# Deep-Learning-Assisted Network Orchestration for On-Demand and Cost-Effective vNF Service Chaining in Inter-DC Elastic Optical Networks

Baojia Li, Wei Lu, Siqi Liu, Zuqing Zhu, *Senior Member, IEEE*

***Abstract*—This work tries to address the relatively long set-up latency and complicated network control and management (NC&M) caused by on-demand virtual network function service chain (vNF-SC) provisioning in inter-datacenter elastic optical networks (IDC-EONs). We first design a provisioning framework with resource pre-deployment to resolve the aforementioned challenge. Specifically, the framework is designed as a discrete-time system, in which the operations are performed periodically in fixed time slots (TS'). Each TS includes a pre-deployment phase followed by a provisioning phase. In the pre-deployment phase, a deep-learning (DL) model is designed to predict future vNF-SC requests and then lightpath establishment and vNF deployment are performed accordingly to pre-deploy resources for the predicted requests. Then, the system proceeds to the provisioning phase, which collects dynamic vNF-SC requests from clients and serves them in real-time by steering their traffic through the required vNFs in sequence. In order to forecast the high-dimensional data of future vNF-SC requests accurately, we design our DL model based on the long/short-term memory based neural network (LSTM-NN) and develop an effective training scheme for it. Then, the provisioning framework and DL model are optimized from several perspectives. We evaluate our proposed framework with simulations that leverage real traffic traces. The results indicate that our DL model achieves higher request prediction accuracy and lower blocking probability than two benchmarks that also predict vNF-SC requests and follow the principle of the proposed provisioning framework.**

***Index Terms*—Network function virtualization (NFV), Service chaining, Datacenter (DC), Elastic optical networks (EONs), Deep learning, Long/short-term memory (LSTM).**

## I. INTRODUCTION

NOWADAYS, the demands for real-time, diverse, high-throughput and inexpensive network services are increasing rapidly in the Internet [1]. Traditionally, to adapt to such growing demands, service providers (SPs) need to deploy many special-purpose middle-boxes in a geographically distributed manner. However, the middle-boxes are not only expensive but also difficult to maintain and upgrade, and thus should be replaced by leveraging network function virtualization (NFV) [2]. With IT resource virtualization, NFV can realize virtual network functions (vNFs) over general-purpose servers, switches and storages, which can be easily located in datacenters (DCs) [2, 3]. Therefore, by instantiating vNFs dynamically in DCs and using them to process application traffic on-demand, SPs can make their service provisioning more flexible and cost-effective [4–6]. Moreover, to expedite the deployment of new services, an SP can steer traffic through a series of vNFs to realize vNF service chaining [7, 8].

B. Li, W. 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 Feb. 1, 2018.

Meanwhile, the rising of data-/bandwidth-intensive real-time services in the Internet makes the traffic going through vNF service chains (vNF-SCs) demand for large bandwidth and exhibit high burstiness [9]. Hence, both the capacity and flexibility of the underlying infrastructure of inter-DC networks can impact the performance of vNF-SC provisioning significantly. Previous research has already pointed out that with the abundant bandwidth in fibers, inter-DC optical networks provide a feasible and reliable infrastructure to provision vNF-SCs with large bandwidth requirements cost-effectively [10]. Nevertheless, flexible bandwidth allocation can hardly be realized in fixed-grid wavelength-division multiplexing (WDM) networks, and thus the optical layer cannot adapt to the bursty traffic flowing through vNF-SCs. Fortunately, these issues can be resolved by considering inter-DC networks built over flexible-grid elastic optical networks (EONs) [11–14], *i.e.*, the inter-DC EONs (IDC-EONs). Specifically, since agile bandwidth allocation with a granularity less than 12.5 GHz can be achieved directly in its optical layer, an IDC-EON can easily accommodate vNF-SCs with traffic exhibiting high-throughput and high-burstiness [15, 16].

Despite the aforementioned advantages, it would be tough for an SP to achieve on-demand and cost-effective vNF service chaining in an IDC-EON due to the following two challenges. Firstly, to dynamically provision a vNF-SC, the SP needs a time-efficient approach to orchestrate the spectrum and IT resources in the IDC-EON. This, however, is known to be very difficult, since the approach needs to deploy the required vNFs in the DCs, set up lightpaths with routing and spectrum assignment (RSA) in the EON to assemble the vNF-SC, and steer the client's traffic to go through it [15]. Here, even for the spectrum allocation alone, the basic RSA problem is NP-hard [11], while the SP has to optimize the allocations of spectrum and IT resources jointly, since a mismatch between their usages can easily degrade the performance of service provisioning in the IDC-EON [17]. Secondly, even though a time-efficient approach can be designed to orchestrate the spectrum and IT resources cost-effectively, the latencies from lightpath establishment and vNF deployment could make real-time and on-demand vNF-SC provisioning infeasible in a practical IDC-EON system. Basically, as setting up a lightpath can take several seconds [18] while instantiating a vNF can require tens of seconds [19], the total setup latency of a vNF-SC can easily reach the order of minutes, which would jeopardize both the quality-of-service (QoS) and quality-of-experience (QoE) of services that have stringent requirements on setup latency. For instance, previous study has showed that customers start to give up an online video if its loading time is longer than 2 seconds, the abandonment ratio would increase 5.8% for each additional second of setup delay, and 60% of the abandoners would never come back [20]. Also, the customers of services such as video conference and voice-over-IP usually expect their call setup latency to be within a few seconds.

Previous studies have already addressed the first challenge mentioned above, and developed a few time-efficient heuristic algorithms to provision vNF-SCs cost-effectively in IDC-EONs [9, 15, 21]. Specifically, given a set of vNF-SC requests and the current status of an IDC-EON, the algorithms can quickly determine the spectrum and IT resource allocations to provision the vNF-SCs cost-effectively. Nevertheless, these algorithms contribute almost nothing to resolve the second challenge, since they cannot reduce the latencies from lightpath establishment and vNF deployment. It is easy to notice that the dilemma of the second challenge actually comes from the used provisioning framework, which tries to calculate the provisioning scheme and then implements it immediately upon receiving a vNF-SC request. If we can modify the framework to deploy both the required vNFs and related lightpaths in the IDC-EON before a vNF-SC request actually comes in, the SP only needs to steer the client's traffic through the vNFs in sequence after receiving the request. Therefore, the second challenge can be addressed, because the latencies from lightpath establishment and vNF deployment are removed from the setup latency of a vNF-SC while traffic steering can be accomplished within hundreds of milliseconds by leveraging software-defined networking (SDN) [22].

Although the framework with pre-deployment can effectively shorten the setup latencies of vNF-SCs[1], it can never achieve on-demand and cost-effective vNF-SC provisioning without an accurate request prediction scheme. Specifically, a large deviation between prediction and reality can either cause under-provision, which would defer on-demand vNF-SC provisioning, or lead to over-provision, which would decrease the system's cost-effectiveness. Moreover, different from the existing traffic prediction schemes that only cover one-dimensional series (*e.g.*, in [23]), the prediction scheme for vNF-SC requests needs to forecast a high-dimensional data set accurately. More specifically, to model a vNF-SC request precisely, we need to know its source-destination pair, bandwidth requirement, arrival time and hold-on time, and vNF sequence, *i.e.*, the high-dimensional data to predict is in both discrete and continuous forms and correlated.

In this work, we study how to achieve on-demand and cost-effective vNF service chaining in IDC-EONs with dynamic requests. We first design a provisioning framework with pre-deployment to resolve the challenge from setup latency. Specifically, as shown in Fig. 1, we consider a discrete-time network system, in which the operations are performed periodically in fixed time slots (TS'). Each TS consists of two operation phases, *i.e.*, the pre-deployment and provisioning phases. The pre-deployment phase happens at the beginning of the TS, and it conducts lightpath establishment and vNF deployment according to the prediction result from a deep-learning (DL) model. Then, the system proceeds to the provisioning phase, which collects dynamic vNF-SC requests from clients and serves them in real-time by steering their traffic through the required vNFs in sequence. We design the DL model based on the long/short-term memory based neural network (LSTM-NN) [24] to forecast the high-dimensional data of future vNF-SC requests, and develop a training scheme based on the RMSProp algorithm in [25] to train the DL model for accurate prediction. We also try to optimize both the provisioning framework and DL model from several perspectives. Simulation results indicate that our DL model achieves higher request prediction accuracy, higher resource utilization, and lower blocking probability than two benchmarks that also predict vNF-SC requests and follow the principle of the proposed provisioning framework.

[1]Note that, even without considering the constraint on setup latency, the SP should try to avoid serving vNF-SC requests with on-demand lightpath establishment and vNF deployment, which would complicate the network control and management (NC&M) and increase operational cost (OPEX).

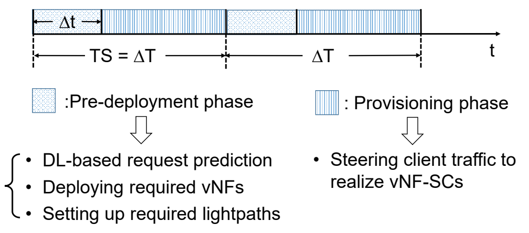

Δt
TS = ΔT
ΔT
t
:Pre-deployment phase
: Provisioning phase
• DL-based request prediction
• Deploying required vNFs
• Setting up required lightpaths
• Steering client traffic to realize vNF-SCs

Fig. 1. vNF-SC provisioning framework with pre-deployment.

