FullSight: a Deep Learning based Collaborated Failure Detection Framework of Service Function Chain

Kuo Guo, Jia Chen*, Ping Dong, Yu Zhao, Deyun Gao, Shang Liu
Beijing Jiaotong University
Beijing, 100044, China
Email: {19111017, chenjia, pdong, 20125135, gaody}@bjtu.edu.cn

***Abstract*—Network Function Virtualization (NFV) is one of the most promising technologies which decouples Network Functions (NFs) from hardware resources to support more flexible network services and network resource allocation. However, these benefits increase the possibility of Service Function Chain (SFC) failures due to hardware failures, software defects and burst traffic, resulting in serious consequences. Unfortunately, existing failure detection methods have several issues, such as simplification of detection functionality, heavy overhead, and low accuracy. This paper introduces a framework FullSight, in which control plane and the programmable data plane can collaboratively detect failure and Deep Learning (DL) based algorithms are adopted for failure detection. FullSight can achieve an all-round perception of the state of the SFC, in which network information is acquired through the data plane, SFC components' message is obtained through the control plane. In addition, a failure detection model based on DL is established. Compared with the state-of-the-art methods, FullSight can support 8 kinds of the fine-grained failure detection. Our comprehensive evaluation of prototypes and simulations shows that FullSight can realize rapid and accurate detection and classification of diversified failures in SFCs. The bandwidth overhead reduces by 57% compared with the existing methods. Additionally, FullSight has a detection accuracy up to 93.5%.**

Keywords*—*Network function virtualization, service function chain, collaborated detection, programmable data plane, deep learning

I. Introduction

Software Defined Networking (SDN) realizes the decoupling of control plane and data plane, and greatly increases the flexibility of the network [1]. NFV enables NFs to be decoupled from dedicated commodity hardware, greatly reducing the cost of telecom operators [2], [3]. SFC is a series of NFs (i.e., Firewall (FW), Deep Packet Inspection (DPI), Intrusion Detection System (IDS), Network Address Translator (NAT), etc.) connected in a particular sequence. SFC based on SDN and NFV has increased the agility and dynamics of service orchestration, effectively reduces costs of Operation Administration and Maintenance (OAM), and has become a powerful network technology in recent years.

However, the agile and dynamic orchestration also enhances the complexity of SFC deployment, and leads to boost the probabilities of SFC failures. After accomplishing SFC deployment, it is necessary for us to continuously monitor the status of the SFC to ensure the quality of experience (QoE) and quality of service (QoS) to satisfy the needs of users. Furthermore, SFC failures due to hardware failures, software defects, etc. may occur at any time.

Some existing failure detection approaches such as the traditional tool *hping3* or probe packets can only discover one failure or a class of failure [4]. Distinguishing the kind of failure is difficult. Even though the failure is discovered, the accuracy and efficiency of the troubleshooting are extremely low. Fig. 1 illustrates the above the case.

① The blue line represents the route when the probe detects the persistent black holes. From the judgment of probe manager, we consider that Service Function Forwarder 2 (SFF2) occurs failure.

② VNF1 (Virtualized Network Function1) breaks down when the probe transmission process. However, we firmly believe that SFF2 is faulty while the actual failures are VNF1 breakdown and persistent black holes, which leads our judgment to be inaccurate.

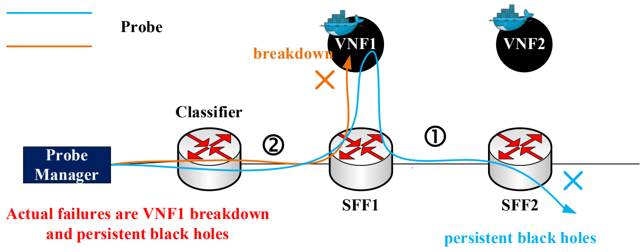

Fig. 1. Some cases of judging failure incorrectness

The main challenges of SFC real-time failure detection are as follows:

1) SFC failures consist of network failures (for instance, congestion, and persistent black holes) and components' failures (links, NFs, Nodes, SFFs, Classifiers (CFs), etc.,), which involve multiple dimension statistical data [4], [5]. In order to monitor failure in real time, we need to gather the status of the network and all SFC components. The scale of data for failure detection is relatively huge.

