This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2019.2923615, Journal of Lightwave Technology

DeepRMSA: A Deep Reinforcement Learning Framework for Routing, Modulation and Spectrum Assignment in Elastic Optical Networks

Xiaoliang Chen, *Member, IEEE*, Baojia Li, Roberto Proietti, Hongbo Lu, Zuqing Zhu, *Senior Member, IEEE*, S. J. Ben Yoo, *Fellow, IEEE, Fellow, OSA*

***Abstract*—This paper proposes DeepRMSA, a deep reinforcement learning framework for routing, modulation and spectrum assignment (RMSA) in elastic optical networks (EONs). DeepRMSA learns the correct online RMSA policies by parameterizing the policies with deep neural networks (DNNs) that can sense complex EON states. The DNNs are trained with experiences of dynamic lightpath provisioning. We first modify the asynchronous advantage actor-critic algorithm and present an episode-based training mechanism for DeepRMSA, namely, DeepRMSA-EP. DeepRMSA-EP divides the dynamic provisioning process into multiple episodes (each containing the servicing of a fixed number of lightpath requests) and performs training by the end of each episode. The optimization target of DeepRMSA-EP at each step of servicing a request is to maximize the cumulative reward within the rest of the episode. Thus, we obviate the need for estimating the rewards related to unknown future states. To overcome the instability issue in the training of DeepRMSA-EP due to the oscillations of cumulative rewards, we further propose a window-based flexible training mechanism, i.e., DeepRMSA-FLX. DeepRMSA-FLX attempts to smooth out the oscillations by defining the optimization scope at each step as a sliding window, and ensuring that the cumulative rewards always include rewards from a fixed number of requests. Evaluations with the two sample topologies show that DeepRMSA-FLX can effectively stabilize the training while achieving blocking probability reductions of more than 20.3% and 14.3%, when compared with the baselines.**

***Index Terms*—Elastic optical networks (EONs), Routing, modulation and spectrum assignment (RMSA), Deep reinforcement learning, Asynchronous advantage actor-critic algorithm.**

I. Introduction

THE explosive growth of emerging applications (e.g., cloud computing) and the popular adoption of new networking paradigms (e.g., the Internet of Things) are demanding a new network infrastructure that can support dynamic, high-capacity and quality-of-transmission (QoT)-guaranteed end-to-end services [1]. Recently, elastic optical networking (EON) has emerged as one of the most promising networking technologies for the next-generation backbone networks [2]. Compared with the traditional fixed-grid (e.g., 50 GHz) wavelength-division multiplexing (WDM) scheme, EON can flexibly set up bandwidth-variable superchannels by grooming series of finer-granularity (e.g., 6.25 GHz) subcarriers and adapting the modulation formats according to the QoT of lightpaths [3].

The flexible resource allocation mechanisms in EON, on the other hand, make the corresponding service provisioning designs more complicated [4], [5]. To fully exploit the benefits of such flexibilities and realize cost-effective EON, previous studies have intensively investigated the routing, modulation and spectrum assignment (RMSA) problem for EON [6]. The authors of [7]–[9] first proposed integer linear programming (ILP) models for solving the static RMSA problems, where all the lightpath requests are assumed to be known in prior. While the ILP models can provide the optimal solutions to the RMSA problems, they are proved to be $\mathcal{NP}$-hard [7] and are intractable for large-scale problems. In this context, a number of heuristic or approximation algorithms have been developed. In [7], Wang *et al.* proposed two algorithms, namely, balanced load spectrum allocation and shortest path with maximum spectrum reuse, to minimize the maximum required spectrum resources in an EON accounting for the given traffic demand. The authors of [8] presented a simulated annealing approach for determining the servicing order of lightpath requests and applied the k-shortest path routing and first-fit (KSP-FF) scheme to calculate the RMSA solution for each request afterward. In [9], [10], the authors investigated to leverage genetic algorithms to realize joint RMSA optimizations. A conflict graph based two-phase algorithm with proved performance level was proposed in [11]. For more heuristic RMSA designs, such as random-fit, exact-fit and most-used spectrum assignment, readers can refer to [6].

Unlike static RMSA problems for which explicit optimization models can be formulated, optimizing dynamic lightpath provisioning in EONs (i.e., dynamic RMSA problems) is more challenging. The dynamic arrivals and departures of lightpath requests as well as the uncertainty of future traffic could dramatically destabilize the EON state and thus deteriorate the efficiency of the optimizations based on the current state. To cope with such dynamics, a few dynamic RMSA designs have been reported lately, in addition to those that can be derived from the aforementioned static RMSA algorithms. The authors in [12] applied the multi-path routing scheme and developed several empirical weighting methods taking into account path lengths, link spectrum utilization, and other features to realize state-aware dynamic RMSA. In [13], Yin *et*

X. Chen, R. Proietti, H. Lu and S. J. B. Yoo are with the Department of Electrical and Computer Engineering, University of California, Davis, Davis, CA 95616, USA (Email: xlichen@ucdavis.edu, sbyoo@ucdavis.edu).

B. Li 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 Dec. 8, 2018.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2019.2923615, Journal of Lightwave Technology

al. investigated the spectrum fragmentation effect in dynamic lightpath provisioning and proposed a fragmentation-aware RMSA algorithm to mitigate spectrum fragmentation. More aggressive service reconfiguration approaches, e.g., spectrum defragmentation [14], [15], have also been proposed as complements to normal RMSA algorithms to enable periodical service consolidations but at the expense of high operational costs. However, the existing works only apply fixed RMSA policies regardless of the time-varying EON states or rely on simple empirical policies based on manually extracted features, i.e., lack of comprehensive perceptions of the holistic EON states, and therefore are unable to achieve real adaptive service provisioning in EONs.

