Detection and Classification of Conflict Flows in SDN Using Machine Learning Algorithms

Journal:	*IEEE Access*
Manuscript ID	Access-2021-16135
Manuscript Type:	Special Section: Intelligent Big Data Analytics for Internet of Things, Services and People
Date Submitted by the Author:	01-May-2021
Complete List of Authors:	hamed, mutaz; University of Technology Malaysia, Enginnering ; SYED ARIFFIN, SHARIFAH HAFIZAH ; Universiti Teknologi Malaysia, Abdul Latiff, Nurul Muazzah; Universiti Teknologi Malaysia - Main Campus Skudai, School of Electrical Engineering; Universiti Teknologi Malaysia Mohamad Yusof, Kamaludin ; Universiti Teknologi Malaysia, School of Electrical Engineering Hassan, Mohamed; University of Technology Malaysia, School of Electrical Engineering; Future University, Faculty of Telecommunication Naji, Fahad; Universiti Teknologi Malaysia - Main Campus Skudai, Communication Engineering Hamdan, Mosab; UTM Skudai, elecrtical engineering Khan, Suleman ; Northumbria University, Computer and Information Sciences MUZAFFAR, HAMZAH; Universiti Malaysia Sabah, Faculty of Computing and Informatics
Keywords: <b>Please choose keywords carefully as they help us find the most suitable Editor to review</b>:	Security, Computer networks, Computer network management, Computerized monitoring, Machine learning algorithms
Subject Category Please select at least two subject categories that best reflect the scope of your manuscript:	Computational and artificial intelligence, Communications technology
Additional Manuscript Keywords:	Software-Defined Network, machine learning algorithms, flows classification

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Detection and Classification of Conflict Flows in SDN Using Machine Learning Algorithms

MUTAZ H. H. KHAIRI[1,2], SHARIFAH H. S. ARIFFIN[1] (Senior Member, IEEE), NURUL MU'AZZAH ABDUL LATIFF[1] (Senior Member, IEEE), KAMALUDIN MOHAMAD YUSOF[1], MOHAMED KHALAFALLA HASSAN[1,2], FAHAD TAHA AL-DHIEF[1] (Student Member, IEEE), MOSAB HAMDAN[1] (Student Member, IEEE), SULEMAN KHAN[3], AND MUZAFFAR HAMZAH[4]

[1]Faculty of Engineering, School of Electrical Engineering, Universiti Teknologi Malaysia (UTM), Johor Bahru 81310, Malaysia
[2]Faculty of Engineering, Future University, Khartoum 10553, Sudan
[3]Department of Computer and Information Sciences, Northumbria University, Newcastle Upon Tyne, NE1 8ST, UK
[4]Faculty of Computing and Informatics, Universiti Malaysia Sabah, 88400 Kota Kinabalu, Sabah, Malaysia

Corresponding author: Mutaz H.H. Khairi (taza1040@gmail.com), Sharifah H. S. Ariffin (shafizah@utm.my), Muzaffar Hamzah (muzaffar@ums.edu.my)

ABSTRACT Software-Defined Networking (SDN) is a new type of technology that embraces high flexibility and adaptability. The applications in SDN have the ability to manage and control networks such as in load balance, access control, and routing. These are considered the most significant benefit of SDN. However, SDN can be influenced by several types of conflict flows which may lead to deterioration on the network performance in terms of efficiency and optimisation. Furthermore, applying machine learning algorithms in the identification and classification of conflict flows has limitations. As a result, this paper discusses various machine learning algorithms for detecting and classifying conflict flows in SDNs. These algorithms include Decision Tree (DT), Support Vector Machine (SVM), Extremely Fast Decision Tree (EFDT) and Hybrid (DT-SVM). In addition to improve the performance of the suggested EFDT and hybrid DT-SVM algorithms, the EFDT and hybrid DT-SVM algorithms were designed and deployed based on DT and SVM algorithms. In this study, the number of flows selected were ranged between 1000 and 100000 with an increment in steps of 10000 flows. Additionally, there are two network topologies being created (i.e., Fat Tree Topology and Simple Tree Topology) using the Mininet simulator and connected to the Ryu controller. To assess the performance of the proposed algorithms in terms of efficiency and effectiveness, a variety of evaluation metrics are used. The experimental results for the detection of conflict flows show the DT algorithm achieves 99.27% accuracy, the SVM algorithm obtains 98.53% accuracy. Meanwhile, the detection accuracies for EFDT and hybrid DT-SVM algorithms achieve 99.49% and 99.27%, respectively. In addition, the proposed EFDT algorithm achieves 95.73% accuracy for the classification between conflict flow types. The proposed EFDT and hybrid DT-SVM algorithms show a high capability of SDN applications that offer fast detection and classification of conflict flows.

INDEX TERMS Software-Defined Network, conflict flows detection, conflict flows classification, machine learning algorithms.

I. INTRODUCTION

The conventional architecture of a network is not fully adaptable to the requirements of using the current network's applications and advanced data centre environments. Therefore, Software-Defined Network (SDN) is proposed; where SDN is aimed to allow administrators and engineers of cloud and network to keep up to ever changing business requirements over a centralized control console [1]. Besides that, the SDN includes a variety of network technologies designed to make the network scalable and robust enough to accommodate storage infrastructure and virtualized servers in a modern data centre. Furthermore, the SDN technique is originally destined for managing, constructing, and designing networks. This is to provide direct network programmability control and independence of the primary infrastructure for network services and applications by separating the network control and forwarding planes. In general, the SDN is cost-effective, manageable, dynamic, and adaptable which makes it appropriate for the dynamic nature of high-bandwidth modern applications [2]. SDN presents a virtualised execution framework which separates the network control functions from the underlying network forwarding traffic [3]. Also, it incorporates various equipment of the network (e.g., routers, switches, and access points). Thus, it allows for the implementation of various network control functions. Furthermore, the controller allows complicated network configuration. SDN main goal is to give users more control over their control configuration while still meeting network efficiency requirements [4]. Besides, the SDN has other advantages such as centralised monitoring that helps in reducing manual communication with the hardware, thus enhancing the network's efficiency. Additionally, separating the control plane and data plane leads to simpler hardware and increases the chances of having more expertise among hardware vendors, as the devices do not rely on commercial software [5]. The infrastructure layer, control layer (SDN controller), and application layer are the three basic components of SDN architecture. Numerous networking devices, such as switches and routers, make up the infrastructure layer. The control layer located the core of the SDN model, where it hosts the centralised SDN controller software. And, the application layer refers to the implementation of typical workings of the network or functions [6]. Fig. 1 presents the SDN architecture. In Fig. 1, the OpenFlow is considered the first SDN standards. Essentially, it presents the connection protocol in SDN environments. It is obvious that the OpenFlow separates the infrastructure layer from the control layer. This is highly beneficial, where developers can modify and develop the application layer. Therefore, the application layer can appropriately be adapted to the changing business requirements [7].

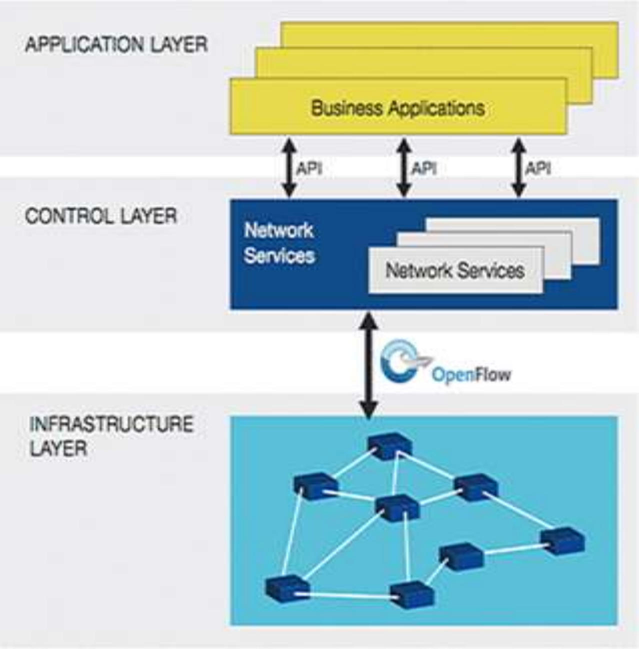


FIGURE 1. **The SDN architecture [8].**

In addition, OpenFlow is a protocol presented to facilitate the connection between network switches and server with respect to received and sent packets. Besides, it allows for sharing the same physical infrastructure with numerous logical networks. Aside from that, the network virtualisation layer includes a collection of controllers for managing a large number of switches. In this case, one switch can belong to numerous virtual networks, controllable through one or more collection of controllers. However, such a design is susceptible to flow conflicts [9]. Nowadays, depending on the destination, data transmission is redirected. This approach provides an efficient application of narrowest routing protocols, but it does not provide fine-grained network traffic control. Nevertheless, there are several suggestions for future internet designs which need network data plane to perform routing and forwarding at the levels of single connections or their aggregate (e.g., network services or network virtualisation) [10]. The reliability of a traditional network is known to be harmed by various types of conflicts, and SDN is no exception. Intelligible Conflicts and Interpretative Conflicts are the two major forms of conflicts that can be categorised according to their laws and effects.

The techniques for detecting and classifying the flow conflicts happening in SDN models are highly imperative. For example, Machine Learning (ML) algorithms have proved their efficiency and effectiveness in detecting and classifying among two or more subjects [11], [12], where these algorithms were applied and used in several domains such as identification of spam emails [13], images classification in the medical domain [14], [15], voice pathology detection [16]–[18], and language identification [19], [20]. In these methods, the algorithms of ML have been implemented to play the main role. The main purpose of using these algorithms is to train and build a system that is efficiently capable to classify subjects with high detection accuracy. However, ML algorithms are still suffering from low detection accuracy in SDN models. Moreover, these algorithms have not gained attention with respect to the detection and classification of the flow conflicts in SDN. In other words, there is no work that represents the detection and classification of flow conflicts types using ML algorithms. Therefore, the following are the key contributions of this paper:

- Proposing two algorithms called Extremely Fast Decision Tree (EFDT) and hybrid Decision Tree-Support Vector Machine (DT-SVM).
- Applying four different algorithms in the detection and classification of conflict flows. These algorithms are DT, SVM, EFDT, and DT-SVM.
- The algorithms used to identify and classify conflict flows use a different number of flows each time.
- Many assessment metrics, such as accuracy, precision, f1-score, recall, and execution time, are used to assess the proposed algorithms.

