A Multi-Task-Learning-based Transfer Deep Reinforcement Learning Design for Autonomic Optical Networks

Xiaoliang Chen, Roberto Proietti, Che-Yu Liu, S. J. Ben Yoo, *Fellow, IEEE, Fellow, OSA*

***Abstract*—Deep reinforcement learning (DRL) enables autonomic optical networking by allowing the network control and management systems to self-learn successful networking policies from operational experiences. This paper proposes a transfer learning approach for effective and scalable DRL in optical networks. We first present a modular DRL agent design to facilitate retrieving and transferring relevant knowledge between tasks requiring different dimensions of network state data. In particular, we partition network state data into common states, which contain generic information critical to multiple tasks (e.g., the spectrum utilization on fiber links), and task-specific states that are only used by a specific task (e.g., the utilization of virtual network functions). Separate neural network blocks are employed to process different state data. Based on the modular agent design, a multi-task learning (MTL) aided knowledge transferring scheme is proposed. The proposed scheme trains an MTL agent that can master multiple tasks simultaneously and thus enables to learn and transfer better-generalized knowledge across tasks. We perform case studies on the proposed transfer DRL approach considering two scenarios, namely, knowledge transferring between routing, modulation and spectrum assignment (RMSA) tasks for different networks and knowledge transferring from RMSA tasks to anycast service provisioning tasks. The DRL designs for RMSA and anycast service provisioning, including the state, action, and reward formulations and the training mechanisms, are also elaborated. Performance evaluations under both scenarios show that the proposed approach can effectively expedite the training processes of the target tasks and improve the ultimate service throughput.**

***Index Terms*—Deep reinforcement learning, autonomic optical networking, transfer learning, multi-task learning, routing, modulation and spectrum assignment, anycast.**

I. Introduction

THE imminent era of 5G networking, along with the emerging edge computing technologies, are posing more stringent capacity, latency, and availability requirements on the underlying optical transport networks. These requirements entail a next-generation optical networking paradigm that can support dynamic and adaptive service provisioning with high resource efficiency and guaranteed quality of service. By virtue of the capability of flexible spectrum manipulation in the physical layer, elastic optical networking (EON) has become an appealing solution [1]. The past few years have witnessed a tremendous amount of research efforts invested in the control, management, and service provisioning design for EON, with various mathematical optimization models and heuristic algorithms being proposed [2]–[5]. While these optimization models are typically too complicated to be applied in practical network operations, the heuristic designs mostly rely on fixed and artificially defined rules, which can suffer from poor adaptability (i.e., unable to adapt to evolving network conditions) or lead to suboptimal resource utilization.

Recently, thanks to a number of exciting breakthroughs [6], machine learning (ML) has become a cynosure of the communication and networking community. By allowing machines to learn complex functions automatically from big data without being explicitly programmed, ML makes it possible to revolutionize the current fixed-rule-based networking principles and to pursue data and knowledge-defined cognitive optical networking [7]. Driven by this goal, researchers have investigated ML designs for a variety of tasks in optical networks, such as optical performance monitoring (OPM) [8]–[10], fault management [11]–[13], and resource allocation [14]–[17]. Most of these existing works apply supervised or unsupervised learning schemes, focusing on learning a specific network or system rule (e.g., quality-of-transmission (QoT) modeling [8], traffic dynamics [14]). As a result, the ML designs serve as a complement to the traditional approaches, and cognitive networking operations are eventually achieved through the combination of them. For instance, the work presented in [15] makes use of an integer linear programming (ILP) model to compute impairment-aware routing, modulation, and spectrum assignment (RMSA) solutions in elastic optical networks (EONs), where impairments are evaluated by an ML-based QoT estimator. Other works, for instance, the one demonstrated in [17], take advantage of the optimal network configuration policies returned by traditional optimization tools (e.g., ILP) to train supervised ML models that can respond quickly during online operations. However, such approaches typically suffer from scalability issues owing to the high complexity of the optimization tools.

Unlike supervised or unsupervised learning, which requires a large amount of data available for training, deep reinforcement learning (DRL) parameterizes operational policies by deep neural networks (DNNs) and learns the optimal policies through direct interactions with the system environments [18], [19]. This makes DRL particularly useful for tackling complex

X. Chen was with the Department of Electrical and Computer Engineering, University of California, Davis, Davis, CA 95616, USA. He is now with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510275, China (Email: xlichen@ieee.org).

R. Proietti, C. Liu and S. J. B. Yoo are with the Department of Electrical and Computer Engineering, University of California, Davis, Davis, CA 95616, USA (Email: {rproietti, cyliu, sbyoo}@ucdavis.edu).

Manuscript received June 30, 2020, revised November 21, 2020.

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/JSAC.2021.3064657, IEEE Journal on Selected Areas in Communications

online decision-making problems. The application of DRL to optical networks enables development of autonomic service provisioning schemes that can self-learn and adapt networking policies from repeated operation experiences [20]–[27]. Thus, DRL can potentially facilitate enhanced network adaptability and resource efficiency while largely eliminating human interventions. Nevertheless, training DRL models is never a trivial task, often demanding a significant amount of computing power or being very time-consuming [18]. Besides, DRL models are vulnerable to converging to suboptimal solutions when confronting complex and unstable systems. It is desirable to address such challenges for improving the applicability of the autonomic applications in optical networks.

Among the various ML techniques, transfer learning (TL) was proposed to improve the effectiveness and to reduce the cost of the training of ML models, by exploiting and reusing relevant knowledge obtained from similar tasks [28]. In particular, recent studies have reported several TL designs in the DRL domain (i.e., transfer DRL) and demonstrated promising benefits from the TL designs in facilitating robotic [29], [30] or computer gaming tasks [31]. Therefore, we envision transfer DRL to be a key enabling technology for realizing scalable autonomic optical networking, which, however, remains underexplored.

In this paper, we extend our previous conference paper in [32] and propose a more comprehensive transfer DRL design for optical networks. First, we present a modular agent structure, where network state data are divided into common and task-specific states and a separate neural network block is used to processes each state partition. The modular structure thereby facilitates retrieving and transferring relevant knowledge between tasks requiring different dimensions of state information. Then, we developed a multi-task learning (MTL) aided knowledge transferring scheme for transferring better-generalized knowledge. This is achieved by training an MTL agent exploiting the knowledge obtained from multiple source tasks simultaneously. Besides the scenario of knowledge transferring between RMSA tasks for different networks, which has been studied in [32], we also investigate the application of the proposed transfer DRL design to a more challenging scenario where knowledge is transferred from RMSA tasks to anycast service provisioning tasks. We present the detailed DRL design for RMSA, including the state, action, and reward formulation and the training mechanism, and further extend the design to enable autonomic anycast service provisioning by devising a dual-agent cooperative learning scheme. We have augmented the simulation setup presented in [32] with more exhaustive numerical simulations that evaluate *i*) the performance of transferring human knowledge to RMSA agents, *ii*) the sensitivity of the proposed design to the selection of source and target tasks, and *iii*) the performance of the proposed design in transferring knowledge from RMSA to anycast service provisioning tasks.

The rest of the paper is organized as follows. In Section II, we provide a brief review of related works. In Section III, we present an autonomic optical networking architecture enabled by DRL. Based on the architecture, we formulate the problem of transfer DRL and elaborate on the modular agent structure as well as the MTL-aided knowledge transferring scheme in Section IV. In Section V, we show the case studies on the proposed design. Finally, we provide the performance evaluations and related discussions in Section VI and conclude the paper with Section VII.

II. Related Work

Existing works on ML-aided cognitive optical networking mostly focus on QoT estimation, OPM, fault management, and resource allocation. In [8], the authors developed a Random Forest classifier that can accurately predict whether the bit-error-rate of an unestablished lightpath can satisfy the system requirement, taking as input various network configurations, such as the traffic volume, the desired routing path and modulation format. In [9], the effectiveness of different ML models, i.e., K-nearest neighbors, logistic regression, support vector machines (SVM), and artificial neural networks (ANNs), on QoT estimation were evaluated. The results indicate that all the ML models can achieve high estimation accuracy while ANNs offer better generalization capabilities. Vela et al. developed two cognitive algorithms in [11] for detecting and identifying the root causes of bit-error-rate degradations due to signal overlap, tight filtering, gradual drift, and cyclic drift. The authors of [12] evaluated two ML models, based on SVM and ANNs, respectively, in detecting and identifying jamming signal attacks with different power intensities. Our previous work in [13] proposed a hybrid supervised/unsupervised learning scheme for anomaly detection in optical networks. Specifically, a data clustering module is first used to screen out abnormal network behaviors from a large amount of unlabeled OPM data, whereafter, a DNN-based data regression module is trained with the obtained knowledge for online anomaly detection. In [14], the authors proposed to realize adaptive virtual network topology reconfiguration by leveraging a data analytics model for predicting the traffic dynamics of optical core networks. A traffic predictor based on long/short-term memory neural networks was developed in [16]. Taking advantage of the traffic predictor, the authors then devised cross-layer orchestration algorithms to optimize lightpath reconfiguration decisions. More recently, in [17], [33], the authors proposed an ML-based classifier design that can learn near-optimal routing and wavelength assignment policies from data generated by ILP models and experimentally demonstrated the integration of the design with an ONOS-based software-defined networking (SDN) testbed.

