Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 10, No. 02, June 2024.

DDoS Attack-detection Approach Based on Ensemble Models Using Spark

Yasmeen Alslman, Ashwaq Khalil, Remah Younisse, Eman Alnagi,
Jaafer Al-Saraireh and Rawan Ghnemat

(Received: 17-Sep.-2023, Revised: 2-Dec.-2023 and 7-Feb.-2024, Accepted: 20-Feb.-2024)

Abstract

We live in an era where time is the most precious resource. Thus, dealing with the vast amount of data collected from different resources for various purposes requires creating systems that can process the data correctly to make it worthwhile. Using big data in machine-learning (ML) and artificial-intelligence (AI) models enhances the efficiency and robustness of such models. This work proposes a DDoS attack detection model using Apache-spark to deal with the CIC-DDOS2019 dataset, a significant public dataset used to train this model. The model is trained to predict the type of DDoS attack among multiclass attacks: SYN, UDP and MSSQL. Two state-of-the-art algorithms, Random Forest (RF) and eXtreme Gradient Boosting (XGBoost), have been chosen as the base of our proposed model. These two algorithms inherit their robustness and efficiency from the ensemble nature of their architecture, where each is constructed of several decision trees with different parameters. To contribute to this work, a stacked ensemble model has been built using both RF and XGBoost to enhance the accuracy of the DDoS attack-detection task. It has been found that using such a combination guarantees the best results. The prolonged execution time that resulted from training such a large dataset, on the other hand, is another issue that should be handled. To tackle the speed problem, the Apache-spark platform has been used. Apache-spark divides the large dataset, distributes the divisions and trains them in parallel using the proposed model. Thus, it enhances the execution time while preserving the accuracy of training the same dataset without Apache-Spark. The proposed model has achieved a high accuracy of (99.94%) while reducing the execution time to almost half of the time when applied without Apache-spark. Using Apache-Spark increases the demand on RAMs; using Spark to build the proposed DDoS attack-detection model urged us to improve the hardware used to run the code on Spark. Other relevant research works focus on accuracy measures and need more suitable time analysis, which is crucial in DDoS attack-detection applications; some other models provide less accuracy than the accuracy provided in this study.

Keywords

Ensemble model, Random forest (RF), XGBoost (XGB), Apache-spark, PySpark, Big data, CIC-DDoS2019, DDoS attacks.

1. Introduction

The term big data emerged in 2011. Over the years, it has been massively used in industry, media, commerce and research [1]. Big data is being created rapidly, and different types of data are generated and used: tabular data, text, images, videos, and others.

Machine learning (ML) is a science that is proposed to handle, analyze, and study data [2]. Classification, regression, and clustering are examples of ML tasks that are conducted on data. The bigger the data used to train such models, the better their yield. Nevertheless, feeding extensive (big) data into such models would increase the computation time needed. So, the state-of-the-art machine-learning models are not adequate anymore to deal with big data [3], and thus, researchers are stuck in the dilemma of the compromise between accuracy and performance.

Unique platforms, such as Apache-Spark, have been proposed in recent years to deal with big data in a way that preserves the accuracy as much as possible and, on the other hand, increases the model's performance [4]. Apache-Spark is a unified platform for large-scale data analytics, usually used with machine-learning models to distribute the data load on multiple machines running the same models. It is designed to enhance the scalability and computational speed needed for big data. The architecture of Apache-Spark is a hierarchical master/slave one, where a master node, running the cluster manager, is used to manage the distribution of data between the slave nodes, thus aggregating the results from

Y. Alslman, A. Khalil, R. Younisse, E. Alnagi, J. Al-Saraireh and R. Ghnemat are with Department of Computer Science, Princess Sumaya University for Technology, Amman, Jordan. Emails: yas20219006@std.psut.edu.jo, ash20219002@std.psut.edu.jo, r.baniyounisse@psut.edu.jo, ema20219005@std.psut.edu.jo, j.saraireh@psut.edu.jo and r.ghnemat@psut.edu.jo

these nodes. Such data splitting may compromise the accuracy of models conducted on this platform. Thus, further enhancement of the ML models is required.

An ensemble model is one proposed solution that enhances the results of several ML models working together, where the best of each is highlighted. Several methods are applied to merge the results of several ML models, such as averaging, voting, maximizing, and others. The literature review summarizes and discusses several papers that applied such models. Ensemble learning was mentioned in [5] as a powerful tool for intrusion-detection AI applications.

Distributed Denial of Service (DDoS) attacks are critical network attacks that can harm the whole network system if applied by attackers. The authors of [6] have created a dataset named CIC-DDoS2019 that resembles a real-time network flow during several DDoS attacks. Figure 1 illustrates the classifications of DDoS attacks tackled and studied in [5] using their generated dataset. DDoS attacks are widely studied and analyzed using different AI methods and scenarios. For example, the study in [7] has addressed the attacks in the context of IPv6 datasets and created a dataset containing the traffic information when a DDoS attack is applied on an IPv6 network.

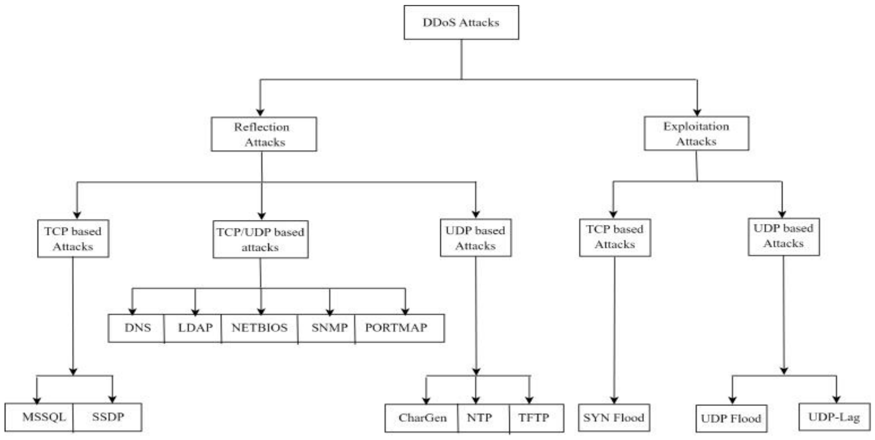

Figure 1. DDoS attack classifications [5].

The proposed work here investigates the effectiveness of ensemble DDoS attack-detection models by analyzing the accuracy, the F1-score and the training of XGBoost and RF models. The work then expands to using Apache-Spark to distribute the used dataset on different slaves to speed up the training process. The time reduction is then analyzed to detect how efficiently Apache-Spark can reduce the training time to build highly robust models.

