Submitted to *INFORMS Journal on Optimization*
manuscript MS-0001-1922.65

Authors are encouraged to submit new papers to INFORMS journals by means of a style file template, which includes the journal title. However, use of a template does not certify that the paper has been accepted for publication in the named journal. INFORMS journal templates are for the exclusive purpose of submitting to an INFORMS journal and should not be used to distribute the papers in print or online or to submit the papers to another publication.

Dynamic Routing and Wavelength Assignment with Reinforcement Learning

Peyman Kafaei
Department of Mathematics and Industrial Engineering Polytechnique Montreal, Montreal, Canada,
peyman.kafaei@polymtl.ca

Quentin Cappart
Computer Engineering and Software Engineering Department Polytechnique Montreal Montreal, Canada,
quentin.cappart@polymtl.ca

Nicolas Chapados
Department of Mathematics and Industrial Engineering Polytechnique Montreal, Montreal, Canada,
nicolas.chapados@servicenow.com

Hamed Pouya
Department of Mechanical and Industrial Engineering University of Toronto, Toronto, Canada,
h.pouya@utoronto.ca

Louis-Martin Rousseau
Department of Mathematics and Industrial Engineering Polytechnique Montreal, Montreal, Canada,
louis-martin.rousseau@polymtl.net

With the rapid developments in the communication systems, and considering the dynamic nature of them, all-optical-networks are becoming increasingly complex. This study proposes a novel method based on Deep Reinforcement Learning for the routing and wavelength assignment problem in all-optical wavelength-decision-multiplexing networks. We consider dynamic incoming requests, in which their arrival and holding time are not known in advance. The objective is to devise a strategy that minimizes the number of rejected packages due to the lack of resources in the long term. We employ Graph Neural Networks to capture crucial latent information from the graph-structured input to develop the optimal strategy. The proposed Deep Q-Networks algorithm selects a route and a wavelength simultaneously for each incoming traffic connection as they arrive. The results demonstrate that the learned agent outperforms the methods widely used in practice and can be generalized on network topologies that did not participate in training.

Key words: Routing and Wavelength Assignment; Deep Q-learning; Graph Neural Networks; Reinforcement Learning

1. Introduction

All-optical-Networks are considered to play a critical role in wide-area backbone networks (Pinto 2002). In this paper, we investigate the performance of novel learning algorithms on the Routing and Wavelength Assignment (RWA) problem (Ramaswami and Sivarajan 1995) in the context of wavelength-division multiplexing (WDM) networks. Solving RWA corresponds to selecting a route between two end-points of all incoming requests and assigning a wavelength to transmit the data across them. In WDM networks, packages that use the same fiber links cannot use the same wavelengths at the same time. Due to the limited number of wavelengths, incoming traffic requests may be blocked if there are no available wavelengths across the links of the selected route.

RWA problems are conventionally categorized into static and dynamic. The static version is based on the assumption that all the information of the incoming requests such as arrival time, and duration (in case of releasing the requests after a finite time) are known. A set of lightpaths (combination of a route and a wavelength) are set up all at once, and they remain in the network. However, in realistic applications, these assumptions do not hold. A lightpath needs to be generated for each incoming request as it arrives. The admitted traffic connection request may be released after a finite amount of time.

he dynamic arrival and departure of the traffic requests and the uncertainty of the future ones substantially destabilize the problem and reduce the accuracy and efficiency of the conventional optimization methods. In general, the lightpaths already assigned cannot be re-routed to accommodate the new requests that arrive. However, aggressive reconfiguration i.e. lightpath defragmentation has recently been proposed to modify the current lightpaths to reduce the blockage probabilities at the expense of high computational and operational costs (Chen et al. 2019, Pouya 2018, Daryalal and Bodur 2020).

In the settings of this work, wavelengths are assigned to each of the links comprising a route for an incoming request. If all of these links use the same wavelength along the route of the request, the information can transfer from source to destination. Otherwise, the incoming request is blocked. This is known as the wavelength continuity constraint, which makes the wavelength-routed networks are different from the traditional circuit-switched telephone networks.

