Machine Learning in OpenFlow Network: Comparative Analysis of DDoS Detection Techniques

Arun Kumar Singh
College of Computing and Informatics, Saudi Electronic University, Kingdom of Saudi Arabia

Abstract: *Software Defined Network (SDN) allows the separation of a control layer and data forwarding at two different layers. However, centralized control systems in SDN is vulnerable to attacks namely Distributed Denial of Service (DDoS). Therefore, it is necessary for developing a solution based on reactive applications that can identify, detect, as well as mitigate the attacks comprehensively. In this paper, an application has been built based on machine learning methods including, Support Vector Machine (SVM) using Linear and Radial Basis Function kernel, K-Nearest Neighbor (KNN), Decision Tree (DTC), Random Forest (RFC), Multi-Layer Perceptron (MLP), and Gaussian Naïve Bayes (GNB). The paper also proposed a new scheme of DDOS dataset in SDN by gathering considerably static data form using the port statistic. SVM became the most efficient method for identifying DDoS attack successfully proved by the accuracy, precision, and recall approximately 100 % which could be considered as the primary algorithm for detecting DDoS. In term of the promptness, KNN had the slowest rate for the whole process, while the fastest was depicted by GNB.*

Keyword: *Support vector machine, software defined network, machine learning, distributed Dos, detection.*

Received May 6, 2020; accepted September 9, 2020
https://doi.org/10.34028/iajit/18/2/11

1. Introduction

The development of computer network technology is pushing some of the core sectors in the network to make radical changes in order to meet the trends of computer networks that are needed today including agile, flexible, and versatile. One common problem that must be analyzed carefully is network management. With the continued increase in computer network devices (Internet of Things), efforts in managing network traffic will increase linearly. In a traditional computer network, a network administrator must manually configure each layer 2 and layer 3 device if there are problems with the device. This must be done because there are different configurations set by each network device vendor with specific procedures [17]. For example, to configure a router with a Cisco vendor it will be different from a Juniper router. Complex computer network trends can trigger misconfiguration or human errors.

Against this background, a computer network paradigm was developed called Software Defined Network (SDN). SDN separates vertical abstractions in traditional network devices consisting of 2 core layers, the controller layer and the forwarding layer. The controller acts as the center of the network settings that are connected directly to the device. Various network management functions can be flexibly managed by installing applications, for example, the routing process, the mapping of network topology, the intrusion detection system, the discovery of the Internet Protocol (IP), the discovery of the Address Resolution Protocol (ARP), and other special functions built with program lines in the application layer on the controller. The application that has been installed in the controller will process each incoming packet and respond according to the algorithm that has been defined [2]. The implementation of a centralized control system provides a security loophole that can be exploited by irresponsible parties to be able to carry out attacks aimed at broadly impacting the topology that is directly connected to a controller. One example of an attack that can be carried out both locally and globally is Distributed Denial of Service (DDoS). This type of attack can saturate links that are connected directly to the controller. The controller will indirectly process the incoming new packet to the topology. This is a major weakness in implementing a centralized control system. With the number of new packages that exceed the normal limit will force the controller to process the package by utilizing all available resources so that it can bring the controller in an unstable condition [3].

There is a southbound Application Programming Interface (API) protocol that functions to standardised the communication so that the layer forwarding device can perform the forwarding function compiled by the controller. Several methods have been implemented before by integrating machine learning. However, its implementation is only based on the process of detection and identification of attacks and used the available dataset that did not maintain the feature

specifically for SDN environment. In the process of detection, several previous studies applied a proactive scheme by utilizing flow integration tools for packet headers without involving controllers. But in this method, there are still scalability issues where the system can only be implemented on forwarding devices that have adequate resources. Using the C4.5 algorithm, bayesian network, decision table, and naïve bayes in the DDoS detection process [13]. The system is developed reactively by providing certain IP subnet block actions, with a dataset that has 4 features, including, the IP of the attacker, the destination host, the attack number, and the timestamp of the attack. This research has not discussed in detail the port and protocol-based attacks. Nanda *et al*. [10], used the Support Vector Machine (SVM), Naïve Bayes, and Neural Network methods to identify DDoS attacks with a selection feature based only on the number of hosts making requests every second without using IP and protocol. Whereas, Housman *et al*. [6] compared several classification methods with datasets obtained from Defense Advanced Research Projects Agency (DARPA) without mitigating and were not accompanied by a description of the mechanism of comparison of the use of methods to the use of existing resources in SDN. Moreover, Polat *et al*. [12] created an ML server which could extract the dataset from the controller based on the Network Laboratory-Knowledge Discovery in Databases (NL-KDD) dataset which gained 98% in accuracy and defend proactively. Dong and Sarem [4] proposed machine learning and deep learning detection without implementing the mitigation method for SDN. The authors used NSL-KDD dataset which gained accuracy of 88% for Deep Neural Network method. Mohammed *et al*. [9] used the combination of 3 distinct available online datasets where the feature extracted using Flow gained 96.5% for the detection rate utilising random forest algorithm [8].

