IHSF: An Intelligent Solution for Improved Performance of Reliable and Time-Sensitive Flows in Hybrid SDN-Based FC IoT Systems

Muhammad Ibrar, Lei Wang, Gabriel-Miro Muntean, *Senior Member, IEEE*, Jenhui Chen, *Senior Member, IEEE*, Nadir Shah, and Aamir Akbar

***Abstract*—The integration of software-defined networking (SDN) into legacy networks causes both operational and deployment issues. In this context, this article proposes a novel approach, called An Intelligent Solution for Improved Performance of Reliable and Time-sensitive Flows in hybrid SDN-based fog computing IoT systems (IHSF). The proposed IHSF approach has three solutions: 1) a novel algorithm to deploy SDN switches between legacy switches to improve network observability; 2) a *K*-nearest neighbor regression algorithm to predict in real time the reliability of legacy links at the SDN controller based on historic data; this enables the SDN controller to make timely decisions, improving system performance; and 3) a reliable and time-sensitive deep deterministic policy gradient algorithm (RT-DDPG), which optimally computes forwarding paths in hybrid SDN-F for time-critical traffic flows generated by IoT applications. The simulation results show that our proposed IHSF solution has a better performance than the existing approach in terms of network observability time, number of disturbed flows, end-to-end delay, and packet delivery ratio.**

***Index Terms*—Fog computing (FC), hybrid software-defined networking (SDN), IoT, link failure, machine learning.**

Manuscript received May 27, 2020; revised August 7, 2020; accepted September 2, 2020. Date of publication September 18, 2020; date of current version February 19, 2021. This work was supported in part by the "National Key Research and Development Plan" under Grant 2017YFC0821003-2, and in part by the "Dalian Science and Technology Innovation Fund" under Grant 2019J11CY004. The work of Gabriel-Miro Muntean was supported by the Science Foundation Ireland through Research under Grant 16/SP/3804 (Enable) and Grant 12/RC/2289_P2 (Insight). *(Corresponding authors: Lei Wang; Jenhui Chen.)*

Muhammad Ibrar is with the School of Software, Dalian University of Technology, Dalian 116024, China (e-mail: mibrar@mail.dlut.edu.cn).

Lei Wang is with the School of Software and the Key Laboratory of Ubiquitous Network and Service Software of Liaoning Province, Dalian University of Technology, Dalian 116024, China, and also with the Center of Underwater Robot, Peng Cheng Laboratory, Shenzhen 518066, China (e-mail: lei.wang@dlut.edu.cn).

Gabriel-Miro Muntean is with the School of Electronic Engineering, Dublin City University, Dublin 09, Ireland (e-mail: gabriel.muntean@dcu.ie).

Jenhui Chen is with the Department of Computer Science and Information Engineering, College of Engineering, and the Artificial Intelligence Research Center, Chang Gung University, Kweishan 33302, Taiwan, and also with the Center for Artificial Intelligence in Medicine, Chang Gung Memorial Hospital, Taoyuan City 333423, Taiwan (e-mail: jhchen@mail.cgu.edu.tw).

Nadir Shah is with the Department of Computer Science, COMSATS University Islamabad, Wah Campus, Taxila 4700, Pakistan (e-mail: nadirshah82@gmail.com).

Aamir Akbar is with the Department of Computer Science, Abdul Wali Khan University Mardan, Mardan 26110, Pakistan (e-mail: amirakbar@awkum.edu.pk).

Digital Object Identifier 10.1109/JIOT.2020.3024560

I. Introduction

INDUSTRY 4.0 and particularly Industrial Internet of Things (IIoT) rely on the interconnection of heterogeneous smart devices, such as actuators, machines, sensors, and controllers, in order to provide Internet connectivity for diverse industrial resources [1], [2]. These smart IoT devices have communication, storage, computing, and sensing capabilities, albeit limited, and produce pervasive sensing data about the industrial environment. For example, the remote intelligent monitoring system (RIMS) dynamically controls the industrial environment through these IoT devices in such a way that it reduces costs and/or increases the execution speed of the production process. IoT also connects the means of production and products via the standard Internet Protocol stack. In order to make the IoT devices virtually limitless in terms of computing, energy, and storage, integration of IoT devices with cloud was performed. Although cloud computing supports the execution of processor-intensive tasks, it does not provide support for reliable communications with low latency to the delay-sensitive applications of smart industries and smart cities. To address these issues, fog computing (FC) [3] is employed in the context of IoT.

In FC-based IoT, fog nodes are deployed near IoT devices and end users. A fog node can be any device (e.g., a router, a switch, or a wireless access point) that provides storage, communication, and computing services at the network edge to IoT devices. The closer proximity of fog nodes to IoT devices reduces end-to-end delay, minimizes bandwidth usage, and also decreases congestion in the backbone network.

In the context of the fifth-generation (5G) smart applications, managing and controlling ultradense distributed Internet-connected smart devices, and synchronizing their operations with a remotely-located cloud is a challenging task [4]. To overcome this problem and meet the QoS requirements of logically unified FC, a new model, called software-defined networking (SDN)-based FC (SDN-F), is proposed by combining the benefits of two emerging technologies: 1) SDN and 2) FC for IoT applications [4]. The core idea of the SDN is to decouple the control plane from the data plane by delegating the control plane from all network devices (i.e., router, switch, and APs) to a logically centralized controller [5]. In SDN, the data and control planes use the standard south-bound API, e.g., OpenFlow protocol, for communication.

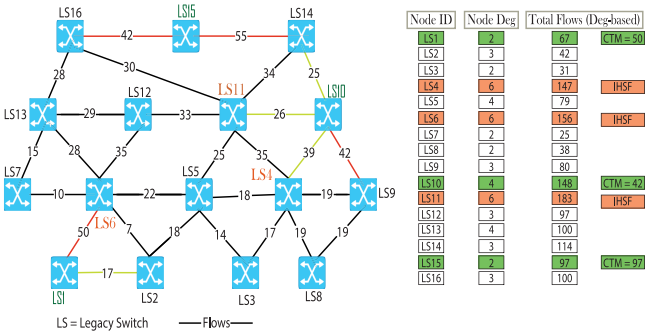

Fig. 1. Placement of SDN switches.

Despite its numerous advantages, the organizations are often reluctant to replace their legacy networks with SDN-based solutions due to several reasons, especially economic-related, as discussed in [6]. Instead, a new hybrid SDN architecture is proposed which deploys SDN devices (including switches) incrementally among the legacy devices in the IoT network, to improve network performance [7]. Although organizations like Google [8] have adopted such an incremental approach to deploying SDN in their data centers over several years, to date, this approach has not been considered in the context of IoT.

This article proposes an intelligent solution for improved performance of reliable and time-sensitive flows in hybrid SDN-based FC IoT systems (IHSF). IHSF solves several problems related to task offloading from IoT devices in a multihop hybrid SDN-F network context. To the best of authors' knowledge, this is the first proposal of a solution for reliability and time-aware data forwarding in hybrid SDN-F, including during critical events, such as link failures. The contributions of this article are as follows.

1) An innovative algorithm for deployment of SDN switches among legacy switches, which considers the combined values of switch degree (i.e., a number of links that are incident to the switch) and a number of flows passing through the switches. This algorithm enables the SDN controller to improve network observability and predict link reliability more accurately.
2) A K-nearest neighbor regression algorithm to predict in real time the reliability of legacy links at the SDN controller based on historic data. In hybrid SDN, legacy switches share their link-state information, including reliability, with the SDN switches using traditional routing protocols (e.g., OSPF). SDN switches pass on this information to the SDN controller, introducing a delay. This algorithm enables the SDN controller to have information about both SDN links (real) and legacy links (estimated) in real time.
3) A new reliable and time-sensitive deep deterministic policy gradient (RT-DDPG) algorithm which optimally computes a forwarding path in hybrid SDN-F for time-critical traffic flow generated by IoT applications is proposed.

The remaining article is organized as follows. We present the problem statement in Section II and our proposed solution in Section IV. Section V includes the performance evaluation of the proposed IHSF solution. The state-of-the-art is reviewed in Section III and conclusions are drawn in Section VI.

II. Problem Statement

This section focuses on SDN switch deployment and link failure management in hybrid SDN-F for IoT. The time-critical applications of IoT devices are adversely affected if flows are dropped or delayed due to link failures. In practical networks, typical failures determine links going down for almost 30 min, severely affecting performance, and this needs to be avoided [9]. Next, the problem is divided into two stages.

A. Network Observability

In a hybrid SDN, let *network observability time* refer to the time for the SDN controller to collect link statistics, including link failure notification. Noticeably, the network observability time (especially for legacy links) depends on the number and location of SDN switches. The researchers have proposed the compressive traffic monitoring (CTM) model [7], which assumes a fixed threshold value for the number of flows on important links (a subset of all the links). The SDN controller gets in real-time information about most flows by polling directly the SDN switches related to load information on the important links.

To clarify the problem statement, consider we have a network, such as the one in Fig. 1. If SDN switches are placed based on the CTM definition of the important links, then LS1, LS10, and LS15 are replaced, and the controller gets directly statistics about four links only (and no info on the legacy links is forwarded). These links carry 189 flows. If we consider all links as SDN links (connected to the SDN switches) in the CTM model, then the SDN controller gets directly statistics related to eight links that transfer 312 flows. Instead, an optimum placement (i.e., based on combined switch importance and a number of flows) of SDN switches (i.e., LS4, LS6, and LS11) results in the SDN controller getting directly information about 486 flows and statistics from 18 links, as shown in Fig. 1. This work focuses on the problem of placement of the SDN switches, such as the SDN controller receives information directly from the largest number of links and flows. In this way, the network observability time can be minimized.