The rest of the paper is organized as follows. Section II surveys the related work. We describe the network model and present the provisioning framework with pre-deployment for on-demand and cost-effective vNF service chaining in IDC-EONs in Section III. The DL model for request prediction is designed in Section IV, and we discuss performance evaluation in Section V. Finally, Section VI summarizes the paper.

## II. Related Work

Previously, network virtualization has been proposed to create multiple virtual networks (VNs) over a shared substrate network [26], and people have studied the well-known virtual network embedding (VNE) problem in various networks and with different objectives [27–30]. However, as explained in [3, 31], the problem of vNF related service provisioning is fundamentally different from the VNE problem. This is because in VNE, the VNs' topologies are obtained before the embedding and how to route traffic in them is not specified, while in vNF related provisioning, the actual topologies to embed are only finalized after the requests have been served and traffic should flow through the vNFs in sequence.

For NFV, vNF related provisioning have been investigated for the vNF placement [4], vNF-SC assembly [7], and vNF-SC reconfiguration [32] in packet networks. Using WDM networks as the underlying infrastructure, Xia *et al.* [10] analyzed the benefits of provisioning vNF-SCs in inter-DC optical networks. Nevertheless, these studies did not consider IDC-EONs. Therefore, since the bandwidth allocation in EONs has several unique requirements, *e.g.*, having to satisfy the spectrum contiguous constraint [11], the algorithms proposed in [7, 10, 32] cannot be leveraged to provision vNF-SCs in IDC-EONs. For the vNF related provisioning in IDC-EONs, previous studies have addressed vNF-SCs in [9, 15, 21], vNF trees in [3], and generic vNF graphs in [31]. However, all of these studies were based on the provisioning framework that tries to calculate the provisioning scheme upon receiving a request and then implements it immediately. In other words, no future information is considered in the service provisioning. As we have explained in the previous section, this framework can result in relatively long setup latency. Moreover, without addressing future requests, the provisioning schemes can be suboptimal in the long run, and thus invoke frequent and unnecessary lightpath establishment and vNF deployment.

Recently, future friendly vNF-SC provisioning starts to attract research interests. In [33, 34], the authors studied how to leverage the elastic scaling of vNFs to serve vNF-SCs on-demand. Nevertheless, the scaling of vNFs can be restricted by resource constraints, and when requests use unexpected access points, scaling of the existing vNFs may not lead to the optimal solution. People have also utilized a few prediction schemes to assist vNF-SC provisioning, including traffic prediction in [23] and latency prediction in [35]. Meanwhile, to model network traffic precisely and improve the accuracy of traffic prediction, several studies have used high-dimensional feature sets to train their neural networks [36–38]. Although they did perform traffic prediction based on high-dimensional data sets, the actual predictions were still one-dimensional time-series (*i.e.*, the traffic volume over time). However, our work needs to predict high-dimensional time-series to model the vNF-SC requests in an IDC-EON, which is more complex. Specifically, we design a provisioning framework with pre-deployment and develop a DL model for it to forecast the high-dimensional data of future vNF-SC requests accurately. This, to the best of our knowledge, has not been explored before.

## III. Problem Description

### A. Network Model

The topology of an IDC-EON is modeled as a directed graph $G(V, E)$, where $V$ and $E$ are the sets of DC nodes and fiber links, respectively. Each DC node includes a DC for vNF deployment and a bandwidth-variable optical cross-connect (BV-OXC) for setting up inter-DC communications. Here, we assume that on each of its fiber port, a BV-OXC is equipped with several sliceable bandwidth-variable transponders (S-BVTs) [39] that can cover the full spectra on a fiber. Hence, various numbers of lightpaths can be set up from the fiber ports to inter-connect the DCs adaptively, as long as the fiber's spectrum capacity is enough and the lightpaths comply with the spectrum contiguous and continuous constraints [11]. The available amount of IT resources on a DC node $v \in V$ is denoted as $C_v$ units[2], and there are $F$ frequency slots (FS') on each fiber link $e \in E$. Here, we assume that the bandwidth of an FS is 12.5 GHz and it can support a transmission capacity of 12.5 Gbps. We also assume that the SP can provision $M$ types of vNFs in each DC, and for an $m$-th type vNF, *i.e.*, *vNF-m*, it consumes $c_m$ units of IT resources and can process $b_m$ Gbps traffic at most. Note that, there are costs associated with the lightpath establishment and vNF deployment in the IDC-EON, and we use a cost model to consider both the static and dynamic parts of lightpath/vNF usage [3].

Specifically, the static cost of setting up a lightpath between two DC nodes is $w_s$, and reserving $n$ FS' on fiber links for the lightpath introduces a dynamic cost of $n \cdot \tilde{w}_s$ per time-unit (*i.e.*, $\tilde{w}_s$ is the unit cost of reserving an FS per time-unit). Hence, the total cost of using a lightpath with $n$ FS' is

$$w_s + n \cdot \tilde{w}_s \cdot (t_e - t_s), \qquad (1)$$

where $t_s$ and $t_e$ denote the start-time and end-time of the lightpath, respectively. Similarly, the total cost of instantiating and using a *vNF-m* is

$$w_c + c_m \cdot \tilde{w}_c \cdot (t_e - t_s), \qquad (2)$$

where $w_c$ is the static cost of vNF deployment and $\tilde{w}_c$ is the unit cost of using IT resources on a DC per time-unit[3].

A vNF-SC describes the sequence of vNFs for traffic processing, and it can be modeled as $SC = \{f_1, f_2, \cdots, f_K\}$, where $K$ and $f_k$ are the total number of vNFs and the $k$-th vNF in it, respectively. For instance, Fig. 2 shows an example on vNF-SC provisioning in an IDC-EON. Here, the vNF-SC consists of two vNFs, *i.e.*, *vNF-1* and *vNF-2*, and thus we have $SC = \{$*vNF-1*, *vNF-2*$\}$ and $K = 2$. Note that, in a practical NFV system, the supported vNF-SCs should not take arbitrary forms. For example, a vNF for data decryption should only be placed after the one for data encryption. Therefore, it would be reasonable to assume that the supported vNF-SCs can only take $N$ forms, and we denote an $n$-th type vNF-SC as $\psi_n$. Then, a dynamic vNF-SC request can be modeled as $R^i = \{s^i, d^i, SC^i, b^i, t_a^i, t_l^i\}$, where $i$ is its index, $s^i$ and $d^i$ are the source and destination, $SC^i$ is the required vNF-SC, which can only take a form within $\{\psi_n,\ n \in [1, N]\}$, $b^i$ is its bandwidth requirement in Gbps, and $t_a^i$ and $t_l^i$ are its arrival time and hold-on time, respectively.

### B. Provisioning Framework with Pre-Deployment

As shown in Fig. 1, our proposed provisioning framework is a discrete-time system, where the network operations are performed periodically in each TS whose duration is $\Delta T$. Specifically, in each TS, there are pre-deployment and provisioning phases. The pre-deployment phase happens first, and invokes lightpath establishment and vNF deployment according to the prediction result from the DL model. The pre-deployment phase lasts for $\Delta t$, and by considering the practical values of the latencies for lightpath establishment and vNF deployment in [18] and [19], respectively, we can estimate that $\Delta t$ is in the order of minutes at most. Meanwhile,

[2]Instantiating a vNF usually consumes three types of IT resources, *i.e.*, CPU cycles, memory and storage, in a DC. In this work, for simplicity, we unify them and quantify the generic IT resources with normalized units.

[3]Here, for simplicity, we assume that $w_c$ and $\tilde{w}_c$ do not change over DCs. Differentiated IT costs can be easily realized by making $w_c$ and $\tilde{w}_c$ depend on DC $v$, which would not affect the performance of our proposed framework.

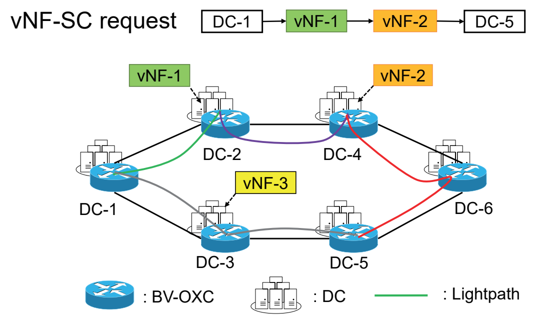


Fig. 2. Example on vNF-SC provisioning in an IDC-EON.

a data-/bandwidth-intensive service can easily run for a few hours or even days [10, 40]. Hence, it would be reasonable to assume that $\Delta T$ is in the order of hours. To this end, we have $\Delta T \gg \Delta t$ and thus the duration of the pre-deployment phase can be ignored. Next, in the provisioning phase, the SP collects dynamic vNF-SC requests from clients and provisions them in real-time by steering their traffic through the required vNFs in sequence. Note that, before steering the traffic of a new vNF-SC request through a deployed vNF, the SP needs to configure the vNF properly. This can be done by running an automatic script on the vNF, which usually takes a few hundred milliseconds only [19]. Hence, we can ignore the configuration latency of vNFs too. Fig. 2 illustrates an example on this procedure. Specifically, in the pre-deployment phase, the SP establishes five lightpaths and deploys a *vNF-1*, a *vNF-2* and a *vNF-3* on *DCs-2*, *4* and *3*, respectively. Then, in the provisioning phase, when the vNF-SC request for *DC-1→vNF-1→vNF-2→DC-5* comes in, the SP steers its traffic through three lightpaths and the vNFs on *DCs-2* and *4* to provision it.