- No research that we are aware of has used machine learning algorithms to identify and classify conflict flows.

The rest of this paper is organized as follow; Section II addresses similar works of ML algorithms used in the SDN domain. The suggested methods are presented in Section III. The experiment results and discussion are detailed in Section IV. Finally, Section V brings the paper to a conclusion.

II. RELATED WORK OF ML ALGORITHMS IN SDN

Machine learning algorithms have opened up several significant opportunities in implementing SDN models, particularly in security applications. These algorithms have widely been used in SDN models in order to elevate the performance of models. Here, we will look at the most up-to-date models for traffic classification, flow detection and classification, security, and traffic management, all of which used different machine learning algorithms. Table 1 also includes a summary of similar machine learning algorithms used in SDN models. In [21], a flow-aware elephant flow detection is implemented on SDN. This article, to effectively accomplish reliable elephant flow identification, uses the suggested approach utilising two classification models, first on SDN switches (i.e., switch-side classifier) and second on the controller (i.e., controller-side classifier), simultaneously. In addition, this strategy helps the elephant flow identification activities among the controller and switches to be exchanged. Therefore, in the switches, several mouse flows can be screened, therefore eliminating the need to give the controller vast quantities of classification demands and signalling notifications. Experimental results show that, in terms of running time, precision, F-measure, and recall, the proposed methodology outperforms contemporary approaches.

The study conducted in [22] is destined as a demonstration of principle when integrating machine learning with SDN applications, in general for detection of network traffic. It has been demonstrated that traffic classification using machine learning algorithms improves performance in the context of SDN. This is achievable due to the potent of this structure to gather knowledge. It is evident that this approach is highly successful. These high-performing, intelligent-based communication principles can boost or even replace traditional network controls in the near future. In [23], this study has discussed the influence of various OpenFlow time windows on the output prediction of various classification algorithms. On OpenFlow flow datasets generated in both virtual and physical SDN environments, a total of 150 prototypes were built and tested. The results of the analysis show that the OpenFlow traffic time interval chosen has a major impact on detection performance — wider time windows result in lower detector output. Moreover, by adding correct time-windows to OpenFlow traffic, they have been able to gain good precision in detecting unidentified threats.

Furthermore, the work in [24] has suggested an intelligent solution to screening and classification (ESCA). The authors have proposed a modern differentiated scheduling method that independently and progressively establishes routes for elephant and mouse flows. ESCA significantly reduces processing overhead and efficiently classifies specimens using a new supervised classification algorithm about data flow similarities by measuring the delivery time of elephant flows and filtering out duplicate specimens. With a focus on low-cost ESCA, a DiffSch feature-aware flow schedules solution that distinguishes between elephant and mouse flow schedules is proposed. According to the general theory, ESCA surpasses the related frameworks. Furthermore, comprehensive experiments demonstrate that ESCA could produce accurate identification with far less collected samples and a shorten detector period, and that certain DiffSch schedule method model outperformed related proposals significantly. In [25], the authors have proposed CyberPulse which is a new powerful preventive method of measuring that underpins a classifier based on machine learning to mitigate LFA in SDN. By classifying network traffic using deep learning methods, CyberPulse conducts network monitoring and is incorporated in the Floodlight controller as an enhanced subsystem as opposed to existing techniques on produced practical networks using Mininet. The precision, false positive rate, and efficiency of CyberPulse were then evaluated. According to the results, CyberPulse could identify suspicious flows with highly accurate and easily reduce them.

In addition, the study in [26] has discussed issues related to flow management introduced by network connectivity. A supervised learning prototype is proposed to reduce the SDN controller's reaction time for large complex architectures, allowing the controller to forecast node mobility and connection failure risk. An alternate path preference structure would instead be introduced, which also ensures efficient traffic balancing when minimising the workload of the control plane. In the commonly utilised network simulator-ns-3, the result of the algorithm SD-WMN model is verified. The findings demonstrate that the designed SD-WMN model with link-failure proactive traffic management has obtained data transmission improvement. The author in [27] has developed a system for detecting and deploying DDoS threats in SDN-focused virtual networks. The suggested framework includes not only the control function dependent on the OpenFlow interface statement (i.e., PACKET IN statement) for a non-timely reply, but also a multi-dimensional information-based flow extracting features method. In addition, creating an efficient nationwide network flow table component behaviour focused on the OpenFlow table function and the flow table entrance stability feature. Evaluating all feedback to the flow table is done by professional SVM. It shows that the identification method efficiently decreases the time for initial attack detection and classification identification by evaluating the test outcomes and with a smaller false alarm rate. The work in [28] has proposed SDN-Home Gateway (SDN-HGW), which expands the regulation for improved end-to-end network security to the connection network (i.e., a housing automation system). Through the classification of data flow in a smarter housing system, the suggested SDN-HGW will gain decentralised device knowledge. For coded data packet, many current traffic identification approaches, e.g., deeper packets analysis, do not offer real device knowledge. Built encoded data classification model (known as DataNets) focused on several deep learning models to solve these problems; An open data library of around 200,000 sets is used by deep learning. To handle total information packages and the checked data set so that DataNets can be developed, a data preparation structure is suggested. The experimental findings indicate that the built DataNets can be used in upcoming smarter housing networking to allow distributed framework SDN-HGW.

TABLE 1. **Summary of related work.**

Year	Algorithm	Dataset	Application	Accuracy	Ref.
2020	Very Fast Decision Tree	MAWI UNI1	Flow Detection	98.64% 99.78%	[21]
2019	SVM DT Random Forest K-Nearest Neighbours	Kaggle	Traffic Classification	96.37% 95.76% 94.92% 71.47%	[22]
2019	SVM Naïve Bayes K-Nearest Neighbours	Flow Generation	Security	-	[23]
2019	Efficient Sampling and Classificatio n Approach	Flow Generation	Network optimization performance	90%	[24]
2019	Deep learning	UCI	Security	85%	[25]
2018	SVM	Flow Generation	Traffic Management	-	[26]
2018	SVM	CAIDA DDoS	Security	-	[27]
2018	SVM Naïve Bayes	Flow Generation	Traffic Classification	-	[28]
2018	DT	Flow Generation	Flow Classification	-	[29]
2017	SVM K-Means	-	Traffic Classification	98% 88%	[30]

Machine learning traffic flow classification methods are used, and SDN rules are detected based on the flow categories produced, the study in [29] has presented a platform that exacerbates this difficulty. Using both supervised learning methods for various forms of traffic depending on pre modelling techniques and unsupervised learning, varying traffic flows are clustered together as well. Finally, a flow grouping classifier defines that flows are normally observed together in an identical time period upon identifying the flows. For classification problems, C4.5 decision tree classifiers with functions for each flow, like cross arrival time, packet size, packet number, and flow tuple, are used. In [30], For network traffic identification, two machine learning algorithms have been evaluated: SVM and K-means. It has been stated that it is possible to obtain an average precision of around 95 percent. In the meantime, through design adjustment and data pre-processing, the efficiency of the machine could be further increased. The configuration and feature choice of the SVM model was carried out due to the classification of traffic. Findings demonstrate that the radial base kernel function based SVM model provides the SVM model with the highest precision and are most effective in numerical terms.

Additionally, the findings of all the studies that used machine learning algorithms in the SDN models can be summarised as follow:

- The supervised learning methods are widely employed. However, although KNN, SVM, DTs and Bayesian approaches are shown in most solutions and gained more interest, there is very little literature on logistic regression of SDN in terms of the range of functional implementations.
- The most supervised learning algorithms obtain a relatively higher average accuracy of over 90 % in identification performance of evaluation metrics.
- In SDN applications, most studies have used Support Vector Machine (SVM) and Decision Tree (DT) algorithms.
- There is no machine learning approach implemented in the detection and classification of conflict flows.
- There are various types of datasets been used, where some studies used datasets from internet sources such as Kaggle, and other studies used flow generation method to create the dataset for machine learning algorithms.
- There is no research showing the key features of flow entries in SDN (e.g., priority and action features) for all forms of dataset used in the current machine learning solution.
- The precision, recall, and f1-measure are the most commonly used assessment metrics to evaluate and validate ML algorithms in most studies. The accuracy and execution time, on the other hand, are negligible.

III. MATERIALS AND METHODS

The proposed model in this study has two main phases: detection and classification of conflict flows. Fig. 2 shows the proposed model for detection and classification of conflict flows. The first phase is the detection between conflict flows and normal flows.