The aforementioned works mostly apply supervised or unsupervised learning schemes. Lately, several works have studied DRL designs for autonomic optical networking. In [20], we proposed an autonomic RMSA framework based on the asynchronous advantage actor-critic (A3C) algorithm and demonstrated performance superior to that of the state-of-the-art heuristic approaches. The authors of [21] developed a reinforcement learning scheme for predictive bandwidth allocation in EON, i.e., learning the policies of determining the number of frequency slots (FS's) to allocate. In [22], Luo et al. targeted a survivable EON design with DRL and proposed a double-agent scheme for mastering the working and protection

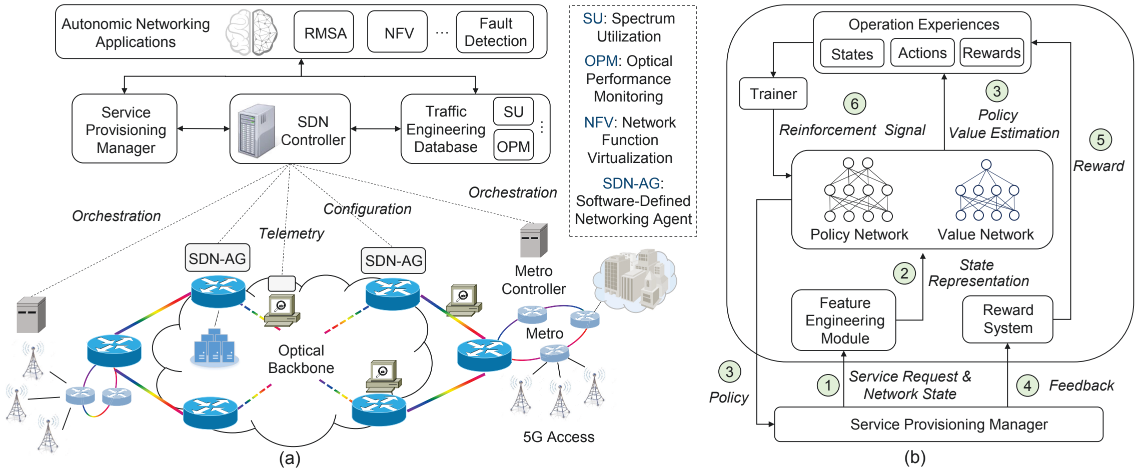

Fig. 1. (a) Autonomic optical networking architecture, and (b) operation principle of an autonomic networking application.

lightpath configurations. In [23], the authors investigated a DRL approach for dynamic slice admission in 5G flexible radio access networks, to maximize network revenue gains. An ingenious network state representation scheme for DRL-based routing in optical transport networks (OTNs) was developed in [24]. The same team later extended the work by incorporating graph neural networks to enhance the generality of the DRL design over different OTN topologies [25]. In [27], Li et al. employed a DRL design similar to that in [20] for determining the duration of service cycles in provisioning virtual network function (vNF).

There have been a few studies on TL for optical networks, but all based on supervised learning. In [34], Yu et al. adopted an ANN-based QoT estimator design, and investigated TL schemes for optical systems with different modulation formats. The results show that TL can remarkably reduce the number of additional data instances required for training a new QoT estimator. A similar research can be found in [35]. The authors of [36] applied TL to assist the training of ML models used to predict the spectrum defragmentation time in space-division multiplexing EONs. In [37], we proposed an evolutionary TL approach to facilitate the QoT estimation tasks for inter-domain lightpaths, where a genetic algorithm is devised to explore the proper sets of knowledge to transfer.

For more comprehensive surveys or tutorials on the application of ML in optical networking and systems, the readers can refer to [38]–[40].

III. Autonomic Optical Networking Architecture

Fig. 1(a) depicts an autonomic optical networking architecture, which is built upon SDN-based network control and management. In the data plane, an optical backbone network carries the traffic aggregated from metro networks to provide interconnectivity among access users, data centers (DCs), and research facilities at different geographical locations. An SDN controller is employed to control the data plane equipment (e.g., optical switches, transponders) of the backbone in a centralized manner. The SDN controller can fetch network state data (e.g., spectrum utilization status, OPM data) on demand using telemetry services [41] and distribute device configuration commands according to the instructions from the service provisioning manager. The SDN controller can also work with metro, access, and DC controllers to realize end-to-end service provisioning across heterogeneous domains. On top of the SDN controller, a set of autonomic networking applications are deployed, forming a knowledge plane. Each autonomic application makes use of a DRL agent to learn the policies for a particular networking task, for instance, RMSA.

Fig. 1(b) shows the operation principle of an autonomic networking application. Specifically, at each operation step (e.g., upon receiving a service request), the service provisioning manager retrieves necessary network state data from the traffic engineering database and invokes the corresponding DRL agent. The feature engineering module of the agent first transforms the data into a state representation that can be recognized by the DNNs employed. The DNNs read the state representation and compute an action policy and a value estimation. The action policy can be a probability distribution over a set of possible actions (e.g., candidate routing paths), or can return an action directly (e.g., determining the amount of bandwidth to allocate), depending on the DRL approach used. The value estimation is used for the training purpose, which will be discussed in the subsequent Sections. The service provisioning manager hereby decides a service scheme to use, whereafter, the controller configures the data plane equipment accordingly (if necessary). Afterwards, the controller provides feedback to the reward system (i.e., indicating key network performance indicators related to the task), which in turn computes a numerical reward for the agent. The agent records the operation experience in a buffer and constantly trains the DNNs by reinforcing actions leading to higher long-term rewards. As such, the agent can progressively learn effective policies and meanwhile adapt to gradual network changes[1].

IV. Transfer DRL Design

In this section, we formulate the problem of transfer DRL for autonomic optical networking and elaborate on the modular

[1] A practical way of deploying autonomic networking applications could be pretraining the DRL agents with network emulators prior to adopting them in online operations [42].

agent structure and the MTL-aided knowledge transferring scheme proposed.

A. Problem Formulation

Let $\mathbb{M}_{src}$ and $\mathbb{M}_{tar}$ represent the sets of source and target DRL tasks (autonomic networking applications), respectively. Each task $M_i(S_i, \Lambda_i, r_i, \mathcal{P}_i) \in \{\mathbb{M}_{src}, \mathbb{M}_{tar}\}$ can be modeled as a Markov decision process, with S_i, Λ_i, r_i, and $\mathcal{P}_i$ denoting its unique state space, action space, reward function, and state transition function, respectively. More specifically, $r_i(s_t, a_t)$ defines the reward a DRL agent can obtain by taking action $a_t \in \Lambda_i$ given state observation $s_t \in S_i$ at system time t. $\mathcal{P}_i(s_{t+1}|s_t, a_t)$ characterizes the system dynamics, indicating the distribution of a next state s_{t+1} as a function of s_t and a_t. Each DRL agent parameterizes a policy function $\pi_i = f_{\theta_{\pi_i}}(a, s_t)$ $(a \in \Lambda_i)$ and a value function $Q_i = f_{\theta_{Q_i}}(s_t)$ with DNNs, where θ_{π_i} and θ_{Q_i} signify the respective sets of parameters of the DNNs. π_i guides the action selection by the agent upon each observation s_t, while Q_i provides an estimation of the long-term reward expectation following the state transition function of the task $(s_{t+\tau} \sim M_i)$ and the current policy $(a_{t+\tau} \sim \pi_i)$, i.e.,

$$Q_i(s_t) = \mathbb{E}_{s_{t+\tau} \sim M_i, a_{t+\tau} \sim \pi_i}\left[\sum_{\tau=0}^{\infty} \gamma^{\tau} r_i(s_{t+\tau}, a_{t+\tau})\right], \quad (1)$$

Here, $\gamma \in [0, 1]$ is a discounting factor used for reducing the values of future rewards $(\tau > 0)$. The goal of each DRL agent is to learn the optimal sets of parameters (denoted as $\theta^*_{\pi_i}$ and $\theta^*_{Q_i}$) so that $Q_i(s_t)$ is maximized. Different training algorithms can be used to achieve such a goal. For instance, with the state-of-the-art A3C algorithm [19], $\theta^*_{\pi_i}$ and $\theta^*_{Q_i}$ can be approximated by minimizing the following policy and value loss functions every time N instances (e.g., $t_0 \rightarrow t_0 + N - 1$) are collected.

$$\begin{aligned} L_{\theta_{\pi_i}} =& -\frac{1}{N}\sum_{t\in[t_0, t_0+N-1]} \delta_t \log f_{\theta_{\pi_i}}(a_t, s_t) \\ & -\frac{\alpha}{N}\sum_{t\in[t_0, t_0+N-1]}\sum_{a\in\Lambda_i} f_{\theta_{\pi_i}}(a, s_t) \log f_{\theta_{\pi_i}}(a, s_t), \end{aligned} \quad (2)$$

$$\begin{aligned} L_{\theta_{Q_i}} =& \frac{1}{N}\sum_{t\in[t_0, t_0+N-1]} (\sum_{\tau\in[t, t_0+N-1]} \gamma^{\tau-t} r_i(s_\tau, a_\tau) \\ & + \gamma^{t_0+N-t} Q_i(s_{t_0+N}) - f_{\theta_{Q_i}}(s_t))^2. \end{aligned} \quad (3)$$

Wherein, $f_{\theta_{\pi_i}}(a, s_t)$ returns a probability of taking action a, and δ_t is the advantage of the action taken, which can be calculated as,

$$\delta_t = \sum_{\tau\in[t, t_0+N-1]} \gamma^{\tau-t} r_i(s_\tau, a_\tau) + \gamma^{t_0+N-t} Q_i(s_{t_0+N}) - Q_i(s_t). \quad (4)$$

In other words, δ_t indicates how much an action turns out to be better than expected. By minimizing the policy loss, the probabilities of actions with larger advantages are increased. Meanwhile, the average policy entropy weighted by $\alpha(0 < \alpha \ll 1)$ (i.e., the second term of the policy loss) is used to encourage exploration. The value loss is defined straightforwardly as the mean square error between the estimated and the observed rewards.