In the meantime, recent advances in deep reinforcement learning (DRL) have demonstrated beyond human-level performance in handling large-scale online control tasks [16], [17]. DRL parameterizes action policies with deep neural networks (DNNs) [18] that can perceive complex system states from high-dimensional input data, such as, images, and traffic matrices. By accumulating action experiences from repeated interactions with the target systems and by reinforcing actions leading to higher rewards, DRL is able to learn successful policies (i.e., correct configurations of the DNNs) progressively. The application of DRL in the communication and networking domain has received intensive research interests during the past two years [19]–[21]. In [20], the authors enhanced the general deep Q-learning framework in [16] with novel exploration and experience replay techniques to solve the traffic engineering problem. The authors of [21] presented a DRL-based framework for datacenter network management and demonstrated a DRL agent which can learn the optimal topology configurations with respect to different application profiles. Nevertheless, the application of DRL in optical networking, or in particular, for addressing the RMSA problem, has not been investigated.

In this paper, we propose DeepRMSA, a DRL-based RMSA framework for learning the optimal online RMSA policies in EONs. The contributions of this paper can be summarized as follows. 1) We propose, for the first time, a DRL framework for optical network management and resource allocation, i.e., RMSA. 2) We propose two training mechanisms for DeepRMSA, taking into account the unique characteristics of the RMSA problem. 3) Numerical results verify the superiority of DeepRMSA over the state-of-art heuristic algorithms.

The rest of the paper is organized as follows. Section II presents the RMSA problem formulation. Section III discusses the operation principle of DeepRMSA. In Section IV, we detail the design of DeepRMSA, including the modeling and the training mechanisms. Then, in Section V, we show the performance evaluations and related discussions. Finally, Section VI concludes the paper.

II. RMSA Problem Formulation

Let $G(V, E, F)$ denote an EON topology, where V and E represent the sets of nodes and fiber links, $F = \{F_{e,f} \mid_{e,f}\}$ contains the state of each frequency slot (FS) $f \in [1, f_0]$ on each fiber link $e \in E$. We model a lightpath request from node o to d ($o, d \in V$) as $\mathcal{R}_t(o, d, b, \tau)$, with b Gb/s and τ denoting the bandwidth requirement and service duration, respectively. To provision $\mathcal{R}_t$, we need to compute an end-to-end routing path $\mathcal{P}_{o,d}$, determine a proper modulation format m to use for QoT assurance, and allocate a number of spectrally contiguous FS's (i.e., the spectrum contiguous constraint) on each link along $\mathcal{P}_{o,d}$ according to b and m. In this work, we assume that the EON is not equipped with the spectrum conversion capability. Therefore, the spectra allocated on different fibers to $\mathcal{R}_t$ must align (i.e., the spectrum continuous constraint). We adopt the impairment-aware model in [22] to decide the modulation format according to the physical distance of $\mathcal{P}_{o,d}$. Specifically, the number of FS's needed can be computed as,

$$n = \left\lceil \frac{b}{m \cdot C_{grid}^{BPSK}} \right\rceil, \tag{1}$$

where C_{grid}^{BPSK} is the data rate an FS of BPSK signal can support and $m \in [1, 2, 3, 4]$ corresponds to BPSK, QPSK, 8-QAM and 16-QAM, respectively. The static RMSA problem (i.e., offline network planning) gives a set of permanent lightpath requests $\mathcal{R} = \{\mathcal{R}_t \mid_t\}$ ($\tau \to \infty$) and requires provisioning all of them in a batch following the link capacity constraint [7]. The objective of the static RMSA problem is to minimize the total spectrum usage. Unlike the static problem where requests are known in prior, in the dynamic RMSA problem (i.e., online lightpath provisioning) being considered in this work, lightpath requests arrive and expire on-the-fly and need to be serviced immediately upon their arrivals. The dynamic RMSA problem aims at minimizing the long-term request blocking probability, which is defined as the ratio of the number of blocked requests to the total number of requests over a period.

III. DeepRMSA Operation Principle

Fig. 1 shows the operation principle of DeepRMSA. DeepRMSA takes advantage of the software-defined networking (SDN) paradigm for centralized and automated control and management of the EON data plane [23]. Specifically, a remote SDN controller interacts with the local SDN agents to collect network states and lightpath requests, and distribute RMSA schemes, while the SDN agents drive the actual device configurations according to the received commands. The operation principle of DeepRMSA is designed based on the framework of DRL. Upon receiving a lightpath request $\mathcal{R}_t$ (*step 1*), the SDN controller retrieves from the traffic engineering database key network state representations, including the in-service lightpaths, resource utilization and topology abstraction, and invokes the feature engineering module to generate tailored state data s_t for DeepRMSA (*step 2*). The DNNs of DeepRMSA read the state data and output an RMSA policy $\pi_t(A|s_t, \theta)$ for the SDN controller, where A is the set of candidate RMSA schemes for $\mathcal{R}_t$ and θ represents the set of parameters of the DNNs (*step 3*). Typically, π_t gives a probability distribution over A. The controller in turn takes an action $a_t \in A$ based on π_t and attempts to set up the corresponding lightpath (*step 4*). The reward system receives the outcome related to the previous RMSA operations as feedback (step 5) and produces an immediate reward r_t for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2019.2923615, Journal of Lightwave Technology

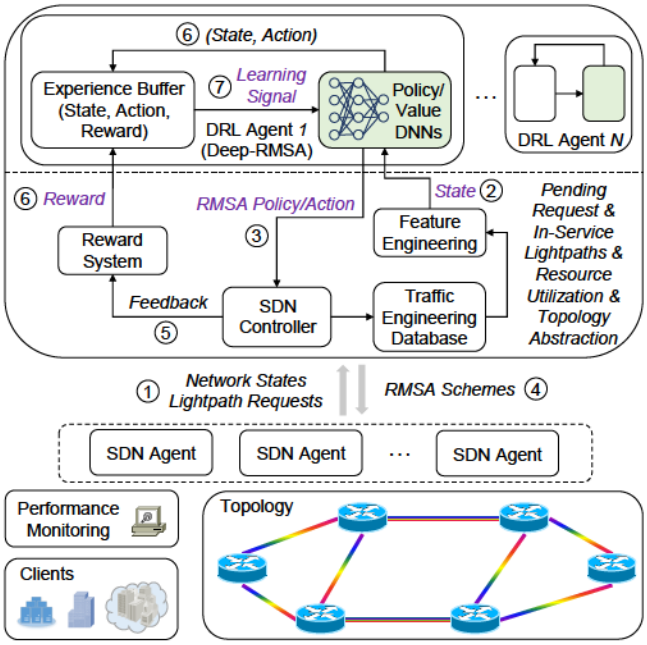

Fig. 1. Operation principle of DeepRMSA.

