Information Systems 99 (2021) 101740

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is

A real-time deep-learning approach for filtering Arabic low-quality content and accounts on Twitter

Reem Alharthi *, Areej Alhothali, Kawthar Moria

Department of Computer Science, King Abdulaziz University, Jeddah, Kingdom of Saudi Arabia

ARTICLE INFO

Article history:
Received 16 September 2020
Received in revised form 2 February 2021
Accepted 7 February 2021
Available online 12 February 2021
Recommended by Senjuti Basu Roy

Keywords:
Low-quality content in social networks
Spam accounts
Real-time detection system
Deep learning techniques

ABSTRACT

Social networks have generated immense amounts of data that have been successfully utilized for research and business purposes. The approachability and immediacy of social media have also allowed ill-intentioned users to perform several harmful activities that include spamming, promoting, and phishing. These activities generate massive amounts of low-quality content that often exhibits duplicate, automated, inappropriate, or irrelevant content that subsequently affects users' satisfaction and imposes a significant challenge for other social media-based systems. Several real-time systems were developed to tackle this problem by focusing on filtering a specific kind of low-quality content. In this paper, we present a fine-grained real-time classification approach to identify several types of low-quality tweets (i.e., phishing, promoting, and spam tweets) written in Arabic. The system automatically extracts textual features using deep learning techniques without relying on hand-crafted features that are often time-consuming to be obtained and are tailored for a single type of low-quality content. This paper also proposes a lightweight model that utilizes a subset of the textual features to identify spamming Twitter accounts in a real-time setting. The proposed methods are evaluated on a real-world dataset (40, 000 tweets and 1, 000 accounts), showing superior performance in both models with accuracy and F1-scores of 0.98. The proposed system classifies a tweet in less than five milliseconds and an account in less than a second.

1. Introduction

Today many people consider social networks a fundamental source of information and valuable outlets for expressing their views. Social media platforms allow users to easily and instantly share and acquire information with no restrictions placed on the account or the shared content. The immediacy and accessibility of social networks have greatly impacted the process of sharing and acquiring information. Twitter has become now the most popular microblogging site, and one of the most preferred medium for sharing viral news and expressing opinions on the world's current events (e.g., policy announcements or business services) [1]. Researchers and other organizations (i.e., profit and non-profit) have also exploited Twitter's public data for numerous notable purposes, such as monitoring customer satisfaction, detecting traffic congestion, and tracking public health [2,3]. Ill-intentioned individuals, on the other hand, leverage these services to fulfill a variety of harmful objectives. At little to no cost, they can target a large number of unaware genuine users with their harmful content. The form and nature of these malicious content differ according to the writers' or spammers' motives. *Phishing*, for example, is a malicious activity in which the attacker attempts to steal financial or confidential information using different means, such as hyperlinks (URLs), phone numbers, or direct messages. The attackers typically follow various tactics to trick their victims, such as using fake websites of Twitter or banks' login page [4]. *Promoting* is another common malicious practice in which individuals or groups utilize social media to bring attention to some services, or products by posting a massive number of duplicate and unsolicited tweets in trending topics. The prompted text is often related to illegal services or businesses, such as illicit or unlicensed drugs, pornography content, or fake followers. Twitter considers promoting without prior consent an illegal activity, even for legitimate organizations or products [5]. *Social spam* is also another popular malicious activity in which spammers circulate the URL to a website that includes malicious software. These malicious practices have numerous negative impacts on social media platforms, users, and researchers. For users, in particular, going through unwanted and irrelevant tweets can negatively affect their satisfaction. These activities also pose risks to users' privacy and security when following external harmful URL links. Moreover, such activities generate a significant amount of low-quality content that negatively affect researchers and other Twitter-dependent applications. It is worth mentioning that researchers use the word "spam" collectively to refer to

* Corresponding author.
E-mail address: ralharthi0111@stu.kau.edu.sa (R. Alharthi).

https://doi.org/10.1016/j.is.2021.101740
0306-4379/

social media practices that are associated with malicious URLs [6]. In this research, we use the term low-quality to describe the different forms of malicious content that might or might not comprise URLs.

Identifying potential malicious content in a timely manner is critical to protect users' digital safety and social network resources. Search engines and Twitter-dependent applications which require high-quality data will also benefit from detecting malicious activities in real-time. Therefore, several researchers have proposed various real-time systems and frameworks which identify malicious content using machine learning techniques. All of the existing methods, however, possess two limitations. *First*, all previous work targets a specific kind of low-quality content, such as tweets with malicious URLs to spread malware [7], scam or phishing content to steal genuine users' personal information [8], or promoting content to point toward a particular product, service, or website [9]. *Second*, they have mostly been built using handcrafted features that can be easily manipulated by spammers. For example, classifiers that examine URL links to identify spamming content will fail to detect spams that include fabricated landing pages to deceive the detection systems' webcrawlers [10]. Spam tweets also might indirectly import or share spamming URLs using the Quote function, which will hinder the ability to detect this type of content. Most existing systems also acquire data using crawling tools and process high dimensional features making related classification tasks such as event identification, tweets summarization, and sentiment analysis hard to be accomplished in real-time. These limitations decrease detection capability because attackers can develop their strategies over time and find a new medium, such as phone numbers, to disseminate their content [11].

In this paper, we propose a real-time deep learning approach to detect low-quality tweets as well as accounts that produce such content on Twitter. The proposed system consists of two models operating in a real-time setting: the first model classifies tweets of any chosen topic (i.e., hashtag or terms) into either genuine or different types of low-quality content, and the second classifies any given account as either malicious or genuine based on its recent tweets. In the *tweet-level model*, a deep learning technique (CNN-LSTM) is used to identify a variety of Twitter low-quality content, namely: promotion, phishing, and spamming. Unlike most of the proposed deep learning models in the field, the tweet-level model is built to differentiate between the deep semantics associated with the different categories of low-quality contents and genuine contents. The tweet-based model was trained and tested on a dataset of 40, 000 tweets collected for this study and achieved high performance with accuracy and F1 scores of 0.98. As for the *account-level model*, this paper proposes a deep learning approach that utilizes the users' recent tweets to distinguish genuine from spamming users (i.e., accounts posting low-quality contents) in real-time. The proposed account-based model is tested on 1000 users and yielded accuracy and F1 scores of 0.98. Additionally, two embeddings methods (word- and character-level) are examined to represent the textual features in both Tweet-model and Account-model. The word-level embedding applies a pre-trained continuous bag of words (CBOW), a neural network model, to map each word to a representative vector. The character-level model encodes each character in a word into a one-hot encoding vector.

It is worth noting that this research is the first attempt to classify Twitter accounts in real-time when accounts post tweets on any topic and the first attempt that uses deep learning techniques to identify spammer (all types of low-quality) users according to their tweets textual information only. The proposed models in this study differ from other spam detection systems as they are trained on an Arabic language dataset, while most of the proposed spam detection systems are trained on English tweets with less effort placed on the detection of non-English spam tweets. The dataset is real-world ground-truth data collected by the authors of the paper, which is also publicly available to the researcher community.[1] The proposed real-time deep learning system is also designed to be a lightweight tool that can be easily integrated into other systems requiring real-time filtering of low-quality content. The proposed method is practical and less complicated than existing real-time detection systems since it only requires the textual information of the tweets to obtain an accurate classification. In contrast, most of the current solutions require processing a large vector of features (e.g., sentiment, timing, and URL features). The proposed systems have shown a competitive performance in terms of detection accuracy and computational time for the following two reasons:

- Unlike other spam detection systems, the classification process does not require the retrieval of the tweets associated URL, which makes it easy to avoid using data that might not be available or may be easily fabricated by attackers. Besides, by employing deep learning methods to extract the textual features, we avoid using statistical features that can also be easily manipulated by attackers [9].
- Our system has a lower computational cost and processing time than other systems, which takes five milliseconds to classify a tweet and less than a second to classify a single account. The reduction in computational time is due to crawling less data than other approaches (it only streams Twitter API for real-time tweets) and computes fewer features (the text of tweets is the main source of features).

