Hybrid SDN Deployment Using Machine Learning

H. W. Siew, S. C. Tan, and C. K. Lee

Abstract Software-Defined Networking (SDN) has attracted tremendous attention in recent years as the future communication network architecture. However, SDN deployment in legacy network will be progressively phased over a period, especially for larger network which consists of hundred or more nodes. Every migration (i.e. replacing or upgrading) of SDN-enabled nodes requires considerable optimization efforts in terms of cost of investment, network stability and performance gains. Hitherto literatures have proposed variety of static heuristic algorithms to compute the migration sequence of SDN-enabled nodes for multi-periods SDN deployment in legacy network. The aim of each computed migration sequence is aims to improve network performance gains with respect to address different constraints. However, the dynamicity of an unique network, such as traffic growth or topology change, cannot be comprehensively addressed using a static heuristic algorithm over the deployment duration. Machine learning (ML), on the other hand, has been proven successfully applied for various dynamic and non-linear problems in diverse domains. In this article, we summarize the generic workflow for ML in networking domain at first. Subsequently, we investigated the problem of SDN deployment in legacy network from the perspective of ML. We proposed a SDN deployment problem that formulated as Markov Decision Process and reinforcement learning techniques, such as Qlearning and SARSA, can be used to model for the problem.

Keywords Machine learning · Software-defined networking · Hybrid SDN deployment

H. W. Siew · S. C. Tan (✉)
Faculty of Computing and Informatics, Multimedia University, Cyberjaya, Malaysia
e-mail: sctan1@mmu.edu.my

C. K. Lee
Faculty of Engineering, Multimedia University, Cyberjaya, Malaysia

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2021

R. Alfred et al. (eds.), *Computational Science and Technology*, Lecture Notes in Electrical Engineering 724, https://doi.org/10.1007/978-981-33-4069-5_19

1 Introduction

Machine learning (ML) applies widely in diverse domains, such as speech recognition, computer vision, and autonomous vehicle. The unprecedented power of ML enables a system to extract knowledge from quality data [1]. Unlike the traditional programming approach of tackling a problem, ML uncover hidden rules or patterns via the discovery process with data (a.k.a. the learning process). The learned pattern, i.e. model, is then used to answer unknown data of the particular problem. In general, there are four classes of problems can leverage on the techniques of ML, namely, classification, clustering, regression, and rule extraction [2]. In classification and regression problems, new input data is mapped respectively to discrete or continuous output value. Whereas, the goal of clustering problem is to divide data points into groups alike. On the contrary, rule extraction problems are inherently different from others which the objective is to establish statistical relationships in data. A different learning paradigm of ML is employed for each class of problems. Supervised learning uses labelled data to identify hidden behaviours within datasets. Commonly, supervised learning is used to create model of classification and regression problems, in which, to classify discrete or to predict continuous output value. However, labelled data are not always available for all problems. Unsupervised learning, on the other hand, utilizes unlabelled data to train and learn knowledge from datasets. Ultimately, unsupervised learning targets to discriminate groups in the data, and this approach best suits clustering problems. In contrast, reinforcement learning is a machine learning paradigm which constantly learns through interactions (a series of actions) with the environment and observe the result to adjust its strategy automatically [3]. Consequently, the agent aims to extract optimal rules or policy for the problem.

Recently, ML has regained attraction in communications and networking domain to improve how networking problems are addressing today [4]. Communication networks are growing shockingly complex with wide spectrum of applications. Network operation and management remains tedious and error prone with human factor involved [5]. The diversity and complexity of communication networks has made designing scalable network solutions difficult. Often, solutions are built for network scenarios specific to its particular, such as type of applications, user demand and topology. Nonetheless, it is challenging to model an accurate representation of a complex network behaviours, for instance, loads pattern in Content Distribution Network (CDN) [6]. Furthermore, the dynamics of network inhibits developing efficient algorithms to cater different scenarios across networks. Therefore, there is a raising demand for cognitive management and operation in today networks [7]. Network operators are growing interest to build a highly resilient autonomic network as proposed in [8]. In which, an autonomic network comprises of self-configuration, self-healing, self-optimization and self-protection characteristics. Future networks is likely to operate autonomously by monitoring its own state and the environment together with their complex configuration. And, techniques of ML serve as a tool to facilitate decision making and network automation [4].

