each hashing vector, resulting in a dimensional vector that represents the main properties for each sliding window. Afterward, the pooling stage merges the vectors resulting from different windows to create a one-dimensional vector, selecting the maximum value in each dimension over the different windows. Together, these operations facilitate the extraction of important values from each input vector, regardless of the location of the represented words within the message.

An additional aspect of this convolution stage involves the exclusive analysis of the message content. In this sense, an additional layer is added to the architecture to prevent the extrapolation of information from the message content with characteristics related to the morphology of the text and the properties of the existing URLs. Hence, this proposed layer concatenates the two additional vectors (text and URL structures) with the resulting vector from the convolution process before entering the final stage of the architecture. Figure 3 shows the main components of the CNN architecture.

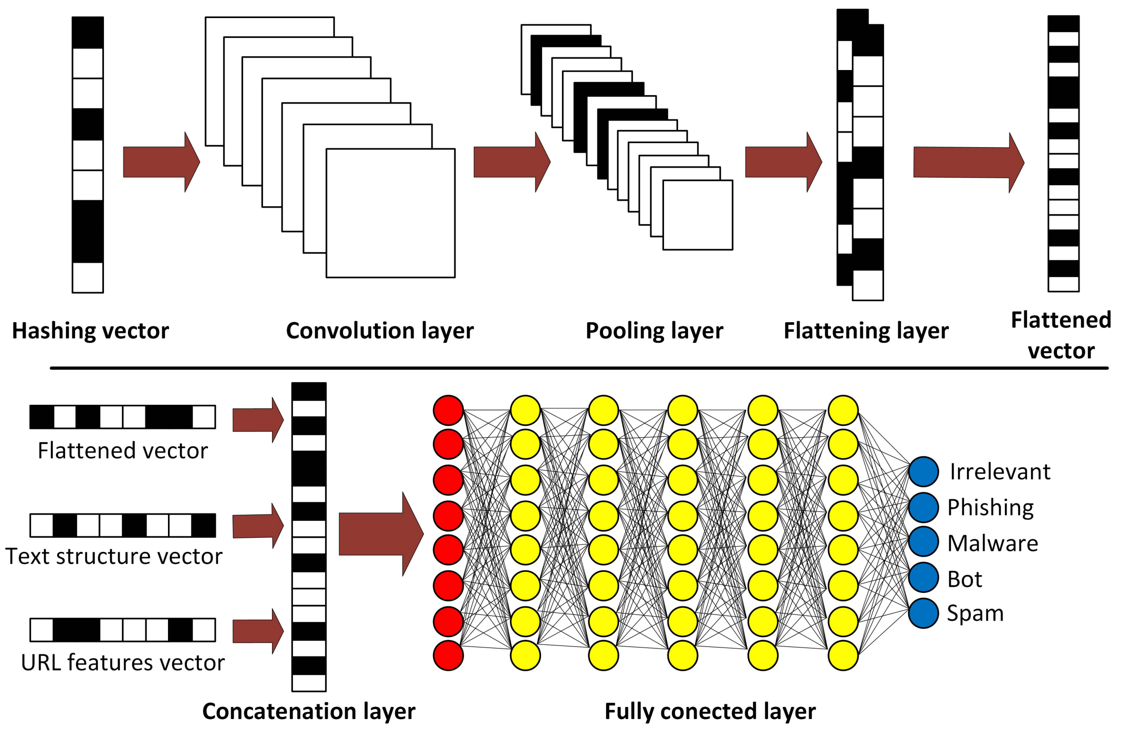

Figure 3. CNN architecture for cyberattack detection and classification.

The properties and values used for the convolution and pooling operations during the experiments are shown below:

- Size of input vector: 93;
- Filters: 8;
- Size of kernel: 8;
- Activation function: *ReLU*;
- Dropout: 0.26;
- Pool size: 2;
- Size of flattened vector: 344;
- Size of concatenated vector: 385.