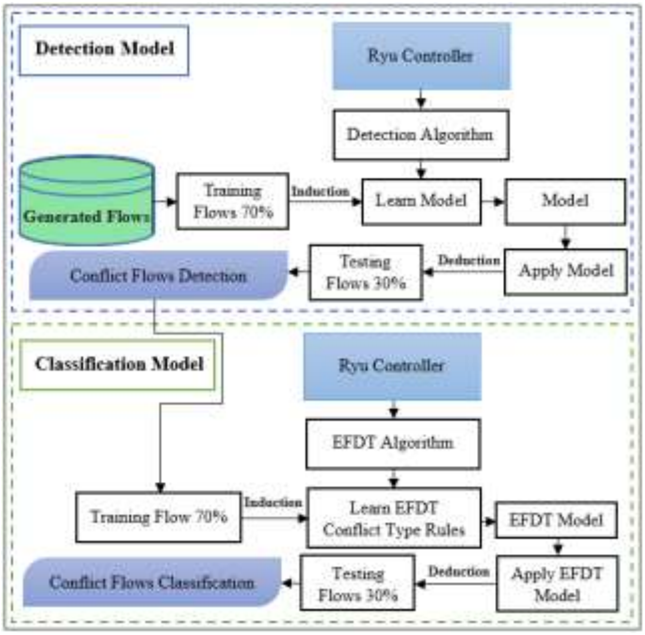


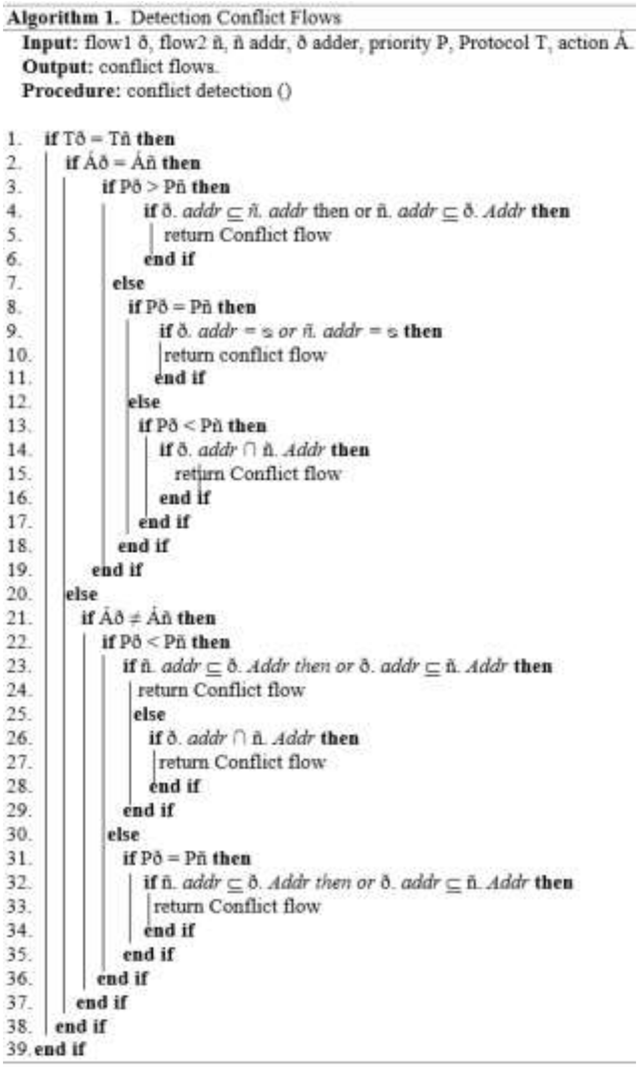
FIGURE 2. The proposed model for the detection and classification of conflict flows.

In this phase, the generated flow will check by the algorithms implemented in the controller plane, to observe the behaviour of flows. There are features in the flows which are significant in differentiating between normal and conflict flows such as Mac address, IP address and action. Accordingly, the result of these features checking algorithms will identify whether the flows are normal or conflict. The normal flows will pass directly to "OpenFlow" and the conflict flows will pass to the next phase to classify which types of conflict occurs in conflict flows.

There are four algorithms proposed to detect the conflict flows in "OpenFlow". These algorithms are Decision Tree (DT), Support Vector Machine (SVM), Extremely Fast Decision Tree (EFDT) and Hybrid (DT-SVM). The EFDT and hybrid algorithms were developed and implemented from the DT and SVM in order to elevate their performance in terms of accuracy and running time.

The algorithms of DT and SVM were selected because they have shown high performance in previous research in different applications of SDN [31]–[34]. Fig. 3 shows the pseudo code for algorithms used in the detection of conflict flows. Furthermore, the steps of the detection phase can be summarised as follow:

1) Implement and running algorithms.
2) The algorithms will check the features of flows.
3) The algorithms will identify the normal flows and conflict flows.
4) Normal flows will pass as normal to OpenFlow.
5) The conflict flows will pass to the classification algorithm.



Algorithm 1. Detection Conflict Flows

Input: flow1 ð, flow2 ñ, ñ addr, ð adder, priority P, Protocol T, action Á.
Output: conflict flows.
Procedure: conflict detection ()

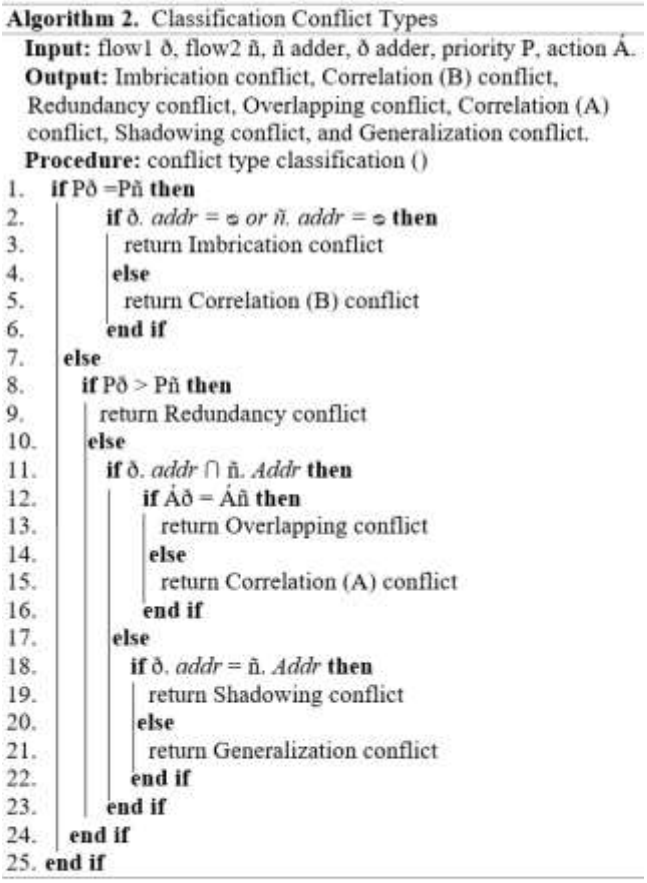
```
if Tð = Tñ then
    if Áð = Áñ then
        if Pð > Pñ then
            if ð. addr ⊆ ñ. addr then or ñ. addr ⊆ ð. Addr then
                return Conflict flow
            end if
        else
            if Pð = Pñ then
                if ð. addr = ≤ or ñ. addr = ≤ then
                    return conflict flow
                end if
            else
                if Pð < Pñ then
                    if ð. addr ∩ ñ. Addr then
                        return Conflict flow
                    end if
                end if
            end if
        end if
    else
        if Áð ≠ Áñ then
            if Pð < Pñ then
                if ñ. addr ⊆ ð. Addr then or ð. addr ⊆ ñ. Addr then
                    return Conflict flow
                else
                    if ð. addr ∩ ñ. Addr then
                        return Conflict flow
                    end if
                end if
            else
                if Pð = Pñ then
                    if ñ. addr ⊆ ð. Addr then or ð. addr ⊆ ñ. Addr then
                        return Conflict flow
                    end if
                end if
            end if
        end if
    end if
end if
```

FIGURE 3. The pseudo code for conflict flows detection.

The second phase of the proposed model is the classification of conflict flows. In this phase, the conflict flows identified in the detection phase will be checked by an algorithm implemented in the controller plane in order to determine the behaviour of flows. There are three features of conflict flows that are priority, IP address, and action. Upon checking process completion, the conflict types will be classified into seven types which are redundancy, shadowing, overlapping, correlation A, correlation B, generalisation, and imbrication. Fig. 4 shows the pseudo code for the EFDT algorithm in classifying conflict flows. Besides, the steps of the classification phase can be summarised as follows:

1) Implement and running the EFDT classifier algorithm.
2) The detection flows will start to check by the algorithm.
3) The algorithms will check the features of priority and IP address of flows.

4) The algorithms will classify the conflict types according to the features in step 3.



```
Algorithm 2. Classification Conflict Types
Input: flow1 ð, flow2 ñ, ñ adder, ð adder, priority P, action Á.
Output: Imbrication conflict, Correlation (B) conflict,
Redundancy conflict, Overlapping conflict, Correlation (A)
conflict, Shadowing conflict, and Generalization conflict.
Procedure: conflict type classification ()
if Pð =Pñ then
        if ð. addr = ⊜ or ñ. addr = ⊜ then
            return Imbrication conflict
        else
            return Correlation (B) conflict
        end if
else
    if Pð > Pñ then
        return Redundancy conflict
    else
        if ð. addr ∩ ñ. Addr then
            if Áð = Áñ then
                return Overlapping conflict
            else
                return Correlation (A) conflict
            end if
        else
            if ð. addr = ñ. Addr then
                return Shadowing conflict
            else
                return Generalization conflict
            end if
        end if
    end if
end if
```

FIGURE 4. The pseudo code for conflict flows detection.

A. DECISION TREE (DT) ALGORITHM

Decision Tree is a machine learning algorithm can be implemented for issues with regression and classification, but is often utilised for finding solutions of classification. It is a tree-structured algorithm where the characteristics of a database are described through internal nodes, branches representing the rules of decision and the result is defined from each leaf node. Decision nodes have been used to make decisions and have several branches, while the performance of those decisions will be Leaf nodes and there are no additional branches. Furthermore, Fig. 5 shows the diagram that explains the general structure of the DT algorithm. It is also possible to describe decision trees as a mix of mathematical and analytical methods to help identify, categorise and generalise a given data set. Data comes from the form's records as shown in the following equation:

$$(\boldsymbol{x}, \boldsymbol{Y}) = (\boldsymbol{x1}, \boldsymbol{x2}, \boldsymbol{x3} \dots . \boldsymbol{xn}, \boldsymbol{Y}) \qquad (1)$$

The conditional factor Y is the reference parameter that learning attempts to describe or categorise. The vector x is made up of the $\boldsymbol{x1}, \boldsymbol{x2}, \boldsymbol{x3}$ etc. characteristics that are used for that role. Additionally, the steps to implement the DT algorithm can be summarised as follow:

- Implement decision tree components in the controller plane.
- Setup the learner function.
- Prepare and import all generated flows from OpenFlow switch for all flow sizes.
- Train the algorithm with 70% of generated flows.
- Predict the response test of 30% for generated flows.
- Evaluate the confusion matrix for the DT algorithm and calculate running time.