2) In SDN, The failure propagates iteratively between the switches based on their dependence relations [6]. Similarly, in SFC, this dependency also exists which affects failure judgment. For example, a link failure can also cause NFs' failure, according to the propagation relation. However, two failures of both link and NFs exist actually, resulting in incorrect judging. Besides, Fig. 1 shows that the same failure characteristics do not imply the same failure, which cannot be visually analyzed from the superficial consequence [7], [8].

3) The total time of SFC failure detection contains link latency (include the latency of the physical and the virtual link), switches processing latency. Aiming at fast failure detection is of great significance that reduces the total time of SFC failure detection [9].

4) The accuracy of failure detection is a key element for troubleshooting [10]. Enhancing the accuracy and efficiency of failure detection is a significant target of failure detection.

To address the above challenges, we put forward FullSight, a collaborated and DL based failure detection framework of SFC, in which data plane can collaboratively work with control plane for failure detection. FullSight can achieve fast and accurate failure detection and reduce the probe overhead. Furthermore, FullSight can classify failure accurately into 8 categories with the aid of different DL algorithms. In general, the main contributions of this paper are as follows:

- We develop a data plane network scheme in FullSight based on container network and Programming Protocol-Independent Packet Processors (P4)-based SDN network. Our network scheme not only achieves the separation of data traffic and control traffic, but also makes the data plane forward packets more flexible and efficient.
- We propose a mechanism to realize collaborated detection of SFC failures with data plane and control plane. The data plane can make up for the deficiencies of the control plane monitoring SFC. The mechanism proposed this paper not only considers severe failures, but also "invisible" degradations. In addition, the mechanism contributes to comprehensively discovering the failures in SFC.
- DL based detection algorithms are proposed to gather and analyze high-dimensional status information of the underlying physical network, virtual network, VNFs, and Nodes. These algorithms can accurately discover the propagation relationship between failures, and manifest high accuracy for failure detection and classification.
- We have verified the performance of FullSight in a prototype system. Compared with the state-of-the-art methods, FullSight decreases the probe overhead by 57%, and supports 8 kinds of the fine-grained failure detection. Additionally, the detection accuracy is up to 93.5%.

The rest of this paper is listed as follows. Section II introduces the related work in failure detection. Section III describes the architecture design and failure modeling. We introduce FullSight in details in Section IV. The performance evaluation of prototypes is shown in Section V. Section IV gives a summary and future work.

II. Related Work

Researching on SFC failure detection have been mostly isolated so far. In addition, there is no relevant research on failure classification. Failure detection methods are mainly divided into two types, active and passive detection methods [11-13]. However, neither of them can conduct comprehensive detection and classification of SFC failures.

The active detection method is designed to construct some special probe packets to detect the failures of SFC. RFC 8924 [5] proposes SFC OAM, a network measurement way for SFC, which provides a simple scheme for detecting failures. The scheme takes up a lot of bandwidth resources and has a low bandwidth utilization. When network congestion is heavy, OAM will aggravate the load of the network, and cannot detect multiple failures on one service chain at the same time. [14] proposes a SFC failure detection tool, SFC path tracer, applied in SDN/NFV network scenarios. SFC path tracer constructs probes by using the *hping3* tool to detect CFs, SFFs, VNFs, and locates failure by reporting to the controller every hop. However, this method has the same flaws as mentioned in RFC 8924. [4] introduces a failure detection method FDM, which reduces the overhead of active detection. However, the method needs to calculate the position where probes are placed in advance. When the topology changes, FDM needs to recalculate the position, which is time-consuming.

Some methods are based on the programmable data plane using in-band network telemetry (INT) for failure discovery. [15] uses INT to detect "gray" failure in data center networks (DCNs), such as silent packet loss and persistent black holes. By placing active probes, the probes collector is used to store the feasible paths in the path information table, and the corresponding aging time is set to detect the occurrence of failures. This method has many shortcomings such as small detection range, poor real-time performance, and low bandwidth utilization. [16] proposes a method in-band network function telemetry (INFT) to effectively monitor the performance of NFs. INFT collects network status through the data plane, generates less overhead (without control plane intervention), and captures transient changes in the entire network. The author adds support for the report, similar to the postcard in the framework, so that a single data packet can be tracked effectively. However, this approach lacks efficiency in the SFC failure detection, and needs to report to the controller every hop, and multiple failure points cannot be found at the same time.