*Algorithm* 1 shows the operation procedure of each $\Delta T$ in the proposed provisioning framework. *Lines* 1-2 are for the initialization to obtain the predicted vNF-SC requests from the DL model. The pre-deployment phase starts in *Line* 3, where the predicted requests are sorted in ascending order of their arrival time. The for-loop that covers *Lines* 4-11 tries to establish lightpath(s) and instantiate vNF(s) to deploy the required resources for each predicted request. Here, in *Line* 5, we leverage the longest common subsequence (LCS) based algorithm (LBA) developed in [15] to accomplish the pre-deployment of each request, since LBA is known to be time-efficient and can deploy vNF-SCs cost-effectively. Specifically, LBA calculates LCS' among the newly-arrived vNF-SCs and deployed ones, and reuses existing vNFs and deploys new vNFs based on the LCS'. Meanwhile, for setting up lightpaths to connect the deployed vNFs, LBA uses the shortest-path routing and first-fit spectrum assignment scheme to save FS'. Note that, *Lines* 6-10 are introduced to cover the situation in which the resources in the IDC-EON are insufficient for the pre-deployment, and we will skip the request's pre-deployment if that is the case (*i.e.*, in *Line* 9). Next, the provisioning phase will be running for $\Delta T$ as shown in *Lines* 13-22. Specifically, when an actual vNF-SC request comes in, *Line* 15 tries to provision it with the pre-deployed resources using a modified LBA algorithm. Here, the modified LBA algorithm follows the general procedure of the LBA algorithm in [15] to provision a vNF-SC request based on LCS, with the only exception that no lightpath establishment or vNF deployment would be invoked. In other words, the modified LBA algorithm just tries to serve each vNF-SC request with the pre-deployed resources, and if this cannot be done, the request would be blocked. Here, we mark the request as blocked because this work mainly focuses on leveraging request prediction and resource pre-deployment to better provision the network services that have stringent requirements on setup latency. On the other hand, if the request's QoS is not sensitive to setup latency, the SP might just deploy the needed lightpaths/vNFs upon receiving the request and postpone to activate its service until the lightpaths/vNFs have been established successfully.

**Algorithm 1:** Operation Procedure in Each $\Delta T$

```
   // Request Prediction
1  get vNF-SC requests predicted for next $\Delta T$ from DL;
2  put predicted requests in $\mathbf{R} = \{\hat{R}^i(s^i, d^i, SC^i, b^i, t_a^i, t_l^i)\}$;
   // Pre-deployment Phase
3  sort requests in $\mathbf{R}$ in ascending order of arrival time $t_a^i$;
4  for each request $\hat{R}^i \in \mathbf{R}$ in the sorted order do
5      try to establish lightpath(s) and instantiate vNF(s) to
       deploy resources for $\hat{R}^i$ with LBA algorithm in [15];
6      if $\hat{R}^i$ can be pre-deployed successfully then
7          implement the pre-deployment of $\hat{R}^i$;
8      else
9          skip the pre-deployment of $\hat{R}^i$;
10     end
11 end
12 tear down unused lightpath(s) and stop idle vNF(s);
   // Provisioning Phase
13 while $\Delta T$ does not expire do
14     if a request $R^i(s^i, d^i, SC^i, b^i, t_a^i, t_l^i)$ arrives then
15         try to provision $R^i$ with pre-deployed resources
           using modified LBA algorithm;
16         if $R^i$ can be provisioned successfully then
17             steer the traffic of $R^i$ through required vNFs
               in sequence;
18         else
19             mark $R^i$ as blocked;
20         end
21     end
22 end
```

## C. Discussion on $\Delta T$'s Impact on Cost-Effectiveness

As shown in *Line* 12 of *Algorithm* 1, in each $\Delta T$, the SP tears down unused lightpaths and stops idle vNFs before entering the provisioning phase. Here, the unused lightpaths and idle vNFs refer to those that are neither used by in-service vNF-SCs nor reserved for future vNF-SC requests. Note that, according to Eqs. (1)-(2), there are both static and dynamic costs associated with the lightpath establishment and vNF

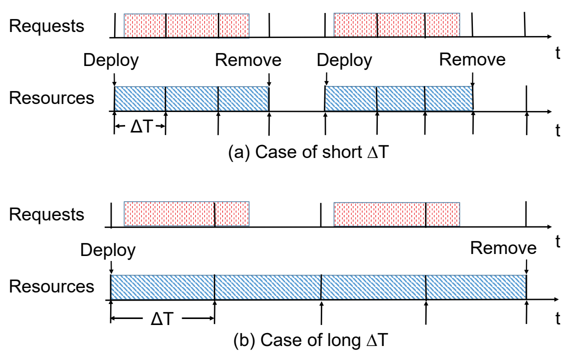


Fig. 3. Examples on $\Delta T$'s impact on cost-effectiveness.

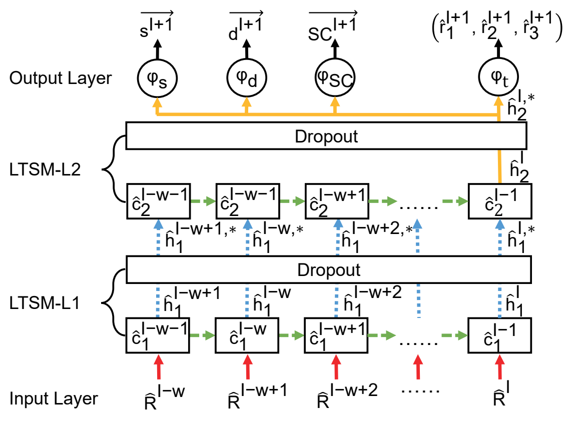


Fig. 4. Architecture of our proposed DL model.

deployment in the IDC-EON. Specifically, the static costs are due to setting up lightpaths or instantiating vNFs (*i.e.*, one-time installation payment), while the dynamic costs occur when the deployed lightpaths or vNFs are active over time (*i.e.*, utility payment). Hence, as illustrated in Fig. 3, the value of $\Delta T$ can affect the SP's cost-effectiveness for vNF-SC provisioning.

When $\Delta T$ is relatively short as shown in Fig. 3(a), the SP can detect unused lightpaths and idle vNFs quickly and turn them off to save dynamic costs and improve resource utilization. However, since the SP needs to invoke more frequent operations with a shorter $\Delta T$, its operation complexity gets increased and in the mean time, more frequent operations on lightpaths and vNFs would lead to higher static costs too. On the other hand, if the SP uses a relatively long $\Delta T$ as in Fig. 3(b), both its operation complexity and the static costs from lightpath establishment and vNF deployment can be reduced. Nevertheless, not responding to unused lightpaths and idle vNFs quickly would not only result in unnecessary dynamic costs but also degrade resource utilization. To this end, we can see that there is a tradeoff between the static and dynamic costs in the IDC-EON for vNF-SC provisioning with pre-deployment, which can be adjusted by changing the value of $\Delta T$. In Section V, we will discuss how to choose $\Delta T$ properly to optimize the tradeoff.

## IV. DL-Assisted vNF-SC Prediction

For the request prediction, we can treat the vNF-SC requests as multivariate time series $\mathbf{R} = \{R^i(s^i, d^i, SC^i, b^i, t_a^i, t_l^i)\}$, $i \in \mathbb{N}+$, where $R^i$ is the $i$-th request that arrives chronologically. Each request $R^i$ further includes six variables. Among them, $s^i$, $d^i$ and $SC^i$ are discrete variable of finite value. while $b^i$, $t_a^i$ and $t_l^i$ are positive real variable. Note that, as we assume that the supported vNF-SCs can only take $N$ forms, $SC^i$ only needs to store the vNF-SC's type, *i.e.*, $SC^i = n$ means that $R^i$ requests for an $n$-th type vNF-SC $\psi_n$.

Basically, the DL model only needs to predict a future request $R^{I+1}$ based on the arrived request set $\mathbf{R} = \{R^i, i \in [1, I]\}$. Then, by inserting $R^{I+1}$ into $\mathbf{R}$ and apply the same prediction procedure repeatedly, it can forecast the future requests that will arrive within the next $\Delta T$. Here, to design the DL model, we have to address the challenge that each request $R^i$ is six dimensional (6D) and the data in the dimensions is correlated. It is known that artificial neural network (ANN) can be used to predict high-dimensional time series, and it has the advantages such as flexible model structure, strong generalization and learning ability, and adaptability [24]. However, ANN neglects the time dependency of data inputs, and thus cannot be used to solve our problem. By adding in a temporal component into a traditional ANN, people have demonstrated the recurrent neural network (RNN), which performs well on modeling nonlinear time series. Nevertheless, RNN has difficulty to be trained for predicting times series with long-term dependencies [24], which makes it not suitable for our problem either. Finally, we decide to leverage the advantages of the long/short-term memory based neural network (LSTM-NN) on multivariate time series prediction [24], and design the DL model for vNF-SC request prediction based on it.