An ensemble of ensemble models is proposed using Apache-Spark. Using an ensemble of ensemble models is often referred to as a stacking ensemble, an ML technique where multiple ensemble models are combined to make predictions. The essential advantage of using such a model is improving the performance of ML algorithms. In addition, the diversity of base ensemble models contributes to the model's robustness, mitigating the risk of overfitting, particularly in scenarios where individual models may underperform on specific subsets of data. Moreover, using stacked ensemble models can enhance adaptability, as it allows for incorporating various models. Figure 2 illustrates the proposed model architecture. Building the proposed model using Apache-Spark enables the efficient handling of large datasets. In other words, the proposed model combines the strengths of RF and XGBoost with Spark's distributed processing capabilities.

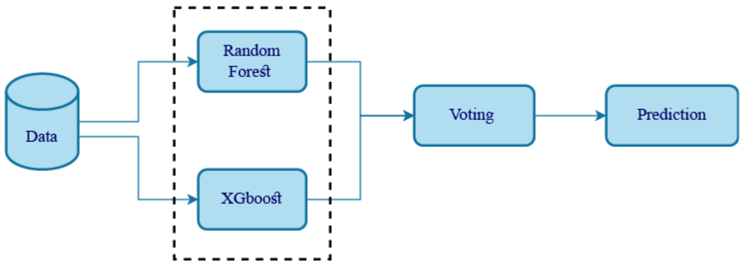

Figure 2. Proposed-model architecture.

The main contributions of this paper can be described in the following points:

1) Using a stacked ensemble model to detect DDoS attacks with high accuracy. The stacked ensemble model used is based on machine-learning algorithms (Random Forest and eXtreme Gradient Boosting), which are also considered ensemble algorithms of their own.
2) Reducing the time needed in the classification process by utilizing Apache-Spark to distribute the big dataset on several slaves to perform the targeted task.
3) The proposed model detects specific DDoS attacks, namely, DDoS-UDP, DDoS-MSSQL, and DDoS-SYN.
4) Due to the short training time needed to build the proposed model, the model can be trained and used to detect new or unknown attacks quickly compared to other approaches.

The rest of this paper is organized as follows: the relevant literature is surveyed and summarized in Section 2, and the research methodology is described in detail in Section 3. Then, in Section 4, the experiments that were conducted are discussed along with their results and evaluation. Finally, the last section presents a conclusion and avenues for future work.

2. Literature Review

The literature has extensively used machine-learning algorithms to perform classification and detection tasks in different research scopes. Random forest and XGBoost are standard algorithms that yield higher accuracy than others. This section concentrates on literature that uses either of these two algorithms separately or in ensemble models. The scope of the selected research has been concentrated on security, especially DDoS attack detection since it is the main scope of the current research.

Public datasets that resemble DDoS attacks are available and used in literature. CIC_DoS and CIC_IDS are noticeably used as valid and big datasets with their versions 2017, 2018, and 2019. Authors of [19][24][28][33] have used the 2017 version, while authors of [8] have used the 2018 version. As for the 2019 version, researchers in [13][28][30] used it along with other datasets. Other public datasets, such as the UNWS_np-15 [18] and NSL_KDD [24], have also been used as training datasets for DDoS attack detection.

Machine learning (ML) and deep learning (DL) models have been used to train these datasets to tackle the problem of DDoS attack detection. Several ML algorithms, such as Naive Bayes (NB) [13][20][29], Decision Tree (DT) [8][13][16][19][20][29], Random Forest (RF) [8][13], [17]-[20], [24], [27]-[29], K-Nearest Neighbors (KNN) [13][19][28], Support Vector Machine (SVM) [13][17], Gradient Boosting (GB) [16][21], Extreme Gradient Boosting (XGB) [13]-[14], [18]-[19], among others, are extensively used in literature. Neural Networks (NN) [22][27][29] and Long Short-term Memory (LSTM) [30]-[31] are examples of DL models that were also used in literature.

Some works have proposed ensemble models using various algorithms, either ML or DL [3][28]. Apache-Spark has been noticeably used in literature as an efficient way to split large datasets and apply parallel training [3], [20]-[22], [24]-[27], [29].

The following sub-sections will categorize literature according to the algorithms used for DDoS attack detection. So, the ones that focused on ML models are discussed in sub-section 2.1, those that used DL models in sub-section 2.2, and finally, those that used spark and/or ensemble models are described in sub-section 2.3. A final sub-subsection, 2.4, is added to address the research gap and limitations in previous literature.

2.1 Literature That Used Machine-learning Algorithms

Starting with the work proposed in [8], the ML models were trained with CIC-DDOS2018 and tested with CC-DDOS2018 to investigate how training the model with specific data and testing the model with another data version can affect the model's accuracy. The Decision Tree and the Random Forest model yielded testing accuracy results over 99.7%. Evaluating ML models with the CIC-DDOS datasets was presented in different works, such as [9]-[11]. Only some works focused on applying big-data approaches for the Intrusion Detection System (IDS) models to speed up the training process so they can be used and re-trained easily during a crisis. On the other hand, the work in [12] emphasized the usefulness of using Apache-Spark with the CIC-DDOS datasets and showed how it can reduce the training time of different models.

XGBoost with DDoS attacks has been discussed in [14], but with software-defined networks; the work presented an adaptive bandwidth profile-based threshold for attack detection and packet drop ratio reduction. It also presented a trigger-based detection and classification method that efficiently uses the XGBoost algorithm for traffic classification into normal or abnormal. Meanwhile, in [13], the CIC-DDoS 2019 dataset was evaluated using different machine-learning models: Naïve Bayes (NB), k-Nearest Neighbors (k-NN), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF) and Extreme Gradient Boosting (XGBoost). The XGBoost achieved higher performance than the other used ML models, while the NB model achieved minimal performance.

On the other hand, the XGBoost model was integrated with the adaptive bandwidth profile-based threshold [13] to detect attacks while minimizing the packet drop ratio. Hyperbolic functions, such as the fuzzy function and entropy, are used in the Euclidean distance-based multi-scale fuzzy entropy (EDM-Fuzzy) model in [15] to find the similarities between two vectors and handle the issues of the Heaviside function based on comparison between vectors. However, these models are not considered distributed systems. In contrast, the FEDFOREST [16] combines federated learning and Gradient Boosting Decision Tree (GBDT), an efficient framework for attack detection while achieving privacy.

In [17], the authors proposed a hybrid machine-learning system consisting of two algorithms: Support Vector Classifier (SVC) and Random Forest. Their goal was to detect benign traffic from DDoS attacks. In their work, they built the model to enhance the classification output from SVC, with further training using Random Forest. When using SVC, some instances may not be precise enough that they belong to a specific class; they may reside on the line that separates these classes and thus, the probability of belonging to the classes is equal. Here comes the role of the Random Forest classifier to give more precise decisions about the questionable instances. The authors created their dataset within a Software-defined Network (SDN) by monitoring the flow of incoming and outgoing switches in the network and gathering statistical information about the flow and ports in a CSV file. They have generated a large dataset of about 100,000 records and 23 features. They have compared their results with those of other machine-learning models and theirs have outperformed the others with an accuracy of 98.8%.