Most passive methods are triggered by SFC abnormal events. [7] uses the error correlation method for the virtual Service Function Chain (vSFC) based on NFV scenarios to detect performance anomalies. This method collects the indicators of multiple elements of the vSFC and analyzes the correlation over a period of time to infer the health of NF. For example, set the correlation coefficient of the CPU utilization of the upstream and downstream VNF to determine whether the upstream VNF is failure or abnormal. However, this approach lacks universality and cannot fully detect SFC failures. [17] judges the reason of the network function failure, according to the queue depth of NFs and the latency of the packet. For instance, the CPU interruption of the upstream NFs will decrease the throughput of the downstream NFs. This way can only judge which the NF fails, and cannot detect the failures of the entire SFC chain. [18] introduces a technique based on extended Berkeley Packet Filter (eBPF) to monitor container network performance in a microservice scenario, and monitors container network events in real time by implementing the dynamic deployment of container network measurement sensors. However, this approach need deploy a network measurement sensor on the network interfaces of each container, which increases the monitoring cost and cannot simultaneously discover multiple failure points on the same path.

In summary, the state-of-the-art technologies cannot fully detect SFC failures. Therefore, we propose a scheme FullSight, which can realize comprehensive, fast and intelligent failure detection and classification of SFC. FullSight can detect and classify failures at the same time, which provides theoretical and practical basis for SFC migration or failure recovery.

III. Architecture Design and Failures Modelling

In this section, we introduce the architecture design of FullSight and diverse failures modelling. FullSight can capture most types of failures and identify severe packet loss, while keeping a low overhead.

A. *FullSight Architecture*

The FullSight architecture based on the SDN/NFV is shown in Fig. 2. FullSight is made up of three planes: data plane, control plane, and knowledge plane.

- **The data plane** implements the function of SFC based on INT and forwards the packets. In addition, we design the agent for obtaining the network information, which achieves fine-grained collection of network status.
- **The control plane** adopts Kubernetes (K8s) and ONOS controller. We develop the agent for the perceiving information of SFC components. The agent can quickly discover possible failures (e.g. Nodes, VNFs, Physical Links, Virtual Links), which greatly reduces the pressure of data plane failure detection with a low monitoring overhead.
- The function of **the knowledge plane** is mainly to preprocess the collected data, analyze it with DL algorithms such as Long Short-Term Memory (LSTM), Bi-directional Long Short-Term Memory (BiLSTM), Convolutional Neural Networks (CNN), and implements SFC failure detection and classification.

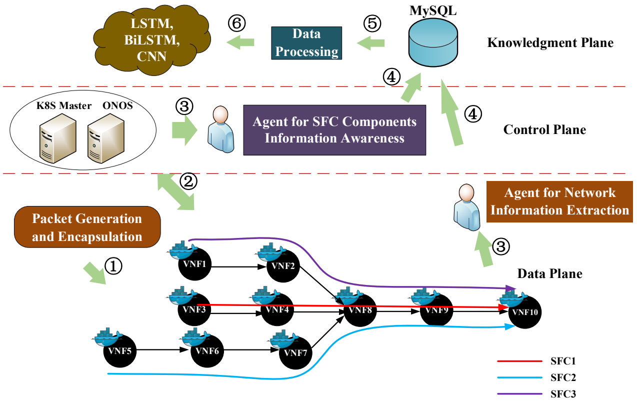

Fig. 2. Architecture of FullSight

The following is the detecting failure workflow of FullSight when the failures occur on SFC. FullSight can make full use of the agents for the control and data plane to achieve collaborated failure detection.

① The user packets are generated and encapsulated with NSH and INT header. Each time through every switch on the chains, FullSight encapsulates the network information in the INT data fields.

② The controllers including K8s master and ONOS obtain the information of the network and SFC components in real time.

③ Two agents gather the information from the control plane and data plane respectively.

④ The agents quickly import data into the MySQL database.