The remainder of this paper is organized as follows: Section 2 outlines previous related works, and Section 3 discusses the dataset collection process. Section 4 introduces the proposed system design in detail. Section 5 presents the performance results of the proposed model in real-time. The limitations of the suggested approach are addressed in Section 6. Finally, the conclusion of this paper and some recommendations for future work are summarized in Section 7.

2. Related works

In this section, we review important and recent work-related to the detection of social media spam and low-quality content. The related works are divided into three sub-categories: Twitter spam detection, deep learning spam detection models, and real-time spam detection systems. The first sub-category gives details about detection systems and state-of-the-art methods developed to detect spam tweets and spam accounts. The second sub-category describes the recent works that have adopted deep learning models to detect spam content on Twitter. The last sub-category reviews systems that have attempted to identify spam content soon after it has been shared (i.e., real-time).

2.1. Twitter spam detection

- **Tweet-level:** The vast majority of work-related to spam tweet detection has adopted machine learning algorithms that utilize statistical features. These features can be related to the tweet itself or the account which posted the tweet. Wang et al. [12] evaluated several features categorized into four categories: user features, content features, n-grams, and sentiment features in detecting spam content. The evaluation results showed that user-based features were the

[1] https://github.com/ReemAlharthi/Arabic-Low-Quality-Tweets-and-Accounts-Dataset

most discriminative set of features, while the sentiment and content-based features were the most time-consuming features. Several studies have also employed similar user-based and content-based features [13,14]. Another research studied the behavioral patterns of accounts that share URLs and those who follow the shared links [15]. They obtained several features that describe these patterns, such as the number of times a particular URL was posted and the total number of clicks. Over time, the problem of spam drift arises as spam tweets tend to have different statistical properties than the used dataset. This issue has a serious impact on the reliability of feature-based machine learning detection approaches. In order to train models over time, authors in [16] proposed a fuzzy-based re-distribution technique to generate spam tweets from a limited sample of spam tweets. According to their results, the fuzzy-based approach has improved the identification efficiency of drifting spam tweets.

Liu et al. [17] studied the class imbalance problem of spam and non-spam tweets and its effect on the efficiency of machine learning-based spam detection methods. They found that as non-spam tweets significantly out-number spam tweets in the dataset, the efficacy of the machine learning methods degrades, but it also contributes to a higher true positive rate. Accordingly, they proposed an ensemble learning approach that incorporates the classification of multiple models trained on the same dataset but with different distributions of spam and non-spam tweets. Li et al. [18] also addressed the class imbalance problem by examining different data sampling or redistribution algorithms such as ROS, RUS, and BSM [19]. Based on their observations, the fuzzy-based sampling system outperformed the other data sampling methods and contributed to an increase in Twitter spam identification accuracy with imbalanced data.

- **Account-level:** Several techniques also have been developed to detect spammer accounts on Twitter. Many approaches adopted supervised machine learning algorithms that are trained on statistical data from Twitter accounts to identify spammers' accounts [20–22]. These studies exploited the spammers' behavioral data (e.g., their posting rate) and other information related to their tweet content to identify spamming accounts [23].

2.2. Deep learning detection models

Several recent studies have adopted deep learning techniques to handle raw text in different domains, e.g., posts, tweets, reviews, and SMS messages. Wu et al. [24] trained a word2vec model using the syntax of their dataset, and based on the representative dataset, they trained a Doc2vec model that maps each tweet to a representative multidimensional vector. The resultant vectors were used as descriptive features in a machine learning algorithm (i.e., multilayer perceptron MLP) to detect spam. Jain et al. [25] employed a deep learning architecture based on a semantic convolutional neural network (SCNN) to obtain a more representative vector of the tweets. The semantic layer included an embedding matrix that held the dense vector representation for each word in their dataset. These vectors were obtained using an available pre-trained word embeddings model [26]. The semantic layer is followed by a convolutional neural network (CNN) layer that consisted of 64 filters. The intuition behind this approach relied on selecting or extracting the most significant features or words in the given text. Finally, a softmax function classifies the features map produced by the convolutional layer into a set of predefined classes. The two approaches use different machine learning approaches, the first approach uses a semantic long short term memory (SLSTM) model [27], and the second adopted a convolutional long short term memory deep learning model (CLSTM) [28]. The models' performance evaluation shows that both models achieved excellent performance with 0.94 to 0.98 accuracy, respectively.

In a similar vein, Madisetty et al. [29] presented an ensemble approach to combine CNN and multiple word embedding models, e.g., word2vec, and Glove. In addition to the CNN models, a feature-based model is used to extract a set of user and content-based features. Eventually, five CNN models and the feature-based model were combined with a neural network-based meta-classifier to classify tweets. Madisetty et al.'s approach achieved 0.92 precision, 0.86 recall, and 0.89 F1-measure. Anand et al. [30] proposed a deep learning model to detect Clickbait (i.e., catchy headlines for a URL that leads to unwanted pages that are often used for phishing or promoting purposes). The deep model consists of three layers: the embedding layer (character level word embedding), the hidden layer (Bi-Directional RNN), and the output layer (Sigmoid neuron). Deep learning techniques generally gave promising results in spam detection [31]. Nevertheless, current deep learning-based spam detection approaches did not clearly present the diversity of spam tweets in their dataset, focusing only on single spamming activity. In contrast, the collected dataset in this paper has several kinds of low-quality tweets (i.e., phishing, spam, and promotion) that used different means to deliver their content (i.e., URLs, photos, and phone numbers). Here, the goal of this dataset is to train a deep learning model in identifying a wide range of low-quality tweets.

2.3. Real-time spam detection systems

Detecting spam or harmful content shortly after being posted before negatively affecting genuine users is very important. This led several researchers to investigate real-time systems in identifying spam tweets to prevent any potential threats. Thomas et al. [32] developed a real-time system, namely, Monarch, that inspects tweets that comprise URLs of harmful content. The system first crawls and accumulates all the target URL data, transfers it into a sparse vector of features, and classifies it into spam or non-spam content. One of the drawbacks of the proposed system is that it extracts a considerable number of features of each URL, such as URL features for every outgoing network request, including scripts, redirects, and embedded content. This high dimensional feature space may hinder the spam detection process in real-time. Also, the authors noted that most of the features used might not be available during the crawling process [33]. Aggarwal et al. [8] developed a browser extension to mark phishing tweets in real-time while users are browsing Twitter. Four feature categories (i.e., URL, WHOIs, Tweet, and Network-based features) and a total number of twenty-two features need to be extracted from a given tweet. Three machine learning classifiers (i.e., Naive Bayes, Decision Tree, and Random Forest) were trained and tested on these features, and the Random Forest classifier gave the best performance with an accuracy of 0.92. Nevertheless, Lee et al. [10] developed a model to detect more challenging spams that are not identifiable by traditional methods that rely on crawling URLs data. These challenging tweets are generated using a new evasion technique that bypasses existing detection systems by redirecting crawlers to benign websites. Based on that and given the fact that attackers have limited resources, they proposed a spam-detection approach that clusters correlated redirected chains into one group. Then, it calculates several attributes (e.g., URL redirect chain length and a number of other sources) to classify the URLs group.

