



Research Article

Predicting Coronavirus Pandemic in Real-Time Using Machine Learning and Big Data Streaming System

Xiongwei Zhang,¹ Hager Saleh ,^{2,3} Eman M. G. Younis,² Radhya Sahal ,⁴ and Abdelmgeid A. Ali²

¹School of Mathematics and Statistics, Yulin University, Yulin 719000, China

²Faculty of Computers and Information, Minia University, Minya, Egypt

³Faculty of Computers and Artificial Intelligence, South Valley University, Hurghada, Egypt

⁴Faculty of Computer Science and Engineering, Hodeidah University, Al Hudaydah, Yemen

Correspondence should be addressed to Hager Saleh; hager.saleh.fci@gmail.com

Received 27 October 2020; Accepted 3 December 2020; Published 22 December 2020

Academic Editor: Ahmed Mostafa Khalil

Copyright © 2020 Xiongwei Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Twitter is a virtual social network where people share their posts and opinions about the current situation, such as the coronavirus pandemic. It is considered the most significant streaming data source for machine learning research in terms of analysis, prediction, knowledge extraction, and opinions. Sentiment analysis is a text analysis method that has gained further significance due to social networks' emergence. Therefore, this paper introduces a real-time system for sentiment prediction on Twitter streaming data for tweets about the coronavirus pandemic. The proposed system aims to find the optimal machine learning model that obtains the best performance for coronavirus sentiment analysis prediction and then uses it in real-time. The proposed system has been developed into two components: developing an offline sentiment analysis and modeling an online prediction pipeline. The system has two components: the offline and the online components. For the offline component of the system, the historical tweets' dataset was collected in duration 23/01/2020 and 01/06/2020 and filtered by #COVID-19 and #Coronavirus hashtags. Two feature extraction methods of textual data analysis were used, n-gram and TF-IDF, to extract the dataset's essential features, collected using coronavirus hashtags. Then, five regular machine learning algorithms were performed and compared: decision tree, logistic regression, *k*-nearest neighbors, random forest, and support vector machine to select the best model for the online prediction component. The online prediction pipeline was developed using Twitter Streaming API, Apache Kafka, and Apache Spark. The experimental results indicate that the RF model using the unigram feature extraction method has achieved the best performance, and it is used for sentiment prediction on Twitter streaming data for coronavirus.

1. Introduction

Coronavirus disease or COVID-19 is a novel virus disease that started in the last year of 2019 [1]. The World Health Organization (WHO), on 11th March 2020, stated the outbreak of COVID-19 as a pandemic [2]. The virus, surprisingly fast, has now propagated throughout the world; and nearly 28 all the countries are now fighting against it, trying their best to stop the propagation of this invisible murderer as much as can be attained [3]. As started from the most populated country in the world, China, COVID-19 has spread and killed thousands of

people from various countries, including Italy, Spain, the USA, Iran, and other European countries. Recently, especially during the last half of May, while this worldwide pandemic has persistently continued to affect the lives of millions in several countries, several countries have no other solution but to resort to total lockdown. Twitter, as considered one of the widely-used social network sites, is a rapidly-growing and invasive online platform through which people can do various activities, including creating, posting, updating, and reading limited-length text messages known as tweets. These tweets form a simple means for users to share and exchange their opinions,

points of view, and ideas about a given topic. On the other side, sentiment analysis (SA) is the method of recognizing and classifying a specific text's polarity in terms of document, sentence, and phrase [4, 5]. Sentiment analysis is a tool based on machine learning methodologies and one of the most significant fields in natural language processing [6, 7]. Recently, the world has been fighting COVID-19 for the last six months, and most people around the world are under lockdown. Many people have used social networks such as Twitter to express their opinions and attitudes towards COVID-19 and share their experiences in facing this virus. Therefore, the importance of Twitter has increased more than ever, and sentiment analysis for Twitter data has become a hot topic in data science research. As a result, Twitter has become rich with much information/data about people's opinions on COVID-19, which has led some researchers to use these data to conduct their research studies and experiments about COVID-19. Some researchers applied sentiment analysis to study people's opinions about COVID-19. For example, Samuel et al. [8] have used machine learning classification for classifying coronavirus tweets. Also, Ali et al. [9] have used machine learning algorithms on tweet data about coronavirus to classify tweets into positive and negative. On the other hand, the scale of data streaming in social networks, such as Twitter, is increasing exponentially [10]. Streaming data is a valuable source of data analysis which is collected in real-time. However, real-time sentiment analysis is considered one of the most exigent research domains which necessitates robust tools of big data analytics like Apache Spark. Handling streaming data using traditional preprocessing methods is a challenge; therefore, researchers and organizations are using big data platforms such as Apache Spark [11] and Apache Kafka [12] to process and store the streaming data. Moreover, recent studies have been using machine learning with big data technologies to make their experiences. For example, Das et al. [13] have developed a real-time sentiment system to predict stock prices' sentiment from Twitter data. Rath et al. [14] have also developed a real-time sentiment analysis system based on targeted advertising. Previous studies applied sentiment analysis techniques to study and analyze attitudes and opinions about coronavirus using the historical data collected from Twitter. There is no study oriented to sentiment analysis in real-time during the coronavirus pandemic to the best of our knowledge. Therefore, this motivates us to introduce a new real-time sentiment prediction system, including Twitter streaming data for coronavirus pandemic. The proposed system results (i.e., classified predicted people's opinions about coronavirus) could be emitted to any data source such as real-time reporting and dashboard, storage, and mobile apps. The predicted results will help healthcare organizations, medical industries, organizational psychology experts, and society monitor current and future studies.

The paper contributions can be summarized as follows:

- (1) Developing a real-time system to predict the sentiment for coronavirus pandemic using Twitter streaming data
- (2) Collecting tweets data about coronavirus using #COVID-19 and #Coronavirus hashtags and then retraining tweets data using TextBlob

- (3) Applying different n-gram sizes with the TF-IDF feature selection method
- (4) Comparing five machine learning classifications to find the optimal model used to predict coronavirus sentiment in real-time

The remainder of this paper is organized as follows. The related work is presented in Section 2. The proposed system of real-time sentiment prediction is introduced in Section 3. The experimental results are discussed in Section 4. Finally, conclusions are presented in Section 5.

2. Related Work

Researchers have recently applied sentiment analysis and word frequency techniques to classify people's attitudes from tweets. We divide the related work into the following two categories: real-time predictive analysis for healthcare and coronavirus pandemic sentiment analysis.

