How to Mislead AI-assisted Network Automation in SD-IPoEONs: A Comparison Study of DRL- and GAN-based Approaches

Min Wang, Hancheng Lu, Siqi Liu, and Zuqing Zhu, *Senior Member, IEEE*

***Abstract*—Recently, the combination of artificial intelligence (AI) and software-defined networking (SDN) has attracted intensive research interests because it realizes and promotes AI-assisted network automation (AIaNA). Despite the initial successes of AIaNA, its vulnerabilities, *i.e.*, the downside of the reduction of human involvement achieved by it, have not been carefully explored. In this work, we use software-defined IP over elastic optical networks (SD-IPoEONs) as the background, and study how to mislead the AIaNA system in them. Specifically, we target our attack on the deep neural network (DNN) based traffic predictor in the AIaNA system, and design an adversarial module (ADVM) that can craft and inject adversarial traffic samples adaptively to disturb its operation. We consider two approaches to design the ADVM, *i.e.*, the deep reinforcement learning (DRL) based on deep deterministic policy gradient (DDPG), and the generative adversarial network (GAN) model. Our proposed ADVM can monitor and interact with a dynamic SD-IPoEON to train itself on-the-fly. This enables it to generate and inject adversarial samples in the most disturbing and hard-to-detect way and to severely affect the AIaNA's performance on multilayer service provisioning. Specifically, IP flows will be served incorrectly to result in unnecessary congestions/under-utilizations on lightpaths, and erroneous network reconfigurations will be invoked frequently. Simulation results confirm the effectiveness of our ADVM designs, and show that the GAN-based ADVM achieves better attack effects with smaller perturbation strength.**

***Index Terms*—Artificial Intelligence (AI), Deep reinforcement learning (DRL), Network Automation, Software-defined IP over elastic optical networks (SD-IPoEONs), Generative Adversarial Network (GAN), Adversarial samples.**

I. Introduction

NOWADAYS, the fast-emerging network services have dramatically changed the characteristics of Internet traffic in backbone networks, *i.e.*, not only pushing the traffic volume to grow rapidly but also making the traffic condition to vary more burstily [1, 2]. Therefore, considering the agile optical layer achieved with the flexible-grid elastic optical networks (EONs) [3–7], we expect that a rational combination of IP and EON technologies would be promising to architect future backbone networks. The resulting network architecture, namely, IP-over-EON (IPoEON) [8, 9], will be able to adaptively allocate spectrum resources in the optical layer to bandwidth-variable lightpaths, for supporting upper-layer applications. However, this advantage cannot be fully explored without an effective network control and management (NC&M) scheme [10, 11], which can groom and route IP flows over the lightpaths cost-efficiently to realize high resource utilization and good quality-of-service (QoS) simultaneously.

The requirements on NC&M can be satisfied by leveraging the symbiosis of software-defined networking (SDN) [12–15] and deep learning (DL) based artificial intelligence (AI) mechanisms [16] to realize AI-assisted network automation (AIaNA) [17]. Specifically, as shown in Fig. 1, by inserting an AI module in the centralized control plane of a software-defined IPoEON (SD-IPoEON), we can realize the AIaNA to orchestrate the network elements in both IP and optical layers in a coordinated and intelligent manner [18]. Through data analytics, the AI module first analyzes and forecasts the traffic condition in the SD-IPoEON, and then makes accurate and timely NC&M decisions. Hence, bandwidth resources can be automatically allocated/adjusted in advance to make the network operation significantly more cost-efficient [19].

Nevertheless, the advances/successes on AIaNA should not glamor us to overlook its vulnerabilities, especially when talking about the implementation in production networks. In other words, the downside of the reduction of human involvement achieved by AIaNA should be carefully analyzed to identify all the reliability and security issues, such that important questions such as whether we can trust AIaNA in production networks, to what extent it can be trusted, and how to replace a human operator with it, can be properly answered. However, most of the existing studies on AIaNA did not pursue the research in this direction. This motivates us to investigate how to mislead the AIaNA system for SD-IPoEONs in this work.

Specifically, we target the attack on the deep neural network (DNN) based traffic predictor in the AIaNA system developed in [18]. This is because the operation of a DNN can be quietly disturbed with data poisoning [20], *i.e.*, a malicious party can mislead the DNN to generate incorrect outputs by mixing well-craft adversarial samples in its inputs. Here, we hope to point out that data poisoning works for a wide range of DNNs in addition to those for time series prediction, *e.g.*, the DNNs for classification can be misled too [21]. Therefore, the proposals and conclusions in this work could be generalized to and across other AIaNA systems that use DNNs for quality-of-transmission (QoT) prediction, anomaly detection, exception localization, resource orchestration, *etc*.

In this work, we design an adversarial module (ADVM) that can craft and inject adversarial traffic samples adaptively to disturb the DNN-based traffic predictor, and in turn mislead

M. Wang, H. Lu, S. Liu, and Z. Zhu are with the School of Information Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, P. R. China (email: zqzhu@ieee.org).

Manuscript received on April 23, 2020.

the AIaNA system of an SD-IPoEON. Two approaches are considered to design the ADVM: 1) the deep reinforcement learning (DRL) based on the deep deterministic policy gradient (DDPG) [22], and 2) the generative adversarial network (GAN) model [23]. Note that, although our preliminary study in [24] addressed the DRL-based ADVM, this work expands the analysis to make the problem-solving more comprehensive and proposes the GAN-based ADVM for the first time.

We demonstrate that our proposed ADVM can monitor and interact with a dynamic SD-IPoEON to train itself on-the-fly, which enables it to generate and inject adversarial traffic samples in the most disturbing and hard-to-detect way. Hence, it can successfully mislead the AIaNA system in the SD-IPoEON to severely affect the performance of multilayer service provisioning. Specifically, IP flows will be groomed and routed incorrectly to result in unnecessary congestions/under-utilizations on lightpaths, and erroneous network reconfigurations will be invoked frequently to lead to waste on operational complexity and costs. More importantly, our simulation results indicate that compared with the DRL-based one, the GAN-based ADVM achieves better attack effects with smaller perturbation strength, and this can be realized automatically with less empirical parameter adjustments.

The rest of the paper is organized as follows. Section II briefly reviews the related work. We describe the SD-IPoEON with AIaNA and the operation principle of ADVM in it in Section III. The designs of the DRL- and GAN-based ADVMs are then presented in Sections IV and V, respectively. Next, we discuss the numerical simulations for performance evaluations in Section VI. Finally, Section VII summarizes the paper.

II. Related Work

The NC&M of an SD-IPoEON essentially covers at least two aspects [8], *i.e.*, normal service provisioning and failure protection/restoration. Because its data plane consists of both IP and optical layers, the service provisioning in an SD-IPoEON operates in the multilayer scenario. Specifically, we need to first set up lightpaths in the optical layer by solving the famous routing and spectrum assignment (RSA) problem [25–27], and then groom and route IP flows with time-varying traffic over the lightpaths [18, 28]. As a complex system, an SD-IPoEON can have numerous failure cases [29], and thus multilayer protection/restoration schemes are necessary to guarantee reliable operations. Previously, considering various failure scenarios, the studies in [30–34] have designed several protection/restoration algorithms, while the SDN-based system demonstrations have been mentioned in [18, 35].

Due to the dynamic nature of IP flows, the NC&M decisions that are made based on current network status might not be able to handle future traffic demands well. For instance, if we groom the bursty IP flows whose traffic fluctuations are synchronized over one lightpath, congestions/under-utilizations could happen in the future even though the current status of the lightpath shows no issues. This dilemma suggests that traffic prediction would be necessary for the AIaNA in SD-IPoEONs [18]. In addition to SD-IPoEONs, DNN-based traffic predictions have been widely used in the AIaNA systems

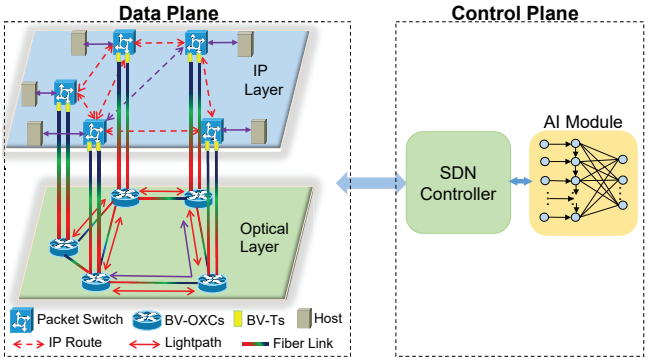


Fig. 1. Architecture of SD-IPoEON with AIaNA, BV-OXC: Bandwidth-variable optical cross-connect, BV-T: Bandwidth-variable transponder.

for various networks too [36–38]. Furthermore, other AIaNA systems also utilized DNNs for QoT prediction [39], security monitoring [40], anomaly detection and fault management [41, 42], network function virtualization [43], datacenter network management [44, 45], *etc*. Nevertheless, none of the aforementioned studies has considered the downside of the reduction of human involvement achieved by AIaNA, *i.e.*, whether or not AIaNA can be fully trusted without human presence.

