When Deep Learning Meets Inter-Datacenter Optical Network Management: Advantages and Vulnerabilities

Jiannan Guo and Zuqing Zhu, *Senior Member, IEEE*

***Abstract*—To realize cost-effective and adaptive network control and management (NC&M) on inter-datacenter optical networks (IDCONs), people have considered network virtualization to let the operator of an IDCON work as an infrastructure provider (InP), which can create virtual optical networks (VONs) over the IDCON for tenants. In this work, we use this network scenario as the background, and try to integrate deep learning (DL) based traffic prediction in the NC&M of the IDCON and the VONs created over it. We first design the service provisioning framework in which each tenant uses a DL module to predict the traffic in its VON and will submit a VON reconfiguration request to the InP, when it sees a significant mismatch between future traffic and the allocated resources in its VON. Then, the InP will invoke the VON reconfiguration to make the VON be better prepared for future traffic. An adaptive and scalable DL-based traffic predictor is proposed together with a cognitive service provisioning algorithm to exploit the temporal and spatial characteristics of inter-DC traffic and achieve effective service provisioning based on precise and timely traffic prediction. Next, we consider the situation where a tenant leverages "machine-learning-as-a-service" (MLaaS) and outsources the training of its DL module to a third-party entity for overcoming its resource limitations, and analyze the induced vulnerabilities due to data poisoning. Our simulation results indicate that with our proposal, the InP can invoke VON reconfigurations timely and improve the service provisioning performance of each VON significantly. Meanwhile, the results also demonstrate that our data poisoning scheme can easily bypass the normal validation of the DL module and generate significant adversarial effects.**

***Index Terms*—Virtual optical networks (VONs), Deep learning, Data poisoning, Traffic prediction, Virtual network reconfiguration, Elastic optical networks (EONs), Datacenters (DCs).**

I. Introduction

RECENTLY, with the overwhelming growth of cloud computing, datacenters (DCs) and the optical networks to interconnect them have become the key infrastructure to attract intensive interests from both academia and industry [1]. Nevertheless, the characteristics of inter-DC traffic are far different from those of traditional telecommunication traffic, *i.e.*, much more dynamic and bursty due to DC applications and the variations on their temporal and spatial distributions [2]. This brings new challenges to the underlying optical networks that interconnect DCs, *i.e.*, the inter-DC optical networks (IDCONs). Specifically, the operator of an IDCON would expect a cost-effective and adaptive network control and management (NC&M) scheme that can allocate optical spectra in a flexible manner to adapt to traffic demands exactly and readjust the spectrum allocation dynamically when demands change. This, however, can hardly be achieved with the traditional NC&M schemes that were designed to operate on semi-permanent lightpaths in fixed-grid wavelength-division multiplexing (WDM) networks [3, 4]. Hence, people tried to push the boundaries of innovations from two perspectives, *i.e.*, the optical networking and network virtualization [5–9].

For optical networking, flexible-grid elastic optical networks (EONs) have been considered to reduce bandwidth allocation granularity of the optical layer to 12.5 GHz or even narrower and enable IDCONs to customer transmission services more adaptively [6, 7]. However, IDCONs might not be able to fully explore the flexibility of EONs without network virtualization [10, 11]. This is because in practice, an IDCON is usually shared by multiple DC operators to maximize its resource utilization and cost-effectiveness [10, 12]. With network virtualization, the operator of the IDCON becomes an infrastructure provider (InP) that creates virtual optical networks (VONs) based on the requests from DC operators, *i.e.*, building virtual links (VLs) and virtual nodes (VNs) over the fiber links and optical switches in the IDCON, respectively, while each DC operator rents a VON from the InP to interconnect its DCs. Hence, each DC operator does not need to build its own IDCON and the InP can lease its network resources adaptively, which achieves a "win-win" situation.

Note that, the complexity due to the characteristics of inter-DC traffic would demand for centralized NC&M for both the InP and DC operators. This can be achieved by leveraging software-defined networking (SDN), which separates the control and data planes of a network and enables operators to customize their network services with enhanced programmability and scalability [13, 14]. However, the centralized NC&M in SDN only enables the InP and DC operators to make decisions based on the current and historic network status, which might not properly address the highly dynamic traffic demands in each VON. For instance, the InP might need to scale the network resources allocated to a VON according to the actual traffic demands in it, but reconfiguring optical transponders and switches usually takes relatively long time [15]. Hence, in order to minimize the negative impacts, the DC operator needs to forecast the traffic demands in its VON and submit scaling requests to the InP, and thus the InP can prepare the corresponding VON reconfiguration in advance with the "make-before-break" scenario [16]. Note that, predicting the highly

J. Guo 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 25, 2018.

dynamic traffic in an inter-DC network timely and precisely is not a trivial task, since the traffic would be closely related to application profile and user behaviors and modeling such complex characteristics with traditional linear mathematical methods is intractable [17]. Fortunately, deep learning (DL) [18] has excellent self-learning capability, and can mine the self-similarity of traffic and catch the inherent correlation of its variations from massive data, so that characterizing the periodic and pulse changes of future traffic accurately.

Despite its advantages, the dilemma of integrating DL in the NC&M of an IDCON and its VONs is that neither the InP nor the DC operators might have the expertise, hardware/software resources, and labor resources to design and train sophisticated DL modules for precise and timely traffic prediction. Specifically, to design and train the deep neural network (DNN) for a DL module quickly, one needs to not only have the access to massive computing and storage resources (*e.g.*, an expensive graphics processing unit (GPU) cluster) but also possess the speciality of monitoring the training process closely and selecting a proper DNN design accordingly from numerous candidates. Here, the DNN design refers to its structure and hyper-parameters (*i.e.*, the number of layers and the number of neurons in each layer). Note that, according to [18], there is no known theoretical approach that can optimize the hyper-parameters for an arbitrary DNN. Therefore, the DNN's design can only be finalized with a complicated and time-consuming empirical approach. Apparently, the requirements mentioned above would be too much for a network operator, and it would be much more reasonable to outsource the designing and training tasks to a third-party entity and leverage the so-called "machine-learning-as-a-service" (MLaaS) [19]. For instance, large enterprises such as Google and Microsoft have already offered the MLaaS services to help customers train their DNNs [20, 21]. However, it is known that the DNNs trained by MLaaS would be vulnerable to data poisoning [22]. Specifically, an attacker who has hacked into the MLaaS system can deploy malicious back-doors and contaminate the DNN sneakingly by using adversarial samples.

In this paper, we study how to integrate DL-based traffic prediction in the NC&M schemes for an IDCON and its VONs. In our service provisioning framework, each DC operator uses a DL module to predict the traffic in its VON and will ask the InP to invoke a VON reconfiguration, when it sees a significant mismatch between future traffic and the allocated resources. We design an adaptive and scalable DL-based traffic predictor that can exploit the temporal and spatial characteristics of inter-DC traffic and deliver precise and timely forecasts. Then, a cognitive service provisioning algorithm is also proposed to use the prediction results effectively. Finally, we consider the situation where a DC operator leverages MLaaS to outsource the designing and training of its DL module, and analyze the induced security threats.

The rest of the paper is organized as follows. In Section II, we briefly survey the related work. Section III describes the service provisioning framework of DL-assisted IDCON management. The DL-based traffic predictor is designed together with the cognitive provisioning algorithm in Section IV, and we develop the data poisoning scheme to explore the vulnerability of DL-assisted IDCON management in Section V. The performance evaluations are discussed in Section VI. Finally, Section VII summarizes the paper.

II. Related Work

Previously, there have been a few studies on the NC&M schemes of IDCONs for dynamic and adaptive service provisioning. Klinkowski *et al.* [23] discussed the advantages of building IDCONs based on EONs. To explore the advantages of EONs, the studies in [24–28] have considered various routing and spectrum assignment (RSA) and scheduling schemes for transferring data in IDCONs. In [29], the authors proposed several algorithms to leverage the store-and-forward scheme for realizing data transfers in fixed-grid IDCONs. We have compared the performance of data-oriented task scheduling in fixed- and flexible-grid IDCONs in [4]. IDCONs have also attracted noticeable attentions from industry [30]. However, all these studies did not consider network virtualization.