Although recently researchers have started applying methods from artificial intelligence in the field of RWA problems, only routing has been under investigation. The wavelength assignment and routing are substantially intertwined and selecting one affects the other. In this paper, an algorithm based on Deep Reinforcement Learning is presented for the dynamic RWA problem. This algorithm selects the route and wavelengths simultaneously. The algorithm is entirely online and trains via simulation and past experiences, and thereby does not require large data sets. We assume that precise network state information is available at any point in time.

The specific contributions are as follows:

1. We propose a Deep Reinforcement Learning algorithm to find the routes and assign wavelengths to incoming traffic requests. Experimental results obtained show that this approach can obtain a high-quality solution against standard baselines used to solve this problem for 4 of the most famous topologies.

2. We exploit the topology of underlying network as a graph-structured data and apply Graph Neural Networks to encode latent information. This enables us to capture more features and leads to a better performance of the proposed algorithm.

3. The DRL agent proposed in this work has the capability to select routes and assign wavelengths for the dynamic RWA problem simultaneously. This outperforms the conventional methods that mostly focus on the routing part of the problem, for incoming connection requests.

The rest of the paper is organized as follows. Section 2 examines the existing literature on the use of Graph Neural Networks and Deep Reinforcement Learning in Routing and Wavelength Assignment as well as the previous algorithms to solve the RWA problem. Section 3 discusses the deep reinforcement learning algorithm to solve this problem. The RL environment and the proposed neural networks architecture are introduced in this Section. Experimental results, discussions, and conclusions are presented in Section 4.

2. Background

In this Section we study previous methods developed to solve the RWA problem.

2.1. Routing and Wavelength Assignment

Route planning and resource management are two fundamental tasks in optical networks. This problem is shown to be NP-hard for a set of traffic requests (Di Ianni 1998). As the size and complexity of the network increase, opting for conventional Shortest Path First routing algorithms may lead to a high number of blocked requests. In addition to routing, assigning wavelengths to routes results in a more complex problem. RWA is an NP-complete problem (Gu et al. 2020). It may be formulated as an Integer Linear Programming (ILP). This approach however attains solutions at the expense of complex, expensive, and time-consuming computations even for medium-sized instances (Martin et al. 2019). The majority of the literature on RWA has focused on heuristic solutions, such as Genetic Algorithm and Simulated Annealing (Zhou et al. 2012). These methods are faster but attain sub-optimal solutions (Daryalal and Bodur 2020, Martin et al. 2019). Most of the conventional, exact algorithms are proposed only to find a route, while the heuristics are capable of finding several routes more readily (Rodriguez et al. 2016). A taxonomy of the RWA problems and their corresponding information is provided in (Zang et al. 2000). Jaumard and Daryalal (2017) propose methods to enable solving RWA instances with realistic sizes. To this end, they represent an exact method based on column generation decomposition as well as two

heuristics for the RWA problem. The authors outlined that the algorithm may fail for large instances in a reasonable solution time; however, it provides ϵ-solutions with high accuracy. Daryalal and Bodur (2020) propose formulated the RWA problem as a stochastic optimization problem where uncertainty exist for the future traffic. They studied two perspectives, namely, while granting the connection requests via the RWA problem, and during the defragmentation process by lightpath rerouting. In their study, the distribution of the random variables are known to the decision maker.

In contrast to routing solution approaches, a number of researchers study wavelength assignment techniques. (Randhawa and Sohal 2010) show that that by increasing the load of the network, the blocking probability increases. Zhou and Yuan (2002) compared different wavelength assignment strategies, random-fit, first-fit, and most-used with the effect of imprecise global network information. In their study, they opted for the widest shortest path algorithm for the routing part of the incoming requests.

While several researchers have studied dynamic RWA problems, the majority of them focus on finding optimal routing while the requests arrive in real-time, and only opted for a fixed heuristic for the wavelength assignment (Zang et al. 2001, Huimin et al. 2017, Duhovnikov et al. 2006). Some also consider traffic grooming by the presence of wavelength conversions throughout the network, to reduce the blocking probability (Mohan and Kaushal 2020). A dynamic algorithm that considers the length of the k-shortest-paths and free wavelengths is developed by combining the weighted least-congestion routing and first-fit wavelength assignment strategy (Chu et al. 2003). It is shown that plugging wavelength converters over the network does not highly contribute to the reduction of the blockage rate as time goes on (Jaumard et al. 2005, 2006).