The idea of incorporating intelligence into network management and operation has been discussed for years in research community. However, such system has not been deployed or developed yet in existing networks [9]. One of the biggest challenges is that exiting network architecture is inherently distributed in nature [10]. Switches and router have only limited view and control over the whole network. For instance, legacy network devices are still restricted by vendor specific commands and functionalities, and It is extremely difficult to orchestrate such devices in heterogeneous network environment. Data availability and processing poses another challenge regards the deployment of autonomic system in existing networks [11]. Questions often arise where and what data can be collected from existing network. In addition, the training process of ML techniques with data requires tremendous computing and storage resources. Nevertheless, the advancement of recent technology has lowered the barrier for ML adoption in networking. Software-Defined Networking (SDN) [12], for example, decouples network controls (control plane) from its forwarding (data plane). The separation of control and data planes offers programmability through centralized controller. The programmability enables external software (e.g. ML applications) to define network behaviours with global network view. The prevalent of cloud computing nowadays alleviates the obstacle of demanding computing resources for ML techniques [13]. Even more, the availability of Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs) in cloud accelerates the model training process of ML.

SDN promotes the applicability of ML for networking in which centralized controller offers global network view on top of programmability. However, the adoption of SDN in legacy network encountering several challenges [14], such as financial and technological constraints. To illustrate, budget consideration prohibits larger network operator to migrate hundreds of nodes to SDN-capable devices at once. Hence, SDN deployment in legacy network is likely to migrate nodes spanning over a period. The coexistence and interoperability of both legacy and SDN-capable devices in a network forms a hybrid SDN (hSDN) [15]. The deployment of hSDN involves legacy nodes selection for SDN-capable device migration in each period. Network operators are eager to understand with limited resources which legacy node and when it should be migrated. Ultimately, the deployment of hSDN can reap most benefits in term of network performance gain. Many studies [16–20] have been contributed to propose algorithms in seeking optimal migration sequence (order of nodes to migrate) of hybrid SDN deployment. In essence, an algorithm captures a snapshot of network traffic condition and compute the maximized performance gain with respect to different constrains, such as budget and link capacity. However, this approach underestimates the volatile of a network where traffic demand fluctuates, and network topology is subject to change over time. Hence, these algorithms fail to adapt the dynamicity of a network in the course of hybrid SDN deployment over months or years.

Literature of hybrid SDN deployment in legacy network exposes a research gap, in which, the dynamicity of a network has not been properly addressed in the previously studies. On the contrary, ML offers inherent adaptability with data learning process which caters dynamicity. Therefore, we are interested to investigate the applicability

of ML in the aspect of SDN node deployment. In this article, we aim to provide our insights and a new perspective to apply ML technique with hSDN deployment in legacy network. The following summarizes the contributions of this.

Generic workflow of ML in networking is summarized here to provide a basic practical guideline for applying ML in networking.

We expand the generic workflow of ML and illustrate the feasibility to apply ML technique for hSDN deployment problem in legacy network.

The following of this articles, related work is discussed in Sect. 2. In Sect. 3, we summarize the generic workflow of ML in networking. We expand the generic workflow in Sect. 4 with respect to SDN node migration in legacy network. Lastly, we conclude the article in Sect. 5.

2 Related Works

Large body of literature [16, 17, 19, 21–23] have been proposed to offer static heuristic algorithms computing the node migration sequence for hSDN deployment in legacy network. On one hand, these works consider the network topology remains unchanged throughout the deployment of hSDN. On the other, these proposed algorithms do not capture the possible growth of traffic in a network. Generally, the assumptions of both, unchanged topology and stagnant traffic, are unrealistic for a deployment timeframe possibly in years. Although [24] take into account of multiple traffic matrices in its computation of migration sequence, it considers only the past traffic fluctuations which does not sufficiently represent the dynamic (i.e. network topology change) a network may have after the deployment started. ML has been proven applicable to solve various problems in vast domains including networking. In networking, ML has play an significant role in traffic prediction [25] to forecast future traffic, traffic classification [26] to facilitate network operations and planning, traffic routing [27] and congestion control [28] to optimize resource utilization. Moreover, ML can also work in network management activities such as QoS management [29], fault management [30] and network security [31]. Nonetheless, SDN deployment planning has not been extensively discussed in the perspective of ML techniques [32]. Hence, instead of the static heuristic approach, we examine the feasibility of applying ML techniques in the problem of hSDN deployment in legacy network here.