In [18], the authors also tackled the problem of DDoS attack classification. They have applied their experiments on the UNWS-np-15 dataset, with about 80,000 records and 45 features. Their proposed model used both the Random Forest classifier and the XGBoost classifier separately. The chosen classifiers yielded 89% and 90% accuracy, respectively. They have compared their results with those of other algorithms applied to different datasets. Their results have outperformed the ones working on the same dataset, UNWS-np-15.

As for [19], the authors examined several machine-learning algorithms to detect DoS and DDoS attacks using the CIC-IDS-2017 dataset. Decision Tree (DT), Random Forest (RF), K-Nearest Neighbors (KNN) and XGBoost (XGB) have been selected for their experiments. They evaluated the four algorithms using precision and recall as the primary evaluation metrics and they found that all four algorithms yielded high Recall, over 99%, with slight differences between them. The maximum Recall was achieved by XGB (99.87%), while KNN had the minimum Recall (99.52%). XGB, on the other hand, yielded the worst precision of 97.6%, while RF achieved 99.76% precision.

Enhancing the performance of the DDoS detection (normal and abnormal) system by reducing the misclassification error was the primary motivation for Alduailij et al. [28]. Two datasets were used in the proposed model (CICIDS2017 and CICDoS2019) for feature selection, mutual information and random forest feature importance. These features have been used in different machine-learning algorithms (Linear Regression LR, Random Forest RF, k-Nearest Neighbor KNN and Weighted Voting Ensemble WVE). It has been concluded that RF achieved the highest accuracy in all cases (when the number of the selected features was 19, 16, and 23).

2.2 Literature That Used Deep-learning Algorithms

A distributed learning environment deploying a neural network learning model was used [22]. The work distribution was achieved by using Spark. The specialty that [22] has gained is that the authors used their DDoS detection model to analyze the live traffic over the network and obtain the results, where they gained an accuracy result of 94%.

Deep-learning algorithms are also used in literature for DDoS attack detection. The authors of [30] have used a hybrid system of both BI-LSTM (Bi-directional) and Gaussian Mixture Model (GMM) to detect DDoS attacks using two datasets, CIC-IDS2017 and CIC-DDoS2019. Their experiments yielded up to a 94% accuracy score. As for [31], the authors have also used BI-LSTM and CNN to conform to a hybrid system of two deep-learning algorithms. They applied training and testing using the CIC-DDoS2019 dataset and yielded an accuracy of up to 94.52%.

2.3 Literature That Used Apache-Spark and Ensemble Models

In [20], the authors proposed a distributed system that takes advantage of the distribution to overcome the latency of multiple classification models. The time factor is an essential measure reflecting the strength of DDoS detection systems; the later the attack is discovered, the worse the consequences are. The classification models are Random Forest, Naive Bayes and Decision Tree. Spark was used to run the three models in parallel to fasten the flow packet classification process with the aim of DDoS attack detection. In contrast, fuzzy-logic rules were used to direct the packets to an algorithm based on traffic.

In the context of DDoS attacks, the Gradient boosting algorithm performance was taken to a new level [21] by parallelizing the gradient boosting algorithm, which was applied to classify packets based on potential DDoS attack threats carried with them. At the same time, the gradient boosting algorithm was supposed to build the machine-learning model out of multiple smaller decision trees and pass the packets through the tree branches to perform the classification process. Spark was used to accomplish the task in parallel, aiming to speed up the operation. The gradient boosting algorithm has enhanced the performance of the classification model. At the same time, Spark has distributed the operation over three slave machines and one master-machine architecture to enhance the time factor.

An unsupervised machine-learning algorithm was developed by [23] to create an analysis system to detect possible DDoS attacks over a vast amount of traffic. Through the analysis of 14 PCAP files recording the traffic on a network under a DDoS attack, the study aimed to reduce the required time to train and run the model for detection.

In [3], the authors compared the performance of two ensemble-learning models in the Spark environment. Through the study, they used the CIDDS dataset, which is used to train machine-learning models of intrusion-detection systems. The two models that were compared are the logistic regression-based blending ensemble and SVM-based blending ensemble and the study concluded that the latter outperformed logistic regression in terms of accuracy by 5%. The SVM-based blending ensemble model accuracy was recorded at 95%.

Distributed machine learning was used in [24] to detect the presence of concept drift in network traffic and detect network-based attacks. Spark archived the distribution. The study uses K-means clustering for detecting drift happening to the network traffic. Random Forest and Linear Regression models were used for intrusion detection. The datasets used in the study are the NSL-KDD dataset, the CIDDS-2017 dataset, and the generated Testbed dataset.

A real-time detection of DDoS attacks using machine-learning classifiers on a distributed-processing platform was proposed in [25]. The authors generated a traffic-simulating DDoS attack and fed the data features into different classifiers distributed over multiple Spark slave machines.

Online distributed denial-of-service attack detection using Spark streaming was studied in [26] and [27]. In the latter, the authors applied two machine-learning models, RF and MLP, for training and testing. Both models were applied with and without big-data approaches. Apache-Spark was deployed for the big-data approach.

Patil et al. [29] proposed a real-time network flow classification using a novel Spark streaming and Kafka-based classification system. The proposed model successfully classified the network flow into seven categories (Normal, DDoS-DNS, DDoS-LDAP, DDoS-MSSQL, DDoS-UDP, DDoS-SYN and DDoS-NetBIOS).

The CICDDoS dataset was used in training the model using four different machine-learning algorithms (RF, MLP, DT and NB), with RF being the best method in classifying the network flow with an accuracy of 89%. Table 1 displays a summary of most related literature reviewed.

Table 1. Literature-review summary.