B. Estimating Path Reliability of Legacy Links and Computing Reliable Path in Hybrid SDN

In a hybrid SDN, a legacy switch periodically broadcasts link status information in the form of link state packets (LSPs) to all switches, including the SDN switches. When the SDN switches receive LSP information from legacy switches, then they forward this information to the SDN controller. As information is not passed directly, in case of a legacy link failure [9]–[11], the SDN controller might not get the link down status in real time. Therefore, it is critical to be able to collect at the controller, the most up-to-date information about the legacy links and based on it to compute a reliable path for the time-critical applications. This work focuses on two additional problems. The first issue is predicting the current reliability of

legacy links from their historical reliability information at the controller. Based on this, the controller problem is to identify a reliable path for any flow based on current reliability values of SDN links, and current predicted reliability values of the legacy links.

III. Related Work

In this section, we classify the existing literature into three avenues, which are discussed in the following sections. Section III-A addresses the SDN-F architecture for IoT applications. The approaches related to hybrid SDN are explained in Section III-B. Section III-C describes the ML-based routing in SDN.

A. Approaches Related to SDN-F for IoT Systems

Currently, SDN is considered a viable solution for orchestrating and managing IoT applications, and FC [12]–[14] is an important avenue. Tomovic *et al.* [12] proposed the SDN-F architecture to improve the performance of IoT applications. The architecture consists of geo-distributed fog nodes and provides scalability and flexibility to IoT applications. Gupta *et al.* [13] highlighted the end-to-end QoS requirements in heterogeneous IoT infrastructure and proposed an SDN-F middle-ware architecture for the wireless domain to minimize end-to-end delay, load balancing, and traffic engineering [14]. Bera *et al.* [14] and Sood *et al.* [15] discussed the advantages of the integration of SDN and IoT architectures.

More recent works [4], [12], [16] have focused on offloading the traffic related to IoT devices in the context of an SDN-F architecture. Misra and Saha [4] proposed a greedy-heuristic scheme to minimize energy consumption and reduce the delay by considering dynamic networks (i.e., SDN rule capacity and link utilization). This is as ubiquitous computing is difficult to achieve using classic approaches and involves increased costs. Therefore, the proposed work considers a multihop topology. Khakimov *et al.* [16] employed an SDN-F architecture-based solution to minimize latency and enable efficient resource utilization. Tomovic *et al.* [12] proposed an SDN-F architecture for IoT that can support a high level of scalability, mobility, and real-time data delivery. Yu *et al.* [3] proposed a fog-based approach for IoT applications, which involves the selection of the most appropriate fog node and channel for the applications. The channel must satisfy both bandwidth and delay constraints. The authors identified two problems in terms of single-application provisioning (SAP) and multiapplication provisioning (MAP) and proved these problems as NP-hard. Chang *et al.* [17] described a queuing analysis algorithm for tasks offloading for both mobile devices to the fog server and fog server to the data center (cloud). Based on the matching theory, Chiti *et al.* [18] suggested a distributed algorithm used by a user to select the appropriate fog node/server based on computing time, transmission delay, and queue delay for the task offloading process. Shah-Mansouri and Wong [19] and Yousefpour *et al.* [20] proposed a hierarchical architecture and solution for the IoT-fog-cloud task offloading process, which minimizes the overall service delay and maximizes the Quality of Experience (QoE).

All the approaches described above have been proposed for pure SDNs (i.e., networks with all devices SDN-enabled). However, these approaches are not applicable to hybrid SDNs [6], [21]. Some approaches suitable for hybrid SDNs are described in the next section.

B. Approaches Applicable to Hybrid SDNs

In the hybrid SDN, it is a challenging task to deploy a given number of SDN switches among legacy switches [5]–[7], [22], [23] in such a way to increase the performance of the network. It is particularly challenging to improve the efficient usage of links' bandwidth and to provide more alternative paths for the flows in case of link failure. For example, Hong *et al.* [22] replace the legacy switches by the SDN switches based on the highest throughput value. To replace a legacy switch with an SDN switch, Poularakis *et al.* [23] decided based on two parameters: 1) maximum passing traffic through the switch and 2) dynamical control of the maximum paths. Agarwal *et al.* [5] used a heuristic algorithm for the deployment of SDN switches among legacy switches. Their selection criteria are based on four parameters: 1) switch with the highest degree; 2) link weights; 3) a node that frequently appears in the paths; and 4) switch with the highest traffic volume.

Cheng and Jia [7] used a CTM model to compute the important links in the network, and then place the minimum number of SDN switches in such a way that all the important links are SDN links. The important links are the subset of the links, through which a large number of flows are passing. The authors used a linear regression algorithm to estimate the current traffic load of legacy links based on both the historical traffic load of legacy links and the current traffic load of SDN links. To minimize the budget constraint, Xu *et al.* [24] proposed to place the SDN switches in such a way that the SDN controller would be able to control the flows with maximum traffic volume. However, the performance of the proposed approach is based on a given traffic matrix of the network. Levin *et al.* [25] placed the SDN switches in such a manner that every flow from a source to a destination should pass through at least one SDN switch, and a single SDN switch is sufficient to enforce traffic load balancing and end-to-end network policies. The proposed idea used a solitary confinement tree mechanism (based on a spanning tree). However, the flows in the proposed idea may be affected by increased latency because they may be diverted to longer paths.

In hybrid SDNs, efficient traffic engineering depends on having a global network view in real time at the controller. The global network view includes information about the status of all links, such as traffic load on links, reliability of the links, utilization of the links. Therefore, it is necessary to monitor the global link status in real time and accurately [9], [10], [25]. The link failure is an important problem in hybrid SDNs and needs to be solved. Existing literature [9], [10], [25] collects real-time links' information for the network and splits the flows on alternative paths. Jia *et al.* [10] proposed a model called *Hybrid-Hei* to achieve fast rerouting and load balancing in a hybrid hierarchical SDN in case single link failure occurs. This proposed solution deals with the link failure in

intra-autonomous systems (AS) and inter-AS. However, this leads to high overhead in terms of configuration and computation time. However, to the best of our knowledge, the existing literature overlooked the parameter of link reliability while placing the SDN switches. Moreover, existing works have not predicted the current reliability of legacy links at the controller from their historical reliability values.

After obtaining/predicting the reliability, end-to-end delay, traffic, and bandwidth utilization information of the data plane, the SDN controller in existing approaches computes the path for a flow based on these parameters. Since the nonlinearity of computing the path based on these parameters, it is an optimization problem and is hard to solve. In the quest to solve this problem, we discuss some existing related approaches for pure SDN in the following section.

C. ML-Based Routing in SDN

In modern heterogeneous network architectures, ML-based models can play an essential role in managing fundamentals problems in networking, such as fault management, traffic prediction, and network security [26]–[29]. For fast traffic classification, Li *et al.* [29] have used a multimachine learning approach to optimize the routing process. In this approach, to extract flow features (i.e., protocols or application type), clustering algorithms (Gaussian mixture model and K-means) is used. A supervised learning mechanism (i.e., Extreme Learning Machine) is then used to estimate traffic demand. Finally, an adaptive multipath routing method based on the investigative hierarchy process is proposed to deal with elephant flows according to the weights of different constraints factors. In their proposed model, the controller installs the learned classification results in switches to achieve fast identification. In contrast to our proposed IHSF solution, the authors did not forecast the path reliability in their proposed model.

Fast reactions to fault management is a critical task in traditional networks as compared to a centralized and controller-based network, such as SDN. Wang *et al.* [30] proposed a ML-based model to predict device failures in SDN-based optical networks. The authors combined double exponential smoothing (DES) and support vector machine (SVM) models to forecast proactively the risk of a device failure. In contrast, our proposed work considers a hybrid SDN in which the controller does not have updated information about the underlying network, particularly about the legacy devices. This, in turn, creates inconsistency in the controller. Therefore, it is a challenging task to predict the legacy links status in real time. Additionally, our model not only predicts the reliable path for time-sensitive for the FC IoT systems but also computes reliable paths based on IoT requirements like delay and bandwidth using the proposed RT-DDPG algorithm.

To optimize the SDN routing performance intelligently, there are several proposed machine-learning-based algorithms [31]–[33]. Parsaei *et al.* [31] proposed a new heuristic algorithm to compute the constraint shortest path (CSP) for telesurgery in the SDN. The authors used an ant colony optimization (ACO) algorithm to handle the CSP's NP-completeness and linear programming problem. To improve the quality of the received video in an operation room, the proposed model computes the optimum path based on delay and link utilization. In their work, the authors also considered a pure SDN architecture and did not take into account link failure scenarios. In such delay-sensitive applications, e.g., telesurgery, path's reliability level is a paramount parameter. Our proposed IHSF model provides a reliable path for time-sensitive applications in a hybrid SDN. Additionally, the proposed heuristic algorithm only works for a specific problem (telesurgery) and has scalability issues. For instance, when the network state changes, the parameters of the proposed algorithm are readjusted by executing the whole process again.

Jiang *et al.* [32] proposed intelligent end-to-end adaptive HTTP streaming in an SDN architecture. The proposed model based on both partially observable Markov decision and Q-learning optimized QoE. Deep reinforcement learning (DRL) is an alternative solution to optimize the routing process and handle dynamic network behavior. DRL can optimize the throughput, routing process, and achieves low latency. The proposed model's main purpose is to achieve fairness bandwidth allocation to all end users based on users' demands and reward function. However, the authors did not address the path reliability problem in the proposed model, and they consider pure SDN. The traditional Q-learning algorithms require massive storage space to store Q-table data. The data contains the following pieces of information: network states, action, and rewards. Consequently, the lookup time of Q-tables also increases with the data size. To handle the aforementioned Q-learning problems, in [33], the authors proposed a neural network and deep Q-networks (DQNs) to optimize the routing process in an SDN architecture. However, the basic algorithms for DRL like DQN only work under discrete-time control and are not suitable for continuous problems [34]. Thus, the underlying network needs a ML algorithm that can learn from experience rather than an accurate mathematical model. Additionally, the DRL-based DQN has a limited action space. Therefore, we proposed RT-DDPG which extends the DDPG algorithm [35] and works well for continuous problems [34].