Martinez-Romo et al. [34] presented a real-time spam detection system that monitors Twitter trending topics to identify

malicious content. The system is built based on the assumption that spammers share malicious tweets and links that have no semantic correlation with the target topic. Accordingly, the system assesses the divergence between the language models of 20 reliable tweets and each suspicious tweet in a topic. Burnup et al. [7] developed a machine learning model that classified URLs according to the URL's interaction data with the client honeypot (Capture-HPC) when the link was clicked [35]. Additionally, Gupta et al. [36] utilized several user and tweet based-features for example, account age, number of followers, number of accounts followed, and number of tweets, along with the bag of word representations of the textual data to distinguish between spam tweets and benign tweets. Lastly, Chen et al. [6] provided an extensive review of low-quality content from social network users' perspective. According to their analysis results, low-quality content such as promoting content has not been sufficiently studied yet, while it is the most disturbing content for genuine users. The authors also built a real-time system that extracts 32 features of two categories, namely, direct features that can be easily obtained and computed and indirect features that require more time to be extracted but are more prominent in classifying the low-quality tweets. Generally speaking, most of the real-time systems rely on heavy features engineering methods that are time and resources consuming [33]. Besides, extracting and processing these features in real-time has a significant impact on the computational time, especially in systems that crawl the web data.

3. Datasets collection

To train the proposed models in this research, we first built the (tweet-level) dataset using Twitter API functions that pull or retrieve Twitter public information such as tweet content. The major goal of this phase is to collect a wide variety of low-quality and genuine tweets, including short- or long-text tweets, as well as tweets written on Modern Standard Arabic (MSA) or varieties of Arabic (we focus on Arabian Peninsula dialect). For tweets of low quality, three forms or types were obtained: phishing tweets, promotional, and spam tweets. Phishing tweets were collected according to the description of several verified accounts that alerted individuals to the danger of some scam links. Such links are circulated via fake Twitter accounts and led to a fraudulent, unauthorized stock trading platform.[2] Thus, tweets that advertise trading through this platform are considered phishing tweets and represent 1, 238 tweets of the dataset. The second type is promotional tweeting, which is the most common and varied form among the different types of low-quality content. This category covers a wide variety of goods and services like promoting cleaning companies, coupon codes for online shopping sites, medications, apparel, and food items. These contents are often shared by short-lived accounts that post vast amounts of repeated tweets and target trending topics. Therefore, in our dataset, promotional tweets are obtained by streaming and filtering trending Twitter topics. In this way, tweets featuring some promotional content shared by the accounts mentioned above are considered a promotional tweet. These tweets constitute a large proportion of the total low-quality tweets in our dataset ($\approx$ 51%,around 10, 321 tweets).

The last form of low-quality content is the spam tweets that focus mainly on promoting external websites. Usually, such tweets use catchy headlines or descriptions of the included URL, such as exclusive news and videos, to increase the click-rate to these sites. These links often lead to websites that are either marked as non-safe (according to Google Safe Browsing) or intrusively download unknown material at the same time as the sites are loaded. Therefore, tweets collected by filtering URL tweets from trendy issues and leading to these threatening websites are labeled as spam tweets in our dataset. Spam tweets represent about 0.42% (8, 441 tweets) of the total low-quality tweets in our dataset. Non-spam tweets were also collected from accounts marked as trusted or verified by Twitter and represent 20, 000 tweets. The tweet-level dataset consisted of a total of 40, 000 tweets that were divided equally between genuine and low-quality tweets. The summary of the dataset class distributions are given in Table 1. Table 2 shows examples of the genuine and low-quality tweets in the dataset and their translation in English.

Table 1
A summary showing class distributions of the annotated dataset.

Category	Fine-grained class	Number of Tweets
Low-quality content	Phishing Tweets	1, 238
	Promotion Tweets	10, 321
	Spam Tweets	8, 441
Total of Low-quality		20, 000
Genuine Content	Genuine Tweets	20, 000
Total of Tweets		40, 000

For the accounts-level dataset, suspended accounts collected in the tweet-level dataset are considered to be equivalent to spam profiles. Various kinds of spam account exist in our dataset and included spammers who attempted to drive traffic or attention to a website, products, or services. The dataset ultimately includes 500 spam accounts. Several genuine accounts (500 verified accounts) were added to the dataset to obtain a balanced dataset. We also ensured that the genuine class contained accounts that discuss one topic, such as world events, sport news, or personal accounts. The purpose here is to ensure that our dataset contains different types of genuine accounts, specifically genuine accounts that could have a considerable self-similarity that resembles spam accounts relatively.

4. Methodology

We have adopted deep learning techniques to detect low-quality tweets and spam accounts in real-time settings. The models first stream data using the Twitter API [37] to collect tweets of a specific topic (e.g., hashtag or term) or accounts. In the Tweet-level model, the raw text of the tweets is then pre-processed by removing non-word symbols (e.g., images, URLs, mentions, and emojis). The modeling process then maps the text from its unstructured form to a numerical dense vector representation. Text modeling techniques can largely influence the deep model accuracy and overall system performance by improving model predictivity and reducing the complexity required to process these features. Accordingly, we evaluated the system performance (speed) and the model accuracy using different representation methods, namely word embedding, and character embedding techniques (additional details can be found in Section 4.2). In the last phase of the Tweet-model, a deep learning model, which was trained on 40, 000 tweets, is used to classify the given tweet as either spam or non-spam class.

The Account-level involves similar phases to the tweet-level. After streaming real-time tweets, the model extracts the author identifier of each tweet, which is then used to collect the recent tweets of that account. The textual features of retrieved tweets are then prepared for the cleaning and modeling processes (additional details about the preparation method are covered in Section 4.4). The last phase in the Account-model, a deep model (Account-model), which was trained on a dataset of 1000 accounts, is then used to classify Twitter accounts as spam or genuine profile. The ultimate goals of this work are to reduce the

[2] https://twitter.com/SAMA_GOV/status/1275005612208926723

Table 2
An example of each type of low-quality tweets in the collected dataset.

Tweet type	Tweet translation in English	Tweet
Phishing Tweet	AvaTrade is a well-established Forex / CFD brokerage firm that was established in 2006. Do not miss the opportunity to find out everything about the company and start trading with it	
Promotion Tweets	Would you like to have a thick beard? And are you thinking what is the solution? Use the American Beard Thickening Kit from Forever Company, the best group to help you get a thick, full beard without spaces. Its benefits: Beard thickening Fill the blanks of the hair on the beard Stimulate hair follicles to grow rapidly	
Spam Tweets	Hurry now to download the iPhone update exclusively Additionally: Download #Snap Plus Snap Plus for Android and iPhone Snapchat Plus without jailbreak, the latest Snapchat Update the new Snap with the latest and newest filters Link to update the iPhone https://is.gd/Q1OvCd Download link https://taempro.net/archives/80#Update_iPhone	

computational time and complexity required to classify tweets in real-time and develop a real-time system that can accurately classify various types of low-quality tweets. Additionally, to build a system that can identify the spam accounts in a particular topic around the time spam content is posted. The following subsections give more details of the methodology stages. Section 4.1 provides some details about the data preprocessing methods in this work. Section 4.2 introduces in detail the proposed plans to extract and represent textual features. Section 4.3 gives a detailed description of the Tweet-level deep learning model architecture that is used to classify tweets. Section 4.4 covers all the account-level model details, including the proposed approach to obtain the account's textual information.

