Exploiting Multi-Task Learning to Achieve Effective Transfer Deep Reinforcement Learning in Elastic Optical Networks

Xiaoliang Chen[1], Roberto Proietti[1], Che-Yu Liu[1], Zuqing Zhu[2], S. J. Ben Yoo[1]

1. University of California, Davis, Davis, CA 95616, USA
2. University of Science and Technology of China, Hefei, Anhui 230027, China
xlichen@ucdavis.edu, sbyoo@ucdavis.edu

Abstract: We propose a multi-task-learning-aided knowledge transferring approach for effective and scalable deep reinforcement learning in EONs. Case studies with RMSA show that this approach can achieve $\sim 4\times$ learning time reduction and $\sim 17.7\%$ lower blocking probability. © 2020 The Author(s)

OCIS codes: (060.1155) All-optical networks; (060.4251) Networks, assignment and routing algorithms.

1. Introduction

Cognitive and knowledge-based service provisioning schemes are crucial for exploiting the benefits of elastic optical networks (EONs) while meeting the challenges of the future Internet [1]. The past few years have witnessed an explosive growth of studies on the applications of machine learning (ML) in optical networks, aiming at quality-of-transmission modeling [2], fault management [3], and resource allocation [4]. Among these studies, the application of deep reinforcement learning (DRL) [5] for pursuing self-learning autonomic EONs is attracting increasing research attention [4,6]. Specifically, DRL allows learning successful service provisioning policies directly from online network operations while eliminating human interventions, and thus, has been regarded as a very promising enabling technique for the next-generation intelligent EONs. In [4], we demonstrated a DRL framework for routing, modulation and spectrum assignment (RMSA) in EONs which can beat the state-of-the-art heuristic designs. The authors of [6] proposed a DRL approach for efficient protection design in EONs. On the other hand, the training of DRL tasks is never trivial and can be extremely time-consuming and costly, leading to non-negligible scalability and applicability issues. In this context, transfer learning (TL) approaches have been investigated for DRL [7]. By exploiting the similarities between tasks and reusing existing knowledge, TL can effectively facilitate the training of DRL. Nevertheless, the aforementioned issues remain untouched for EONs.

In this paper, we propose a knowledge transferring design to assist effective and scalable DRL in EONs. We first present a modularized DRL agent design by taking into account shared and task-specific state spaces. With the new agent design, we develop a multi-task learning (MTL) mechanism to enable learning and transferring better generalized and unbiased knowledge. We perform a case study of the proposed transfer DRL design with RMSA tasks for different topologies. Performance evaluations show remarkable benefits of the proposed approach.

2. Transfer DRL Design

Let $\mathbb{M}_{src}$ and $\mathbb{M}_{tgt}$ denote the sets of source and target DRL-based service provisioning tasks in EONs, respectively. Each of the tasks can be modeled as a Markov decision process $M_i(S_i, \Lambda_i, R_i, P_i)$, where S_i and Λ_i represent its unique state and action spaces, respectively, R_i is the reward function, and P_i is the state transition function. We parameterize the policy for M_i (denoted as π_i) by a deep neural network (DNN), namely, $\pi_i = f_{\theta_i}(s,a)$ $(s \in S_i, a \in \Lambda_i)$, where θ_i is the set of trainable parameters. π_i indicates a probability distribution over the action space or generates an action directly given s. Let us assume that (1) the optimal policy function $f_{\theta_i^*}(s,a)$, which produces the maximum long-term cumulative reward, has been learned for each $M_i \in \mathbb{M}_{src}$, and (2) the source and target tasks share certain similarities, for instance, they all involve spectrum allocation procedures. Our objective is to expedite the training phases and to improve the performance of the learned policies for $\mathbb{M}_{tgt}$ by transferring the knowledge acquired for $\mathbb{M}_{src}$, i.e., by taking advantage of θ_i^*.