In pure SDNs, the controller has status information about the network in real time, but in hybrid SDNs, the controller does not have updated status information about the network in real time [7]. Compared to the existing literature, e.g., [31], [32], and [33], our proposed ISHF model provides a reliable path for time-sensitive flows in a IHSF in a scenario where the links can fail due to any reason. The logically unified controller in a hybrid SDN has inconsistent status information about the network, particularly about the legacy switches and legacy links. To minimize the inconsistency problem in the controller, first, we deploy the SDN switches on the optimum location to minimize the network observability time. Second, the K-NNR module is used to predict the status of the legacy links. Third,we propose the RT-DDPG algorithm to compute a reliable path based on IoT traffic demands, such as bandwidth and delay.

Table I summarizes the existing literature on task offloading in FC IoT systems in terms of major aspects. A detailed analysis of the existing literature shows that there is a research gap for reliable and time-sensitive flows in IHSFs, and there

TABLE I
EXISTING LITERATURE: SUMMARY

Existing Literature	Hybrid SDN	SDN	Link Failure	Multi-hop	Delay	Path Reliability	ML	Bandwidth	FC IoT Systems
Sendra *et al.* [33]	×	✓	×	✓	✓	×	✓	✓	×
Parsaei *et al.* [31]	×	✓	×	✓	✓	×	✓	✓	×
Jiang *et al.* [32]	×	✓	×	✓	✓	×	✓	✓	×
Misra *et al.* [4]	×	✓	×	✓	✓	×	×	✓	✓
Yousefpour *et al.* [20]	×	×	×	×	✓	×	×	×	×
Cheng *et al.* [7]	✓	×	×	✓	✓	×	✓	✓	×
Yu *et al.* [3]	×	×	×	✓	✓	×	×	✓	✓
Zhilong *et al.* [30]	×	✓	✓	✓	×	✓	✓	×	×
Khakimov *et al.* [16]	×	✓	×	✓	✓	×	×	×	✓
Chang *et al.* [17]	×	✓	×	✓	✓	×	×	×	✓
Chiti *et al.* [18]	×	✓	×	✓	✓	×	×	×	✓
Chu *et al.* [9]	✓	×	✓	✓	✓	×	×	✓	×
Caria *et al.* [11]	✓	×	✓	✓	✓	×	×	✓	×
Xu *et al.* [24]	✓	×	×	✓	✓	×	×	✓	×
Proposed Solution *(IHSF)*	✓	×	✓	✓	✓	✓	✓	✓	✓

is a need to take into account link reliability, multihop paths, and dynamic network conditions such delay and bandwidth. Therefore, our proposed solution addresses these concerns and outperforms existing approaches, as demonstrated in Section IV.

IV. INTELLIGENT SOLUTION FOR IMPROVED PERFORMANCE OF RELIABLE AND TIME-SENSITIVE FLOWS IN HYBRID SDN-BASED FC IOT SYSTEMS

This section describes the intelligent solution for improved performance of reliable and time-sensitive flows in IHSFs. IHSF addresses both problems described in the previous section: placement of SDN switches in a hybrid SDN-based FC IoT context and path computation for reliable and time-sensitive flows.

Consider we have a legacy network represented by an undirected graph $G(V, E)$. In the graph G, V describes the set of switches (i.e., $V = \{v_1, v_2, \ldots, v_n\}$), and E represents the set of links connecting those switches [i.e., $e_i(v_i, v_j)$], such that v_i, $v_j \in V$, and $v_i \neq v_j$. IHSF has three stages, as follows.

A. Placement of SDN Switches

In a hybrid SDN-F, the placement of SDN switches is a nontrivial optimization problem. As already mentioned, our proposed algorithm selects legacy switches to be replaced by SDN switches according to both switch importance and their degree values.

For the selection process of important switches, we extend the concept introduced by the CTM model [7] to consider joint switch and flow importance. First, we compute the importance values for switches. A switch importance is assessed in terms of the number of flows passing through it. The importance values are stored in the set $\text{imp}_{\text{val}} = \{\text{imp}_1, \text{imp}_2, \ldots, \text{imp}_n\}$, such that imp_i represents the importance value of switch $v_i \in V$. Next, the degree values for all switches are computed. The degree of a switch is defined as the number of links that are incident to the switch. The degree distribution for the switches is represented by the set $\text{deg}_{\text{val}} = \{\text{deg}_1, \text{deg}_2, \ldots, \text{deg}_n\}$, where deg_i is the degree associated with switch $v_i \in V$. Then we compute the normalized importance values and degree values of the switches using a linear normalization method, as shown in

$$\text{norm}_{\text{imp}_i} = \left\lceil \frac{\text{imp}_i}{\max_i(\text{imp}_i)} \right\rceil \tag{1}$$

$$\text{norm}_{\text{deg}_i} = \left\lceil \frac{\text{deg}_i}{\max_i(\text{deg}_i)} \right\rceil \tag{2}$$

and represent the normalized important values of the switches through the set $\text{norm} - \text{imp}_{\text{val}} = \{\text{norm}_{\text{imp}_1}, \text{norm}_{\text{imp}_2}, \ldots, \text{norm}_{\text{imp}_n}\}$, where $\text{norm}_{\text{imp}_i}$ represents the normalized importance value of switch $v_i \in V$. Further, the normalized degree values of the switches are represented by the set $\text{norm} - \text{deg}_{\text{val}} = \{\text{norm}_{\text{deg}_1}, \text{norm}_{\text{deg}_2}, \ldots, \text{norm}_{\text{deg}_n}\}$, where $\text{norm}_{\text{deg}_i}$ is the degree associated with switch $v_i \in V$. The joint switch importance-degree is computed as follows:

$$J_i = \frac{w_1 * \text{norm}_{\text{deg}_i} + w_2 * \text{norm}_{\text{imp}_i}}{w_1 + w_2} \tag{3}$$

$w_1 = w_2$ is considered here for equal relevance of switch degree and importance. We rank the switches according to J_i values and based on the ranks, we select the subset of M number of important switches with highest ranks, $\psi \subset V$.

B. Network Configuration and Path Computation for Flow

After selecting the set of important switches M, we replace these legacy switches with SDN switches. The SDN switches communicate with the SDN controller via an out-band model [24]. When a controller receives a request for path computation for a flow from the data plane, the controller computes the path based on link bandwidth utilization B_u, delay d, and reliability of link $\Re_e$ in the network by using our proposed IHSF for hybrid SDN-F as discussed in Section IV-D. In IHSF, SDN controller gets the current traffic passing through SDN links directly, and computes the traffic of legacy links using an approach like the linear regression algorithm [7]. SDN controller gets the reliability of SDN links directly in real time, and the reliability of the legacy links is estimated as described in Section IV-C. To get the bandwidth utilization of B_u, we subtract the current traffic load from the link bandwidth (MB). In IHSF, we consider a multihop network scenario. Therefore, the cost function of delay d consists of flow installation delay (I), transmission delay (T), propagation delay (P), and queuing

delay (Q). Thus, the delay d of a path can be represented as $d_p = \sum_{h_x} (I_{h_x}^{f_n} + T_{h_x}^{f_n} + P_{h_x}^{f_n} + Q_{h_x}^{f_n})$, where h_x and f_n refer to the IoT device and fog node, respectively.

C. Prediction of Legacy Links' Reliability Levels

In a hybrid SDN-F, the SDN controller may not have the current reliability values of legacy links. Therefore, in the second stage of IHSF, we introduce an algorithm to compute an estimated reliability level of the legacy links from historical legacy links' status.

Consider a graph denoted by a quadruple, $G = (V, E, \delta, \Re)$, where V represents the set of n nodes, E is a set of e edges ($E \subseteq V \times V$), and δ are external conditions that influence the edge reliability. Formally, in-network, external condition δ usually has the following possibilities: data link layer the interface configuration problems; network protocol configuration, or communication congestion problems. We hereinafter refer to such external conditions as the edge's failure and downtime δ_e probability. Additionally, the reliability of edge $\Re_e : E_e \rightarrow (0, 1]$ is a function that assigns a reliability level to each edge in the specific time slot ($t_i \in T$), as shown in

$$\Re_e(\delta_e/t_i) = 1 - \sum_{t_i \in T} (1 - \Re_e(\delta_e/t_i)) \tag{4}$$

where $\Re_e(\delta_e/t_i)$ represents the reliability's level of the edge e that gives an edge is more or less reliable in the given time slot (t_i).

Lemma 1: In hybrid SDN, the SDN controller does not have up-to-date information about the legacy links reliability in a given time $T^- \subseteq T$.

Theorem 1: Based on Lemma 1, this problem creates an uncertain graph G^u produces 2^e deterministic graphs $G^u \subseteq G$ in the time T^-, (i.e., $G^u \subseteq G|T^-$).

Analysis: The reliability of all links (SDN link and legacy links) in graph G can be calculated as follows:

$$\Re_{e \in E}\left(\delta_{G^u}/T^-\right) = \underbrace{\sum_{e_s \in E_{G^u}} \Re_{e_s}\left(\delta_{e_s}/T^-\right)}_{\text{(a) SDN links}} + \underbrace{\sum_{e_l \in E \;\; E_{G^u}} \left(\Re_{e_l}\left(\delta_{e_l}/T^*\right)\right)}_{\text{(b) Legacy links}} \tag{5}$$

where $G^u = (V, \; E_{G^u})$, $E_{G^u} \in E$, and $T^* < T^-$. In (5), the SDN controller computes the reliability level of SDN links directly. Still, computing the reliability level of the legacy links is a critical task in time T^-. In the intended solution, we would like to predict the reliability level of the legacy links $\Re_{e_l}(\delta_{e_l}/T^*)$ as much as closer to the $\Re_{e_l}(\delta_{e_l}/T^-)$ in a given time T^-. Therefore, the K-NNR module predicts the reliability level of legacy links $\Re_{e_l}$ based on their historical failure statistics. Note that $\Re_{e_l} \cup \Re_{e_s} \subseteq E$.