DeepRMSA. r_t, together with s_t and a_t, are stored in an experience buffer (*step 6*), from which DeepRMSA derives training signals for updating the DNNs afterward (*step 7*). The objective of DeepRMSA upon servicing $\mathcal{R}_t$ is to maximize the long-term cumulative reward defined as,

$$\Gamma_t = \sum_{t' \in [t,\infty)} \gamma^{t'-t} \cdot r_{t'}, \tag{2}$$

where $\gamma \in [0,1]$ is the discount factor that decays future rewards. Eventually, DeepRMSA enables a self-learning capability that can learn and adapt RMSA policies through dynamic lightpath provisioning. Note that, by deploying multiple parallel DRL agents, each for a particular application or functionality (e.g., protection [23] and defragmentation [15]), we can extend DeepRMSA to build an intact autonomic EON system.

IV. Modeling and Training Mechanism

In this section, we first present the modeling of DeepRMSA, including the definitions of state representation, action space, and reward. Then, we take into account the unique characteristics of dynamic lightpath provisioning and develop two training mechanisms for DeepRMSA.

A. Modeling

1) State: The state representation s_t for DeepRMSA is an $1 \times (2|V|+1+(2J+3)K)$ array containing the information of $\mathcal{R}_t$ and the spectrum utilization on the K-shortest[1] candidate paths for $\mathcal{R}_t$. Let us define the array as,

$$s_t = \left\{ o, d, \tau, \{\{z_k^{1,j}, z_k^{2,j}\}|_{j\in[1,J]}, z_k^3, z_k^4, z_k^5\}|_{k\in[1,K]} \right\}. \tag{3}$$

[1]We typically set a small value of K (e.g., $K = 5$) [7] instead of enumerating all possible candidate paths for every o d pair because the gain from using those excessive long routing paths is very limited.

Specifically, we use $2|V|+1$ elements of s_t to convey o, d (in the one-hot format), and τ, where $|V|$ represents the number of nodes in V. For each k of the K paths, we calculate the sizes $z_k^{1,j}$ and the starting indices $z_k^{2,j}$ of the first J available FS-block, the required number of FS's z_k^3 based on the applicable modulation format, the average size of the available FS-blocks z_k^4, and the total number of available FS's z_k^5. When the number of possible candidate paths between o and d (denoted as $\hat{K}_{o,d}$) is smaller than K, we assign $\{\{z_k^{1,j}, z_k^{2,j}\}|_{j\in[1,J]}, z_k^3, z_k^4, z_k^5\}$ ($\forall k > \hat{K}_{o,d}$) an array of -1 to ensure a consistent format of s_t. Hence, we aim to extract key features on different candidate paths, from which DeepRMSA can sense the global EON state. Fig. 2 shows an example of constructing s_t in DeepRMSA. For the sake of simplicity, we omit the details related to modulation format selection and assume two or three FS's are required when path 1-2-4 ($k = 1$) or 1-3-5-4 ($k = 2$) is used. We obtain $J = 2$ available FS-blocks (marked by the dashed boxes) on each of the paths. For instance, the first available FS-block ($j = 1$) on path 1-2-4 has a size of $z_1^{1,1} = 3$ and a starting index of $z_1^{2,1} = 0$. Note that, a more comprehensive design could include the original two-dimensional spectrum state F in s_t directly to avoid any information loss. However, this would dramatically increase the scale of s_t (i.e., requiring $f_0 \cdot |E|$ elements simply for conveying F) and cause scalability issues. Moreover, making DeepRMSA extract useful features from the large-scale binary matrix while incorporating also the topology connectivity and the spectrum continuous and contiguous constraints in EON is not trivial. An interesting solution could be applying a distributed learning approach while constructing topological state representations and learning with graph neural networks [24]. We will keep this as one of our future research tasks.

2) Action: DeepRMSA selects for each $\mathcal{R}_t$ a routing path from the K candidates and one of the J FS-blocks on the selected path. Therefore, the action space (denoted as A) includes $K \cdot J$ actions.

3) Reward: DeepRMSA receives an immediate reward r_t of 1 if $\mathcal{R}_t$ is successfully serviced. Otherwise, $r_t = -1$.

4) DNNs: DeepRMSA employs a policy DNN $f_{\theta_p}(s_t)$ for generating the RMSA policy π_t and a value DNN $f_{\theta_v}(s_t)$ for estimating the value of s_t (i.e., the discounted cumulative reward defined by Eq. 2). θ_p and θ_v are the sets of parameters of the policy and the value DNNs, respectively. $f_{\theta_p}(s_t)$ and $f_{\theta_v}(s_t)$ share the same fully-connected DNN architecture [18] except for the output layers. The output layer of $f_{\theta_p}(s_t)$ consists of $K \cdot J$ neurons, whereas $f_{\theta_v}(s_t)$ has only one output neuron.

B. Training

We designed the training of DeepRMSA based on the framework of the A3C algorithm [17]. Basically, A3C makes use of multiple parallel actor-learners (child threads of a DRL agent), each interacting with its own copy of the system environment, to achieve learning with more abundant and diversified samples. The actor-learners maintain a set of global DNN parameters θ_p^* and θ_v^* asynchronously.