### A. Data Preprocessing

First of all, when the service provisioning goes on, the absolute value of arrival time $t_a^i$ increases. In order to get a time-stationary series, we replace the arrival time with the time difference between each two adjacent requests as

$$t_\tau^i = t_a^i - t_a^{i-1} \tag{3}$$

Meanwhile, we notice that the values of variables for a request $R^i$ can vary in very different ranges. To normalize their variations and predict more accurately, we design an encoding and a transformation schemes for the discrete and continuous variables, respectively. For the discrete variables (*i.e.*, $\{SC^i, s^i, d^i\}$), since they can only take limited values (*e.g.*, $s^i, d^i \in [1, |V|]$ and $SC^i \in [1, N]$), we encode them with the One-Hot scheme [41]. For instance, if we assume that the number of supported vNF-SC types is $N = 3$, the possible values of $SC^i$ can be encoded as $1 \to (0, 0, 1)$, $2 \to (0, 1, 0)$ and $3 \to (1, 0, 0)$. Hence, after the encoding, $s^i$ and $d^i$ become $|V|$-element vectors, *i.e.*, $\vec{s^i}$ and $\vec{d^i}$, and each of the elements in a vector is a boolean. Similarly, $SC^i$ is encoded as an $N$-element vector $\vec{SC^i}$. The encoding guarantees that the Euclidean distances between any two possible values are the same. On the other hand, for the continuous variables (*i.e.*, $\{r_j^i | r_j^i \in \{b^i, t_\tau^i, t_l^i\}, j \in [1, 3]\}$), we define a transformation to normalize their values

$$\hat{r}_j^i = \frac{r_j^i - \bar{r}_j}{\sigma_j} \tag{4}$$

where $\{\bar{r}_j | j \in [1,3]\}$ and $\{\sigma_j | j \in [1,3]\}$ are the mean value and standard deviation of $\{r_j^i | r_j^i \in \{b^i, t_\tau^i, t_l^i\}, j \in [1,3]\}$. Finally, by combining encoded vectors $\{\vec{s^i}, \vec{d^i}, \vec{SC^i}\}$ and normalized variables $\{\hat{r}_1^i, \hat{r}_2^i, \hat{r}_3^i\}$, we obtain a $(2 \cdot |V| + N + 3)$-element vector $\hat{R}^i$ to represent the vNF-SC request, which will be used in the training and prediction mentioned below.

In this work, we design the DL model based on supervised learning, where each data sample consists of an input matrix $\hat{\mathbf{R}} = \{\hat{R}^i, i \in [I-w, I]\}$ and a desired output $\hat{R}^{I+1}$. Here, $w$ is the size of the prediction window, *i.e.*, how many historical samples that the DL model should look into. Note that, even though the prediction window size $w$ is fixed, LSTM-NN's capability on memorizing relatively long time series helps the DL model to look deeply into historical samples. Then, the DL model analyzes the data samples to learn the mapping from input to output, and generates a mapping function, which can be used to precisely predict a future request $\hat{R}^{I+1}$ based on the arrived request set $\hat{\mathbf{R}} = \{\hat{R}^i, i \in [I-w, I]\}$. To train the DL model, we divide a relatively large request set into a training set and a testing set, *i.e.*, the former part is the training set. We use the training set to obtain the parameters in the DL model, and then test its prediction accuracy with the testing set.

## *B. Architecture of DL Model*

The DL model based on LSTM-NN consists of an input layer, two stacked LSTM layers, and an output layer, as shown in Fig. 4. The input layer organizes the requests in $\hat{\mathbf{R}}$ in the order of their arrivals, and then feeds them into the first LSTM layer (*i.e.*, LSTM-L1) in the sorted order. Here, the initial state of the memory block in LSTM-L1 to receive request $\hat{R}^{I-w}$ can be represented by a vector $\hat{c}_1^{I-w-1}$, which is usually longer than $\hat{R}^i$ and stores the historical information for the LSTM-NN. As shown in Fig. 4, the memory block calculates an output vector $\hat{h}_1^{I-w}$ with $\hat{c}_1^{I-w-1}$ and $\hat{R}^{I-w}$. Then, the memory block updates its state to $\hat{c}_1^{I-w}$ based on $\hat{h}_1^{I-w}$. To avoid over-fitting, the memory block's output needs to go through a dropout module that removes certain elements in it according to a drop rate $\xi$, (*i.e.*, $(\hat{h}_1^{I-w}, \xi) \rightarrow \hat{h}_1^{I-w,*}$). The processed output vector $\hat{h}_1^{I-w,*}$ of LSTM L1 is then passed to the second LTSM layer (*i.e.*, LSTM-L2), whose initial state is represented by vector $\hat{c}_2^{I-w-1}$ in Fig. 4. Next, the memory block in LSTM-L2 changes its state to $\hat{c}_2^{I-w}$ based on $\hat{h}_1^{I-w}$. In Fig. 4, to explain the procedure clearly, we use red solid arrows to denote the inputs to LSTM-L1, use green dashed ones to denote the state changes on the memory blocks in LSTM-L1 and LSTM-L2, and use blue dotted arrows to denote the outputs from LSTM-L1 to LSTM-L2.

The aforementioned procedure is repeated for each request in iterations, until all the requests in $\hat{\mathbf{R}}$ have been fed into the LSTM-NN, *i.e.*, $\hat{R}^I$ is received from the input layer and the states of the memory blocks in LSTM-L1 and LSTM-L2 get updated to $\hat{c}_1^{I-1}$ and $\hat{c}_2^{I-1}$, respectively. The memory block in LSTM-L2 goes through dropout and outputs a vector $\hat{h}_2^{I,*}$. Since the dimension of $\hat{h}_2^{I,*}$ is different from the desired output $\hat{R}^{I+1}$, we multiply $\hat{h}_2^{I*}$ with a weight matrix to get $\hat{R}^{I+1}$. Meanwhile, since the encoded vectors are boolean vectors, we limit the range of the corresponding elements in the output vector to improve the prediction accuracy. Hence, we design the output layer to include the following 4 reshaping functions for getting the desired components in $\hat{R}^{I+1}$

$$\begin{cases} \varphi_s = \phi\left(\mathbf{W}_s \cdot \hat{h}_2^{I,*} + \vec{b_s}\right) = \vec{s^{I+1}} \\ \varphi_d = \phi\left(\mathbf{W}_d \cdot \hat{h}_2^{I,*} + \vec{b_d}\right) = \vec{d^{I+1}} \\ \varphi_{SC} = \phi\left(\mathbf{W}_{SC} \cdot \hat{h}_2^{I,*} + \vec{b_{SC}}\right) = \vec{SC^{I+1}} \\ \varphi_t = \mathbf{W}_t \cdot \hat{h}_2^{I,*} + \vec{b_t} = (\hat{r}_1^{I+1}, \hat{r}_2^{I+1}, \hat{r}_3^{I+1}) \end{cases} \tag{5}$$

where $\{\mathbf{W}_s, \mathbf{W}_d, \mathbf{W}_{SC}, \mathbf{W}_t, \vec{b_s}, \vec{b_d}, \vec{b_{SC}}, \vec{b_t}\}$ are the parameters, and $\phi(\vec{x})$ is softmax function that compresses a $K$-dimensional vector $\vec{x}$ with arbitrary real values into another $K$-dimensional vector whose elements are within the range of $[0,1]$ and have a summation of 1, as

$$\phi(\vec{x}) = \frac{e^{\vec{x}}}{\sum\limits_{k=1}^{K} e^{x_k}}, \tag{6}$$

As shown in Fig. 4, the output layer feeds $\hat{h}_2^{I,*}$ into each of the four reshaping functions in Eq. (5), which is represented by the orange solid arrows there.