Knowledge transferring is typically achieved by reusing θ_i^* for the initialization of the policy function of a target task. Since different tasks in EONs can require different amounts of network state information, leading to different dimensions of state representations, a straightforward copying of θ_i^* is infeasible. For instance, a DRL agent for RMSA needs to take as input the spectrum utilization information, whereas, an agent for service function chaining requires additionally the states of the virtual network functions (vNFs) deployed. To facilitate knowledge transfer

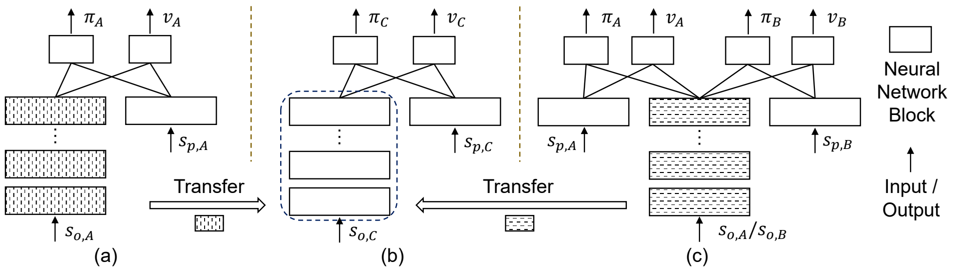


Fig. 1. Principle of the proposed knowledge transferring approach: architectures of DRL agents for (a) source task M_A, (b) target task M_C, and (c) multiple source tasks (tasks M_A and M_B).

between such tasks, we first present a modularized learning agent design. Fig. 1(a) shows the DNN architecture of an agent (w.l.o.g., for task M_A). We partition each $s_A \in S_A$ into a task-specific state $s_{p,A}$ and a shared state $s_{o,A}$. In particular, $s_{o,A}$ conveys network state information shared by different tasks (e.g., spectrum utilization), while $s_{p,A}$ can contain the specific request model and network state information demanded (e.g., vNF utilization). The agent processes $s_{o,A}$ and $s_{p,A}$ with separate neural network blocks (denoted as $f_{\theta_{o,A}}(s_{o,A})$ and $f_{\theta_{p,A}}(s_{p,A})$, respectively) to extract high-level and useful features for the task. $f_{\theta_{o,A}}(s_{o,A})$ adopts an architecture identical across different tasks. Finally, two neural network blocks are deployed on the top to generate π_A and the value estimation v_A (i.e., the estimated long-term reward given $s_{o,A}$ and $s_{p,A}$) combining the extracted features.

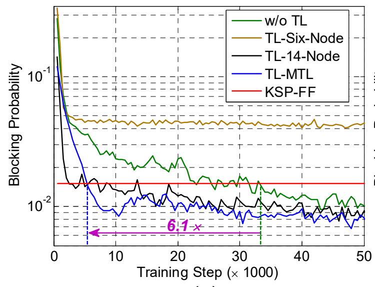
With the aforementioned modularized agent design, a most intuitive approach of transferring knowledge from a source task M_A to a target task M_C is copying $f_{\theta_{o,A}}(s_{o,A})$ to $f_{\theta_{o,C}}(s_{o,C})$ while having $f_{\theta_{p,C}}(s_{p,C})$ and the rest of the neural network blocks randomly initialized (depicted by Figs. 1(a) and (b)). The derived DNN then serves as a starting point for task M_C, when a classic training process is evoked. Although $s_{o,A}$ and $s_{o,C}$ can have different distributions, the best feature extractor for task M_C (i.e., $f_{\theta_{o,C}}(s_{o,C})$) would resemble that for task M_A, and thus, the knowledge transfer helps us more quickly approach the learning target. Note that, the feature extractor learned for a specific source task is likely to be overfitted or biased, which can result in negative transferring effects, especially when the source and target tasks have significantly different characteristics. To overcome this issue, we propose to leverage MTL for learning better-generalized feature extractors. Fig. 1(c) illustrates the MTL-based agent design in the case of two source tasks (tasks M_A and M_B). The DNN architecture for each of the source tasks is much the same as that shown in Fig. 1(a), with the only difference being that a common neural network block (denoted as $f_{\theta_{o,AB}}$) is shared for processing the shared states. The multi-task agent is trained to be expert in both source tasks, after which, $f_{\theta_{o,AB}}$ is transferred to the target task (depicted by Figs. 1(b) and (c)). This way, we enable to learn feature extractors that better generalize across tasks while exploiting knowledge from multiple tasks.