Unfortunately, it is known that DNNs are vulnerable to the data poisoning with well-crafted adversarial samples and can be easily misled to generate incorrect outputs [20, 21]. Moreover, using visual classification as the background, the survey in [46] showed that an attacker can even force DNN models to produce pre-selected outputs using hard-to-detect adversarial samples. To this end, the vulnerability of DNN-based AIaNA systems should not be overlooked.

Previously, we studied how to mislead the AIaNA for inter-datacenter networks with manually-crafted adversarial traffic samples in [47]. However, as the adversarial samples were generated in a static manner, they would not be effective when traffic condition changed. The authors of [48] proposed a deterministic algorithm to launch black-box adversarial attacks on a DNN-based network traffic classifier. Nevertheless, the adversarial sample generation was not adaptive, which means that the attacks could be easily detected with another deterministic algorithm (*e.g.*, the one in [49]). Therefore, it would be interesting to consider how to leverage AI-based methods to craft the adversarial samples adaptively [50]. However, to the best of our knowledge, none of the existing studies on AIaNA has pursued the research in this direction, except for our previous work [24]. Although the DRL-based ADVM designed in [24] could disturb the operation of the AIaNA system for SD-IPoEONs, the study was still preliminary, because the tradeoff between perturbation strength and attack effect was not optimized and GAN-based approach was not considered.

III. Network Architecture and Operation Principle

In this section, we show the architecture of an SD-IPoEON that leverages DNN-based traffic predictor to realize AIaNA, and describe the operation principle of our proposed ADVM, which can disturb the AIaNA's operation with well-crafted adversarial traffic samples.

A. Network Architecture

Fig. 1 illustrates the network architecture of an SD-IPoEON with AIaNA [18]. The data plane consists of the IP and optical layers, both of which are managed by the centralized control plane. The optical layer is essentially an EON that is built with a few bandwidth-variable optical cross-connects (BV-OXCs) and fiber links. Following the instructions from the SDN controller, each BV-OXC can transparently switch lightpaths with flexible-grid spectrum allocations, and can also insert/terminate lightpaths to/from fiber links if the lightpaths use it as an end-node. Therefore, lightpaths can be established, reconfigured, and torn down to adapt to IP traffic. The packet switches in the IP layer are also managed by the SDN controller, which can install flow tables in them to groom and route dynamic IP traffic generated by the hosts over the lightpaths in the optical layer. Each packet switch connects to a BV-OXC locally via several bandwidth-variable transponders (BV-Ts), which can generate/terminate lightpaths.

The SDN controller monitors the status of the data plane and manages all the data plane elements accordingly. Specifically, it models the IP layer as a graph $G(V, E)$, where V is the set of packet switches and E denotes all the logic links for interconnecting the switches. Here, each logic link $(u, v) \in E$ is actually a lightpath between the BV-OXCs that are local to switches u and v, and the BV-OXCs might not be adjacent in the optical layer, *i.e.*, the lightpath can bypass certain switches in the IP layer. Similar to the work in [18], we still assume that when setting up a new lightpath, the SDN controller always assigns the highest line-rate that the lightpath's transmission distance permits[1]. Hence, the capacity of a logic link (u, v) is determined at the time when its lightpath is set up, and $G(V, E)$ is a dynamic topology whose links can change over time. Then, based on the traffic condition in the data plane, the controller manages the lightpaths in the optical layer to update $G(V, E)$, and grooms and routes IP flows in $G(V, E)$ to not only maximize overall bandwidth utilization but also minimize lightpath congestions.

B. Operation of AIaNA

The cross-layer orchestration in the controller manages dynamic IP traffic in units of IP flows. Here, because we consider the SD-IPoEON as a backbone network, each IP flow ($u \rightarrow v$, $u, v \in V$) is actually an aggregated one that includes the traffic of many socket connections from the hosts attached on *Switch* u to those on *Switch* v. Therefore, even though the traffic on each flow is still highly dynamic, it fluctuates according to a predictable pattern and lasts for a relatively long period of time (*e.g.*, tens of hours or even days) [51]. Meanwhile, because traffic from different network services should be handled independently, we assume that different IP flows can share the same source-destination switch pair [18]. In this case, each packet switch identifies the IP flows by checking their switching labels instead of IP addresses, by using the multi-protocol label switching (MPLS) protocol, which is commonly-used in backbone networks for traffic grooming [30]. Then, the packet switch records historical traffic samples of an IP flow by leveraging the counters associated with the corresponding flow tables, and indices the results with the IP flow's switching label. Next, the SDN controller can poll the packet switch periodically to collect the traffic samples, and use them for AIaNA.

[1]Each BV-T supports a set of line-rates, each of which has a maximum transmission reach according to the QoT constraint [3].

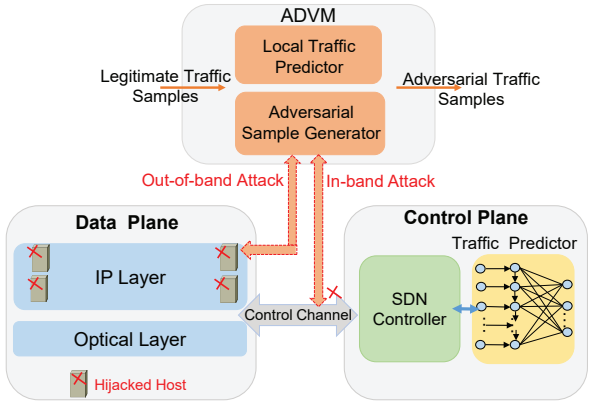


Fig. 2. Operation principle of ADVM to mislead the AIaNA in SD-IPoEON.

As the IP flows have time-varying bandwidth requirements, the grooming and routing of them can be more effective to avoid congestions/under-utilizations on lightpaths in advance, if we leverage a DNN-based traffic predictor to forecast future bandwidth demands accurately. This actually realizes AIaNA in the SD-IPoEON. In other words, the control plane accomplishes the cycle of "observe-analyze-act" to adjust resource allocations in the data plane, such that the cost-effectiveness of network operation can be improved significantly. We build the traffic predictor based on the long/short-term memory based DNN (LSTM-DNN) [52], because it is one of the most widely-used DNN models for forecasting time series. In [18], we have demonstrated a traffic predictor based on LSTM-DNN, which can produce highly-accurate predictions when being tested with the realistic traffic traces taken by Internet service providers (ISPs) in different countries.

Then, based on the accurate traffic predictions and current network status, the AIaNA system calculates and implements suitable multilayer provisioning schemes for the IP flows, to realize proactive cross-layer orchestration in the SD-IPoEON. The AIaNA system can utilize the congestion-relieving value (CRV) algorithm developed in [18] to achieve the proactive cross-layer orchestration. Specifically, based on traffic predictions provided by the DNN-based traffic predictor, the CRV algorithm generates intelligent on-line NC&M decisions to re-groom and reroute IP flows and to reconfigure lightpaths such that the performance tradeoff among lightpath utilization, congestion probability, and reconfiguration frequency can be balanced well. The algorithm determines how to re-groom the IP flows by comparing their future traffic fluctuations and avoiding the cases that those whose peak time overlaps with each other are groomed on the same lightpath.

C. Operation of ADVM

In normal operation of the AIaNA system, the accuracy of the traffic predictor is crucial. Specifically, the AIaNA analyzes

traffic predictions to obtain future network status, and when it foresees congestions/under-utilizations on certain lightpaths, it will invoke the corresponding network reconfigurations to maintain the performance and cost-effectiveness of network operation. That is to say, the AIaNA relies on accurate traffic predictions to properly balance the tradeoff among overall bandwidth utilization, number of lightpath congestions, and times of network reconfigurations.

However, LSTM-DNNs are vulnerable to data poisoning and can be easily misled to generate incorrect predictions with well-crafted adversarial samples [20, 21]. In order to investigate how badly the traffic predictor can be misled and what the harmful effects are, we design an ADVM to disturb the AIaNA system in the most devastating and hard-to-detect manner. We assume that the ADVM has the capability to monitor the traffic conditions on certain lightpaths in the SD-IPoEON, which can be achieved with a few well-known methods [53]. Based the traffic conditions, the ADVM generates and injects adversarial traffic samples adaptively to make the traffic predictor malfunction. As shown in Fig. 2, the ADVM has the options to launch the adversarial-sample-based attacks in the in-band and out-of-band manners.