⑤ The data processing module handles the high-dimensional data collected by the agents, and takes shape a dataset afterwards.

⑥ LSTM, BiLSTM, CNN are the algorithms storing in the knowledge base, which are used to analyze the dataset to acquire the results of failure detection and classification.

B. *Network Modelling*

In order to detect all failures in SFC, we build a mathematical model of the network. SFC status includes SFC components and network status, which are represented by S_C and S_N respectively. The states of the SFC components include physical resources and virtual resources are defined by S_P and S_V, respectively. The physical resources status are abstracted into $S_P =< S_N^P, S_L^P >$, where S_N^P and S_L^P respectively denote the status of physical nodes and physical links. The physical nodes can be represented by $S_N^P =< U_C^P, U_S^P >$, where U_C^P and U_S^P respectively mean computer resource and storage resource. VNFs resource is denoted by $S_{NF}^V =< U_C^{NF}, U_S^{NF} >$, where U_C^{NF}, U_S^{NF} represents respectively the VNFs resource of computer and storage. Network status S_N includes: *Switch ID, Queue Depth, Port ID, Throughput, Link Delay, Processing Delay*, abstracted as $S_N =< ID_{SW}, Q, ID_{Port}, D_{Link}, D_P >$.

C. *Failures Modelling*

Some failures include Links, SFFs, SFFs, VNFs, etc., which can be measured directly or indirectly based on their status through the agent of the control plane. However, the failures caused by congestion and persistent black holes need to obtain their state set through the agent for the data plane, and use the following ways to establish a failure model.

1) Congestion Failure

When the transmission latency of the packet is greater than the latency threshold, we define this situation as the congestion failure. Let t_D be the transmission latency, and t_T be the latency threshold. Congestion failure is denoted by $t_D > t_T$. The causes of congestion failures include excessive link utilization, CPU, Memory and other related indicators. When any probe packet is transmitted in the SFC, the difference between the ingress timestamps of entering the switch and egress timestamps of leaving the switch is greater than the threshold, which is defined by $t_{TI}^i - t_{TE}^i > t_T$, $\forall i \subseteq N$, N is the total number of probe packets.

2) Persistent Black holes Failure

The persistent black holes show that all the transmitted packets are lost. Therefore, the existence of failures cannot be detected only from the control plane. FullSight can make up for the defects of the control plane detection. Fig. 3 shows the packet path when persistent black holes occur. We use C_{ij}^w to indicate the network and VNFs status collected from the control plane, where j represents the type of collected status, including physical\virtual links, VNFs. w defines the status including normal and failure. Let r and d be normal and failure respectively. If a switch occurs the persistent black holes, the SFC components state obtained through the control plane is normal, and the upstream network state obtained through INT is also normally. However, the downstream state has no data, which is presented by *None*. Table 1 shows that the persistent black holes occur in the mth switch.

TABLE I. DATA STATISTICS OF A CERTAIN CHAIN OCCURING PERSISTENT BLACK HOLES

C_i^{jw}	C_1^{jr}	C_2^{jr}	C_m^{jr}	C_{m+1}^{jr}	C_N^{jr}
D_i^{jw}	D_1^{jr}	D_2^{jr}	D_m^{jr}	*None*	*None*

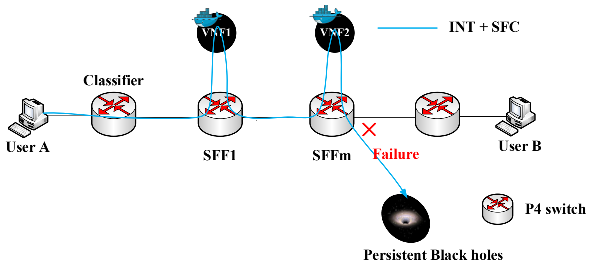

Fig. 3. Failure for the persistent black holes

IV. IMPLEMENTATION

In this section, the design details for implementing FullSight are presented. To address the packet forwarding, we design the network based on container and P4 and propose the header format of the packet. In addition, we introduce in details the process of the collaborated failure detection and describe the three DL algorithms.