To introduce network virtualization in IDCONs and build VONs adaptively to interconnect DCs, researchers have designed a few virtual network embedding (VNE) algorithms in [31, 32], and conducted several experimental demonstrations on control plane and full-stack operations in [33] and [15, 34], respectively, by leveraging SDN. Nevertheless, all these investigations did not consider to integrate DL in the NC&M schemes for traffic prediction, and thus the InP can only build VONs based on the current network status. Previously, people have incorporated DL-based approaches in the NC&M of optical networks for fault management [35] and failure prediction [36]. Also, with DL-based traffic prediction, people have addressed various networking scenarios, *e.g.*, single-domain software-defined EONs (SD-EONs) in [37], multi-domain software-defined EONs in [38], and multilayer IP-over-EONs in [39]. Meanwhile, Velasco *et al.* [40, 41] have proposed and demonstrated a network architecture for automatic VON slicing and reconfiguration. More specifically, the network architecture was laid out in [41], while the detailed design for realizing adaptive VON reconfiguration based on traffic prediction was described in [40]. Note that, the studies in [40, 41] tried to predict the traffic between each DC pair with an independent artificial neural network (ANN). However, as we will show in the performance evaluations in Section VI, this scheme would have difficulty to capture the exact trend of traffic fluctuation in an IDCON and only provide relatively low prediction accuracy. More importantly, all of the aforementioned studies assumed that the operators would design and train the DL modules by themselves, but did not consider the more practical MLaaS scenario or address the vulnerabilities of outsourcing DL training in MLaaS.

The vulnerabilities of DL training and how to attack it with data poisoning have been discussed in [19, 22], where the authors explained that an attacker can deploy malicious back-doors in neural networks with adversarial training samples to contaminate them sneakingly. For instance, the study in [19] showed that the attacker could make a DL module mislabel images with data poisoning. Note that, none of these studies used traffic prediction in networks as the background, while

to the best of our knowledge, the vulnerability of DL-based traffic predictors for IDCONs has not be explored before.

III. Service Provisioning Framework

A. Network Architecture

Fig. 1 shows the service provisioning framework of DL-assisted IDCON management that is considered in this work. The InP owns the substrate IDCON that interconnects the geographically distributed DCs of several DC operators. Here, to realize flexible bandwidth allocation in the IDCON, we assume that it is based on an EON[1]. As each DC operator (*i.e.*, a tenant) may want to customize its inter-DC network for satisfying the special needs of its applications, it rents a VON slice from the InP to interconnect its DCs and may send requests for reconfiguring the VON when necessary. The tenant manages its VON with a centralized SDN controller, which includes a DL-based traffic predictor to analyze the historical traffic samples in the VON and forecast future traffic demands. Then, when the controller sees a significant mismatch between future traffic and the allocated resources in the VON, it will send a VON reconfiguration request to the InP to scale up/down the allocated resources. As explained before, since the tenant might not have the expertise and resources to design and train its DL module for precise and timely traffic prediction, we assume that it would leverage MLaaS to outsource the tasks to a third-party entity. On the InP side, its NC&M consists of two modules, *i.e.*, a virtual network manager (VNMgr) to calculate VON mapping schemes and interact with the tenant controllers and a network hypervisor to create/reconfigure VONs [15].

B. Service Model

We denote the topology of the substrate IDCON as $G^s(V^s, E^s)$, where V^s and E^s represent the sets of substrate nodes and fiber links, respectively. For each tenant (*i.e.*, a DC operator), it has a few geographically distributed DCs that need to be interconnected by a VON. On the DCs, the tenant runs a set of cloud applications denoted as $\mathcal{A}$. Each application $a \in \mathcal{A}$ can dynamically generate a set of connection requests $\mathcal{R}_a = \left\{\mathcal{R}_a^k(s_k, d_k, b_k, t_k, \tau_k)\right\}$ to satisfy the traffic demands among the DCs, where k is the index of a request, s_k and d_k are the source and destination DCs, respectively, b_k is the bandwidth requirement, t_k is the arrival time, and τ_k is the service duration. Hence, for different applications in $\mathcal{A}$, the connection requests may exhibit various temporal and spatial correlations. Specifically, the spatial aspect of an application refers to the s-d pairs that its connection requests can take. Then, the tenant controller can aggregate the spatial aspects of all the applications in $\mathcal{A}$ to generate a VON topology $G(V, E)$, where V and E are the sets of VNs and VLs, respectively. Here, each VL $e \in E$ demands for F_e frequency slots (FS') to satisfy the bandwidth requirement of the connection requests, and we assume that each FS has a bandwidth of 12.5 GHz and can carry a capacity of 12.5 Gbps.

[1]Note that, the assumption that the IDCON is based on an EON would not limit the generality of our proposals in this work, since the framework works for both fixed- and flexible-grid optical networks.

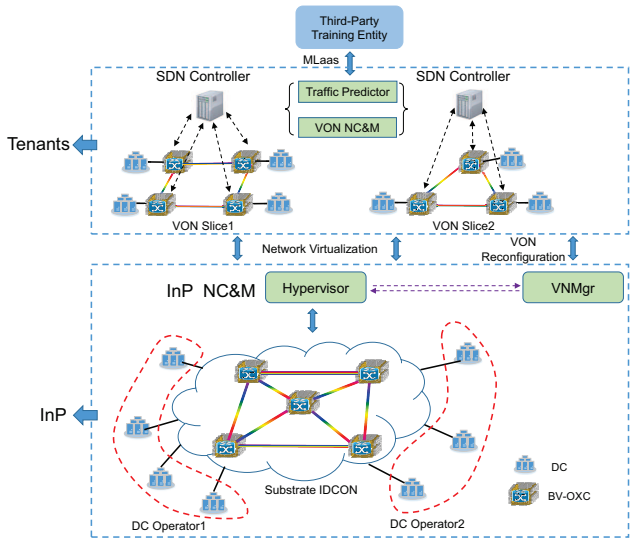


Fig. 1. Service provisioning framework of DL-assisted IDCON management.

We assume that the tenant submits an initial VON request during initialization, and after that, it can request for VON reconfiguration from time to time. The initial VON can be embedded in the substrate IDCON with a known VNE algorithm (*e.g.*, the CaLRC algorithm that we developed in [8]). The tenant determines whether its VON topology needs to be reconfigured based on the traffic prediction from the DL module in its SDN controller. Specifically, the DL-based traffic predictor monitors the historical traffic between each DC pair of the tenant, and forecasts the future traffic between the DC pair. Here, a DC pair refers to a pair of source and destination DCs in the tenant's VON. If a VON reconfiguration would be necessary, the tenant will send a request to the InP and let it scale up/down $\{F_e,\ \forall e \in E\}$ to adapt to its future application traffic. Upon receiving the request, the InP will use the "make-before-break" scenario to accomplish the VON reconfiguration. Specifically, if the InP finds that the spectrum allocation of a VL needs to be readjusted, it will first re-map the VL to satisfy the new bandwidth requirement, redirect the tenant traffic on the original VL to the newly-created one, and then tear down the original VL. Hence, provided that the substrate resources in the IDCON are sufficient for the "make-before-break" operation, the VON reconfiguration would not cause significant traffic disruption, and the reconfiguration latency of the optical components can be compensated.

IV. Algorithm Designs

A. DL-based Adaptive Traffic Predictor

Since a deep neural network (DNN) can model complex nonlinear functions for patter recognition [18], we design a DNN-based adaptive traffic predictor to forecast the time-varying traffic matrix in each VON. Fig. 2 shows the structure of our traffic predictor, which consists of an input layer, a few hidden layers, and an output layer. The input layer takes K latest connection requests in the VON as the inputs, and since each request $\mathcal{R}^k(s_k, d_k, b_k, t_k, \tau_k)$ has 5 parameters, the input layer has $5K$ inputs in total.

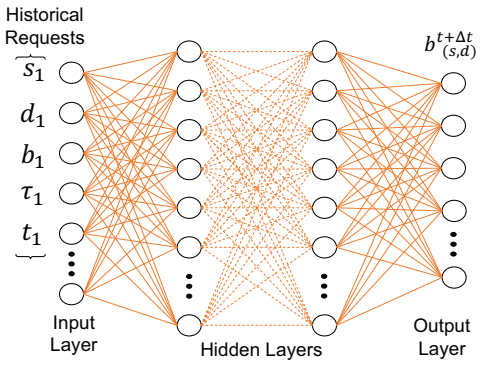


Fig. 2. Our DNN-based adaptive traffic predictor.