Different from general DRL tasks that can be modeled as Markov decision processes (i.e., the state transition

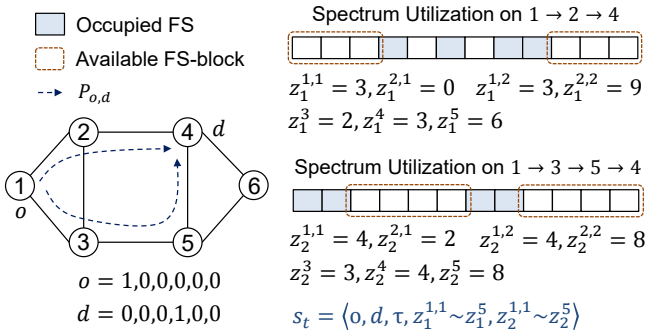

Fig. 2. An illustrative example of state representation in DeepRMSA ($K = 2, J = 2$).

from s_t to s_{t+1} follows a probability distribution given by $P(s_{t+1}|s_t, a_t)$), DeepRMSA involves state transitions which are difficult to be modeled. In particular, due to the fact that $\mathcal{R}_{r+1}$ can be random, there can be infinite possible states for s_{t+1} in DeepRMSA. Thus, we first slightly modified the standard A3C algorithm by defining an episode as the servicing of N lightpath requests, and by making N equal to the training batch size. Here, an episode defines the optimization scope of a DRL task. This way, we eliminate the need for estimating the value of s_{t+1}. We denote DeepRMSA with the episode-based training mechanism as DeepRMSA-EP. *Algorithm* 1 summarizes the procedures of an actor-learner thread in DeepRMSA-EP. In line 1, the actor-learner initiates an empty experience buffer Λ and sets ε as 1. Then, for each $\mathcal{R}_t$, the actor-learner checks whether Λ is empty (i.e., a new episode starts), and if true, synchronizes the local DNNs with the sets of global parameters (lines 3-5). Line 6 updates the EON state by releasing the resources allocated to lightpaths that expire. In line 7, the actor-learner obtains s_t based on the model discussed in Section IV-A. In line 8, the actor-learner invokes the policy and value DNNs to generate an RMSA policy and a value estimation for s_t. Note that, in DeepRMSA-EP, we make s_t include one more element to indicate the position of $\mathcal{R}_t$ regarding the current episode. For instance, if $\mathcal{R}_t$ is the i-th request of the episode, we calculate a position indicator as $(N - i + 1)/N$. The actor-learner decides an RMSA scheme based on the generated policy (lines 9-10) and receives a reward accordingly (line 11). Specifically, with a probability of ε, the actor-learner applies the Roulette strategy, otherwise, it simply selects the action corresponding to the highest probability. The RMSA sample is then stored in the buffer (line 12). With lines 13-21, DeepRMSA-EP performs training every time the buffer contains N samples. Specifically, in the for-loop of lines 14-16, the algorithm first calculates for each sample $\chi_{t'}$ in the buffer the discounted cumulative reward (staring from $\mathcal{R}_{t'}$ till the end of the episode) as,

$$\Gamma_{t'} = \sum_{i\in[0,N-1],\chi_{t'+i}\in\Lambda} \gamma^i \cdot r_{t'+i}. \tag{4}$$

Then, the advantage of each action being taken can be obtained by,

$$\delta_{t'} = \Gamma_{t'} - f_{\theta_v}(s_{t'}), \tag{5}$$

which indicates how much an action turns out be better than estimated. Lines 17-18 calculate the policy and values losses L_{θ_p} and L_{θ_v}, from which policy and value gradients can be derived. In particular, L_{θ_p} is defined as,

$$\begin{aligned} L_{\theta_p} = &-\frac{1}{N}\sum_{\chi_{t'}\in\Lambda}\delta_{t'}\log f_{\theta_p}(s_{t'},a_{t'}) \\ &-\frac{\alpha}{N}\sum_{\chi_{t'}\in\Lambda}\sum_{a\in A} f_{\theta_p}(s_{t'},a)\log f_{\theta_p}(s_{t'},a), \end{aligned} \tag{6}$$

where α $(0 < \alpha \ll 1)$ is a weighting coefficient. The rationale behind Eq 6 is to reinforce actions (i.e., improving the probabilities) with larger advantages while encouraging exploration (by introducing the total entropy of the policies as a secondary penalty term). The definition of the value loss is straightforward as the mean square error from value estimations, i.e.,

$$L_{\theta_v} = \frac{1}{N}\sum_{\chi_{t'}\in\Lambda}\left(f_{\theta_v}(s_{t'}) - \Gamma_{t'}\right)^2. \tag{7}$$

In line 19, the actor-learner applies the gradients to tune the global DNN parameters with training algorithms such as *RMSProp* or *Adam* [25]. Finally, the actor-learner empties the buffer and updates ε (line 20) to get prepared for the next episode.

Algorithm 1: Procedures of an actor-learner thread in DeepRMSA-EP

```
1  initiate Λ = ∅, ε = 1;
2  for each R_t do
3      if Λ == ∅ then
4          set θ_p = θ_p^*, θ_v = θ_v^*;
5      end
6      release the spectra occupied by expired requests;
7      obtain s_t with R_t and G(V, E, F);
8      calculate f_{θ_p}(s_t), f_{θ_v}(s_t);
9      calculate the cumulative sum of f_{θ_p}(s_t) as ζ;
10     decide an RMSA scheme a_t = arg min_a {ζ(a) ≥ rand()}
         with probability of ε, otherwise, a_t = arg max_a {f_{θ_p}(s_t)};
11     attempt to service R_t with a_t and receive a reward r_t;
12     store (s_t, a_t, f_{θ_v}(s_t), r_t) in Λ;
13     if |Λ| == N then
14         for each χ_{t'} = (s_{t'}, a_{t'}, f_{θ_v}(s_{t'}), r_{t'}) in Λ do
15             calculate Γ_{t'} and δ_{t'} with Eqs. 4 and 5;
16         end
17         calculate L_{θ_p} and L_{θ_v} with Eqs. 6 and 7;
18         obtain the policy and value gradients with L_{θ_p}, L_{θ_v};
19         apply the gradients to update θ_p^* and θ_v^*;
20         empty Λ and set ε = max{ε − ε_0, ε_min};
21     end
22 end
```

Note that, the uncertainty of dynamic lightpath requests can result in unpredictable trajectories of s_t, which in turn can cause oscillations of the cumulative rewards and destabilize the training process. This problem becomes especially severe when the numbers of requests involved are small. Recall the calculation of cumulative rewards in Eq. 4, $\Gamma_{t'}$ decreases when $\chi_{t'}$ is getting closer to the end of the buffer and eventually contains the reward from only one request. To cope with this