3 Generic Workflow for Machine Learning in Networking

Figure 1 illustrates a generic workflow to apply machine learning in various domains including fields of networking [13]. In summary, the workflow consists of multiple steps, namely problem formulation, data collection, feature engineering, model construction and evaluation. Each step in the workflow is not independent but strongly interrelated between problem, training data and learning paradigm [11]. For instance,

Fig. 1 The generic workflow of machine learning in networking

the result of a ML model depends largely on the collection and preprocessing of available data. In this section, we review each step in details in order to properly develop machine learning applications for networking related problems.

Problem Formulation: There are many different possible approaches to leverage ML for a networking problem. However, the process of training a ML model often requires huge amount of resources (e.g. time and investment). Therefore, it is utmost important to formulate correctly a networking problem at hand in the first step. Otherwise, an ill-formed problem will end at unsatisfactory performance as a result. In this step, a well formulated problem can be categorized into one of the ML paradigms, such as supervised, unsupervised or reinforcement learning. This helps to determine what kind of data are required for collection, and what learning algorithms to choose from for model construction. For example, a problem cannot be formulated as supervised learning if there is lacking label data for model training.

Data Collection: Various ML techniques share one common requirement which large amount of unbiased representative data is necessary to build an effective model for a designated problem in networking. According to the needs, network data can be monitored and recorded from different network layers, for instance Simple Network Management Protocol (SNMP) [11]. It is important to note that representative data vary from one problem to another even in the same domain. For instance, traffic prediction and traffic classification require different details of network data. Typically, data collection in the context of ML for networking is accomplished in two phases, offline and online. A sizable amount of historical data is gathered in the offline phase for model training and testing purpose. In online phase, real-time network data (e.g. performance information and network state) are collected and feed as input for the model retraining or used as a feedback to the model.

Feature Engineering: Every problem in networking is characterized by a number of factors. However, only few of defining factors (i.e. features) has the significant impact on the targeted network problem. Broadly, these features are categorized by its granularity level, for instance, connection-level, flow-level, and packet-level features [11]. The extraction of defining features in this step is crucial to unleash the pattern in data via different ML paradigms. The goal of this step attempts to analyze historical data and extract the effective features for model construction in next. Nonetheless, network data collected are often noisy or incomplete. Therefore, it is necessary to clean up data by going through a preprocessing phase prior to feature extraction. Feature extraction can be difficult which requires to have a thorough understanding regards the target problem with domain-specific knowledge [6]. Deep learning, on

the other hand, can ease to automate feature extraction in some of the problems in networking.

Model Construction: In model construction step, a suitable learning algorithm is chosen in accordance to the characteristics (e.g. problem category, size of dataset and etc.) of the target network problem defined. Prior to start training the selected model, the collected historical dataset is divided into training, validation and test datasets. It is important that all training, validation and test datasets are independent but follow the same probability distribution [11]. This prevents the generalization of training outcome which leads to model overfitting or under-fitting. Along the way of training selected model, training dataset also helps for hyper-parameter tuning in the offline phase. The process of parameters tuning involves finding acceptable parameters for building the selected model. While training dataset used for training, test dataset is used to evaluate the accuracy of the trained model. It is possible that this step is repeated until satisfied result is obtained.

Model Validation: Validation is an essential step for ML workflow in order to assess the performance of the trained model and how well it would generalize to new data. K-fold cross validation is often used to validate the accuracy of a model in overall. The result of validation offers insights on how to further optimize the model.

Furthermore, sources of error can be identified through model validation to determine if the model or feature are appropriate or data are representative enough for the target problem. If necessary previous steps can be re-visited according to the error sources discovered in the validation step.

4 Machine Learning for hSDN Deployment

The usages of the Internet have evolved over the past decades which increasingly demand network infrastructure to handle dynamic nature of future network operations and applications. For this reason, SDN has gained tremendous attention for the past few years as a next generation architecture of communication network. Among all, the most distinctive features of SDN are listed as follow,

1. Separation of network control (control plane) from forwarding (data plane)
2. A centralized controller with global network view
3. Network programmability by the external applications.