Recently, different Machine Learning techniques have been applied to solve this complex problem. In Martin et al. (2019), the authors presented supervised learning solution methods based on logistic regression and deep neural networks. They formulated the RWA problem as a classification problem. Several optimal RWA configurations are computed prior to training the model using ILPs and heuristics in the literature. Their method learns relations between features of a problem, such as the topology of the network, capacity, and wavelengths with one of the corresponding optimal configurations. Although this approach attains solutions quickly, it only considers a few possible configurations for all diverse network configuration inputs. Several researchers leverage Reinforcement Learning to solve the RWA problem. Kiran et al. (2007) formulates the path and wavelength selection in Optical Burst Switched (OBS) networks as a multi-armed bandit problem and solves it with a tabular Q-learning method. They provide separate algorithms for path and wavelength selection. The corresponding Q-value of path selection and wavelength selection are updated based on a successful burst in the network. The selection of the route and wavelength are not directly related in their proposed method. Applying DRL to network optimization problems

has received high interest in recent years. Chen et al. (2019) propose a DRL framework for a Routing, Modulation, and Spectrum Assignment (RMSA) in Elastical Optical Networks. They deploy advantage actor-critic methods to parameterize the action selection policy, and showed a reduction in request blockage following their proposed algorithm. In (Xu et al. 2018), the authors utilize experience replay to enhance the performance of a proposed Q-learning method to solve the traffic engineering problem in communication networks. Pointurier and Heidari (2007) solve the routing problem in Optical Networks with physical impairment by developing a Linear Reward ϵ-penalty method. They disaggregate the wavelength assignment from the routing and opt for an first fitted heuristic. The experimental results show a lower blocking rate in comparison with the shortest path and uniform path selection strategies. By proposing a novel elaborate representation of the states in a DRL algorithm, Suárez-Varela et al. (2019) show an increase in the performance of applying DRL to the routing problem in Optical Transport Networks which outperforms traditional heuristics. A few studies use GNNs as a part of the RL approach to solve variants of the RWA problem. Swaminathan et al. (2021) developed a GNN-driven RL algorithm, GraphNet, for optimal routing of the incoming requests on software-defined networks (SDNs) to minimize the package delay. Unlike WDM, they do not consider the existence of multiple wavelengths over the fiber links of the network. In order to predict the distribution of per request delay in SDNs, Rusek et al. (2020) proposed a novel network-based on message-passing GNNs using Recurrent Neural Networks as aggregation functions. Similar to other algorithms devised for SDNs, RL-routing uses RL for routing packages with the objective of throughput and delay of the incoming traffic. A recent comprehensive survey on machine learning methods for optical networks and SDNs is presented in (Gu et al. 2020) and (Amin et al. 2021).

As discussed, most of the previous studies develop novel methods for the *routing* part of the problem, and resort to accepted heuristics for the *wavelength* assignment. A few other provide new approaches for wavelength assignment, but fail to fully consider the intertwined relation between routing and wavelength assignment, especially in a dynamic setting.