This architecture utilized 70% of the data for training and 30% for testing. Thus, the main parameters and features of the architecture after the convolution process that yielded the most satisfactory results during these tasks were as follows:

- One input layer for the concatenated vector (285 neurons);
- Six hidden layers (155, 105, 50, 25, 25, and 10 neurons);
- Output layer (5 neurons);
- Adam optimizer;
- Kullback–Leibler divergence as the loss function.

Meanwhile, for the fully connected layers, the following activation functions were implemented from the input to the output layers: three *Sigmoid*, four *ReLU*, and one *Softmax*. Once the vector from the flattened layer reaches this section of the architecture, the outcome should come from one of the five neurons in the final layer. In this section, the result comes from the neuron with the maximum value, and its activation function in this layer is defined by the Softmax function $\sigma : \mathbb{R}^N \mapsto (0,1)^N$ when $N \geqslant 1$. This result is defined by:

$$\sigma(Z_i) = \frac{e^{Z_i}}{\sum\limits_{j=1}^{N} e^{Z_j}} \quad \text{for } i = 1, \cdots, N \text{ and } Z = (Z_1, \cdots, Z_N) \in \mathbb{R}^N, \tag{1}$$

where the input vector Z consists of N real numbers in the last layer, and each component must be in the interval (0, 1). The values in the Z vector should add up to 1, and each value represents the probability of each neuron. The maximum value from this layer represents the outcome of this architecture.

Finally, the outcome of this architecture represents the analysis performed on the content of messages from social networks. Its objective is to address scenarios where it is vital to detect any of the four variants of cyberattacks mentioned above. The goal is to prevent users from falling victim to malicious acts that can compromise both digital assets and private personal information. However, this approach is limited to these four variants of cyberattack due to the type of data on which it was trained, which involves supervised learning. However, by not depending on a specific data source or a third-party service, the design of this architecture could be trained with other types of cyberattacks that rely on how the elements of the training dataset are categorized for the correct detection and classification of threats.

5. Results and Discussion

This section presents, interprets, and discusses the results obtained using this CNN architecture for cyberattack detection and classification tasks. Firstly, the results, graphs, tables, and confusion matrices related to the outcomes and behavior of the proposed approach are presented. Secondly, in the discussion subsection, the results of this study are compared to the works mentioned in Section 2.

5.1. Results

The labeled messages used to conduct the training and testing tasks come from the review tasks carried out by the research team and from previously cited information sources. Figure 4 shows the performance of the model using the final settings of the architecture, where the graphs represent the behavior of the loss function and accuracy value during the training and testing processes.