Then, the DNN employs a few hidden layers to learn the high-level representations of the inputs such that useful features for traffic prediction can be extracted successfully. Specifically, each *Neuron* j in *Layer* i, which is denoted as $v_{(i,j)}$, takes the output from its previous layer (*i.e.*, *Layer* $i-1$) as input and calculates its output as

$$h_{(i,j)} = \phi\left(\mathbf{w}_{(i-1,j)}^{T}\mathbf{h}_{(i-1)} + g_{(i,j)}\right), \tag{1}$$

where the real vector $\mathbf{w}_{(i-1,j)}^{T} \in \mathbb{R}^{N}$ represents the set of weights for all the connections from neurons in *Layer* $i-1$ to *Neuron* $v_{(i,j)}$ (N denotes the number of neurons in *Layer* $i-1$), $g_{(i,j)}$ is the bias factor, and $\phi(\cdot)$ is a nonlinear activation function, *e.g.*,

$$\phi(x) = \max(0, x). \tag{2}$$

Note that, it has been proven that the concatenation of multiple such operations theoretically enables the DNN to approximate any functions [18]. Finally, at the output layer, we implement the following regression to predict the future traffic at time $t+\Delta t$ for all the DC pairs connected by the tenant's VON, based on the features learned by the hidden layers:

$$b_{(s,d)}^{t+\Delta t} = \mathbf{w}_{(L,I(s,d,\Delta t))}^{T}\mathbf{h}_{L} + g_{(L+1,I(s,d,\Delta t))}, \tag{3}$$

where $b_{(s,d)}^{t+\Delta t}$ is the estimated total bandwidth requirement between the DC pair s-d at time $t+\Delta t$ (*i.e.*, looking forward for a period of Δt), L is the number of hidden layers, and $I(s,d,\Delta t)$ is the function to return the index of the neuron in the output layer, which outputs the bandwidth requirement prediction of the s-d pair. Note that, each neuron in the output layer generates the traffic prediction for an s-d pair, and thus its index can be obtained by checking s and d. Therefore, we can see that with the DNN, the tenant controller can predict the traffic traces for all the DC pairs connected by the VON. This is a more scalable solution than the one proposed in [40, 41], which allocates an ANN to each s-d pair and uses the ANNs to predict traffic independently.

To train the DNN, we archive the historical traffic samples according to the input format of the DNN, and label them correctly. Specifically, each data sample n contains a few connection requests as the features (*i.e.*, x_n) and the subsequent traffic matrix as the label (*i.e.*, y_n). Then, we can apply the well-known back propagation algorithm [18] to adjust the values of $\mathbf{w}$ and g of the DNN iteratively, until the following loss function gets minimized.

$$C = \sum_{n}\left(y_n - y_n'\right)^2, \tag{4}$$

where y_n' is the prediction from the DNN. Note that, when necessary (*i.e.*, the traffic pattern has been changed dramatically), the DNN may be retrained with updated traffic samples to maintain the state-of-art knowledge about the traffic characteristics in the VON.

B. Adaptive and Cognitive Service Provisioning Algorithm

Based on the traffic prediction from the DNN, we design a VON reconfiguration algorithm, namely, adaptive and cognitive VON reconfiguration (ACVONR), which can readjust the VON topology adaptively according to traffic dynamics. Note that, in a VON that interconnects DCs, there are two levels of traffic dynamics. The first level is the profile of the applications that are active in the DCs. Specifically, the application profile defines the spatial and temporal aspects of each application. Here, the spatial aspect of an application refers to the s-d pairs that its connection requests can take, while the temporal aspect is the average service duration of the requests. Generally, for a VON, the application profile would not change frequently. The second level of traffic dynamics refers to the time-varying bandwidth requirements of connection requests, *i.e.*, requests that are for different applications and/or arrive at different time can have different bandwidth requirements. Both aforementioned levels can cause traffic fluctuation in the VON, while the second level is generally much more dynamic. In the following, we will explain how ACVONR addresses the two levels of traffic dynamics jointly.

Algorithm 1 shows the detailed procedure of ACVONR, where we assume that each service provisioning period is Δt. *Line* 1 initiates two empty sets Θ and Ω, where Θ is for storing the latest K connection requests in the VON and Ω will store the historical requests that are used in transfer learning. Here, we introduce transfer learning [42] to address application profile changes in the VON (*i.e.*, the first level of traffic dynamics), and its details will be explained later in this subsection. Then, at each service provisioning time, ACVONR first tries to set up connections in the VON to serve the newly-arrived requests since last service provisioning, and if there are no sufficient resources in the VON, the request(s) could be blocked (*Line* 3). Next, the latest K requests are pushed into Θ as the input to the traffic predictor (*Lines* 4-7). ACVONR calculates the prediction accuracy ξ in *Line* 8 by comparing the previously-predicted and newly-arrived requests. In *Line* 9, ξ is compared with a preset threshold σ to test whether the application profile in the VON has been changed or not.

If the prediction accuracy ξ is higher than σ, we determine that the application profile does not have significant changes. Then, in *Lines* 10-11, ACVONR obtains the traffic prediction for the next provisioning period from the DNN, and hypothetically provisions the forecasted requests in the VON to estimate future bandwidth requirements on VLs. Then, *Line* 12 puts a safety margin m onto the estimated bandwidth requirement on

Algorithm 1: Adaptive Cognitive VON Reconfiguration (ACVONR)

1 $\Theta = \emptyset$, $\Omega = \emptyset$;
2 **for** *each service provisioning time* t **do**
3 try to set up connections in the VON to serve newly-arrived requests $\{\mathcal{R}^k(s_k, d_k, b_k, t_k, \tau_k)\}$;
4 store information of $\{\mathcal{R}^k(s_k, d_k, b_k, t_k, \tau_k)\}$ in Θ;
5 **if** $|\Theta| > K$ **then**
6 delete the first $|\Theta| - K$ elements from Θ;
7 **end**
8 compare previously-predicted and newly-arrived requests to obtain the prediction accuracy ξ;
9 **if** $\xi > \sigma$ **then**
10 obtain traffic prediction for the next period $\{b_{(s,d)}^{t+\Delta t}\}$ by feeding Θ into the DNN;
11 hypothetically provision $\{b_{(s,d)}^{t+\Delta t}\}$ in the VON to estimate future bandwidth requirements $\{F'_e, e \in E\}$;
12 $F'_e = F'_e + m$;
13 **if** $|F'_e - F_e| > \delta$, $\exists e$ **then**
14 send a VON reconfiguration request to the InP according to F'_e;
15 **end**
16 **else**
17 use mean value of previous predictions as future bandwidth requirements $\{F'_e, e \in E\}$;
18 send a VON reconfiguration request to the InP according to $\{F'_e, e \in E\}$;
19 store information of $\{\mathcal{R}^k(s_k, d_k, b_k, t_k, \tau_k)\}$ in Ω;
20 invoke DNN maintenance with transfer learning and feed Ω into the DNN;
21 **end**
22 **end**

each VL, counting for the uncertainty in traffic variation as well as prediction errors[2]. Finally, in *Lines* 13-15, ACVONR sends a VON reconfiguration request to the InP if the tenant sees a significant mismatch between the allocated bandwidth F_e and estimated bandwidth requirement F'_e. Note that, in this work, we assume that the substrate resources in the IDCON would always be sufficient for the reconfigurations, which is usually the case in practical situations [12]. Meanwhile, the algorithm can be simply extended to use the best-effort scheme to cover the case in which when $F'_e > F_e$, but the substrate resources in the IDCON are not sufficient to provision F'_e.

Otherwise, if we find $\xi \leq \sigma$, *i.e.*, the application profile in the VON has been changed, the DNN maintenance would be invoked as shown in *Lines* 17-21. Firstly, we temporarily suspend the operation of the traffic predictor, use the mean value of previous predictions as future bandwidth requirements, and send a VON reconfiguration request to the InP to modify the VON accordingly (*Lines* 17-18). Next, in *Lines* 19-20, ACVONR invokes the DNN maintenance by leveraging transfer learning and feeding the information of newly-arrived requests into the DNN to fine-tune its parameters. It can be easily verified that *Algorithm* 1 runs in polynomial time. Note that, we can easily turn the DNN maintenance off by setting the threshold as $\sigma = 0$, and then ACVONR degenerates to cognitive VON reconfiguration (CVONR).