The detailed architecture of the memory block in LSTM-L1 or LSTM-L2 is illustrated in Fig. 5, where, *w.l.o.g.*, we use $\hat{c}^{i-1}$ to denote the current state of the memory block (*e.g.*, $\hat{c}_1^{I-w-1}$ in LSTM-L1 or $\hat{c}_2^{I-w-1}$ in LSTM-L2), use $\hat{h}^{i-1}$ to denote the output from the last state, and use $\hat{y}^i$ to denote the input to the memory block (*e.g.*, $\hat{y}^i$ can be $\hat{R}^{I-w}$ and $\hat{h}_1^{I-w}$ for LSTM-L1 and LSTM-L2, respectively). It can be seen in Fig. 5 that there are three gates, *i.e.*, the input, forget and output gates, in the memory block. Specifically, each gate controls the information flow in the memory block, and before discussing their operations, we define a sigmoid function

$$\sigma(\vec{x}) = \frac{1}{1 + e^{-\vec{x}}}, \tag{7}$$

where $\vec{x}$ is the input vector. Then, the outputs of the three gate can be obtained as

$$\begin{cases} \hat{g}_{in} = \sigma\left(\mathbf{W}_{in} \cdot [\hat{y}^i, \hat{h}^{i-1}, \hat{c}^{i-1}] + \vec{b}_{in}\right), \text{ input gate}, \\ \hat{g}_{fg} = \sigma\left(\mathbf{W}_{fg} \cdot [\hat{y}^i, \hat{h}^{i-1}, \hat{c}^{i-1}] + \vec{b}_{fg}\right), \text{ forget gate}, \\ \hat{g}_{op} = \sigma\left(\mathbf{W}_{op} \cdot [\hat{y}^i, \hat{h}^{i-1}, \hat{c}^{i}] + \vec{b}_{op}\right), \text{ output gate}, \end{cases} \tag{8}$$

where the operation $[\hat{y}^i, \hat{h}^{i-1}, \hat{c}^{i-1}]$ means to concatenate vectors $\hat{y}^i$, $\hat{h}^{i-1}$ and $\hat{c}^{i-1}$ to get a long vector, $\mathbf{W}_{in}$, $\mathbf{W}_{fg}$ and $\mathbf{W}_{op}$ are the weight matrices for the three gates, respectively, and $\vec{b}_{in}$, $\vec{b}_{fg}$ and $\vec{b}_{op}$ are the corresponding constant bias vectors. Then, with the three gates, we can update the state of the memory block and get its output as follows. We first define a tanh sigmoid activation function for a vector $\vec{x}$ as

$$\tanh(\vec{x}) = \frac{e^{\vec{x}} - e^{-\vec{x}}}{e^{\vec{x}} + e^{-\vec{x}}} \tag{9}$$

Then, we can get an intermediate state $\hat{c}^*$ with

$$\hat{c}^* = \tanh\left(\mathbf{W}_c \cdot [\hat{y}^i, \hat{h}^{i-1}] + \vec{b}_c\right), \tag{10}$$

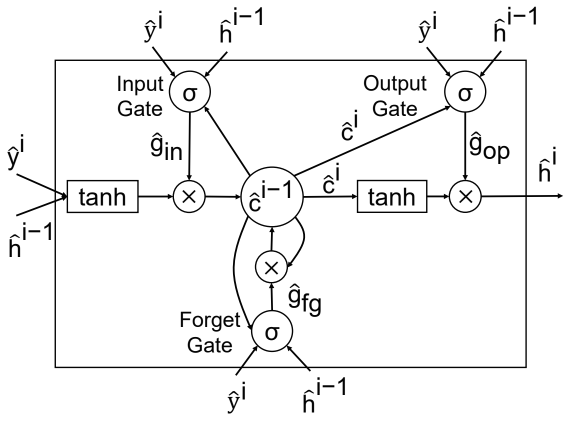


Fig. 5. Detailed architecture inside a LSTM memory block.

where $\mathbf{W}_c$ and $\vec{b}_c$ are the weight matrix and constant bias vector. Then, the output of the forget gate $\hat{g}_{fg}$ is a boolean vector, *i.e.*, each element is either 0 or 1, to denote whether certain information of the current state should be used in the state update or not. Specifically, the state update is as follows.

$$\hat{c}^i = \hat{g}_{fg} \otimes \hat{c}^{i-1} + \hat{g}_{in} \otimes \hat{c}^*, \tag{11}$$

where $\otimes$ refers to the operation that multiplies the corresponding elements of two vectors. Finally, by putting the updated state $\hat{c}^i$ into the last equation in Eq. (8), we can get the output of the memory block. Finally, how to predict a request with the DL model is explained. Note that, in the DL model, all the parameters, such as $\mathbf{W}_{in}$, $\mathbf{W}_{fg}$, $\mathbf{W}_{op}$, $\vec{b}_{in}$, $\vec{b}_{fg}$, $\vec{b}_{op}$, *etc.*, are obtained by training the model with historical samples, which will be explained in the next subsection.

## C. Training and Predicting of DL Model

To train the DL model, we define a loss function as follows to measure the loss between an actual request after preprocessing $\hat{R}^{I+1}$ and a predicted request $\hat{R}^{I+1,*}$. Then, the objective of the training is to get all the parameters of the DL model such that the loss can be minimized in prediction. As there are boolean vectors and real variables in each predicted request $\hat{R}^{I+1,*}$, we leverage the categorical cross entropy and mean square error (MSE) to measure the difference between $\hat{R}^{I+1,*}$ and $\hat{R}^{I+1}$ as

$$\begin{aligned} L(\hat{R}^{I+1,*}, \hat{R}^{I+1}) = &-\sum_{j=1}^{|V|} \left[ s_j^{I+1} \cdot \log\left(s_j^{I+1,*}\right) + d_j^{I+1} \cdot \log\left(d_j^{I+1,*}\right) \right] \\ &-\sum_{j=1}^{N} \left[ SC_j^{I+1} \cdot \log\left(SC_j^{I+1,*}\right) \right] \\ &+\frac{1}{3}\sum_{j=1}^{3} \left( \hat{r}_j^{I+1,*} - \hat{r}_j^{I+1} \right)^2, \end{aligned} \tag{12}$$

where $s_j^{I+1}$ denotes the $j$-th boolean element in vector $\vec{s}^{I+1}$, and so do similar ones, while $\{\hat{r}_j^{I+1}, j \in [1,3]\}$ denote the normalized values of the real variables $\{b^{I+1}, t_\tau^{I+1}, t_l^{I+1}\}$ in $\hat{R}^{I+1}$, respectively, and so do $\{\hat{r}_j^{I+1,*}, j \in [1,3]\}$.

*Algorithm* 2 shows the training procedure of the DL model. Here, *Lines* 1-4 are for the initialization. *Line* 1 divides the historical request series into $M$ supervised training samples, each of which consists of a sequence of arrived requests $\hat{\mathbf{R}}_\mathbf{m}$ and the adjacent next request $\hat{R}_m^{I+1}$ ($m \in [1, M]$). The parameters of the DL model gets initialized in *Lines* 2-3, while *Line* 4 initializes the global learning rate $\epsilon$, which decides the initial update rate of the parameters in each training iteration. The while-loop covering *Lines* 5-12 is for the training to get the DL model's parameters. In order to balance the tradeoff between the complexity and accuracy of the training, *Lines* 6-7 only choose a small batch of the training samples to train the DL model in each iteration [42]. Then, the loss $L$ of the current training iteration is calculated with Eq. (12) in *Line* 8 and stored in sequence $\Psi$ in *Line* 9. In *Line* 10, we leverage the well-known RMSProp algorithm in [25] with $\Psi$ and $\epsilon$ to update the parameters of the DL model based on the adaptive gradient calculation. Specially, the global learning rate $\epsilon$ multiplied by gradient decent values decide the update value of parameters. Finally, *Line* 12 validates the performance of the updated parameters on loss function in Eq. (12) by using the testing set. Here, the training and testing sets are obtained with the data preprocessing discussed in Section IV-A.

After being trained by *Algorithm* 2, the DL model can minimize the loss between a predicted request $\hat{R}^{I+1,*}$ and an actual future request $R^{I+1,*}$. Then, the DL model is applied in the provisioning framework to predict vNF-SC requests in each pre-deployment phase. Specifically, we preprocess the historical requests, feed them into the DL model to get the next request, include the predicted request in the historical request set, and repeat the procedure until all the requests that can come in within the next $\Delta T$ have been predicted. Note that, after obtaining a predicted request $\hat{R}^{I+1,*}$ from the DL model, we need to convert the components in it to those of an actual future request $R^{I+1,*}$ by reversing the encoding and transformation operations described in Section IV-A. Finally, to minimize the cumulative error in the prediction process, we

**Algorithm 2:** Training Procedure of DL Model

```
1  divide requests in the training set into $M$ supervised training samples $\{[\hat{\mathbf{R}}_\mathbf{m}, \hat{R}_m^{I+1}], m \in [1, M]\}$;
2  initialize weight matrices $\mathbf{W} \in \{\mathbf{W}_{in}, \mathbf{W}_{fg}, \mathbf{W}_{op}, \mathbf{W}_c, \mathbf{W}_p, \mathbf{W}_s, \mathbf{W}_d, \mathbf{W}_{SC}, \mathbf{W}_t\}$ with random Gaussian variables $\mathcal{N}(\mu = 0, \sigma^2 = 1)$;
3  initialize bias vectors $\vec{b} \in \{\vec{b}_{in}, \vec{b}_{fg}, \vec{b}_{op}, \vec{b}_c, \vec{b}_s, \vec{b}_d, \vec{b}_{SC}, \vec{b}_t\}$ and memory blocks' states $\{\hat{c}^i,\ i \in [I - w, I]\}$ as 0;
4  set initial global learning rate $\epsilon = 0.001$;
5  while loss $L$ from training does not converge do
6      choose a few samples $\{[\hat{\mathbf{R}}_\mathbf{1}, \hat{R}_1^{I+1}], [\hat{\mathbf{R}}_\mathbf{2}, \hat{R}_2^{I+1}], \cdots\}$;
7      input the selected samples to DL model to get outputs $\{\hat{R}_1^{I+1,*}, \hat{R}_2^{I+1,*}, \cdots\}$;
8      calculate average loss $L$ for all selected samples;
9      insert the average loss $L$ in sequence $\Psi$;
10     apply the RMSProp algorithm in [25] with $\Psi$ and $\epsilon$ to update parameters $\mathbf{W}$ and $\vec{b}$;
11     validate parameter performance on loss function in Eq. (12) by using the testing set;
12 end
```

update the states of the memory blocks in the DL model with the actual arrived requests, when the current $\Delta T$ has expired.