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2019.2923615, Journal of Lightwave Technology

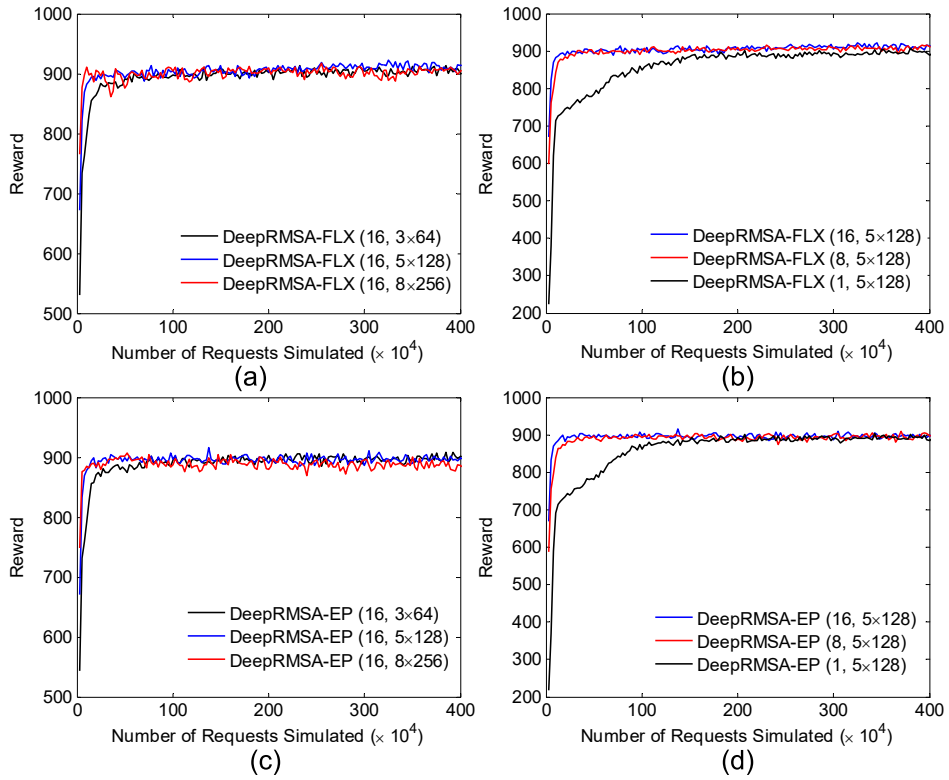

Fig. 4. Cumulative rewards from DeepRMSA-FLX and DeepRMSA-EP with different (a), (c): DNN sizes, and (b), (d): numbers of actor-learners.

issue, we propose a window-based flexible training mechanism for DeepRMSA, namely DeepRMSA-FLX. Basically, DeepRMSA-FLX invokes the training process each time the buffer contains $2N-1$ samples. DeepRMSA-FLX slides a window of length N through the buffer and calculates the cumulative reward for each of the first N samples, still with Eq. 4. Thus, every cumulative reward involves the rewards from servicing N requests. By doing so, we aim to smooth out the oscillations equally for all the samples (if N is sufficiently large[2]). Then, the algorithm calculates the policy and value losses with these N samples and updates the global DNN parameters accordingly. The N samples are removed from the buffer afterward. Meanwhile, the condition for synchronizing local DNNs (line 3 of *Algorithm* 1) becomes $|\Lambda|$ being equal to $N-1$ in DeepRMSA-FLX.

V. Evaluation

A. Simulation Setup

We evaluated the performance of DeepRMSA with numerical simulations. We first used the 14-node NSFNET topology in Fig. 3 and assumed that each fiber link could accommodate 100 FS's. The dynamic lightpath requests were generated according to a Poisson process following a uniform traffic distribution, with the average arrival rate and service duration being 10 and 25 time units, respectively. The bandwidth requirement of each request is evenly distributed between 25 and 100 Gb/s. The DNNs used *ELU* as the activation function for the hidden layers. We set γ, α, N, ε_0, ε_{min} and the learning rate as 0.95, 0.01, 50, 10^{-5}, 0.05, and 10^{-5}, respectively. We used the *Adam* algorithm [25] for training. Note that, we normalized every field of s_t before feeding it to the DNNs.

[2]Note that, we typically set N moderate values, e.g., 50, to allow training signals being applied to the DNNs quickly.

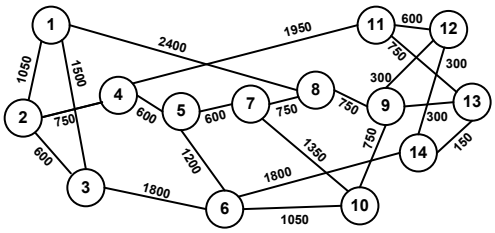

Fig. 3. 14-node NSFNET topology (link length in kilometers).

B. Comparison between Different DeepRMSA Configurations

We first assessed the impact of the scale of the DNNs on the performance of DeepRMSA. We set $K=5$ and $J=1$. Hence, DeepRMSA selected only the routing paths and applied the first-fit scheme for spectrum allocation (evaluations with different setups of K and J will be presented later). We fixed the number of actor-learners as 16, and implemented DNNs of three setups for both DeepRMSA-EP and DeepRMSA-FLX, i.e., 3 hidden layers of 64 neurons (3×64), 5 hidden layers of 128 neurons (5×128), and 8 hidden layers of 256 neurons (8×256). Figs. 4(a) and (c) show the evolutions of cumulative rewards (collected from every 1000 requests) with different DNN setups during training. We can see that for both of the algorithms, DNNs with larger scales facilitate faster training. In average, it requires simulating $750,000$ and $250,000$ requests for DeepRMSA to converge with DNNs of 3×64 and 5×128 (or 8×256), respectively. Eventually, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2019.2923615, Journal of Lightwave Technology