Let us assume that $\theta^*_{\pi_i}$ and $\theta^*_{Q_i}$ have been successfully learned for tasks in $\mathbb{M}_{src}$ and that tasks in $\mathbb{M}_{tar}$ share certain similarities with those in $\mathbb{M}_{src}$. For instance, the source and target tasks all involve routing and spectrum allocation procedures. The goal of this work is to investigate effective transfer DRL designs to facilitate the training of $\mathbb{M}_{tar}$ by exploiting the knowledge learned for $\mathbb{M}_{src}$ (i.e., $\theta^*_{\pi_i}$ and $\theta^*_{Q_i}$).

B. Modular Agent Structure

Different autonomic networking applications can require different amounts of network state information, leading to different dimensions of state spaces for the DRL agents. For instance, compared with an RMSA agent that principally works with the link spectrum utilization information, a DRL agent for service function chain provisioning also demands the utilization states of vNFs deployed at various network locations. In principle, extracting useful features from heterogeneous state information entails different knowledge. Thus, it is essential to distill relevant (and right) knowledge for realizing effective knowledge transferring, which is, however, difficult with a conventional DRL agent design that makes use of a unified neural network block to process all the state inputs jointly. In this context, we propose to leverage a modular agent structure as shown in Fig. 2(a). In particular, we partition the state space of each task (without loss of generality, task M_A) into a common space $S_{A,O}$ and a task-specific space $S^-_{A,O}$. Each $s_{A,O} \in S_{A,O}$ conveys common information used by multiple tasks (e.g., spectrum utilization), whereas $s^-_{A,O} \in S^-_{A,O}$ carries state data that are only to be mined by the specific task (e.g., vNF utilization). State partitioning can be conducted manually by network experts, or, through an automated process where we model each type of state data as an element and find the maximal set of shared elements between tasks to form the common space. Separate neural network blocks, $f_{\theta_{A,O}}(\cdot)$ and $f_{\theta^-_{A,O}}(\cdot)$, are employed to extract high-level features from $s_{A,O}$ and $s^-_{A,O}$, respectively, where $\theta_{A,O}$ and $\theta^-_{A,O}$ represent the corresponding sets of parameters. Note that, the architecture of $f_{\theta_{i,O}}(\cdot)$ of every task $M_i \in \{\mathbb{M}_{tar}, \mathbb{M}_{tar}\}$ should be identical. On top of $f_{\theta_{A,O}}(\cdot)$ and $f_{\theta^-_{A,O}}(\cdot)$, we add another two neural network blocks as the policy and value heads, which combine the extracted features to compute π_A and Q_A. Alternatively, we can decouple the policy and value functions by structuring separate DNNs with architectures similar to that in Fig. 2(a) but only having a single policy or value head. The modular agent structure allows us to easily identify the correct knowledge (e.g., $\theta^*_{A,O}$) to transfer between tasks.

C. Knowledge Transferring Schemes

With the modular agent structure, a straightforward knowledge transferring scheme is directly copying $\theta^*_{A,O}$ of a source task $M_A \in \mathbb{M}_{src}$ to $\theta_{B,O}$ of a target task $M_B \in \mathbb{M}_{tar}$ while leaving the rest of the parameters randomly initialized. Afterward, M_B can be trained with a classic training algorithm taking the derived DNN(s) as a starting point. Note that, the weights copied from $\theta^*_{A,O}$ can either be locked or unlocked

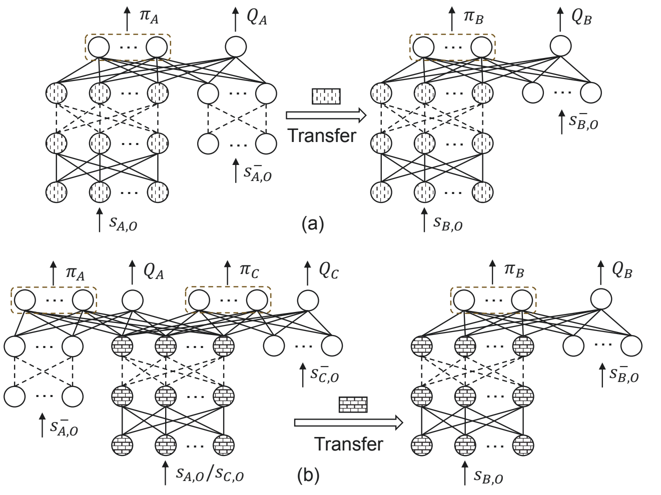

Fig. 2. An illustrative example for the proposed transfer DRL design: (a) straightforward knowledge transferring between two DRL agents (from M_A to M_B), and (b) multi-task learning aided knowledge transferring (from M_A and M_C to M_B).

(i.e., trainable) during the training of M_B, and in this work, we make all the weights unlocked for performance considerations [43]. Fig. 2(a) shows an illustrative example of the straightforward knowledge transferring scheme. Since the optimal feature extractors of different tasks for the same state information (i.e., $f_{\theta^*_{A,O}}(\cdot)$ and $f_{\theta^*_{B,O}}(\cdot)$) will likely share certain structural similarities, we can potentially approach $\theta^*_{B,O}$ more quickly by fine-tuning $\theta^*_{A,O}$ with fewer samples. On the other hand, it should be noted that the knowledge learned for a source task can be overfitted or biased. In the situations where the differences between source and target tasks are significant, for instance, $s_{A,O}$ and $s_{B,O}$ have very different distributions, the benefit of transferring $\theta^*_{A,O}$ can be marginal or even negative.

To overcome the deficiency of the straightforward knowledge transferring scheme, we devise an MTL-aided knowledge transferring scheme as illustrated by Fig. 2(b). We slightly extend the agent structure in Fig. 2(a) by integrating the DNNs of multiple source tasks (tasks M_A and M_C in the example) into a single MTL agent. In particular, we let M_A and M_C share a common neural network block $f_{\theta_{A/C,O}}(\cdot)$ for $s_{A,O}$ and $s_{C,O}$, while the task-specific neural network blocks (including the policy and value heads) remain unchanged. The MTL agent is first trained to master both M_A and M_C, whereafter, $\theta^*_{A/C,O}$ is transferred to M_B in a way similar to that of the straightforward transferring scheme. By leveraging MTL to exploit the knowledge obtained from multiple source tasks, we can potentially learn and transfer better-generalized knowledge across tasks for improved transferring effectiveness.

The MTL agent can be trained through either a DRL or a supervised learning approach. With a DRL approach, we can make the agent interact with the system environments of M_A and M_C in parallel while applying a classic training algorithm used for the single-agent scenario. More specifically, the MTL agent can maintain a separate experience buffer for each task. At every training step, the agent picks samples related to M_A or M_C and only tunes the set of associated parameters. However, successfully training an MTL agent with a DRL approach can be challenging. This is because the gradients from one task can interfere with the training of other tasks, compromising the performance of individual tasks or even leading to tasks being dominated [44], [45]. Moreover, such an approach requires a specific MTL environment (often unavailable) where the multiple source tasks are present simultaneously and learn at roughly the same pace (i.e., generating gradients of the same order of magnitude during each period). Therefore, in this study, we alternatively apply a more flexible supervised learning scheme inspired by the actor-mimic approach presented in [31]. In particular, we first train DRL agents for M_A and M_C independently. Then, we sample a long state trajectory over the learned policy from each of the source tasks, i.e., $\mathcal{D}_i = \{(s, \pi_i, Q_i)|_{s\sim M_i, a\sim\pi_i}\}$ ($i \in \{A, C\}$). We augment the obtained data instance with proper paddings to fit them to the DNN structure of the MTL agent. For instance, we expand an instance (s, π_A, Q_A) to be $([s, s^-_{C,O} = \mathbf{0}], [\pi_A, \pi_C = \mathbf{0}], [Q_A, Q_C = 0])$ with zero paddings for all the elements with respect to M_C. Similarly, we obtain an augmented instance $([s^-_{A,O} = \mathbf{0}, s], [\pi_A = \mathbf{0}, \pi_C], [Q_A = 0, Q_C])$ from $(s, \pi_C, Q_C(s))$. Let $\mathcal{D} = \{\mathcal{D}_A, \mathcal{D}_C\}$, we train the DNN in Fig. 2(b) with $\mathcal{D}$ by minimizing the overall loss given by,