A. Container Network and Underlying Physical Network

For the data plane, we put forward a scheme for P4+Macvlan network, as shown in Fig. 4. Macvlan is a network plugin of K8s, which can virtualize multiple network interfaces for one network interface of the container. Compared with Bridge, Macvlan is a more efficient container network solution. The main interface in the Macvlan network uses virtual network interface cards (VNICs).

We use VNICs to connect the switches and the containers. Switches are connected by network interface cards (NICs). NIC1 and NIC2 represent control interface and data interface respectively. By this means, we can separate the traffic of control plane and data plane. Besides, based on CNI-Genie Pods can be assigned multiple network interfaces. In our experiments, containers are all in the same subnet.

B. Protocol Design for INT and SFC

The header format of SFC based on INT is shown in Fig. 6. We adopt the Network Service Header (NSH) to encapsulate the SFC packet. In order to distinguish SFC traffic and the other traffic, we use the first 6 bits of the DSCP field in the IPv4 header to mark SFC packets. The INT header is encapsulated in the UDP/TCP Payload. The last 2 bits of the DSCP field are used to mark the INT packet. Compared with a *PING* packet at least containing 64 Bytes, telemetry information occupies a total of 28 Bytes which includes INT header and INT data.

Fig. 7 demonstrates P4 source program based INT and SFC handles the workflow of the packets. Classifier implements NSH and INT header encapsulation. SFFs push the INT data stack and decapsulate the NSH. The INT sink (SFF3) decapsulates the NSH, INT header, and INT data based on source and destination IP.

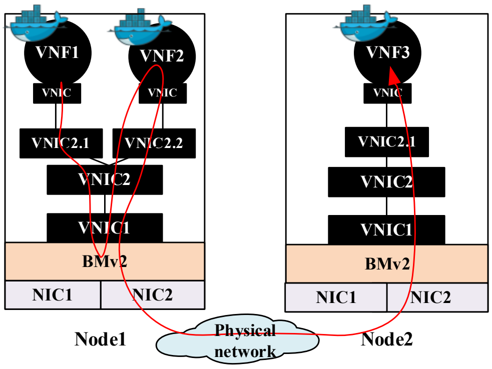

Fig. 4. The scheme of FullSight network

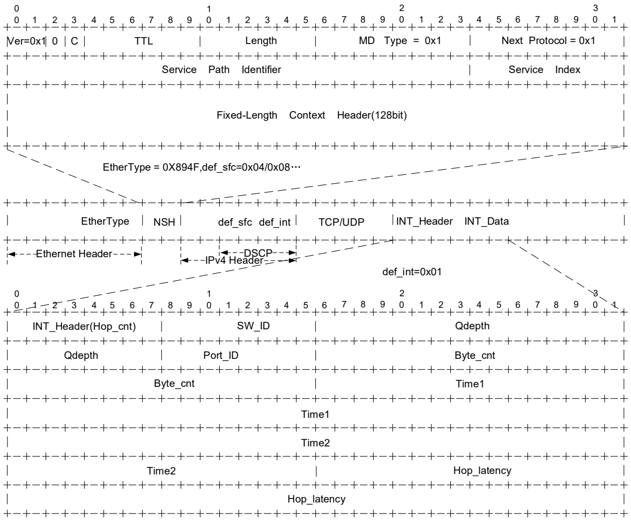

Fig. 5. INT-SFC header format

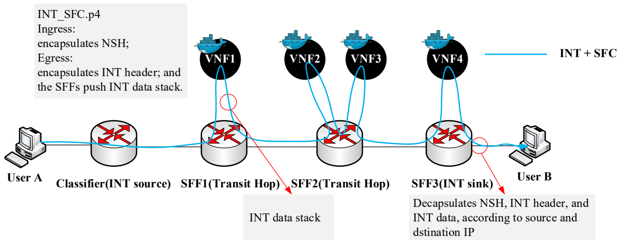

Fig. 6. The workflow of handling packets by P4 switches

C. Collaborated Failure Detection

Collaborated failure detection mainly uses the data plane and control plane to perform failure detection together. We exploit four modules, including Agents for Control\Data Plane Failure Detection, Information Database, Data Processing, DL Detection Algorithms. The agent of control plane mainly collects real-time data such as VNF status, Node status, including CPU, MEMORY, running state, and the other information through the Kubernetes REST API. The agent of data plane uses INT technology to collect fine-grained data, including Queue Depth of SFF, Port ID, SFF ID, Latency, Throughput, etc.. Data plane makes up for the lack of control plane, and can discover the "invisible" failure, such as a high resource utilization, persistent black holes. The control plane can provide a broader sight for the data plane. The collaborated failure detection mechanism is less affected by the propagation relationship between failures.