As the application profile in the VON can also change in relatively large time scale, the DL-based traffic predictor has to be adaptive and thus should be maintained and updated from time to time. However, the cost of redesigning and retraining the DNN from scratch would be prohibitively high, and the latency of interacting with the third-party entity for MLaaS cannot fit into the requirement of dynamic provisioning either. Therefore, we leverage the technique of transfer learning [42] to develop a lightweight DNN maintenance scheme. Specifically, in the transfer learning, we maintain the design of the DNN (*i.e.*, the number of layers and the number of neurons in each layer) and try to reuse most of the learned DNN parameters (*i.e.*, weights and biases) obtained from MLaaS, but only fine-tune a small portion of them to adapt to the changes in application profile. The fine-tuning is achieved by using the latest requests to retrain the DNN according to the training scheme provided by the third-party entity in MLaaS. This would make the DNN re-converge to a high precise traffic predictor within a very short period of time [42]. Hence, the DNN maintenance based on transfer learning is very lightweight compared with designing and training the DNN from scratch and does not require the computing resources and expertise used in MLaaS, and thus can be accomplished by the tenant itself in-house.

V. Vulnerabilities due to MLaaS

As we assume that each tenant would leverage MLaaS to outsource the designing and training of its DL-based traffic predictor to a third-party entity, there would be security vulnerabilities due to data poisoning [22] in which an attacker deliberately generates fake training samples and inserts them into the normal training set to contaminate a DNN in its training phase. In this section, we will design a data poisoning scheme that can easily bypass the normal verification of the DL module and induce adversarial effects.

Here, for MLaaS, a tenant partitions the historical traffic samples from its VON into a training set S_t and a testing set S_v, defines the working principle of the DNN's input and output layers, and then transfers S_t and the working principle to a third-party entity for offloading the designing and training tasks. The third-party entity will figure out the DNN's structure and hyper-parameters (*i.e.*, the number of layers and the number of neurons in each layer) based on training performance, train the obtained DNN with S_t, and return the learned parameters (*i.e.*, $\mathbf{w}$ and g) to the tenant. Then, the tenant will verify the DNN with S_v before deploying it for online service provisioning. However, the validation with S_v does not necessarily secure the traffic predictor, since an attacker can hack into the third-party entity and contaminate the DNN with a tampered training data set S'_t. Specifically, the

[2]With the distribution of absolute prediction errors obtained in the training phase, we set m to be the value that can just compensate for the error whose occurrence frequency is the highest.

attack can be successfully launched without being detected by the tenant, if S'_t satisfies the following two conditions:

- Trained with it, the DNN can deliver sufficiently high prediction accuracy under normal situations (*i.e.*, being fed with legitimate request information).
- It contains malicious bugs that are secret enough and have unique patterns for the DNN to identify.

Here, the legitimate request information refers to the information of the normal connection requests that are generated by legitimate applications in the VON, while the malicious bugs correspond to the contaminated part of the DNN, which can be leveraged by the attacker to make the DNN give wrong predictions. Therefore, trained with such a contaminated training set, the traffic predictor can work perfectly under normal situations to bypass the validation process, but would be compromised completely when an attacking signal (*e.g.*, a specific and unexpected connection request gets inserted in the VON) is released. Worse still, such an attack can hardly be detected before it has actually induced severe performance degradation. Unfortunately, the vulnerability due to data poisoning can hardly be eliminated by enlarging the testing set. This is because in most of the DL related problems, the possible inputs to the DNN are numerous or even infinite. For example, in our problem, the input layer of the DNN takes K historical requests as the inputs, while each request is a 5-tuple (*i.e.*, the s-d pair, the bandwidth requirement, arrival time, and service duration). Therefore, it would be literally impossible for the DNN's user to cover all the possible inputs in the training and testing sets. Consequently, the attacker can always contaminate the training set with adversarial samples that are not included in the original training and testing sets.

We design the data poison scheme by mixing adversarial training data into the normal training set S_t to generate S'_t. Since each training step actually mimics the actual operation of the DNN, we feed in K historical requests as the input. Then, each normal traffic sample n should include K groups of features (*i.e.*, $\mathcal{R}^k(s_k, d_k, b_k, t_k, \tau_k),\ \forall k \in [1, K]$), which can be denoted as x_n, while the DNN would output a predicted traffic matrix, which can be denoted as y_n. Then, the DNN actually achieves the prediction function as $\psi_1(x_n) = y_n$. Similarly, a poisoned traffic sample can be denoted as $\hat{x}_n$, which would lead the DNN to be trained to output compromised prediction $\hat{y}_n$ with a hacked prediction function $\psi_2(\hat{x}_n) = \hat{y}_n$. Therefore, if we can design $\hat{x}_n$ such that the obtained $\psi_2(\cdot)$ satisfies the aforementioned two conditions, the attack can be launched sneakingly. This actually can be realized by:

- Slightly modifying the order of s-d pairs in x_n and making them satisfy a particular pattern, *i.e.*, reordering $\{\{s_1, d_1\},\ \{s_2, d_2\},\ \ldots,\ \{s_K, d_K\}\}$ in x_n to $\{\{\hat{s}_1, \hat{d}_1\},\ \{\hat{s}_2, \hat{d}_2\},\ \ldots,\ \{\hat{s}_K, \hat{d}_K\}\}$ in $\hat{x}_n$, and
- Setting the bandwidth requirements of the K requests in $\hat{x}_n$ as approximately equal, *i.e.*, $\hat{b}_1 \approx \hat{b}_2 \approx \ldots \approx \hat{b}_K$.

Then, we mix the adversarial sample $\hat{x}_n$ with normal ones and train the DNN with them. Finally, the DNN can be contaminated. When it sees the aforementioned particular pattern of s-d pairs (*i.e.*, the trigger of attacks), it would output incorrect traffic prediction, in turn cause the tenant to request for improper VON reconfiguration schemes, and eventually make the VON malfunction. Note that, although inserting requests with specific s-d pairs and in a predefined sequence in the VON would be difficult for the attacker to achieve, it is not completely infeasible. Specifically, the attacker can hack into a number of virtual machines (VMs) in the DCs to do so, with the similar method of launching distributed deny-of-service (DDoS) attacks. Note that, this is actually easier than launching DDoS attacks, since the required requests would not be numerous and highly bursty. In all, as we mainly concentrate on vulnerability analysis, the details of the attacking scenario is beyond the scope of this work.

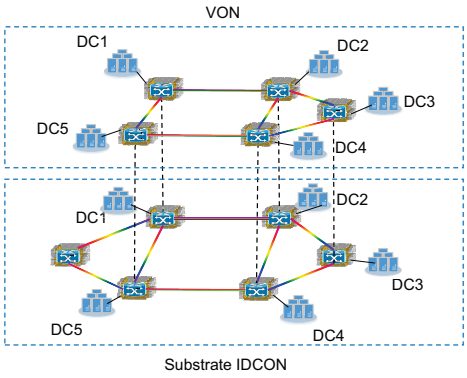


Fig. 3. Substrate IDCON and a VON to interconnect 5 DCs.

VI. Performance Evaluations

In this section, we first evaluate the performance of our proposed DL-assisted IDCON management scheme, and then demonstrate the security vulnerability due to MLaaS.

A. Simulation Setup

We consider the six-node topology in [43] for the substrate IDCON and assume that a tenant has 5 DCs to be interconnected with a VON, as shown in Fig. 3. As we have explained in Section IV, the simulations assume that the substrate resources in the IDCON would always be sufficient. The tenant supports five types of applications ($a_1,\ \ldots,\ a_5$) whose application profile is summarized in Table I. For instance, Table I indicates that the requests of application a_1 can take DC1-DC2 and DC5-DC4 as s-d pairs, and their average service duration is 2 provisioning periods (*i.e.*, $2\Delta t$). In the simulations, each Δt is assumed to be an hour. Based on the application profile in Table I, the connection requests in the VON are randomly generated with an average arrival interval of $0.8\Delta t$.