In different mechanism, several papers conducted their experiment by using their own dataset extraction process which was considered compatible with the SDN environment such as flow rule statistic. Singh and Sharma [15] used Flow dockers to extract the flow data from the SDN switches and utilised the feature extraction process gaining 98.3% accuracy for the K-Nearest Neighbor (KNN) classifier. Filho *et al*. [5] proposed an almost similar pattern of a dataset, using 4 main features (flow length, flow duration, flow size, and flow ratio). The researchers developed an improved KNN method and achieved around 99.4% for the recall variable. Moreover, Singh [16] also implemented flow stat request for generating 6 features as the main resource to classify using SVM resulting 95.24% for the accuracy of the detection rate. In similar pattern, Abdelraza *et al*. [1] used the flow statistic mechanism for developing dataset features in nmeta2 architecture which then classified by using SVM, KNN, and Random Forest.

From the background of previous research that has been done before, then this research will be implemented to develop and analyses applications to detect, identify, and mitigate DDoS attacks on SDN networks from various machine learning classification methods including, Radial Basis Function-Super Vector Machine (RBF SVM), Linear SVM, KNN, Decision Tree (DTC), Random Forest (RFC), Multi-Layer Perceptron (MLP), and Gaussian Naïve Bayes (GNB) reactively. This research also proposed new feature scheme for the dataset which could create an almost static form of dataset increasing to the accuracy variable. Later, each method will be analysed based on the accuracy of the detection and mitigation results as well as the accumulative data of resource usage from the controller when running the application. In addition, the application will implement the DDoS mitigation scheme by utilizing the priority rule feature provided by the OpenFlow standard. So packets that are classified as malicious packets will automatically be blocked. The details of the application scheme method and the research method are explained further in the research method section [18].

2. Research Method

The research was emulated using Mininet [11] as the emulator software. The topology consisted of three main SDN switches, one RYU SDN Framework (RYU) [13, 16] controller as the centre of the networking management process, and six Hosts. Software-defined networking technology is an approach to network management that enables dynamic, programmatically efficient network configuration in order to improve network performance and monitoring, making it more like cloud computing than traditional network management. Figure 1 illustrates the overview of the specified topology. The main concern of the attacker was directed to perform a DDoS attack directly to the controller by sending randomly generated data test packets. The H1 acted as the attacker by sending a large number of packets to the H4.

The controller by default implemented learning switch mechanism for mapping the network environment. The flow of the proposed reactive machine learning application is described in Figures 2 and 3. Upon receiving new packets, the switch will directly filter the packet's header with the available flow selector installed on the switch. If there is no packet's selector that can filter the incoming packet the switch will encapsulate the incoming packet in OPFT_PACKET_IN message to the controller [14].

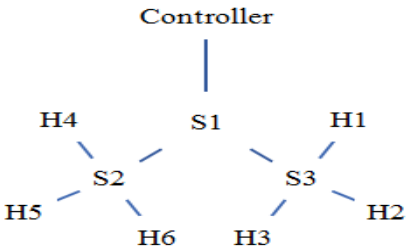

Figure 1. Emulation's topology.

Since the construction of the packet consisted of either random source port or IP source, the SDN switch would automatically forward the incoming DDoS packet to the controller for further assessment. Otherwise, the SDN switch will perform the traffic treatment specified by the controller, such as forwarding, dropping, crafting, etc., [19] Subsequently, the application processes the header's information and converts it into float format using a standard scaler. Furthermore, the application applies the classification model to predict the result of the current packet's header information. If the result states that the packet is considered as the normal packet, the application will transfer the packet processing mechanism to learning switch application. On the other hand, if the packet is identified as DDoS packet, the controller creates OFPT_FLOW_MOD message consists of flow rule construction command for blocking the DDoS attack based on the identified protocol's type then sends it to the corresponding SDN switch (directly connected to the attacker). Therefore, the attack will be blocked or dropped by the SDN switch using an idle time period 60s.