To launch the in-band attacks, the ADVM hacks into the control channels between the control and data planes, eavesdrops legitimate traffic samples, produces adversarial samples based on them, and injects the adversarial samples back. This is feasible, because OpenFlow usually sets up control channels with the transport layer security (TLS) or secure socket layer (SSL) connections, which are known to be susceptible to the man-in-the-middle attack [54], especially by using the technique of padding oracle on downgraded legacy encryption (POODLE) [55]. Regarding the out-of-band manner, the attacker deploys a few probes in the data plane to collect traffic samples quietly [53], generates adversarial samples accordingly, and instructs several hijacked hosts to pump traffic into the targeted lightpaths for adversarial sample injection. Note that, host-hijacking is widely used by malicious parties to launch various attacks, *e.g.*, the well-known distributed denial of service (DDoS) attack. However, compared with DDoS, our ADVM needs to hijack fewer hosts and use them to pump much less traffic into the network. From the perspective of adversarial sample generation, the in-band and out-of-band manners do not have fundamental differences, except for that the adversarial samples generated in the out-of-band manner are always greater than the corresponding legitimate ones. Hence, we will not specify the attack scenario in the following discussions, and restrict that the adversarial samples should always be greater than the corresponding legitimate ones.

Fig. 2 also explains how the adversarial samples are generated in the ADVM. First of all, by monitoring the traffic condition in the SD-IPoEON, the ADVM collects legitimate traffic samples, and uses them to train its local traffic predictor for making accurate predictions, such that the local traffic predictor can imitate the operation of the legitimate traffic predictor attached to the SDN controller. Note that, as the two predictors are trained independently, they do not have to use the same architecture. This means that the ADVM can architect its local traffic predictor without any knowledge about the legitimate one. Next, the trained local traffic predictor generates an emulated network environment, and then the adversarial sample generator interacts with it to craft and inject adversarial traffic samples adaptively, for disturbing the legitimate traffic predictor. As the whole process does not count on any information about the legitimate traffic predictor, the ADVM realizes the "black-box" adversarial attack. Moreover, the ADVM only needs to collect and inject traffic samples in an SD-IPoEON to launch the adversarial-sample-based attacks, and its operation does not count on any other network information regarding the SD-IPoEON.

With the trained local traffic predictor, the adversarial sample generator modifies legitimate samples to adversarial ones, inputs the results to the local traffic predictor to see how the predictions will change, and optimizes the results to achieve the largest attack effect with the smallest perturbation strength.

Definition 1. *The **perturbation strength** refers to not only the percentage of the legitimate samples that will be modified but also the maximum relative error (MRE) made on each modified sample, and both of them should be kept as small as possible to make the attacks hard-to-detect.*

We denote the traffic predictions with legitimate and adversarial samples as $P = \{p_1, \cdots, p_M\}$ and $\hat{P} = \{\hat{p}_1, \cdots, \hat{p}_M\}$, respectively, where M is the number of predicted samples in each series[2]. Their mean squared error (MSE) is

$$\text{MSE}(P, \hat{P}) = \frac{1}{M} \sum_{i=1}^{M} (\hat{p}_i - p_i)^2 , \tag{1}$$

where p_i and $\hat{p}_i$ are the i-th samples in the corresponding predictions, and they both have been normalized. Hence, the MSE changes within $[0, 1]$, while a larger MSE suggests a better attack effect. Meanwhile, the similarity of the predictions' fluctuations over time can be quantified with the Pearson correlation coefficient (PCC) as

$$\text{PCC}(P, \hat{P}) = \frac{\text{cov}(P, \hat{P})}{\sqrt{\text{var}(P) \cdot \text{var}(\hat{P})}}, \tag{2}$$

where $\text{cov}(P, \hat{P})$ calculates the covariance of P and $\hat{P}$, and $\text{var}(\cdot)$ obtains the standard deviation of a series. The PCC in Eq. (2) varies within $[-1, 1]$, and its value increases when P and $\hat{P}$ fluctuate more similarly. We have $\text{PCC}(P, \hat{P}) = -1$, if they fluctuate oppositely (*i.e.*, having the total negative linear correlation). Note that, if the adversarial samples can make the traffic predictor to produce inaccurate predictions that have the opposite fluctuation of the actual traffic trend (*e.g.*, those in Fig. 3), the ADVM can mislead the AIaNA to the maximum extend. This is because the AIaNA can mistakenly consider future traffic valleys as peaks, or *vice versa*, which will degrade its performance in all the three aspects, *i.e.*, the overall bandwidth utilization, number of lightpath congestions,

[2]Note that, to facilitate precise traffic prediction, we normalize the input samples to the traffic predictors used in this work with respect to their maximum value (*i.e.*, the throughput of the lightpath that carries them). Hence, both the collected and predicted traffic samples discussed in this paper are normalized ones that vary within $[0, 1]$, except for those in Section VI, because the simulations restore traffic samples to their actual values after prediction.

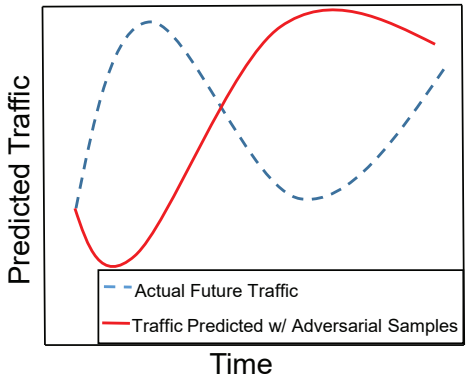


Fig. 3. Example on attack effect of adversarial traffic samples.

and times of network reconfigurations. To this end, the attack effect becomes better when PCC decreases.

Definition 2. *If the traffic predictions with legitimate and adversarial samples are P and $\hat{P}$, respectively. The* ***attack effect*** *of the adversarial samples is quantified as*

$$\eta = MSE(P, \hat{P}) - PCC(P, \hat{P}). \tag{3}$$

In the next two sections, we will design the adversarial sample generator for ADVM based on two approaches, *i.e.*, a DRL model using DDPG and a GAN model, respectively. Both approaches can interact with the dynamic SD-IPoEON to train their models on-the-fly. Through the online training, they can generate and inject adversarial samples adaptively such that the attack effect in Eq. (3) can be maximized while the perturbation strength is kept as small as possible.

IV. Design of DRL-based ADVM

In this section, we design the adversarial sample generator for ADVM based on a DRL model that uses DDPG [22]. The DRL model utilizes an intelligent agent to interact with the time-varying environment emulated by the local traffic predictor, and selects proper actions based on the observed states to generate adversarial traffic samples adaptively and inject them in the data plane of the SD-IPoEON. Consequently, the traffic predictor attached to the SDN controller can be misled to output incorrect predictions, and the AIaNA in the SD-IPoEON will be disturbed with the maximized attack effect. We design the DRL model as follows.

- **State**: state $\mathbf{S}_i$ refers to the state of the historical traffic samples collected at time instant i. We have $\mathbf{S}_i = \{s_{i-M_1+1}, \cdots, s_i\}$, where s_i is the instant traffic volume on a monitored lightpath at time i, and M_1 is the number of historical samples collected for traffic prediction. We define $\mathbf{S}_0$ as the initial state, which refers to the historical samples collected by the ADVM when it first starts.
- **Action**: action $\mathbf{A}_i$ is the action taken at time i, which indicates how to modify the legitimate traffic samples (the location and magnitude of perturbations) within a preset look-ahead time, *i.e.*, time instance $j \in [i+1, i+M_2]$, where M_2 is the look-ahead duration.
- **Reward**: reward $\mathbf{R}_i$ of action $\mathbf{A}_i$ is calculated by comparing the traffic predictions based on legitimate and adversarial samples. As shown in Fig. 4, the local traffic predictor first takes $\mathbf{S}_0$ as the input and forecasts M_3 legitimate traffic samples as P_0, and when the observed state transferred to $\mathbf{S}_i$ at time i, the local predictor obtains a new prediction P_i (also with M_3 samples). We take the parts of P_0 and P_i, which cover the overlapped time duration, denote them as P and $\hat{P}$ (*i.e.*, the traffic predictions with legitimate and adversarial samples), and calculate the reward $\mathbf{R}_{i-1} = R(S_0, S_i, P_0, P_i, A_{i-1})$ as

$$\mathbf{R}_{i-1} = \Big[\text{MSE}(P, \hat{P}) - \text{PCC}(P, \hat{P})\Big] - \frac{1}{M_1}\sum_{j=1}^{M_1}\frac{\hat{s}_j - s_j}{s_j}, \tag{4}$$

where we redefine the overlapped parts of P_0 and P_i as $P = \{p_1, \cdots, p_M\}$ and $\hat{P} = \{\hat{p}_1, \cdots, \hat{p}_M\}$, respectively. We use $S_0 = \{s_1, \cdots, s_{M_1}\}$ to denote the legitimate samples, and define $S_i = \{\hat{s}_1, \cdots, \hat{s}_{M_1}\}$ as the adversarial samples generated based on S_0 by action A_{i-1}, where M_1 is the number of concerned samples. The first term in Eq. (4) is the attack affect defined in Eq. (3), and the second one is the perturbation strength.

Because the aforementioned DRL model has relatively large state and action spaces, we design it by leveraging the DDPG scheme, which is known to be powerful on optimizing actions in states while both of their spaces are high-dimensional and continuous [22]. Specifically, as shown in Fig. 4, the DRL-based adversarial sample generator adopts the advantage actor-critic (A2C) learning strategy to avoid the difficulty of action selection due to the need of traversing the entire action and state spaces. Here, the DRL agent has a double network architecture, which consists of an actor neural network (A-NN) for outputting specific actions based on states and a critical neural network (C-NN) for evaluating the selected actions.