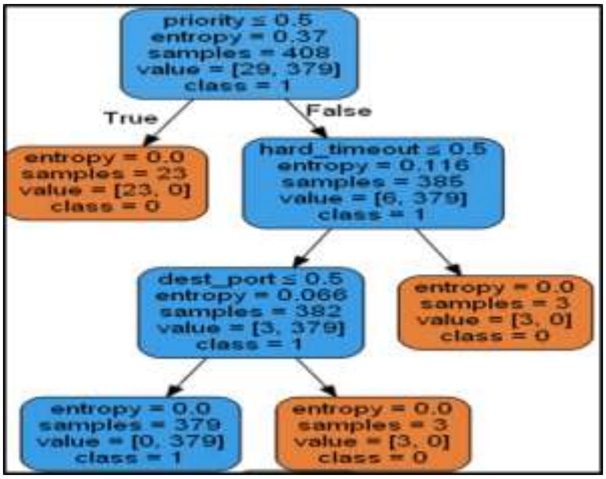


FIGURE 5. The diagram of DT algorithm.

B. SUPPORT VECTOR MACHINE (SVM) ALGORITHM

The Support Vector Machine, or SVM, is a binary supervised classifier employed in Machine Learning. The aim of the SVM algorithm is to build the perfect lines or determination boundaries for dividing n-dimensional area into categories so that specific data points can be easily placed in the appropriate category in the future. A hyper-plane is a term used to describe the best decision boundary. SVM selects the unique points/vectors that aid in the construction of the hyperplane. Help vectors are a term used to describe these extreme situations. In the classification method, two distinct classes use a decision boundary or hyper-plane, as shown in Fig. 6.

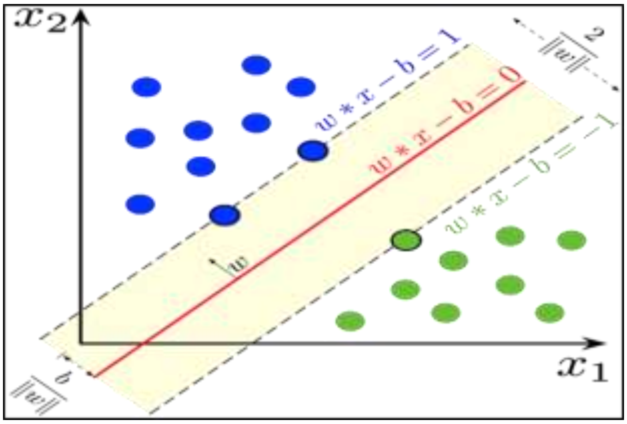


FIGURE 6. The two distinct groups for SVM algorithm.

A learning database of n points in the formula prescribed.

$$(x_1, y_1) \; .... \; (x_n, y_n) \tag{2}$$

For which $\mathbf{y_n}$ have either been 1 or 1, showing which category the value $\mathbf{x_i}$ corresponds for. Every $\mathbf{x_i}$ is a valid ***p-dimensional*** vector. The segment about which $\mathbf{x_i}$ corresponds is indicated by either 1 or 1 is yn. Each hyperplane can be defined as a collection of nodes that satisfy $\mathbf{x_i}$.

$$\boldsymbol{w^T x - b = 0} \tag{3}$$

The standard vector to the hyperplane is $\boldsymbol{w}$. That's also identical to the normal state of Hesse, with the exception that w is not actually a unit vector. Component b/(||w||) just specifies the offset of the hyperplane of its source across the standard vector $\boldsymbol{w}$. In Equation 4, everything else on or above that boundary belongs to a single class, marked 1. While in Equation 5, anything else on and below that boundary, with mark −1 is of the other class.

$$\boldsymbol{w^T x - b = 1} \tag{4}$$

$$\boldsymbol{w^T x - b = -1} \tag{5}$$

The range between such two hyper-planes, geometrically is 2/(||w||). It is important to reduce the ||w|| in order to increase the gap among the planes. The distance from a point to a plane equation is calculated using the distance. To prevent sets of data without falling into the margin, it also imposes the following constraints. Equation 6 or 7 is appropriate for each $\boldsymbol{i}$.

$$\boldsymbol{w^T x_i - b \geq 1, y_i = 1} \tag{6}$$

Or

$$\boldsymbol{w^T x_i - b < 1, y_i = \text{-}1} \tag{7}$$

Then each information pointed must have been on the right location of the line, according to these constraints. This can be rewritten as the following equation.:

$$\boldsymbol{y_i\,(w^T x_i - b) \geq 1, \textbf{for all } 1 \leq i \leq n} \tag{8}$$

Furthermore, the steps to implement the SVM algorithm can be summarised as follow:

- Implement support vector components in the Ryu controller.
- Setup the learner of the linear module.
- Apply hard margin function.
- Prepare and import all generated flows from OpenFlow switch for all flow sizes.
- Train the SVM classifier with 70% of flows generated.
- Predict the response test of 30% in generated flows.
- Evaluate the confusion matrix for the SVM algorithm and calculate running time.

C. EXTREMELY FAST DECISION TREE (EFDT)

The Extremely Fast Decision Tree (EFDT) is a novel learning algorithm that, when implemented with the Hoeffding Anytime Tree SEA Generator, is systematically more effective than the existing accepted decision tree algorithm. On several traditional benchmark tasks, the EFDT outperforms the Hoeffding Tree implementation Very Fast Decision Tree (VFDT) in terms of prequential accuracy. Domingos and Hulten implemented one of the first algorithms for progressively constructing a decision tree in their highly lauded research [35]. The Hoeffding Tree is the name of their method. Hoeffding Tree checks whether the difference between the average information improvements of the highest two parameters is going to provide a great meaning in almost any given potential break.

Hoeffding Bound: If $\boldsymbol{n}$ is independent random variables $\boldsymbol{r_1..r_n}$, with a wide variety $\boldsymbol{R}$ and mean $\boldsymbol{\check{r}}$, the Hoeffding bound declares in conjunction with probability $\mathbf{1-\delta}$ the real mean is at the very minimum $\boldsymbol{\check{r} - \epsilon}$ [36].

$$\epsilon = \frac{\sqrt{\boldsymbol{R^2\, ln(\frac{1}{\delta})}}}{\boldsymbol{2n}} \tag{9}$$

The Hoeffding Tree uses this deterministic guarantee to determine if the calculated variation of information changes is between the Xa and Xb attributes with the maximum data gains, respectively, around each node. Thus, $\Delta\breve{G}$ ($\boldsymbol{X_a}$) – $\Delta\breve{G}$ ($\boldsymbol{X_b}$), is positive and non-zero. Unless, for the tolerance stated, δ, it has $\boldsymbol{\Delta\breve{G} > \epsilon}$, then it confidently declares that $\boldsymbol{X_a}$ is the more advantageous division. It's worth noting that it aims to determine the best selection segment. The probabilities are monitored in the manner described before that $\boldsymbol{X_a}$ is superior to $\boldsymbol{X_b}$. However, the probability that $\boldsymbol{X_a}$ as superior to any other $\boldsymbol{X_c}$ feature is not regulated. If the selection of attributes increases, it is becoming more likely that every other category will prove to be better. In such situation, there is no recourse to modify the tree. Furthermore, the steps to implement the SVM algorithm can be summarised as follow:

- Setup the SEA Generator into DT components.
- Implement the Hoeffding Tree to the classifier.
- Setup and modify the Hoeffding Tree estimator to check action and IP address rules for generated flows.
- Setup new variables to control the loop of checking action and IP address rules.
- Prepare and import all generated flows from OpenFlow switch for all flow sizes.
- Train the EFDT algorithm with 70% of flows generated.
- Predict the response test for 30% of generated flows.
- Evaluate the confusion matrix for the EFDT algorithm and calculate running time.

D. HYBRID (DT-SVM) ALGORITHM

The learner of two algorithms was designed and implemented in one learner to enhance the precision and speed of execute time The Hybrid algorithm was purposefully implemented from two algorithms decision tree classifier and super vector classifier to enhance the performance of the two algorithms. One-vs-the-rest (OvR) multiclass strategy is utilised. This process is also known as one-vs-all, and consisted of fitting one classifier per class. One goal of this method is its interpretability, in addition to its fast computation (only n-class classifiers are required). While each class is identified only by one classifier, it is imperative to obtain information of the class by examining the related classifier. This is the most widely used multiclass classification technique and is a rational choice by default. Furthermore, the steps to implement the hybrid DT-SVM algorithm can be summarised as follows:

- Implement the DT classifier object together with the SVM classifier.
- Implement the OvR classifier.
- Setup and modify the OvR classifier for DT and SVM classifiers for action and IP address rules.
- Integrate DT and SVM classifiers.
- Setup and implement voting classifier to combine the prediction of DT and SVM classifiers.
- Prepare and import all generated flows from OpenFlow switch for all flow sizes.
- Train the hybrid DT-SVM algorithm with 70% of flows generated.
- Predicting the response test for 30% of generated flows.
- Evaluate the confusion matrix for the hybrid DT-SVM algorithm and calculate running time.

IV. EXPERIMENT RESULTS AND DISCUSSION

We have considered SDN dataset from our past studies in this analysis [37] for normal and conflict flows. There are seven types of conflict flows which are redundancy, shadowing, overlapping, correlation A, correlation B, generalisation, and imbrication. Fig. 7 shows the conflict flow types used in this study.

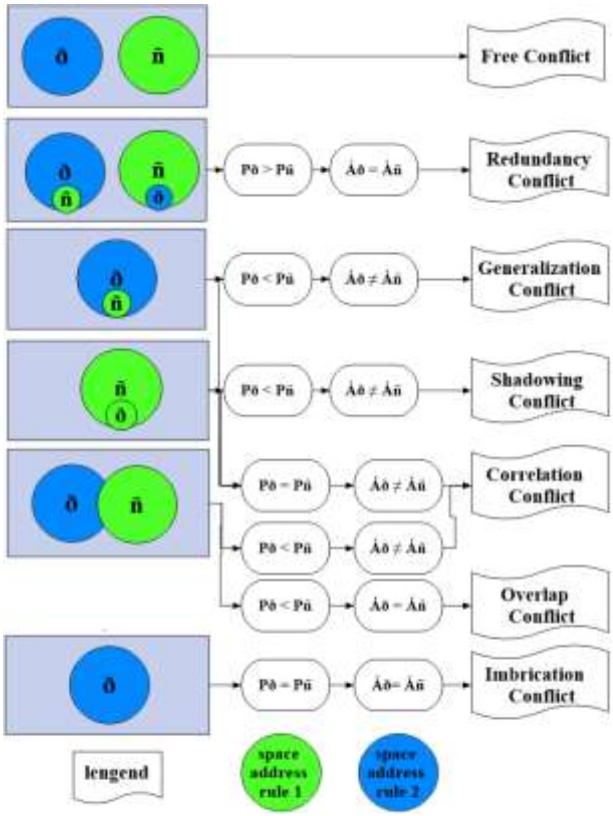


FIGURE 7. The conflict flow types.