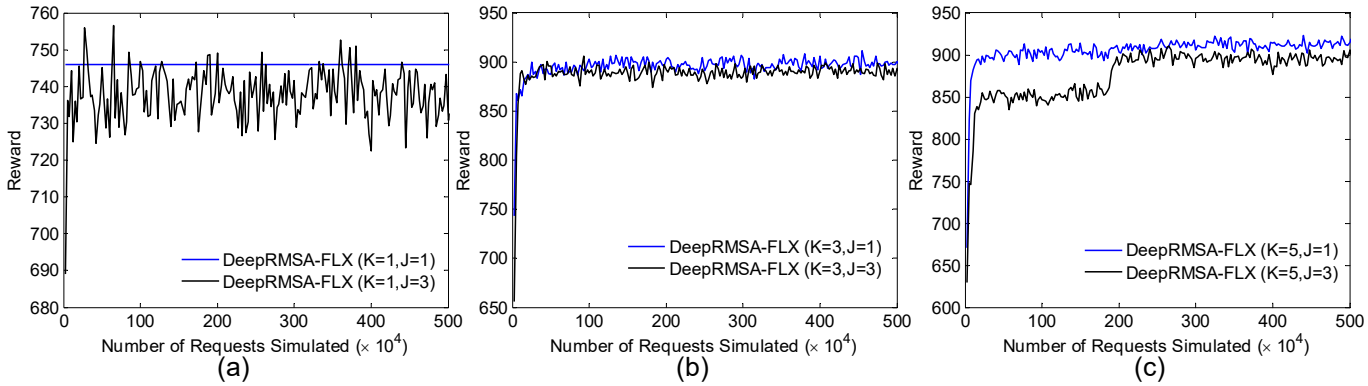

Fig. 5. Cumulative rewards from DeepRMSA-FLX with different J when: (a) $K = 1$, (b) $K = 3$, and (c) $K = 5$.

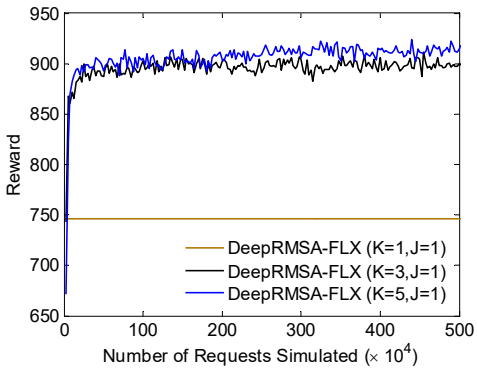

Fig. 6. Cumulative rewards from DeepRMSA-FLX with different K.

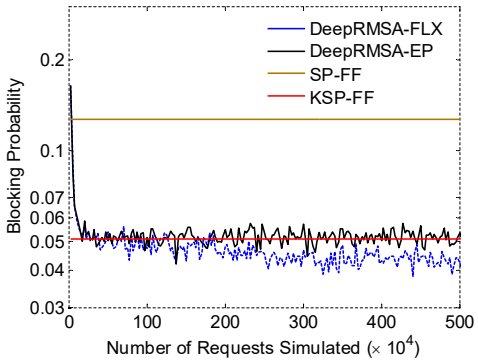

Fig. 7. Request blocking probability.

rewards associated with the three setups are very close, with 5×128 performing slightly better. This is because 5×128 enables a better ability of data representation when compared with 3×64, and in the meantime does not suffer from the overfitting issue as encountered by 8×256.

Then, we evaluated the impact of the number of actor-learners by fixing the sizes of the DNNs as 5×128 and implementing DeepRMSA with 1, 8 and 16 actor-learners. Figs. 4(b) and (d) show the corresponding evolutions of cumulative rewards. Again, we can draw the same observations from both of the algorithms, i.e., increasing the number of actor-learners leads to faster convergence and slightly higher rewards. In particular, increasing the number of actor-learners from 1 to 8 can accelerate the training speed by a factor of nearly 10 as multiple parallel actor-learners enable more diversified explorations of the problem. Since the performance gain from further increasing the number of actor-learners is marginal, we expect DeepRMSA with 16 actor-learners to achieve the best performance. Hence, we fixed the scale of the DNNs and the number of actor-learners as 5×128 and 16, respectively, for later evaluations.

We also conducted simulations with different setups of K and J to study how they affect the performance of DeepRMSA. Figs. 5(a)-(c) show the comparisons of the cumulative rewards from DeepRMSA-FLX with different setups of J. Note that, for the case where both K and J are equal to 1, DeepRMSA is reduced to a simple shortest-path routing and first-fit spectrum allocation (SP-FF) algorithm. We can see that, for all $K = 1$, 3, and 5, allowing more flexibility in spectrum allocation (i.e., increasing the value of J) does not benefit the performance of DeepRMSA. We presume that the reason is two-fold. First, a larger value of J means more information provided to the DNNs and larger action spaces, which increase the difficulty for DeepRMSA to learn successful RMSA policies. Second, when compared with the first-fit scheme, such flexibility can cause severer spectrum fragmentation [26] and downgrade the spectrum utilization in EONs. Next, we plotted the results with different setups of K in Fig. 6, fixing J as 1. The results indicate that increasing routing diversity can improve the performance of DeepRMSA effectively. We have also tested the performance of DeepRMSA-EP with different setups of K and J and observed similar results. For all the evaluations afterward, we set $K = 5$ and $J = 1$.