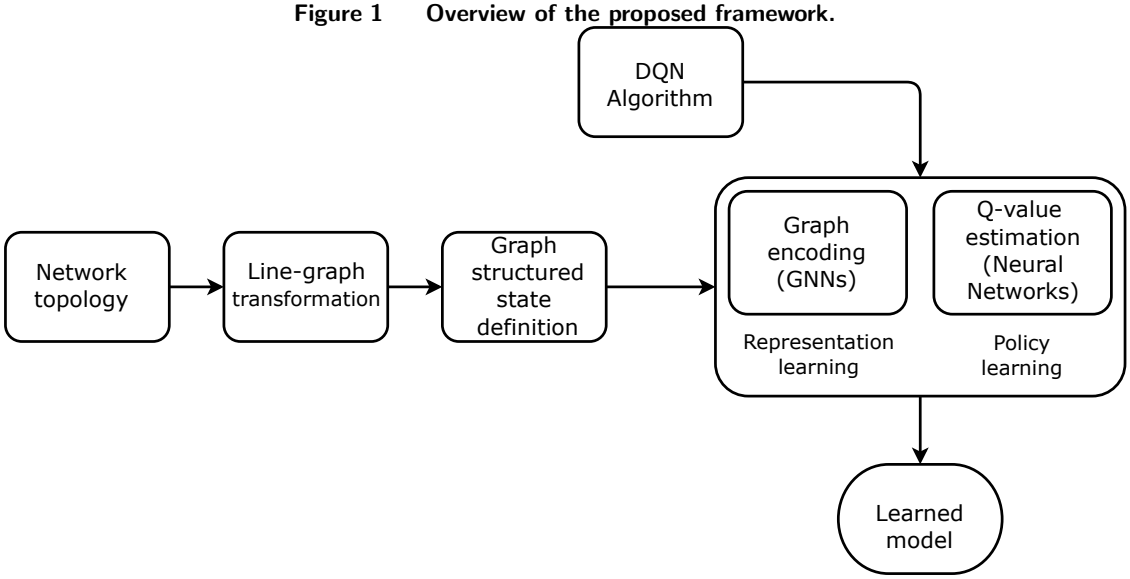
3. Methodology

We propose a solution method based on Graph Neural Networks (GNNs) and Deep Reinforcement Learning (DRL). We use GNNs to model the problem at each time so that a DRL agent be able to learn how to operate it to achieve an optimization goal. An overview of the proposed framework is presented in Figure 1.

3.1. Graph Neural Networks

GNNs (Scarselli et al. 2009) are novel neural networks that operate over graph-structured inputs. They generate a vectorial representation of an input graph. This is done by an iterative process

Figure 1 Overview of the proposed framework.



that associates the latent features of the neighbouring elements (nodes and edges) of a graph. Through this iterative message-passing algorithm, the features of these elements are updated to generate and learn an output vector. We use Graph Attention Networks (Veličković et al. 2018).

The final learned vectorial representation of the graph encodes hidden information of the graph that can be used to solve complex Combinatorial Optimization (CO) problems (Cappart et al. 2021). Since Network topologies can be directly translated to a graph, using GNNs offers more advantages than applying other neural networks architectures (fully connected neural networks, convolutional neural networks, and etc.). This potential performance is shown in recent works of (Rusek et al. 2019, Almasan et al. 2019).

3.2. Deep Reinforcement Learning

In Deep Reinforcement Learning, an agent interacts with the environment by performing actions and observing information about the environment (Sutton and Barto 2018). The information contains the state of the environment and an immediate reward, evaluating the performed action. DRL agent aims to maximize a long-term goal in an optimization problem. They have no prior knowledge of the environment. They learn the optimal policy by exploring the environment and this search is directed by a reward function. At each state $s \in S$, an action $a \in A$ is performed. The state is thereby transitioned to a new state, s', and the environment sends a scalar immediate reward r to the agent. The DRL agent finds a strategy to maximize the cumulative reward when the episode is finished. This problem can be modeled as a Markov Decision Process (MDP).

Q-learning is an RL algorithm that learns an optimal policy π, a mapping from states to actions (Watkins and Dayan 1992), and is used to solve MDPs. Throughout the algorithm, the value of performing an action in a state, called Q-value, is updated according to the rewards generated

by the environment. Q-values for different state-action pairs are initiated at zero or randomly generated values and are stored in a table (also known as Q-table). For a more sophisticated problem where the number of states or actions are larger, Q-table will be substantially large, and updating Q-values will be inefficient. In these situations, Q-values are directly approximated using deep neural networks. Deep Q-Network (DQN) (Mnih et al. 2013) is an algorithm used in such cases. DQN employs the advances of the deep neural networks to closely approximate Q-value functions by receiving the features of the states. The high generalization capabilities of deep neural networks enable the DRL agent to apply accurately approximate the Q-value for states never seen before during the training.