The type of conflict can be specified and classified according to priority, action, protocol and IP source address of the flow rule. There would be conflicting flow entries between the flows as per the flow rule in the open flow switch. SDN can be influenced by conflict in various situations. These conflicts can affect the efficiency and optimisation of the network such as redundancy, overlap and correlation conflict. Moreover, it can affect the security of network such as shadowing generalisation and imbrication conflict [38].

Furthermore, there are two topologies used in this study, which are Fat Tree Topology and Simple Tree Topology. The Ryu controller is used in this experimental to make a link to an OpenFlow switch version 1.3, which enables both topologies to analyze data. These two topologies were performed in mininet and then connected to the Ryu controller to automatically generate the traffic. Fig. 8 and Fig. 9 show the architecture of Fat Tree and Simple Tree topologies, respectively. Besides, the Fat Tree topology contains 7 switches and 8 hosts, and the Simple Tree topology contains 3 switches and 4 hosts. The Ryu controller is associated to all switches and hosts in these topologies. Topo.py is a Python application that connects switches and hosts in these topologies, programmed and deployed over a python programming language. In order to produce the flows, traffic generation is performed to produce flows in the range of 1000-100000 flows (i.e., it starts at 1000 flows and finished at 100000 flows) in steps of 10000 flows increment.

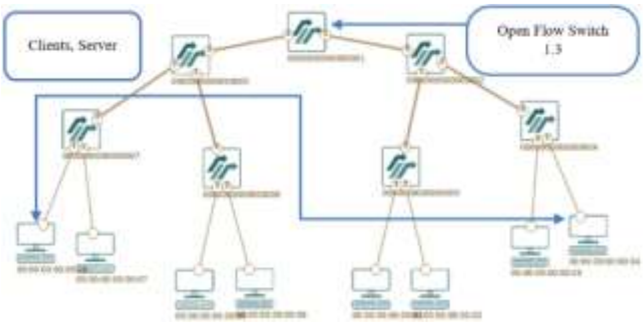


FIGURE 8. Fat Tree Topology.

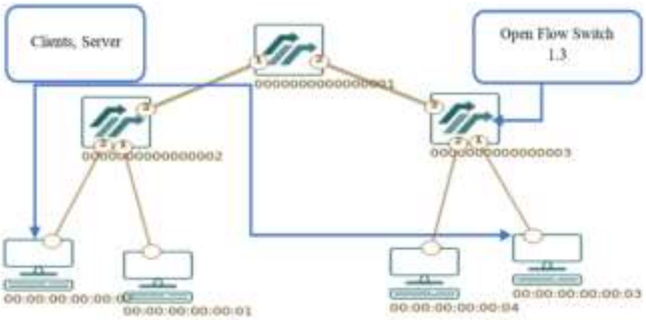


FIGURE 9. Simple Tree Topology.

Every host starts with 10 iperf servers, each of which listens to different ports such as 8089, 8082, and 8081. A simple switch is needed in the flow entries production step. As the L4 Match application is created, it was chosen as the basis framework. The src/dst ip, src/dst port, and protocols have been utilized to build different flows. The controller receives each packet. The controller then creates a new flow in the switch. Generally, after the topologies were created by running the *Topo* app, the number of flows will be selected and then the Ryu manager app will start running to generate normal flows. After the number of selected value of flows were generated, the conflicts rules will be implemented in the Ryu controller by running the conflicts flow app. When all generation of normal and conflict flows were completed, the flowstat app will be performed to collect and save all flows generated in a CSV file. Fig. 10 shows the flowchart used to produce and generate the flows.

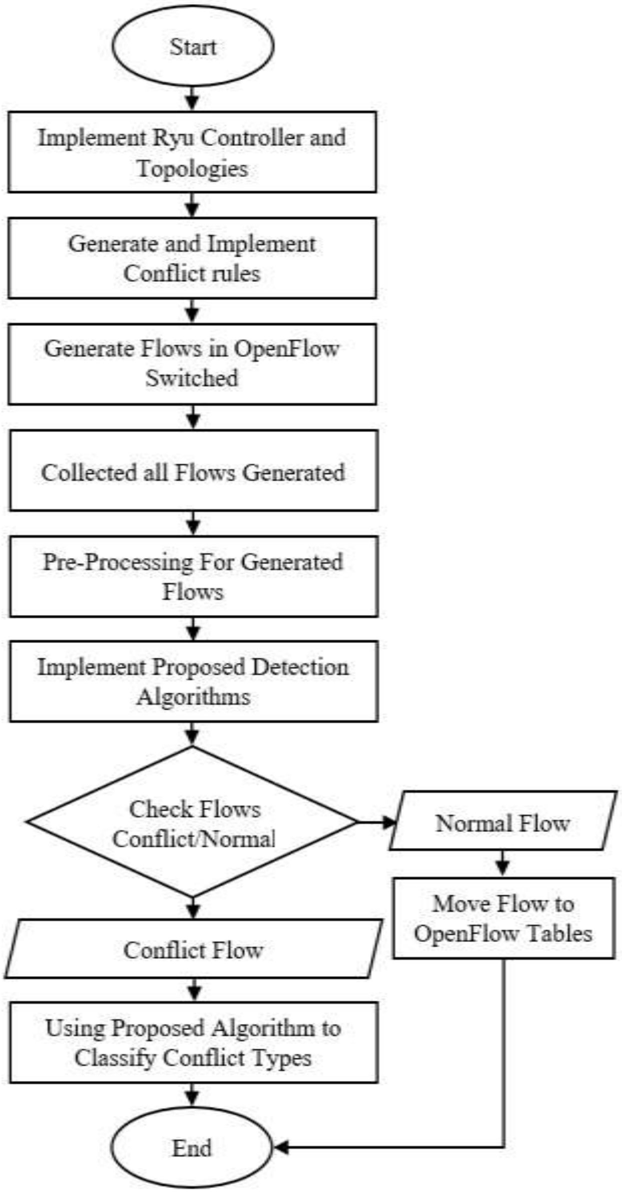

Start
Implement Ryu Controller and Topologies
Generate and Implement Conflict rules
Generate Flows in OpenFlow Switched
Collected all Flows Generated
Pre-Processing For Generated Flows
Implement Proposed Detection Algorithms
Check Flows Conflict/Normal
Normal Flow
Move Flow to OpenFlow Tables
Conflict Flow
Using Proposed Algorithm to Classify Conflict Types
End

FIGURE 10. **Flowchart of the flows generation.**

Both tests were carried out on a PC running Ubuntu 18.04, with an Intel Core-i5 CPU and 12 GB of RAM and using the Python programming language version 2.7. We utilized a variety of assessment measures such as accuracy, precision, f1-score, recall, and execution time to assess the performance of the proposed algorithms during the identification and classification of conflict flows in terms of efficiency and effectiveness. These evaluation measurements are computed as shown in equations (10-14).

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP+TN}}{(\boldsymbol{TP+TN+FN+FP})} \tag{10}$$

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}} \tag{11}$$

$$\boldsymbol{F1 - score} = \boldsymbol{2} \times \frac{(\boldsymbol{precision} \times \boldsymbol{recall})}{(\boldsymbol{precision} + \boldsymbol{recall})} \tag{12}$$

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \tag{13}$$

$$\boldsymbol{Execution\,Time}\ (\boldsymbol{T}) = \boldsymbol{T_{start}} - \boldsymbol{T_{finish}} \tag{14}$$

- The number of conflicts flows correctly classified is referred to as true positives (TPs).
- The number of correctly categorised normal flow records is known as true negatives (TNs).
- The number of natural flow records incorrectly labelled is known as false positives (FP).
- The number of conflict flow instances incorrectly graded is known as false negatives (FN).

The output of the suggested implementations has shown a variety of findings based on the experiments. When the number of flows was 1000, the algorithms of DT, SVM, and hybrid DT-SVM achieved the highest detection results with respect to accuracy precision, f1-score, and recall. The highest detection accuracies for DT, SVM, and hybrid DT-SVM algorithms were 99.27%, 98.53, and 99.27, respectively. In addition, the highest detection results for the EFDT algorithm were achieved when the number of flows was 100000. Meanwhile, the maximum detection accuracy for the EFDT algorithm was 99.49%. It is worth noting that the EFDT algorithm's highest precision, f1-score, and recall results were all 100%. Furthermore, the lowest execution time for all algorithms (i.e., DT, SVM, hybrid DT-SVM, and EFDT) was 0.00021 second when the number of flows is 20000. It is obvious now that the EFDT algorithm has obtained the best results as compared to the results of DT, SVM, and hybrid DT-SVM algorithms. However, when the number of flows was 10000, the minimum detection accuracies for DT, SVM, and hybrid DT-SVM algorithms were 72.60%, 64.60%, and 72.60%, respectively. While the minimum detection accuracy for the EFDT algorithm was 94.01% when the number of flows was 1000. Moreover, the longest execution time has taken more than 7 seconds for all algorithms when the number of flows was 1000 and 10000. Table 2 shows the detection results for DT, SVM, hybrid DT-SVM, and EFDT algorithms.

TABLE 2. Detection results by using all algorithms.