$$\mathcal{L}_\theta(\mathcal{D}) = \alpha_1 \cdot \mathcal{L}_{\theta_{\pi_A}, \theta_{\pi_C}}(\mathcal{D}) + \alpha_2 \cdot \mathcal{L}_{\theta_{Q_A}, \theta_{Q_C}}(\mathcal{D}), \qquad (5)$$

where, the first and second terms signify the policy and value losses, respectively, α_1 and α_2 are weighting factors, θ is the set of all the parameters, $\theta_{\pi_i} = \{\theta_{A/C,O}, \theta^-_{i,O}, \theta_{\pi_i,H}\}$, and $\theta_{Q_i} = \{\theta_{A/C,O}, \theta^-_{i,O}, \theta_{Q_i,H}\}$, $i \in \{A, C\}$. $\theta_{\pi_i,H}$ and $\theta_{Q_i,H}$ represent the sets of parameters of the respective policy and value heads. The policy and value losses can be calculated as,

$$\mathcal{L}_{\theta_{\pi_A}, \theta_{\pi_C}}(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{n=1}^{|\mathcal{D}|} \sum_{i \in \{A,C\}} \sum_{a \in \Lambda_i} -g_i^n \pi_i^n(a) \log f_{\theta_{\pi_i}}(a, s^n), \qquad (6)$$

$$\mathcal{L}_{\theta_{Q_A}, \theta_{Q_C}}(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{n=1}^{|\mathcal{D}|} \sum_{i \in \{A,C\}} g_i^n \left(Q_i^n - f_{\theta_{Q_i}}(s^n)\right)^2, \qquad (7)$$

where, g_i^n is an indicator which is equal to 1 if the instance belongs to $\mathcal{D}_i$, and 0 otherwise. We can see that by minimizing Eq. 5, we make the MTL agent mimic the learned policy and value functions of M_A and M_C simultaneously, and thus, master both tasks. The supervised learning approach enables us to rely on simply the existing DRL frameworks and to freely combine different source tasks. Note that, an appropriate source task selection is critical to the performance of knowledge transfer as the usefulness of the learned knowledge is highly dependent on the similarities between source and target tasks. A potential solution for effective source task selection could be devising metrics to quantitatively measure the similarities between tasks and select tasks having higher similarity scores to a target task as the source tasks, similar to the idea presented in [46]. Meanwhile, it is desirable to maintain the diversity of source tasks as otherwise, the problem reduces to a single source task one. We will consider and investigate a more comprehensive design for source task selection as one of our future research tasks.

V. Case Studies

In this section, we present the case studies on the proposed transfer DRL design with two agent-to-agent knowledge transferring scenarios, i.e., knowledge transferring between RMSA agents for different networks and knowledge transferring from RMSA agents to agents for anycast service provisioning (between applications).

A. Knowledge Transferring between RMSA Agents

Learning effective RMSA policies is imperative for achieving spectrum-efficient service provisioning in EONs. In our previous work [20], we have demonstrated a successful DRL-based RMSA agent design outperforming the state-of-the-art heuristic approaches. However, training the RMSA agents can be time-consuming. It is necessary to study knowledge transferring between RMSA agents to enable prompt training of agents for new networks, for instance, in the case of network upgrades or failures/disasters where the effectiveness of the learned RMSA policies can deteriorate remarkably (due to the dramatic changes of network conditions, e.g., topology), or, when an operator provisions virtual optical networks (VONs) with different topologies and traffic profiles [3], [47].

Agent design: we slightly modify the RMSA agent design in [20] based on the modular agent structure discussed in Section IV-B. Let $G(V, E)$ denote the network topology of M_i, where V and E represent the sets of nodes and links, respectively. Meanwhile, let $R(v_{orig}, v_{dst}, bw, \mu)$ denote a lightpath request, with v_{orig} and v_{dst} being the origin and destination nodes, bw being the required bandwidth (in Gb/s), and μ being the service duration. Based on the fact that the spectrum utilization information is the most fundamental state in EON [2], [3], [15], [27], we make $s_{i,O}$ containing the information of the spectrum utilization state on K shortest candidate routing paths, while $s^{-}_{i,O}$ is composed of the information of R (v_{orig} and v_{dst} in one-hot forms). Therefore, the scale of $s^{-}_{i,O}$ depends on $|V|$. For each path, we compute five key features of spectrum state, including *i*) the number of FS's required based on the modulation format determined according to the impairment-aware model in [48] and the calculation of Eq. 1 in [20], *ii*) the total number of available FS's, *iii*) the average size of available FS blocks (i.e., blocks of consecutive available FS's), and *iv*) the size and *v*) starting position of the first available FS block. In total, $s_{i,O}$ contains $5K$ elements. We adopt a fully-connected neural network block of five hidden layers ($[128, 128, 128, 128, 128]$[2]) for $f_{\theta_{i,O}}(\cdot)$ and single-layer (also fully-connected) policy and value heads taking as input the concatenation of $s^{-}_{i,O}$ and $f_{\theta_{i,O}}(s_{i,O})$. The hidden layers use ELU as the activation function. An RMSA agent selects one from the K path candidates as the routing path for R and applies the first-fit spectrum allocation scheme afterward.

Reward function: the objective of an RMSA task is to maximize the number of requests successfully serviced during long-term network operations. Thus, we assign an agent a reward of 1 if R is successfully serviced, and otherwise -1.

Training: the DeepRMSA-FLX algorithm in [20] is used for training. We apply the A3C algorithm (see Eqs. 1-4) and adopt the following modifications. First, we make an RMSA agent optimize the total discounted reward within a window of length N (instead of ∞ as in Eq. 1) at each step t. The reason for doing so is two-fold: *i*) we make the objective value bounded, *ii*) we eliminate the term $Q_i(s_{t_0+N})$ from Eqs. 3 and 4, as the future state s_{t_0+N} is unknown (due to the uncertainty of further requests). Then, we perform training every time $2N-1$ samples are recorded but only the first N samples will contribute to the policy and value losses. This way, we can easily obtain the target value and advantage regarding each sample using the observed rewards.

Knowledge transferring: given the aforementioned RMSA agent design, we apply knowledge transfer from RMSA agents which have been trained to RMSA agents for unseen networks.

B. Knowledge Transferring between Applications

A more challenging scenario is knowledge transferring between applications where the differences between tasks are more prominent. In this work, we investigate specifically the case where we leverage the knowledge learned by RMSA agents to facilitate the training of DRL agents for anycast service provisioning tasks in EONs. The problem of anycast service provisioning can be defined as follows. Let V_{dc} be the set of DC nodes where computing resources are deployed and $R_{ac}(v_{orig}, bw, cp, \mu)$ denote an anycast service request demanding cp units of computing resources. The network operator provisions each R_{ac} by allocating sufficient amount of computing resources at a DC node $v \in V_{dc}$ and by establishing a lightpath with bandwidth of bw to connect v_{orig} and v [49]. Either cp or bw not being able to be satisfied will lead to service blocking. Therefore, an anycast service provisioning task can be seen as composed of two correlated subtasks, i.e., DC node (or destination) selection and RMSA. Realizing effective anycast service provisioning in EONs requires the joint optimization of routing, spectrum and computing resource allocations.

Agent design: the state space of an anycast service provisioning task is much larger than that of an RMSA task as we need to feed the related agent the spectrum utilization information on routing paths to multiple potential destinations (in contrast, a single destination for an RMSA task). To facilitate knowledge transferring from RMSA agents and to achieve enhanced model scalability to the size of V_{dc}, we propose a dual-agent cooperative learning scheme for autonomic anycast service provisioning. In particular, we divide the provisioning of R_{ac} into two stages. First, an RMSA agent is used to determine a routing path for each v_{orig} and $v \in V_{dc}$ pair by selecting one from the K candidates. The RMSA agent essentially elects a set of most promising solutions according to the spectrum utilization state, reducing the size of the action space from $K \times |V_{dc}|$ to $|V_{dc}|$, where $|V_{dc}|$ represents the number of nodes in V_{dc}. Then, in the second stage, we employ a master agent aiming at learning an optimal policy over the $|V_{dc}|$ candidate solutions generated by the RMSA agent, with which the eventual service scheme for R_{ac} can

[2] The neural network architecture is decided based on the architectural study performed in [20].

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/JSAC.2021.3064657, IEEE Journal on Selected Areas in Communications

be decided. The architecture of the master agent resembles that of the RMSA agent, with $s_{i,O}$ conveying the resource utilization state on the $|V_{dc}|$ candidate paths and $s^-_{i,O}$ carrying the information of R_{ac} and the computing resource utilization of each $v \in V_{dc}$. We adopt a fully-connected architecture with five hidden layers of 128 neurons for both $f_{\theta_{i,O}}(\cdot)$ and $f_{\theta^-_{i,O}}(\cdot)$. Slightly different from the RMSA agent design, we stack $f_{\theta_{i,O}}(s_{i,O})$ and $f_{\theta^-_{i,O}}(s^-_{i,O})$ to get a feature matrix of 2×128 and process the feature matrix using a convolutional layer with one filter (size $= 2 \times 1$, stride $= 1$) before employing the final fully-connected policy and value layers. The convolutional layer acts as a coupler that combines the features extracted from the spectrum and computing resource utilization states to produce useful features for joint resource optimization. Again, ELU is used as the activation function for all the hidden layers.