2.1. Real-Time Prediction Systems for Healthcare.

Recently, some researchers have been applying machine learning and big data techniques on tweets data to do experiments in real-time. For example, Hager et al. [15] have introduced a real-time system to predict heart disease from tweets streaming based on Apache Spark and Apache Kafka. They have applied regular machine learning algorithms, including DT, SVM, RF, and LR, with cross-validation and grid search to optimize models and achieve the best performance. The researchers have found that the RF classifier has obtained the best performance compared with the other models, and then they have selected it to predict the heart disease status from patients tweet in real-time. Zaki et al. [16] have also developed a framework to collect, process, predict, and visualize Twitter data. They also have a real-time analytic model to predict Iraqi sentiment from streaming tweets based on Apache Spark. Also, Hashim [17] has introduced a real-time system to detect Twitter fake accounts based on Apache Spark. He used Spark's MLlib and developed different types of machine learning algorithms, including DT, SVM, and Naïve Bayes (NB). In addition, he has developed a real-time reporting and dashboard component to visualize the sentiment analysis results.

2.2. Coronavirus Pandemic Sentiment Analysis.

Rajput et al. [18] have used two techniques, which are word frequency and sentiment analysis, to analyze tweet messages about the coronavirus outbreak. They have used unigram, bigram, and trigram to describe one word's rates, two words, and three words, respectively. In sentiment analysis, a TextBlob, i.e., Python package, was used to classify tweets into positive, negative, and neutral. Recently, Bhat et al. [19] have collected tweets using two hashtags, #COVID-19 and #Coronavirus. The authors have applied the sentiment analysis for Twitter data to classify them into positive, negative, and neutral. Also, Dubey [20] has collected tweets that are retaliated to COVID-19 from 11/03/2020 to 31/03/2020. He has identified people's emotions about COVID-19 from different countries in the world. He has used NRC Lexicon to analyze

and classify the tweets into eight emotions. Furthermore, Manguri et al. [21] have collected tweets data about coronavirus for seven days from 09/04/2020 to 15/04/2020. Also, they have used the TextBlob to classify tweets into positive, negative, and neutral. The results show that neutral has got the highest percentage. Moreover, Medford et al. [22] have extracted tweets related to COVID-19 and measured the frequency of keywords of infection restraint practices, treatment, and phylogenetic prejudice.

3. Real-Time Sentiment Prediction System

The proposed system of real-time sentiment prediction consists of two main components, which are developing an offline sentiment analysis model and an online sentiment prediction pipeline, as shown in Figure 1. Each component is described in detail in the following subsections.

3.1. Developing an Offline Sentiment Analysis Model. The offline sentiment analysis model has been developed to train and test the machine learning models to find the optimal model used in the online sentiment prediction pipeline. The machine learning models which have been used are decision tree (DT), support vector machine (SVM), random forest classifier (RF), logistic regression classifier (LR), and K-nearest neighbor (KNN). The machine learning models have been trained and tested using the collected tweets dataset about coronavirus. Figure 2 depicts this component's primary stages/steps, including data collection, data preprocessing, pretrained tweets, data splitting, feature extraction, optimization/training models, and model evaluation. Each stage is described as follows:

3.1.1. Data Collection. Twitter is one of the most widely-used social platforms for people's interaction, content posting, sharing, and commenting on various topics people discuss, including health issues. Nowadays, the world suffers from the COVID-19 pandemic, and people start posting a deluge of tweets about it. This massive information could be a good source, but it needs analysis in the age of the COVID-19 pandemic in its positivity and negativity. Therefore, the sentiment analysis of Twitter streaming data plays a role in disseminating medical information for the coronavirus pandemic. To actualize the proposed system's data collection stage, Twitter APIs have been considered for ingesting streaming data. Twitter's Streaming APIs are categorized into the Search API and Stream API [23]. Twitter's Search API is used to gather historical Twitter data offline, while Twitter's Stream API is used to stream real-time data through the online phase. For the offline stage, we have collected historical Twitter data about coronavirus in duration between 23/01/2020 and 01/06/2020 filtered by #COVID-19 and #Coronavirus hashtags. In doing so, we have created an authorization connection with Twitter using Twitter Streaming API. In particular, the OAuth authentication protocol is used to authorize applications to access Twitter services. Twitter's Stream API is used to stream real-time data from Twitter, which uses in the online phase. Examples of the collected

tweets about coronavirus pandemic using #COVID-19 and #Coronavirus hashtags are shown in Table 1.

3.1.2. Data Preprocessing. Data preprocessing is critical in any social network-based analysis system (i.e., sentiment analysis of streaming Twitter data) as it directly impacts the effectiveness of the sentiment analysis due to the data's complexity. According to our work, although Twitter is considered a gold mine of data, it is regarded as one of the noisiest data because it consists of many links, hashtags, special symbols, emojis, and so on. Therefore, the collected Twitter data have been preprocessed using the following steps: noise removal, tokenization, normalization, and stems, which are described as follows:

Noise Removal. In this phase, the useless data are removed in the following steps:

- (i) *Lower Casing.* The lowercasing is the most effective form of text preprocessing, which guarantees correlation within the feature set and sparsity issue. For example, CovId and CovID ought to be converted to COVID-19.
- (ii) *Removal of URL's.* In this step, we have removed irrelevant links embedded into Twitter posts.
- (iii) *Removal of Special Symbols.* In this step, we have removed special symbols like punctuations.
- (iv) *Removal of Hashtag.* The Twitter hashtag is used to index keywords or topics on Twitter, written with a #symbol. In this work, the essential hashtags #COVID-19 and #Coronavirus were removed.
- (v) *Removal of Stop Word.* The stop words are insignificant words in a language and useless in the sentiment analysis, which are used for language grammar structure. We have filtered out these stop words, including articles, conjunctions, prepositions, some pronouns, and common words such as the, a, an, about, by, from, to, and so on.

Tokenization. Tokenization in preprocessing refers to breaking longer strings of a text into tokens (i.e., smaller pieces). These tokens could be paragraphs that can be further split into shorter sentences, which can be, in turn, divided into words. For example, consider this sentence before tokenization: "hard work pays off."

Normalization. The normalization step in preprocessing is transforming a text into a standard form to increase the uniformity of text preprocessing. It includes the conversion of all text to either upper or lower case.

Stemming. After the tokenization step, the next step is stemming. The stemming step is changing the words into their original form (i.e., root form to decrease the number of word types or classes in the data). For example, the terms "Walking," "Walked," and "Walker" will be reduced to the word "walk".