The A-NN directly selects an action $\mathbf{A}_i$ based on state $\mathbf{S}_i$ with the deterministic policy selection function $\mu(\mathbf{S}_i|\theta^a)$ as

$$\mathbf{A}_i = \mu(\mathbf{S}_i|\theta^a), \tag{5}$$

where θ^a denotes the A-NN's parameters. The C-NN uses an action-state value function, *i.e.*, the Q function ($Q(\mathbf{S}_i, \mathbf{A}_i|\theta^c)$), to evaluate the quality of action $\mathbf{A}_i$ on state $\mathbf{S}_i$.

$$Q(\mathbf{S}_i, \mathbf{A}_i|\theta^c) = \mathbf{R}_i + Q(\mathbf{S}_{i+1}, \mathbf{A}_{i+1})|\mathbf{S}_i, \mathbf{A}_i, \tag{6}$$

where θ^c is the C-NN's parameters. The C-NN then sends the action gradient ($\nabla_{\mathbf{A}_i} Q(\mathbf{S}_i, \mathbf{A}_i|\theta^c)$) to the A-NN for increasing its probability of selecting the action with a larger Q.

In online training, the A-NN continuously optimizes its policy selection function $\mu(\mathbf{S}_i|\theta^a)$ using the policy gradient

$$\nabla_{\theta^a} J \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_{\theta^a}\mu(\mathbf{S}_i|\theta^a) \cdot \nabla_{\mathbf{A}_i} Q(\mathbf{S}_i, \mathbf{A}_i|\theta^c), \tag{7}$$

where J denotes its overall performance metrics, N is the number of iterations in training, and $\nabla_{\theta^a}\mu(\mathbf{S}_i|\theta^a)$ is the gradient of $\mu(\mathbf{S}_i|\theta^a)$. Meanwhile, the C-NN optimizes $Q(\mathbf{S}_i, \mathbf{A}_i|\theta^c)$ by minimizing the squared loss between the expected and estimated Q values, which is defined as

$$L = \frac{1}{N}\sum_{i=1}^{N}\left[\mathbf{R}_i + \kappa \cdot Q(\mathbf{S}_{i+1}, \mathbf{A}_{i+1}|\theta^c) - Q(\mathbf{S}_i, \mathbf{A}_i|\theta^c)\right]^2, \tag{8}$$

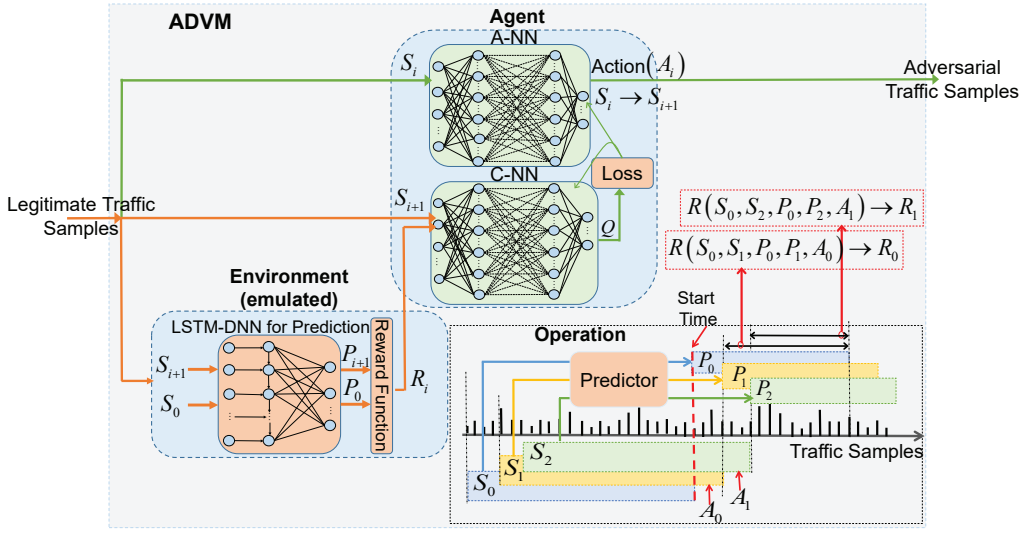


Fig. 4. Architecture and operation principle of DRL-based ADVM.

where $\kappa \in (0, 1)$ is a constant for the discount factor.

The operation of the DRL-based adversarial sample generator is explained in *Algorithm* 1. *Lines* 1-2 is for the initialization. Next, in the for-loop that covers *Lines* 3-15, the DRL model trains its A-NN and C-NN in the online manner, which means that it interacts the time-varying environment of the SD-IPoEON continuously and optimizes its decision-making on-the-fly to maximize the attack effect. Here, to ensure that the DRL can handle relatively long traffic series, we divide its operation into episodes, each of which covers K time instants[3]. In each episode of online operation, *Lines* 4-5 are for the initialization. Then, the while-loop covering *Lines* 6-14 explains the operation of the DRL at each time instant in an episode, where *Lines* 11-13 explain how the A-NN and C-NN conduct the online training to optimize themselves with the entries stored in the experience buffer **EB**.

Algorithm 1: DRL-based Adversarial Sample Generation

1 initialize parameters of A-NN and C-NN (θ^a and θ^c) randomly;
2 empty the experience buffer **EB**;
3 **for** *each episode of online operation* **do**
4 obtain initial state $\mathbf{S}_0$;
5 get original traffic prediction P_0 with local predictor;
6 **for** $i = 1$ *to* K **do**
7 A-NN selects action $\mathbf{A}_i$ based on $\mathbf{S}_i$ with Eq. (5);
8 execute $\mathbf{A}_i$ and obtain new state $\mathbf{S}_{i+1}$;
9 get reward $\mathbf{R}_i$ with Eq. (4);
10 store $\{\mathbf{S}_i, \mathbf{A}_i, \mathbf{R}_i, \mathbf{S}_{i+1}\}$ as an entry in **EB**;
11 select N continuous entries in **EB** randomly;
12 C-NN uses the batch of entries to update θ^c such that the loss in Eq. (8) is minimized;
13 A-NN optimizes $\mu(\mathbf{S}_i|\theta^a)$ by applying Eq. (7) on the batch of entries;
14 **end**
15 **end**

V. Design of GAN-based ADVM

In this section, we propose the adversarial sample generator for ADVM based on GAN [23]. The benefit of this approach is that the GAN can automatically balance the tradeoff between perturbation strength and attack effect.

A. Architecture of GAN-based ADVM

As we have explained in Section III-C, the ADVM should have the capability to optimize the tradeoff between perturbation strength and attack effect. However, for the DRL model designed in the previous section, the size of its action space increases dramatically if we do not apply an empirically-determined upper-bound on the percentage of the legitimate samples that will be modified. This, however, restricts the adaptivity of the DRL-based ADVM.

Therefore, we leverage the GAN model to design another adversarial sample generator. As shown in Fig. 5, the GAN consists of two neural networks, which are the generator neural network (Gen-NN) for crafting the adversarial samples, and the discriminator neural network (Dis-NN) for ensuring that the perturbation strength of the generated adversarial samples is minimized. The GAN model is also trained in the online manner. Specifically, the Gen-NN gets trained to capture the distribution of legitimate samples for crafting adversarial ones, while the Dis-NN is trained to estimate the probability that a sample is legitimate (*i.e.*, unmodified). We train the Gen-NN and Dis-NN simultaneously, until the adversarial samples generated by the Gen-NN can maximize the Dis-NN's error rate. Hence, the tradeoff between perturbation strength and attack effect gets optimized automatically in the training.

We denote the predicted legitimate traffic samples as $\mathbf{S}$, which is obtained with the local traffic predictor using the scheme that is similar to the one used for the DRL-based

[3]Note that, the value of K should be selected empirically according to traffic dynamics, and we set $K = 240$ in our simulations in Section VI.

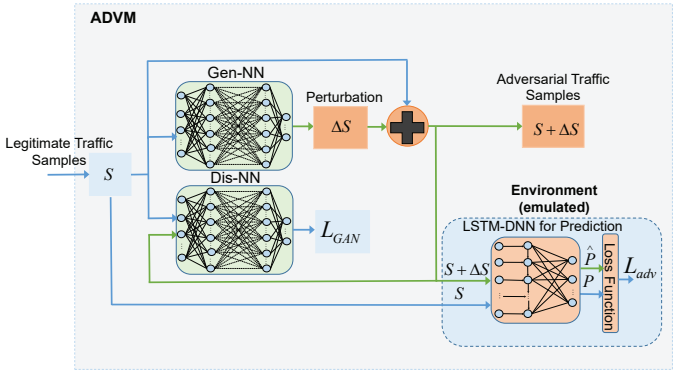


Fig. 5. Architecture and operation principle of GAN-based ADVM.