We capture INT data by Scapy script and use MySQL to construct an information database of failure detection and import all collected data into the information database for decision-making by the knowledge plane.

The data processing module preprocesses the collected multi-dimensional data, removes information irrelevant to failure detection and classification, and ensures the consistency of multi-dimensional information in time. We propose three DL algorithms (Algorithm 1) based on LSTM, BiLSTM and CNN, to detect and classify failures. The dataset is divided into training set, validation set, and test set, with a ratio of 6:2:2 respectively. Before feeding the dataset to train models, we first convert the dataset into word2vec word vectors, Each word corresponds to a unique high-dimensional vector, and then forms the input matrix of the training model, where each row of the matrix corresponds to a word. Our dataset includes 34-dimensional attributes of each SFC, from which n-dimensional useful attributes are selected and preprocessed, and each row of data is filled with n words.

Then we establish the relevant network model, use the activation function ReLU, and use the regularization technology DROPOUT to speed up the convergence of the model. Finally, the Adam optimizer is used for optimization, and the SFC failure detection models based on the DL algorithms are obtained.

Algorithm 1 The Training Process of Algorithms

```
Input: Dataset $D=\{(x_1,y_1),(x_2,y_2),\ldots,(x_m,y_m)\}$,
        $x_i \in \{U_{Ci}^{NF}, U_{Si}^{NF}, ID_{SWi}, Q_i, ID_{Proti}, D_{Lnki}, D_{Pi}, U_{Ci}^{P}, U_{Si}^{P}\}$;
        $i$ represents the $i-th$ SFC
Output: Models-pool $\Im$ with various parameters
Initialize the model's parameters $\Re_0=0$, $N_0=0.1$, $L_0=0$, Learning rate $\eta$
Data processing using word_embedding method
for each step of training epochs do
    for each iteration do
        compute the cross_entropy $C$;
        compute the loss $\overline{C}$;
        $\Gamma \leftarrow \lambda(\ddot{Y}-\dot{Y})^2$;
        $\overline{C} \leftarrow C+\Gamma$;
        update model's parameters according to its gradient descent algorithms
        $\overline{N} \leftarrow N' - \eta\nabla$;
        $\overline{\Re} \leftarrow \Re' - \eta\nabla$;
    end for
    Save $\overline{N}$ and $\overline{\Re}$ to pool $\Im$
end for
```

V. Experimental Evaluation

A. Experiment Setting

The network topology is shown in Fig. 7. A total of 7 servers include 1 control node (deploying K8S and ONOS) and 6 worker nodes. Server configuration are 7 DELL PowerEdge R740s, with 16-Core 2.4 GHz Intel Xeon CPU, 64GB DDR4 RAM and 8 1-Gbps Ethernet NICs. We implement our method and run experiment based on Ubuntu Linux operation system version 18.04. In order to diversify the data, we construct 4 different VNFs, which produce a total of 4 SFCs such as $h1 \rightarrow h11$, $h2 \rightarrow h22$, $h3 \rightarrow h33$, $h4 \rightarrow h44$. We end up with around 1,5K rows of data with 34 features from the actual system and manually inject 8 types of SFC failures. We use the open-source tool *stress-ng* to exhaust CPU and MEMORY resources. In the experiment, both clients and servers run *iperf* and exchange UDP traffic.