4.1. Data preprocessing

Choosing the appropriate text preprocessing methodology is essential for any natural language processing classification task and more critical in real-time detection systems. Preprocessing methods, such as correcting misspelled words or normalizing text, may increase the classification accuracy but might increase model computational complexity and processing time. Therefore, in this proposed system, many essential preprocessing techniques have been considered in the data cleaning phase. To prepare the tweets for the training or testing stages in the real-time setting, we performed the following preprocessing methods: (a)

Remove the nonlinguistic features, e.g., image, video, punctuation, mention, retweet signs, and emoji. (b) replace URLs and numbers with a representative word (i.e., link, num). (c) Normalize Arabic text by unifying the orthography of alif, hamzah, and alif maqsurah, as well as removing Arabic diacritics such as fatha and tanwin.

4.2. Feature extraction

Textual data can be converted into a numerical format using multiple existing vectorization methods. Standard vectorization methods are the bag-of-words (BoW), term frequency–inverse document frequency(TF–IDF), and neural network embedding methods (word2vec and Glove). BoW representation has several disadvantages, including ignoring words' order, requiring computational and storage resources to handle vectors sparsity, and lacking the ability to capture semantic meaning. The TF–IDF model is more powerful than the bag-of-word model, but as previous studies have indicated, it requires a long time to be procured [14]. Neural network embedding models have received considerable attention in the latest years, showing superior performance in many classification tasks.

In this work, we focus on training and evaluating the proposed deep learning model using two text embedding methods, namely word embedding and character embedding. For simplification, the word embedding-based deep learning model is called the Word-level model, and the character embedding-based deep learning model is called the character-level model. A detailed description of the word-level model and character-level model are given in the following two subsections.

- Word-level Model As previously mentioned, textual data must be represented with some numerical values that capture word semantic meaning. Hence at the word embedding level, the text of the tweets is first tokenized into chunks of words. Each word is then represented as a dense 100-dimensional vector of numerical values obtained from AraVec pertained word2vec model [38]. Eventually, an embedding matrix is used to import the words' vectors to the embedding layer, which is the first layer in the Word-level deep model. AraVec model [38] was trained using the CBOW approach on an Arabic Twitter dataset that contained over 1, 476, 000 words. Even with this large number of words, we still encountered instances where words do not exist. For 68, 651 unique words, there are 62, 745 matching words in the model vocabulary, while 5, 906 words were not available in the vocabulary. Non-Arabic words written in Arabic, misspelled, repeated characters' words are also common examples of the kinds of words that were not found in the pre-trained model.
- Character-level Model Tweets are an example of user-generated data that contain noise, misspellings, infrequent words, and slang words, which might not be found in the pre-trained word-embedding vocabulary. To overcome this problem, a character-level convolutional network [39] was proposed to represents each character of words with a representative digit. The character embedding model encoded each character in a predefined set of characters as a one-hot encoding vector. The model then learned the vector of a word using six 1D CNN layers, which convolve over the sequence of the word's characters. Compared with the word-level embedding, the character-level requires fewer

pre-processing steps where there is no need to correct misspelled words. This research adopts the same approach where the alphabet set is first defined based on our dataset. Using one-hot encoding, the text of the tweets is converted into a vector representation as a matrix of 280 × 138 (tweet length* alphabet set size). Two 1D CNN layers are then used to learn the characters' sequence. A different number of layers were assessed, and the performance of the two layers was found to be similar to three and above layers. A max-pooling layer, an LSTM layer, and fully connected dense layers followed the CNN layer to obtain the classification decision.

4.3. Tweet-level model

Convolutional neural networks (CNN) have proven to be effective in many NLP tasks, including text classification, sentiment analysis, and part-of-speech tagging. CNN consists of several filters with a specific kernel size, which then convolves over the target text to select or generate the feature maps. CNN layers work as a feature selector that captures local patterns in a given text. The long short-term memory (LSTM), on the other hand, learns long-term dependencies in sequential data, for example, text and time-series data. In this work, the deep learning model architecture inspired by [40] consists of a CNN layer followed by an LSTM layer, as shown in Fig. 1. The first layer of the model comprises an embedding layer that holds the words' vectors in the case of the word-level model or the one-hot vector encoding of the words' characters in the character-level model. A CNN layer then follows to generate the feature maps, and a max-pooling layer is then used to reduce the dimensionality of the CNN output. The max-pooling layer's output is then fed to the LSTM layer, which is connected to a dense layer followed by a sigmoid function to classify the tweets as a genuine or low-quality class. As previously mentioned, the system performance is evaluated on both word-level and character-level using the standard classification metrics, namely F1 score, precision, recall, accuracy, and receiver operating characteristics (ROC).

4.4. Account-level model

To identify spam accounts, we developed a lightweight deep-learning approach that utilizes the tweets' textual features only. Given a set of i accounts $X = x1, x2, ..., x_i$, in which each account has n tweets. The proposed approach, first, construct the profile p of the x_i account is as:

$$X_i^p = t_1 \oplus t_2 \oplus \cdots \oplus t_n, \quad (1)$$

where the symbol $\oplus$ is the concatenation operator, and t_n is the number of tweets in the account. The n tweets were defined as the maximum number of tweets in one profile and zero-pad profiles, which have less than the maximum length. The account profile x_i^p is then treated as a document or a single text for the cleaning process and the classification task. Suppose that f denotes the deep learning model and y is the predicted class, then the class of account x_i can be obtained as follows:

$$y = f(x_i^p). \quad (2)$$

As previously noted in Section 2, most previous work that attempted to classify Twitter accounts has relied on intensive feature engineering, a time-consuming and error-prone process. Furthermore, these features are based on previous views or expectations of the spam accounts, which might be less accurate or not generalizable to other datasets. This study shows that classifying Twitter accounts can be easily and accurately performed using deep learning techniques and a sample of the accounts' tweets. Our experiment found that Twitter's accounts can be classified accurately based on 15 tweets to 10 tweets (more detail can be found in Section 5.2). To construct accounts' textual features, only a single call is required to retrieve a given account's latest tweets from Twitter API. Like the tweet-model, we have tested the account models with two embedding methods, word-level and character-level. The deep learning model (denoted by account-model) takes the word vectors in the case of word-level or the one hot-coding of characters in the character-level as input and classify it as spam or non-spam profile. The account-models (account-word and account-character) have a similar architecture to the deep model in the tweet-level model, with only one variation in the input data size, which was defined based on the maximum length of the tweets over all the samples of tweets in the dataset.

5. Experiments and results

This section discusses the proposed system performance in a number of experiments conducted on the collected dataset with different word representations and the model of deep learning. The performance of deep models in differentiating low-quality tweets from legitimate tweets is discussed in Section 5.1, while Section 5.2 presents the performance of the account-level model.

5.1. Tweet-level models results

This section first discusses the methods used for optimizing the word-level and the character-level models' hyper-parameters, as well as, the results of the models' evaluation on the collected dataset. The section also addresses the Tweet-level model's performance, including the accuracy and computational time to classify a tweet.