3.1.3. Pretrained Tweets. Sentiment analysis identifies the emotions or attitudes the writer holds (i.e., Twitter handle/user), whether these emotions/attitudes can be positive,

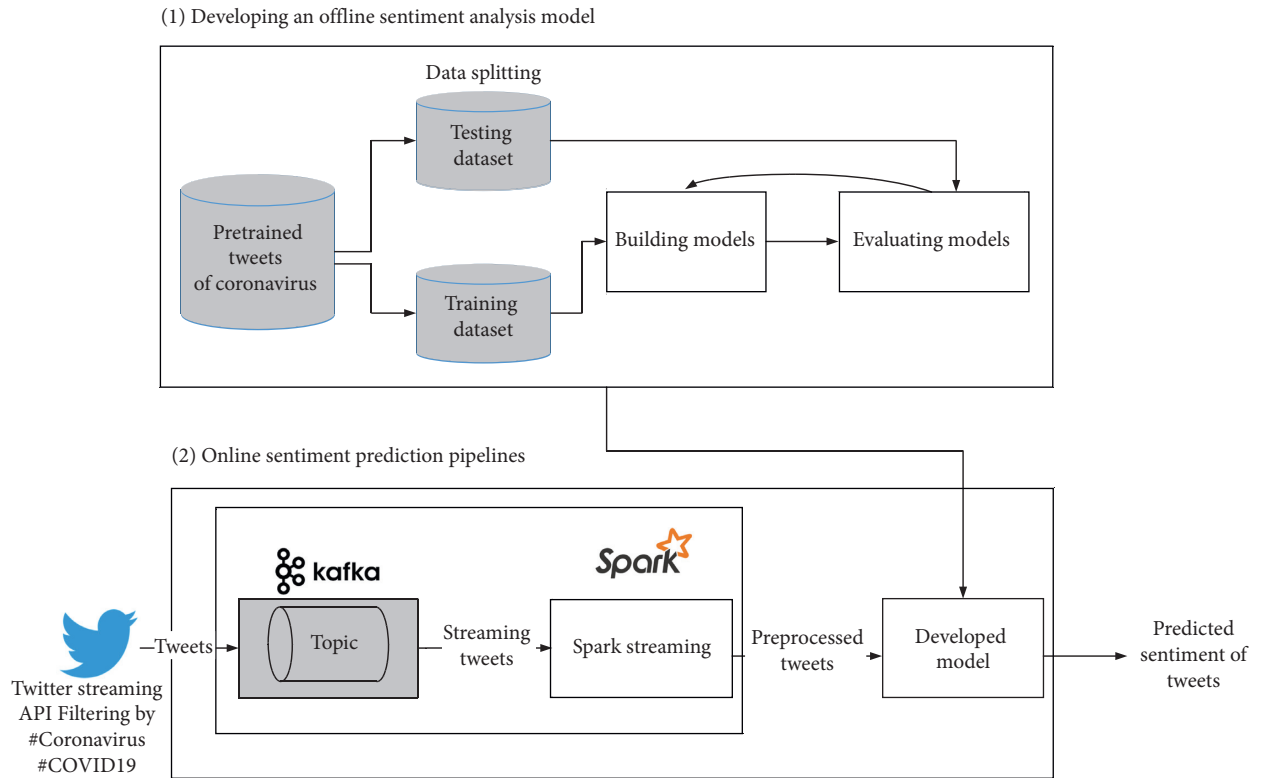


FIGURE 1: The architecture of the real-time sentiment prediction system.

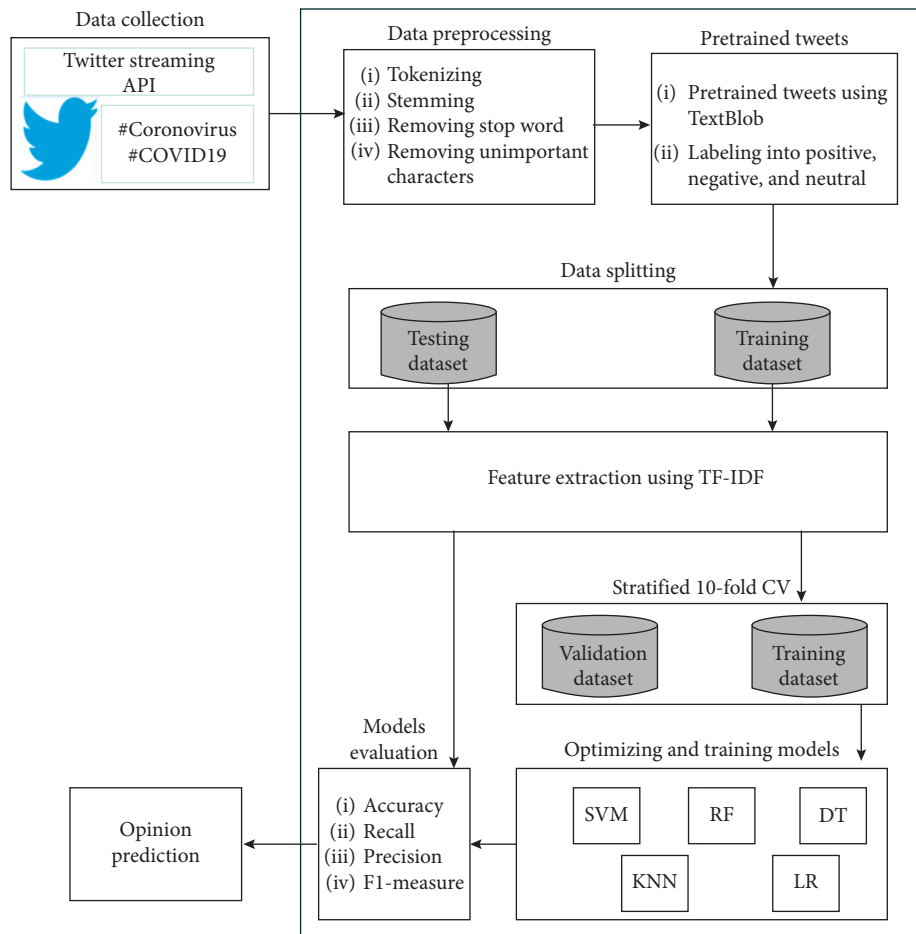


FIGURE 2: The offline model of the real-time sentiment prediction system.

TABLE 1: Examples of the collected tweets about coronavirus pandemic using #COVID-19 and #Coronavirus hashtags.