C. Comparison with Baseline Algorithms

We compared the performance of DeepRMSA-EP and DeepRMSA-FLX with that of the baseline algorithms, i.e., SP-FF and K-shortest-path routing and first-fit spectrum allocation (KSP-FF) [3]. KSP-FF has been shown to achieve the state-of-art performance among the existing heuristic designs [13]. For fair comparisons, we set $K = 5$ for KSP-FF. Fig. 7 plots the evolution of request blocking probability from the algorithms. We can see that DeepRMSA-EP and DeepRMSA-FLX perform similarly at the beginning and outperform SP-FF after a training period with only $50,000$ requests. However, DeepRMSA-FLX successfully beats KSP-FF after a training period with $1,875,000$ requests, whereas the performance of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2019.2923615, Journal of Lightwave Technology

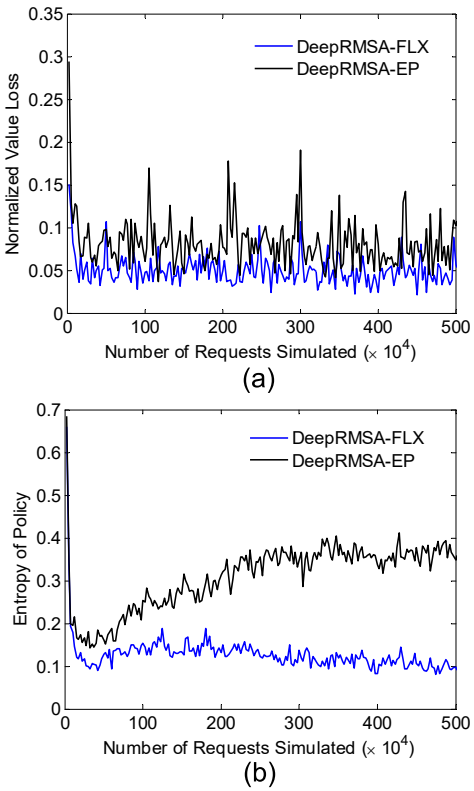

Fig. 8. (a) Normalized value loss, and (b) entropy of policy during training.

DeepRMSA-EP eventually merely fluctuates around that of KSP-FF. After training with $5,000,000$ requests, DeepRMSA-FLX can achieve a blocking reduction of 20.3% compared with KSP-FF. To reveal the rationale behind the behaviors of DeepRMSA-EP and DeepRMSA-FLX, Figs. 8(a) and (b) present the results of normalized value loss and entropy of policy during training, respectively. It can be seen that the proposed window-based training mechanism facilitates more accurate value estimations (lower value losses) and stabilized training, while the training of DeepRMSA-EP starts to diverge after $500,000$ requests being simulated. Note that, training periods with thousands of requests are too costly for practical network operations. A more efficient way of training DeepRMSA is expected to be performing offline training with an RMSA simulator first, before enrolling it in online lightpath provisioning for fine tuning [21].

D. Robustness Evaluation

All the previous evaluations assumed a uniform traffic distribution. To verify the robustness of DeepRMSA, we tested its performance by applying a nonuniform traffic model presented in [27] with the average request arrival rate and service duration set as 16 and 25 time units, respectively. Fig. 9 shows the results of blocking probability. Again, we can see that DeepRMSA-FLX entirely beats KSP-FF after training with $2,000,000$ requests and eventually achieves a blocking reduction of 11.4%. After simulations with $5,000,000$ requests, we switched the traffic distribution back to the uniform model defined in Section V-A to assess DeepRMSA's capability of adapting to traffic changes. It can be seen that the performance

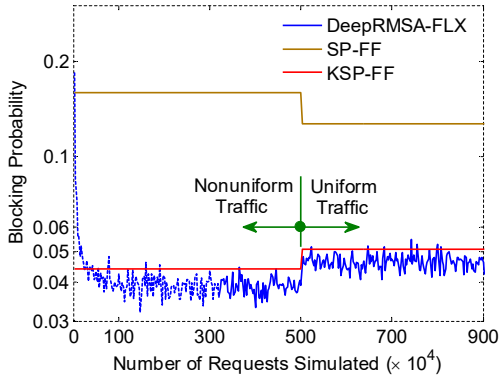

Fig. 9. Request blocking probability with nonuniform and uniform traffic distributions.

of DeepRMSA is not bound to traffic distributions and that by leveraging the previously learned knowledge, DeepRMSA still outperforms KSP-FF under the new traffic model.

Next, we performed simulations with the 11-node COST239 topology in Fig. 10(a). We used a uniform traffic distribution and set the average request arrival rate and service duration as 20 and 30 time units, respectively. All the rest of the parameters remained the same as those for the evaluations with the NSFNET topology. Fig. 10(b) shows the results of request blocking probability with the COST239 topology, which demonstrates a clear performance difference between DeepRMSA-EP and DeepRMSA-FLX. Eventually, DeepRMSA-FLX can achieve a blocking probability that is 14.3% and 18.9% lower than those of KSP-FF and DeepRMSA-EP, respectively.

VI. Conclusion

In this paper, we proposed DeepRMSA, a DRL-based RMSA framework for learning the optimal online RMSA policies in EONs. DeepRMSA parameterizes RMSA policies with DNNs and trains the DNNs progressively with experiences from dynamic lightpath provisioning. By taking into account the unique characteristics of the RMSA problem, we developed two training mechanisms for DeepRMSA based on the framework of A3C. Simulation results show that the proposed training mechanisms facilitate successful training of DeepRMSA, which can achieve blocking reductions of more than 20.3% and 14.3% in the NSFNET and COST239 topologies, respectively, when compared with the baselines.

An interesting future research topic would be partitioned DeepRMSA or hierarchical-DeepRMSA where multiple DeepRMSA agents cooperate hierarchically (within the same autonomous system) or interact peer-to-peer through brokers (in a multi-domain EON scenario [28]) to achieve scalability of DeepRMSA applied to topologies with larger scales. Meanwhile, multi-agent DeepRMSA applied to multiple autonomous system networks will introduce game-theoretic approaches similar to the discussions in [29], [30], thus yielding more interesting yet practical multi-agent competitive/cooperative learning problems.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2019.2923615, Journal of Lightwave Technology

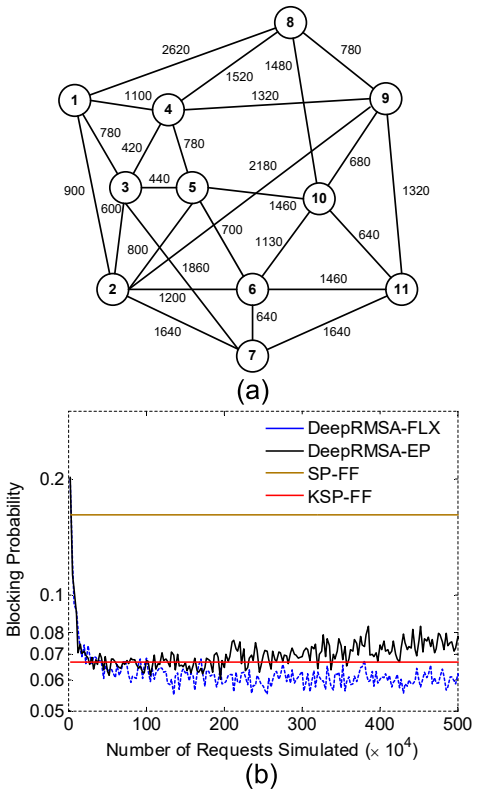

Fig. 10. (a) 11-node COST239 topology (link length in kilometers), and (b) request blocking probability with the COST239 topology.