The history of any e_i link is represented as 5-tuple like $e_i = (\text{Day}_{\text{time}}, \text{link}_{\text{downtimestart}}, \text{link}_{\text{uptime}}, \text{link}_{\text{downfrequency}}, \text{linkfailure}_{\text{reason}})$. In the proposed work, this 5-tuple is chosen as an independent variable to the K-NNR algorithm is to predict the reliability level of the link. Let $Z = \{(e_1, r_1), (e_2, r_2), \ldots, (e_n, r_n)\} \in (e, \; r)^n$ show that the data on n (5-tuple) independent variables, where the random pairs (e_i, r_i), e_i represent an instance in a D-dimensional feature space of the link like $e_i = [e_{i1}, e_{i2}, \ldots, e_{iD}]$ and r_i shows the dependent variable (reliability level) of each associated link e_i. The objective of K-NNR to predict the reliability level r for a new link-down event e_x using learn function $f : e_x \rightarrow r_x$. In K-NNR, the nearest neighbor's method assigns a reliability level to e_x as the closet neighbor/instance in Z, using Euclidean, Manhattan, or Makowski distance. In this work, we consider the Euclidean distance, $(r_i, e_x) = \sqrt{(r_i, e_x)^2}$, where $(r_i, \; e_x)$ shows the distance between the target reliability value of r_i.

The performance of the K-NNR algorithm strongly depends upon choosing the optimal value for the k. For example, there are k nearest neighbors to the e_x with reliability level set $r = (r_{nn1}, r_{nn2}, \ldots, r_{nnk})$, such that for any r_i is not in the nearest neighbor set. Then the distance$(r_i, e_x) \geq$ distance (r_{nnk}, e_x), this means that any other observation in the historical data is further away than the kth nearest neighbors set. Therefore, for $k > 1$, the K-NNR algorithm estimates the target reliability level $f(e_x)$ by averaging the target reliability level values of its kth nearest neighbors, $f(e_x) = 1/k \sum_{i=1}^{k} r_{nni}$.

Lemma 2: A reliable path is a path in which the minimum reliability's level of each link is maximum.

Theorem 2: Based on Lemma 2, here, we define a conditional reliable path indicator $\mathcal{R}_{\rangle\rangle}$.

Analysis: The $\mathcal{R}_{\rangle\rangle}$ is used for the selection of a most reliable path for task offloading process $\Re_p((h_x, f_n)|T^-)$ from IoT device [h_x. also known as (a.k.a.) IoT devices] to the fog node (f_n. a.k.a., fog node), in the given time set T^-, which can be computed by

$$\mathcal{R}_{\rangle\rangle} = \max\left(\min\left(\prod_{p_i \in P}^{P_n} \Re_{e \in p_i}\left(\delta_{G^u}/T^-\right)\right)\right) \tag{6}$$

$$\Re_p\left((h_x, f_n)|T^-\right) = \sum_{G^u \subseteq G \;|\; T^-} \left\{\mathcal{R}_{\rangle\rangle}(h_x, f_n)\left(\Re_e\left(\delta_{G^u}/T^-\right)\right)\right\} \tag{7}$$

where $\mathcal{R}_{\rangle\rangle}$ is a function used to compute the reliable paths. A reliable path is one in which the minimum reliability's level of each link is maximum, as shown in (6). A path $p_i(h_x, f_n)$, in which more links have a minimum reliability level, has not been taken into account unless and until if there is no other path available. Therefore, the most reliable path in the network can be as follows, as described in (7).

In Algorithm 1, the test data set, and remaining samples are used as a training data set. After this, the K-NNR function $K-\text{NNR}(.)$ predicts the accuracy of the legacy link's reliability level and returns the predicted value by summing up root mean squared error (RMSE) values for each k value. The $K-\text{NNR}(.)$ function calculates the distance between sample (X_{test}) and all other samples in the training data set (X_{train}) using Euclidean distance. Then, it selects the nearest k neighbors of X_{test}. Finally, it estimates the predicted value by the mean of the label values and returns the predicted value of the legacy link. The K-NNR algorithm then selects k with the lowest

Algorithm 1: K-NNR—Prediction of Links Reliability Level

input : E, Dataset (Z) with samples (m), feature space (d), k
output: return reliability level of $\Re_{e\in E}$
for k = *1 to n* **do** // $\mathcal{O}(n)$
  RMSE[k] ← 0 // Root Mean Squared Error
  K-NNR(X_{test}, X_{train}, k)
  // K-NNR algorithm, estimate the $\Re_{e\in E}$
  for y = *1 to n* **do** // $\mathcal{O}(n^2)$
    E-Dst[y] ← 0 // Euclidean Distance **for** z = *1 to m* **do** // $\mathcal{O}(n^2 + dk)$
      E-Dst[y] ← E-Dst[y] + $(X_{test_z}, X_{train_{yz}})^2$
    end
    E-Dst[y] ← *sqrt*(E-Dst[y])
  end
  // Sort Euclidean Distance List
  for s = *1 to k* **do** // $\mathcal{O}(n + c)$
    neareastSet[s] = E-Dst[z]
  end
  sum ← 0
  for $s \in$ *neareastSet* **do** // $\mathcal{O}(n + c)$
    sum ← *sum* + s
  end
  $pred_{rel}$ ← *sum*/k
  return $pred_{rel}$
end
for j = *1 to n* **do** // $\mathcal{O}(n)$
  X_{test} ← x_j
  X_{train} ← Z - $\{x_j\}$
  $pred_{rel}$ ← K-NNR(X_{test}, X_{train}, k)
  RMSE[k] ← RMSE[k] + $(pred_{rel_j} - rel_{val_j})^2$
end
// predict $\Re_{e\in E}$
k_{best} *argmin*$_k$ RMSE[k]

value of RMSE, as this is associated with the highest accuracy. Finally, the K-NNR algorithm forecasts the legacy link's reliability level according to the best nearest value. In this article, we used the leave-one-out cross-validation (LOOCV) approach [37], [38] for accurate prediction of the legacy links.

Complexity of Algorithm 1: K-NNR Algorithm 1 predicts the legacy link reliability level after a number of iterations. The algorithm consists of statements and loops to predict the reliability of legacy links at the SDN controller based on historical data. Suppose, we have m samples with D-dimensional feature space d of a link $e \in E$. Initially, the value of k is set to 0 for all observations in the X_{train} set and the Euclidean Distance *(E-Dst)* is computed from a new observation to X_{train}. Each *E-Dst* computation requires $\mathcal{O}(n)$ runtime, and due to all observations, the runtime of *E-Dst* is $\mathcal{O}(n^2)$. Now, for a one-sample, the runtime is $\mathcal{O}(n)$, and for all iterations, the required runtime is $\mathcal{O}(n^2+dk)$, where dk shows the memory constraints. The runtime of other loops is $\mathcal{O}(n)$. Therefore, the complexity of Algorithm 1 is $\mathcal{O}(n^2 + dk)$.

D. DRL-Based Framework for Hybrid SDN-F

The IHSF third stage involves the computation of the path for an IoT flow with given requirements in terms of bandwidth and delay. This is performed in SDN-F, with dynamic changes in both link reliability levels (including link failures) and available link bandwidth. To solve this problem, a novel RT-DDPG mechanism, which extends the DDPG algorithm introduced in [35], is proposed. The RT-DDPG-based path computation solution employs the following components: state, action, reward, and agent, described next.

1) State (S): In RT-DDPG, the state reflects the flow conditions of the hybrid SDN-F network. In hybrid SDN-F, for each flow, the agent performs the flow installation process in units of time slots signified as $t \in \{t_1, t_2, t_3, \ldots, T\}$. Suppose each time slot is set to t seconds and the total time for flow installation from IoT device to fog node (h_x, f_n), the agent takes time T. In ISHP, the state of the flow includes path reliability, delay, bandwidth utilization, and the number of disturbed flows (DF) in case of link's failure. Therefore, the agent obtains the corresponding reward by path reliability $\Re_p$, bandwidth utilization B_u, delay d, and the number of DFs Υ_f. Thus, in the IHSF, the state vector in one-time slot t might be explained, as shown in the following equation:

$$s(t) = \begin{bmatrix} \Re_{p1}(t), \Re_{p2}(t), \Re_{p3}(t), \ldots, \Re_{pn}(t) \\ B_{up1}(t), B_{up2}(t), B_{up3}(t), \ldots, B_{upn}(t) \\ \Upsilon_{f1}(t), \Upsilon_{f2}(t), \Upsilon_{f3}(t), \ldots, \Upsilon_{fn}(t) \\ d_1(t), d_1(t), d_1(t), \ldots, d_1(t) \end{bmatrix} \tag{8}$$

where for $\Re_p$ we use the K-NNR algorithm [based on (7)] and bandwidth utilization on path p [i.e., $\sum_{p\in\Re_{p1}:e\in p} \mathcal{L}(p) \leq C_e$] should be less than or equal to link capacity, where $\mathcal{L}(p)$ shows the bandwidth allocation on path p in time-slot t.

2) Action (A): The agent computes the optimal path for a flow based on traffic policy for the time-critical IoT applications in the hybrid SDN-F; therefore, the main objective of the agent is to map the states' space with actions' space. For the time-critical application, the action in IHSF includes three portions: 1) selection of the most reliable path $\Re_p$; 2) bandwidth allocation according to the IoT traffic demand; 3) and minimum delay. The agent decides which path is reliable for a flow, and the action is explained in the following equation:

$$a(t) = \left[a_{h1}^{\Re_p}(t), a_{h2}^{\Re_p}(t), a_{h3}^{\Re_p}(t), \ldots, a_{hn}^{\Re_p}(t) \right] \tag{9}$$

where $a_{h1}^{\Re_p}(t)$ shows a vector $a_{h1}^{\Re_p}(t) = \{a_{(h_x,f_n)}^{\Re_p}(t) \mid \Re_p \in \{\Re_{p1}, \ldots, \Re_{pn}\}\}$, and $a_{h1}^{\Re_p}(t)$ signify the reliability of path from $\Re_p(h_x, f_n) \in (0, 1]$. Additionally, the action should have met the bandwidth requirement of the flow.