Ref.#	**Model**	**Advantages**	**Limitations**
Manickam et al., 2022 [6]	Decision Tree, KNN, SVM, Naïve Bayes and CNN.	The evaluation of DDoS attack detection in the context of IPV6 networks.	The accuracy was not within satisfying levels.
De Araujo et al., 2021[12]	XGBoost.	Presented an efficient feature-selection method with the XGBoost model when used for DDoS attack detection applications.	Multi-class classifier accuracy was low.
Alamri et al., 2021 [14]	Several traditional machine-learning models: XGBoost, NB, k-NN, SVM, DT and RF.	Utilizing different types of machine-learning techniques to evaluate the performance of DDOS detection.	The authors did not consider the ensemble, distributed system. Training overhead time was mentioned.
Alamri et al., 2020 [13]	Bandwidth-control mechanism and XGBoost model.	High accuracy.	Using a bandwidth-control mechanism reduces the accuracy of the multi-classification task using XGBoost compared with previous works. Furthermore, the distributed system was not considered.
Zhou et al., 2021 [15]	Euclidean Distance-based Multi-scale Fuzzy Entropy (EDM-Fuzzy).	Achieved training stability by handling the traditional distance-computation issues, such as bouncing between 0 and 1 produced by the Heaviside function.	The authors did not use distributed systems and self-learning classifiers, such as deep learning to train the large dataset.
Dong et al., 2022 [16]	Federated Learning and Gradient Boosting Decision Tree.	Achieved data privacy and utilized a distributed system.	It is sensitive to hyper-parameter tuning.
Ahuja et al., 2021 [17]	Hybrid system consisting of SVC and RF.	Created their dataset or about 100,000 records and 23 features. They have enhanced the results of SVC by using RF as a first decision layer.	The authors have only tested their dataset over binary classes to detect benign traffic.
Mohmand et al., 2022 [18]	RF and XGBoost.	Used UNWS-np-15 Dataset with 80,000 records and 45 features.	Although the results have outperformed those of others who used the same datasets, but the percentage of 90% is still low compared with other research in the same scope.
Zewdie and Girma, 2022 [19]	DT, RF, KNN and XG-Boost.	Used CIC-IDS-2017 dataset, each algorithm has yielded a high Recall score separately.	Some algorithms have yielded better Recall than others; thus, merging them into an ensemble model could yield higher scores.
Alsirhani et al., 2018 [20]	RF, NB and DT.	Using Spark to run the three models in parallel, a fast classification process, using Fuzzy logic to decide the best algorithm to be used.	There is a trade-off between accuracy and speed, which lowers the accuracy of the models.
Alsirhani et al., 2018 [21]	Gradient Boosting Algorithm GBT.	Using Spark to enhance the performance of the model, two datasets are used for experiments.	Datasets used contains only two classes.
Hsieh and Chan, 2016 [22]	Neural Network.	Analysis of live traffic over the network using Spark.	Accuracy is comparatively low to other research 49%
Jain and Kaur, 2021 [24]	RF, LR and SVM.	Two datasets are used to generate Testbed dataset.	97% accuracy for the ensemble model.
Alduailij et al., 2022 [28]	Several machine-learning algorithms: LR, RF, WVE and KNN.	High accuracy with a minimum number of features.	Using the proposed model as a detection model (normal and abnormal) rather than classification.
Patil et al., 2022 [29]	Using four different machine-learning algorithms (RF, MLP, DT and NB).	Provided a real-time net-workflow classification using a novel Spark streaming and Kafka-based classification system.	Low accuracy (89%) when comparing it to the literature.
Chartuni et al., 2021[32]	Multi-class CNN classifier composed of seven layers.	Multi-class classifier with high accuracy.	The model was not evaluated with a computer network the flow of which was not previously seen by the model.

In [33], the authors investigated DDoS attack detection in the context of Internet of Things (IoT). The work uses an ensemble model of different ML models to enhance the detectability of the attacks.

In the work presented in [34], the authors highlighted the importance of using deep-learning models in IDSs and suggested using Apache-Kafka stream and distributing the data on multiple workstations. The work analysis is conducted on online streamed data. It shows that the proposed technique surpasses the baseline strategies by 11% in the F1-measure when the number of workers is two and by 25% when the number of workers is equal to 32.

In their research paper, the authors of [35] developed a new model named Stacked Convolutional Neural Network and Bidirectional Long Short-Term Memory (SCNN-Bi-LSTM) for the purpose of intrusion detection in wireless sensor networks. The work utilizes Spark to distribute the workload among multiple nodes. The model was able to achieve an impressive classification accuracy of 99.9%. However, the paper does not delve into the details of how the use of Spark has improved the model's accuracy or the time required for the work.

2.4 Research Gaps and Limitations

The previously discussed literature that focused on creating IDSs against DDoS attacks did consider time. Many works presented binary IDS models rather than multi-class models. Additionally, the multi-class classification models achieved lower accuracy than the accuracy achieved in this work.

Hence, in this work, we focus on building an ensemble model of two ensemble models, the Random Forest and the XGBoost, to prepare an intrusion-detection system that can efficiently operate with big datasets in a reasonable amount of time. No feature selection was used on the datasets, since there is no need for further pre-processing techniques as long as the proposed model accuracy is already sufficiently high. Compared to other works, this work can be considered a simple approach to building an efficient ensemble multi-class DDoS attack-detection model, which requires simple pre-processing steps and provides the classification results in an easily understandable form. The presented model can easily be used with similar datasets concerning the multi-class classification in the context of DDoS attacks. The methodology followed in this work is discussed in the next section.

Most of the mentioned works focus on analyzing the accuracy results or the false-positive measures of the proposed DDoS attack-detection models while the required time was not analyzed or mentioned. In this work we believe that decreasing the time required for detecting such attacks is very critical, since these attacks are based on paralyzing network systems by establishing unnecessary communications with the systems. Hence, detecting these attacks early makes them much less harmful. The accuracy is critical, yes, but with a very large dataset such as CIC-DDoS2019, building an accurate detection system is not as challenging as decreasing the detection time.

3. METHODOLOGY

This section discusses the methodology used to build the proposed model. It starts from the dataset selection, goes through its pre-processing and ends with implementing the proposed model using Apache-Spark. This paper proposes an ensemble model of random forest and XGB regressor for multiclass classification using Apache-Spark.

As mentioned before, the main reason for choosing such algorithms is to build a stacked ensemble model, which is an ensemble model built using ensemble-based models, such as random forest and XGB regressor.

Spark typically results in enhanced performance, scalability and the ability to handle larger datasets efficiently. Its impact on the model's development and deployment can be significant, providing a competitive edge in handling big data and complex analytics tasks. Spark's ability to distribute data processing across a cluster of machines is crucial for handling large datasets. Its in-memory processing capability accelerates computations. Spark was also used due to its adept at stream processing *via* Spark streaming, enabling real-time analytics on data streams. Furthermore, Spark supports various programming languages (like Scala, Java and Python), making it accessible and convenient. Figure 2 shows the proposed stacked ensemble model used on the CIC-DDOS2019 dataset.

It can be noticed that after selecting that data, it will be passed to two ensemble models; namely, random forest and XGB regressor. Afterwards, the resulting predictions from both models will be passed to a voting mechanism, which in turn produces the final result. The steps below summarize the basic steps of building the proposed model.

1. Read the dataset into a DataFrame df_pyspark.
2. Pre-process the dataset.
3. Split data into training and testing subsets using the specified train_test_split ratio.
4. Train a Random Forest classifier model on the training subset.
5. Train the XGB regressor classifier model on the training subset.
6. Evaluate the Random Forest classifier model on the testing subset.
7. Evaluate the XGB regressor classifier model on the testing subset.
8. Combine predictions from both models to form the ensemble prediction.
9. Evaluate the final result using the evaluation metrics.