To achieve the best performance with the deep models, we conducted various hyperparameter optimization experiments. There are two types of hyperparameters: the optimizer's hyperparameters, and model-specific hyperparameters. Optimizer's hyperparameters include the parameters that are involved in the learning process, e.g., learning rate, batch size, and epochs. The Adam optimizer [41] is used to compute adaptive learning rates for the weights and parameters of the deep learning model. The two models achieved better performance when the batch sizes and the number of epochs are set to 40 and 10.

Model-specific hyperparameters are the parameters related to the structure of the deep learning model (e.g., number of layers, number, and type of hidden nodes). The deep learning model consists of four layers: a convolutional layer, a max-pooling layer, the LSTM layer, and a dense layer that is followed by a sigmoid function, as shown in Fig. 1. In the convolutional layer, two parameters are optimized, which are the size of the kernel and the number of filters. The kernel size is defined as the width × height of the kernel matrix that convolves with the input text. We evaluated the models' performance using four kernel sizes: 1, 3, 5, and 7 (see Table 3 for the results). The number of filters indicates the number of features or patterns that we attempted to detect in the input data. For both models, Table 3 indicates that the more features are used by increasing the kernel size, the higher performance is obtained. The models' performance was also evaluated using four values for the number of filters, which are 16,32,46, and 128. Similarly, the larger the number of filters used, the better output is obtained, as shown in Table 4.

In the LSTM layer, the number of units that define the LSTM's hidden state's size is optimized. The LSTM layer was first eliminated from the structure of the two models. The character-level model was found to achieve better performance without this

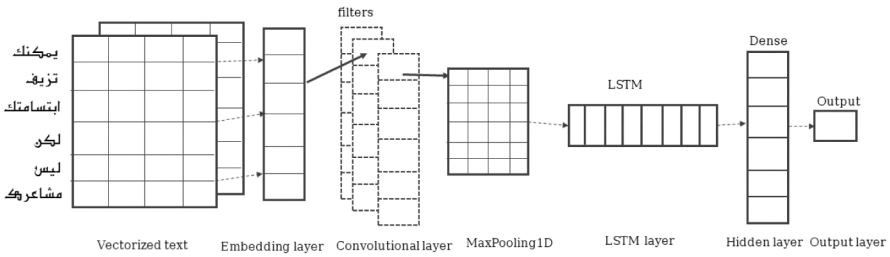

Fig. 1. The CNN-LSTM deep model architecture.

Table 3
The effect of the size of the CNN kernel.

Model	Performance measure	Kernel size			
		1	3	5	7
Word embedding level	F1 score	0.97	0.97	0.98	0.97
	Precision	0.96	0.97	0.98	0.96
	Recall	0.98	0.97	0.97	0.98
Character embedding level	F1 score	0.93	0.98	0.97	0.98
	Precision	0.97	0.98	0.98	0.98
	Recall	0.9	0.98	0.97	0.98

Table 4
The effect of the number of filters on the CNN layer.

Model	Performance measure	CNN filters			
		16	32	64	128
Word embedding level	F1 score	0.97	0.97	0.97	0.98
	Precision	0.98	0.97	0.98	0.98
	Recall	0.96	0.97	0.97	0.97
Character embedding level	F1 score	0.97	0.98	0.98	0.98
	Precision	0.98	0.99	0.99	0.98
	Recall	0.97	0.97	0.97	0.98

Table 5
The effect of the number of LSTM units on the models' performance.

Model	Performance measure	LSTM unit			
		0	25	50	100
Word embedding level	F1 score	0.96	0.97	0.97	0.98
	Precision	0.95	0.97	0.98	0.98
	Recall	0.97	0.97	0.97	0.97
Character embedding level	F1 score	0.98	0.98	0.97	0.98
	Precision	0.97	0.99	0.97	0.99
	Recall	0.98	0.97	0.98	0.97

Table 6
The effect of the number of neurons on the dense layer.

Model	Performance measure	Neurons			
		0	10	15	25
Word embedding level	F1 score	0.97	0.97	0.97	0.98
	Precision	0.97	0.98	0.98	0.99
	Recall	0.97	0.97	0.97	0.97
Character embedding level	F1 score	0.98	0.98	0.97	0.98
	Precision	0.96	0.98	0.97	0.99
	Recall	0.97	0.98	0.98	0.97

Table 7
The effect of the dropout rate on the models' performance.

Model	Performance measure	Dropout rate			
		0.2	0.5	0.7	0.9
Word embedding level	F1 score	0.97	0.97	0.97	0.97
	Precision	0.97	0.96	0.99	0.97
	Recall	0.97	0.97	0.96	0.97
Character embedding level	F1 score	0.98	0.98	0.98	0.98
	Precision	0.98	0.99	0.99	0.99
	Recall	0.98	0.97	0.97	0.97

layer, while the word-level model performs worse. Another number of hidden units, e.g., 25, 50, and 100 units, are also examined (see Table 5). As seen in Table 5, the increase in the number of LSTM units has contributed to improved results, but also to a rise in training time. In the fully connected/dense layers, the number of neurons in the dense layer that links the input of the previous layer to the sigmoid /classification layer is also investigated (see Table 6).

Overfitting occurs when a model performs very well on the training data and fails to classify new inputs successfully. Having small training examples with sparse feature space, having an approximation of a higher degree, or training with too many epochs can cause overfitting in machine learning and deep learning models. Deep learning models are often prone to overfitting due to the deep structure and approximation complexity of their model. Dropout [42] is one of the regularization techniques that is widely used with deep learning techniques. Thus, in this study, we employed the dropout technique to prevent model overfitting. The two models' performance was tested using different dropout ratios (i.e., the percentage of nodes that are excluded from contributing to the output). For the Word-level model, we found that the best performance was achieved with a dropout rate of 0.7, while the character model accomplishes excellent performance with a dropout rate of over 0.5 (see Table 7).

After tuning the models' parameters, the Word-level and Character-level models were trained on the collected dataset, which was divided into a validation set (30%) and a training set (70%). The two models were first trained with a binary classification objective (only two classes, genuine and low-quality). Figs. 2 and 3 show the accuracy and loss curves for the training and validation sets for both models. We can say that the Character-level model has a good fit where there is a small generalization gap between the training and validation for both the loss and accuracy curves. In comparison, the Word-level model was found to be less able to incorrectly classify the validation set, although it still achieved an accuracy of over 0.97. Secondly, the Word-level and Character-level models were trained on the collected dataset with a multi-class objective (four classes: genuine, phishing, promotion, and spam tweets). The Word-level model achieved 0.98 accuracy on the training examples and 0.96 classification accuracy on the validation set.

On the other hand, the Character-level model achieved 0.90 accuracy on the training examples and 0.89 classification accuracy on the validation set. To compare our results, we select the study's by Chen et al. [6] as they also addressed the problem of detecting low-quality content on Twitter. On a dataset of 100, 000 tweets, their method achieved 0.97 accuracy, 0.0075 false-positive rate, and 0.83 F1-score. Their solution focuses on the extraction of lightweight attributes that did not include the text of the tweet. Our approach relies primarily on the text of the tweet and has obtained superior performance, as seen in previous tables.