3) Reward (R): In RT-DDPG, the agent gets the rewards from the feedback of the network G based on the current state $s_i \in S$ and current action $a_i \in A$. Subsequently, the agent continuously improves to compute a more accurate path for the flow by evaluating the effectiveness of rewards. This is because different actions infer different rewards. In the proposed hybrid SDN-F architecture, the path's reliability level and also other QoS requirements, including the minimum number of DFs, maximum bandwidth utilization, and minimum delay, and these parameters are the main reward factors. Therefore, the transmission of IoT flows at each time slot contains the following parts: 1) transmission of flow on a reliable path $\Re_p(h_x, f_n)$. 2) the path should fulfill these other QoS requirements. Thus, the reward $R(t)$ at time-slot t can be computed as follows, as

shown in the following equation:

$$R(t) = \frac{1}{Z}\sum_{z\in Z}\left(\beta R_{h_x}^{\Re_p}(t)\right) + \left(\alpha R_{h_x}^{QoS}(t)\right) \tag{10}$$

where Z shows the communication session occurring at time-slot t, and $|Z|$ signifies the total number of the communication sessions in a network, where β and α are tuning weight and determined by the specific situation and $\beta + \alpha = 1$. Additionally, $R_{h_x}^{\Re_p}(t)$ shows the reliability reward determined based on the K-NNR algorithm and also see the (6) and (7). Due to link failure, the $R_{h_x}^{\Re_p}(t)$ can be computed as follows:

$$R_{h_x}^{\Re_p}(t) = -\Upsilon_{h_x} - d_{h_x} \tag{11}$$

where Υ_{h_x} shows the number of DFs in case of a link failure, and d_{h_x} signifies the delay to compute a path for a flow and also show the delay to reroute the DFs.

Additionally, the Reward of QoS $R_{h_x}^{QoS}(t)$, as shown in (10), determined based on link constraints, like flow's drop ratio, bandwidth utilization, and delay at time–time can be signified by

$$R_{h_x}^{QoS}(t) = \sum_{\Re_p\in\Re_{p1},\ldots,\Re_{pn}} a_{h1}^{\Re_p}(t)\left(-d_{(h_x,f_n)} - \Upsilon_{(h_x,f_n)} - B_{u(h_x,f_n)}\right) \tag{12}$$

where $d_{(h_x,\ f_n)}$, $\Upsilon_{(h_x,\ f_n)}$, and $B_{u(h_x,\ f_n)}$ represent the delay, DFs ratio, and bandwidth utilization between a IoT and fog node (h_x, f_n), respectively.

4) Agent (AG): The agent is located at the level of the SDN controller and is the brain of the RT-DDPG. The agent collects the information of the underlying network G from the SDN controller and generates a corresponding routing path for a flow. The agent evaluates the network performance after the path is executed by forwarding the flow to determine the rewards obtained by the routing path and adjust parameters to achieve higher reward values. In the training session, the agent learns enough about the network based on historical interaction data/records. Therefore, the agent computes an approximately optimal path in the real hybrid SDN-F network for IoT applications.

Lemma 3: The RT-DDPG agent optimizes the path computation process based on new states ($s \in S$), actions ($a \in A$), and rewards ($r \in R$) functions.

Theorem 3 Based on Lemma 3, in hybrid SDN-F, the RT-DDPG computes the path based on $d_{(h_x,\ f_n)}$, $\Upsilon_{(h_x,\ f_n)}$, and $B_{u(h_x,\ f_n)}$.

Analysis: In RT-DDPG, the agent layer consists of an actor–critic model. The actor contains two networks, that is, the primary actor network $\Psi(s|\theta^{\Psi})$ and target actor network Ψ^*, as shown in Fig. 2. The target actor network Ψ^* and target critic network Q^* have the same components as primary actor network $\Psi(s|\theta^{\Psi})$. Additionally, the input and output of the Ψ^* and Q^* are same to the $\Psi(s|\theta^{\Psi})$ and $Q(s, a)$. However, the main difference between the primary actor network [i.e., $\Psi(s|\theta^{\Psi})$ and $Q(s, a)$] and target critic network (i.e., Ψ^* and Q^*) is that the input of target critic networks is the transformed state (TS), not the current state (s).

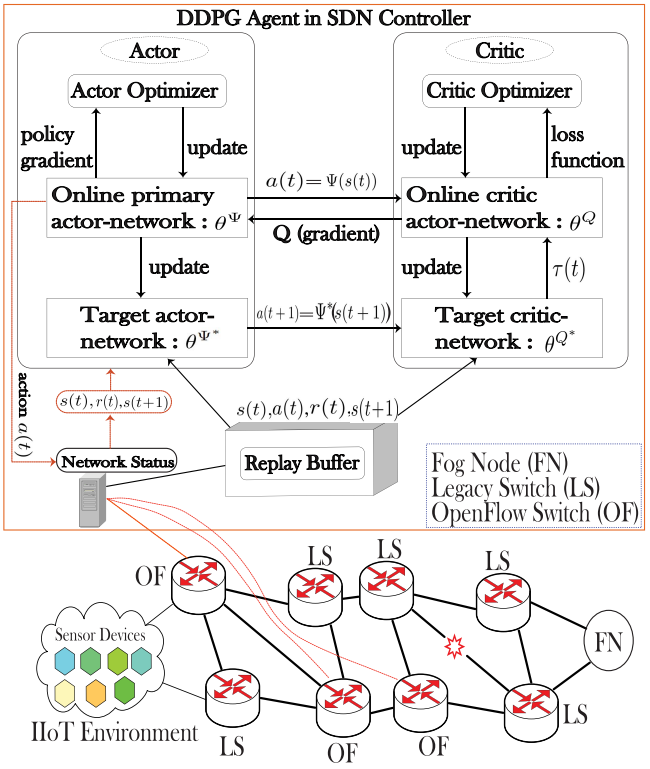

Fig. 2. RT-DDPG-based IHSF architecture.

The RT-DDPG model continuously gets the samples of state (s) by interacting with the network G at time-slot t. Thus, RT-DDPG uses replay buffer (mini-batches) to store samples and uses these samples to improve the training and learning process in each batch. In the RT-DDPG model, the primary critic network $Q(s, a)$ is trained to minimize the loss function, as shown in in the following equation:

$$\Pi(\theta) = \frac{1}{k}\sum_{t}\left(\tau(t) - Q\left(s(t),\ a(t)\mid\theta^{Q}\right)\right)^2 \tag{13}$$

where $\tau(t)$ is the target Q-value. In the above equation, deep Q-learning (DQL) is used to train the primary critic network and $Q(s(t), a(t) \mid \theta^{Q})$ is obtained by adding $a(t)$ into the primary critic network. By taking state $s(t)$ as the input, we can obtain $a(t)$, as shown in

$$Q(s(t), a(t)) \leftarrow \left[\begin{array}{c} Q(s(t), a(t) + \psi(r(s(t), a(t))) \\ + \max\limits_{a(t+1)} Q(s(t+1),\ a(t+1)) - Q(s(t), a(t)) \end{array}\right] \tag{14}$$

The target value $\tau(t)$ of Q-value can signify as follows:

$$\tau(t) = r(t) + \lambda Q^*\left(s(t+1), \Psi^*(s(t+1)\mid\theta^{\Psi^*})\mid\theta^{Q^*}\right). \tag{15}$$

By adding the current reward $r(t)$ and the $Q(s, t)$ value $Q^*(s(t+1), \Psi^*(s(t+1)|\theta^{\Psi^*})|\theta^{Q^*})$, we can obtain the value of target critic network Q^* in the next batch. The output action $\Psi^*(s(t+1)|\theta^{\Psi^*})$ is obtained at the next batch when the input state is s(t + 1), and here λ is considered as the discount factor. Using policy gradient (PG) process, the gradient of actor

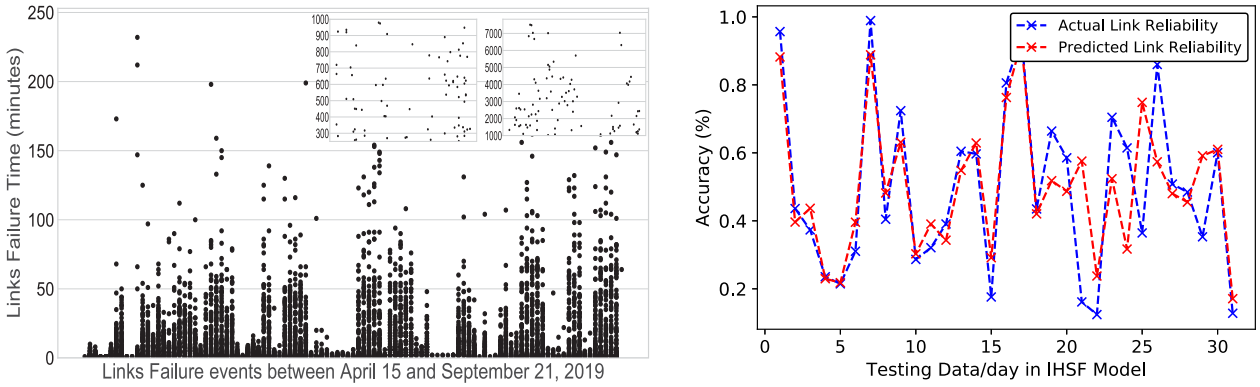

Fig. 3. Accuracy prediction of K-NNR in IHSF Model. (a) Real-time captured notification of link failure events over timescale (April 15, September 21, 2019). (b) Prediction accuracy per day.

network is shown as

$$\frac{\rho H(\theta^{\Psi})}{\rho(\theta^{\Psi})} = G_s\left[\frac{\rho Q(s,a|\theta^{Q})}{\rho a}\frac{\rho \Psi^{*}(s|\theta^{\Psi})}{\rho \theta^{\Psi}}\right]. \tag{16}$$

In the proposed hybrid SDN-F architecture, the agent, the actor network, updates the network's parameters, as shown in

$$\nabla_{\theta^{\Psi}} H \approx \frac{1}{K}\sum_{t}\begin{bmatrix}\nabla_a Q(s,a|\theta^{Q})|_{s=s(s(t)),a=\Psi(s(t))} \\ \nabla_{\theta^{\Psi}}\Psi(s|\ \theta^{\Psi})|_{s(t)}\end{bmatrix}. \tag{17}$$

The actor network updates depend on a reverse PG. More precisely, the primary actor can output multiple actions for the same $s(t)$.