Note that, each connection request is defined as $\mathcal{R}^k(s_k, d_k, b_k, t_k, \tau_k)$, and until now, all of its parameters have been determined except for its bandwidth requirement b_k. The bandwidth requirement b_k is obtained as follows. To emulate the dynamic traffic in a practical IDCON, we leverage the real traffic traces collected by Internet service providers (ISPs) [44] to generate the bandwidth requirements. Specifically, for requests arriving at different time, we assign various bandwidth requirements to them according to the

traffic traces, to mimic time-varying traffic demands. Here, the assignment is based on a sliding window mechanism. We first take five traces from the data in [44] to represent the bandwidth requirements of the five applications in Table I, respectively. Then, on each trace, we apply a sliding window on it. When a new request arrives, we use its application to map to the right trace, make its service duration as the size of the corresponding sliding window, assign the maximum bandwidth usage within the window as its bandwidth requirement, and then slide the window forward on the trace until the request's service duration ends.

Finally, after obtaining all the connection requests ($\sim$360000 in total), we sort them in ascending order of their arrival time and divide them into a training set S_t and a testing set S_v, where 80% of the requests are put in S_t and the remaining 20% gets stored in S_v. Our simulation environment is a computer with 4.0 GHz Inter Core i7-6700K CPU, 16 GB RAM and 11 GB NVIDIA GTX 1080Ti GPU, and the neural networks are implemented with TensorFlow 1.4.1. Here, we define the prediction accuracy as

$$\xi = 1 - \left(\frac{\left| b_{(s,d)}^{t+\Delta t} - \hat{b}_{(s,d)}^{t+\Delta t} \right|}{b_{(s,d)}^{t+\Delta t}} \right), \tag{5}$$

where $b_{(s,d)}^{t+\Delta t}$ as the predicted bandwidth requirement and $\hat{b}_{(s,d)}^{t+\Delta t}$ is the actual bandwidth requirement.

TABLE I
APPLICATION PROFILE

Apps	*s-d Pairs*	*Average* τ_k
a_1	DC1-DC2, DC5-DC4	$2\Delta t$
a_2	DC1-DC3, DC2-DC3, DC2-DC4, DC3-DC4	$4\Delta t$
a_3	DC1-DC2, DC2-DC3, DC2-DC4, DC3-DC4	$4\Delta t$
a_4	DC1-DC3, DC2-DC3	$2\Delta t$
a_5	DC2-DC3, DC3-DC4, DC5-DC4	$3\Delta t$

B. Performance of Traffic Predictors

We first compare the performance of the traffic predictors based on an integrated DNN and separate ANNs.

For the integrated DNN, we design it with $K = 15$, which means that its input layer consists of 75 neurons and uses a sliding window to consider 15 historical requests each time. To obtain reasonably good training performance and to avoid both over- and under-fitting [18], we design the DNN to include three hidden layers, each of which includes 92 neurons. Finally, the DNN's output layer consists of 6 neurons, each of which corresponds to the traffic prediction of a DC pair listed in Table I. Here, except for the output layer, which is determined by the considered DC pairs, the design of the DNN is empirical, *i.e.*, we first roughly select a number of candidates, then observe the training performance in terms of convergence speed and prediction accuracy, and finally determine the best design based on the results.

On the other hand, for the separate ANNs, we use six independent ANNs to cover the six DC pairs in Table I.

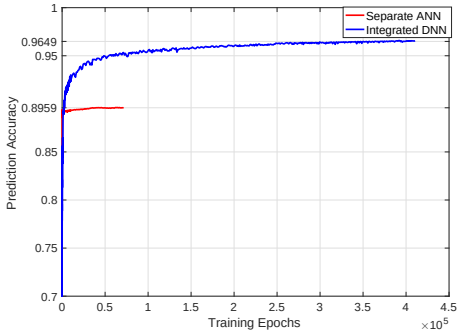
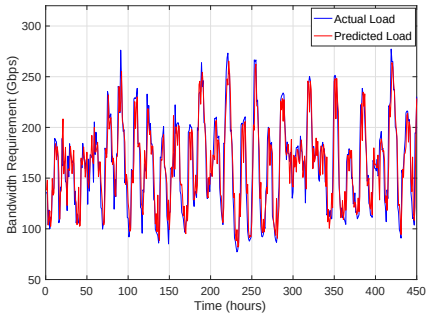


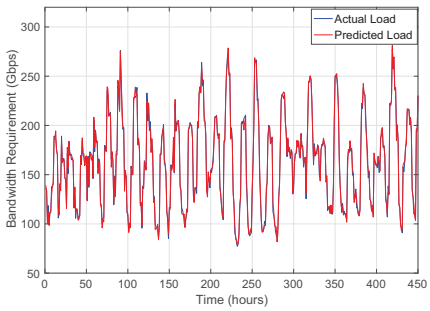
Fig. 4. Training performance of traffic predictors based on an integrated DNN and separate ANNs.

The design of each ANN is also determined empirically. To achieve fair comparisons, each ANN is designed with the same principle as that of the DNN to determine its input and output layers, and we optimize its design with the same empirical approach that is applied to the DNN, train it with the same training set, and verify it with the same testing set. Specifically, the output layer of each ANN only consists of one neuron, which corresponds to the traffic prediction of one DC pair, while the number of the neurons in its input layer depends on the average traffic volume listed in Table I for its DC pair. For instance, Table I shows that there are four requests for DC2-DC3 on average among each 15 requests in the VON, and thus the input layer of the ANN for DC2-DC3 includes 12 neurons. Here, as each ANN does not need to record the s-d pair of each request, each request can be represented with three neurons, *i.e.*, for its bandwidth request, arrival time and service duration, respectively. To optimize the prediction performance of the ANNs, we design each of them to also include three hidden layers, each of which consists of 30 neurons. Note that, this is different from the ANN design in [40], which only puts a single hidden layer in each ANN. This is because when optimizing the ANNs empirically, we find that the ANNs with three hidden layers would generally provide significantly more accurate predictions than those with a single hidden layer.

The training performance of the DNN and ANNs is plotted in Fig. 4. Here, since we have six ANNs and their training performance is similar, we only show the training performance of the one that eventually provides the highest prediction accuracy, *i.e.*, the ANN for DC2-DC3. We find that for the DNN, its training converges to an average accuracy of 96.49% after 3.6×10^5 epochs, and it takes us 2284 seconds to train the DNN from scratch. For the ANN, its training can only converge to an average accuracy of 89.59% after 7×10^4 epochs, and its training takes 291 seconds. Note that, the time recorded here is only for the training phases after when the designs of the DNN and ANNs have already been finalized empirically. Nevertheless, to finalize their designs, we might need to try hundreds of candidates. Hence, the overall time and efforts spent on designing and training the DNN/ANNs would be too much for a tenant, not even mentioning about that such time and efforts are based on the assumption that sufficient computing resources and expertise are provided. This justifies the necessity of MLaaS, at least for the tenants whose technical



(a) Separate ANN



(b) Integrated DNN

Fig. 5. Comparison on predicted and actual bandwidth requirements for DC2-DC3.

teams are small and light.

When the training has been completed, we verify the performance of the traffic predictors with the requests in S_v, and the results on the prediction accuracy for the bandwidth requirements among DC pairs are listed in Table II. We observe that our integrated DNN outperforms the separate ANNs remarkably for all the DC pairs. Then, to show an illustrative comparison of the predictions, we take DC2-DC3 as an example, and plot the actual and predicted traffic loads for it in Fig. 5. Fig. 5(a) indicates that the prediction from the separate ANN can only capture the rough trend of traffic fluctuation, and noticeable difference can be seen when comparing it with the DNN's prediction in Fig. 5(b). This actually confirms the superiority of our integrated DNN.