3. Case Study with RMSA

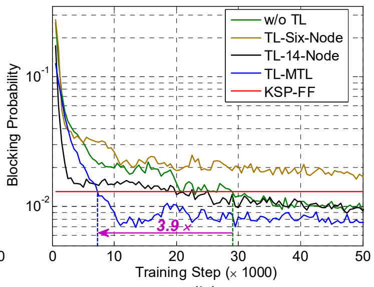
Learning effective RMSA policies is essential for EONs. Meanwhile, in the context of disaster recovery where the traffic profile and topology of an EON can change dramatically, or, in the scenario of network slicing where virtual EONs with different topologies are created dynamically [8], new RMSA policies need to be learned. Therefore, in this work, we study knowledge transferring between RMSA tasks for different topologies. We make use of the DeepRMSA framework presented in [4] and slightly modify it according to the agent design shown in Fig. 1(a). Specifically, for each task M_i, $s_{p,i}$ contains the information of a lightpath request (i.e., source and destination nodes, service duration) and $s_{o,i}$ expresses the spectrum utilization state on K pre-calculated routing paths for the request. All the neural network blocks adopt fully-connected architectures. The DeepRMSA agent selects one from the K candidate paths at each system state and applies the first-fit spectrum allocation scheme. The agent is granted a reward of 1 if the request is successfully provisioned (otherwise, -1). We apply the DeepRMSA-FLX algorithm [4], which is based on the A3C framework in [5], for the training of individual tasks. We train a multi-task agent (see Fig. 1(c)) with supervised learning by mimicking the policies learned through DRL. In particular, we generate large amounts of training data by recording the state, policy and value tuples (i.e., $\{s_i^t, \pi_i^t, v_i^t\}|_t$) over a long trajectory of operation experiences from each agent of source tasks. Then, we tune all the neural network blocks jointly, so that, for every $M_i \in \mathbb{M}_{src}$, the following policy and value loss functions are minimized.

$$L_i^{\pi}(\theta^{\pi}) = -\frac{1}{N}\sum_t \sum_{a \in \Lambda_i} \pi_i^t(a) \log f_{\theta^{\pi}}(s_i^t, a), \qquad L_i^{v}(\theta^{v}) = \frac{1}{N}\sum_t \left(f_{\theta^{v}}(s_i^t) - v_i^t\right)^2. \tag{1}$$

Here, θ^{π} and θ^{v} are the sets of DNN parameters contributing to the policy and value outputs, respectively, and N is the number of training data instances. Note that, we can also train a multi-task agent with DRL directly by making it interact with the environments of multiple source tasks simultaneously. However, as the reward structures and the intensities of reinforcement signals from different tasks can be quite different, the interferences among tasks



(a)



(b)

	Target Task					
	Ten-Node		11-Node		24-Node	
	TS-Th	γ	TS-Th	γ	TS-Th	γ
w/o TL	33,500	28.2%	29,500	23.6%	14,000	37.9%
TL-Six-Node	N.A.	-1.8%	N.A.	-35%	9,000	52.2%
TL-14-Node	14,000	40.6%	25,500	27.1%	**3,500**	44.2%
TL-MTL	**5,500**	**48.3%**	**7,500**	**40.1%**	7,000	**54.3%**

(c)

Fig. 2. Evolutions of request blocking probability in training with (a) the ten-node and (b) the 11-node topologies, and (c) results of training steps to threshold (TS-Th) and blocking reduction compared with KSP-FF (γ). w/o TL: learning without TL; TL-M_i: learning with knowledge transferred from task M_i.

can severely destabilize the training or leading to certain tasks being dominated. Finally, we apply the transfer DRL design discussed in Section 2 for knowledge transferring between RMSA tasks.