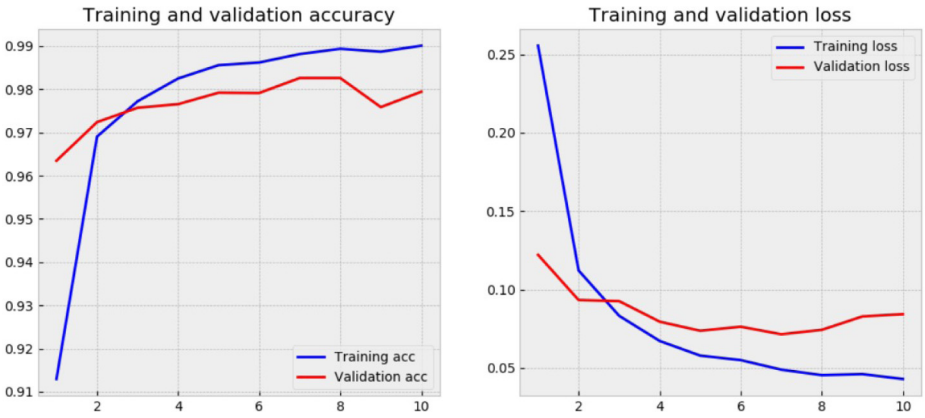

Fig. 2. The accuracy and loss of the word-level model.

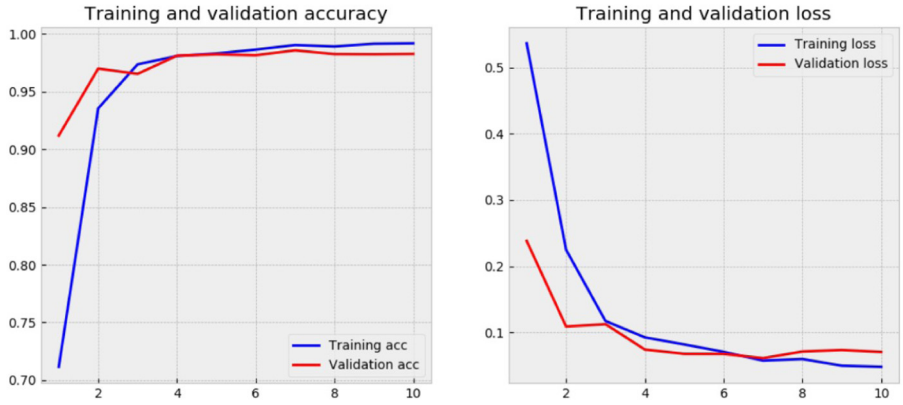

Fig. 3. The accuracy and loss of the character-level model.

Table 8
Comparing the time (in seconds) required by our system (the word-level and character-level models) and previous systems.

System	Speed	System setup
(Burnap, 2015) [7]	30–60	N/A
(Thomas, 2011) [32]	30	Fedora Core 13/4 core 2.8 GHz Xeon processor/8 GB RAM
(Lee, 2012) [10]	2.42	Two Intel Quad Core Xeon CPUs/ 24 GB RAM
(Aggarwal, 2012) [8]	0.522	Intel Xeon 16 core Ubuntu server 2.67 GHz
Word-level model	0.0050	Intel Core i7 CPU/1.88 GHz/16.0 GB RAM
Character-level model	0.0043	Intel Core i7 CPU/ 1.88 GHz/ 16.0 GB RAM

To evaluate the real-time system's performance, we assess the time required by the system to classify a given tweet into spam or a genuine tweet. To achieve real-time classification, the proposed system (the tweet-level) was built based on the queue mechanism, where the first incoming n tweets are the first to be processed and classified by the system. As previously mentioned, the Tweet-level model consisted of three segments: tweet collection, pre-processing, modeling or vectorization, and finally, classification. After the first step of the model collected the n tweets, the second step (pre-processing) began, etc. This approach facilitates handling large volumes of tweets, which is especially useful for heavily discussed topics where processing tweets one at a time is inefficient. In the future, we plan to parallelize these tasks to achieve instantaneous classification. We defined the running time (computational time) to classify a tweet as the interval between adding a tweet to the queue and classifying it. The results show that our system required less time to classify tweets in real-time than other systems (see Table 8). The results show that for 500 tweets, the average time required by the word-level model is 0.0050 s (sec) with a median time of 0.0041 s and a minimum time of 0.0033 s The character-level model is even faster than the word-level model with an average time of 0.0043 s, a median time of 0.0037 s, and a minimum time of 0.0024 s.

During the training stage, we observed that the word-level model required less training time compared to the character-level model. This was an expected outcome since the character-level model interpreted each tweet as a matrix of 280 $\times$ 138 (tweets size $\times$ alphabet set size). In contrast, the word-level model encoded each word into a vector of 100 dimensions. Secondly, with an additional layer of CNN, the character-level model learned the embedding of each character in the alphabet set. To compare the two models' real-time performance, we added 500 additional tweets collected from five trending topics and manually labeled them (see Table 9). Similar to the previously obtained results, both models classify a tweet in less than five milliseconds, and the character-level model is still faster than the word-level model. The predictive accuracy of the two models using unseen real-time data is above 0.91, which is an excellent result.

Further analysis of the misclassified samples showed that tweets written in two languages are more likely to be misclassified by the models. Genuine tweets that are falsely classified as

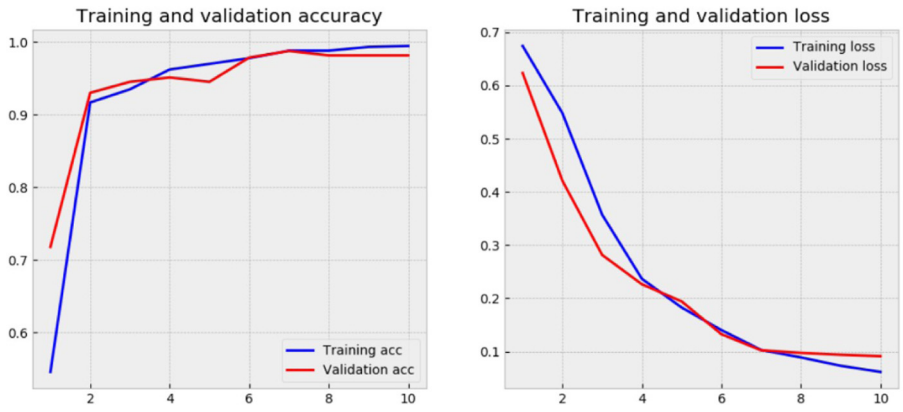

Fig. 4. The accuracy and loss of the word-level model for the 15 Tweets dataset.

Table 9
Comparison of The Word-Level and Character-Level Models.

	Word-level model	Character-level model
Training time	2–3 min on average.	10–12 min on average.
Real-time classification time	0.0018837 s	0.001478 s

Table 10
The performance of the word-level model and the number of tweets for each account.

Metrics	Number of Tweets				
	10	15	20	25	30
Accuracy	0.97	0.98	0.97	0.97	0.98
Precision score	0.98	1	0.96	0.97	0.98
Recall score	0.95	0.96	0.98	0.96	0.98
F1 score	0.97	0.98	0.97	0.97	0.98

low-quality content are often tweets posted on many trending topics. Also, legitimate users sometimes include several hashtags in their tweets to seek great recognition. Tweets without or with short texts are the most frequently misclassified examples in the dataset, where the small number of extracted features negatively affects model performance.

5.2. Account-level classification results

The same hyper-parameters previously discussed (Section 5.1) was also used in the Accounts-models. In the LSTM, we found that replacing the LSTM layer entirely improved the performance of both account-word and account-character models. In the case of the account-character model, 12 epochs led to achieving better and more stable performance.