ACKNOWLEDGMENTS

This work was supported in part by DOE DE-SC0016700, and NSF ICE-T:RC 1836921.

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, 2015.

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

[3] M. Jinno, B. Kozicki, H. Takara, A. Watanabe, Y. Sone, T. Tanaka, and A. Hirano, "Distance-adaptive spectrum resource allocation in spectrum-sliced elastic optical path network," *IEEE Commun. Mag.*, vol. 48, no. 8, pp. 138–145, Aug. 2010.

[4] 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.

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

[6] B. Chatterjee, N. Sarma, and E. Oki, "Routing and spectrum allocation in elastic optical networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1776–1800, thirdquarter 2015.

[7] Y. Wang, X. Cao, and Y. Pan, "A study of the routing and spectrum allocation in spectrum-sliced elastic optical path networks," in *Proc. of INFOCOM*, April 2011, pp. 1503–1511.

[8] 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.

[9] M. Klinkowski, M. Ruiz, L. Velasco, D. Careglio, V. Lopez, and J. Comellas, "Elastic spectrum allocation for time-varying traffic in flexgrid optical networks," *J. Sel. Areas Commun.*, vol. 31, no. 1, pp. 26–38, January 2013.

[10] L. Gong, X. Zhou, W. Lu, and Z. Zhu, "A two-population based evolutionary approach for optimizing routing, modulation and spectrum assignments (RMSA) in O-OFDM networks," *IEEE Commun. Lett.*, vol. 16, pp. 1520–1523, Sept. 2012.

[11] H. Wu, F. Zhou, Z. Zhu, and Y. Chen, "On the distance spectrum assignment in elastic optical networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2391–2404, Aug 2017.

[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, no. 1, pp. 15–22, Jan 2013.

[13] Y. Yin, H. Zhang, M. Zhang, M. Xia, Z. Zhu, S. Dahlfort, and S. J. B. Yoo, "Spectral and spatial 2D fragmentation-aware routing and spectrum assignment algorithms in elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, no. 10, pp. A100–A106, Oct 2013.

[14] F. Cugini, F. Paolucci, G. Meloni, G. Berrettini, M. Secondini, F. Fresi, N. Sambo, L. Poti, and P. Castoldi, "Push-pull defragmentation without traffic disruption in flexible grid optical networks," *J. Lightw. Technol.*, vol. 31, pp. 125–133, Jan. 2013.

[15] M. Zhang, C. You, and Z. Zhu, "On the parallelization of spectrum defragmentation reconfigurations in elastic optical networks," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2819–2833, October 2016.

[16] V. Mnih and K. Kavukcuoglu et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[17] V. Mnih, A. Badia, M. Mirza, A. Graves, T. Harley, T. Lillicrap, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. of Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[19] N. Luong, D. Hoang, S. Gong, D. Niyato, P. Wang, Y. Liang, and D. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *arXiv preprint arXiv:1810.07862*, 2018.

[20] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," *arXiv preprint arXiv:1801.05757*, 2018.

[21] S. Salman, C. Streiffer, H. Chen, T. Benson, and A. Kadav, "DeepConf: Automating data center network topologies management with machine learning," in *Proc. of NetAI*, 2018, pp. 8–14.

[22] B. Kozicki, H. Takara, Y. Sone, A. Watanabe, and M. Jinno, "Distance-adaptive spectrum allocation in elastic optical path network (SLICE) with bit per symbol adjustment," in *Proc. of OFC*, 2010, pp. 1–3.

[23] X. Chen, M. Tornatore, S. Zhu, F. Ji, W. Zhou, C. Chen, D. Hu, L. Jiang, and Z. Zhu, "Flexible availability-aware differentiated protection in software-defined elastic optical networks," *J. Lightw. Technol.*, vol. 33, pp. 3872–3882, Sept. 2015.

[24] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. of NIPS*, 2016, pp. 3844–3852.

[25] D. Kingma and J. Ba. (2014) Adam: A Method for Stochastic Optimization.

[26] M. Zhang, C. You, H. Jiang, and Z. Zhu, "Dynamic and adaptive bandwidth defragmentation in spectrum-sliced elastic optical networks with time-varying traffic," *J. Lightw. Technol.*, vol. 32, pp. 1014–1023, Mar. 2014.

[27] X. Chen, F. Ji, Y. Wu, and Z. Zhu, "Energy-efficient resilience in translucent optical networks with mixed regenerator placement," *J. Opt. Commun. Netw.*, vol. 5, no. 7, pp. 741–750, July 2013.

[28] S. J. B. Yoo, "Multi-domain cognitive optical software defined networks with market-driven brokers," in *Proc. of ECOC*, Sept. 2014, pp. 1–3.

[29] X. Chen, Z. Zhu, L. Sun, J. Yin, S. Zhu, A. Castro, and S. J. B. Yoo, "Incentive-driven bidding strategy for brokers to compete for service provisioning tasks in multi-domain SD-EONs," *J. Lightw. Technol.*, vol. 34, no. 16, pp. 3867–3876, 2016.

[30] X. Chen, Z. Zhu, J. Guo, S. Kang, R. Proietti, A. Castro, and S. J. 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.*, vol. 10, no. 2, pp. 1–9, 2018.