## V. Performance Evaluation

### A. Simulation Setup

We use the 14-node NSFNET topology in [43] as the physical topology of the IDC-EON in the simulations. We assume that each fiber link can accommodate $F = 358$ frequency slots (FS'), each of which is in the C-band and has a bandwidth of 12.5 GHz. On each node in the physical topology, there is a DC whose IT resource capacity is $C_v = 100$ units. We consider that the IDC-EON supports $M = 5$ types of vNFs, which can form $N = 10$ types of vNF-SCs. Here, each type of vNF consumes IT resources within $[0.4, 0.8]$ units and can process $[40, 80]$ Gbps traffic at most, and each type of vNF-SC consists of $[2, 4]$ vNFs randomly. We assume that a time-unit equals an hour. For the cost coefficients in Eqs. (1) and (2), we have $w_s = 80$ cost-units, $w_c = 80$ cost-units, $\tilde{w}_s = 1$ cost-unit/FS/hour, and $\tilde{w}_s = 1$ cost-unit/IT-unit/hour.

Note that, in the ideal case, the simulations should be based on the traces for dynamic vNF-SC requests in a practical inter-DC network. However, there are no such traces currently available for us, and thus we decide to emulate the requests based on the traces for real wide-area TCP connections in [44]. Specifically, we process the TCP traces in [44] as follows. To get a dynamic request $R^i = \{s^i, d^i, SC^i, b^i, t_a^i, t_l^i\}$, we map the source and destination addresses of a TCP connection to two DCs in the IDC-EON to get $s^i$ and $d^i$, map the application protocol of the TCP connection to a vNF-SC type, and scale the start-time, duration, and average bandwidth requirement of the TCP connection properly to get $t_a^i$, $t_l^i$ and $b^i$, respectively. The obtained vNF-SC requests' bandwidth requirements are within $[8.625, 152.875]$ Gbps and have an average value of 70.625 Gbps, while their hold-on time is within $[2, 26]$ hours and has a mean of 6.74 hours. Next, we preprocess the vNF-SC requests with the scheme discussed in Section IV-A, and use the first 80% requests as the training set, *i.e.*, the remaining 20% are put into the testing set. The training set is then utilized to train the DL model with the procedure in *Algorithm* 2. Here, we set $w = 19$, which means that the DL model is trained to predict a future request based on 20 most recent requests.

### B. Training Performance

We first conduct simulations to verify the training performance of our proposed DL model based on LSTM-NN. Fig. 6 shows the DL model's training performance on the training and testing set through training iterations, where the y-axis is the output of the loss function defined in Eq. (12). It can be seen that the loss on the training set decreases rapidly with the iterations and converges after ~300 rounds of training, which confirms that the proposed DL model based on LSTM-NN can fit the high-dimensional data in the training set well. Meanwhile, the loss on the testing set performs similarly in Fig. 6. This suggests that after being trained, the generalization of the DL model is excellent, which enables it to predict dynamic vNF-SC requests precisely.

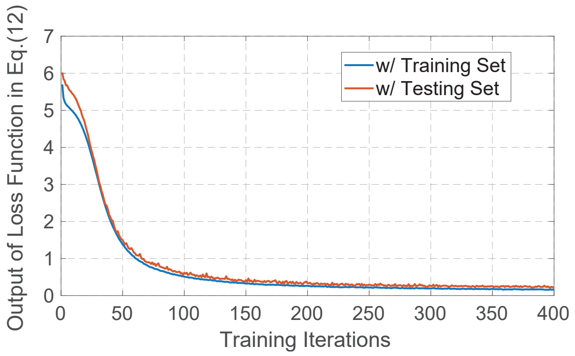


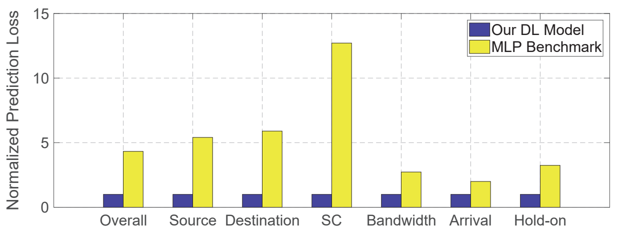
Fig. 6. DL model's loss on data sets through training iterations.

Then, in order to further confirm the DL model's training performance, we introduce a benchmark that is based on a traditional ANN using multilayer perceptron (MLP) since it is known that such a scheme also performs well on predicting high-dimensional time series [45]. We design the MLP benchmark to include two hidden layers, which is similar to our DL model, and put 1024 neurons in each hidden layer to realize reasonably good learning capability. Meanwhile, the input and output layers of the MLP benchmark have the same structures as those in our DL model. The results on running time indicate that the training process of our DL model takes ~1500 seconds to converge while the MLP benchmark converges faster, *i.e.*, within ~1000 seconds. Here, the 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 algorithms are implemented with TensorFlow 1.4.1.

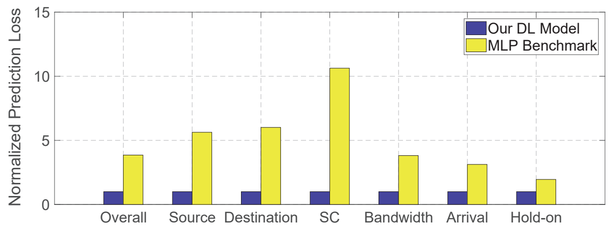
Nevertheless, even though the MLP benchmark converges faster, the results on prediction loss in Fig. 7 indicate that our DL model can predict the dynamic vNF-SC requests much more accurately, *i.e.*, not only on the prediction loss for overall vNF-SC requests but also on that for each individual component in the requests. Note that, the prediction loss here is calculated with the loss function in Eq. (12) or each corresponding term in it, but since the absolute value of a prediction loss is almost meaningless, we normalize all the results with the corresponding prediction losses from our DL model in Fig. 7. Moreover, we observe that the MLP benchmark's prediction performance on the discrete variables (*i.e.*, the source, destination and vNF-SC) is much worse than that on the continuous ones (*i.e.*, the bandwidth requirement, arrival time and hold-on time). However, to predict a vNF-SC request precisely, the discrete variables are actually much more important than the continuous ones. For instance, a prediction error on the source or destination of a request would definitely be more harmful than that on its bandwidth requirement.

### C. Provisioning of vNF-SC Requests

Next, we apply our DL model and the MLP benchmark in the pre-deployment phase for request prediction, and conduct simulations for dynamic vNF-SC provisioning. Here, we introduce an additional benchmark, namely, modified affiliation-aware vNF placement and forecast-assisted online deployment (mAaP-FaOD), which is based on the AaP-FaOD algorithm



(a) Results with training set



(b) Results with testing set

Fig. 7. Comparisons on prediction loss.

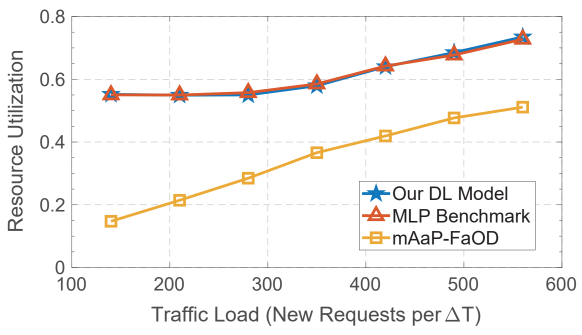


Fig. 8. Results on resource utilization in provisioning phase.

developed in [23] and modified to adapt to the vNF-SC provisioning in IDC-EONs. Specifically, mAaP-FaOD can only predict the numbers of different types of vNFs required in the upcoming provisioning phase with a Fourier-Series-based method and pre-deploy the vNFs in an affiliation-aware manner, but it can forecast neither where exactly the vNFs will be needed nor other parameters of vNF-SC requests.

The simulations on dynamic vNF-SC provisioning compare our DL model, the MLP benchmark, and mAaP-FaOD in terms of two performance metrics, *i.e.*, the blocking probability and resource utilization in the provisioning phase. As we have explained in Section III-B, only the pre-deployed vNFs and lightpaths will be used to assemble vNF-SCs on-demand. Therefore, a vNF-SC request can be blocked due to either inaccurate prediction in the pre-deployment phase or insufficient resources in the IDC-EON[4]. The blocking probability considers both of these two blocking cases. The resource utilization refers to the average usage of the pre-deployed spectrum and IT resources by the actually-arrived vNF-SC requests in the provisioning phase. We calculate the spectrum utilization by first summarizing the products of used spectra and time of usage and then dividing the result by the product of total pre-deployed spectra and hold-on time. The IT resource utilization is obtained similarly. Finally, we get the average value of spectrum and IT resource utilizations as the resource utilization. As the average hold-on time of vNF-SC requests is 6.74 hours, the simulations here use $\Delta T = 7$ hours while the effects of $\Delta T$ and how to choose an optimal value for it will be discussed in the next subsection.