Reward function: we assign the master and RMSA agents the same team reward of 1 if R_{ac} is correctly provisioned (otherwise, -1), to make the two agents learn cooperatively toward maximizing the long-term throughput of anycast service provisioning.

Training: we extend the DeepRMSA-FLX algorithm to the dual-agent setting by applying an independent learning scheme, where each agent learns with the DeepRMSA-FLX algorithm while treating the behavior of the other agent as part of the system environment. In other words, the agents learn implicit cooperative policies driven by only team rewarding. Meanwhile, we adopt the following tricks to facilitate the successful training of the agents. First, we sort the candidate paths selected by the RMSA agent in the ascending order of path lengths and arrange $s_{i,O}$ and $s^-_{i,O}$ for the master agent accordingly. By doing this, we intend to stabilize the distribution of state data for the master agent when the behavior of the RMSA agent continuously evolves. Besides, this processing makes the distribution of $s_{i,O}$ for the master agent more similar to that for an RMSA agent (for which, candidate paths are also ordered according to path lengths), and thereby can potentially benefit knowledge transferring. Second, among the $|V_{dc}|$ samples (each for a DC node $v \in V_{dc}$) generated by the RMSA agent upon provisioning R_{ac}, only the sample related to the DC node ultimately selected by the master agent is retained and used for training afterward. This is because the other samples do not have direct correlations with the reward received and can be too noisy for training. For instance, a wrong decision made by the master agent can lead to a negative reward even when some of the actions taken by the RMSA agents are correct.

Knowledge transferring: We apply the proposed transfer DRL design to transfer the knowledge obtained by RMSA agents to both the master and RMSA agents for anycast service provisioning.

VI. Performance Evaluation

We evaluated the performance of the proposed transfer DRL design with dynamic service provisioning simulations using the topologies depicted in Fig. 3. The capacity of each fiber link was set as 100 FS's. We generated dynamic service requests (i.e., lightpath and anycast service requests) following Poisson processes. Each request demands a bandwidth randomly distributed within $[25, 100]$ Gb/s. The computing resource requirement from an anycast service request was set to be equal to the bandwidth requested [49]. We varied the traffic loads from task to task so that the request blocking probabilities could fall into a reasonable range (i.e., around 0.01). For all the tasks, K was set to be 5. We trained RMSA agents for the six-node and the 14-node Japan topologies separately using the approach discussed in Section V-A as the two source tasks, while the remaining three topologies were used for target tasks. For the anycast service provisioning tasks, we set V_{dc} to be $[3, 4, 5, 6, 10]$, $[3, 5, 8, 10, 12]$, and $[3, 6, 10, 16, 17]$ for the COST 239, the NSFNET, and the US Backbone topologies, respectively. Meanwhile, each DC node $v \in V_{dc}$ was assumed to be equipped with $3,600$, $3,600$, and $4,000$ units of computing resources for the three topologies, respectively. In training each target task, we used three actor-

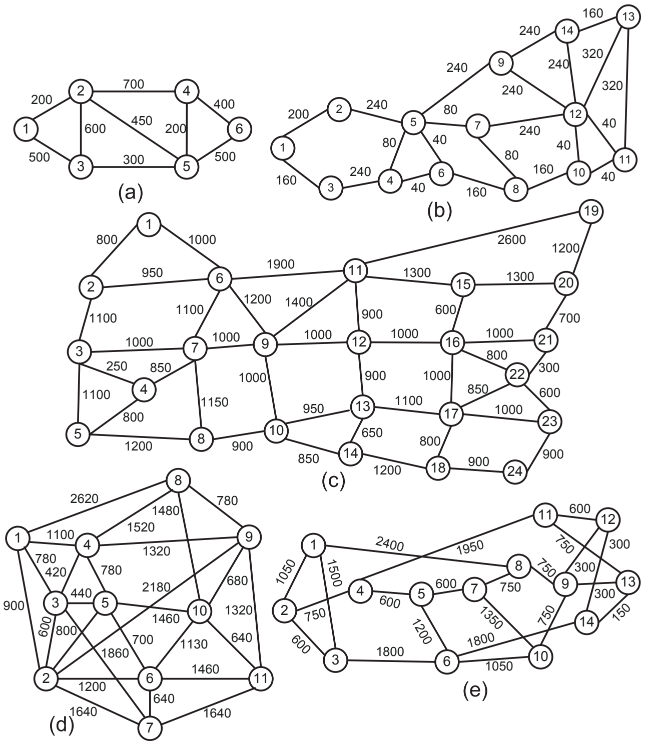

Fig. 3. EON topologies used in the simulations: (a) six-node, (b) 14-node Japan, (c) 24-node US Backbone, (d) 11-node COST 239, and (e) 14-node NSFNET topologies.

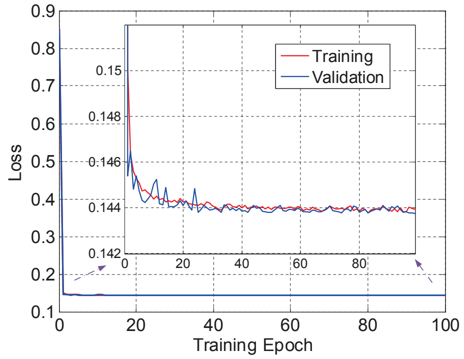

Fig. 4. Policy losses in training the MTL agent.

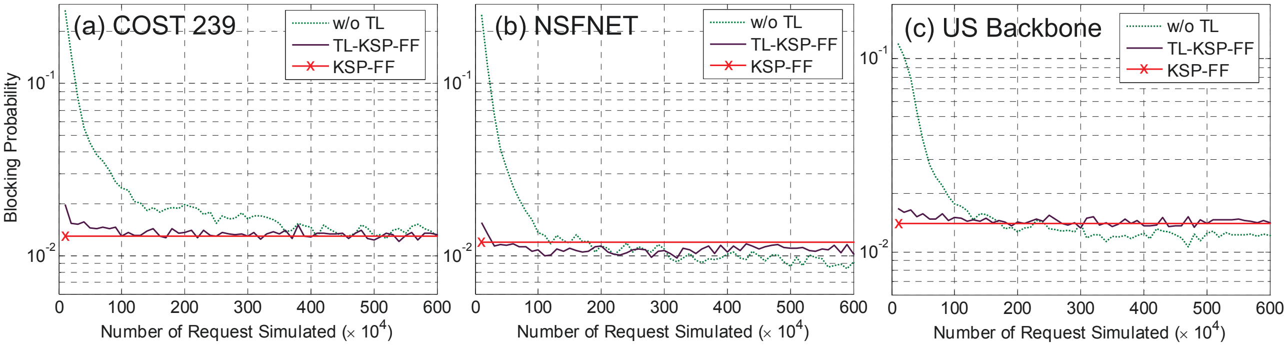

Fig. 5. Performance comparison between TL-KSP-FF and w/o TL.

learners[3] [19]. The number of samples used in each training step was set as 200. The rest of the parameter setups for training were the same as those in [20]. We sampled a state trajectory of $150,000$ instances from each source task with the learned policies for training the MTL agent, where the Adam optimizer was used [50]. Fig. 4 shows the evolution of policy losses in training the MTL agent, which suggests a successful training without the presence of noticeable overfitting (as the validation loss converges with the training loss).

For each target task, we evaluated agents applying different knowledge transferring schemes and an agent learning from scratch without transferring learning (denoted as w/o TL). In particular, we denote the agents that learn with the knowledge transferred from the MTL agent and the RMSA agents for the six-node and the 14-node Japan topologies as TL-MTL, TL-Six-Node, and TL-14-Node, respectively. Since we were interested in understanding whether human knowledge would be beneficial, we also evaluated the scenarios where agents learned on top of heuristic policies for RMSA tasks. Specifically, we selected the K-shortest path routing and first-fit spectrum allocation (KSP-FF) algorithm ($K = 5$), which has been shown to achieve the state-of-the-art performance [51], and trained RMSA agents for the target tasks by generating large sets of data instances (i.e., $100,000$) with KSP-FF and by using a supervised learning approach similar to that for training an MTL agent (see Section IV-C). Note that, KSP-FF adopts a deterministic policy, i.e., we will get from each sample a policy $\pi_i(a) = 1, \exists a \in \Lambda_i$ and $\pi_i(a') = 0, \forall a' \in \Lambda_i \setminus a$. To enable exploration capabilities, we added a random noise with a mean of 0.05 to π_i. The supervised learning approach embedded the heuristic policies in the trained agents, whereafter, we made the agents continue to learn with DRL. We denote such agents as TL-KSP-FF. Meanwhile, we selected KSP-FF and the BL-Single-DC-4 algorithm from [49], which has the best performance, as the heuristic baselines for the RMSA and the anycast service provisioning tasks, respectively.