In RT-DDPG, as mentioned earlier, the different actions $a(t)$ might be entered into the primary-critic network and can obtain different Q values. Based on the Q value, the probability of the corresponding action can be decreased or increased to get a more considerable Q value. Thus, the target network is updated by using

$$\theta^{Q^{*}} \leftarrow \mu\theta^{Q} + (1-\mu)\theta^{Q^{*}} \tag{18}$$

$$\theta^{\Psi^{*}} \leftarrow \mu\theta^{\Psi} + (1-\mu)\theta^{\Psi^{*}}. \tag{19}$$

V. Performance Evaluation and Results

We simulated a real network topology[1] using the Mininet emulator and the POX controller. The topology consists of 38 switches and 516 links. For results in Figs. 3(b)–5, for performance evaluation and experimental results of the proposed IHSF model and CTM it is compared against, we use the following parameters in the simulation setup. We generate 6 IoT applications, and each application has between 2 to 10 flows (randomly generated) from different IoT devices with random delay in the [10, 20] ms interval. Additionally, for each IoT flow, the bandwidth demand was also generated randomly from the [1, 20] Mb/s interval.

For the prediction of legacy links reliability level, we use a data set of link failure statistics of about six months period (April 15, September 21, 2019). We collected the network data and filtered the syslog file to reduce the number of transient events. The filtered syslog file consists of a smaller set of actionable events. This is as syslog messages, in particular, can be incorrect as network switches send several messages even though a link functions well. The syslog file consists of more than 750 000 events per hour, but the actionable events are 75%. The event log contains information about what type of network element experienced the event, type of event, small descriptive text, and ID. From the syslog file, primarily, we extract all "down" events for two types of failures, i.e., switches and links. These events are detected by SNMP monitoring on the interface state of switches. We use a real network data set to evaluate the performance of the IHSF model. Fig. 3(a) shows the distribution of link failures events that occurs over an almost six months period (April 15, September 21, 2019). After cleaning the syslog data, we use the proposed K-NNR algorithm [37] and predict the reliability level of legacy links and then compared them with the actual reliability level of the links.

[1]COMSATS University Islamabad (CUI), Attock Campus, Pakistan, https://attock.comsats.edu.pk/.

A. Accuracy Prediction of K-NNR in IHSF Model

We divided the link failure data set into two subsets: 1) 70% training set and 2) 30% testing set. We used the ML library named *Scikit-learn* for Python and we performed various simulations by varying the k value from 1 to 20 to examine the effect of the k parameter on the prediction accuracy, as explained in Algorithm 1. In IHSF, the data set size of our realistic network is not large enough; therefore, for accuracy predication of the legacy links reliability level, we used the LOOCV approach [37], [38].

From the result presented in Fig. 3(b), it is noted that the error rate increases with increasing the value of k. In the various simulations, the average values of RMSE (Root Mean Square Error) are $0.0095821382 \pm 0.0074189614$, and the average values of mean absolute error (MAE) are $0.0040949598 \pm 0.0023610452$. It is observed from the results that RMSE and MAE values increase with k when $k > 9$, and decrease when $k < 9$, and between 5 to 9, the K-NNR algorithm gives a smaller mean error. Therefore, based on analysis, we select the k value 7 for reliability prediction in the simulations, as shown in Fig. 3(b). The K-NNR algorithm predicts

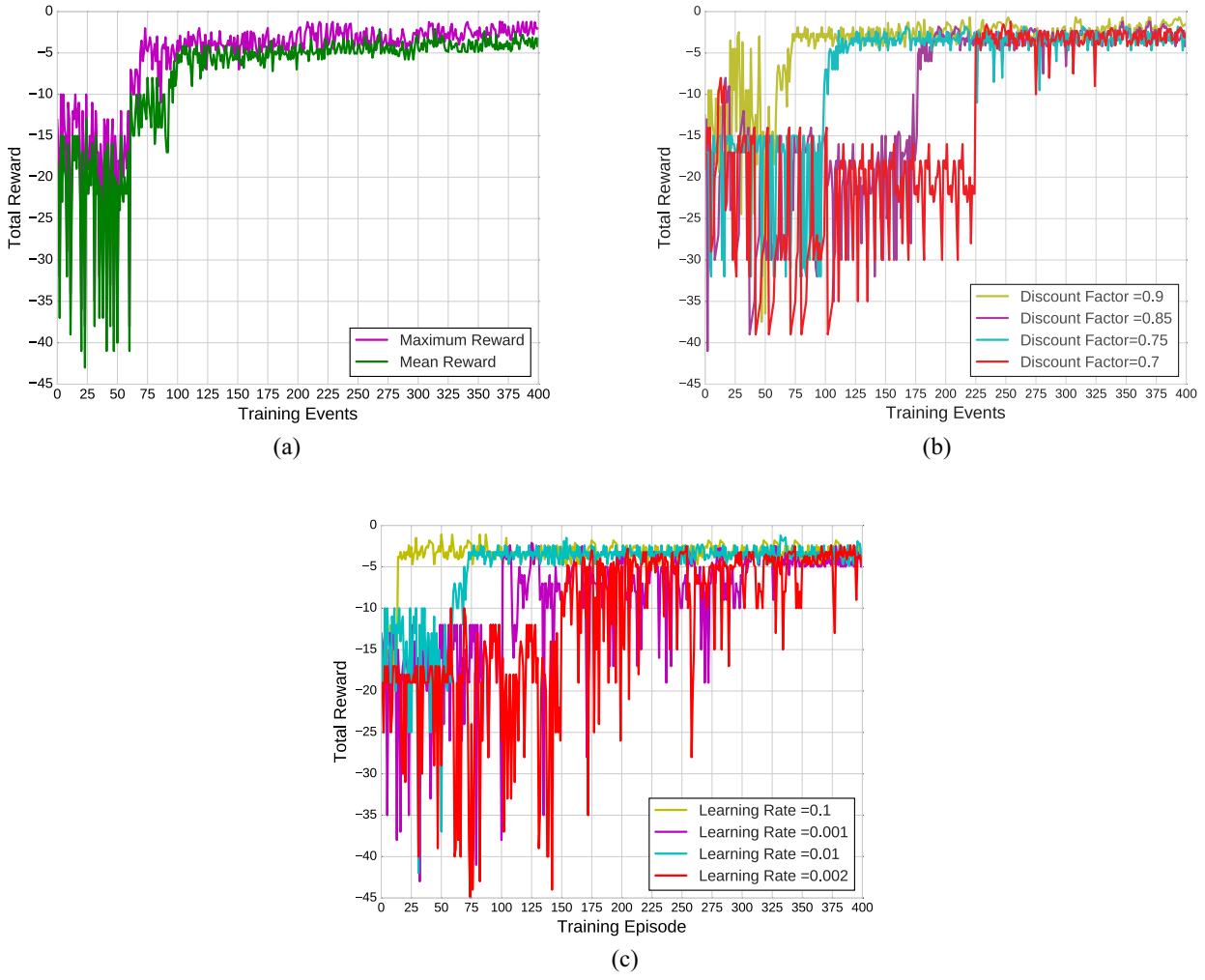

Fig. 4. Convergence of RT-DDPG in IHSF Model. (a) Rewards versus training event. (b) Reward versus discount factor λ. (c) Reward versus learning rate ψ.

very well the link reliability levels, and the prediction accuracy is 86.54%.

B. Convergence of RT-DDPG in IHSF

To optimize the performance of the RT-DDPG agent of the proposed IHSF, we train the agent by varying the numbers of flows. Additionally, in the proposed, we employed TensorFlow, an open-source platform for ML, in the back-end. For agent implementation, Python with Keras is used, which is a high-level neural network (HL-NN) API. On top of TensorFlow, this API runs efficiently. In the simulation (RT-DDPG), we use training events/episodes up to 400. For computing the value of the accumulated rewards, we set the maximum step to 15 in each event. Before explaining the results, next, the convergence process of RT-DDPG is discussed.

Results in Fig. 4(a) shows the effectiveness and convergence of the RT-DDPG-based routing in the proposed solution. Based on the defined parameters in the reward function [see (10)], Fig. 4(a) presents a learning curve of the RT-DDPG-based routing in the network, which shows the total and mean total reward values in several training events. The proposed routing method improves the reward value in each training episode. By having maximum rewards, it means that our proposed approach computes more often the optimum path based on model parameters, such as maximum reliable path, bandwidth utilization, minimum delay, and a number of DFs. Noticeably, the convergence ratio of the proposed RT-DDPG routing is better for time-critical applications.

To compute the optimal path for time-critical IoT applications in the proposed solution, we train the agent of RT-DDPG with different Learning Rate (ψ) and various discount factors (λ), as explained in (14). Fig. 4(b) shows the convergence level for different discount factors (λ). As shown in Fig. 4(b), the convergence level is better for the RT-DDPG-based routing when $\lambda = 0.9$. The learning rate (ψ) also has a significant impact on the convergence level, as shown in Fig. 4(c). From these results, for further simulation (i.e., for results in Fig. 5), we choose $\psi = 0.1$, and $\lambda = 0.9$. Additionally, we use the same reward value for both parameters, i.e., $(\beta + \alpha) = (0.5, 0.5)$.

C. Comparative IHSF and CTM Results

In this section, first we discuss the performance metrics that will be used to compare our proposed approach with CTM, an existing state-of-the-art approach. For time-critical applications, these parameters are essential for consideration in the hybrid SDN-F.