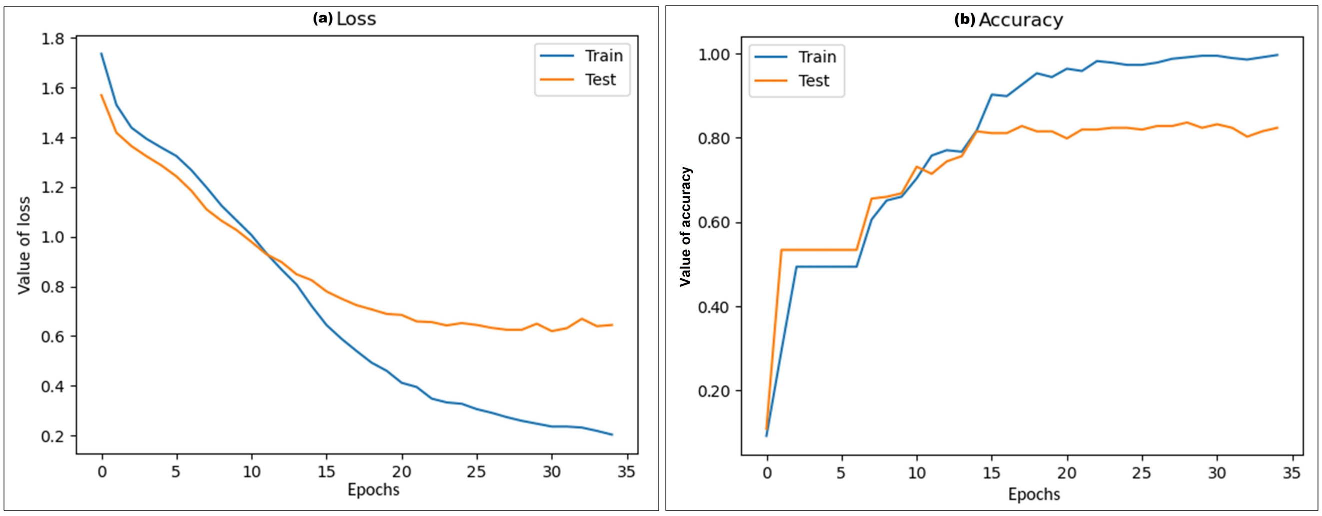

Figure 4. Behavior of the CNN architecture during the training and testing processes using the values shown in Section 4.3. (**a**) Value of the loss function. (**b**) Accuracy value of the CNN architecture.

As can be seen, Figure 4a shows that the loss function or cost function tended to decrease in value simultaneously during training and testing, reaching a point where the error value of the model was much lower for the training data compared to the test data due to the system adjusting the values during training. Meanwhile, the test data show the effect on the model using these settings. Furthermore, this graph depicts that the performance of the model did not fall into overfitting when the training process attempted to decrease the error value, achieving an error cost of 0.23 on the training data and 0.64 on the test data during the loss function evaluation.

On the other hand, Figure 4b shows the accuracy value obtained during the optimization of the model. As can be seen, the model reached a maximum level of 0.96 on the training data. Similarly, the behavior of the model on the test data increased, achieving an accuracy value of 0.82, demonstrating that these values remained almost constant during training and testing, presenting only minimal variation. Therefore, there was no reason to continue training the model and avoid falling into overfitting.

To address the first part of the problem outlined in Section 3, which involved detecting whether a message contains characteristics of any of the four analyzed cyberattacks, the accuracy, precision, recall, and F1 score values were obtained using the model with the test data, as shown in Table 3.

Table 3. Cyberattack detection results.

	Precision	**Recall**	**F1 Score**
Irrelevant	0.89	0.93	0.91
Cyberattack	0.93	0.88	0.90
Accuracy			**0.91**

For the second part of the problem, the messages were classified according to the characteristics of cyberattacks related to spam, phishing, malware, and bots, as shown below. In addition, a subset of irrelevant messages was utilized to evaluate the accuracy in situations where false positives occur during the detection task of the model, achieving an accuracy value of 0.82 for this task. Table 4 shows the precision, recall, F1 score, and accuracy values achieved in the evaluation task.

Table 4. Cyberattack classification results.

	Precision	**Recall**	**F1 Score**
Irrelevant	0.89	0.93	0.91
Phishing	0.71	0.77	0.74
Malware	0.76	0.56	0.65
Bot	0.60	0.52	0.55
Spam	0.85	0.92	0.88
Accuracy			**0.82**

Figure 5 demonstrates the confusion matrices related to the results of the messages analyzed using this CNN architecture with the test data to evaluate the cyberattack detection and classification tasks. Figure 5a shows the number of cyberattacks and irrelevant messages correctly detected, or true positives and negatives. As can be seen, the correctly detected results are much higher than the remaining values in this confusion matrix, validating the cyberattack detection performance with an accuracy of 0.91.

Figure 5b displays the distribution of the results when the model performs tasks to classify the message according to the type of cyberattack identified. It shows how the predicted results appear with respect to the actual labeled messages in the confusion matrix that encompasses the five categories. Hence, this CNN architecture demonstrates its effectiveness by accurately classifying true positives and negatives while addressing the remaining values within this matrix.