DT Algorithm					
Dataset	Accuracy	Precision	F1-score	Recall	Time (sec)
1000	**99.27%**	**98%**	**98%**	**97%**	**7.64E-06**
10000	72.60%	57%	61%	67%	7.63E-05
20000	81.74%	60%	63%	67%	0.00021
30000	81.37%	60%	63%	67%	0.01055
40000	82.08%	45%	47%	50%	0.01082
50000	81.26%	45%	47%	50%	0.01116
60000	81.08%	60%	63%	67%	0.01157
70000	80.91%	45%	47%	50%	0.01205
80000	81.43%	60%	63%	67%	0.01259
90000	81.03%	45%	47%	50%	0.0132
100000	80.48%	60%	63%	67%	0.01387
SVM Algorithm					
Dataset	Accuracy	Precision	F1-score	Recall	Time (sec)
1000	**98.53%**	**99%**	**95%**	**91%**	**7.73E-06**
10000	64.60%	64%	64%	65%	7.64E-05
20000	77.08%	67%	66%	67%	0.00021
30000	70.97%	65%	65%	65%	0.00042
40000	82.02%	45%	47%	50%	0.00069
50000	81.26%	45%	47%	50%	0.00103
60000	81.08%	60%	63%	67%	0.00144
70000	80.91%	45%	47%	50%	0.00191
80000	81.43%	60%	63%	67%	0.00245
90000	81.03%	45%	47%	50%	0.00306
100000	80.48%	60%	63%	67%	0.00374
EFDT Algorithm					
Dataset	Accuracy	Precision	F1-score	Recall	Time (sec)
1000	94.01%	92%	96%	**100%**	**7.44E-06**
10000	97.95%	98%	98%	99%	7.58E-05
20000	98.85%	99%	99%	99%	0.00021
30000	99.10%	99%	99%	99%	0.00041
40000	99.21%	99%	99%	**100%**	0.00069
50000	99.29%	99%	99%	**100%**	0.00102
60000	99.36%	**100%**	**100%**	**100%**	0.00143
70000	99.38%	**100%**	**100%**	99%	0.0019
80000	99.43%	**100%**	**100%**	**100%**	0.00248
90000	99.46%	**100%**	**100%**	**100%**	0.00308
100000	**99.49%**	**100%**	**100%**	**100%**	0.00377
Hybrid DT-SVM Algorithm					
Dataset	Accuracy	Precision	F1-score	Recall	Time (sec)
1000	**99.27%**	**99%**	**95%**	**91%**	**7.59E-06**
10000	72.60%	57%	61%	67%	7.73E-05
20000	81.74%	60%	63%	67%	0.00021
30000	81.37%	60%	63%	67%	0.00043
40000	82.08%	45%	47%	50%	0.0007
50000	81.26%	45%	47%	50%	0.00104
60000	81.08%	60%	63%	67%	0.00145
70000	80.91%	45%	47%	50%	0.00192
80000	81.43%	60%	63%	67%	0.00248
90000	81.03%	45%	47%	50%	0.00309
100000	80.48%	60%	63%	67%	0.00377

According to the detection results, we have selected the EFDT algorithm to be performed during classification phase in order to identify the conflict flows types. The EFDT algorithm was chosen because it has the best performance in detecting conflict flows. During the classification process, the number of flows is also chosen from a range of 1000 to 100000, with a 10000-flow multiplier. The performance of the proposed EFDT algorithm in classifying between conflict flows types has achieved the highest results based on a different number of flows, where the highest achieved accuracy and f1-score for the EFDT algorithm were 95.73% and 96.64% when the number of flows was 10000. The highest achieved precision was 97.61% when the flows were 1000. While the highest achieved recall was 100% when the flows were 30000, 70000, 80000, and 90000. Also, the lowest execution time taken for the classification using EFDT algorithm was 0.3248 second. However, the minimum classification accuracy was 90.16% when the flows were 1000. Table 3 shows the classification results using the proposed EFDT algorithm.

TABLE 3. **Classification results by using EFDT algorithm.**

Dataset	Accuracy	Precision	F1-score	Recall	Time (sec)
1000	90.16%	**97.61%**	91.56%	86.47%	**2.96E-02**
10000	**95.73%**	95.85%	**96.64%**	97.61%	0.3248
20000	93.58%	91.76%	95.35%	99.26%	1.0836
30000	93.75%	91.54%	95.58%	**100%**	2.31071
40000	93.31%	91.33%	95.24%	99.55%	14.2404
50000	94.08%	92.10%	95.79%	99.81%	16.4857
60000	95.05%	93.47%	96.50%	99.78%	19.1472
70000	94.68%	92.75%	96.23%	**100%**	22.3985
80000	94.48%	92.54%	96.12%	**100%**	26.0132
90000	94.32%	92.27%	95.98%	**100%**	30.1229
100000	93.99%	92.49%	95.65%	99.04%	34.7073

Furthermore, the suggested EFDT algorithm was compared to other methods in terms of efficiency [39] and [40] in term of the execution time during detection and classification of conflict flows. The work in [39] is implemented in the security policy analysis using the Brew module. The number of flows in this work is selected between 10000 to 100000 flows. Fig. 11 (a) shows the comparison of execution time between the proposed EFDT algorithm with the Brew module. While the work in [40] has presented a comprehensive framework called Flow Guard in the OpenFlow networks. In this work, the number of flows is selected between 10000 to 40000 flows. The proposed EFDT algorithm and the Flow Guard method are compared in terms of execution time in Fig. 11 (b). Both of these studies were performed in the detection and classification of conflict flows. The proposed EFDT algorithm outperformed its comparative methods in terms of execution time in the detection and classification of conflict flows, due to the experiment results.

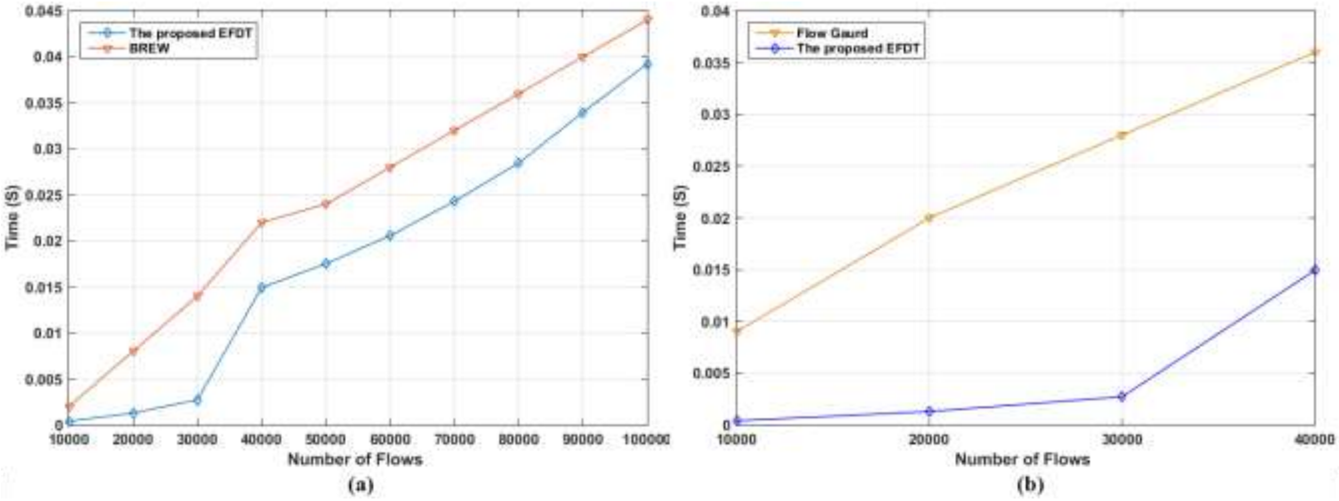


FIGURE 11. **Comparison of execution time between EFDT and other methods.**

V. CONCLUSION

We have provided various machine learning algorithms for detecting and classifying conflict flows in the SDN model in this study. The type of conflict can be detected and classified based on the priority, action, protocol, and IP source address of the flow rules. Furthermore, the algorithms that were utilized in this research are Decision Tree (DT), Support Vector Machine (SVM), Extremely Fast Decision Tree (EFDT) and the Hybrid (DT-SVM). The proposed EFDT and DT-SVM algorithms were developed and performed based on DT and SVM algorithms in order to enhance their performance with respect to efficiency and effectiveness. Besides, there were two network topologies designed which are Fat Tree Topology and Simple Tree Topology. These network topologies were created using the Mininet simulator and connected to the Ryu controller. For the dataset, the number of flows selected were to start at 1000 flows and finish at 100000 flows with an increment step of 10000 flows. The proposed algorithms were then evaluated by various evaluation measurements such as accuracy, precision, f1-score, recall, and execution time. Based on the experiment results, the proposed EFDT algorithm has achieved the best results compared to DT, SVM, and DT-SVM algorithms, where the EFDT has obtained 99.49% detection accuracy. While in the classification between conflict flow types, the proposed EFDT has achieved 95.73% accuracy. The proposed algorithm has the ability to achieve promising results in the SDN applications for the detection and classification of conflict flows. Other machine learning algorithms for detecting and classifying conflict flows may be applied and examined in the dataset in the future.

ACKNOWLEDGMENT

The authors would like to thank all those who had contributed towards making this research successful, especially the support from research team Computer Networks and System (CSNET). The authors also wish to express their gratitude to the Ministry of Education Malaysia (MOE) and Research Management Centre UTM (RMC).

REFERENCES

[1] C. Lo, P. Wu and Y. Kuo, "Flow entry conflict detection scheme for software-defined network," *2015 International Telecommunication Networks and Applications Conference (ITNAC)*, Sydney, NSW, Australia, 2015, pp. 220-225, doi: 10.1109/ATNAC.2015.7366816.

[2] M. H. H. Khairi, S. H. S. Ariffin, N. M. A. Latiff, A. S. Abdullah, and M. K. Hassan, "A Review of Anomaly Detection Techniques and Distributed Denial of Service (DDoS) on Software Defined Network (SDN)," *Engineering, Technology & Applied Science Research*, vol. 8, no. 2, pp. 2724-2730, Apr. 2018.

[3] A. Abdelaziz, A. T. Fong, A. Gani, S. Khan, F. Alotaibi, and M. K. Khan, "On Software-Defined Wireless Network (SDWN) Network Virtualization: Challenges and Open Issues," *The Computer Journal*, vol. 60, no. 10, pp. 1510-1519, Oct. 2017.

[4] M. Hamdan et al., "A comprehensive survey of load balancing techniques in software-defined network", *Journal of Network and Computer Applications*, vol. 174, no. 15, p. 102856, Jan. 2021.

[5] S. Khan, A. Gani, A. W. Abdul Wahab, M. Guizani and M. K. Khan, "Topology Discovery in Software Defined Networks: Threats, Taxonomy, and State-of-the-Art," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 303-324, Firstquarter 2017, doi: 10.1109/COMST.2016.2597193.