In short, SDN offers flexible configurability by external software, and allows network to be dynamically optimized in conjunction with the global network status. The benefit of SDN over its advantage of network controllability is tempting to network operators. However, full SDN deployment in an existing legacy network encounters challenges in organizational, economical, and technical aspects [14]. Nonetheless, literature has suggested that the benefit of SDN can be realized without the need to fully deploy SDN nodes in a legacy network [16, 20]. In most cases, network operators tend to gradually deploy hSDN in legacy networks which span

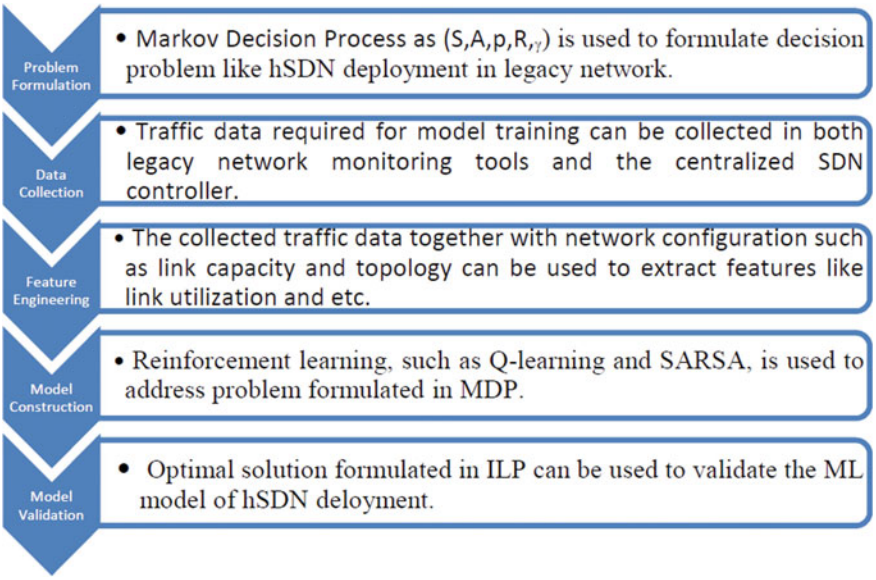

Fig. 2 ML framework for hSDN deployment in legacy network

across multiple periods over months or years. The gradual deployment of hSDN in legacy network mainly results from the budget and technical considerations, especially for large network which consists of hundred or more nodes. In each deployment period, a number of nodes are selected to be migrated (e.g. replace or upgrade) to SDN enable devices in line with the budget available and other constraints (e.g. link capacity). Unlike the deployment of full SDN, hSDN deployment requires detail planning in selection of nodes to be migrated. Network operators are keen to understand which node at when should be migrated to maximize the return of investment. In this section, we apply the previously discussed workflow of ML for the problem of SDN deployment in legacy network as summarized in Fig. 2.

Problem Formulation: The deployment of hSDN in legacy network has been formulated as an optimization problem in literature. In which, hSDN deployment aims to optimize particular network performance matrix, e.g. maximum link utilization (MLU) or alternative paths, with respect to a number of constraints such as link capacity, budget and etc. Moreover, the hSDN deployment problem is an offline migration operation during network planning phase. During the planning process, decisions have to be made which legacy node at when should be migrated for hSDN deployment. Consequently, the decision making nature of hSDN deployment problem can be cast as Markov Decision Process (MDP) as illustrated in Fig. 3. In general, a tuple (S, A, p, R, γ) is used to represent a MDP where S denotes a finite set of states, A denotes a finite set of actions, p denotes the probability of state transition from s with action a to s', R is the reward obtained after execution of action a, and discount factor $\gamma \in [0,1]$ represents the importance of future reward as compared to

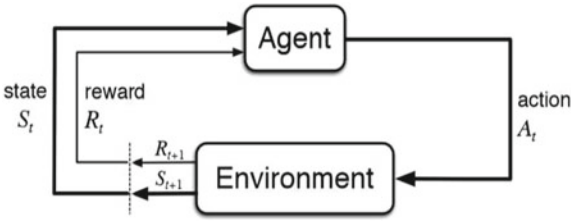

Fig. 3 Markov decision process

the immediate reward. Simply put, the goal of a MDP is to maximize the total reward by finding an optimal policy $\pi *$, and policy π is a function mapping from a state to an action.