We consider a WDM optical network topology to be represented by a graph $G = (V, E)$ where V and E correspond to nodes and the fiber links of the physical network. For each fiber link, a set of possible wavelengths Ω indexed by ω exist. The number of possible wavelengths are fixed and denoted by $\mathcal{W} = |\Omega|$. All wavelengths are assumed to have identical capacity. At particular stochastic times τ, traffic demands are requested from the network. A traffic request is represented by source node n^1, destination node n^2 and a service time duration δ. We represent a lightpath request for an incoming traffic demand as ℓ_ω^κ where κ denotes the path between the source and the destination (among the k-shortest paths), and ω being the wavelength assigned to it. Feasible lightpaths are generated such that no two paths sharing a link are assigned the same wavelength and all the links throughout its path must use the same wavelength. The latter is called wavelength continuity constraint. Figure 2 depicts a few possible light paths for sample topology with $|\Omega| = 2$ and an arriving request.

In the static RWA problem, a set of predetermined, incoming traffic connection requests are given (Jaumard and Daryalal 2017). The objective of this problem is generally to minimize the number of used resources i.e. the number of wavelengths used (Zang et al. 2000).

The dynamic RWA problem is defined as follows. For a WDM network represented by a graph G, the problem finds a lightpath for an incoming traffic request at the time of its arrival. We assume that no wavelength conversion exists over the network; therefore, the wavelengths assigned must follow the wavelength continuity constraint. The traffic adds and drops over time by assigning new lightpaths and releasing of the wavelength by the end of their corresponding durations. The objective of the problem is to reduce the blockage rate of traffic requests over time.

Requests are generated by uniformly selecting a source destination from the set of the nodes. Each request has an arrival time and a duration. Arrival times follow a Poisson distribution and the durations are generated by an Exponential Distribution. These values represent the load of the network at each time of the simulation. Load is measured in Erlangs, and is calculated by multiplying the arrival rate and the average duration of the requests. It is therefore the average

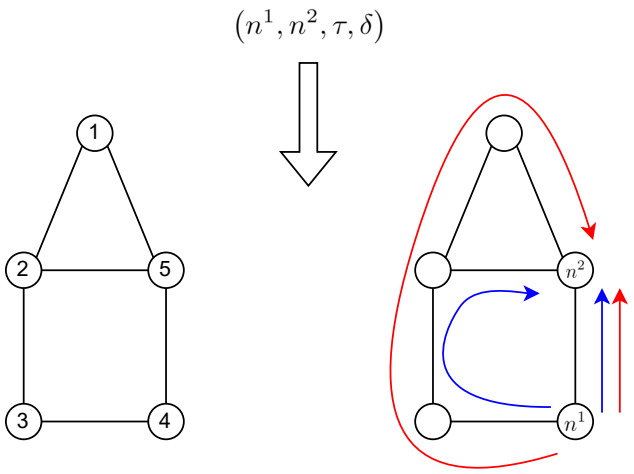


Figure 2 **Some possible lightpaths for a sample topology with $|\Omega| = 2$ and an arriving request between nodes n^1 to n^2 at arrival time τ and for duration δ.**

number of connections measured at any time in the network (Zang et al. 2001). No information regarding these distributions and the future is available to the agent at each decision point. This results in a challenging optimization problem for the agent to devise an optimal strategy.

3.3. DRL Environment

Here we describe the proposed DQN method to solve the dynamic RWA problem. At the arrival of a new incoming traffic request, the agent receives the state of the environment as a graph. The request contains the source, destination, and the duration of the package. The requests do not have any preference over individual routes or wavelengths. By observing the state, in particular node and edge features, the agent decides on the lightpath to assign to the current traffic request.

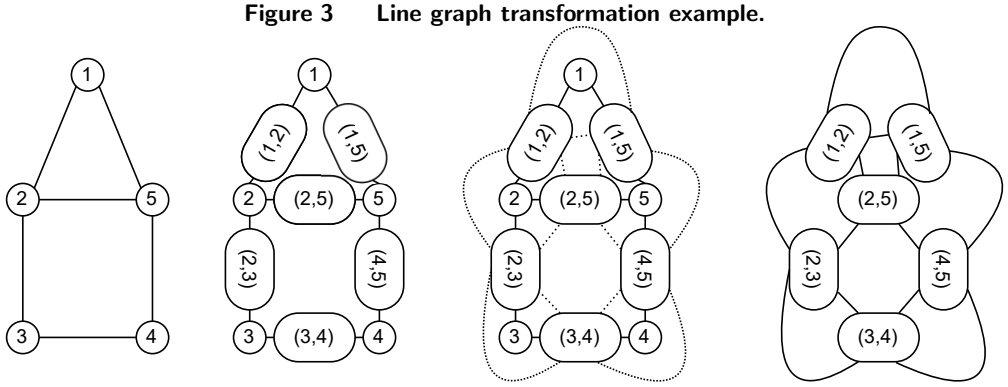
The role of the GNN is to receive the state as the graph, process it, and output a new representation. To this end, the GNN aggregates the information of the neighboring nodes and edges of the graph and encodes them with a new representation. This new encoded vectorized representation of the state graph is then forwarded to a fully connected neural network that approximates the $Q(s, a)$ for all the possible actions $a \in A$ (lightpath) to be performed in state s.