4. Performance Evaluation

We evaluated the performance of the proposed transfer DRL design with numerical simulations incorporating two source tasks (RMSA for the six-node and the 14-node NSFNET topologies) and three target tasks (RMSA for the 10-node, the 11-node COST 239, and the 24-node US Backbone topologies). We set the link capacity to be 100 frequency slots and assumed dynamic and uniformly distributed lightpath requests with arrivals following Poisson processes and bandwidth requirements evenly distributed within $[25,100]$ Gb/s. Each DRL agent (for task M_i) was implemented with $f_{\theta_{o,i}}$ of five hidden layers (128 neurons each layer), a policy layer of five neurons, and a value layer of one neuron. Both the policy and value layers take as input $s_{p,i}$ and the outputs of $f_{\theta_{o,i}}$. We employed eight and one actor-learner for the source and target tasks, respectively. The rest of the parameter setups for DRL were similar to those presented in [4]. For the training of the multi-task agent, we generated $100,000$ data instances from every source task. For each target task, we evaluated four cases, i.e., learning (1) without knowledge transferring (w/o TL), with knowledge transferred from (2) the six-node topology (TL-Six-Node), (3) the 14-node topology (TL-14-Node), and (4) the MTL agent (TL-MTL). We also implemented the K-shortest path routing ($K = 5$) and first-fit spectrum allocation algorithm (KSP-FF) with the state-of-the-art performance as the baseline.

Fig. 2(a) shows the evolutions of request blocking probability during training with the ten-node topology. We can see that TL-14-Node and TL-MTL can remarkably expedite the learning process of the target task, whereas, negative transferring is observed for TL-Six-Node, leading to performance even worse than that of learning from scratch. We presume that the negative effect is due to that the difference between the two tasks is significant and that the knowledge transferred is severely biased. However, it is interesting to notice that such biased knowledge is still very helpful when integrated into an MTL scenario. It can be seen that TL-MTL always outperforms TL-14-Node after the initial stage. This is because TL-MTL enables to exploit both source tasks and learn better-generalized knowledge. Let us set the blocking probability of KSP-FF as a threshold of acceptable performance, TL-MTL can reduce the number of training steps required to completely surpass the threshold (TS-Th) by $6.1\times$ when compared with learning from scratch. Fig. 2(b) presents the performance evolutions from different learning models with the 11-node topology. The results confirm the above observations while showing a more clear advantage of TL-MTL against TL-14-Node. Finally, we summarize the results for different target tasks in Fig. 2(c). We can see that TL-MTL can effectively reduce TS-Th for all the cases (by a factor of $4\times$ on average), whereas, TL-Six-Node only works with the 24-node topology. We also measured the eventual performance of each model by counting its blocking reduction against KSP-FF after training of $50,000$ steps (γ). Again, the results show that TL-MTL performs the best amongst the models. Compared with learning from scratch, TL-MTL achieves 20.1%, 16.5%, and 16.4% extra blocking reductions, respectively, for the three target tasks.

5. Conclusion

We proposed an MTL-aided knowledge transferring approach for DRL in EONs and verified its effectiveness with evaluations on RMSA. Our future work will investigate knowledge transferring between different applications.

References

1. X. Chen *et al.*, *IEEE Commun. Mag.*, **56**, 152–158 (2018).
2. R. Proietti *et al.*, *J. Opt. Commun. Netw.*, **11**, A1-A10 (2019).
3. D. Rafique *et al.*, *J. Lightw. Technol.*, **36**, 1443-1450 (2018).
4. X. Chen *et al.*, *J. Lightw. Technol.*, **37**, 4155-4163 (2019).
5. V. Mnih *et al.*, in Proc. of *ICML*, **48**, 1928–1937 (2016).
6. X. Luo *et al.*, *Opt. Express*, **27**, 7896-7911 (2019).
7. E. Parisotto *et al.*, arXiv:1511.06342 (2015).
8. L. Gong *et al.*, *J. Lightw. Technol.*, **32**, 450–460 (2014).

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