We conducted an experiment to determine the number of tweets that are required to classify an account accurately. In this experiment, we prepared five datasets using data from the 1000 accounts. For each dataset, a specific number of tweets were pulled from each account, for example, 10 tweets were collected for each username in the 10 Tweets dataset. The models' performance is then tested using the 10, 15, 20, 25, 30 tweets dataset (see Tables 10 and 11) According to our experiment, 15 tweets were optimal for both models (see Figs. 4 and 5). The tables show also that increasing the number of tweets does not necessarily improve the accuracy, similar to the findings reported in previous studies [43]. Therefore, we found that 15 tweets sufficiently described accounts' recent activities rather than the entire tweet history, which might include old tweets, as in the case of compromised accounts.

We also computed the required time to classify a given account using the character- and word-based models. The two models took a long time to classify accounts than tweets as a result of the addition step, which requires retrieving the latest 15 tweets using Twitter API. The running or computational times for this step in the word-account and character-account models were 0.93 s and 0.92 s, respectively. For 100 accounts, the average time required by the word- account model was 0.945 s, the median time was 0.968 s, and the minimum time was 0.87 s For the character-account model, the average time was 0.935 s, the median was 0.940 s, and the minimum time was 0.923 s.

Table 11
The performance of the character-level model and the number of tweets for each account.

Metrics	Number of Tweets				
	10	15	20	25	30
Accuracy	0.97	0.98	0.98	0.96	0.94
Precision score	1	0.99	0.97	0.98	0.97
Recall score	0.94	0.97	1	0.92	0.88
F1 score	0.96	0.98	0.98	0.95	0.93

6. Limitations

The proposed approach substantially relies on the tweets' textual data as the only source of features. Consequently, the system classification accuracy is fundamentally tied to the length of the tweets' text. As mentioned in the Tweet-level model results, tweets that contain short or nil text, such as tweets with an image or URL only are most likely to be misclassified. This places a series of limitations on the performance of the proposed system. Nevertheless, this might not be seen as a limitation in some existing systems where tweets with short text are ignored since it does not provide any valuable information such as the case of sentiment analysis applications. The second limitation in the proposed approach is a language-based system that had been trained in one language. Therefore, tweets that include textual data of more than one language (e.g., Arabic and English) are more likely to be wrongly classified. This also applies to the accounts classification, in which accounts containing two or more languages are more likely to be wrongly classified. Lastly, the size of the parameter n might negatively impact the classification speed in the case of topics with a smaller number of coming tweets. In such a way, the tweets in the waiting queue will take a long time in the meantime waiting for the next coming tweets until the queue is full and ready for the cleaning and modeling phase.

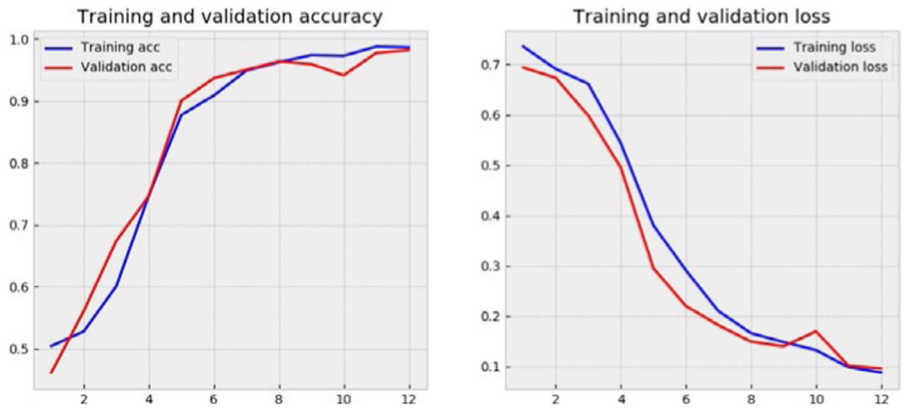

Fig. 5. The accuracy and loss of the character-level model for the 15 Tweets dataset.

7. Conclusions and future work

This paper presented a real-time deep learning approach to filter low-quality tweets and spam accounts in particular topics. The proposed system consists of three models: streaming tweets from a particular topic, cleaning, and numerically representing the tweet's textual features, and then classifying the tweets using the deep learning model. The results of the deep learning models' evaluation using a real-world dataset showed an outstanding performance in distinguishing low-quality tweets from genuine tweets. Regarding the time required to classify tweets, the proposed approach also achieves superior performance compared with existing real-time systems. The findings also showed that embedding at character-level yields higher classification accuracy and takes less time than word-level embedding. This paper further introduced a new approach to classify Twitter accounts as spam or genuine according to their most recent tweets using a deep learning model. The proposed approach was tested on a dataset with 1000 accounts. The experimental results showed that the deep learning model successfully classified accounts with an accuracy of 0.98 and an F1 measure of 0.98.

For the overall design of the proposed system, different adaptations, tests, and experiments have been left for future work. For instance, further analysis might be carried to determine the best value for the n parameter in the system, which indicates the size of the tweets queue. Performing the model steps in parallel also needs to be investigated to achieve instantaneous classification, specifically the cleaning stage and the CNN layer in the deep model. Also, to maintain the excellent system performance, a self-training mechanism should be developed, such as a real-time semi-supervised algorithm.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, Saudi Arabia, under grant No. (DF-777-165-1441). The authors, therefore, gratefully acknowledge DSR technical and financial support.

References

[1] A. Al-Rawi, Viral news on social media, Digit. Journal. 7 (1) (2019) 63–79, http://dx.doi.org/10.1080/21670811.2017.1387062.
[2] D.B. Kurka, A. Godoy, F.J.V. Zuben, Online social network analysis: A survey of research applications in computer science, 2016, arXiv:1504.05655.
[3] U. Kursuncu, M. Gaur, U. Lokala, K. Thirunarayan, A. Sheth, e.N. Arpinar, I. Budak, N. Dokoohaki, S. Tokdemir, Predictive Analysis on Twitter: Techniques and Applications, Springer International Publishing, 2019, pp. 67–104, http://dx.doi.org/10.1007/978-3-319-94105-9_4.
[4] S. Chhabra, A. Aggarwal, F. Benevenuto, P. Kumaraguru, Phi.sh/$ocial: the phishing landscape through short urls, in: Proceedings of the 8th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference, 2011, pp. 92–101, http://dx.doi.org/10.1145/2030376.2030387.
[5] Twitter, Developer policy terms, 2019, https://developer.twitter.com/en/docs/. (Accessed 9 June 2019).
[6] W. Chen, C.K. Yeo, C.T. Lau, B.S. Lee, A study on real-time low-quality content detection on twitter from the users' perspective, PLoS One 12 (8) (2017) 1–22, http://dx.doi.org/10.1371/journal.pone.0182487.
[7] P. Burnap, A. Javed, O.F. Rana, M.S. Awan, Real-time classification of malicious urls on twitter using machine activity data, in: 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, No. 8, ASONAM, 2015, pp. 970–977, http://dx.doi.org/10.1145/2808797.2809281.
[8] A. Aggarwal, A. Rajadesingan, P. Kumaraguru, Phishari: Automatic realtime phishing detection on twitter, in: 2012 eCrime Researchers Summit, 2012, pp. 1–12, http://dx.doi.org/10.1109/eCrime.2012.6489521.
[9] S. Sedhai, A. Sun, Semi-supervised spam detection in twitter stream, IEEE Trans. Comput. Soc. Syst. 5 (1) (2018) 169–175, http://dx.doi.org/10.1109/TCSS.2017.2773581.
[10] S. Lee, J. Kim, Warningbird: A near real-time detection system for suspicious urls in twitter stream, IEEE Trans. Dependable Secure Comput. 10 (3) (2013) 183–195, http://dx.doi.org/10.1109/TDSC.2013.3.
[11] S. Gupta, D. Kuchhal, P. Gupta, M. Ahamad, M. Gupta, P. Kumaraguru, Under the shadow of sunshine: Characterizing spam campaigns abusing phone numbers across online social networks, in: Proceedings of the 10th ACM Conference on Web Science, 2018, pp. 67–76.
[12] B. Wang, A. Zubiaga, M. Liakata, R. Procter, Making the most of tweet-inherent features for social spam detection on twitter, arXiv preprint arXiv:1503.07405.
[13] C. Chen, J. Zhang, X. Chen, Y. Xiang, W. Zhou, 6 million spam tweets: A large ground truth for timely twitter spam detection, in: 2015 IEEE International Conference on Communications, ICC, IEEE, 2015, pp. 7065–7070.
[14] M. Ashour, C. Salama, M.W. El-Kharashi, Detecting spam tweets using character n-gram features, in: 2018 13th International Conference on Computer Engineering and Systems, ICCES, IEEE, 2018, pp. 190–195.
[15] C. Cao, J. Caverlee, Detecting spam urls in social media via behavioral analysis, in: European Conference on Information Retrieval, Springer, 2015, pp. 703–714.
[16] S. Liu, J. Zhang, Y. Xiang, Statistical detection of online drifting twitter spam, in: Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, 2016, pp. 1–10.
[17] S. Liu, Y. Wang, J. Zhang, C. Chen, Y. Xiang, Addressing the class imbalance problem in twitter spam detection using ensemble learning, Comput. Secur. 69 (2017) http://dx.doi.org/10.1016/j.cose.2016.12.004.
[18] C. Li, S. Liu, A comparative study of the class imbalance problem in twitter spam detection, Concurr. Comput.: Pract. Exper. 30 (2018) http://dx.doi.org/10.1002/cpe.4281.