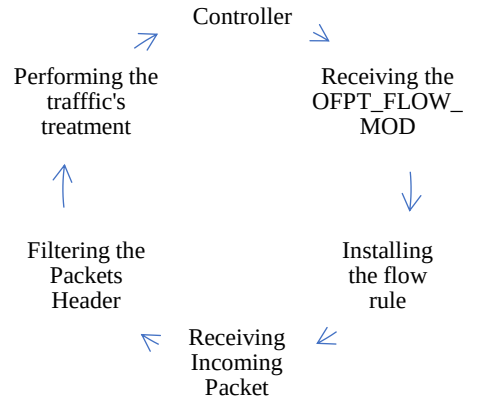

Figure 2. SDN switch's block diagram.

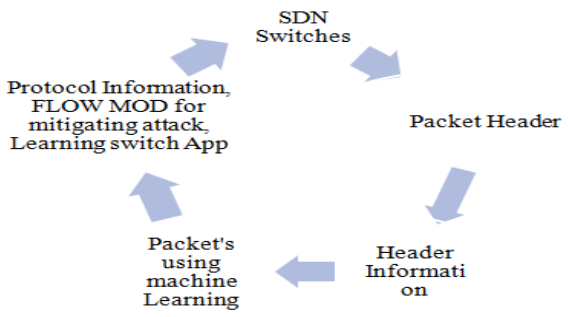

Figure 3. SDN controller's block diagram.

The classification algorithms that were used during the experiment consisted of RBF SVM, Linear SVM, KNN, DTC, RFC, MLP, Qualitative Data Analysis (QDA), and GNB. Each of the algorithms has its own hyperparameter which is illustrated in Table 1. In order to analyses the effectiveness of the specified algorithms, several variables were used to measure which was the most accurate algorithm for classifying DDoS attack. The variables included accuracy, precision, recall, and F1 score. Figure 3 shows a generic block diagram of a simple SDN where the external packet arrives at the switch and the switch is connected to a controller. The switch block and controller block in Figure 3 is modelled as a queue. There are three important phases an SDN model must capture.

- *Phase* (1), the first packet of a flow arrives at the switch and there is no matching for the packet.
- *Phase* (2), the packet without a matching flow entry is forwarded to the controller or a packet with the matching is serviced by the switch and forwarded to the destination.
- *Phase* (3), the controller feeds the forwarding information back to the switch and updates the flow table in the switch.

Note that the controller is not a physical part of the switch, but plays a critical role in switch forwarding and network performance and cannot be neglected. Table 1 showing the parameters classification.

Table 1. Parameters classification.

ALGORITHM	HYPERPARAMETERS
RFC	NUMBER OF TREES: 10 SPLIT FUNCTION: GINI INFORMATION GAIN: ENTROPY MAXIMUM TREE'S DEPTH: 5
MLP	HIDDEN LAYER: 1 ACTIVATION FUNCTION: RELU SOLVER: ADAM MAXIMUM ITERATION: 200
ADC	MAXIMUM NUMBER OF ESTIMATOR: 50 BOOSTING ALGORITHM: SAMME.R
GNB	THE LARGEST VARIANCE: 1E-09 PRIOR PROBABILITIES: ADJUSTED BASED ON DATA
RBF SVM	KERNEL: RBF KERNEL COEFFICIENT: 0.01 LAMBDA: 10

3. Research Dataset

Instead of using flow statistic data which was introduced, the experiment was carried out using different dataset structure that could be categorized as a static form of data. Ultimately SDN does the same thing as traditional TCP/IP networks. Carry traffic from one place to another. You could also use Mininet to emulate an OpenFlow network, and use that data. The dataset can be found in [7]. The data extraction process used the IPv4, Internet Control Message Protocol (ICMP), Transmission Control Protocol (TCP), and

User Datagram Protocol (UDP) header information added by some information regarding to the port's statistic extracted from the reply of OFPMP_PORT_STATS requests. The list of features is described in Table 2.

Table 2. Feature list.