A. Human Knowledge Transferring to RMSA Agents

We first evaluated the performance of transferring human knowledge. Fig. 5 shows the performance comparison between TL-KSP-FF and w/o TL, where the value of each data point corresponds to the request blocking probability of a provisioning period of $100,000$ lightpath requests. At the initial stages, w/o TL has much higher blocking probabilities because of the randomly initialized policy and value functions. TL-KSP-FF also performs slightly worse than KSP-FF due to the random noises introduced. However, for all the three target tasks, TL-KSP-FF can quickly achieve performance close to that of KSP-FF through training (recall that a training step is invoked every 200 requests being simulated). On the other hand, the performance of TL-KSP-FF can hardly be further improved with training afterward and starts to oscillate around that of KSP-FF. We presume that the agents have been trapped in local optima. In contrast, after certain amounts of training performed, w/o TL can outperform KSP-FF in the NSFNET and the US Backbone topologies. The results suggest that the real effective policy functions are much more sophisticated than what can be fine-tuned from the fixed heuristic policies, which motivates the studies of more comprehensive transfer DRL designs.

B. Knowledge Transferring between RMSA Agents

Next, we evaluated the performance of the proposed transfer DRL design. Figs. 6 (a)-(c) show the evolutions of request blocking probability related to the different knowledge transferring schemes. First, we can see that TL-Six-Node and TL-14-Node can effectively expedite the learning processes from random policies to policies having performance comparable to that of KSP-FF, when compared with w/o TL. Note that, even assisted by knowledge transferring, the initial policies of agents tend to be random because the policy and value heads are still randomly initialized. Quantitatively, TL-Six-Node and TL-14-Node can reduce the number of requests simulated from around four (for the COST 239 topology) or two million (for the NSFNET and the UC Backbone topologies) as required by w/o TL to less than 400 thousand. The initial learning curves from TL-MTL are less sharp than those from TL-Six-Node and TL-14-Node as the agents start with more generalized knowledge. Such knowledge, on the other hand, allows TL-MTL to achieve much better ultimate blocking performance, while the advantages of TL-Six-Node and TL-14-Node over w/o TL are not evident. Figs. 6 (d)-(f) show the plots of complementary cumulative distribution function (CCDF) for blocking reduction against KSP-FF from the different schemes

[3] The A3C algorithm makes use of multiple actor-learners with the same structure, which can be virtually seen as multiple incarnations of a DRL agent. Each actor-learner interacts with its own copy of the environment and updates the global DNNs asynchronously.

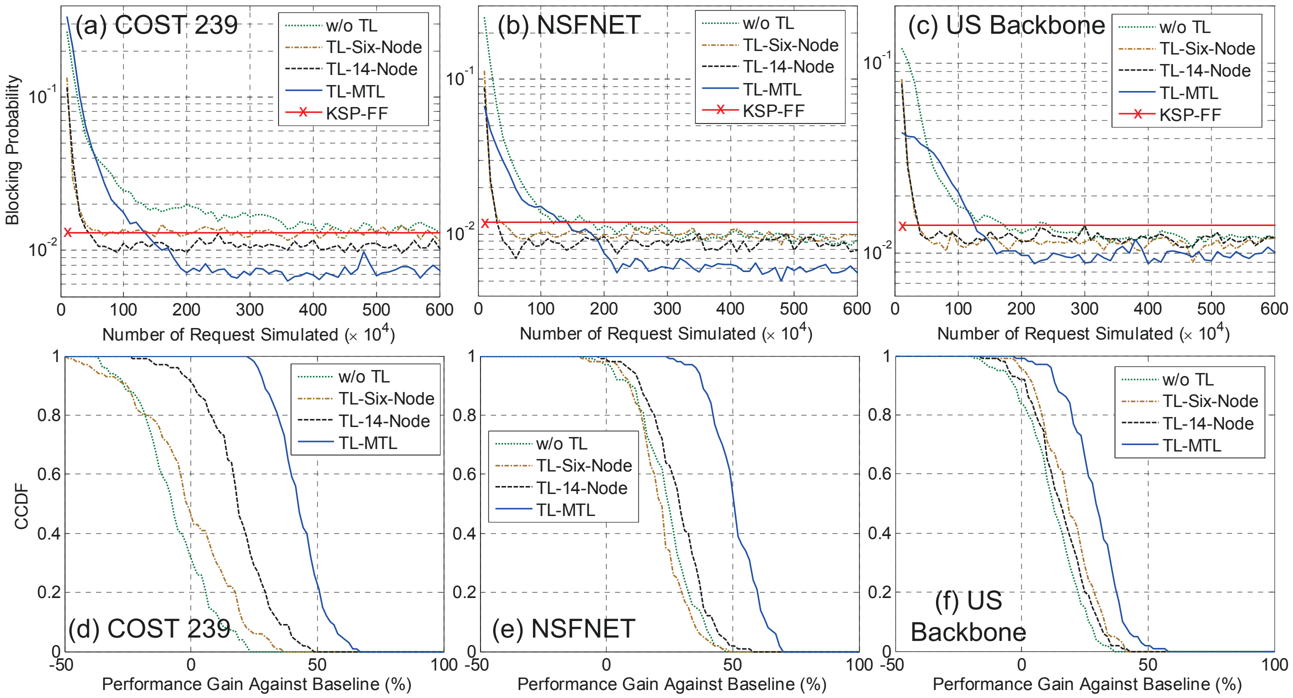

Fig. 6. Results for knowledge transferring between RMSA agents: (a)-(c) evolutions of request blocking probability in training, and (d)-(f) complementary cumulative distribution function (CCDF) of performance gain (blocking reduction) against the KSP-FF algorithm after $5,000,000$ requests simulated.

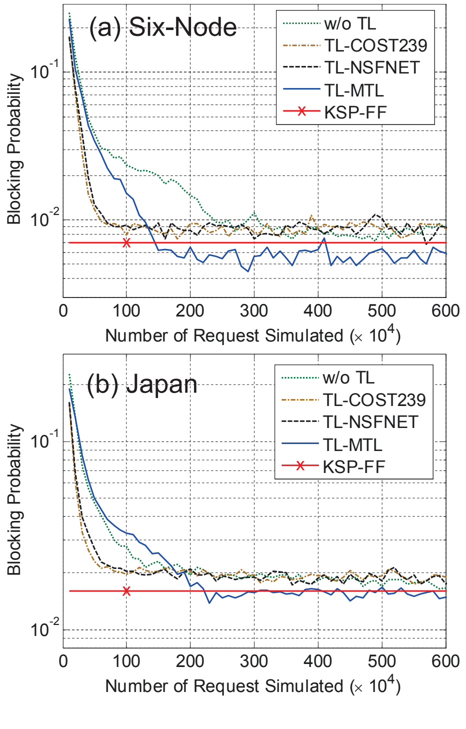

Fig. 7. Evolutions of request blocking probability for (a) the Japan and (b) the six-node topologies with the RMSA agents for the COST 239 and the NSFNET topologies as the source tasks.

after training with five million requests. To obtain the plot for each scheme, we measured the blocking probability of every $10,000$ requests and counted the frequency of achieving performance gain larger than each considered value (the horizontal axis). For the COST 239 topology, TL-MTL assists in achieving blocking reduction of at least 22%, whereas there are probabilities of around 68%, 54%, and 9% to get performance worse than that of the baseline with w/o TL, TL-Six-Node and TL-14-Node, respectively. Similar trends can be observed for the NSFNET and the UC Backbone topologies. On average, TL-MTL can achieve 42.7%, 50.6%, and 28.9% blocking reductions with the three target tasks, respectively, which are $2\times$ of those from TL-Six-Node and TL-14-Node.

We also evaluated the sensitivity of the proposed design to the selection of source and target tasks. Specifically, we transferred the knowledge learned by the RMSA agents for the COST 239 and the NSFNET topologies (which we denote as TL-COST239 and TL-NSFNET, respectively) to the training of RMSA agents for the six-node and the Japan topologies. The results of blocking probability are plotted in Fig. 7, which show trends consistent with those we can see from Fig. 6. TL-COST239 and TL-NSFNET facilitate faster convergence of the learning processes while TL-MTL achieves the best ultimate learning performance. After training with five million requests, TL-MTL achieves blocking reductions of $> 18\%$ and $> 3\%$ against the baseline for the six-node and the Japan topologies, respectively. Whereas the blocking probabilities with w/o TL, TL-COST239, and TL-NSFNET are on average 17%, 22%, and 20% higher than those with the baseline. The results confirm the effectiveness of TL-MTL, and, to a certain extent, verify its robustness to the selection of tasks.