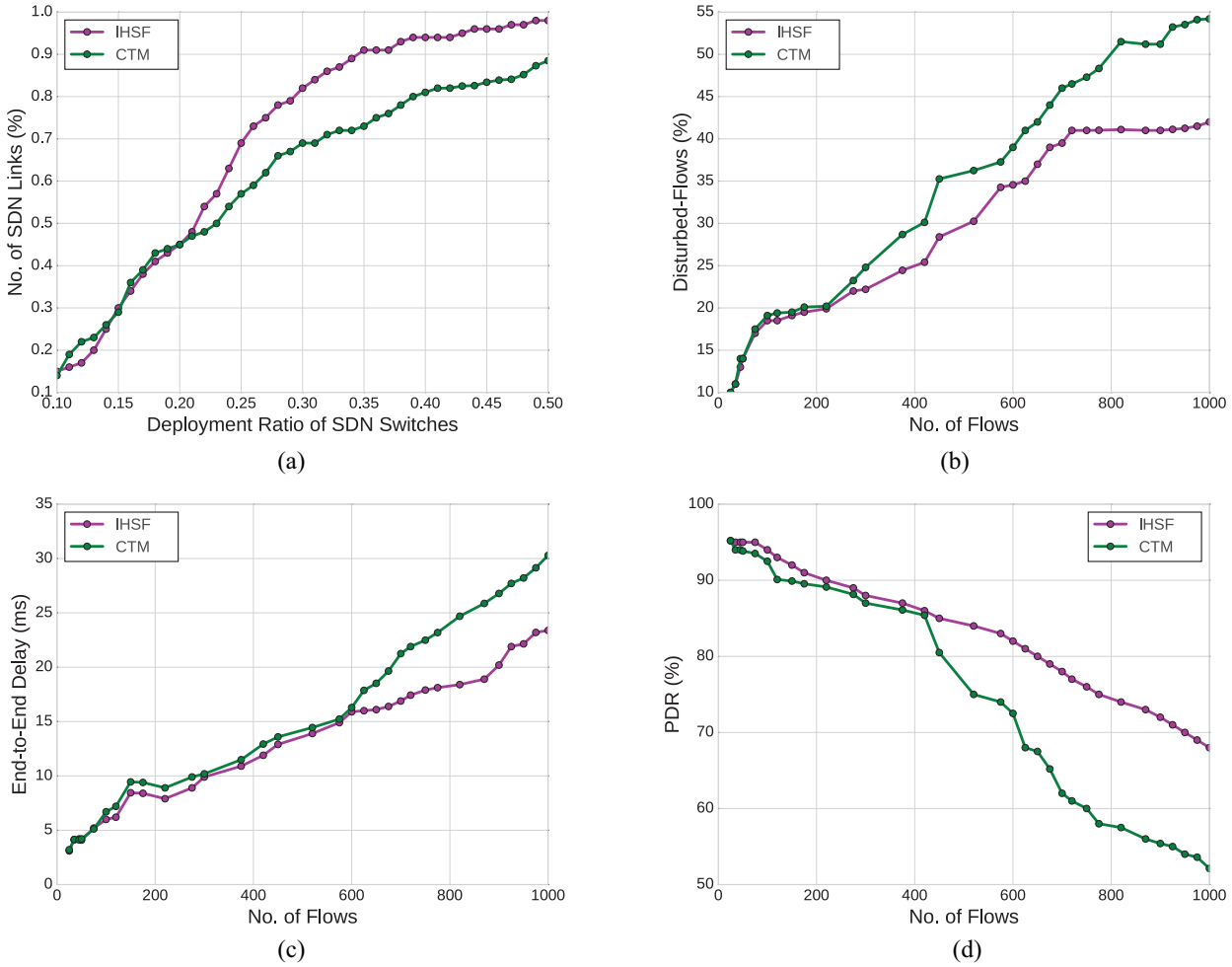

Fig. 5. Performance comparison between IHSF and CTM. (a) SDN switches ratio. (b) DF versus no of flows. (c) E2E-D (ms) versus no of flows. (d) PDR (%) versus no. of flows.

1) Number of SDN Links: Fig. 5(a) shows that our proposed approach has higher % of SDN links as compared to CTM. By having more % of SDN links in our proposed approach, it means that the SDN controller gets the link state information, including the reliability value of more links in real time, and has to predict the reliability of less number of legacy links. This improves the accuracy of a computed path in our approach. In the proposed approach, the controller obtains more links' information as the ratio of SDN switches is between 0.25 and 0.45. The controller gets almost 90% links' information at 0.35% SDN ratio. The main reason is that the proposed algorithm focus on the optimal value of important switch and maximum degree of a switch.

The main objective of our proposed model to upgrade a minimum of the legacy switch into important SDN switches can approximately predict the link's reliability level of legacy links. Therefore, for results, we only upgrade 0.25% switches to SDN switches, and remaining are legacy switches. Based on this observation, in the proposed model, the number of SDN links is 70%, and legacy links are 30%. In the CTM model, the number of SDN links is 58% and 42% legacy links.

2) Number of Disturbed Flows: The main objective of this article is to minimize the impact of a link failure on the time-critical and computational intensive flows. The proposed model uses the *K*-NNR module to predict the reliability of the legacy links and provides a reliable path for the flows. The ratio of DF increases in both the approaches as the number of flows increased. This is because of the number of flows per link increases. The figure also shows that the link failure event has a significant impact on the number of DFs. However, the number of DF is comparatively very less in the proposed model as compared to the CTM model, as shown in Fig. 5(b).

3) End-to-End Delay (E2E-D): E2E-D shows the time a flow takes from the IoT device to the fog node. In our simulation, the E2E-D includes flow installation process, transmission delay, queuing delay, and processing delay. Additionally, the same E2E-D parameters are considered for rerouting the DFs. From Fig. 5(c), one can note that the proposed approach significantly reduces the E2E-D as compared to the CTM due to following reasons. First, our proposed approach considers both the link reliability and available bandwidth of the links in computing the path for a flow. Second, our approach has less number of DF, as shown in Fig. 5(b). This is because our approach considers the link reliability in computing the path. This, in turn, reduces the occurrence of recomputing the path for the distributed flows. Third, our approach reduces

the traffic at the controller by reducing the number of DF. This allows the controller to compute the path for other flows. Fourth, our approach uses the RT-DDPG algorithm to compute the path, and its accuracy has increased, as shown in Fig. 3(b).

4) Packet Delivery Ratio: Packet delivery ratio (PDR) is the ratio of a total number of packets transmitted by the origin node (IoT) to the total number of packets successfully received at the destination nodes (fog nodes). The larger PDR shows the better performance of the proposed model. We analyze the PDR ratio of the proposed approach as compared to the CTM model. Fig. 5(d) shows that the PDR ratio decreases with the increase of the number of flows in the network. From the results, we can provide the analysis that the PDR ratio in the proposed model is almost 68%; however, in CTM, the PDR ratio decreased to almost 52%. We can also see that at 1000 flows, the PDR of the IHSF model is 16% greater than the CTM model. Thus, our novel proposed IHSF model for IoT applications performs outstandingly in terms of SDN switches selection, minimize the end-to-end delay, and minimize the DFs. Additionally, our novel proposed IHSF model is also applicable other than IoT traffic. In this article, throughout, we only focused on IoT applications because the IoT applications need more and more reliable communication services.

VI. Conclusion

This article proposed IHSF, a novel solution that improves the performance of hybrid SDN-based FC architecture for IoT applications. First, IHSF selects a subset of legacy switches and replaces them with SDN switches by considering both switch importance and number of flows in order to improve network observability. Second, IHSF predicts the current reliability of legacy links based on their previous reliability values at the SDN controller using an improved K-NNR algorithm, which achieves a prediction accuracy of 86.54%. Third, IHSF uses RT-DDPG, a novel DRL algorithm to find the optimal path for a flow based on maximum path reliability, minimum delay, and bandwidth utilization. Testing results show that the proposed solution outperforms a current state-of-the-art approach. Future research will focus on improving the prediction accuracy of legacy link reliability and testing in a more realistic IoT scenario, including both static and mobile IoT nodes.

References

[1] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, opportunities, and directions," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4724–4734, Jul. 2018.

[2] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.

[3] R. Yu, G. Xue, and X. Zhang, "Application provisioning in fog computing-enabled Internet-of-Things: A network perspective," in *Proc. IEEE INFOCOM*, 2018, pp. 783–791.

[4] S. Misra and N. Saha, "Detour: Dynamic task offloading in software-defined fog for IoT applications," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1159–1166, May 2019.

[5] S. Agarwal, M. Kodialam, and T. Lakshman, "Traffic engineering in software defined networks," in *Proc. IEEE INFOCOM*, 2013, pp. 2211–2219.

[6] L. Csikor *et al.*, "Transition to SDN is HARMLESS: Hybrid architecture for migrating legacy Ethernet switches to SDN," *IEEE/ACM Trans. Netw.*, vol. 28, no. 1, pp. 275–288, Jan. 2020.

[7] T. Y. Cheng and X. Jia, "Compressive traffic monitoring in hybrid SDN," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 12, pp. 2731–2743, Sep. 2018.

[8] S. Jain *et al.*, "B4: Experience with a globally-deployed software defined WAN," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, 2013.

[9] C.-Y. Chu, K. Xi, M. Luo, and H. J. Chao, "Congestion-aware single link failure recovery in hybrid SDN networks," in *Proc. IEEE INFOCOM*, 2015, pp. 1086–1094.

[10] X. Jia, Y. Jiang, and J. Zhu, "Link fault protection and traffic engineering in hybrid SDN networks," in *Proc. IEEE INFOCOM*, 2018, pp. 853–858.

[11] M. Caria and A. Jukan, "Link capacity planning for fault tolerant operation in hybrid SDN/OSPF networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2016, pp. 1–6.