ADVM. Taking $\mathbf{S}$ as the input, the Gen-NN produces a perturbation $\Delta\mathbf{S}$. Hence, the adversarial samples will be $\mathbf{S}+\Delta\mathbf{S}$, which is sent to the Dis-NN together with $\mathbf{S}$ for obtaining

$$L_{\text{GAN}} = \text{avg}(\mathbf{S}) \cdot \log\{Dis(\mathbf{S}) \cdot [1 - Dis(\mathbf{S}+\Delta\mathbf{S})]\}, \quad (9)$$

where L_{GAN} is the adversarial loss, avg$(\cdot)$ returns the average value of a time series, and $Dis(\cdot)$ denotes the output of the Dis-NN after taking a time series as the input. Meanwhile, both the legitimate and adversarial samples are fed into the local traffic predictor to get the traffic predictions based on them, namely, P and $\hat{P}$, respectively. Then, the ADVM calculates the loss caused by the adversarial-sample-based attack as

$$L_{\text{adv}} = \text{avg}(\mathbf{S}) \cdot \eta, \quad (10)$$

where η is the attack effect computed with P and $\hat{P}$ using Eq. (3). Next, we add a soft hinge loss [23] based on the ℓ^2-norm of $\Delta\mathbf{S}$ to restrict the perturbation strength

$$L_{\text{per}} = \text{avg}(\mathbf{S}) \cdot \max(0, \|\Delta\mathbf{S}\|_2 - 1). \quad (11)$$

Finally, we obtain the overall loss function as

$$L = L_{\text{GAN}} + L_{\text{per}} - L_{\text{adv}}, \quad (12)$$

which measures the GAN's performance in the online training, *i.e.*, a smaller overall loss L indicates that it performs better.

B. Online Training of GAN

The online training updates the parameters of the Gen-NN and Dis-NN (*i.e.*, θ^g and θ^d, respectively) simultaneously to see the convergence of the overall loss in Eq. (12). We define the loss functions of the Gen-NN and Dis-NN as

$$L_G(\mathbf{S}|\theta^g) = \text{avg}(\mathbf{S}) \cdot \log[1 - Dis(\mathbf{S}+\Delta\mathbf{S})] + L_{\text{per}} - L_{\text{adv}}, \quad (13)$$

$$L_D(\mathbf{S}|\theta^d) = \text{avg}(\mathbf{S}) \cdot \log\{Dis(\mathbf{S}) \cdot [1 - Dis(\mathbf{S}+\Delta\mathbf{S})]\}. \quad (14)$$

Algorithm 2 explains the procedure of the GAN-based adversarial sample generation. *Lines* 1-2 are for the initialization, and the subsequent while-loop describes the GAN's operation. Here, *Lines* 5-15 are for the online training. Specifically, we train the GAN model repeatedly during network operation to ensure its adaptivity. Each training runs for M iterations (*Lines* 6-14). In each iteration, the Dis-NN first gets trained for K times (*Lines* 7-10) to minimize its loss function defined in Eq. (14) (*i.e.*, making sure that it can distinguish the legitimate and adversarial samples accurately), and then we train the Gen-NN with *Lines* 11-13 to guarantee that it can generate the adversarial samples whose attack effect is maximized under the current perturbation strength. Therefore, through the online training, we optimize the Gen-NN and Dis-NN simultaneously until they both cannot be improved anymore. At this moment, the adversarial samples generated by the Gen-NN can mislead the traffic predictor to obtain the maximum attack effect, while the Dis-NN has the smallest success rate to distinguish the generated adversarial samples from the legitimate ones (*i.e.*, the adversarial-sample-based attacks become hard-to-detect).

Algorithm 2: GAN-based Adversarial Sample Generation

```
1  initialize parameters of Gen-NN and Dis-NN (θ^g and θ^d) randomly;
2  empty the traffic database TD;
3  while the ADVM is operational do
4      collect historical legitimate traffic samples and store them in TD;
5      if it is the time for online training then
6          for i = 1 to M do
7              for j = 1 to K do
8                  select N continuous historical samples S from TD randomly;
9                  update Dis-NN (θ^d) according to stochastic gradient ∇_{θ_d} L_D(S|θ^d);
10             end
11             select N continuous historical samples S from TD randomly;
12             get traffic prediction P by inputting S in local traffic predictor;
13             update Gen-NN (θ^g) according to stochastic gradient ∇_{θ_g} L_G(S|θ^g);
14         end
15     end
16     generate and inject adversarial samples;
17 end
```

VI. Performance Evaluations

In this section, we perform numerical simulations to compare the DRL- and GAN-based ADVMs.

A. Simulation Setup

The simulations use the 14-node NSFNET in Fig. 6 as the topology of the optical layer in the SD-IPoEON. We assume that each BV-T in the SD-IPoEON can support the line-rates within $\{10, 25, 40, 50, 75, 100\}$ Gbps, whose maximum transmission reaches are $\{3732, 2995, 2671, 2438, 2112, 1880\}$ km, respectively [18]. The simulations consider dynamic network environments, where the number of initial lightpaths is distributed within $[44, 50]$. Then, dynamic IP flows are generated according to the Poisson model, where the traffic fluctuation of each flow follows a realistic trace taken from the data set in [56]. Here, the sampling interval of each trace is 5 minutes, and the peak rate of each flow is randomly select

within $[4, 10]$ Gbps. We denote one traffic sampling interval as a time slot (TS) (*i.e.*, each TS is 5 minutes), and leverage the CRV algorithm developed in [18] to calculate the multilayer provisioning schemes of the dynamic IP flows.

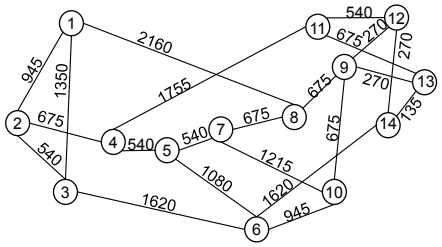
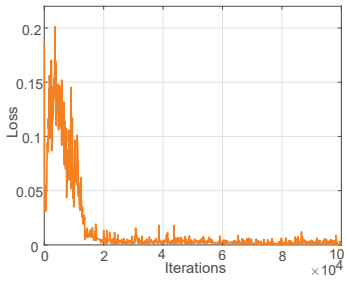
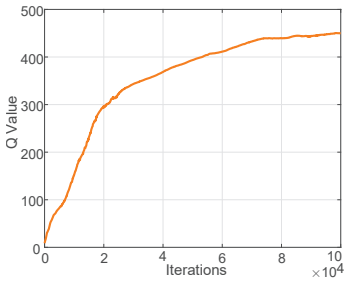


Fig. 6. NSFNET topology with link lengths marked in kilometers.

The traffic predictor in the AIaNA system and the local traffic predictor in the ADVM are trained independently, also with the realistic traces in [56], and we confirm that both of them can achieve a higher than 95.9% prediction accuracy on testing data sets. Similar to the setting in [24], we still assume that the DRL-based ADVM can only modify at most 40% of legitimate samples to launch its adversarial-sample-based attacks. Note that, the actual portion of samples to modify is determined by its DRL agent and could be much less than 40%. For the GAN-based ADVM, we do not set this percentage upper-bound because it can minimize the portion of adversarial samples automatically. In the simulations, we average the results from 10 independent runs to get each data point, for ensuring sufficient statistical accuracy.



(a) Loss value

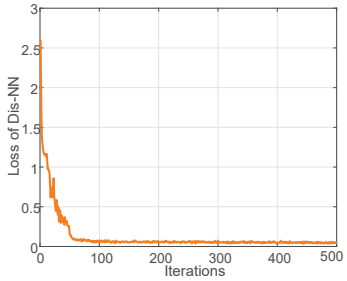


(b) Q value

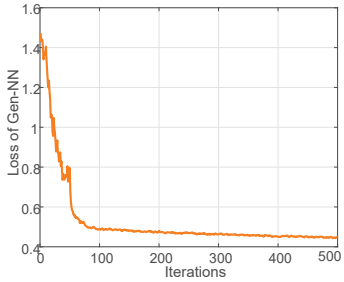
Fig. 7. Training performance of DRL-based ADVM.

B. Online Training Performance of ADVMs

We first leverage a lightpath whose traffic has $170,000$ time-varying samples to explain the performance of the ADVMs' online training. Here, when generating adversarial samples, the ADVMs limit the maximum relative error (MRE) as 20%.



(a) Loss value of Dis-NN

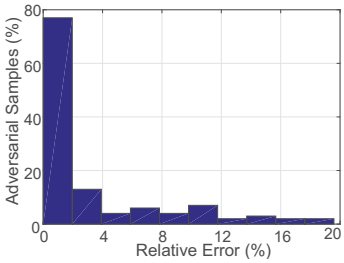


(b) Loss value of Gen-NN

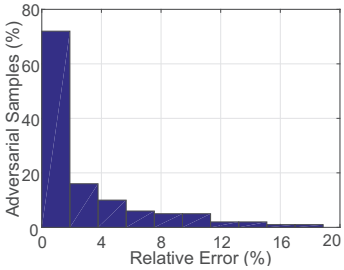
Fig. 8. Training performance of GAN-based ADVM.