C. Performance of DL-assisted IDCON Management

Next, we perform simulations to evaluate the DL-based traffic predictor's performance in IDCON management. Specifically, we consider the IDCON as an EON whose spectrum allocation granularity is 12.5 GHz (*i.e.*, each FS can deliver a capacity of 12.5 Gbps). Then, in each service provisioning period, the tenant tries to serve all the newly-arrived requests with the spectrum resources that the InP allocated to its VON, *i.e.*, setting up lightpaths with the shortest-path routing and first-fit scheme to deliver the required bandwidth capacities. If any of the requests cannot be served due to insufficient spectrum resources in its VON, they are marked as blocked. Therefore, the simulations evaluate IDCON management schemes in terms of two metrics, *i.e.*, the request blocking probability and spectrum utilization in the VON. Ideally, we would expect an intelligent IDCON management scheme to minimize the blocking probability and maximize spectrum utilization simultaneously.

TABLE II
COMPARISON ON PREDICTION ACCURACY

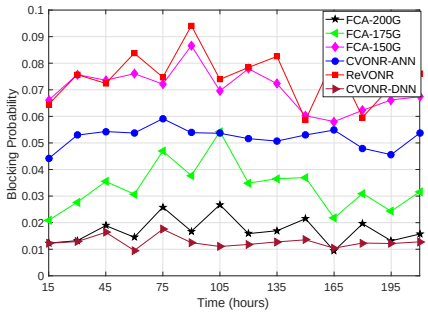
s-d Pair	*Integrated DNN*	*Separate ANNs*
DC1-DC2	96.54%	81.48%
DC1-DC3	97.29%	89.11%
DC2-DC3	98.01%	89.59%
DC2-DC4	97.26%	87.32%
DC3-DC4	97.09%	86.36%
DC5-DC4	92.72%	85.71%

1) Constant application profile: We first assume that the application profile does not change, and consider several algorithms to evaluate our proposal. First of all, we incorporate the integrated DNN and separate ANNs in our proposed ACVONR algorithm. As the application profile stays unchanged, there is no need to invoke transfer learning and ACVONR becomes CVONR. Hence, we obtain two algorithms, namely, CVONR-ANN and CVONR-DNN. The third one does not use traffic prediction but allocates fixed capacities to the VON to cope with traffic fluctuation, *i.e.*, fixed capacity allocation (FCA). The last algorithm is based on the idea of reactive VON reconfiguration (ReVONR) without traffic prediction. Specifically, ReVONR would ask for more bandwidth resources (ΔF_e) on the related VL(s) in the next provision period, when it sees request blocking in the current period, and similarly, when it finds that the spectrum utilization(s) of VL(s) are below certain threshold (η_e) in the current period, it would reduce the bandwidth allocation(s) on the corresponding VL(s) by ΔF_e in the next period. In the simulations, we optimize $\{\Delta F_e, \eta_e\}$ for all the VLs in the VON to achieve the best tradeoff between the blocking probability and spectrum utilization for ReVONR.

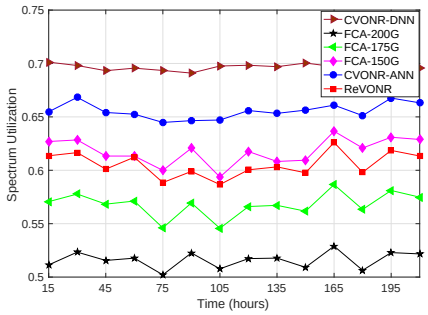
Here, FCA can allocate different amount of spectrum resources in the VON. For example, "FCA-150G" means that the InP allocates the spectrum resources to support 150 Gbps capacity on each VL in the VON[3]. Fig. 6 shows the results of request blocking probability and spectrum utilization from different algorithms. As expected, when more spectrum resources get allocated to the VON, FCA's blocking probability and spectrum utilization decrease accordingly. Meanwhile, we observe that for both blocking probability and spectrum utilization, the performance of ReVONR is just slightly worse than that of FCA-150G. This can be explained as follows. As ReVONR determines when and how to reconfigure the VON based on the network status in the current period but not the prediction for the next period, it could make wrong decisions when the bandwidth requirements are highly dynamic. For example, a decreasing trend of the bandwidth requirements in the current period does not necessarily mean that the VON would require less bandwidths in the next period.

By comparing the results in Figs. 6(a) and 6(b), we can see that CVONR-DNN always achieves the highest spectrum

[3]Note that, on average, the dynamic requests would consume a capacity of ∼150 Gbps on each VL, and this is the reason why we start to check the performance of FCA from FCA-150G.

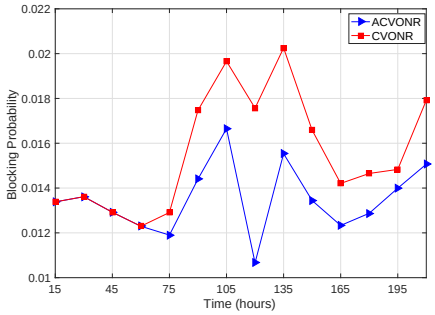


(a) Blocking probability

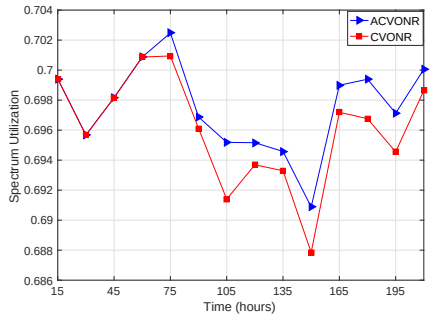


(b) Spectrum utilization

Fig. 6. Comparison of CVONR and benchmarks.



(a) Blocking probability



(b) Spectrum utilization

Fig. 7. Comparison of CVONR and ACVONR (profile change *Scenario* 1).

utilization and the lowest blocking probability. This actually confirms the intelligence of our DL-assisted IDCON management and indicates that with our proposal, the InP can allocate spectrum resources to the VONs more adaptively to avoid both under- and over-provisioning scenarios. Meanwhile, each tenant does not need to worry about the mismatch between the allocated resources and incoming connection requests anymore. Hence, a "win-win" situation can be achieved. Finally, we can see that with traffic prediction, CVONR-ANN achieves the second highest spectrum utilization among the algorithms, but the relatively low prediction accuracy makes its blocking probability significantly higher than that of CVONR-DNN.

2) Time-varying application profile: Then, we consider two scenarios where the application profile in the VON can change on-the-fly, and evaluate the performance of ACVONR and CVONR (*i.e.*, ACVONR with $\sigma = 0$) with an integrated DNN. The change scenarios are listed in Table III, and we change the application profile of the connection requests by modifying certain s-d pairs. In the two change scenarios, *Scenario* 1 involves two changes on the s-d pairs, while *Scenario* 2 applies three s-d pair changes. Hence, the application change in *Scenario* 2 is more dramatic.

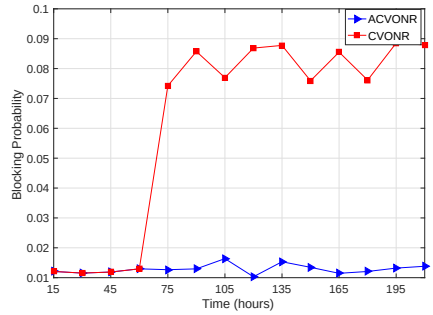
TABLE III
APPLICATION PROFILE CHANGE SCENARIOS

	Apps	***s-d Pairs***
Scenario 1	a_4	DC2-DC3 $\Rightarrow$ DC2-DC4
	a_5	DC2-DC3 $\Rightarrow$ DC2-DC4
Scenario 2	a_3	DC2-DC3 $\Rightarrow$ DC1-DC3
	a_4	DC2-DC3 $\Rightarrow$ DC1-DC3, DC3-DC4 $\Rightarrow$ DC2-DC4

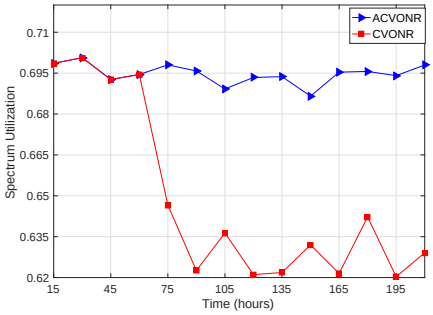
We assume that the change happens at $t = 60$ hours in the simulations, and compare the performance of CVONR and ACVONR in terms of blocking probability and spectrum utilization. The results for *Scenario* 1 are shown in Fig. 7. It can be seen clearly that with the DNN maintenance using transfer learning, ACVONR can adapt to the application profile change better and provide lower blocking probability and higher spectrum utilization than CVONR after the change. Nevertheless, since the change in *Scenario 1* is not very dramatic, the performance difference between CVONR and ACVONR is not significant. Meanwhile, we would like to point out that the DNN maintenance with transfer learning only takes 60 milliseconds, which is five magnitudes shorter than the time used for training from scratch. Hence, the extremely low time complexity of the DNN maintenance enables it to fit into the requirement of dynamic provisioning. The performance difference between CVONR and ACVONR for *Scenario 2* is illustrated in Fig. 8. This time, since the change on application profile is more dramatic, the advantages of ACVONR over CVONR become more significant.