[12] S. Tomovic, K. Yoshigoe, I. Maljevic, and I. Radusinovic, "Software-defined fog network architecture for IoT," *Wireless Pers. Commun.*, vol. 92, no. 1, pp. 181–196, 2017.

[13] H. Gupta, S. B. Nath, S. Chakraborty, and S. K. Ghosh, "SDFog: A software defined computing architecture for QoS aware service orchestration over edge devices," 2016. [Online]. Available: arXiv:1609.01190.

[14] S. Bera, S. Misra, and A. V. Vasilakos, "Software-defined networking for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1994–2008, Aug. 2017.

[15] K. Sood, S. Yu, and Y. Xiang, "Software-defined wireless networking opportunities and challenges for Internet-of-Things: A review," *IEEE Internet Things J.*, vol. 3, no. 4, pp. 453–463, Sep. 2015.

[16] A. Khakimov, A. A. Ateya, A. Muthanna, I. Gudkova, E. Markova, and A. Koucheryavy, "IoT-fog based system structure with SDN enabled," in *Proc. Int. Conf. Future Netw. Distrib. Syst.*, 2018, pp. 1–6.

[17] Z. Chang, Z. Zhou, T. Ristaniemi, and Z. Niu, "Energy efficient optimization for computation offloading in fog computing system," in *Proc. IEEE Global Commun. Conf.*, 2017, pp. 1–6.

[18] F. Chiti, R. Fantacci, and B. Picano, "A matching theory framework for tasks offloading in fog computing for IoT systems," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5089–5096, Sep. 2018.

[19] H. Shah-Mansouri and V. W. Wong, "Hierarchical fog-cloud computing for IoT systems: A computation offloading game," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3246–3257, May 2018.

[20] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing IoT service delay via fog offloading," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 998–1010, Apr. 2018.

[21] R. Amin, M. Reisslein, and N. Shah, "Hybrid SDN networks: A survey of existing approaches," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3259–3306, 4th Quart., 2018.

[22] D. K. Hong, Y. Ma, S. Banerjee, and Z. M. Mao, "Incremental deployment of SDN in hybrid enterprise and ISP networks," in *Proc. Symp. SDN Res.*, 2016, pp. 1–7.

[23] K. Poularakis, G. Iosifidis, G. Smaragdakis, and L. Tassiulas, "One step at a time: Optimizing SDN upgrades in ISP networks," in *Proc. IEEE INFOCOM*, 2017, pp. 1–9.

[24] H. Xu, X.-Y. Li, L. Huang, H. Deng, H. Huang, and H. Wang, "Incremental deployment and throughput maximization routing for a hybrid SDN," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1861–1875, Jun. 2017.

[25] D. Levin, M. Canini, S. Schmid, F. Schaffert, and A. Feldmann, "Panopticon: Reaping the benefits of incremental SDN deployment in enterprise networks," in *Proc. Annu. Tech. Conf. USENIX*, 2014, pp. 333–345.

[26] R. Boutaba *et al.*, "A comprehensive survey on machine learning for networking: Evolution, applications and research opportunities," *J. Internet Services Appl.*, vol. 9, no. 1, p. 16, 2018.

[27] J. Xie *et al.*, "A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 393–430, 3rd Quart., 2018.

[28] W. G. Hatcher and W. Yu, "A survey of deep learning: Platforms, applications and emerging research trends," *IEEE Access*, vol. 6, pp. 24411–24432, 2018.

[29] W. Li, G. Li, and X. Yu, "A fast traffic classification method based on SDN network," in *Proc. Int. Conf. Electron. Commun. Netw.*, 2015, pp. 223–229.

[30] Z. Wang *et al.*, "Failure prediction using machine learning and time series in optical network," *Opt. Exp.*, vol. 25, no. 16, pp. 18553–18565, 2017.

[31] M. R. Parsaei, R. Mohammadi, and R. Javidan, "A new adaptive traffic engineering method for telesurgery using ACO algorithm over software defined networks," *Eur. Res. Telemed.*, vol. 6, nos. 3–4, pp. 173–180, 2017.
[32] J. Jiang, L. Hu, P. Hao, R. Sun, J. Hu, and H. Li, "Q-FDBA: Improving QoE fairness for video streaming," *Multimedia Tools Appl.*, vol. 77, no. 9, pp. 10787–10806, 2018.
[33] S. Sendra, A. Rego, J. Lloret, J. M. Jimenez, and O. Romero, "Including artificial intelligence in a routing protocol using software defined networks," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, 2017, pp. 670–674.
[34] Z. Xu *et al.*, "Experience-driven networking: A deep reinforcement learning based approach," in *Proc. IEEE INFOCOM*, 2018, pp. 1871–1879.
[35] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[36] I. I. Awan, N. Shah, M. Imran, M. Shoaib, and N. Saeed, "An improved mechanism for flow rule installation in-band SDN," *J. Syst. Archit.*, vol. 96, pp. 1–19, Sep. 2019.
[37] J. Han, J. Pei, and M. Kamber, *Data Mining: Concepts and Techniques*. London, U.K.: Elsevier, 2011.
[38] S.-K. Lee, P. Kang, and S. Cho, "Probabilistic local reconstruction for K-NN regression and its application to virtual metrology in semiconductor manufacturing," *Neurocomputing*, vol. 131, pp. 427–439, May 2014.

Muhammad Ibrar received the B.S. degree in telecommunication and networking from COMSATS University Islamabad, Abbottabad Campus, Abbottabad, Pakistan, in 2010, and the M.S. degree in telecommunication and networking from Bahria University, Islamabad, Pakistan, in 2014. He is currently pursuing the Ph.D. degree with the School of Software, Dalian University of Technology, Dalian, China.

His research interests include software-defined networking, fog computing, wireless *ad hoc*, and sensor networks.

Lei Wang received the Ph.D. degree from Tianjin University, Tianjin, China, in 2002.

He is currently a Full Professor with the School of Software, Dalian University of Technology, Dalian, China. He serves as a Research Fellow with the Key Lab of Ubiquitous Network and Service Software of Liaoning Province and the Center of Underwater Robot, Peng Cheng Laboratory. He was a Member of Technical Staff with Bell Labs Research China, Shanghai, China, from 2001 to 2004, a Senior Researcher with Samsung, Seoul, South Korea, from 2004 to 2006, a Research Scientist with Seoul National University, Seoul, from 2006 to 2007, and a Research Associate with Washington State University, Vancouver, WA, USA, from 2007 to 2008. His research interests involve sensor networks, social networks, and network security.

Gabriel-Miro Muntean (Senior Member, IEEE) received the Ph.D. degree from Dublin City University Ireland, Dublin, Ireland, in 2004, awarded for a thesis on novel adaptive multimedia delivery solutions.

He is an Associate Professor with the School of Electronic Engineering, Dublin City University, where he is the Co-Director of the Performance Engineering Laboratory. He has published over 400 papers in top-level international journals and conferences, including four authored and six edited books. His research interests include quality, performance, and energy saving issues related to rich media delivery, technology-enhanced learning, and other data communications over heterogeneous networks.

Dr. Muntean is an Associate Editor of IEEE TRANSACTIONS ON BROADCASTING, the Multimedia Communications Area Editor of IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and Chair and reviewer for important international journals, conferences, and funding agencies.

Jenhui Chen (Senior Member, IEEE) received the B.S. and Ph.D. degrees in computer science and information engineering (CSIE) from Tamkang University, Taipei, Taiwan, in July 1998 and January 2003, respectively.

He has been joined the Department of CSIE, College of Engineering, Chang Gung University, Kweishan, Taiwan, since 2003, where he is currently a Full Professor and the Chairman of the Department of CSIE. He serves, meanwhile, the Deputy Director of the Artificial Intelligence (AI) Research Center, Chang Gung University. He is also a Professor of the Center for AI in Medicine, Chang Gung Memorial Hospital, Taoyuan City, Taiwan. His main research interests include design, analysis, and implementation of human-like intelligence, communication protocols, wireless networks, and data science.

Prof. Chen is currently a Senior Editor of Cogent Engineering. He served as the Technical Program Committee Member for IEEE Globecom, IEEE VTC, IEEE ICC, IEEE ICCC, IEEE ICCCN, IEEE 5G World Forum, and ACM CCIOT. He also served as a Reviewer for many famous academic journals which are organized by ACM, Elsevier, IEEE, and Springer.

Nadir Shah received the B.Sc. and M.Sc. degrees in computer science from Peshawar University, Peshawar, Pakistan, in 2002 and 2005, respectively, the M.S. degree in computer science from International Islamic University, Islamabad, Pakistan, in 2007, and the Ph.D. degree from Sino-German Joint Software Institute, Beihang University, Beijing, China, in 2011.

He is currently an Associate Professor with the COMSATS University Islamabad (Wah Campus), Taxila, Pakistan. He has authored several research papers in international journals/conferences, such as *ACM Computing Surveys* and IEEE COMMUNICATION LETTERS. His current research interests include computer networks, distributed systems, and network security.

He is serving on the Editorial Board for *International Journal of Computer Systems* (Wiley), IEEE SOFTWARIZATIONS, *Ad Hoc and Sensor Wireless Networks*, and *Malaysian Journal of Computer Science*. He has been serving as a reviewer for several journals/conferences, including ICC, INFOCOM, WCNC, *Computer Networks* (Elsevier), IEEE COMMUNICATIONS LETTERS, *IEEE Communication Magazine*, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, and *Computer Journal*.

Aamir Akbar received the M.Sc. degree from Oxford Brookes University, Oxford, U.K., in 2012, and the Ph.D. degree from Aston University, Birmingham, U.K., in 2019.

He was a Lecturer with COMSATS University, Islamabad, Pakistan. He is currently a Lecturer of Computer Science with Abdul Wali Khan University Mardan, Mardan, Pakistan. His research is on mobile cloud computing, including but not limited to, edge/fog computing, IoT, and SDN. His work leverages multiobjective optimization, evolutionary computation, machine learning, self-adaptivity, and self-awareness to tackle problems.