Given the underlying network, the links carry the related information such as capacity and the available wavelengths at the current time during the simulation. To reflect this in our implementation, we use the line-graph $\mathcal{L}$ transformation of the original network graph G. In such a transformation, $\mathcal{L}$ has a node for each edge in G, and an edge joins those nodes if the two edges in G share a common node. This conversion facilitates the message passing steps in the GNN. Figure 3 illustrates a sample line-graph transformation.

The environment of the proposed DQN algorithm is formally defined as follows.

State Upon receiving a traffic demand (n^1, n^2, δ), and consequently a lightpath request, the agent observes a graph-structured representation of the environment. The GNN module

Figure 3 Line graph transformation example.



retrieves the state features as well as the graph structure of the state s. The features of the graph elements illustrate the state. Node features include the wavelength assigned to a particular edge, the number of available wavelengths, the relative popularity of the wavelengths in previous selections, the remaining service times of the already assigned incoming request, and the edge betweenness centrality (Freeman 1977). The latter computes the number of all pair shortest paths that go through a particular edge in a graph network. The higher the betweenness centrality, the more likely is the edge to be selected in different paths. State features and their corresponding types are shown in Table 1.

Table 1 Features of each link in the graph-structured state

Feature	Description	Type
f_1	Wavelength assigned to the link	One-hot encoding vector
f_2	Remaining service time of traffic request	Vector
f_3	Betweenness centrality	Scalar
f_4	Popularity of the wavelengths	Vector
f_5	Number of available wavelengths	Scalar

Action Considering all possible paths between any selection of nodes of the underlying network graph results in large action space, even for small networks. There are numerous combination of links to create a path between every two node in a network. For instance, the average action space size considering all routes for the topologies used in this work is around 4500. This means that at each decision point, one of these actions need to be selected. This reduces the capabilities of the proposed algorithm as it needs to approximate Q-values for every possible path and every wavelength. To overcome this challenge, a limited number of paths are selected. For every source-destination node pair in G, the first k shortest paths are generated prior to training the model. Upon arrival of incoming traffic demand, the DQN agent receives the new representation of the state and the information of the current lightpath request (source, destination, and duration). Besides k paths to choose from, the agent needs to select a wavelength

assigned to every path. Therefore, the dimension of the possible actions is $\mathcal{W}k+1$. It should be pointed out that the action state is different based on incoming traffic requests since the source and destination nodes of requests are different. At each decision point, the action a is the decision to either (1) Assign a lightpath to the current request arriving at that time, or (2) reject the request (null action $a_\emptyset$).

$$a \in \{\ell_1^1, \ldots, \ell_{\mathcal{W}}^k\} \cup \{a_\emptyset\} \tag{1}$$

Reward The objective of the proposed method is to minimize the blocked traffic connections (or maximize admitted ones). To this end, A positive reward is returned by performing an action (not null action) corresponding to the action a successful provisioning of an incoming traffic connection request. Therefore the reward is constructed as:

$$r = \begin{cases} 1 & \text{if } a \neq a_\emptyset, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