3.1 Dataset Selection

The Distributed Denial of Service (DDoS) attack uses malicious traffic to exhaust the target networks, where the CIC-DDoS2019 dataset is the latest released version of the DDoS datasets. Each record consists of features indicating the traffic status, whether it is an attack or not. The CIC-DDoS2019 collects many malicious and normal traffic cases collected in two days. In 2019, the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick created this dataset. The dataset was created to have a more professionally engineered and diversified set of DDoS detection attacks. These attacks are used to evaluate the proposed ensemble model. In the SYN attack, the attackers aim to send a large number of SYN (synchronization) requests; TCP packets are used to connect to a server to overwhelm it with an open connection or a half-open connection, which aims to overwhelm the targeted server with fake connections. On the other hand, the UDP attack overwhelms random ports on the targeted host with IP packets containing UDP datagrams, making the targeted server unable to process the flood of arriving packets and serve legitimate users. MSSQL is a web-application attack that uses a bad application design that does not sanitize inputs, exposing application vulnerabilities.

The dataset is one of the most comprehensive and used datasets in the scope of building DDoS attack-detection models. The main limitation in this dataset is the existence of NaN and infinity values among many tuples. This problem was solved in this work by replacing these values with zeros. The other limitation is its big size, which means that building an IDS system using the dataset will require a long time. This limitation is solved by using Spark to reduce the training and testing times of the proposed IDS system.

The dataset has been organized in several CSV files that consist of millions of records that present different types of attacks. As shown in Figure 3, two files have been selected: Syn-file and UDP-file, which consist of about eight million records, considered significant for the proposed model. The files for these attacks were chosen to be analyzed through this study, because the study focuses on multi-class-classification problems when massive datasets are used. These attacks have around eight million records and they are famous, repeatedly discussed in the literature and can cause serious harm to the attacked systems.

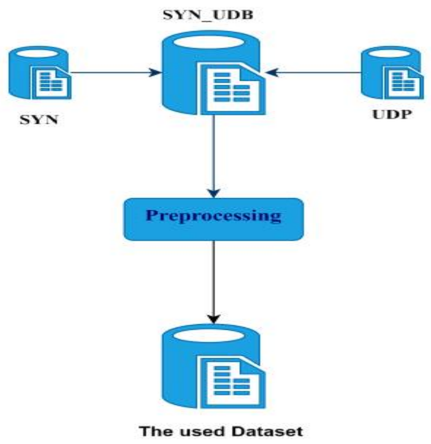

Figure 3. The data-selection method.

This dataset is chosen for this research, because it is one of the most recent and essential datasets in the IDS field. It is widely used in other studies; hence, the comparison can be valid. Moreover, since the research focuses on using IDS with large datasets, this dataset is a convenient choice.

3.2 Random Forest

The first supervised machine-learning technique used in the proposed ensemble model is the random forest (RF). The RF is a robust machine-learning model that reduces overfitting and performs efficiently on large datasets. The RF randomly divides the dataset's features into sub-sets of features, where each sub-set is trained using the decision-tree model separately and independently of other sub-trees. In the training process, each dataset sample is trained in a particular sub-tree, while in the testing, the entire test data is trained in each sub-tree. The final result is aggregated by producing an average of these results. The decision-tree model consists of nodes and branches. At each node, evaluate the sub-set of features to generate and divide the observations into other nodes in the training set or to flow to a specific path when making a prediction.

3.3 eXtreme Gradient Boosting (XGBoost)

The second robust algorithm that handles the bias-variance trade-offs and provides a parallel tree boosting for classification is the XGBoost model. Like the RF algorithm, the XGBoost model uses gradient boosting, providing adaptation and generalization. It is considered an ensemble of multiple learners, such as decision trees, where the final decision is an ensemble of the subtrees' outputs. Consequently, the XGBoost prevents poor performance. XGBoost uses the gradient descent algorithm to optimize the model by updating weight and reducing cost value and the discrepancy between the expected and actual values. The mean squared error (MSE) cost function is used as an evaluation metric for classification tasks.

In the next section, the experiments are discussed by starting with the data pre-processing phase, then concluding the results of experiments with and without Spark, along with an evaluation and discussion of these results.

4. Experiment Results

4.1 Data Pre-processing and Experiment Setup

Data pre-processing is crucial in the data analysis and machine-learning pipeline. It contributes to data quality, integrity and suitability for modeling, leading to more accurate and reliable results.

As aforementioned, the CSV files selected from the CIC-DDoS2019 dataset for evaluation in the proposed model were UDP.csv and SYN.csv. The UDP file consists of three classes: UDP attack, MSSQL attack and benign. At the same time, the SYN file consists of two classes: SYN attack and benign. These two files are under exploitation attacks, as shown in Figure 1. Table 2 represents the number of samples for each class before and after the pre-processing phase. The data pre-possessing consists of the following steps:

1) Eliminating duplicate records using drop_doplicate function provided by Python, where the number of samples for SYN and Benign classes is reduced from 4,284,751 to 3,806,356 and from 35,790 to 31,386, respectively, as shown in Table 2. However, there were no duplicate records for UDP and MSSQL. This step is essential, as it helps maintain data quality and consistency. Duplicate records can skew analysis results, leading to biased models or overfitting problems.
2) Eliminating attributes with summation and variances of zero can lead to more efficient, generalizable models and it is a common data pre-processing step. These attributes are: Bwd Packet Length Std, Bwd PSH Flags, Fwd URG Flags, Bwd URG Flags, FIN Flag Count, PSH Flag Count, ECE Flag Count, Fwd Avg Bytes/Bulk, Fwd Avg Packets/Bulk, Fwd Avg Bulk Rate, Bwd Avg Bytes/Bulk, Bwd Avg Packets/Bulk, Bwd Avg Bulk Rate and Similar HTTP. This step reduces the number of features from 88 to 74.
3) Replacing infinity and null values with zeros using replace(nan, 0) function provided by numpy module in Python.

4) Encoding categories including source IP, flow ID, destination IP and timestamp using LabelEncoder function provided by sklearn module in Python. This step is crucial, since many machine-learning algorithms work with numerical data; so, categorical variables must be encoded into a suitable numerical format.

Table 2. The number of records for each class.

Class	Samples count before pre-possessing	Samples count after pre-possessing
UDP attack	3754680	3754680
MSSQL attack	24392	24392
SYN attack	4284751	3806356
Benign (no attack)	38924	34520

The dataset was split randomly into 80% for training and 20% for testing. Consequently, the number of records in training and testing sets are 6095958 and 1523990, respectively. Furthermore, the default parameters of the Random Forest and XGBoost models were determined as follows: The number of sub-trees of both models is 100 and the weak learner in the XGBoost is the decision trees. Table 3 and Table 4 show the hyper-parameters of RF and XGBoost used to train the model.

Table 3. Random forest hyper-parameters.

RF hyper-parameter	Value
Number of trees.	100
Function to measure the quality of a split.	gini
The minimum number of samples required to split an internal node.	2
The minimum number of samples required to be at a leaf node.	1

Table 4. XGBoost hyper-parameter.