Fig. 8 illustrates the results on resource utilization, where the traffic load refers to the average number of new vNF-SC requests that will be arrived in each provisioning phase (*i.e.*, $\Delta T$). It can be seen that the resource utilization from mAaP-FaOD is much less than those from our DL model and the MLP benchmark. This is because without the AI-assisted vNF-SC request prediction, mAaP-FaOD can only forecast the numbers of different types of vNFs required in the upcoming provisioning phase, while it does not know where exactly the vNFs will be needed or other parameters of future requests. Therefore, there would be a large deviation between prediction and reality, which leads to the lowest resource utilization.

The resource utilizations from our DL model and the MLP benchmark are almost the same and range within $[0.517, 0.735]$. Such resource utilizations are mainly caused by the provisioning framework, *i.e.*, future vNF-SC requests are predicted and deployed in the pre-deployment phase and only traffic steering will be performed when the requests actually arrive in the provisioning phase. Hence, the pre-deployed resources will be idle before the predicted requests come in, and this decreases the resource utilization significantly. The analysis above can be verified, if we check the resource utilizations' trend. Specifically, they actually increase with the traffic load in general, which is because when the traffic load is higher, new requests arrive more frequently and thus reduce the average idle time of the pre-deployed resources. To this end, we can see that the cost-effectiveness of the provisioning framework can be affected by the value of $\Delta T$, and we will elaborate on this in the next subsection.

Meanwhile, since the impact of $\Delta T$ dominates the resource utilization, the results from our DL model and the MLP benchmark are almost the same in Fig. 8, but their performance can be distinguished if we check their results on blocking probability in Fig. 9. Our DL model achieves the lowest blocking probability among the three algorithms, especially for the low traffic cases. In the meantime, it is interesting to notice that when the number of new requests per $\Delta T$ decreases from 560 to 280, the blocking probability from our DL model just decreases with the trend of a generic blocking probability curve, but when the traffic load keeps decreasing, the decreasing speed of the blocking probability slows down dramatically and a "floor" can be seen in its curve. This is because when the traffic load is relatively high, the request blocking due to insufficient resources dominates and makes the curve follow the trend of a generic blocking probability curve, but when the traffic load is relatively low, the request blocking due to inaccurate prediction in the pre-deployment phase dominates and causes the "floor" in the curve.

The similar trend can be observed in the blocking probability curve of the MLP benchmark in Fig. 9. Actually, when

[4]Note that, in *Algorithm* 1, *Line* 9 would skip the pre-deployment of a predicted request if the resources in the IDC-EON are insufficient for it.

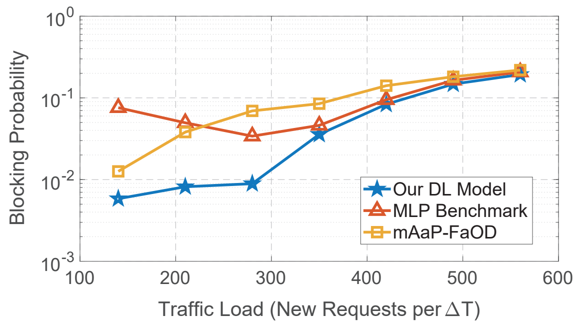


Fig. 9. Results on blocking probability in provisioning phase.

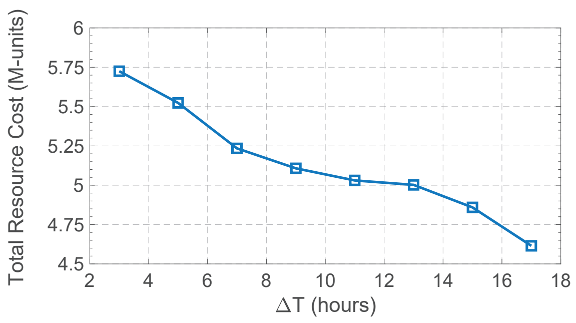


Fig. 11. Total resource cost changing with $\Delta T$ (40 new requests per hour on average).

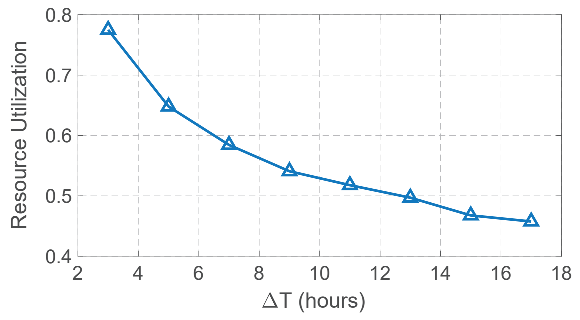


Fig. 10. Resource utilization changing with $\Delta T$ (40 new requests per hour on average).

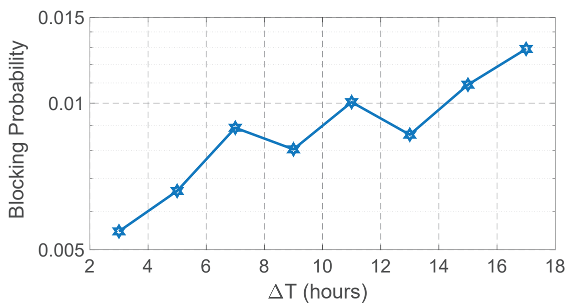


Fig. 12. Blocking probability changing with $\Delta T$ (40 new requests per hour on average).

the number of new requests per $\Delta T$ decreases from 280 to 140, the blocking probability from the MLP benchmark shows an increasing trend. This is caused by the relatively high prediction loss of the MLP benchmark. Note that, when the traffic load is lower, the AI-assisted traffic prediction module forecasts and pre-deploys a smaller number of vNF-SCs. Statistically, since fewer resources are pre-deployed, the high prediction loss of the MLP benchmark might lead to a higher blocking probability and degrade the algorithm's performance at low traffic loads. As the prediction performance of mAaP-FaOD is the worst among the three, it provides the highest blocking probability at most traffic loads.

### *D. Effect of $\Delta T$ on Cost-Effectiveness*

Both the results in Fig. 8 and our discussion in Section III-C suggest that the value of $\Delta T$ can affect the cost-effectiveness of the proposed vNF-SC provisioning framework. Therefore, we conduct more simulations with different $\Delta T$ in this subsection. Here, we only consider our DL model and fix the traffic load as 280 new requests per 7 hours (*i.e.*, 40 new requests per hour), since the blocking probability in Fig. 9 suggests that the request blocking due to insufficient resources would start to dominate when the traffic load is higher than this value. Fig. 10 shows how the resource utilization changes with $\Delta T$. As expected, the resource utilization decreases with $\Delta T$. This is because with a longer $\Delta T$, the pre-deployed resources would be idle for a longer time statistically. Meanwhile, we notice that with $\Delta T = 3$ hours, the resource utilization is 0.775, which is significantly higher than that with $\Delta T = 7$ hours. Hence, if we use a shorter $\Delta T$ to update the pre-deployed resources more timely, the resource utilization can be improved effectively. However, a shorter $\Delta T$ also makes the request predication and resource pre-deployment happen more frequently. This will complicate NC&M and increase OPEX, which can be verified with the cost results in Fig. 11.

Fig. 11 plots the results on total resource cost for different $\Delta T$. Here, we calculate the total cost of deployed resources with Eqs. (1) and (2), *i.e.*, both the static and dynamic costs of resource deployment have been considered. It can be seen that the total resource cost decreases with $\Delta T$. This phenomenon can be understood as follows. When $\Delta T$ is shorter, the system needs to turn on and off resources more frequently for the pre-deployment, which will increase the total static cost a lot. Note that, in real network systems, the static cost usually dominates the total resource cost [46]. For example, the cost of setting up a lightpath or instantiating a virtual machine for vNF deployment is usually much higher than that of running them afterwards. Therefore, as the total static cost increases fast when $\Delta T$ decreases, a shorter $\Delta T$ makes the system more costly. To this end, by combining the results in Figs. 10 and 11, we can find that there is a clear performance tradeoff between resource utilization and total resource cost, which can be adjusted by changing $\Delta T$.

The trend of blocking probability changing with $\Delta T$ is plotted in Fig. 12, which indicates that in general, the blocking probability can increase slightly with $\Delta T$. This is because a longer $\Delta T$ makes the pre-deployed resources be updated less timely. Hence, even though increasing $\Delta T$ can reduce the total resource cost, the system's performance on the resource utilization and blocking probability would also be degraded. In summary, to optimize the cost-effectiveness of the vNF-SC provisioning framework, one should choose $\Delta T$ carefully.