Hashtags	Tweet
#covid19	great job 7 corner VA store employ outside stop without mask nice explain protocol great service commun covid19
#coronavirus	103 000 dead american count trump viru coronaviru covid19 trump gop maga election2020 voteblue2020 rememberinnovemb trumpdeath toll103k
#covid19	the kobe george floyd connect the whole world IS A stage via george floyd protest kolibri covid19
#covid19	evolut covid19 follow amaz meme igshid 5x3z8d9bmc3h

negative, or neutral. Therefore, we used a TextBlob [24], a Python library, to perform sentiment analysis on data collected from Twitter. As determined by the TextBlob, TextBlob employs the Naïve Bayes (NB) model for classification, and it returns two properties as outputs, namely, polarity and subjectivity. The TextBlob contradiction means identifying sentiment orientation (positive, neutral, and negative), whereas subjectivity means expressing some personal emotions, feelings, opinions, or beliefs. We have used the output polarities of tweets to label the collected dataset to be fitted in the machine learning models in the evaluation step regarding this work.

3.1.4. Data Splitting. In this step, the resulting pretrained dataset is split into 90% of the training dataset and 10% of the testing dataset using a stratified method. The training set is used to optimize and train the machine learning models, while the unseen test set is used to evaluate the machine learning models.

3.1.5. Feature Extraction. One of the textual data analysis challenges is feature extraction due to learning from the high-dimensional data [25]. It is best to use some feature extraction methods to convert text into a matrix (or vector) of features. Therefore, we have applied two of the most popular feature extraction methods on the historically collected tweet data, namely, n -gram and TF-IDF.

N -gram modeling is a popular feature selection and analysis method extensively used in text mining and natural language processing. According to textual data analysis, n -gram is used to compute a contiguous sequence of words with length n within a given window. In this work, we have used the n -gram method, including $n = 1$ to $n = 4$ (i.e., unigram, bigram, trigram, and four-gram) to represent the context the Twitter data.

The term frequency-inverse document frequency (TF-IDF) is a famous method used in evaluating the importance level of a word in a document used in retrieving information and natural language processing. The goal of TF-IDF is to calculate the word frequency within the text in a massive document corpus. The TF-IDF method uses the relative frequency level through the reference document corpus, which can be considered great merit.

3.1.6. Optimization and Training Models. The machine learning models used are as follows: SVM [26], DT [27], KNN [28], RF [29], and LR [30]. For every experiment, the

training set is used to optimize the hyperparameters of the model by using the grid search technique. The grid search method with stratified 10-fold cross-validation (CV) has been employed to discover the optimal hyperparameters of ML algorithms. Finally, the resulting models were evaluated by using the unseen test set. We have used a grid search with stratified 10-fold cross-validation to find the optimal hyperparameters of all models.

3.1.7. Evaluating the Models. Four standard metrics were utilized to evaluate the models' accuracy, precision, recall, and F1-score, where TP is true positive, TN is true negative, FP is false positive, and FN is a false negative given in the following equations:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \quad (1)$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad (2)$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad (3)$$

$$\text{F1-score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \quad (4)$$

3.2. Online Sentiment Prediction Pipeline. The online sentiment prediction pipeline component aims to predict the sentiment analysis of the coronavirus tweets in real-time and evaluate the proposed system's ability to work in real-time. To do so, it collects streaming tweets, and then it performs real-time processing of the data that are fed to the ML model to predict the sentiment analysis of the coronavirus tweets. The online prediction pipeline component is developed using Twitter streaming API, a distributed message system (i.e., Apache Kafka), and big data processing platform (i.e., Apache Spark). It is introduced in two steps: twitter streaming data collection and real-time sentiment analysis and prediction. Each step can be described as follows.

3.2.1. Twitter Streaming Data Collection. In this step, we have used Twitter Streaming API to stream Twitter data filtered by a #Coronavirus and #COVID-19 hashtags and Apache Kafka to ingest data from Twitter. Twitter Streaming API is used to retrieve data about coronavirus produced in real-time to infer how positive, negative, or neutral the feeling towards this pandemic is. For connecting to the API and retrieving Twitter data, we have used a Python library

called Tweepy. We have used a persistent HTTP connection and a user authorization supported by OAuth protocol.

After the connection to Twitter Streaming API is established, we have performed the developed script to retrieve streaming tweets from 20/06/2020 to 30/06/2020 filtered by #COVID-19 and #Coronavirus hashtags. For example, if somebody posts the following message (see Figure 3), we will add the message to the collected streaming dataset. Then, the real-time Twitter streaming data are ingested on-the-fly to Kafka topic.

3.2.2. Real-Time Sentiment Analysis and Prediction. After listing out the intersteps for the twitter streaming data collection step, streaming data are ingested from Twitter to Kafka's topic. Spark streaming and machine learning capabilities are then utilized to process streaming tweets and perform the best prediction model for sentiment analysis. In particular, Spark streaming preprocesses the collected tweets of coronavirus on-the-fly to convert them into vectors to be fitted in the best machine learning model; and then the best machine learning model realizes the corresponding sentiment prediction of each one in real-time. Substantially, the intersteps for real-time sentiment analysis and prediction are listed as follows:

- (i) For the analysis step, Spark Streaming API retrieves tweets from Kafka's topic and performs the preprocessing steps, including noise removing, tokenization, normalization, and stemming. Then, it extracts features to send them in a vector structure to the best model.
- (ii) For the prediction step, Spark uses the best prediction model obtained in the offline phase to classify each tweet's sentiment about coronavirus into three notable classes: positive, negative, and neutral real-time. For example, using the retrieved tweet in Figure 3, the proposed system can predict that the Twitter user has negative feelings about corona because he is afraid of being sick. The online prediction results could be emitted to any data source such as real-time reporting and dashboard, storage, and mobile apps.

4. Results and Discussion

In this section, the experimental evaluation of the proposed system is presented, starting by describing the experiment setup.

4.1. Experiment Setup. Our proposed system for real-time sentiment analysis has been developed in Python using Spark's Mlib to implement machine learning models including RF, DT, LR, SVM, and KNN. Twitter Streaming API was used to collect data from Twitter, and Apache Kafka was used to receiving streaming data then ingesting it into Kafka topic. Spark streaming was used to read tweets streaming from a Kafka topic. The experiments have been performed using Spark cluster version 2.6.0 which consists of one

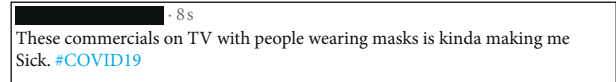


FIGURE 3: Real-time stream Twitter message using #COVID-19 hashtag.

master node and two worker nodes configured with a 20 GB of RAM, 7 cores, 100 GB disk and running over Ubuntu Linux 18.04.