1) DRL-based ADVM: Fig. 7 shows the training performance of the DRL-based ADVM, which suggests that the training converges quickly after $\sim 20,000$ iterations. Specifically, after having been trained for $20,000$ iterations, the loss value (defined in Eq. (8)) in Fig. 7(a) approaches to 0, while its Q value (in Eq. (6)) in Fig. 7(b) starts to increase slowly.

2) GAN-based ADVM: The training performance of GAN-based ADVM is in Fig. 8. The training converges quickly within ~ 100 iterations, and specifically, the loss value of Dis-NN (defined in Eq. (14)) in Fig. 8(a) approaches to 0 after 50 iterations, while the loss value of Gen-NN (defined in Eq. (13)) in Fig. 8(b) has also converged after 80 iterations.

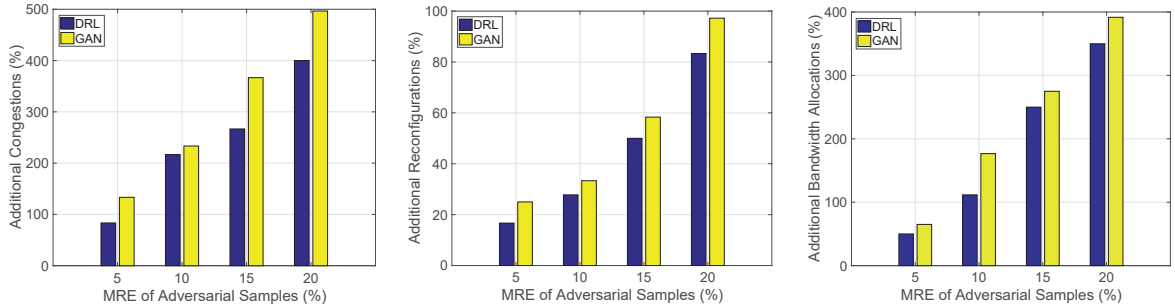


(a) DRL-based ADVM



(b) GAN-based ADVM

Fig. 9. Distribution of relative errors on adversarial samples.



(a) Additional lightpath congestions (b) Additional network reconfigurations (c) Additional bandwidth allocations

Fig. 10. Adverse effects of ADVMs on AIaNA system in SD-IPoEON.

Fig. 9 compares the distributions of the relative errors introduced by the ADVMs to generate adversarial samples. In Fig. 9(a), we notice that 90.9% of the adversarial samples from the DRL-based ADVM have their relative errors below 5%, and the average relative error of them is 3.21%. On the other hand, the results regarding the GAN-based ADVM in Fig. 9(b) indicate that the ratio of the adversarial samples whose relative errors are less than 5% is 91.2%, while the average relative error is 2.90%. Hence, the GAN-based approach introduces a smaller perturbation strength, and thus its adversarial samples would be harder to be detected.

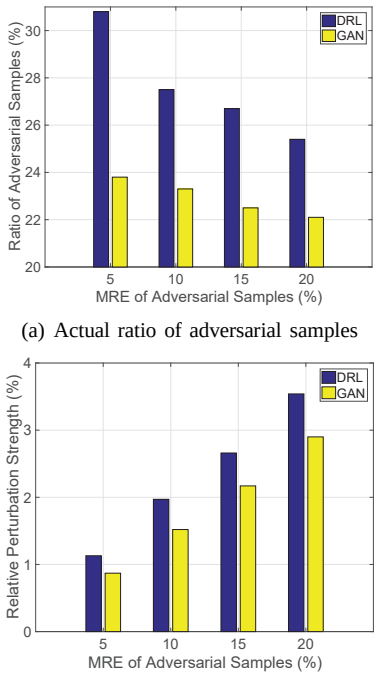
C. Adverse Effects of ADVMs on AIaNA in SD-IPoEON

We insert the ADVMs in the SD-IPoEON, let them launch adversarial-sample-based attacks with MREs within $[5\%, 20\%]$, and check their adverse effects on the AIaNA system. Specifically, in the simulations, the ADVMs monitor the traffic condition in the SD-IPoEON and leverage their DRL/GAN models to determine when and how to inject adversarial traffic samples in targeted lightpaths. Each simulations runs for $4,500$ TS', and we compare the scenarios with and without the ADVMs. As the ADVMs disturb the operation of the AIaNA to make NC&M decisions based on incorrect traffic predictions, they can cause additional lightpath congestions, bandwidth allocations, and network reconfigurations, which can quantify their adverse effects.

Fig. 10 summarizes the ADVMs' adverse effects. It can be seen that the ADVMs induce substantial increases on all the three metrics, and this confirms that the operation of the AIaNA system can be disturbed significantly. As expected, the adverse effects become larger when the MRE of the adversarial samples increases. The results in Fig. 10 also suggest that the adverse effects of the GAN-based ADVM are larger than their counterparts caused by the DRL-based ADVM.

Definition 3. *For legitimate traffic samples* $\mathbf{S}$*, if an ADVM generates the adversarial samples as* $\mathbf{S}+\Delta\mathbf{S}$*, then the* ***relative perturbation strength*** *is defined as* $\frac{sum(\Delta\mathbf{S})}{sum(\mathbf{S})}$*, where* $sum(\cdot)$ *returns the summation of a time series.*

Fig. 11 compares the perturbation strengths from the DRL- and GAN-based ADVMs, which not only shows the actual ratios of adversarial samples in Fig. 11(a) but also compares their relative perturbation strengths in Fig. 11(b). Here, the relative perturbation strength actually quantifies the volume of the adversarial traffic that needs to be injected for launching the adversarial-sample-based attacks, and thus it can also quantify the potential cost of the attacks. We observe that the actual ratios of adversarial samples generated by the DRL-based ADVM can be far below the preset upper-limit 40%, while those from the GAN-based ADVM are even less. This verifies that the ADVMs are adaptive and effective, and the GAN-based one is more intelligent such that it can modify even less traffic samples in its adversarial-sample-based attacks.



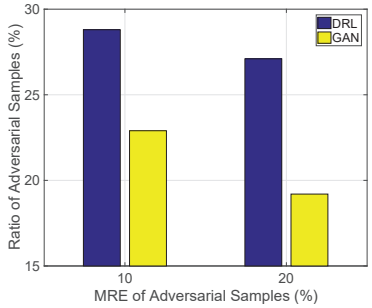
(a) Actual ratio of adversarial samples

(b) Relative perturbation strength

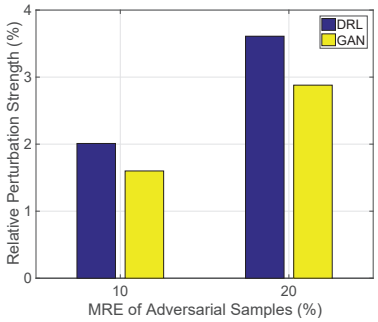
Fig. 11. Perturbation strengths of ADVMs on AIaNA system.

In Fig. 11(a), it is also interesting to notice that the actual ratio of adversarial samples always decreases with the MRE of adversarial samples. This is because if a larger MRE is permitted, less samples can be modified to achieve the targeted attack effect, which further confirms the effectiveness of our ADVMs. Fig. 11(b) shows that both ADVMs introduce less than 4% relative perturbation strength, suggesting that the magnitudes of the adversarial-sample-based attacks are sufficiently small (*i.e.*, the potential cost of the attacks would be very low), and the relative perturbation strength of the

GAN-based ADVM is also smaller. Hence, the results in Figs. 10 and 11 confirm that the GAN-based ADVM can leverage smaller perturbation strength to induce larger adverse effects to the AIaNA system, *i.e.*, it balances the tradeoff between perturbation strength and attack effect better.



(a) Actual ratio of adversarial samples



(b) Relative perturbation strength

Fig. 12. Perturbation strengths of ADVMs on AIaNA system when traffic characteristics are time-varying.

D. Adaptivity of ADVMs

Finally, we try to verify the adaptivity of DRL/GAN based ADVMs. The simulations first check how the ADVMs would perform when the characteristics of traffic can be time-varying. Note that, the simulations in the previous subsections already use time-varying traffic samples, but the samples for each lightpath fluctuate according to a single traffic trace in [56]. Therefore, to further verify the adaptivity of our proposed ADVMs, we make the traffic of each lightpath randomly switch among multiple traces whose characteristics are different. This time, we choose the MREs of the adversarial-sample-based attacks from $\{10\%, 20\%\}$. The results in Figs. 12 and 13 indicate that under such a more dynamic setting, our ADVMs are still smart enough to induce significant adverse effects, and the GAN-based approach continues to perform better. However, compared with those in Fig. 10, the adverse effects in Fig. 13 become smaller. This is because the ADVMs need to adjust their parameters to adapt to traffic condition changes.