FEATURE'S NAME	FEATURE'S ORIGIN
DATAPATH_ID	OFPT_PACKET_IN
VERSION	IPV4'S HEADER
HEADER_LENGTH	IPV4'S HEADER
OFFSET	IPV4'S HEADER
TTL	IPV4'S HEADER
PROTO	IPV4'S HEADER
CSUM	IPV4'S HEADER
SRC_IP	IPV4'S HEADER
DST_IP	IPV4'S HEADER
SRC_PORT	UDP'S/TCP'S HEADER
DST_PORT	UDP'S/TCP'S HEADER
TCP_FLAG	TCP'S HEADER

Data extraction process (Figure 4) started from the attacker by sending two*.pcap files that contained the train data and test data. The flooding process was performed individually for both files. The authors configured the SDN switches for passing the incoming packet directly to the controller. Simple switch application in RYU had been extended as well in order to retrieve the packet's header information –

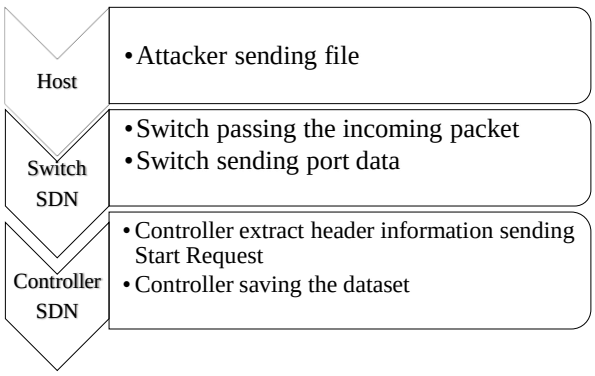

Figure 4. Dataset extraction process.

Concerning the IPv4, TCP, UDP, and ICMP as well as sending OFPPortStatsRequest to the corresponding switch for acquiring the port statistic.The average number of bytes of the incoming packet was calculated by differentiating the number of bytes received and the number of packets received (rx_bytes_ave). The attack was conducted by sending 3 distinct types of attacks including the ICMP, TCP, and UDP flood attack. The available class consisted of 6 class which included the DDOS_ICMP, DDOS_TCP, DDOS_UDP, NORMAL_ICMP, NORMAL_TCP, and NORMAL_UDP. In term of the data proportion, the number of train data is 420,0000 data while the test data is divided into two distinct number including the 18,000 and 36,000 data tests. Both data types (train and test) were generated on different occasion and consisted of different structures [20].

4. Results and Discussion

The experiment's results showed several variables that could be used to measure the capability of each proposed algorithm for detecting and identifying the DDoS attack. Table 3 showed the training time for each algorithm to create the classifier model based on twodistinct data proportion (420,000:18,000 and420,000:36,000). The longest time was depicted by KNN followed by MLP and SVM respectively while the other algorithms could finish the model before or equal to 1 s. This behavior might occur since the SVM used Linear and RBF kernel which were considered as the complex kernel. KNN also took a considerably long time to develop a model because of the calculation of the distance between data point. Table 3 showing the training time.

Table 3.Time.

Classifier Type	Training Time (s)
SVM (RBF)	6.514
SVM (LIN)	3.085
KNN	85.781
DTC	1
RFC	0.7
MLP	18.971
GNB	0.1

Table 4. Experiment's result without SDN for 18000 dataset.

Classifier Type	Accuracy (18000) %	Precision (18000) %	Recall (18000) %	F1 (18000) %
SVM (RBF)	99.3	99.3	99.3	99.3
SVM (LIN)	99.9	99.9	99.9	99.9
KNN	99	99	99	99
DTC	100	100	100	100
RFC	69.5	80	69.5	62.7
MLP	35.6	33.2	35.6	22.3
GNB	99.7	99.7	99.7	99.7

Moreover, the experiment was conducted into two different scenarios which included the prediction scheme without involving the SDN controller and generating the prediction using the SDN controller. Table 4 showing the result without SDN for 18000 dataset. The positive trend illustrated by SVM, KNN, RFC, MLP, and GNB. The results showed the highest accuracy depicted by both linear SVM and GNB for all data test size. However, this trend was not described in the second scenario due to data redundancy [21].

Table 5. Experiment's Resultwithout SDN for 36000 dataset.