The experiments of developing the offline model aim to find the optimal machine learning model with the highest performance to use in the real-time prediction of the sentiment polarity. We examine the five machine learning models' performance using the collected tweets dataset, which is about coronavirus (Twitter hashtags are #Coronavirus and #COVID-19) during the period between 30/01/2020 and 01/06/2020. The dataset is preprocessed, including cleaning, tokenization, stop-word removing, and stemming steps, as described in the previous sections. Then, the dataset is pretrained into positive, negative, and neutral using TextBlob API. The machine learning models were first trained with 90% of data and then tested with the remaining 10% of data. The five machine learning classifiers were implemented using the Scikit-learn 0.21.3 package in Python 3.7. for classification. In addition, the stratified 10-fold cross-validation is used for hyperparameter tuning and model training. Four standard metrics are used to evaluate the models, including accuracy, precision, recall, and F1-score. We used TF-IDF features extraction methods with different sizes of n-gram from $n = 1$ to $n = 4$. Therefore, for the offline phase, our experiments could be divided into unigram, bigram, trigram, and four-gram with two matrix sizes: 1000 and 3000. The following two sections discuss the collected results of the offline phase and the online phase in detail.

4.2. Results of the Offline Phase. In this section, the results of applying five machine learning models including cross-validation results and testing results are described. Each machine learning model performance is discussed using four sizes of TF-IDF feature extraction including, unigram, bigram, trigram, and four-gram, and two matrix sizes: 1000 and 3000.

4.3. Cross-Validation Results. We experimentally demonstrate the performance of the 10-fold CV results of the five machine learning models over the used dataset with two different matrix sizes (i.e., 1000 and 3000). As the results shown in Tables 2–6, for DT, KNN, LR, RF, and SVM, respectively, the dataset using 3000 matrix size has higher performance for all TF-IDF feature extraction methods, including unigram, bigram, trigram, and four-gram. We attribute this behaviour to the larger number of words within the matrix. When the number of words is slightly larger, the weighting metric becomes more significant, which improves the machine learning model performance. However, the machine learning models' performance using 3000 matrix sizes varied based on the model and the feature

extraction method. For example, as shown in Table 2, the DT model using unigram with 3000 matrix size has obtained the highest performance (accuracy of 87.09%, precision of 87.14%, recall of 87.15%, and F1-score of 86.4%). However, the worst DT model performance has been achieved using four-gram and 1000 matrix size (accuracy of 81.32%, precision of 81.93%, recall of 81.37%, and F1-score of 81.14%). Similar to the LR model, the highest performance has been achieved using unigram with 3000 matrices (accuracy of 89.22%, precision of 89.36%, recall of 89.22%, and F1-score of 89.08%) (Table 3). The worst LR model performance has been achieved using four-gram and 1000 matrix size (accuracy of 83.47%, precision of 84.79%, recall of 83.47%, and F1-score of 83.05%). Regarding KNN, the highest performances have been obtained using four-gram and 300 matrix size among other feature extraction (accuracy of 69.25%, precision of 76.16%, recall of 69.25%, and F1-score of 66.56%) (see Table 4). Furthermore, KNN has recorded the worst performance using unigram and 1000 matrix size (accuracy of 65.75%, precision of 73.33%, recall of 65.75%, and F1-score of 63.5%). As the results shown in Table 5, the RF model has achieved the highest performance using unigram and 3000 matrix size compared with other feature extraction methods (accuracy of 89.56%, precision of 90.05%, recall of 89.62, and F1-score of 89.3%). However, we have noticed that RF has recorded the lowest performance using four-gram and 1000 matrix size (accuracy of 85.29%, precision of 86.62%, recall of 85.34%, and F1-score of 84.96%). It can be seen in Table 6 that SVM has recorded the highest improvements using unigram and 3000 matrix size (accuracy of 88.8%, precision of 89.52%, recall of 88.8%, and F1-score of 88.54%). In comparison, the lowest performance was obtained using the four-gram method with 100 matrix size (accuracy of 84.56%, precision of 86.27%, recall of 84.56%, and F1-score of 84.1%).

4.4. Testing Results. In this section, we discuss the five machine learning models' generalization performance using the unseen test dataset with two different matrix sizes (i.e., 1000 and 3000). Tables 2–6 have described the testing performance of machine learning models including DT, KNN, LR, RF, and SVM, respectively. As shown in Table 2, the DT model has reached the highest testing performances using the bigram feature extraction method and 3000 matrix size (accuracy of 81.13%, precision of 80.91%, recall of 81.13%, and F1-score of 80.88%). However, it has the worst testing performances with 1000 matrix size using the trigram method (accuracy of 77.92%, precision of 77.92%, recall of 77.92%, and F1-score of 77.53%). LR has obtained consistent testing performances with cross-validation performances. It has achieved the highest testing performances using the unigram method and 3000 matrix size of the dataset (accuracy of 82.94%, precision of 83.01%, recall of 82.94%, and F1-score of 82.61%), and the lowest performances were reported using the four-gram method with 1000 matrix size (accuracy of 80.23%, precision of 80.87%, recall of 80.32%, and F1-score of 79.85%) (see Table 4). Although KNN reported the lowest testing performances, they are consistent

with the cross-validation performances (see Table 3). For example, KNN has obtained the highest test performances using the four-gram method (accuracy of 64.62%, precision of 71.04%, recall of 64.62%, and F1-score of 60.06%). For the RF model, the highest testing performances have been obtained using the unigram method with 3000 matrix size (accuracy of 84.71%, precision of 85.8%, recall of 84.71%, and F1-score of 84.06%) (see Table 5). Also, it can be seen that the lowest testing performances have been reported using the bigram method with 1000 matrix size. For the SVM model described in Table 6, the unigram using 3000 matrix size of the dataset has achieved the highest testing performances compared with the other feature extraction methods (accuracy of 81.17%, precision of 83.33%, recall of 81.17%, and F1-score of 80.35%). However, it can be noticed that the unigram method has reported the lowest testing performances using 1000 matrix size. Significantly, LR and KNN models have reported consistent testing performances with cross-validation performances, while DT, RF, and SVM models have not, even though all testing performances are lower compared with cross-validation performances.