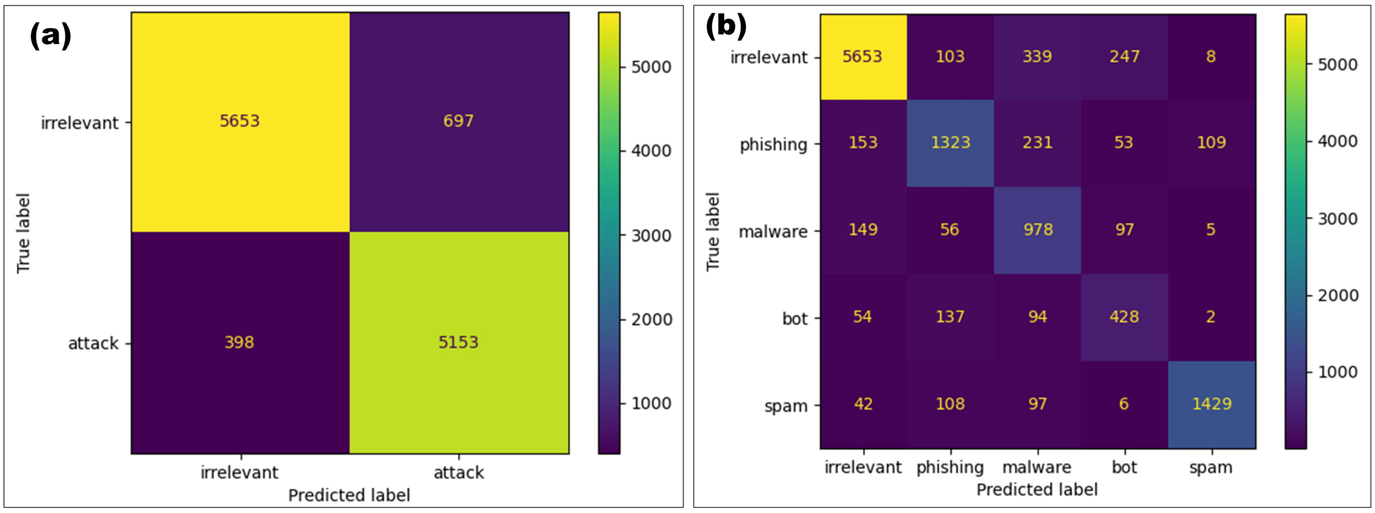

Figure 5. Confusion matrices of messages analyzed during the evaluation of the model utilizing the proposed CNN architecture. (**a**) Cyberattack detection. (**b**) Cyberattack classification.

Even though the proposed approach achieved a lower accuracy value in the classification process compared to the detection process, it could still identify a considerable number of messages related to any of the four types of cyberattacks in the problem. Likewise, if the model misclassified the content into another category, it could still identify a threat as a possible cyberattack rather than irrelevant content.

5.2. Discussion

One of the notable aspects of this work is the integration of NLP tools with a deep learning model, which has proven to be a synergistic approach in cyberattack detection tasks, as demonstrated in the works by Ashour et al. [21], Wu et al. [22], and Feng et al. [23]. Table 5 shows a comparison of the results of the proposed model with the results obtained in the works discussed in the literature review presented in Section 2.

Table 5. Comparison of cyberattack detection results.

Model	Cyberattacks	Accuracy	F1 Score
Random Forest [19]	Phishing	0.948	0.95
Random Forest [26]	Phishing	0.95	0.94
Deep NN [22]	Spam	0.942	0.94
Our CNN	**Spam, malware, phishing, and bot attacks**	**0.91**	**0.904**
CNN [24]	Spam	0.957	0.894
Random Forest [25]	Spam, malware, and phishing	0.971	0.838
n-grams [21]	Spam	0.794	0.794
CNN [23]	Spam	0.9136	—
Naive Bayes [20]	Spam and malware	0.703	—

These works are arranged in descending order according to the F1 scores. The F1 score is a more effective metric for solving class imbalance issues compared to the accuracy metric, which is better suited for balanced classes.