XGboost hyper-parameter	Value
Number of boosting rounds.	100
Loss function	Squared error
Boosting model (weak learner)	Decision trees (gbtree)
The feature-importance type	Information gain
Maximum depth	6

4.2 Performance Metrics

To evaluate the proposed model, four metrics were utilized: accuracy, precision, recall and time. The assessment was conducted both with and without the Spark platform. Accuracy indicates the percentage of correctly classified instances out of the total number of cases evaluated, as per Equation 1, while time denotes the training time in minutes. Meanwhile, the ensemble model employing Apache-Spark was evaluated through a classification report that includes precision, recall and F1-Score. The formula for calculating the measures are presented in Equations 1-3.

$$\text{Accuracy} = \text{TP+TN/(TP+TN+FP+FN)} \tag{1}$$

$$\text{Precision} = \text{TP/(TP+FP)} \tag{2}$$

$$\text{Recall} = \text{Precision.Recall/Precision+Recall} \tag{3}$$

TN stands for True Negative, FP for False Positive, TP for True Positive and FN for False Negative. These measures were utilized due to data imbalance, with F1-Score being commonly used to evaluate IDSs, as it combines precision and recall and provides valuable insights into the study outcomes.

The value of each measurement ranges from 0 to 1, with higher values indicating a more robust model that is better suited for detecting possible intrusions. Time measures were utilized to compare the performance of the proposed model with and without Spark, highlighting the value added by Spark usage. With Spark, the required times for building and training the model, as well as for running the detection model, are significantly reduced.

4.3 Experiments and Results

Two experiments have been conducted. Apache-Spark's distributed computing capabilities make it a valuable tool for handling large datasets. Its scalability, in-memory processing and distributed model enable it to process vast amounts of data efficiently. However, one of its potential limitations is the resource requirements. As a result, both experiments have been implemented on Colab Pro with (25 GB) RAM; the first experiment was done without Apache-Spark by applying the RF and XGBoost ensemble model.

The same model with the same environment has been applied again but with the use of Apache-Spark using the PySpark library, considered an open-source interface for Apache-Spark. It allows SQL-like analysis on large amounts of structured or semi-structured data. Figure 4 and Table 5 illustrate the training time needed to conduct Random Forest and XGBoost separately, once under the Apache-Spark environment and the other without using it.

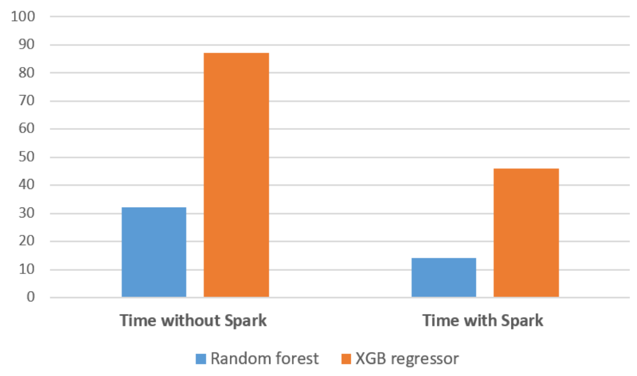

Figure 4. The training time required to train random forest and XGBoost models with/without Spark.

Table 5. The training time using the CIC-DDOS 2019 dataset.

Model	Training time without Spark in minutes	Training time with Spark in minutes
Random forest	32	14
XGB regressor	87	46

Table 6 illustrates the accuracies from conducting Random Forest, XGBoost and the proposed ensemble model, either with Spark or without it. It can be noticed how the results are close in both cases. As for Figure 5, the Recall, precision and F1-score measures are compared for classes SYN, MSSQL, UDP and Normal when conducting the proposed ensemble model.

Table 6. The accuracy using the CIC-DDOS 2019 dataset.

Model	Accuracy without Spark (%)	Accuracy with Spark (%)
Random forest	99.9995	99.9155
XGB regressor	99.9762	99.9942
Ensemble	99.94	99.9419

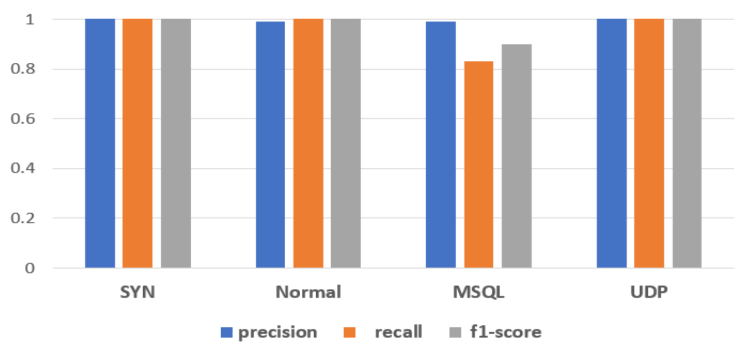

Figure 5. The performance of the proposed ensemble model for each class.

Table 7 conducts a comparison between selected proposed models in the literature. The comparison has been conducted regarding accuracy, F1-score, number of classes predicted and whether Spark is used or not. In reference [29], the F1-measure was calculated by averaging the F1-scores for the seven classes mentioned in that work.

Table 7. Comparison between the proposed model and literature.

Model	Accuracy (%)	F1-score	No. of classes	Spark
Extreme Gradient Boosting (XGBoost) [12]	99.7	99.79	two classes	-
XGBoost [13]	99.7	100	two classes	-
XGBoost [13]	91.26	92	multi-class	-
FEDFOREST (L+L) [16]	67.03	94.60	two classes	-
Random forest [28]	99.99	-	two classes	-
Parallel NB, DT and RF [20]	61.4, 97.9 and 97.3	51.3, 97.9 and 9.73	two classes	✓
Random forest [29]	89	89	seven classes	✓
CNN [32]	94.21	94.12	seven classes	-
Proposed (ensemble -RF and XGBoost-)	99.94	97.5	four classes	✓

The code that we have written and used throughout this study is publicly available in [36].

4.4 Discussion

The accuracy of the proposed model has been one of many concerns during the extraction of experiment results. The training time has been considered one vital criterion to concentrate on to prove our work's high performance and validity.

Table 3 and Figure 4 show that the time needed to train either a Random Forest or an XGBoost model has been reduced to about a half when using Spark. The required time to train Random Forest and XGBoost using Spark is reduced by 18 and 41 minutes, respectively. This time reduction makes the model more efficient, since detecting DDoS attacks is usually deployed in sensitive applications where time matters. Expanding the required time to train and use the model makes it less reliable and less usable in such applications.

From Table 4, it can be noted that the proposed ensemble model achieved a high accuracy regardless of whether or not Spark was used. Table 4 proves that even splitting the dataset in a way that Spark can use did not affect the accuracy of the proposed model. Thus, the model's performance has been enhanced while preserving its accuracy.

Figure 5 illustrates that both RF and XGBoost, working as a stacked ensemble model, can successfully distinguish the SYN, UDP, MSQL and benign. However, Recall and F1-score have been slightly reduced when using Spark in MSSQL attack. This is because of the number of its samples, which is considered small compared with the rest of the classes, as represented in Table 2. Thus, splitting and training small samples may yield less F1-score for such classes.