5. Discussion

From the results obtained in our experiments, Figures 4 and 5 depict the empirical results in the big picture for the cross-validation performances and the testing results, respectively. They are showing the performance of the best models for each feature extraction method. To summarize the compared models' performance, we explore the average cross-validation and the testing results of each model using different sizes of feature extraction methods, n -gram from $n = 1$ to $n = 4$ and two sizes of matrix 1000 and 3000. Again, it can be noticed that all models have utilized the larger matrix size of the dataset using 3000 matrix size of the dataset to improve their results. On average, the RF model has achieved the highest cross-validation average and the testing performance average compared with other regular machine learning models. For cross-validation results, the RF model has reached the accuracy of 89.56%, precision of 90.05%, recall of 89.62%, and F1-score of 89.3% using the unigram feature extraction method. For performance testing, RF has achieved an accuracy of 84.71%, precision of 85.8%, recall of 84.71%, and F1-score of 84.06% using the unigram method. LR has achieved the second-best results of cross-validation performance using the unigram method over 3000 matrix size of the dataset (the accuracy of 82.94%, precision of 83.01%, recall of 82.94%, and F1-score of 82.61%). Also, it has been reported to be the second-best results using the unigram method of performance testing (the accuracy of 82.94%, precision of 83.01%, recall of 82.94%, and F1-score of 82.61%). SVM achieved the third rank on the average of cross-validation performance and the testing results unigram. The cross-validation results have recorded the accuracy of 88.8%, precision of 89.52%, recall of 88.8%, and F1-score of 88.54%, and for testing results, the accuracy of 81.17%, precision of 83.33%, recall of 81.17%, and F1-score of 80.35% were recorded. DT and KNN have reported the lowest cross-validation and testing results, where DT achieved the fourth

TABLE 2: The performance of the DT model.

Feature extraction method	Matrix size	Testing performance				Cross-validation performance			
		Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Unigram	1000	78.91	78.91	78.91	78.48	82.85 ± 0.37	83.32 ± 0.49	82.89 ± 0.35	82.38 ± 0.56
	3000	80.31	80.09	80.31	80.06	87.09 ± 0.75	87.14 ± 0.66	87.15 ± 0.66	86.84 ± 0.66
Bigram	1000	78.34	78.32	78.34	77.96	82.17 ± 0.22	82.72 ± 0.29	82.09 ± 0.28	81.78 ± 0.26
	3000	81.13	80.91	81.13	80.88	85.76 ± 0.5	85.86 ± 0.45	85.86 ± 0.47	85.59 ± 0.58
Trigram	1000	77.92	77.92	77.92	77.53	82.23 ± 0.53	82.84 ± 0.52	82.25 ± 0.48	81.93 ± 0.47
	3000	80.31	80.1	80.31	80.09	86.23 ± 0.87	86.13 ± 0.86	86.09 ± 0.81	85.98 ± 0.87
Four-gram	1000	77.97	77.96	77.97	77.6	81.32 ± 0.59	81.93 ± 0.43	81.37 ± 0.49	81.14 ± 0.53
	3000	80.37	80.15	80.37	80.09	85.73 ± 0.75	85.66 ± 0.74	85.73 ± 0.82	85.45 ± 0.72

TABLE 3: The performance of the KNN model.

Feature extraction method	Matrix size	Testing performance				Cross-validation performance			
		Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Unigram	1000	62.15	69.94	62.15	58.99	65.75 ± 0.52	73.33 ± 0.65	65.75 ± 0.52	63.5 ± 0.6
	3000	63.77	70.11	63.77	59.36	68.36 ± 0.61	74.72 ± 0.54	68.36 ± 0.61	65.7 ± 0.76
Bigram	1000	62.97	70.96	62.97	59.5	66.09 ± 0.59	73.85 ± 0.89	66.09 ± 0.59	63.74 ± 0.76
	3000	64.49	71.02	64.49	59.96	69.13 ± 0.76	76.04 ± 0.56	69.13 ± 0.76	66.44 ± 0.97
Trigram	1000	63	70.72	63	59.57	66.08 ± 0.54	73.69 ± 0.71	66.08 ± 0.54	63.75 ± 0.65
	3000	64.54	70.69	64.54	60.07	69.07 ± 0.75	75.61 ± 0.66	69.07 ± 0.75	66.39 ± 0.96
Four-gram	1000	62.93	71.24	62.93	59.53	66.09 ± 0.63	73.76 ± 0.93	66.09 ± 0.63	63.75 ± 0.8
	3000	64.62	71.04	64.62	60.06	69.25 ± 0.82	76.16 ± 0.66	69.25 ± 0.82	66.56 ± 1.05

TABLE 4: The performance of the LR model.

Feature extraction method	Matrix size	Testing performance				Cross-validation performance			
		Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Unigram	1000	80.82	81.22	80.82	80.38	84.54 ± 0.4	85.53 ± 0.48	84.54 ± 0.4	84.16 ± 0.43
	3000	82.94	83.01	82.94	82.61	89.22 ± 0.4	89.36 ± 0.42	89.22 ± 0.4	89.08 ± 0.41
Bigram	1000	80.8	81.33	80.8	80.33	83.98 ± 0.31	85.11 ± 0.35	83.98 ± 0.31	83.56 ± 0.32
	3000	82.32	82.84	82.32	81.84	88.52 ± 0.38	88.86 ± 0.43	88.52 ± 0.38	88.31 ± 0.39
Trigram	1000	80.56	81.08	80.56	80.09	83.92 ± 0.32	85.04 ± 0.39	83.92 ± 0.32	83.5 ± 0.33
	3000	82.32	82.84	82.32	81.84	88.52 ± 0.38	88.86 ± 0.43	88.52 ± 0.38	88.31 ± 0.39
Four-gram	1000	80.32	80.87	80.32	79.85	83.47 ± 0.31	84.69 ± 0.33	83.47 ± 0.31	83.05 ± 0.33
	3000	82.32	82.35	82.32	82.01	88.36 ± 0.43	88.55 ± 0.46	88.36 ± 0.43	88.18 ± 0.44

TABLE 5: The performance of the RF model.

Feature extraction method	Matrix size	Testing performance				Cross-validation performance			
		Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Unigram	1000	83.36	84.69	83.36	82.73	86.43 ± 0.48	87.41 ± 0.46	86.37 ± 0.49	86.02 ± 0.57
	3000	84.71	85.8	84.71	84.06	89.56 ± 0.34	90.05 ± 0.46	89.62 ± 0.34	89.3 ± 0.48
Bigram	1000	83.05	84.41	83.05	82.38	85.79 ± 0.51	86.87 ± 0.52	85.81 ± 0.49	85.4 ± 0.61
	3000	84.7	85.81	84.7	84.09	89.48 ± 0.36	89.79 ± 0.45	89.44 ± 0.35	89.12 ± 0.45
Trigram	1000	83.11	84.49	83.11	82.45	85.81 ± 0.46	86.93 ± 0.49	85.76 ± 0.41	85.4 ± 0.44
	3000	84.67	85.82	84.67	84.04	89.39 ± 0.44	89.9 ± 0.35	89.48 ± 0.39	89.2 ± 0.33
Four-gram	1000	83.07	84.49	83.07	82.46	85.29 ± 0.51	86.62 ± 0.57	85.34 ± 0.57	84.96 ± 0.56
	3000	84.61	85.82	84.61	84	89.41 ± 0.44	89.85 ± 0.41	89.4 ± 0.39	89.11 ± 0.41

rank and KNN achieved the fifth rank. For cross-validation results, DT has reported the accuracy of 87.09%, precision of 87.14%, recall of 87.15%, and F1-score of 86.4% using the unigram method.