[6] S. Khan, M. A. Bagiwa, A. W. A. Wahab, A. Gani and A. Abdelaziz, "Understanding Link Fabrication Attack in Software Defined Network using Formal Methods," *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, 2020, pp. 555-562, doi: 10.1109/ICIoT48696.2020.9089520.

[7] Z. Bozakov and V. Sander, "OpenFlow: A Perspective for Building Versatile Networks," In: *Clemm A., Wolter R. (eds) Network-Embedded Management and Applications*, New York, NY, USA, Springer, 2013, pp. 217-245. [Online]. Available: https://doi.org/10.1007/978-1-4419-6769-5_11.

[8] A. A. Neghabi, N. Jafari Navimipour, M. Hosseinzadeh and A. Rezaee, "Load Balancing Mechanisms in the Software Defined Networks: A Systematic and Comprehensive Review of the Literature," *IEEE Access*, vol. 6, pp. 14159-14178, 2018, doi: 10.1109/ACCESS.2018.2805842.

[9] C. N. Tran and V. Danciu, "On Conflict Handling in Software-Defined Networks," *2018 International Conference on Advanced Computing and Applications (ACOMP)*, Ho Chi Minh City, 2018, pp. 50-57, doi: 10.1109/ACOMP.2018.00016.

[10] S. Xu, X. Wang and M. Huang, "Software-Defined Next-Generation Satellite Networks: Architecture, Challenges, and Solutions," *IEEE Access*, vol. 6, pp. 4027-4041, 2018, doi: 10.1109/ACCESS.2018.2793237.

[11] M. A. A. Albadr, S. Tiun, and F. T. Al-Dhief, "Evaluation of Machine Translation Systems and Related Procedures," *ARPN Journal of Engineering and Applied Sciences*, vol. 13, no. 12, pp. 3961-3972, Jun. 2018.

[12] M. A. A. Albadr and S. Tiun, "Extreme Learning Machine: A Review," *International Journal of Applied Engineering Research*, vol. 12, no. 14, pp. 4610-4623, 2017.

[13] M. A. Mohammed *et al*., "An Anti-Spam Detection Model for Emails of Multi Natural Language," *JOURNAL OF SOUTHWEST JIAOTONG UNIVERSITY*, vol. 54, no. 3, pp. 1-14, Jun. 2019.

[14] M. A. A. Albadr, S. Tiun, M. Ayob, F. T. Al-Dhief, K. Omar, F. A. Hamzah, "Optimised genetic algorithm-extreme learning machine approach for automatic COVID-19 detection," *PLoS ONE*, vol. 15, no. 12, pp. 1-28, Dec. 2020.

[15] O. I. Obaid, M. A. Mohammed, M. K. A. Ghani, S. A. Mostafa, F. T. AL-Dhief, "Evaluating the Performance of Machine Learning Techniques in the Classification of Wisconsin Breast Cancer," *International Journal of Engineering & Technology*, vol. 7, no. 4.36, pp. 160-166, Dec. 2018.

[16] F. T. AL-Dhief *et al*., "Voice Pathology Detection Using Machine Learning Technique," *2020 IEEE 5th International Symposium on Telecommunication Technologies (ISTT)*, Shah Alam, Malaysia, 2020, pp. 99-104, doi: 10.1109/ISTT50966.2020.9279346.

[17] F. T. Al-Dhief *et al*., "A Survey of Voice Pathology Surveillance Systems Based on Internet of Things and Machine Learning Algorithms," *IEEE Access*, vol. 8, pp. 64514-64533, 2020, doi: 10.1109/ACCESS.2020.2984925.

[18] M. A. Mohammed *et al*., "Voice Pathology Detection and Classification Using Convolutional Neural Network Model," *Applied Sciences*, vol. 10, no. 11, pp. 1-13, May 2020.

[19] M. A. A. Albadr, S. Tiun, F. T. Al-Dhief, M. A. M. Sammour, "Spoken language identification based on the enhanced self-adjusting extreme learning machine approach," *PLoS ONE*, vol. 13, no. 4, pp. 1-27, Apr. 2018.

[20] M. A. A. Albadr, S. Tiun, M. Ayob, and F. T. Al-Dhief, "Spoken language identification based on optimised genetic algorithm–extreme learning machine approach," *International Journal of Speech Technology*, vol. 22, no. 3, pp. 711-727, Sep. 2019.

[21] M. Hamdan *et al*., "Flow-Aware Elephant Flow Detection for Software-Defined Networks," *IEEE Access*, vol. 8, pp. 72585-72597, 2020, doi: 10.1109/ACCESS.2020.2987977.

[22] M.P.J. Kuranage, K. Piamrat, and S. Hamma, "Network Traffic Classification Using Machine Learning for Software Defined Networks," in *International Conference on Machine Learning for Networking*, Springer, Cham, Apr. 2020, pp. 28-39, doi: 10.1007/978-3-030-45778-5_3.

[23] S. Khamaiseh, E. Serra, Z. Li and D. Xu, "Detecting Saturation Attacks in SDN via Machine Learning," *2019 4th International Conference on Computing, Communications and Security (ICCCS)*, Rome, Italy, 2019, pp. 1-8, doi: 10.1109/CCCS.2019.8888049.

[24] F. Tang, H. Zhang, L. T. Yang and L. Chen, "Elephant Flow Detection and Differentiated Scheduling with Efficient Sampling and Classification," *IEEE Transactions on Cloud Computing*, doi: 10.1109/TCC.2019.2901669.

[25] R. U. Rasool, U. Ashraf, K. Ahmed, H. Wang, W. Rafique and Z. Anwar, "Cyberpulse: A Machine Learning Based Link Flooding Attack Mitigation System for Software Defined Networks," *IEEE Access*, vol. 7, pp. 34885-34899, 2019, doi: 10.1109/ACCESS.2019.2904236.

[26] K. Bao, J. D. Matyjas, F. Hu and S. Kumar, "Intelligent Software-Defined Mesh Networks With Link-Failure Adaptive Traffic Balancing," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 2, pp. 266-276, Jun. 2018, doi: 10.1109/TCCN.2018.2790974.

[27] Y. Yu, L. Guo, Y. Liu, J. Zheng and Y. Zong, "An Efficient SDN-Based DDoS Attack Detection and Rapid Response Platform in Vehicular Networks," *IEEE Access*, vol. 6, pp. 44570-44579, 2018, doi: 10.1109/ACCESS.2018.2854567.

[28] P. Wang, F. Ye, X. Chen and Y. Qian, "Datanet: Deep Learning Based Encrypted Network Traffic Classification in SDN Home Gateway," *IEEE Access*, vol. 6, pp. 55380-55391, 2018, doi: 10.1109/ACCESS.2018.2872430.

[29] D. Comaneci and C. Dobre, "Securing Networks Using SDN and Machine Learning," *2018 IEEE International Conference on Computational Science and Engineering (CSE)*, Bucharest, Romania, 2018, pp. 194-200, doi: 10.1109/CSE.2018.00034.

[30] Z. Fan and R. Liu, "Investigation of machine learning based network traffic classification," *2017 International Symposium on Wireless Communication Systems (ISWCS)*, Bologna, 2017, pp. 1-6, doi: 10.1109/ISWCS.2017.8108090.

[31] Y. Chen, J. Pei and D. Li, "DETPro: A High-Efficiency and Low-Latency System Against DDoS Attacks in SDN Based on Decision Tree," *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, 2019, pp. 1-6, doi: 10.1109/ICC.2019.8761580.

[32] C. Liu, Y. Chang, C. Tseng, Y. Yang, M. Lai and L. Chou, "SVM-based Classification Mechanism and Its Application in SDN Networks," *2018 10th International Conference on Communication Software and Networks (ICCSN)*, Chengdu, China, 2018, pp. 45-49, doi: 10.1109/ICCSN.2018.8488219.

[33] M. Balta and I. Özçelik, "A 3-stage fuzzy-decision tree model for traffic signal optimization in urban city via a SDN based VANET architecture," *Future Generation Computer Systems*, vol. 104, pp. 142-158, Mar. 2020.

[34] R. Santos, D. Souza, W. Santo, A. Ribeiro, and E. Moreno, "Machine learning algorithms to detect DDoS attacks in SDN," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 16, pp. 1-14, Jun. 2019.

[35] P. Domingos and G. Hulten, "Mining high-speed data streams," in Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, USA, 2000, pp. 71–80, doi: 10.1145/347090.347107.

[36] W. Hoeffding, "Probability Inequalities for sums of Bounded Random Variables," *The Collected Works of Wassily Hoeffding*, New York, NY, USA, Springer, 1994, pp. 409-426. [Online]. Available: https://doi.org/10.1007/978-1-4612-0865-5_26.

[37] M. H. H. Khairi, S. H. S. Ariffin, N. M. Abdul Latiff, and K. M. Yusof, "Generation and collection of data for normal and conflicting flows in software defined network flow table," Indonesian Journal of Electrical Engineering and Computer Science, vol. 22, no. 1, pp. 307 314, Apr. 2021.

[38] V. Danciu and C. N. Tran, "Side-Effects Causing Hidden Conflicts in Software-Defined Networks," SN Computer Science, vol. 1, no. 5, pp. 1-16, Aug. 2020.

[39] S. Pisharody, J. Natarajan, A. Chowdhary, A. Alshalan and D. Huang, "Brew: A Security Policy Analysis Framework for Distributed SDN-Based Cloud Environments," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 6, pp. 1011-1025, 1 Nov.-Dec. 2019, doi: 10.1109/TDSC.2017.2726066.

[40] H. Hu, W. Han, G.-J. Ahn, and Z. Zhao, “FlowGuard: Building robust Firewalls for software-defined networks,” in *Proc. 3rd Workshop Hot Topics Softw. Defined Netw*., 2014, pp. 97–102.

MUTAZ HAMED HUSSIEN is a researcher and ICT specialist with 16 years of wide range of research and ICT experience. Received his B.S. in Computer Engineering from Future University in 2002 and M.S. degrees in Electrical Engineering from Linkoping University in 2007, PhD Student at Universiti Technologi Malaysia (UTM) respectively. In 2002 -2007, he worked at Future University as a Lecturer in Faculty of Engineering and Director of Information Technology Department. His main research interests are Software Define Network (SDN), Machine learning, Telecommunication Network and Antenna Design and implementation.