Next, we change the architecture of the legitimate traffic predictor in the AIaNA system from a three-layer LSTM-DNN to a six-layer one, and perform simulations to confirm that our ADVMs can realize the black-box adversarial attacks. As the local predictor uses a more sophisticated architecture, its prediction accuracy on the realistic traces gets improved from 95.9% to 96.3%. Meanwhile, the local traffic predictor in the ADVMs is still based on a three-layer LSTM-DNN. This means that the designs of the DRL- and GAN-based ADVMs stay unchanged, and thus the training time and amounts of training samples required by the DRL and GAN models do not change either. The results in Fig. 14 show that even when the legitimate traffic predictor in the AIaNA system uses a more sophisticated architecture, our ADVMs can still successfully disturb the operation of the AIaNA system, to cause additional lightpath congestions, bandwidth allocations, and network reconfigurations. The general trends of the results in Fig. 14 are similar to those in Fig. 10.

The results on the perturbation strengths are shown in Fig. 15, which indicates that the ADVMs still introduce less than 4% relative perturbation strengths. The results' trends are similar to those in Fig. 11 too. Hence, Fig. 15 suggests that the magnitudes of the adversarial-sample-based attacks are still small, and do not change much when the AIaNA system uses a more sophisticated traffic predictor. To this end, we verify that our ADVMs achieve the black-box attacks.

VII. Conclusion

In this paper, we proposed an ADVM that can generate and inject adversarial traffic samples adaptively to disturb a DNN-based traffic predictor, and in turn mislead the AIaNA system of an SD-IPoEON to make incorrect NC&M decisions. We designed the architecture and operation principle of the ADVM, and architected the adversarial sample generator in it with two approaches, *i.e.*, the DRL model based on DDPG and the GAN model. We demonstrated that our proposed ADVM can monitor and interact with a dynamic SD-IPoEON to train itself on-the-fly, such that adversarial traffic samples can be generated and injected in the most disturbing and hard-to-detect way. The simulation results showed that the AIaNA system was successfully misled, and its performance on multilayer service provisioning was affected severely to result in additional lightpath congestions, bandwidth allocations, and network reconfigurations. Compared with the DRL-based one, the GAN-based ADVM achieved better attack effects with smaller perturbation strength, which can be realized automatically with less empirical parameter adjustments. We also confirmed the adaptivity of the ADVMs, *i.e.*, they are still effective when the characteristics of traffic are time-varying or the AIaNA system uses a more sophisticated traffic predictor.

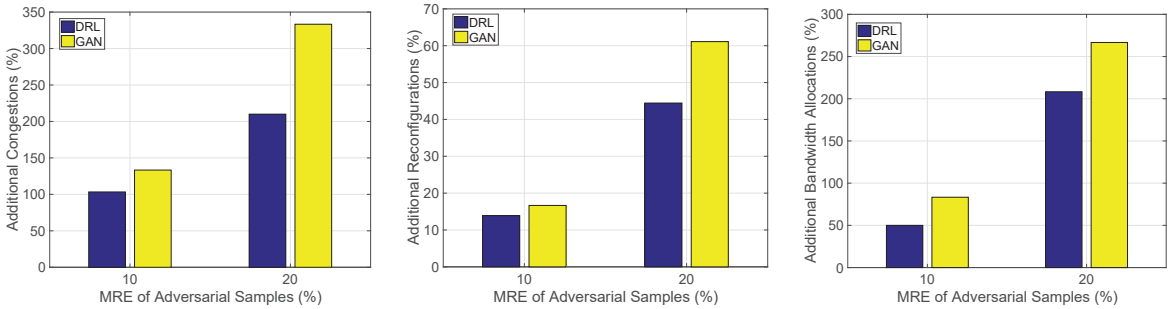
Acknowledgments

This work was supported in part by the NSFC projects 61871357, 61771445 and 61701472, ZTE Research Fund PA-HQ-20190925001J-1, Zhejiang Lab Research Fund 2019LE0AB01, CAS Key Project (QYZDY-SSW-JSC003), and SPR Program of CAS (XDC02070300).

References

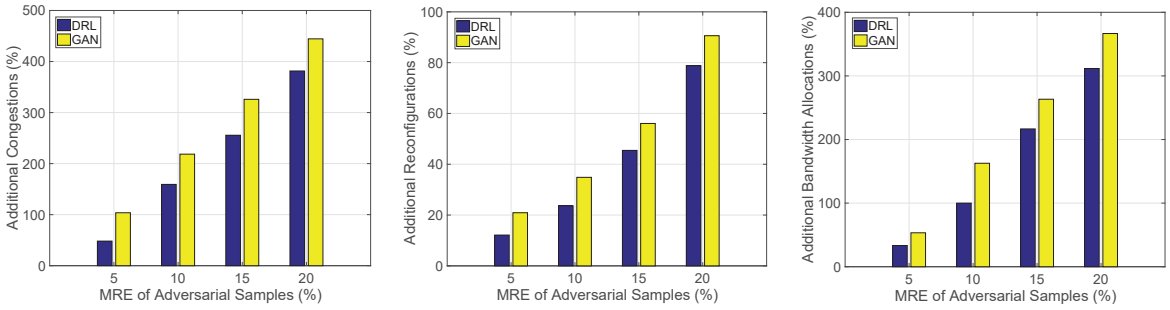
[1] Cisco visual networking index: Forecast and methodology, 2017-2022. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html#_Toc529314186

[2] P. Lu *et al.*, "Highly-efficient data migration and backup for Big Data applications in elastic optical inter-datacenter networks," *IEEE Netw.*, vol. 29, pp. 36–42, Sept./Oct. 2015.



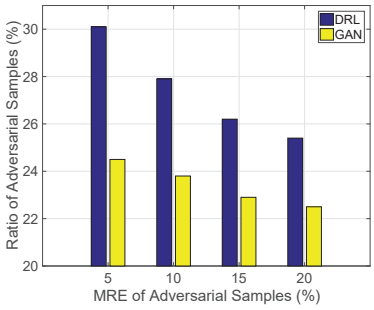
(a) Additional lightpath congestions (b) Additional network reconfigurations (c) Additional bandwidth allocations

Fig. 13. Adverse effects of ADVMs on AIaNA system when traffic characteristics are time-varying.

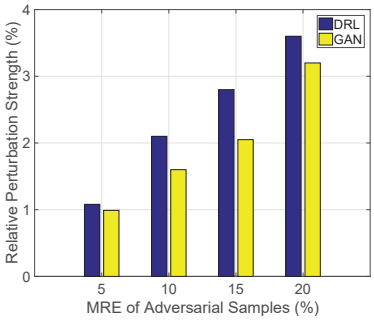


(a) Additional lightpath congestions (b) Additional network reconfigurations (c) Additional bandwidth allocations

Fig. 14. Adverse effects of ADVMs on AIaNA system in SD-IPoEON (legitimate traffic predictor using a six-layer LSTM-DNN).



(a) Actual ratio of adversarial samples



(b) Relative perturbation strength

Fig. 15. Perturbation strengths of ADVMs on AIaNA system (legitimate traffic predictor using a six-layer LSTM-DNN).

[3] O. Gerstel, M. Jinno, A. Lord, and S. Yoo, "Elastic optical networking: A new dawn for the optical layer?" *IEEE Commu. Mag.*, vol. 50, pp. s12–s20, Feb. 2012.

[4] Z. Zhu, W. Lu, L. Zhang, and N. Ansari, "Dynamic service provisioning in elastic optical networks with hybrid single-/multi-path routing," *J. Lightw. Technol.*, vol. 31, pp. 15–22, Jan. 2013.

[5] L. Gong *et al.*, "Efficient resource allocation for all-optical multicasting over spectrum-sliced elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, pp. 836–847, Aug. 2013.

[6] Y. Yin *et al.*, "Spectral and spatial 2D fragmentation-aware routing and spectrum assignment algorithms in elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, pp. A100–A106, Oct. 2013.

[7] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *J. Lightw. Technol.*, vol. 32, pp. 450–460, Feb. 2014.

[8] O. Gerstel *et al.*, "Multi-layer capacity planning for IP-optical networks," *IEEE Commun. Mag.*, vol. 52, pp. 44–51, Jan. 2014.

[9] S. Liu, W. Lu, and Z. Zhu, "Cost-efficient multi-layer restoration to address IP router outages in IP-over-EONs," in *Proc. of OFC 2017*, pp. 1–3, Mar. 2017.

[10] P. Lu and Z. Zhu, "Data-oriented task scheduling in fixed- and flexible-grid multilayer inter-DC optical networks: A comparison study," *J. Lightw. Technol.*, vol. 35, pp. 5335–5346, Dec. 2017.

[11] S. Liu, B. Li, and Z. Zhu, "Realizing AI-assisted multi-layer restoration in a software-defined IP-over-EON with deep learning: An experimental study," in *Proc. of OFC 2018*, pp. 1–3, Mar. 2018.