The proposed model achieves the best accuracy when compared with other models in the literature. Table 5 shows that the proposed model exceeds the existing models in terms of accuracy when the model is trained in more than two classes. It also outperformed the accuracy of models that classified only two classes, considering that binary classification is usually more accessible than multi-class classification.

Overall, the combination of stacked ensemble models under the Apache-Spark environment outperformed other models described in the literature.

5. Conclusion

This work proposed a distributed ML model for DDoS attack detection using Apache-Spark. Ensemble learning was used to build a robust and efficient IDS. The system can detect three DDoS attacks: UDP, MSSQL and SYN. The model comprises two trusted ML models: Random Forest and

XGB regressor. Hence, it can be considered a stacked ensemble model. Apache-Spark was used to train data distribution in parallel using the proposed model. Data distribution using Spark has enhanced the required time to train the model, as the time was reduced to around a half. At the same time, the accuracy was preserved at a level of over 99%. The required training time for the XGBoost regression model was reduced from 87 minutes to 46 minutes when Spark was used. The required training time for the Random Forest model was reduced from 32 to 14 minutes when Spark was used. The time reduction comes at the cost of increasing the used RAMs, which is considered the main limitation of the proposed approach in this work. This limitation can be avoided by using computers with large RAM capacity.

The presented methodology can be considered an efficient IDS approach that can be used with DDoS attacks, such as the attacks the data of which is recorded in the CIC-DDOS2019 dataset. The presented distributed model is also robust and fast. Hence, it can be used when online intrusion-detection models are not affordable, complicated or down. Using an ensemble IDS of XGBoost and Random Forest with Apache-Spark has proved to be an easily built and trained model. The approach guarantees short training time and robustness against failure; when one distributed node in the Apache-Spark platform is down, the other nodes are available and a replacement node can take over the failed one.

In the future, more ML models will be added to the distributed ensemble model for the DDoS IDS by editing the used code and adding more slaves (nodes) to the ensemble model. Furthermore, the work will be expanded to detect other intrusions by training the model with datasets related to other attacks.

REFERENCES

[1] J. S. Ward and A. Barker, "Undefined by Data: A Survey of Big Data Definitions," arXiv preprint, arXiv: 1309.5821, 2013.

[2] M. I. Jordan and T. M. Mitchell, "Machine Learning: Trends, Perspectives and Prospects," Science, vol. 349, no. 6245, pp. 255-260, 2015.

[3] G. Kaur and M. Jain, "A Comparison of Two Blending-based Ensemble Techniques for Network Anomaly Detection in Spark Distributed Environment," Int. J. of Ad Hoc and Ubiquitous Computing, vol. 35, no. 2, pp. 71-83, 2020.

[4] M. Zaharia et al., "Apache Spark: A Unified Engine for Big Data Processing," Communications of the ACM, vol. 59, no. 11, pp. 56-65, 2016.

[5] I. Sharafaldin, A. H. Lashkari, S. Hakak and A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy," Proc. of the 2019 IEEE Int. Carnahan Conf. on Security Technology (ICCST), pp. 1-8, Chennai, India, 2019.

[6] S. Manickam et al., "Labelled Dataset on Distributed Denial-of-Service (DDoS) Attacks Based on Internet Control Message Protocol Version 6 (ICMPv6)," Wireless Communications and Mobile Computing, vol. 2022, Article ID 8060333, DOI: 10.1155/2022/8060333, 2022.

[7] T. H. Chua and I. Salam, "Evaluation of Machine Learning Algorithms in Network-based Intrusion Detection Using Progressive Dataset," Symmetry, vol. 15, no. 6, p. 1251, 2023.

[8] B. I. Farhan and A. D. Jasim, "Performance Analysis of Intrusion Detection for Deep Learning Model Based on CSE-CIC-IDS2018 Dataset," Indonesian Journal of Electrical Engineering and Computer Science, vol. 26, no. 2, pp. 1165-1172, 2022.

[9] A. Elhanashi, K. Gasmi, A. Begni, P. Dini, Q. Zheng and S. Saponara, "Machine Learning Techniques for Anomaly-based Detection System on CSE-CIC-IDS2018 Dataset," Proc. of the Int. Conf. on Applications in Electronics Pervading Industry, Environment and Society (ApplePies 2022), Part of the Lecture Notes in Electrical Engineering Book Series, vol. 1036, pp. 131-140, Springer, 2022.

[10] I. F. Kilincer, F. Ertam and A. Sengur, "A Comprehensive Intrusion Detection Framework Using Boosting Algorithms," Computers and Electrical Engineering, vol. 100, p. 107869, 2022.

[11] R. Atefinia and M. Ahmadi, "Performance Evaluation of Apache Spark MLlib Algorithms on an Intrusion Detection Dataset," arXiv preprint, arXiv: 2212.05269, 2022.

[12] P. H. H. N. de Araujo et al., "Impact of Feature Selection Methods on the Classification of DDoS Attacks using XGBoost," Journal of Communication and Information Systems, vol. 36, no. 1, pp. 200-214, 2021.

[13] H. A. Alamri and V. Thayananthan, "Bandwidth Control Mechanism and Extreme Gradient Boosting Algorithm for Protecting Software-defined Networks against DDoS Attacks," IEEE Access, vol. 8, pp. 194269-194288, 2022.

[14] H. A. Alamri and V. Thayananthan, "Analysis of Machine Learning for Securing Software-defined Networking," Procedia Computer Science, vol. 194, pp. 229-236, 2021.

[15] R. Zhou, X. Wang, J. Yang, W. Zhang and S. Zhang, "Characterizing Network Anomaly Traffic with

Euclidean Distance-based Multiscale Fuzzy Entropy," Security and Communication Networks, vol. 2021, Article ID 5560185, DOI: 10.1155/2021/5560185, 2021.

[16] T. Dong, S. Li, H. Qiu and J. Lu, "An Interpretable Federated Learning-based Network Intrusion Detection Framework," arXiv preprint, arXiv: 2201.03134, 2022.

[17] N. Ahuja, G. Singal, D. Mukhopadhyay and N. Kumar, "Automated DDOS Attack Detection in Software Defined Networking," Journal of Network and Computer Applications, vol. 187, p. 103108, DOI: 10.1016/j.jnca.2021.103108, 2021.

[18] M. I. Mohmand et al., "A Machine Learning-based Classification and Prediction Technique for DDoS Attacks," IEEE Access, vol. 10, pp. 21443-21454, 2022.

[19] T. G. Zewdie and A. Girma, "An Evaluation Framework for Machine Learning Methods in Detection of DoS and DDoS Intrusion," Proc. of the 2022 IEEE Int. Conf. on Artificial Intelligence in Information and Communication (ICAIIC), pp. 115-121, Jeju Island, Korea, 2022.

[20] A. Alsirhani, S. Sampalli and P. Bodorik, "DDoS Attack-detection System: Utilizing Classification Algorithms with Apache Spark," Proc. of the 2018 9th IEEE IFIP Int. Conf. on New Technologies, Mobility and Security (NTMS), pp. 1-7, Paris, France, 2018.