C. Knowledge Transferring between Applications

Lastly, we evaluated the performance of the proposed design with the anycast service provisioning tasks. Figs. 8 (a)-(c) show the evolutions of request blocking probability in training. Without knowledge transferring, the agents cannot converge to policies with performance close to that of BL-

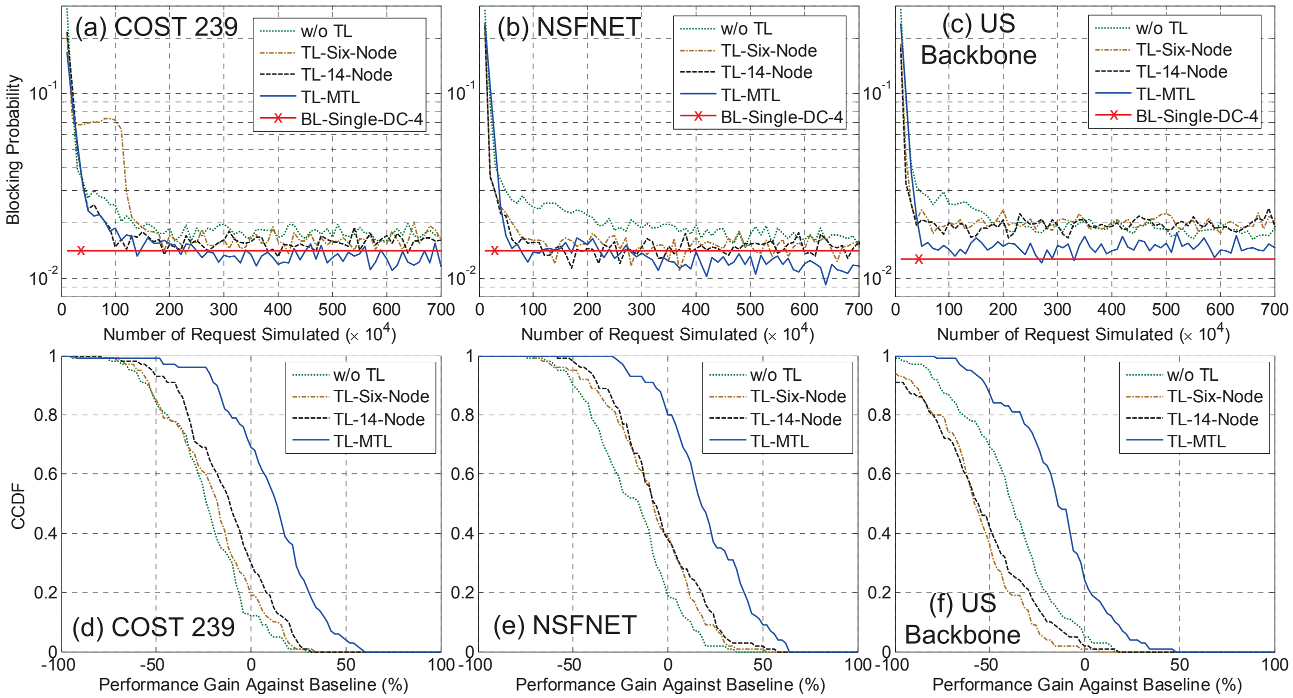

Fig. 8. Results for knowledge transferring from RMSA agents to anycast service provisioning agents: (a)-(c) evolutions of request blocking probability in training, and (d)-(f) CCDF of performance gain (blocking reduction) against the BL-Single-DC-4 algorithm after $6,000,000$ requests simulated.

Single-DC-4. The reason is that with the independent learning scheme, both the master and RMSA agents need to deal with changing and unstable system environments, making learning effective cooperative policies very challenging. Similar to the observation that can be drawn from the evaluations for the RMSA tasks, TL-Six-Node and TL-14-Node facilitate faster policy improvements at the initial training stages compared with w/o TL (with the exception that TL-Six-Node performs the worst under the COST 239 topology), while their ultimate performance gains are hardly noticeable or even negative. TL-MTL is the only scheme that can eventually achieve better or comparable performance compared with BL-Single-DC-4. Figs. 8 (d)-(f) show the CCDF plots of blocking reduction against BL-Single-DC-4 after training with six million requests. For the COST 239 and the NSFNET topologies, TL-MTL outperforms the baseline in 70% and 80% of the cases, with the average performance gains being 10.6% and 19.6%, respectively. In contrast, the average performance with w/o TL, TL-Six-Node and TL-14-Node is worse than that of the baseline, and their frequencies of achieving positive gains are less than 30% and 40%, respectively. Similarly, the results under the US Backbone topology show significant advantage of TL-MTL over the rest three schemes, despite that TL-MTL outperforms the baseline in only 24% of the cases. Note that, the goal of this work is to develop effective knowledge transferring schemes for DRL applications in optical networks while not optimizing the DRL design for a specific application. Developing more advanced agent designs and training algorithms for autonomic anycast service provisioning will be left as one of our future research tasks.

VII. Conclusion

In this paper, we proposed an MTL-aided transfer DRL design for autonomic optical networking. Cases studies under two scenarios, i.e., knowledge transferring between RMSA agents for different topologies and knowledge transferring from RMSA agents to anycast service provisioning agents, verified the effectiveness of the proposed design. Future research directions include: (1) designing more universal state representation and feature extraction schemes, for instance, using graph neural networks to process the original per-link FS state matrices with connectivity information, to extend the applicability of the proposed design to a wider range of applications in optical networks (e.g., service function chain provisioning), (2) studying more comprehensive transfer DRL design exploiting the similarities between tasks (e.g., similarities in state distributions and reward structures), and (3) studying transfer DRL design for multi-domain networks where domain privacy must be secured.

Acknowledgments

This work was supported in part by NSF award # ICE-T:RC 1836921.

References

[1] 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.

[2] 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.

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

[4] 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.

[5] P. Layec, A. Dupas, A. Bisson, and S. Bigo, "Qos-aware protection in flexgrid optical networks," *J. Opt. Commun. Netw.*, vol. 10, no. 1, pp. A43–A50, Jan. 2018.

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

[7] X. Chen, R. Proietti, H. Lu, A. Castro, and S. J. B. Yoo, "Knowledge-based autonomous service provisioning in multi-domain elastic optical networks," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 152–158, Aug. 2018.

[8] S. Shahkarami, F. Musumeci, F. Cugini, and M. Tornatore, "Machine-learning-based soft-failure detection and identification in optical networks," in *Proc. Conf. Opt. Fiber Commun.*, Mar. 2018, pp. 1–3.

[9] R. Morais and J. Pedro, "Machine learning models for estimating quality of transmission in DWDM networks," *J. Opt. Commun. Netw.*, vol. 10, no. 10, pp. D84–D99, Oct. 2018.

[10] X. Chen, B. Li, R. Proietti, C. Liu, Z. Zhu, and S. J. B. Yoo, "Demonstration of distributed collaborative learning with end-to-end QoT estimation in multi-domain elastic optical networks," *Opt. Express*, vol. 27, no. 24, pp. 35 700–35 709, Nov. 2019.

[11] A. Vela, M. Ruiz, F. Fresi, N. Sambo, F. Cugini, G. Meloni, L. Potí, L. Velasco, and P. Castoldi, "BER degradation detection and failure identification in elastic optical networks," *J. Lightw. Technol.*, vol. 35, no. 21, pp. 4595–4604, Nov. 2017.

[12] C. Natalino, M. Schiano, A. Giglio, L. Wosinska, and M. Furdek, "Field demonstration of machine-learning-aided detection and identification of jamming attacks in optical networks," in *Proc. Eur. Conf. Opt. Commun.*, Sept. 2018, pp. 1–3.

[13] X. Chen, B. Li, R. Proietti, Z. Zhu, and S. J. B. Yoo, "Self-taught anomaly detection with hybrid unsupervised/supervised machine learning in optical networks," *J. Lightw. Technol.*, vol. 37, no. 7, pp. 1742–1749, April 2019.

[14] F. Morales, L. Gifre, F. Paolucci, M. Ruiz, F. Cugini, P. Castoldi, and L. Velasco, "Dynamic core VNT adaptability based on predictive metro-flow traffic models," *J. Opt. Commun. Netw.*, vol. 9, no. 12, pp. 1202–1211, Dec. 2017.

[15] M. Salani, C. Rottondi, and M. Tornatore, "Routing and spectrum assignment integrating machine-learning-based QoT estimation in elastic optical networks," in *Proc. IEEE Conf. Comput. Commun.*, April 2019, pp. 1738–1746.

[16] S. Liu, B. Niu, D. Li, M. Wang, S. Tang, J. Kong, B. Li, X. Xie, and Z. Zhu, "DL-assisted cross-layer orchestration in software-defined IP-over-EONs: From algorithm design to system prototype," *J. Lightw. Technol.*, vol. 37, no. 17, pp. 4426–4438, Sept. 2019.

[17] I. Martín, S. Troia, J. A. Hernández, A. Rodríguez, F. Musumeci, G. Maier, R. Alvizu, and Ó. González de Dios, "Machine learning-based routing and wavelength assignment in software-defined optical networks," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 3, pp. 871–883, 2019.

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

[19] 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.

[20] X. Chen, B. Li, R. Proietti, H. Lu, Z. Zhu, and S. J. B. Yoo, "DeepRMSA: A deep reinforcement learning framework for routing, modulation and spectrum assignment in elastic optical networks," *J. Lightw. Technol.*, vol. 37, no. 16, pp. 4155–4163, Aug. 2019.