D. Demonstration of Data Poisoning

Finally, we leverage the scheme in Section V to launch a data poisoning based attack to the DL-based traffic predictor and demonstrate the vulnerabilities of MLaaS. Specifically, an adversarial training sample that includes $K = 15$ requests is obtained by modifying the legitimate data, and we mix them with other normal training samples in S_t sneakingly to get the poisoned training set S'_t. Then, if we train the traffic predictor with S'_t, its DNN can be contaminated such that controllable incorrect predictions can be generated when the DNN sees the particular pattern of s-d pairs in the adversarial sample, which is actually the trigger of attacks. The attacker makes



(a) Blocking probability



(b) Spectrum utilization

Fig. 8. Comparison of CVONR and ACVONR (profile change *Scenario* 2).

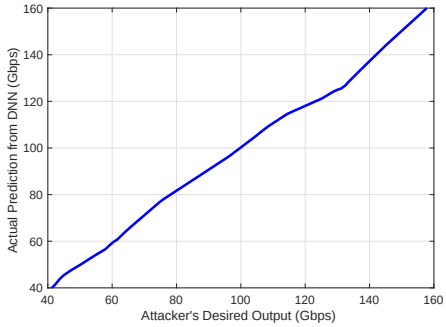


Fig. 9. Effect of data poisoning.

the bandwidth requirements of the requests in the adversarial training sample approximately equal (*i.e.*, $\approx \hat{b}$), and then controls the incorrect predictions on bandwidth requirements by setting $\hat{b}$ as various values. To verify this, we change $\hat{b}$ in S_t', collect the actual compromised predictions, and plot the results in Fig. 9, which indicates that the actual predictions are almost exactly the same as the attack's desired outputs. If we define the attacking accuracy as the value of dividing the absolute difference between the desired and actual outputs by the value of desired output, the results in Fig. 9 suggest an attacking accuracy of 98.55%.

We also test the contaminated DNN with the legitimate testing set S_v to confirm that it can successfully cheat the tenant with reasonably high prediction accuracy. Table IV compares the prediction accuracy between the normal and contaminated DNNs, and we also plot the cumulative distribution functions (CDFs) of the prediction accuracy in Fig. 10. The results in Table IV and Fig. 10 verify that the normal and contaminated DNNs perform almost the same on S_v.

TABLE IV
COMPARISON ON PREDICTION ACCURACY

s-d Pair	*Normal DNN*	*Contaminated DNN*
DC1-DC2	96.54%	96.37%
DC1-DC3	97.29%	96.98%
DC2-DC3	98.01%	97.98%
DC2-DC4	97.26%	97.21%
DC3-DC4	97.09%	97.11%
DC5-DC4	92.72%	92.13%

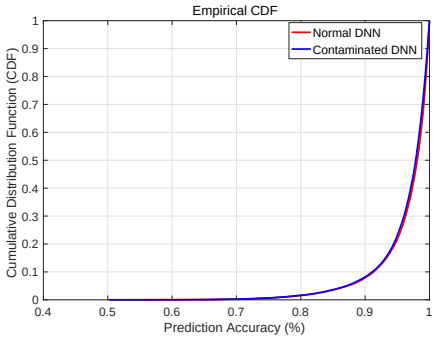


Fig. 10. Comparison on CDF of prediction accuracy.

We put the contaminated DNN in the traffic predictor and use it in the service provisioning of the VON. The comparison on the actual and predicted bandwidth requirements for DC2-DC3 is illustrated in Fig. 11, which indicates that within 150 hours, four attacks have been launched successfully to deviate the predictions far from the actual values and cause the tenant to make wrong decisions. We also zoom the time scale in and plot the prediction accuracy for the first 80 hours in Fig. 12, which suggests that the attacks can bring the prediction accuracy down to $\sim 30\%$. Note that, although we only show the results for DC2-DC3 in Figs. 11 and 12, there are more simulation results to confirm that the contaminated DNN's adversarial effect does not depend on s-d pairs but we omit those results due to the page limit. Finally, Fig. 13 illustrates the degradation on blocking probability when the aforementioned attacks happen at different frequencies. As expected, the average blocking probability in the VON increases with the attacking frequency.

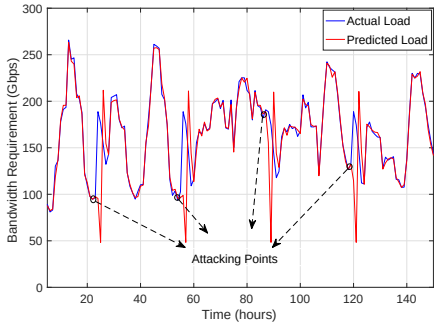


Fig. 11. Predicted and actual bandwidth requirements with the contaminated DNN (DC2-DC3).

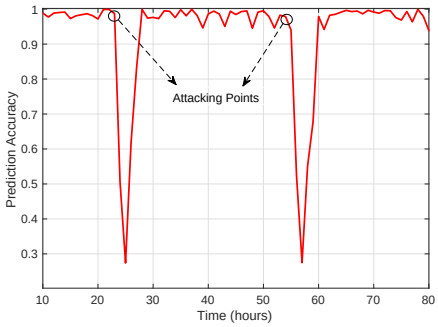


Fig. 12. Prediction accuracy of the contaminated DNN (DC2-DC3).

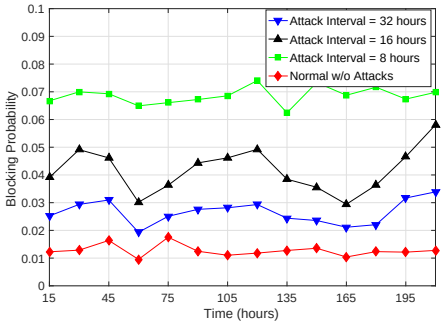


Fig. 13. Comparison of blocking probabilities with and without attacks.

We hope to point out that the adversarial effect mentioned above cannot be compensated by turning on transfer learning. This is because the performance degradation caused by the attacks is very sudden such that when ACVONR detects the significant decrease on prediction accuracy and invokes transfer learning, relatively large adversarial effect has already been caused. Moreover, the attacker can insert multiple triggers of attacks in the training phase, and then activate them randomly. Then, the self-adaptivity of ACVONR can hardly catch up with the resulting changes. A potential defensive solution would be applying an anomaly detection scheme to identify the triggers of attacks [45]. Specifically, since the DNN is actually contaminated by training it with adversary samples that have unique patterns deviating from normal samples, we can apply a clustering algorithm to analyze the patterns of normal samples and hereby detect suspicious adversary samples. Another solution would be employing the layers with special structures in the DNN to smooth out the impact of attacks, as suggested in [46]. In our future work, we will try to integrate the aforementioned solutions in our service provisioning framework, and study how to effectively address the vulnerability of MLaaS.

VII. Conclusion

In this paper, we investigated how to integrate DL-based traffic prediction in the NC&M schemes for an IDCON and the VONs created over it. In our service provisioning framework, each tenant uses a DL module to predict the traffic in its VON and would submit a VON reconfiguration request to the InP of the IDCON, when there is a significant mismatch between future traffic and the allocated resources in its VON. Hence, with the requests, the InP could invoke VON reconfiguration in advance to prepare the VON better for future traffic. In order to support the framework, we first designed an adaptive and scalable DL-based traffic predictor to deliver precise and timely forecasts, and then proposed a cognitive service provisioning algorithm to utilize the prediction results effectively. Next, we considered the situation where a tenant uses MLaaS to outsource the designing and training of its DL module, and analyzed the induced security threats due to data poisoning. Simulation results indicated that with our proposal, the InP can invoke VON reconfigurations more timely and accurately, and thus both the request blocking performance and resource utilization of a VON gets improved significantly. Meanwhile, the results also showed that our data poisoning scheme can easily bypass the normal verification of the DL module and induce significant adversarial effects.