[21] A. Alsirhani, S. Sampalli and P. Bodorik, "DDoS Detection System: Utilizing Gradient Boosting Algorithm and Apache Spark," Proc. of the 2018 IEEE Canadian Conf. on Electrical & Computer Engineering (CCECE), pp. 1-6, Quebec, Canada, 2018.

[22] C. J. Hsieh and T. Y. Chan, "Detection DDoS Attacks Based on Neural-Network Using Apache Spark," Proc. of the 2016 IEEE Int. Conf. on Applied System Innovation (ICASI), pp. 1-4, Okinawa, Japan, 2016.

[23] K. Kato and V. Klyuev, "Development of a Network Intrusion Detection System Using Apache Hadoop and Spark," Proc. of the 2017 IEEE Conf. on Dependable and Secure Computing, pp. 416-423, Taipei, Taiwan, 2017.

[24] M. Jain and G. Kaur, "Distributed Anomaly Detection Using Concept Drift Detection Based Hybrid Ensemble Techniques in Streamed Network Data," Cluster Computing, vol. 24, pp. 2099-2114, 2021.

[25] S. Gumaste, D. G. Narayan, S. Shinde and K. Amit, "Detection of DDoS Attacks in OpenStack-based Private Cloud Using Apache Spark," Journal of Telecommunications and Information Technology, vol. 2020, no. 4, pp. 62-71, 2020.

[26] B. Zhou, J. Li, J. Wu, S. Guo, Y. Gu and Z. Li, "Machine-learning-based Online Distributed Denial-of-Service Attack Detection Using Spark Streaming," Proc. of the 2018 IEEE Int. Conf. on Communications (ICC), pp. 1-6, Kansas City, USA, 2018.

[27] M. J. Awan et al., "Real-time DDoS Attack Detection System Using Big Data Approach," Sustainability, vol. 13, no. 19, p. 10743, 2021.

[28] M. Alduailij, Q. W. Khan, M. Tahir, M. Sardaraz, M. Alduailij and F. Malik, "Machine-learning-based DDoS Attack Detection Using Mutual Information and Random Forest Feature Importance Method," Symmetry, vol. 14, no. 6, p. 1095, 2022.

[29] N. V. Patil, C. R. Krishna and K. Kumar, "SSK-DDoS: Distributed Stream Processing Framework Based Classification System for DDoS Attacks," Cluster Computing, vol. 25, no. 2, pp. 1355-1372, 2022.

[30] C. S. Shieh et al., "Detection of Unknown DDoS Attacks with Deep Learning and Gaussian Mixture Model," Applied Sciences, vol. 11, pp. 11, p. 5213, 2021.

[31] D. Alghazzawi, O. Bamasag, H. Ullah and M. Z. Asghar, "Efficient Detection of DDoS Attacks Using a Hybrid Deep Learning Model with Improved Feature Selection," Applied Sciences, vol. 11, no. 24, p. 11634, 2021.

[32] A. Chartuni and J. Márquez, "Multi-classifier of DDoS Attacks in Computer Networks Built on Neural Networks," Applied Sciences, vol. 11, no. 22, p. 10609, 2021.

[33] Y. Yilmaz and S. Buyrukoglu, "Development and Evaluation of Ensemble Learning Models for Detection of Distributed Denial-of-Service Attacks in Internet of Things," Hittite Journal of Science & Engineering, vol. 9, no. 2, pp. 73-82, 2022.

[34] M. Seydali, F. Khunjush and J. Dogani, "Streaming Traffic Classification: A Hybrid Deep Learning and Big Data Approach," Cluster Computing, DOI: 10.1007/s10586-023-04234-0, 2024.

[35] S. M. S. Bukhari et al., "Secure and Privacy-preserving Intrusion Detection in Wireless Sensor Networks: Federated Learning with SCNN-Bi-LSTM for Enhanced Reliability," Ad Hoc Networks, vol. 155, p. 103407, 2024.

[36] "ColabCode," [Online], Available: Https://Colab.Research.Google.Com/Drive/1oZu2czCK9tJSwcjEfyv LiZnrI0JqYW62?Usp=Sharing, December 22, 2023.

ملخص البحث:

إنّنا نعيش في عَصْرٍ يُعَدّ فيه الوقت المورِد الأغلى. لذا، فإنّ التّعاملَ مع الكمّ الهائل من البيانات التي تُجمّع من مصادر مختلفة لأغراض مختلفة يتطلّب إيجاد أنظمةٍ يمكنها معالجة البيانات بشكلٍ صحيح يجعلها ذات معنى. وإنّ استخدام البيانات الضّخمة في نماذج تعلُّم الآلة والذّكاء الاصطناعي من شأنه أن يُحسِّن فعالية تلك النّماذج ومتانتها.

تقترح هذه الورقة نموذجاً لكشْف الهجمات الموزّعة المتعلّقة برفض الخدمة (DDoS) باستخدام مجموعة بيانات عامّة معروفة لتدريب النّموذج. ويتمّ تدريب النّموذج بحيث يُمكنه توقُّع نوع الهجمة من بين أنواع متعدّدة من الهجمات يستخدمها المخترقون. وتُستخدم اثنتان من الخوارزميات الواردة في أدبيات الموضوع بوصفهما أساسَ النّموذج المقترح. وتستمدّ هاتان الخوارزميتان فعاليتهما ومتانتهما من الطبيعة المجمَّعة لتركيبهما، حيث تتكون كلٌّ منهما من عددٍ من شجرات القَرار (decision trees) بمتغيرات مختلفة. وقد جرى بناء نظام مجمَّع باستخدام الخوارزميتين من أجل تحسين دقّة كشْف الهجمات؛ واتّضح أنّ استخدام تلك التركيبة المجمَّعة يعطي أفضل النتائج.

من ناحيةٍ أخرى، فإنّ طول زمن التّنفيذ المترتِّب على تدريب النّموذج باستخدام مجموعة بيانات ضخمة يُعدّ مسألةً أخرى لا بدّ من أخذها بعين الاعتبار. ولتسريع عمل النّموذج، فقد تمّ استخدام "الشّرارة" (Apache-Spark) الّتي يؤدي استخدامها إلى تجزئة مجموعة البيانات ومعالجة تلك الأجزاء بالتّوازي، الأمر الّذي يقلّل زمن التّنفيذ مع المحافظة على دقّة كشْف الهجمات.

لقد حقق النّموذج المقترح دقّةً وصلت إلى 99.94%، وقلّل استخدام الشّرارة زمن التّنفيذ إلى ما يقرب من النّصف مقارنة بعدم استخدام الشّرارة. وبالمقارنة مع عددٍ من نماذج كشْف الهجمات الواردة في أدبيات الموضوع، تبيّن أنّ تلك النّماذج حققّت دقّة كشفٍ أقلّ من النّموذج المقترح في هذه الدّراسة.