In contrast, KNN has reported an accuracy of 69.25%, a precision of 76.16%, a recall of 69.25%, and an F1-score of

66.56% using the four-gram method. Similar to the testing results, DT has reported the accuracy of 81.13%, precision of 80.91%, recall of 81.13%, and F1-score of 80.88% using the bigram method. In comparison, KNN has reported the accuracy of 64.62%, precision of 71.04%, recall of 64.62%, and F1-score of 60.06% using the four-gram method.

TABLE 6: The performance of the SVM model.

Feature extraction method	Dataset	Testing performance				Cross-validation performance			
		Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Unigram	1000	79.66	82.38	79.66	78.52	85.63 ± 0.48	87.03 ± 0.51	85.63 ± 0.48	85.26 ± 0.52
	3000	81.17	83.33	81.17	80.35	88.8 ± 0.32	89.52 ± 0.37	88.8 ± 0.32	88.54 ± 0.36
Bigram	1000	79.43	82.28	79.43	78.26	85.07 ± 0.48	86.63 ± 0.53	85.07 ± 0.48	84.65 ± 0.53
	3000	80.79	83.08	80.79	79.95	88.44 ± 0.4	89.19 ± 0.45	88.44 ± 0.4	88.15 ± 0.44
Trigram	1000	79.61	82.25	79.61	78.47	85.04 ± 0.5	86.6 ± 0.52	85.04 ± 0.5	84.62 ± 0.55
	3000	80.78	83.07	80.78	79.93	88.43 ± 0.37	89.19 ± 0.45	88.43 ± 0.37	88.15 ± 0.42
Four-gram	1000	79.62	82.29	79.62	78.49	84.56 ± 0.55	86.27 ± 0.54	84.56 ± 0.55	84.14 ± 0.59
	3000	80.5	82.8	80.5	79.67	88.33 ± 0.37	89.09 ± 0.44	88.33 ± 0.37	88.04 ± 0.42

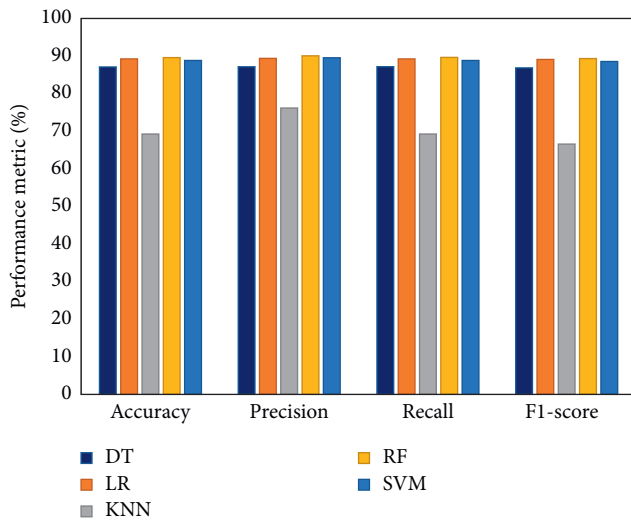


FIGURE 4: The best cross-validation performance.

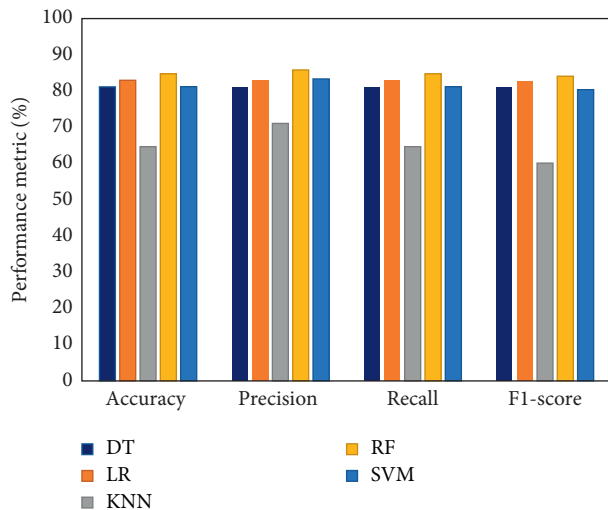


FIGURE 5: The best testing performance.

Consequently, RF is outperforming SVM, DT, LR, and KNN for cross-validation and testing results. Based on these results, it can be tentatively concluded that an RF classifier using the unigram feature extraction method will be used in the sentiment prediction model's real-time performance.

5.1. Real-Time Performance of the Proposed System. The online sentiment prediction pipeline component aims to assess the proposed system's ability to work in real-time by receiving tweets streaming from Twitter. In particular, it is used to estimate the strength of the system to predict people's opinions from tweets in real-time. After testing and developing an offline sentiment analysis model component, the best developed model which is RF with unigram and 3000 sizes of feature matrix is used to predict people's opinions about coronavirus in real-time. The proposed system collects tweets from Twitter using Twitter Streaming API then ingests it into Kafka's topic. Spark streaming reads tweets as streaming from Kafka's topic and applies analysis steps in real-time. Preprocessed tweets are sent to the best model to predict sentiment including neutral, positive, or negative in real-time. In our work, the real-time performance of the system has been evaluated using 2000 tweets. The tweets are classified into 800 neutrals, 800 positives, and 400 negatives.

6. Conclusions

This paper has presented a system for real-time sentiment prediction on Twitter streaming data for coronavirus pandemic. The proposed system has been developed using Twitter Streaming API, Apache Kafka, Apache Spark, and regular machine learning models. It consists of two components, namely, developing an offline sentiment analysis model and an online prediction pipeline. The offline model component is used to obtain the best machine learning model, which will be used on the online sentiment prediction using n-gram and TF-ID feature extraction methods. We have evaluated five machine learning models, which are DT, LR, KNN, RF, and SVM, using the collected dataset (i.e., historical streaming tweets in the period 23/01/2020 and 01/06/2020 filtered by #COVID-19 and #Coronavirus hashtags). The empirical results have proved that the RF model using the unigram feature extraction method has achieved the best performance compared with the other models. The online prediction pipeline component is used to predict the coronavirus tweets' sentiment polarity in real-time. It has used the Twitter Streaming API to collect streaming tweets about coronavirus in real-time then sends them to Kafka. Spark streaming has analyzed the ingested tweets and forwards them to the best machine learning model, which is the RF model, to predict the sentiment polarity about the

coronavirus in tweets in real-time. The experimental results show that the RF model using the unigram feature extraction method has achieved the best performance.