In this comparison, it can be seen that the approaches of Liew et al. [19] and Djaballah et al. [26] were the top performers with the highest F1 values. They analyzed URLs in social network messages to detect phishing. However, the proposed approach performs the detection task regardless of whether there is a URL present in the content. Even if multiple URLs are present, it detects any of the four types of cyberattacks mentioned earlier, making it possible to explore whether more than one cyberattack appears in a single message.

Although this work did not obtain the highest F1 score, its score was promising with a value of 0.904. One notable aspect of this work is its ability to detect four different types of cyberattacks. Meanwhile, the message analysis tasks in the works by Chen et al. [25] and Erkal et al. [20] could identify up to three different types of cyberattacks but achieved F1 values of less than 0.84.

Finally, the analysis of messages based on the content and characteristics of social network accounts has proven to be a complementary resource in detection tasks, as demonstrated in the works by Madisetty et al. [24] and Chen et al. [25], who developed methodologies based on features of specific social networks. On the other hand, the approach in this work presents a model that relies solely on the message content, independent of the data source, allowing its implementation for the detection and classification of cyberattacks in content from diverse social networks.

6. Conclusions

Nowadays, social networks are being used as attack vectors to distribute malicious content to numerous users. As an alternative countermeasure, this work presented an approach for analyzing social network messages through a series of procedures based on NLP tools and a CNN architecture to detect and classify four types of cyberattacks: malware, phishing, spam, and a defined version of a bot attack.

In this work, a collection of labeled messages was utilized as a corpus, where each element was categorized based on its content as either a cyberattack or considered irrelevant if it did not fit into any of the established cyberattack categories. Thus, this corpus is defined as an imbalanced dataset because the number of messages is different for each class, allowing represent the environment where, by itself, the amount of irrelevant messages is superior to those identified as cyberattacks.

Regarding the initial problem, this approach showed promising results in detecting cyberattacks, or elements that belong to the set C, with an accuracy of 0.91. It also demonstrated a favorable trend in classifying malicious content into one of the four cyberattack categories, or the classification of elements into subsets of C (C_1, C_2, C_3, and C_4) in the second part of the problem, with an accuracy of 0.82.

Finally, the proposed approach showed that it can operate without relying on specific social networks, third-party services, or blacklists that identify users or addresses reported and updated by a service provider. Therefore, this work does not rely on the specific characteristics of particular social networks, and it can be implemented to analyze the content from various social media services.

In future work, we will focus on enhancing the performance of cyberattack classification tasks by considering other message characteristics or integrating other NLP techniques, such as sentiment analysis or vector representations for words like GloVe. The idea behind these enhancements is that they could potentially improve the results achieved in this work. On the other hand, since this model is not dependent on a third-party service, a specific data source, or a platform, future tasks could involve increasing the number of cyberattacks the model can detect. This would require efforts in gathering and categorizing a comprehensive set of messages for the training dataset, based on the different threats to be included in this model.

Author Contributions: These authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: This work was conducted with partial support from the Mexican Government under grant number A1-S-47854 of CONAHCYT, Mexico, and grant numbers 20220852 and 20220859 of the Secretaría de Investigación y Posgrado of the Instituto Politécnico Nacional, Mexico. The authors thank CONAHCYT for the computing resources supplied through the Plataforma de Aprendizaje

Profundo para Tecnologías del Lenguaje of the Laboratorio de Supercómputo of the INAOE, Mexico, and acknowledge the support of Microsoft through the Microsoft Latin America Ph.D. Award.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Abbreviations

The following abbreviations are used in this manuscript:

CNN	convolutional neural network
DL	deep learning
IDF	Inverse Document Frequency
IDS	Intrusion Detection System
IoC	Indicators of Compromise
IoT	Internet of Things
IPS	Intrusion Prevention System
ML	machine learning
NER	Named Entity Recognition
NLP	natural language processing
NN	neural network
POS	Part-of-Speech
RF	Random Forest
RNN	recurrent neural network
TF	Term Frequency
URL	Uniform Resource Locator

References

1. Subrahmanyam, K.; Reich, S.M.; Waechter, N.; Espinoza, G. Online and offline social networks: Use of social networking sites by emerging adults. *J. Appl. Dev. Psychol.* **2008**, *29*, 420–433. [CrossRef]
2. Meana-Llorián, D.; González-García, C.; Pelayo, B.C.; Cueva, J.M. BILROST: Handling actuators of the internet of things through tweets on twitter using a domain-specific language. *Int. J. Interact. Multimed. Artif. Intell.* **2021**, *6*, 133–144. [CrossRef]
3. Kumar, R.; Anand, A.; Kumar, P.; Kumar, R.K. Internet of Things and Social Media: A review of Literature and Validation from Twitter Analytics. In Proceedings of the 2020 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 12–14 March 2020; pp. 158–163.
4. Bendovschi, A. Cyber-Attacks—Trends, Patterns and Security Countermeasures. *Procedia Econ. Financ.* **2015**, *28*, 24–31. [CrossRef]
5. Appiah, V.; Asante, M.; Kofi Nti, I.; Nyarko-Boateng, O. Survey of websites and web application security threats using vulnerability assessment. *J. Comput. Sci.* **2018**, *15*, 1341–1354. [CrossRef]
6. Grover, P.; Kar, A.K.; Janssen, M.; Ilavarasan, P.V. Perceived usefulness, ease of use and user acceptance of blockchain technology for digital transactions–insights from user-generated content on Twitter. *Enterp. Inf. Syst.* **2019**, *13*, 771–800. [CrossRef]
7. Pindado, E.; Barrena, R. Using Twitter to explore consumers' sentiments and their social representations towards new food trends. *Br. Food J.* **2021**, *123*, 1060–1082. [CrossRef]
8. Sukma, E.A.; Hidayanto, A.N.; Pandesenda, A.I.; Yahya, A.N.; Widharto, P.; Rahardja, U. Sentiment Analysis of the New Indonesian Government Policy (Omnibus Law) on Social Media Twitter. In Proceedings of the 2020 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS), Jakarta, Indonesia, 19–20 November 2020; pp. 153–158.
9. Khurniawan, F.S.; Ruldeviyani, Y. Twitter Sentiment Analysis: Case Study on the Revision of the Indonesia's Corruption Eradication Commission (KPK) Law 2019. In Proceedings of the 2020 International Conference on Data Science and Its Applications (ICoDSA), Bandung, Indonesia, 5–6 August 2020; pp. 1–6.
10. Ruan, Y.; Durresi, A.; Alfantoukh, L. Using Twitter trust network for stock market analysis. *Knowl. Based Syst.* **2018**, *145*, 207–218. [CrossRef]
11. Lee, H.S. Exploring the Initial Impact of COVID-19 Sentiment on US Stock Market Using Big Data. *Sustainability* **2020**, *12*, 6648. [CrossRef]
12. Coyac-Torres, J.E.; Sidorov, G.; Anaya-Aguirre, E. Detección de ciberataques a través del análisis de mensajes de redes sociales: Revisión del estado del arte. *Res. Comput. Sci.* **2020**, *149*, 1031–1041.
13. Kunwar, R.S.; Sharma, P. Social media: A new vector for cyber attack. In Proceedings of the 2016 International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Spring), Dehradun, India, 8–9 April 2016; pp. 1–5.