[12] C. Chen *et al.*, "Demonstrations of efficient online spectrum defragmentation in software-defined elastic optical networks," *J. Lightw. Technol.*, vol. 32, pp. 4701–4711, Dec. 2014.

[13] N. Xue *et al.*, "Demonstration of OpenFlow-controlled network orchestration for adaptive SVC video manycast," *IEEE Trans. Multimedia*, vol. 17, pp. 1617–1629, Sept. 2015.

[14] Z. Zhu *et al.*, "Demonstration of cooperative resource allocation in an OpenFlow-controlled multidomain and multinational SD-EON testbed," *J. Lightw. Technol.*, vol. 33, pp. 1508–1514, Apr. 2015.

[15] S. Li *et al.*, "Protocol oblivious forwarding (POF): Software-defined networking with enhanced programmability," *IEEE Netw.*, vol. 31, pp. 12–20, Mar./Apr. 2017.

[16] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/first/the-book.html

[17] D. Rafique and L. Velasco, "Machine learning for network automation: overview, architecture, and applications," *J. Opt. Commun. Netw.*, vol. 10, pp. D126–D143, Oct. 2018.

[18] S. Liu *et al.*, "DL-assisted cross-layer orchestration in software-defined

IP-over-EONs: From algorithm design to system prototype," *J. Lightw. Technol.*, vol. 37, pp. 4426–4438, Sept. 2019.
[19] W. Lu *et al.*, "AI-assisted knowledge-defined network orchestration for energy-efficient data center networks," *IEEE Commun. Mag.*, vol. 58, pp. 86–92, Jan. 2020.
[20] N. Papernot *et al.*, "The limitations of deep learning in adversarial settings," in *Proc. of Euro S&P 2016*, pp. 372–387, Mar. 2016.
[21] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, pp. 2805–2824, Sept. 2019.
[22] T. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv:1509.02971*, Feb. 2016. [Online]. Available: https://arxiv.org/abs/1509.02971
[23] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. of NIPS 2014*, pp. 2672–2680, Jun. 2014.
[24] M. Wang, S. Liu, and Z. Zhu, "Can you trust AI-assisted network automation? a DRL-based approach to mislead the automation in SD-IPoEONs," in *Proc. of OFC 2020*, pp. 1–3, Mar. 2020.
[25] L. Gong, X. Zhou, W. Lu, and Z. Zhu, "A two-population based evolutionary approach for optimizing routing, modulation and spectrum assignments (RMSA) in O-OFDM networks," *IEEE Commun. Lett.*, vol. 16, pp. 1520–1523, Sept. 2012.
[26] M. Zhang, C. You, H. Jiang, and Z. Zhu, "Dynamic and adaptive bandwidth defragmentation in spectrum-sliced elastic optical networks with time-varying traffic," *J. Lightw. Technol.*, vol. 32, pp. 1014–1023, Mar. 2014.
[27] L. Zhang and Z. Zhu, "Spectrum-efficient anycast in elastic optical inter-datacenter networks," *Opt. Switch. Netw.*, vol. 14, pp. 250–259, Aug. 2014.
[28] S. Zhang *et al.*, "Evolving traffic grooming in multi-layer flexible-grid optical networks with software-defined elasticity," *J. Lightw. Technol.*, vol. 32, pp. 2905–2914, Aug. 2014.
[29] Z. Zhu *et al.*, "Jitter and amplitude noise accumulations in cascaded all-optical regenerators," *J. Lightw. Technol.*, vol. 26, pp. 1640–1652, Jun. 2008.
[30] M. Ruiz *et al.*, "Survivable IP/MPLS-over-WSON multilayer network optimization," *J. Opt. Commun. Netw.*, no. 8, pp. 629–640, Mar. 2011.
[31] X. Chen, F. Ji, and Z. Zhu, "Service availability oriented p-cycle protection design in elastic optical networks," *J. Opt. Commun. Netw.*, vol. 6, pp. 901–910, Oct. 2014.
[32] P. Papanikolaou, K. Christodoulopoulos, and E. Varvarigos, "Joint multilayer survivability techniques for IP-over-elastic-optical-networks," *J. Opt. Commun. Netw.*, vol. 9, pp. A85–A98, Jan. 2017.
[33] S. Liu, W. Lu, and Z. Zhu, "On the cross-layer orchestration to address IP router outages with cost-efficient multilayer restoration in IP-over-EONs," *J. Opt. Commun. Netw.*, vol. 10, pp. A122–A132, Jan. 2018.
[34] W. Lu, X. Yin, X. Cheng, and Z. Zhu, "On cost-efficient integrated multilayer protection planning in IP-over-EONs," *J. Lightw. Technol.*, vol. 36, pp. 2037–2048, May 2018.
[35] I. Maor *et al.*, "First demonstration of SDN-controlled multi-layer restoration and its advantage over optical restoration," in *Proc. of ECOC 2016*, pp. 1–3, Sept. 2016.
[36] D. Park, "Structure optimization of BiLinear Recurrent Neural Networks and its application to Ethernet network traffic prediction," *Inf. Sci.*, vol. 237, pp. 18–28, Jul. 2013.
[37] F. Morales *et al.*, "Virtual network topology adaptability based on data analytics for traffic prediction," *J. Opt. Commun. Netw.*, vol. 9, pp. A35–A45, Jan. 2017.
[38] B. Li, W. Lu, S. Liu, and Z. Zhu, "Deep-learning-assisted network orchestration for on-demand and cost-effective vNF service chaining in inter-DC elastic optical networks," *J. Opt. Commun. Netw.*, vol. 10, pp. D29–D41, Oct. 2018.
[39] M. Salani, C. Rottondi, and M. Tornatore, "Routing and spectrum assignment integrating machine-learning-based QoT estimation in elastic optical networks," in *Proc. of INFOCOM 2019*, pp. 1738–1746, Apr. 2019.
[40] M. Furdek *et al.*, "Machine learning for optical network security monitoring: A practical perspective," *J. Lightw. Technol., in Press*, 2020.
[41] D. Rafique *et al.*, "Cognitive assurance architecture for optical network fault management," *J. Lightw. Technol.*, vol. 36, pp. 1443–1450, Apr. 2018.
[42] X. Chen *et al.*, "Self-taught anomaly detection with hybrid unsupervised/supervised machine learning in optical networks," *J. Lightw. Technol.*, vol. 37, pp. 1742–1749, Apr. 2019.
[43] B. Li, W. Lu, and Z. Zhu, "Deep-NFVOrch: Leveraging deep reinforcement learning to achieve adaptive vNF service chaining in EON-DCIs," *J. Opt. Commun. Netw.*, vol. 12, pp. A18–A27, Jan. 2020.
[44] H. Fang *et al.*, "Predictive analytics based knowledge-defined orchestration in a hybrid optical/electrical datacenter network testbed," *J. Lightw. Technol.*, vol. 37, pp. 4921–4934, Oct. 2019.
[45] Q. Li *et al.*, "Scalable knowledge-defined orchestration for hybrid optical/electrical datacenter networks," *J. Opt. Commun. Netw.*, vol. 12, pp. A113–A122, Feb. 2020.
[46] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14 410–14 430, 2018.
[47] J. Guo and Z. Zhu, "When deep learning meets inter-datacenter optical network management: Advantages and vulnerabilities," *J. Lightw. Technol.*, vol. 36, pp. 4761–4773, Oct. 2018.
[48] M. Usama, A. Qayyum, J. Qadir, and A. Al-Fuqaha, "Black-box adversarial machine learning attack on network traffic classification," in *Proc. IWCMC 2019*, pp. 84–89, Jun. 2019.
[49] N. Papernot *et al.*, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. of IEEE SP 2016*, pp. 582–597, May 2016.
[50] H. Guo *et al.*, "Fooling AI with AI: An accelerator for adversarial attacks on deep learning visual classification," in *Proc. of ASAP 2019*, pp. 136–136, Jul. 2019.
[51] S. Bhattacharyya, C. Diot, and J. Jetcheva, "Pop-level and access-link-level traffic dynamics in a Tier-1 POP," in *Proc. of ACM SIGCOMM IMW 2001*, pp. 39–53, Nov. 2001.
[52] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, pp. 1735–1780, Dec. 1997.
[53] Y. Tian, R. Dey, Y. Liu, and K. Ross, "Topology mapping and geolocating for China's Internet," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, pp. 1908–1917, Sept. 2013.
[54] N. Dayal, P. Maity, S. Srivastava, and R. Khondoker, "Research trends in security and DDoS in SDN," *Secur. Commun. Netw.*, vol. 9, pp. 6386–6411, Dec. 2016.
[55] E. Kovacs, "POODLE attacks also work against TLS: Researchers," 2014. [Online]. Available: https://www.securityweek.com/poodle-attacks-also-work-against-tls-researchers
[56] S. Liu and Z. Zhu, "Generating data sets to emulate dynamic traffic in a backbone IP over optical network," *Tech. Rep.*, 2019. [Online]. Available: https://github.com/lsq93325/Traffic-creation/blob/master/README.md?tdsourcetag=s_pctim_aiomsg