SHARIFAH HAFIZAH SYED ARIFFIN received her B.Eng. (Hons) from London, in 1997, and obtained her MEE and Ph.D. in 2001 and 2006 from Universiti Teknologi Malaysia, and Queen Mary, University of London, London, respectively. She is currently an Associate Professor with the Faculty of Electrical Engineering, Universiti Teknologi Malaysia. Her current research interest are on Internet of Things, Ubiquitous computing and smart devices, Wireless sensor networks, IPV6, Handoff Management, Network and Mobile Computing System. She had published 116 papers, 17 copyrights, 1 integrated circuit, and 1 trademark.

NURUL MU’AZZAH ABDUL LATIFF is currently an Associate Professor at School of Electrical Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia, where she has been since 2002. She received her B.Eng. in Electrical-Telecommunications from Universiti Teknologi Malaysia in 2002, before pursued for her M.Sc. in Communications and Signal Processing at Newcastle University, UK in 2003. She completed her PhD in Wireless Telecommunication Engineering in 2008. Her research interests lie in the area of wireless networks, ranging from theory, to design and implementation. In particular, she has great interest in wireless sensor networks, mobile ad hoc networks, cognitive radio networks, Internet of Things, optimisation of wireless network based on Artificial Intelligence and machine learning algorithm for healthcare applications. She is a senior member of IEEE and an active volunteer in IEEE Communication/Vehicular Technology Society in Malaysia. She is also a Chartered Engineer of the Institution of Engineering and Technology (IET).

KAMALUDIN MOHAMAD YUSOF received the B.Eng. degree in Electrical - Electronics Engineering and M.Eng. Degree in Electrical Engineering from Universiti Teknologi Malaysia. He received Ph.D. degree from University of Essex, U.K. He is currently a senior lecturer in the Division of Communication Engineering, School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia and member of Communication Network System (CNetS). His current research interests include Internet-of-Things, Big Data and Software-Defined Network.

MOHAMED KHALAFALLA HASSAN is a researcher and ICT specialist with 16 years of wide range of research and ICT experience. Received his BSc in computer engineering in 2004 from future university Sudan and MSc from Universiti Putra Malaysia (UPM) in 2009 in communication network engineering. currently he is a PhD candidate in communication engineering at Universiti Technology Malaysia (UTM), he has 15 paper published in international peer reviewed conferences and journals, his main research interests are Forwards scattering Radar, Machine learning, NFV, vSDN and resources management in communication networks.

FAHAD TAHA AL-DHIEF (Student Member, IEEE) was born in Amarah, Iraq. He received the B.S. degree in software engineering from Imam Ja'afar Al-Sadiq University, Iraq, in 2013, and the M.S. degree in computer science from Universiti Kebangsaan Malaysia, Malaysia, in 2016. He is currently pursuing the Ph.D. degree in the School of Electrical Engineering, Universiti Teknologi Malaysia, Malaysia. His research interests are sensor networks, routing protocols, mobile ad hoc networks, social networks, the Internet of Things, machine learning, artificial neural networks, deep learning, and location-based service. He is a member of the IEEE Communications Society.

Mosab Hamdan (Student Member, IEEE) received a Ph.D. degree in the School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia (UTM), Malaysia, in 2021. He also received an M.Sc. in Computer Architecture and Networking from the University of Khartoum (UofK), Sudan, in 2014 and a B.Sc. degree in Computer and Electronic System Engineering from the University of Science and Technology (UST), Sudan, in 2010. He has published several high-impact research articles in reputed international journals and conferences. His current research interests are software-defined networking (SDN), load balancing, network traffic classification, internet-of-things (IoT), cloud computing, network security, and future network.

Suleman Khan received the Ph.D. degree (Distinction) in computer science and information technology from the Universiti Malaya, Malaysia, in 2017. He was a Faculty Member of the School of Information Technology, Monash University, Malaysia, from June 2017 to March 2019. He is currently a Faculty Member of the Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne, U.K. He has published more than 60 high-impact research articles in reputed international journals and conferences. His research areas include, but are not limited to, network forensics, software-defined networks, the Internet-of-Things, cloud computing, and vehicular communications.

Muzaffar Hamzah is a Deputy Dean (Graduate and Internationalization) and senior lecturer at the Faculty of Computing and Informatics, Universiti Malaysia Sabah, Malaysia. He obtained his BSc and MSc degree in computer science from University of Malaya, and PhD in information technology from University of South Australia. He has also completed his postdoctoral research in geovisualization with Nottingham University. He actively involved in research about HCI and information visualization. Currently he is interested in research related to information theory, visual analytics, and formal methods. He has also led several research grants at national level as principal investigator and recently recognized by IBIMA for Best Paper Award.

Authors Bibliography

MUTAZ HAMED HUSSIEN is a researcher and ICT specialist with 18 years of wide range of research and ICT experience. Received his B.S. in Computer Engineering from Future University in 2002 and M.S. degrees in Electrical Engineering from Linkoping University in 2007, PhD Student at Universiti Technologi Malaysia (UTM) respectively. In 2002 -2007, he worked at Future University as a Lecturer in Faculty of Engineering and Director of Information Technology Department. His main research interests are Software Define Network (SDN), Machine learning, Telecommunication Network and Antenna Design and implementation.

SHARIFAH HAFIZAH SYED ARIFFIN received her B.Eng. (Hons) from London, in 1997, and obtained her MEE and Ph.D. in 2001 and 2006 from Universiti Teknologi Malaysia, and Queen Mary, University of London, London, respectively. She is currently an Associate Professor with the Faculty of Electrical Engineering, Universiti Teknologi Malaysia. Her current research interest are on Internet of Things, Ubiquitous computing and smart devices, Wireless sensor networks, IPV6, Handoff Management, Network and Mobile Computing System. She had published 116 papers, 17 copyrights, 1 integrated circuit, and 1 trademark.

NURUL MU'AZZAH ABDUL LATIFF is currently an Associate Professor at School of Electrical Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia, where she has been since 2002. She received her B.Eng. in Electrical-Telecommunications from Universiti Teknologi Malaysia in 2002, before pursued for her M.Sc. in Communications and Signal Processing at Newcastle University, UK in 2003. She completed her PhD in Wireless Telecommunication Engineering in 2008. Her research interests lie in the area of wireless networks, ranging from theory, to design and implementation. In particular, she has great interest in wireless sensor networks, mobile ad hoc networks, cognitive radio networks, Internet of Things, optimisation of wireless network based on Artificial Intelligence and machine learning algorithm for healthcare applications. She is a senior member of IEEE and an active volunteer in IEEE Communication/Vehicular Technology Society in Malaysia. She is also a Chartered Engineer of the Institution of Engineering and Technology (IET).

KAMALUDIN MOHAMAD YUSOF received the B.Eng. degree in Electrical - Electronics Engineering and M.Eng. Degree in Electrical Engineering from Universiti Teknologi Malaysia. He received Ph.D. degree from University of Essex, U.K. He is currently a senior lecturer in the Division of Communication Engineering, School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia and member of Communication

Network System (CNetS). His current research interests include Internet-of-Things, Big Data and Software-Defined Network.

MOHAMED KHALAFALLA HASSAN is a researcher and ICT specialist with 16 years of wide range of research and ICT experience. Received his BSc in computer engineering in 2004 from future university Sudan and MSc from Universiti Putra Malaysia (UPM) in 2009 in communication network engineering. currently he is a PhD candidate in communication engineering at Universiti Technology Malaysia (UTM), he has 15 paper published in international peer reviewed conferences and journals, his main research interests are Forwards scattering Radar, Machine learning, NFV, vSDN and resources management in communication networks.

FAHAD TAHA AL-DHIEF (Student Member, IEEE) was born in Amarah, Iraq. He received the B.S. degree in software engineering from Imam Ja'afar Al-Sadiq University, Iraq, in 2013, and the M.S. degree in computer science from Universiti Kebangsaan Malaysia, Malaysia, in 2016. He is currently pursuing the Ph.D. degree in the School of Electrical Engineering, Universiti Teknologi Malaysia, Malaysia. His research interests are sensor networks, routing protocols, mobile ad hoc networks, social networks, the Internet of Things, machine learning, artificial neural networks, deep learning, and location-based service. He is a member of the IEEE Communications Society.

Mosab Hamdan received a Ph.D. degree in the School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia (UTM), Malaysia, in 2021. He also received an M.Sc. in Computer Architecture and Networking from the University of Khartoum (UofK), Sudan, in 2014 and a B.Sc. degree in Computer and Electronic System Engineering from the University of Science and Technology (UST), Sudan, in 2010. He has published several high-impact research articles in reputed international journals and conferences. His current research interests are software-defined networking (SDN), load balancing, network traffic classification, internet-of-things (IoT), cloud computing, network security, and future network.

Suleman Khan received the Ph.D. degree (Distinction) in computer science and information technology from the Universiti Malaya, Malaysia, in 2017. He was a Faculty Member of the School of Information Technology, Monash University, Malaysia, from June 2017 to March 2019. He is currently a Faculty Member of the Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne, U.K. He has published more than 60 high-impact research articles in reputed international journals and conferences. His research areas include, but are not limited to, network forensics, software-defined networks, the Internet-of-Things, cloud computing, and vehicular communications.

Muzaffar Hamzah is a Deputy Dean (Graduate and Internationalization) and senior lecturer at the Faculty of Computing and Informatics, Universiti Malaysia Sabah, Malaysia. He obtained his BSc and MSc degree in computer science from University of Malaya, and PhD in information technology from University of South Australia. He has also completed his postdoctoral research in geovisualization with Nottingham University. He actively involved in research about HCI and information visualization. Currently he is interested in research related to information theory, visual analytics, and formal methods. He has also led several research grants at national level as principal investigator and recently recognized by IBIMA for Best Paper Award.

Authors Photographs

MUTAZ HAMED HUSSIEN

SHARIFAH HAFIZAH SYED ARIFFIN

NURUL MU’AZZAH ABDUL LATIFF

KAMALUDIN MOHAMAD YUSOF

MOHAMED KHALAFALLA HASSAN

FAHAD TAHA AL-DHIEF

Mosab Hamdan

Suleman Khan

Muzaffar Hamzah

Conflict of Interest

The authors declared that there is no conflict of interest among any authors.