Data Collection: Representative network data, for instance traffic matrix (TM), is necessary to properly train a ML model for hSDN deployment problem. Before hSDN deployment in legacy network, historical TMs can be acquired from network monitoring tools such as Cisco Net-Flow and IP Flow Information Export [11]. Once the deployment of hSDN started, network data can be aggregated from both legacy network monitoring tools and the centralized SDN controller. Among all, dynamic network data like TMs helps to understand the fluctuation of traffic demand in a network, and other data such network topology, links capacity and links weight settings are required to understand the configuration of a network.

Feature Engineering: MDP requires states to be defined for an environment. In the context of hSDN deployment, the environment consists of both legacy and SDN-enable nodes in a network. With every action of deploying a SDN node, the environment changes (i.e. states) in return for network performance gain. Eventually, a series of SDN node placement needs to be determined in order to achieve the optimal performance improvement. A state, in that sense, should be efficiently represented by distinctive features that reflects the performance gain acquired through the placement of SDN node. Specifically, features for hSDN deployment problem include link utilization, MLU, number alternative paths and etc. These features can be extracted from collected data such as TM and network topology configuration.

Model Construction: Commonly, reinforcement learning (RL) is used to address problem formulated as an MDP. Reinforcement learning algorithms learn through trial and error by interacting with the environment. An agent sequentially makes decisions (actions) and observes the outcomes (rewards) in environment (states). Ultimately, an agent targets to achieve the optimal policy through adjusting its strategy in making decisions [3]. For a problem which transition probability and reward models are known, model-based reinforcement learning such as value-iteration or policy-iteration algorithm can be used to solve the problem. In most cases, however, transition probability and reward models are difficult to define in a dynamic environment like the hSDN deployment in legacy network. Thus, a model-free reinforcement learning such as Q learning or SARSA can be used for ML model construction.

Model Validation: Typically, hSDN deployment is formulated as integer linear programming (ILP) problem in literature. In which, the ILP of hSDN deployment maximizes the network performance gain with respect to constraints, such as budget and link capacity. Optimal solution of such ILP problem can be computed using an ILP solver. Therefore, an optimal solution serves as a good candidate for benchmarking the RL model developed for hSDN deployment. However, ILP solver can take significant amount of time to compute an optimal solution for a large network. Heuristic algorithms developed in literature can then be used to approximate a sub-optimal solution for validating the RL model.

5 Conclusion

In summary, the dynamic nature of future communication network demands flexible network controllability. SDN has attracted tremendous attention recent years to provide such controllability via standardization and centralized controllers. However, full SDN deployment in legacy network is not always possible especially for large network due to various considerations. Large body of literature had proposed static approaches to offer gradual deployment of SDN in legacy network. However, networks' dynamicity, such as growth of traffic or change in topology, has not been properly addressed in the past. Machine learning, on the other hand, has been applied in various aspects of networking other than SDN deployment in legacy network. Techniques of machine learning have shown great success in revealing pattern with dynamic data. Thereby, we summarized the generic workflow of ML in networking, and we proposed to formulate hSDN deployment in legacy network using Markov Decision Process. In additional, technique of reinforcement learning, such as Q-learning or SARSA, can be used for model construction and validation. Moreover, further work is necessary to develop and evaluate the proposed approach here for hSDN deployment in legacy network.

References

1. Brill E, Lin J, Banko M, Dumais S, Ng A (2002) Data-intensive question answering. In: Proceedings of the TREC-10 conference, pp 183–189
2. Practical machine learning problems. https://machinelearningmastery.com/practical-machine-learning-problems/. Accessed 11 Nov 2019
3. Luong NC et al (2019) Applications of deep reinforcement learning in communications and networking: a survey. IEEE Commun Surv Tutor 21(4):3133–3174
4. Chemouil P et al (2019) Artificial intelligence and machine learning for networking and communications. IEEE J Sel Areas Commun 37(6):1185–1191
5. Mahmoud QH (2007) Cognitive networks: towards self-aware networks. Wiley, Hoboken
6. JiangJ, Sekar V, Zhang H, Milner H, Shepherd D, Stoica I (2016) CFA: a practical prediction system for video QoE optimization