Classifier Type	Accuracy (36000) %	Precision (36000) %	Recall (36000) %	F1 (36000) %
SVM (RBF)	100	100	100	100
SVM (LIN)	100	100	100	100
KNN	99.4	99.4	99.4	99.4
DTC	83.3	75	83.3	77.7
RFC	97	97.5	97	97
MLP	50.4	50.4	50.4	50.4
GNB	100	100	100	100

In term of the first scenario portrayed by Tables 5 and 6, the accuracy for DTC were decreasing because the uniformity of data test dropping along with the growth of the number of data test. Surprisingly, the pattern did not occur for some classification algorithm on the second scenario since the prediction results appeared to be redundant. This circumstance might occur because there was a lot of feature processing mechanism that should be maintained by the SDN controller. Therefore, the percentage of prediction loss was pointed at around 79.8% for 36,000 data tests which meant only 7,272 data successfully classified illustrated in Table 8. In details, the algorithm that could maintain its accuracy variable was Radial Basis Function-Super Vector Machine (RBF SVM) and Linear SVM approximately at 100% indicating that the increase of the data test did not reduce the robustness of the classification process (Tables 6 and 7). The increasing trend in accuracy also produced by KNN. The remaining algorithms including DTC, RFC, MLP, and GNB generated a decreasing percentage for all variables which indicated that the algorithms could not handle the classification efficiently.

Table 6. Experiment's result with SDN for 18000 dataset.

Classifier Type	Accuracy (18000) %	Precision (18000) %	Recall (18000) %	F1 (18000) %
SVM (RBF)	100	100	100	100
SVM (LIN)	100	100	100	100
KNN	87.8	93.1	88.2	86.5
DTC	100	100	100	100
RFC	84.2	86.4	84.6	83.1
MLP	34.5	32.5	36.3	23
GNB	93.9	95	94.1	93.5

Table 7. Experiment's ResultWith SDN for 36000 dataset.

Classifier Type	Accuracy (36000) %	Precision (36000) %	Recall (36000) %	F1 (36000) %
SVM (RBF)	100	100	100	100
SVM (LIN)	100	100	100	100
KNN	95.6	89.2	92.9	87.5
DTC	89.7	78.4	83.3	80.4
RFC	80.7	88.4	84.4	82.5
MLP	65.3	52.4	66.6	54.2
GNB	59.1	68.6	83.1	70

Table 8. Percentage of prediction loss in SDN.

ClassifierType	Prediction Loss (18000) in %	Prediction Loss (36000) in %
SVM (RBF)	2.7	79.8
SVM (LIN)	2.7	79.8
KNN	2.7	79.8
DTC	2.7	79.8
RFC	2.7	79.8
MLP	2.7	79.8
GNB	2.7	79.8

5. Conclusions

Overall, DDoS still became one of the most significant issues in SDN in regard to centralized control management. The authors proposed new scheme of dataset which utilized the packet's library and the port statistic request for extracting 25 features in total. Machine learning methods was comparatively analyzed and can be deduced that the SVM whether used the Linear or RBF kernel could produce the highest accuracy among several similar researches. SDN allows the separation of a control layer and data forwarding at two different layers. However, centralized control systems in SDN is vulnerable to attacks namely DDoS. Therefore, it is necessary for developing a solution based on reactive applications that can identify, detect, as well as mitigate the attacks comprehensively. In this paper, an application has been built based on machine learning methods including, SVM using Linear and Radial Basis Function kernel, KNN, DTC, RFC, MLP, and GNB. The paper also proposed a new scheme of DDOS dataset in SDN by gathering considerably static data form using the port statistic. SVM became the most efficient method for identifying DDoS attack successfully proved by the accuracy, precision, and recall approximately 100 % which could be considered as the primary algorithm for detecting DDoS. In term of the promptness, KNN had the slowest rate for the whole process, while the fastest was depicted by GNB. The authors will try to analyze the significance of Deep Learning method for detecting the similar attacks in the future.

References

[1] Abdelraza D., Abu-Soud S., and Awajan A., "A Machine Learning System for Distinguishing Nominal and Verbal Arabic Sentences," *The International Arab Journal of Information Technology*, vol. 15, no. 3A, pp. 576-584, 2018.

[2] Bakker J., Ng B., and Seah W., "Can Machine Learning Techniques Be Effectively Used in Real Networks Against DDoS Attacks?," *in Proceedings of 27th International Conference on Computer Communication and Networks*, Hangzhou, pp. 1-6, 2018.

[3] Dey K. and Rahman M., "Effects of Machine Learning Approach in Flow-Based Anomaly Detection on Software-Defined Networking," *Symmetry*, vol. 12, no. 1, pp. 1-7, 2019.

[4] Dong S. and Sarem M., "DDoS Attack Detection Method Based on Improved KNN with the Degree of DDoS Attack in Software-Defined Networks," *IEEE Access*, vol. 8, pp. 5039-5048, 2020.