14. Saidi, F.; Trabelsi, Z.; Salah, K.; Ghezala, H.B. Approaches to analyze cyber terrorist communities: Survey and challenges. *Comput. Secur.* **2017**, *66*, 66–80. [CrossRef]
15. Lippmann, R.; Campbell, J.; Weller-Fahy, D.; Mensch, A.; Campbell, W. Finding malicious cyber discussions in social media. *Linc. Lab. J.* **2016**, *22*, 46–59.
16. Grisham, J.; Samtani, S.; Patton, M.; Chen, H. Identifying mobile malware and key threat actors in online hacker forums for proactive cyber threat intelligence. In Proceedings of the 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), Beijing, China, 22–24 July 2017; pp. 13–18.
17. Liao, X.; Yuan, K.; Wang, X.; Li, Z.; Xing, L.; Beyah, R. Acing the IOC Game: Toward Automatic Discovery and Analysis of Open-Source Cyber Threat Intelligence. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 755–766.
18. Iocs, ¿realmente Conocemos sus Capacidades? Available online: https://www.pandasecurity.com/spain/mediacenter/seguridad/iocs-y-sus-capacidades (accessed on 11 October 2022).
19. Liew, S.W.; Sani, N.F.M.; Abdullah, M.T.; Yaakob, R.; Sharum, M.Y. An Effective Security Alert Mechanism for Real-Time Phishing Tweet Detection on Twitter. *Comput. Secur.* **2019**, *83*, 201–207. [CrossRef]
20. Erkal, Y.; Sezgin, M.; Gunduz, S. A New Cyber Security Alert System for Twitter. In Proceedings of the 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 9–11 December 2015; pp. 766–770.
21. Ashour, M.; Salama, C.; El-Kharashi, M.W. Detecting Spam Tweets using Character N-gram Features. In Proceedings of the 2018 13th International Conference on Computer Engineering and Systems (ICCES), Cairo, Egypt, 8–19 December 2018; pp. 190–195.
22. Wu, T.; Liu, S.; Zhang, J.; Xiang, Y. Twitter Spam Detection Based on Deep Learning. *Proc. Australas. Comput. Sci. Week Multiconference* **2018**, *3*, 1–8.
23. Feng, B.; Fu, Q.; Dong, M.; Guo, D.; Li, Q. Multistage and Elastic Spam Detection in Mobile Social Networks through Deep Learning. *IEEE Netw.* **2018**, *32*, 15–21. [CrossRef]
24. Madisetty, S.; Desarkar, M.S. A Neural Network-Based Ensemble Approach for Spam Detection in Twitter. *IEEE Trans. Comput. Soc. Syst.* **2018**, *5*, 973–984. [CrossRef]
25. Chen, W.; Yeo, C.K.; Lau, C.T.; Lee, B.S. A study on real-time low-quality content detection on Twitter from the users' perspective. *PLoS ONE* **2017**, *12*, e0182487. [CrossRef] [PubMed]
26. Djaballah, K.A.; Boukhalfa, K.; Ghalem, Z.; Boukerma, O. A new approach for the detection and analysis of phishing in social networks: The case of Twitter. In Proceedings of the 2020 Seventh International Conference on Social Networks Analysis, Management and Security (SNAMS), Paris, France, 14–16 December 2020; pp. 1–8.
27. Lampu, B. SMS_ Spam_Ham_Prediction. Available online: https://www.kaggle.com/datasets/lampubhutia/email-spam-ham-prediction (accessed on 22 June 2022).
28. Behzadan, V.; Aguirre, C.; Bose, A.; Hsu, W. Corpus and Deep Learning Classifier for Collection of Cyber Threat Indicators in Twitter Stream. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 10–13 December 2018; pp. 5002–5007.
29. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
30. Brownlee, J. *Deep Learning for Natural Language Processing*, 1st ed.; Machine Learning Mastery: Vermont, VIC, Australia, 2017; p. 322.
31. Abadi, M.; Agarwal, A.; Barham, P. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. *arXiv* **2015**, arXiv:1603.04467.