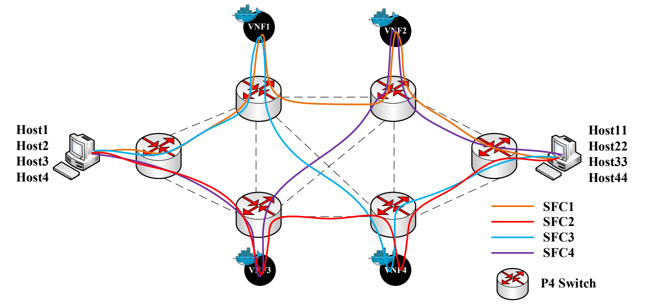

Fig. 7. Prototype topology

B. Metrics

Our experiments use four performance indicators to evaluate the FullSight performance. We let F_A, F_P, F_R and $F1score$ be the accuracy, precision, recall and comprehensive evaluation index respectively, which are defined by:

$$F_A = \frac{F_{TP}+F_{TN}}{F_{TP}+F_{TN}+F_{FP}+F_{FN}} \tag{1}$$

$$F_P = \frac{F_{TP}}{F_{TP}+F_{FP}} \tag{2}$$

$$F_R = \frac{F_{TP}}{F_{TP}+F_{FN}} \tag{3}$$

$$F1score = \frac{2*F_P*F_R}{F_P+F_R} \tag{4}$$

where $F_{TP}, F_{TN}, F_{FP}, F_{FN}$ represent true positive, true negative, false positive, false negative, respectively.

C. Evaluation

Table 2 shows that FullSight can accurately identify 8 types of failure. It can be concluded from our experimental results that Our algorithms can identify multiple failure on a service chain at the same time. We exhibit some experimental consequence to demonstrate the effectiveness of our scheme.

Fig. 8 shows the results that we use the accuracy, precision, recall, and $F1score$ to evaluate the three detection algorithms proposed. Experimental results show that the effect of CNN is better. The accuracy, precision, recall, and $F1score$ of FullSight are up to 93.5%, 89.2%, 88.9%, and 91.0%, respectively.

Fig. 9 demonstrates that the detection time of FullSight is much lower than that of *hping3* when one service chain exists more than one failure. However, when there is only one failure in SFC, the time of detecting the failure by *hping3* is lower than that of FullSight. The failures types of Fig. 9 includes

TABLE II. TYPES OF DETECTION FAILURE

Failure categories	*Normal*	*Node*	*Link*	*VNF*	*Node+VNF*	*Persistent Black holes*	*Link+VNF*	*Persistent Black holes+ Pod*	*Qdepth*
ID	0	1	2	3	4	5	6	7	8

vLink, Pod. Based on the above situation, FullSight has shown the excellent performance. In addition, FullSight's telemetry information occupies 28 Bytes per packet, compared with the traditional method *hping3*, it reduces probe overhead by 57%.

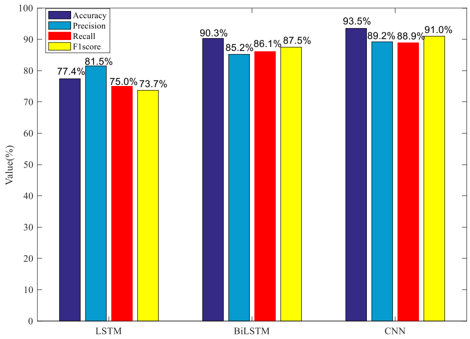

Fig. 8. Comparision of F1score and accuracy with three algorithms

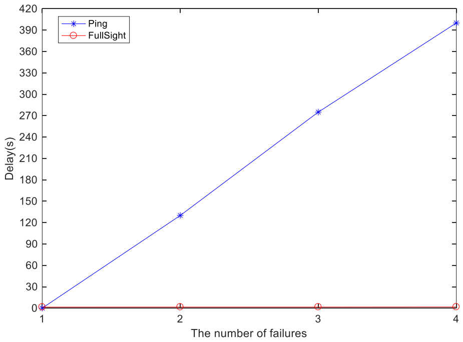

Fig. 9. Comparison of detection latency with different number of failures

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose a collaborative detection framework FullSight, in which the control plane works collaboratively with data plane and DL algorithms based are adopted for failure detection. Experimental results show that, compared with the existing methods, FullSight can detect diversified failures and can achieve a lower overhead and higher detection accuracy. When there are multiple failures in SFC, FullSight has an extremely low detection latency. In the next step, we will continue to optimize FullSight in two aspects: reducing the detection delay when there is only one failure in SFC, and improving the accuracy of the algorithms.