## VI. Conclusion

In this work, we proposed a AI-assisted provisioning framework to realize on-demand and cost-effective provisioning of vNF-SCs in IDC-EONs. The framework was designed as a discrete-time system, in which the operations are performed periodically in fixed TS' and each TS includes a pre-deployment phase followed by a provisioning phase. We developed a DL model to predict future vNF-SC requests accurately in the pre-deployment phase for guiding the lightpath establishment and vNF deployment for the predicted requests. Then, the provisioning phase only needs to collect dynamic vNF-SC requests from clients and serve them in real-time by steering their traffic through the required vNFs in sequence. The proposed framework was evaluated with simulations that leveraged real traffic traces, and the results indicated that our DL model achieves higher request prediction accuracy and lower blocking probability than two benchmarks.

Moreover, our analysis on the effect of the provisioning period $\Delta T$ suggested that the value of $\Delta T$ should be carefully chosen for achieving high cost-effectiveness. This actually points out an interesting direction for us to explore in the future, to further improve the performance of our proposed vNF-SC provisioning framework.

## 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).

## References

[1] P. Lu, L. Zhang, X. Liu, J. Yao, and Z. Zhu, "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] "Network functions virtualization (NFV)," Jan. 2012. [Online]. Available: https://portal.etsi.org/portal/server.pt/community/NFV/367

[3] M. Zeng, W. Fang, and Z. Zhu, "Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks," *J. Lightw. Technol.*, vol. 34, pp. 3330–3341, Jul. 2016.

[4] S. Clayman, E. Maini, A. Galis, A. Manzalini, and N. Mazzocca, "The dynamic placement of virtual network functions," in *Proc. of NOMS 2014*, pp. 1–9, May 2014.

[5] M. Zeng, W. Fang, J. Rodrigues, and Z. Zhu, "Orchestrating multicast-oriented NFV trees in inter-DC elastic optical networks," in *Proc. of ICC 2016*, pp. 1–6, Jun. 2016.

[6] W. Lu, L. Liang, and Z. Zhu, "On vNF-SC deployment and task scheduling for bulk-data transfers in inter-DC EONs," in *Proc. of ICCC 2017*, pp. 1–4, Oct. 2017.

[7] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Proc. of CloudNet 2014*, pp. 7–13, Oct. 2014.

[8] X. Chen, L. Sun, Z. Zhu, H. Lu, and S. Yoo, "On efficient incentive-driven VNF service chain provisioning with mixed-strategy gaming in broker-based EO-IDCNs," in *Proc. of OFC 2017*, pp. 1–3, Mar. 2017.

[9] W. Lu, L. Liang, and Z. Zhu, "Orchestrating data-intensive vNF service chains in inter-DC elastic optical networks," in *Proc. of ONDM 2017*, pp. 1–6, May 2017.

[10] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network function placement for NFV chaining in packet/optical datacenters," *J. Lightw. Technol.*, vol. 33, pp. 1565–1570, Apr. 2015.

[11] K. Christodoulopoulos, I. Tomkos, and E. Varvarigos, "Elastic bandwidth allocation in flexible OFDM-based optical networks," *J. Lightw. Technol.*, vol. 29, pp. 1354–1366, May 2011.

[12] 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.

[13] L. Gong, X. Zhou, X. Liu, W. Zhao, W. Lu, and Z. Zhu, "Efficient resource allocation for all-optical multicasting over spectrum-sliced elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, pp. 836–847, Aug. 2013.

[14] 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. Technol.*, vol. 33, pp. 3719–3729, Sept. 2015.

[15] W. Fang, M. Zeng, X. Liu, W. Lu, and Z. Zhu, "Joint spectrum and IT resource allocation for efficient vNF service chaining in inter-datacenter elastic optical networks," *IEEE Commun. Lett.*, vol. 20, pp. 1539–1542, Aug. 2016.

[16] W. Lu, M. Zeng, W. Fang, and Z. Zhu, "Network function virtualization in optical inter-datacenter elastic optical networks," in *Proc. of OECC 2017*, pp. 1–2, Jul./Aug. 2017.

[17] W. Fang, M. Lu, X. Liu, L. Gong, and Z. Zhu, "Joint defragmentation of optical spectrum and IT resources in elastic optical datacenter interconnections," *J. Opt. Commun. Netw.*, vol. 7, pp. 314–324, Mar. 2015.

[18] J. Yin, J. Guo, B. Kong, H. Yin, and Z. Zhu, "Experimental demonstration of building and operating QoS-aware survivable vSD-EONs with transparent resiliency," *Opt. Express*, vol. 25, pp. 15 468–15 480, 2017.

[19] W. Cerroni and F. Callegati, "Live migration of virtual network functions in cloud-based edge networks," in *Proc. of ICC 2014*, pp. 2963–2968, Jun. 2014.

[20] S. Krishnan and R. Sitaraman, "Video stream quality impacts viewer behavior: Inferring causality using quasi-experimental designs," *IEEE/ACM Trans. Netw.*, vol. 21, pp. 2001–2014, Dec. 2013.

[21] X. Chen, Z. Zhu, J. Guo, S. Kang, R. Proietti, A. Castro, and B. Yoo, "Leveraging mixed-strategy gaming to realize incentive-driven VNF service chain provisioning in broker-based elastic optical inter-datacenter networks," *J. Opt. Commun. Netw., in Press*, 2017.

[22] Z. Zhu, C. Chen, S. Ma, L. Liu, X. Feng, and S. Yoo, "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.

[23] Q. Sun, P. Lu, W. Lu, and Z. Zhu, "Forecast-assisted NFV service chain deployment based on affiliation-aware vNF placement," in *Proc. of GLOBECOM 2016*, pp. 1–6, Dec. 2016.

[24] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[25] T. Tieleman and G. Hinton, "Lecture 6.5 RMSProp: Divide the gradient by a running average of its recent magnitude," 2012. [Online]. Available: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

[26] L. Gong, Y. Wen, Z. Zhu, and T. Lee, "Toward profit-seeking virtual network embedding algorithm via global resource capacity," in *Proc. of INFOCOM 2014*, pp. 1–9, Apr. 2014.

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

[28] 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.

[29] 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.

[30] W. Hou, Z. Ning, L. Guo, Z. Chen, and M. Obaidat, "Novel framework of risk-aware virtual network embedding in optical data center networks," *IEEE Syst. J., in Press*, 2017.

[31] 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.

[32] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On dynamic service function chain deployment and readjustment," *IEEE Trans. Netw. Serv. Manag.*, vol. 14, pp. 543–553, Sept. 2017.

[33] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, "Elastic virtual network function placement," in *Proc. of CloudNet 2015*, pp. 255–260, Oct. 2015.

[34] X. Wang, C. Wu, F. Le, A. Liu, Z. Li, and F. Lau, "Online VNF scaling in datacenters," in *Proc. of CLOUD 2016*, pp. 140–147, Jun. 2016.

[35] L. Gupta, M. Samaka, R. Jain, A. Erbad, D. Bhamare, and C. Metz, "COLAP: A predictive framework for service function chain placement

in a multi-cloud environment," in *Proc. of CCWC 2017*, pp. 1–9, Jan. 2017.

[36] D. Park and D. Woo, "Prediction of network traffic using dynamic BiLinear recurrent neural network," in *Proc. of ICNC 2009*, pp. 419–423, Aug. 2009.

[37] 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.

[38] Y. Li, H. Liu, W. Yang, D. Hu, X. Wang, and W. Xu, "Predicting inter-data-center network traffic using elephant flow and sublink information," *IEEE Trans. Netw. Serv. Manag.*, vol. 13, pp. 782–792, Dec. 2016.

[39] N. Sambo, P. Castoldi, A. D'Errico, E. Riccardi, A. Pagano, M. Moreolo, J. Fabrega, D. Rafique, A. Napoli, S. Frigerio, M. Salas, G. Zervas, M. Nolle, J. Fischer, A. Lord, and J. Gimenez, "Next generation sliceable bandwidth variable transponders," *IEEE Commun. Mag.*, vol. 53, pp. 163–171, Feb. 2015.

[40] J. Yao, P. Lu, L. Gong, and Z. Zhu, "On fast and coordinated data backup in geo-distributed optical inter-datacenter networks," *J. Lightw. Technol.*, vol. 33, pp. 3005–3015, Jul. 2015.

[41] A. Coates and A. Ng, "The importance of encoding versus training with sparse coding and vector quantization," in *Proc. of ICML 2011*, pp. 921–928, Jul. 2011.

[42] M. Li, T. Zhang, Y. Chen, and A. Smola, "Efficient mini-batch training for stochastic optimization," in *Proc. of SIGKDD 2014*, pp. 661–670, Aug. 2014.

[43] 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.

[44] V. Paxson, "Empirically derived analytic models of wide-area TCP connections," *IEEE/ACM Trans. Netw.*, vol. 2, pp. 316–336, Aug. 1994.

[45] M. Gardner and S. Dorling, "Artificial neural networks (the multilayer perceptron) - a review of applications in the atmospheric sciences," *Atmos. Environ.*, vol. 32, pp. 2627–2636, Aug. 1998.

[46] 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.