[19] M. Lin, K. Tang, X. Yao, Dynamic sampling approach to training neu ral networks for multiclass imbalance classification, IEEE Trans. Neural Netw. Learn. Syst. 24 (4) (2013) 647–660, http://dx.doi.org/10.1109/TNNLS.2012.2228231.
[20] M. Fazil, M. Abulaish, A hybrid approach for detecting automated spammers in twitter, IEEE Trans. Inf. Forensics Secur. 13 (11) (2018) 2707–2719, http://dx.doi.org/10.1109/TIFS.2018.2825958.
[21] Z. Alom, B. Carminati, E. Ferrari, Detecting spam accounts on twitter, in: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM, IEEE Computer Society, Los Alamitos, CA, USA, 2018, pp. 1191–1198, http://dx.doi.org/10.1109/ASONAM.2018.8508495.
[22] K. Lee, J. Caverlee, S. Webb, Uncovering Social Spammers: Social Honeypots Machine Learning, Association for Computing Machinery, New York, NY, USA, 2010, http://dx.doi.org/10.1145/1835449.1835522.
[23] F. Benevenuto, G. Magno, T. Rodrigues, V. Almeida, Detecting spammers on twitter, in: Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference, Vol. 6, CEAS, 2010, p. 12, URL http://ceas.cc/2010/papers/Paper2021.pdf.
[24] T. Wu, S. Liu, J. Zhang, Y. Xiang, Twitter Spam Detection Based on Deep Learning, Association for Computing Machinery, New York, NY, USA, 2017, http://dx.doi.org/10.1145/3014812.3014815.
[25] G. Jain, M. Sharma, B. Agarwal, Spam detection on social media using semantic convolutional neural network, Int. J. Knowl. Discov. Bioinform. 8 (2018) 12–26, http://dx.doi.org/10.4018/IJKDB.2018010102.
[26] Google, word2vec, 2013, https://code.google.com/archive/p/word2vec/. (Accessed 9 June 2019).
[27] G. Jain, M. Sharma, B. Agarwal, Optimizing semantic lstm for spam detection, Int. J. Inf. Technol. 11 (2) (2019) 239–250, http://dx.doi.org/10.1007/s41870-018-0157-5.
[28] G. Jain, M. Sharma, B. Agarwal, Spam detection in social media using convolutional and long short term memory neural network, Ann. Math. Artif. Intell. 85 (2019) http://dx.doi.org/10.1007/s10472-018-9612-z.
[29] S. Madisetty, M.S. Desarkar, A neural network-based ensemble approach for spam detection in twitter, IEEE Trans. Comput. Soc. Syst. 5 (4) (2018) 973–984, http://dx.doi.org/10.1109/TCSS.2018.2878852.
[30] A. Anand, T. Chakraborty, N. Park, We used neural networks to detect clickbaits: You won't believe what happened next!, in: European Conference on Information Retrieval, Springer, 2017, pp. 541–547, arXiv:1612.01340.
[31] G. Jain, B. Agarwal, et al., An overview of rnn and cnn techniques for spam detection in social media, Int. J. Adv. Res. Comput. Sci. Softw. Eng. 6 (10) (2016) 126–132.
[32] K. Thomas, C. Grier, J. Ma, V. Paxson, D. Song, Design and evaluation of a real-time url spam filtering service, in: 2011 IEEE Symposium on Security and Privacy, IEEE, 2011, pp. 447–462, http://dx.doi.org/10.1109/SP.2011.25.
[33] T. Wu, S. Wen, Y. Xiang, W. Zhou, Twitter spam detection: Survey of new approaches and comparative study, Comput. Secur. 76 (2018) 265–284, http://dx.doi.org/10.1016/j.cose.2017.11.013.
[34] J. Martinez-Romo, L. Araujo, Detecting malicious tweets in trending topics using a statistical analysis of language, Expert Syst. Appl. 40 (8) (2013) 2992–3000, http://dx.doi.org/10.1016/j.eswa.2012.12.015.
[35] S.a.R. Seifert, Christian, Capture – Honeypot client (capture-HPC), 2019, https://projects.honeynet.org/capture-hpc. (Accessed 9 June 2019).
[36] H. Gupta, M.S. Jamal, S. Madisetty, M.S. Desarkar, A framework for real-time spam detection in twitter, in: 2018 10th International Conference on Communication Systems & Networks, COMSNETS, IEEE, 2018, pp. 380–383.
[37] Twitter, About Twitter's APIs, 2019, https://help.twitter.com/en/rules-and-policies/twitter-api. (Accessed 9 November 2019).
[38] A.B. Soliman, K. Eissa, S.R. El-Beltagy, Aravec: A set of arabic word embedding models for use in arabic nlp, Procedia Comput. Sci. 117 (2017) 256–265, http://dx.doi.org/10.1016/j.procs.2017.10.117, arabic Computational Linguistics.
[39] X. Zhang, J. Zhao, Y. LeCun, Character-level convolutional networks for text classification, 2016, arXiv:1509.01626.
[40] C. Zhou, C. Sun, Z. Liu, F.C.M. Lau, A c-lstm neural network for text classification, 2015, arXiv:1511.08630.
[41] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017, arXiv:1412.6980.
[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (56) (2014) 1929–1958, URL http://jmlr.org/papers/v15/srivastava14a.html.
[43] E. Clark, J. Williams, R. Galbraith, C. Jones, C. Danforth, P. Dodds, Sifting robotic from organic text: a natural language approach for detecting automation on twitter, J. Comput. Sci. 16 (2016) http://dx.doi.org/10.1016/j.jocs.2015.11.002.