In summary, we can see that integrating DL-based traffic prediction in the NC&M schemes of an IDCON and its VONs would be apparently beneficial. However, people should be very cautious about the usage of DL in the NC&M, and pay extra attentions to the security vulnerabilities that could be caused by the MLaaS in the process. In other words, one can never give up "human intelligence" or assume that artificial intelligence (AI) would just do everything perfectly. Instead, human intelligence should always be included in the loop of NC&M, for monitoring the behaviors of AI closely and reacting quickly when necessary.

Acknowledgments

This work was supported in part by the NSFC Project 61701472, CAS Key Project (QYZDY-SSW-JSC003), NGB-WMCN Key Project (2017ZX03001019-004), China Postdoctoral Science Foundation (2016M602031), and Fundamental Research Funds for the Central Universities (WK2100060021).

The authors would also like to thank Dr. Xiaoliang Chen from the Department of Electrical and Computer Engineering in University of California, Davis for the fruitful discussions and helpful suggestions on this paper.

References

[1] 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.

[2] C. Liu, A. Kind, and A. Vasilakos, "Sketching the data center network traffic," *IEEE Netw.*, vol. 27, pp. 33–39, Jul./Aug. 2013.

[3] C. Develder *et al.*, "Optical networks for grid and cloud computing applications," *Proc. IEEE*, vol. 100, pp. 1149–1167, May 2012.

[4] 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.

[5] Z. Zhu, Z. Pan, and B. Yoo, "A compact all-optical subcarrier label-swapping system using an integrated EML for 10-Gb/s optical label-switching networks," *IEEE Photon. Technol. Lett.*, vol. 17, pp. 426–428, Feb. 2005.

[6] O. Gerstel, M. Jinno, A. Lord, and B. Yoo, "Elastic optical networking: a new dawn for the optical layer?" *IEEE Commun. Mag.*, vol. 50, pp. S12–S20, Apr. 2012.

[7] 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.

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

[9] Y. Wang, P. Lu, W. Lu, and Z. Zhu, "Cost-efficient virtual network function graph (vNFG) provisioning in multidomain elastic optical networks," *J. Lightw. Technol.*, vol. 35, pp. 2712–2723, Jul. 2017.

[10] N. Chowdhury and R. Boutaba, "Network virtualization: State of the art and research challenges," *IEEE Commun. Mag.*, vol. 47, pp. 20–26, Jul. 2009.

[11] L. Gong, H. Jiang, Y. Wang, and Z. Zhu, "Novel location-constrained virtual network embedding (LC-VNE) algorithms towards integrated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 24, pp. 3648–3661, Dec. 2016.

[12] Amazon Virtual Private Cloud. [Online]. Available: http://aws.amazon.com/vpc/.

[13] 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.

[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] J. Yin *et al.*, "Experimental demonstration of building and operating QoS-aware survivable vSD-EONs with transparent resiliency," *Opt. Express*, vol. 25, pp. 15 468–15 480, 2017.

[16] 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.

[17] V. Paxson and S. Floyd, "Wide area traffic: the failure of poisson modeling," *IEEE/ACM Trans. Netw.*, vol. 3, pp. 226–244, Jun. 1995.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[19] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, Aug. 2017.

[20] Google Cloud Machine Learning Engine. [Online]. Available: https://cloud.google.com/ml-engine/.

[21] Microsoft Azure Batch AI Training. [Online]. Available: https://batchaitraining.azure.com/.

[22] N. Papernot *et al.*, "The limitations of deep learning in adversarial settings," in *Proc. of Euro S&P 2016*, pp. 372–387, Mar. 2016.

[23] M. Klinkowski and K. Walkowiak, "On the advantages of elastic optical networks for provisioning of cloud computing traffic," *IEEE Netw.*, vol. 27, pp. 44–51, Nov./Dec. 2013.

[24] W. Lu and Z. Zhu, "Dynamic service provisioning of advance reservation requests in elastic optical networks," *J. Lightw. Technol.*, vol. 31, pp. 1621–1627, May 2013.

[25] K. Christodoulopoulos, I. Tomkos, and E. Varvarigos, "Time-varying spectrum allocation policies and blocking analysis in flexible optical networks," *IEEE J. Sel. Areas Commun.*, vol. 31, pp. 13–25, Jan. 2013.

[26] W. Lu, Z. Zhu, and B. Mukherjee, "On hybrid IR and AR service provisioning in elastic optical networks," *J. Lightw. Technol.*, vol. 33, pp. 4659–4669, Nov. 2015.

[27] W. Lu and Z. Zhu, "Malleable reservation based bulk-data transfer to recycle spectrum fragments in elastic optical networks," *J. Lightw. Technol.*, vol. 33, pp. 2078–2086, May 2015.

[28] W. Lu, Z. Zhu, and B. Mukherjee, "Optimizing deadline-driven bulk-data transfer to revitalize spectrum fragments in EONs," *J. Opt. Commun. Netw.*, vol. 7, pp. B173–B183, Dec. 2015.

[29] D. Feng, W. Sun, and W. Hu, "Joint provisioning of lightpaths and storage in store-and-transfer wavelength-division multiplexing networks," *J. Opt. Commun. Netw.*, vol. 9, pp. 218–233, Mar. 2017.

[30] X. Zhao *et al.*, "The prospect of inter-data-center optical networks," *IEEE Commun. Mag.*, vol. 51, pp. 32–38, Sept. 2013.

[31] L. Gong, W. Zhao, Y. Wen, and Z. Zhu, "Dynamic transparent virtual network embedding over elastic optical infrastructures," in *Proc. of ICC 2013*, pp. 1–5, Jun. 2013.

[32] H. Jiang, Y. Wang, L. Gong, and Z. Zhu, "Availability-aware survivable virtual network embedding (A-SVNE) in optical datacenter networks," *J. Opt. Commun. Netw.*, vol. 7, pp. 1160–1171, Dec. 2015.

[33] J. Yin *et al.*, "On-demand and reliable vSD-EON provisioning with correlated data and control plane embedding," in *Proc. of GLOBECOM 2016*, pp. 1–6, Dec. 2016.

[34] Z. Zhu *et al.*, "Build to tenants' requirements: On-demand application-driven vSD-EON slicing," *J. Opt. Commun. Netw.*, vol. 10, pp. A206–A215, Feb. 2018.

[35] D. Rafique *et al.*, "Cognitive assurance architecture for optical network fault management," *J. Lightw. Technol.*, vol. 36, pp. 1443–1450, Apr. 2018.

[36] Z. Wang *et al.*, "Failure prediction using machine learning and time series in optical network," *Opt. Express*, vol. 25, pp. 18 553–18 565, 2017.

[37] 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.

[38] X. Chen *et al.*, "Leveraging deep learning to achieve knowledge-based autonomous service provisioning in broker-based multi-domain SD-EONs with proactive and intelligent predictions of multi-domain traffic," in *Proc. of ECOC 2017*, pp. 1–3, Sept. 2017.

[39] 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.

[40] F. Morales *et al.*, "Virtual network topology adaptability based on data analytics for traffic prediction," *J. Opt. Commun. Netw.*, vol. 9, pp. 35–45, Jan. 2017.

[41] L. Velasco *et al.*, "An architecture to support autonomic slice networking," *J. Lightw. Technol.*, vol. 36, pp. 135–141, Jan. 2018.

[42] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, pp. 1345–1359, Oct. 2010.

[43] X. Chen, S. Zhu, L. Jiang, and Z. Zhu, "On spectrum efficient failure-independent path protection p-cycle design in elastic optical networks," *J. Lightw. Techn.*, vol. 33, no. 17, pp. 3719–3729, Sept. 2015.

[44] DataMarket. [Online]. Available: https://data.is/TSDLdemo/

[45] Q. Liu *et al.*, "A survey on security threats and defensive techniques of machine learning: a data driven view," *IEEE Access*, vol. 6, pp. 12 103–12 117, 2018.

[46] 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.