[5] Filho F., Silveira F., Junior A., Vargas-Solar G., and Silveira l., "Smart Detection: An Online Approach for DoS/DDoS Attack Detection Using

Machine Learning," *Security and Communication Networks*, vol. 2019, 2019.
[6] Housman O., Isnaini H., and Fauzi S., *SDN-DDOS* (*ICMP*, *TCP*, *UDP*), Mendeley Data, 2020.
[7] Kokila R., Selvi S., and Govindarajan K., "DDoS Detection and Analysis in SDN-Based Environment Using Support Vector Machine Classifier," *in Proceedings of 6th International Conference on Advanced Computing*, Chennai, pp. 205-210, 2014.
[8] Meti N., Narayan D., and Baligar V., "Detection of Distributed Denial of Service Attacks Using Machine Learning Algorithms in Software Defined Networks," *in Proceedings of International Conference on Advances in Computing, Communications and Informatics*, Udupi, pp. 1366-1371, 2017.
[9] Mohammed S., Hussain R., Senko O., Bimaganbetov B., Lee J., Hussain F., Kerrache C., Kerrache E., and Bhuiyan M., "A New Machine Learning-based Collaborative DDoS Mitigation Mechanism in Software-Defined Network," *in Proceedings of 14th International Conference on Wireless and Mobile Computing, Networking and Communications*, Limassol, pp. 1-8, 2018.
[10] Nanda S., Zafari F., DeCusatis C., Wedaa E., and Yang B., "Predicting Network Attack Patterns in SDN Using Machine Learning Approach," *in Proceedings of IEEE Conference on Network Function Virtualization and Software Defined Networks*, Palo Alto, pp. 167-172,2016.
[11] OpenFlow Switch Specification Version 1.3.0 Wire Protocol 0x04, https://www.opennetworking.org/wpcontent/uploads/2014/10/openflow-spec-v1.3.0.pdf),Last Visited, 2020.
[12] Polat H., Polat O., and Cetin A., "Detecting DDoS Attacks in Software-Defined Networks Through Feature Selection Methods and Machine Learning Models," *Sustainability*, vol. 12, no. 3, 2020.
[13] RYU Project Team, RYU SDN Framework, Ryubook 1.0 Documentation, https://osrg.github.io/ryu-book/en/Ryubook.pdf, Last Visited, 2020.
[14] Sezer S., Scott-Hayward S., Chouhan P., Fraser B., Lake D., Finnegan J., Viljoen N., Miller M., and Rao N., "Are we Ready for SDN? Implementation Challenges for Software-Defined Networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 36-43, 2013.
[15] Singh A. and Sharma S., "Digital Era in the Kingdom of Saudi Arabia: Novel Strategies of the Telecom Service Providers Companies," *Webology*, vol. 17, no. 1, pp.227-245, 2020.
[16] Singh A., "An Intelligent Reallocation of Load for Cluster Cloud Environment," *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 8, pp. 711- 714, 2019.
[17] Singh A., "Texture-based Real-Time Character Extraction and Recognition in Natural Images," *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 8, pp. 3302-3306, 2019.
[18] Singh A., "A Wireless Networks Flexible Adoptive Modulation and Coding Technique in advanced 4G LTE," *International Journal of Information Technology*, vol. 11, no. 1, pp. 55-66, 2019.
[19] Software Defined Networking: The Norm for Networks (2012).
[20] Uqaili I. and Ahsan S., "Machine Learning Based Prediction of Complex Bugs in Source Code," *The International Arab Journal of Information Technology*, vol. 17, no. 1, pp. 26-37, 2020.
[21] Ye J., Cheng X., Zhu J., Feng L., and Song L., "A DDoS Attack Detection Method Based on SVM in Software Defined Network," *Security and Communication Networks*, vol. 2018, 2018.

Arun Kumar Singh is working in the College of Computing and Informatics, Saudi Electronics University, Kingdom of Saudi Arabia (KSA). He received Ph.D. in CE/IT-2013 by the School of Computer Engineering/Information Technology, SU, Meerut, India, under the guidance of Dr. Neelam Srivastava (IET Lucknow) and Dr. R. P. Agarwal (IIT Roorkee), M.Tech. (IT-WCC) degree in 2005 from IIIT-Allahabad under the guidance of Prof. M. Radha Krishnan and B.E. (ECE) degree in 2002 from Dr. B. R. Ambedkar University, Agra, India. His research interests are Machine Learning, IoT, AI, Big Data, Network Management, Wireless networking, Social Networking and Mobile computing