[21] T. Panayiotou, K. Manousakis, S. Chatzis, and G. Ellinas, "A data-driven bandwidth allocation framework with QoS considerations for EONs," *J. Lightw. Techn.*, vol. 37, no. 9, pp. 1853–1864, May 2019.

[22] X. Luo, C. Shi, L. Wang, X. Chen, Y. Li, and T. Yang, "Leveraging double-agent-based deep reinforcement learning to global optimization of elastic optical networks with enhanced survivability," *Opt. Express*, vol. 27, no. 6, pp. 7896–7911, Mar. 2019.

[23] M. Raza, C. Natalino, P. Öhlen, L. Wosinska, and P. Monti, "Reinforcement learning for slicing in a 5G flexible RAN," *J. Lightw. Technol.*, vol. 37, no. 20, pp. 5161–5169, Oct. 2019.

[24] J. Suarez-Varela, A. Mestres, J. Yu, L. Kuang, H. Feng, A. Cabellos-Aparicio, and P. Barlet-Ros, "Routing in optical transport networks with deep reinforcement learning," *J. Opt. Commun. Netw.*, vol. 11, no. 11, pp. 547–558, Nov. 2019.

[25] P. Almasan, J. Suárez-Varela, A. Badia-Sampera, K. Rusek, P. Barlet-Ros, and A. Cabellos-Aparicio, "Deep reinforcement learning meets graph neural networks: exploring a routing optimization use case," 2019, arXiv:1910.07421.

[26] X. Chen, R. Proietti, and S. J. B. Yoo, "Building autonomic elastic optical networks with deep reinforcement learning," *IEEE Commun. Mag.*, vol. 57, no. 10, pp. 20–26, Oct. 2019.

[27] B. Li, W. Lu, and Z. Zhu, "Deep-NFVOrch: leveraging deep reinforcement learning to achieve adaptive vNF service chaining in DCI-EONs," *J. Opt. Commun. Netw.*, vol. 12, no. 1, pp. A18–A27, Jan. 2020.

[28] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[29] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine, "Learning invariant feature spaces to transfer skills with reinforcement learning," 2017, arXiv:1703.02949.

[30] A. Barreto, W. Dabney, R. Munos, J. Hunt, T. Schaul, H. van Hasselt, and D. Silver, "Successor features for transfer in reinforcement learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4055–4065.

[31] E. Parisotto, J. Ba, and R. Salakhutdinov, "Actor-mimic: Deep multitask and transfer reinforcement learning," 2015, arXiv:1511.06342.

[32] X. Chen, , R. Proietti, C. Liu, Z. Zhu, and S. J. B. Yoo, "Exploiting multi-task learning to achieve effective transfer deep reinforcement learning in elastic optical networks," in *Proc. Conf. Opt. Fiber Commun.*, 2020, paper M1B.3.

[33] S. Troia, S. Rodriguez, I. Martín, I. Hernández, O. G. De Dios, R. Alvizu, F. Musumeci, and G. Maier, "Machine-learning-assisted routing in SDN-based optical networks," in *Proc. Eur. Conf. Opt. Commun.*, 2018, pp. 1–3.

[34] J. Yu, W. Mo, Y. Huang, E. Ip, and D. C. Kilper, "Model transfer of QoT prediction in optical networks based on artificial neural networks," *J. Opt. Commun. Netw.*, vol. 11, no. 10, pp. C48–C57, 2019.

[35] L. Xia, J. Zhang, S. Hu, M. Zhu, Y. Song, and K. Qiu, "Transfer learning assisted deep neural network for OSNR estimation," *Opt. Express*, vol. 27, no. 14, pp. 19 398–19 406, July 2019.

[36] Q. Yao, H. Yang, A. Yu, and J. Zhang, "Transductive transfer learning-based spectrum optimization for resource reservation in seven-core elastic optical networks," *J. Lightw. Technol.*, vol. 37, no. 16, pp. 4164–4172, 2019.

[37] C. Liu, X. Chen, R. Proietti, and S. J. B. Yoo, "Evol-TL: Evolutionary transfer learning for QoT estimation in multi-domain networks," in *Proc. Conf. Opt. Fiber Commun.*, 2020, paper Th3D.1.

[38] D. Rafique and L. Velasco, "Machine learning for network automation: Overview, architecture, and applications [invited tutorial]," *J. Opt. Commun. Netw.*, vol. 10, no. 10, pp. D126–D143, Oct. 2018.

[39] F. Khan, Q. Fan, C. Lu, and A. Lau, "An optical communication's perspective on machine learning and its applications," *J. Lightw. Technol.*, vol. 37, no. 2, pp. 493–516, Jan. 2019.

[40] F. Musumeci, C. Rottondi, A. Nag, T. Macaluso, D. Zibar, M. Ruffini, and M. Tornatore, "An overview on application of machine learning techniques in optical networks," *IEEE Commun. Surv. Tutor.*, vol. 21, no. 2, pp. 1383–1408, Secondquarter 2019.

[41] F. Paolucci, A. Sgambelluri, F. Cugini, and P. Castoldi, "Network telemetry streaming services in SDN-based disaggregated optical networks," *J. Lightw. Technol.*, vol. 36, no. 15, pp. 3142–3149, 2018.

[42] S. Salman, C. Streiffer, H. Chen, T. Benson, and A. Kadav, "DeepConf: Automating data center network topologies management with machine learning," in *Proc. Workshop Netw.Meets Artif. Intell. Mach. Learn.*, 2018, pp. 8–14.

[43] A. Brock, T. Lim, J. Ritchie, and N. Weston, "FreezeOut: Accelerate training by progressively freezing layers," 2017, arXiv:1706.04983.

[44] Y. Teh, V. Bapst, W. Czarnecki, J. Quan, J. Kirkpatrick, R. Hadsell, N. Heess, and R. Pascanu, "Distral: Robust multitask reinforcement learning," 2017, arXiv:1707.04175.

[45] M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. Hasselt, "Multi-task deep reinforcement learning with PopArt," 2018, arXiv:1809.04474.

[46] A. Lazaric, M. Restelli, and A. Bonarini, "Transfer of samples in batch reinforcement learning," in *Proc. of Int. Conf. Mach. Learn.*, 2008, p. 544–551.

[47] 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.

[48] 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. Conf. Opt. Fiber Commun.*, 2010, paper OMU3.

[49] L. Zhang and Z. Zhu, “Spectrum-efficient anycast in elastic optical inter-datacenter networks,” *Opt. Switch. Netw.*, vol. 14, pp. 250 – 259, 2014.

[50] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014.

[51] 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.

Xiaoliang Chen received his Ph.D. degree from the University of Science and Technology of China in 2016. He is an Associate Professor at the Sun Yat-Sen University. Prior to joining the Sun Yat-sen University, he was a postdoctoral researcher at the University of California, Davis (UC Davis). His research interests include optical networks, optical interconnection architectures for data centers, software-defined networking, and cognitive networking. He is an Associate Editor of Springer’s Telecommunication Systems Journal and Photonic Network Communications Journal, and a TPC member of IEEE ICNC 2018-2020, and ICC 2018-2021.

Roberto Proietti received the M.S. degree in telecommunications engineering from the University of Pisa, Pisa, Italy, in 2004, and the Ph.D. degree in electrical engineering from Scuola Superiore Sant Anna, Pisa, in 2009. He is a Project Scientist with the Next Generation Networking Systems Laboratory at UC Davis. His research interests include optical switching technologies and architectures for supercomputing and data center applications, high-spectral-efficiency coherent transmission systems, and elastic optical networking.

Che-Yu Liu was born in Tainan, Taiwan, in 1989. He received the B.S. degree in electrical and computer engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2012, and the M.S. degree in electrical and computer engineering, UC Davis, in 2016. He is currently working toward the Ph.D. degree in computer science, UC Davis. His research interest focuses on optical networks, software-defined networking, and machine learning.

S. J. Ben Yoo is a Distinguished Professor at UC Davis. His research at UC Davis includes 2D/3D photonic integration for future computing, cognitive networks, communication, imaging, and navigation systems, micro/nano systems integration, and the future Internet. Prior to joining UC Davis in 1999, he was a Senior Research Scientist at Bellcore, leading technical efforts in integrated photonics, optical networking, and systems integration. His research activities at Bellcore included the next-generation Internet, reconfigurable multiwavelength optical networks (MONET), wavelength interchanging cross connects, wavelength converters, vertical-cavity lasers, and high-speed modulators. He led the MONET testbed experimentation efforts, and participated in ATD/MONET systems integration and a number of standardization activities. Prior to joining Bellcore in 1991, he conducted research on nonlinear optical processes in quantum wells, a four-wave-mixing study of relaxation mechanisms in dye molecules, and ultrafast diffusion-driven photodetectors at Stanford University (BS 84’, MS 86’, PhD 91’, Stanford University). Prof. Yoo is Fellow of IEEE, OSA, NIAC and a recipient of the DARPA Award for Sustained Excellence (1997), the Bellcore CEO Award (1998), the Mid-Career Research Faculty Award (2004 UC Davis), and the Senior Research Faculty Award (2011 UC Davis).