ACKNOWLEDGMENT

This work is supported by National Key Research and Development Program of China (No. 2018YFE0206800), the 2021JBZD003), and National Natural Science Foundation of Fundamental Research Funds for the Central Universities (No. China (No. 61471029, 61972026).

REFERENCES

[1] N. McKeown et al., “OpenFlow: Enabling innovation in campus networks,” ACM SIGCOMM Comput. Commun. Rev, vol. 38, pp. 69–74, Apr. 2008.

[2] ETSI. NFV Whitepaper. Accessed: Oct. 22, 2012. [Online]. Available: https://portal.etsi.org/nfv/nfv_white_paper.pdf.

[3] J. Chen, J. Chen and H. Zhang, “DRL-QOR: Deep Reinforcement Learning based QoS/QoE-Aware Adaptive Online Orchestration in NFV-Enabled Networks,” IEEE TNSM, pp. 1–16, Jan. 2021.

[4] S. Zhang, Y. Wang, W. Li and X. Qiu, “Service failure diagnosis in service function chain,” 19th APNOMS, Seoul, Korea (South) , Sept. 27-29, 2017, pp. 70-75.

[5] RFC 8924, Service Function Chaining (SFC) Operations, Administration, and Maintenance (OAM) Framework.

[6] M. Guo and P. Bhattacharya, "Controller Placement for Improving Resilience of Software-Defined Networks," 2013 Fourth International Conference on Networking and Distributed Computing, 2013, pp. 23-27.

[7] Cotroneo D, Natella R, Rosiello S. “A fault correlation approach to detect performance anomalies in virtual network function chains,” IEEE 28th ISSRE, Toulouse, France, Oct. 23-26, 2017, pp. 90-100.

[8] Gong J, Li Y, Anwer B, et al. “Microscope: Queue-based Performance Diagnosis for Network Functions,” ACM SIGCOMM, New York, NY, USA, July, 2020, pp. 390-403.

[9] S. G. Kulkarni, G. Liu, K. K. Ramakrishnan, M. Arumaithurai, T. Wood and X. Fu, "REINFORCE: Achieving Efficient Failure Resiliency for Network Function Virtualization-Based Services," in IEEE/ACM ToN, vol. 28, no. 2, April, 2020, pp. 695-708.

[10] A. Elmajed, A. Aghasaryan and E. Fabre, "Machine Learning Approaches to Early Fault Detection and Identification in NFV Architectures," 2020 6th IEEE NetSoft, 2020, pp. 200-208.

[11] L. Tan, W. Su, W. Zhang, et al. “In-band network telemetry: A survey,” Computer Networks, vol. 186, pp. 1-1, Feb. 2021.

[12] J. Gong, R. Miao, M. Yu, et al. “LossRadar: Fast Detection of Lost Packets in Data Center Networks,” ACM SIGCOMM, December, 2016, pp. 481–495.

[13] Tan L, Su W, Miao J, et al. “FindINT: Detect and Locate the Lost In-band Network Telemetry Packet,” IEEE Networking Letters, pp. 1-1, Mar. 2021.

[14] R. A. Eichelberger, T. Ferreto, S. Tandel and P. A. P. R. Duarte, “SFC Path Tracer: A troubleshooting tool for Service Function Chaining,” 2017 IFIP/IEEE IM, Lisbon, Portugal, May. 8-12, 2017, pp. 568-571.

[15] Jia C , Pan T , Bian Z , et al. “Rapid Detection and Localization of Gray Failures in Data Centers via In-band Network Telemetry,” NOMS IEEE/IFIP, Budapest, Hungary, April. 20-24, 2020, pp. 1-9.

[16] Liang J, Bi J, Zhou Y, et al. “In-band network function telemetry,” ACM SIGCOMM 2018, Budapest, Hungary, Aug. 20-25, 2018, pp. 42-44.

[17] Gong J, Li Y, Anwer B, et al. “Microscope: Queue-based Performance Diagnosis for Network Functions,” ACM SIGCOMM, Virtual Event, USA, July, 2020, pp. 390-403.

[18] Shiraishi T, Noro M, Kondo R, et al. “Real-time Monitoring System for Container Networks in the Era of Microservices,” 21th APNOMS, Daegu, Korea (South), Sept. 22-25, 2020, pp. 161-166.