7. Ramming JC, Wroclawski JT, Clark DD, Partridge C (2015) A knowledge plane for the internet. In: Proceedings of the 2007 conference on applications, technologies, architectures, and protocols for computer communication, pp 3–10
8. White RS, Hanson EJ, Whalley I, Chess MD, Kephart JO (2004, October) An architectural approach to autonomic computing ('An architectural blueprint for autonomic computing'). In: International conference on autonomic computing, pp 2–9
9. Mestres A et al (2017) Knowledge-defined networking. Comput Commun Rev 47(3):1–10
10. Ayoubi S et al (2018) Machine learning for cognitive network management. IEEE Commun Mag 56(January):158–165
11. Boutaba R et al (2018) A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. J Internet Serv Appl 9(1)
12. Chen J, Zheng X, Rong C (2015) Survey on software-defined networking. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol 9106, no 1, pp 115–124
13. Wang M, Cui Y, Wang X, Xiao S, Jiang J (2018) Machine learning for networking: workflow, advances and opportunities. IEEE Netw 32(2):92–99
14. Vissicchio S, Vanbever L, Bonaventure O (2014) Opportunities and research challenges of hybrid software defined networks. ACM SIGCOMM Comput Commun Rev 44(2):70–75
15. Amin R, Reisslein M, Shah N (2018) Hybrid SDN networks: a survey of existing approaches. IEEE Commun Surv Tutorials 20(4):3259–3306
16. Poularakis K, Iosifidis G, Smaragdakis G, Tassiulas L (2019) Optimizing gradual SDN upgrades in ISP networks. IEEE/ACM Trans Netw 27(1):288–301
17. Poularakis K, Iosifidis G, Smaragdakis G, Tassiulas L (2017) One step at a time: optimizing SDN upgrades in ISP networks. In: Proceedings—IEEE INFOCOM, pp 1–9
18. Guo Y, Wang Z, Yin X, Shi X, Wu J, Zhang H (2016) Incremental deployment for traffic engineering in hybrid SDN network. In: 2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC 2015)
19. Xu H, Li XY, Huang L, Deng H, Huang H, Wang H (2017) Incremental deployment and throughput maximization routing for a hybrid SDN. IEEE/ACM Trans Netw 25(3):1861–1875
20. Levin D, Canini M, Schmid S, Feldmann A (2013) Incremental SDN deployment in enterprise networks. In: Proceedings of the ACM SIGCOMM 2013 conference SIGCOMM—SIGCOMM'13, p 473
21. Tanha M, Sajjadi D, Ruby R, Pan J (2018) Traffic engineering enhancement by progressive migration to SDN. IEEE Commun Lett 22(3):438–441
22. Yuan T, Huang X, Ma M, Zhang P (2017) Migration to software-defined networks: the customers view. China Commun 14(10):1–11
23. Wang W, He W, Su J (2017) Boosting the Benefits of Hybrid SDN. In: Proceedings of the international conference on distributed computing systems, pp 2165–2170
24. Guo Y, Wang Z, Yin X, Shi X, Wu J (2017) Traffic engineering in hybrid SDN networks with multiple traffic matrices. Comput Netw 126:187–199
25. Poupart P et al (2016) Online flow size prediction for improved network routing In: Proceedings of the International Conference on Network Protocols ICNP, December 2016, pp 1–6
26. Wang R, Liu Y, Yang Y, Zhou X (2006) Solving the app-level classification problem of P2P traffic Via optimized support vector machines. In: Proceedings of the ISDA 2006 sixth international conference on intelligent systems design and applications, vol 2, pp 534–539
27. Hu T, Member S, Fei Y (2010) QELAR: a Machine-learning-based adaptive routing protocol for. IEEE Trans Mob Comput 9(6):796–809
28. Jayaraj A, Venkatesh T, Murthy CSR (2008) Loss classification in optical burst switching networks using machine learning techniques: improving the performance of TCP. IEEE J Sel Areas Commun 26(6):45–54
29. Demirbilek E, Gregoire JC (2017) Machine learning based reduced reference bitstream audio-visual quality prediction models for realtime communications. In: Proceedings of the IEEE international conference on multimedia and expo, vol 13, no 2, pp 571–576

30. Kumar Y, Farooq H, Imran A (2017) Fault prediction and reliability analysis in a real cellular network. In: 2017 13th International Conference on Wireless and Mobile Communications IWCMC 2017, pp 1090–1095
31. Cannady JD (1998) Artificial neural networks for misuse detection. In: Proceedings of the 21st national information systems security conference, pp 368–381
32. Wei SH, Chin TS, Binlun JN, Kwang LC, Kapsin R, Yusoff Z Machine learning as a means to adapt requirement changes for SDN deployment process in SDN migration. In: Advances in computational intelligence, pp 629–639.