Data Availability

The historical tweets' dataset was collected in duration from 23/01/2020 to 30/06/2020 from Twitter and filtered by #COVID-19 and #Coronavirus hashtags.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] H. Wang, Z. Wang, Y. Dong et al., "Phase-adjusted estimation of the number of coronavirus disease 2019 cases in Wuhan, China," *Cell Discovery*, vol. 6, no. 1, pp. 1–8, 2020.
- [2] W. H. Organization, "Coronavirus," 2020, https://www.who.int/health-topics/coronavirus#tab=tab_1.
- [3] G. Barkur and G. B. K. Vibha, "Sentiment analysis of nationwide lockdown due to COVID 19 outbreak: evidence from India," *Asian Journal of Psychiatry*, vol. 51, Article ID 102089, 2020.
- [4] N. F. F. da Silva, L. F. S. Coletta, E. R. Hruschka, and E. R. Hruschka Jr., "Using unsupervised information to improve semi-supervised tweet sentiment classification," *Information Sciences*, vol. 355–356, pp. 348–365, 2016.
- [5] N. Coletta, "An ensemble classification system for twitter sentiment analysis," *Procedia Computer Science*, vol. 132, pp. 937–946, 2018.
- [6] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: a survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, 2014.
- [7] B. Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures on Human Language Technologies*, vol. 5, no. 1, pp. 1–167, 2012.
- [8] J. Samuel, G. G. M. N. Nawaz Ali, M. M. Rahman, E. Esawi, and Y. Samuel, "Covid-19 public sentiment insights and machine learning for tweets classification," *Information*, vol. 11, no. 6, p. 314, 2020.
- [9] R. Ali, A. Bharathi, and K. Saritha, "COVID-19 outbreak: tweet based analysis and visualization towards the influence of coronavirus in the world," *Gedrag en Organisatie*, vol. 33, no. 2, 2020.
- [10] M. J. Koval, W. W. Lawton, J. G. Tyler, and S. L. Winters, "Data stream protocol for multimedia data streaming data processing system," *Google Patents*, 2014, <https://patents.google.com/patent/US5339413>.
- [11] Apache Spark, "Apache Spark," 2020, <https://spark.apache.org>.
- [12] Apache Kafka, "Apache Kafka," 2020, <https://kafka.apache.org>.
- [13] S. Das, R. K. Behera, M. Kumar, and S. K. Rath, "Real-time sentiment analysis of twitter streaming data for stock prediction," *Procedia Computer Science*, vol. 132, pp. 956–964, 2018.
- [14] L. R. Rath, S. D. Shetty, and S. Deepak Shetty, "Streaming big data analysis for real-time sentiment based targeted advertising," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7, no. 1, p. 402, 2017.
- [15] A. Hager, E. M. G. Younis, A. Hendawi, and A. A. Ali, "Heart disease identification from patients' social posts, machine learning solution on Spark," *Future Generation Computer Systems*, vol. 111, pp. 714–722, 2020.
- [16] N. D. Younis, N. Y. Hashim, Y. M. Mohialden, M. A. Mohammed, T. Sutikno, and A. H. Ali, "A real-time big data sentiment analysis for iraqi tweets using spark streaming," *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 4, pp. 1411–1419, 2020.
- [17] D. Hashim, "A spark-based big data analysis framework for real-time sentiment prediction on streaming data," *Software: Practice and Experience*, vol. 49, no. 9, pp. 1352–1364, 2019.
- [18] N. K. Rajput, B. A. Grover, and V. K. Rathi, "Word frequency and sentiment analysis of twitter messages during coronavirus pandemic," 2020, <https://arxiv.org/abs/2004.03925>.
- [19] M. Bhat, M. Qadri, Noor-ul-Asrar Beg, M. Kundroo, N. Ahanger, and B. Agarwale, "Sentiment analysis of social media response on the Covid19 outbreak," *Brain, Behavior, and Immunity*, vol. 87, pp. 136–137, 2020.
- [20] A. D. Dubey, "Twitter sentiment analysis during COVID19 outbreak," file:///C:/Users/12628/Downloads/SSRN-id3572023.pdf, 2020.
- [21] K. H. Manguri, R. N. Ramadhan, and P. R. M. Amin, "Twitter sentiment analysis on worldwide COVID-19 outbreaks," *Kurdistan Journal of Applied Research*, vol. 5, no. 3, pp. 54–65, 2020.
- [22] R. J. Medford, S. N. Saleh, A. Sumarsono, T. M. Perl, and C. U. Lehmann, "An "infodemic": leveraging high-volume twitter data to understand public sentiment for the COVID-19 outbreak," *Open Forum Infectious Diseases*, vol. 7, no. 7, 2020.
- [23] Twitter, "Twitter streaming API," 2020, <https://developer.twitter.com/en/docs/tweets/filter-realtime/guides/connecting.html>.
- [24] Python, "TextBlob," 2020, <https://textblob.readthedocs.io/en/dev/>.
- [25] H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using n-gram analysis and machine learning techniques," in *Proceedings of the International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, Springer, Vancouver, Canada, October 2017.
- [26] I. Ahmad, M. Basher, M. J. Iqbal, and A. Rahim, "Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection," *IEEE Access*, vol. 6, pp. 33789–33795, 2018.
- [27] B. Basher, D. Wang, S. Cheng, and X. Xie, "Modeling and analysis for vertical handoff based on the decision tree in a heterogeneous vehicle network," *IEEE Access*, vol. 5, pp. 8812–8824, 2017.
- [28] Z. Wang, X. Zhu, D. Cheng, M. Zong, and S. Zhang, "Efficient k NN classification algorithm for big data," *Neurocomputing*, vol. 195, pp. 143–148, 2016.
- [29] M. Zhu, "Random forest classifier for remote sensing classification," *International Journal of Remote Sensing*, vol. 26, no. 1, pp. 217–222, 2005.
- [30] D. G. Kleinbaum, M. Klein, and E. R. Pryor, *Logistic Regression*, Springer, Berlin, Germany, 2002.