3.4. Efficient Formulation

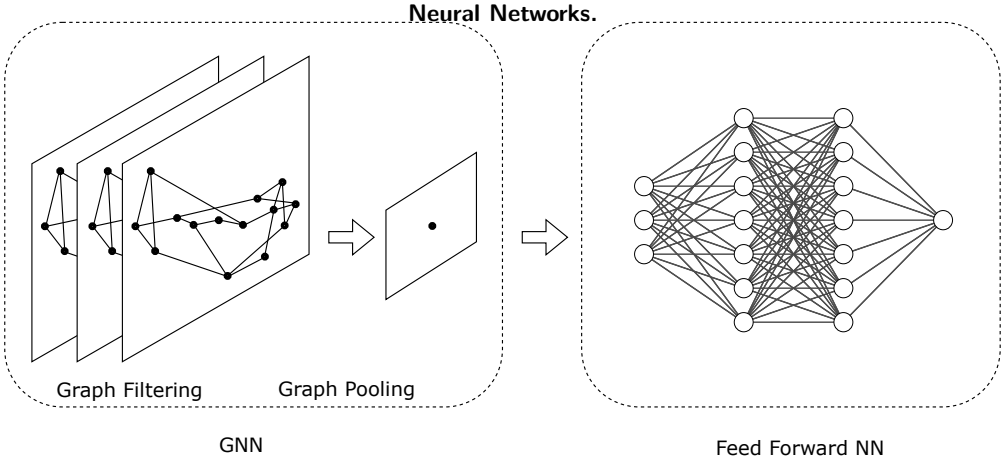
We observed that a large number of transitions (changing from the current state to the next state) are driven by the null action. As we assume the transition function is deterministic as a result of the selected action, such transitions are trivial. For instance, when all the potential lightpaths for an incoming request are not available, the decision is automatically driven by $a_\emptyset$. We propose a framework that only models transitions when $a \neq a_\emptyset$. In such a framework, a non-trivial state is defined at a situation where for an incoming request, at least one lightpath be available. Therefore, the cases $a = a_\emptyset$ do not contribute to the training process, meaning we ignore the trivial state-action pairs in our algorithm.

3.5. Neural Network Architecture

The Neural Network used in this paper is composed of two main modules, namely the GNN and the feed forward neural network that outputs the predicted Q-values. GNN is a process of representation learning of a graph, in particular the state graph in this problem. Therefore, the aim of this module is to learn the representative features of the entire graph by leveraging the node features and graph structure. The algorithm performs 3 steps of message passing and stacks the results. it takes node features and graph structure as an input and outputs a new set of node features. This step is called graph filtering. We use Graph Attention Networks (Veličković et al. 2018) to capture the information on the nodes of the state graph. For each node, the algorithm iterates over its neighbour and aggregates the related information from their features. Since we require to achieve a new graph-level representation, we need to generate features for the entire graph from the node features. Therefore, we apply a graph pooling layer which utilizes the graph

structure to guide the pooling process. To summarize, the GNN module takes the state graph as an input and outputs a coarsened graph with fewer nodes as an output. The described Neural Network architecture is are represented in Figure 4. This abstract, high-level representation of the state graph is then passed through a Feed Forward Neural Network to approximate the Q-values for the current state and action. We use 32 hidden layers, each with 128 hidden units and Rectified Linear Units (ReLU) activation function.

Figure 4 Neural Networks architecture is comprised of two modules: Graph Neural Networks and Feed Forward Neural Networks.



3.6. Learning Algorithm

The proposed algorithm is represented in Algorithm 1. In the beginning, we create a replay memory buffer to store the experiences at the memory initialization iterations and during the training phase (line 1). By initializing the environment and generating an instance (lines 7, 8), we create a graph G based on the underlying topology where the nodes and edges of the graph correspond to the nodes and the fiber links of the original network, respectively. Centrality betweenness is computed for all the links and $k = 4$ shortest paths are generated for each pair of nodes. As described in 3.3, we generate a line-graph $\mathcal{L}$ and the edge feature of G is converted to node features for $\mathcal{L}$. Arrival times and duration of service for each traffic request are created following a Poisson and an Exponential distribution, respectively. This information is not known to the agent, and they only become available as they are introduced at the corresponding arrival time (line 10). In general, the size of the output layer of the feed forward neural networks used in DQN algorithms equals the size of the action space. The value of each output unit in such a structure represents the predicted Q-value for each possible action at the current state of the environment. In this study, however, as the set of possible actions is different at each particular state, we embed the information of these possibilities into the state features. This is called the *action-in* network structure for DQN.

The output layer of the feed forward neural network is thus comprised of one unit, which also depicts the Q-value for the current state-action. To this end, at each state s, we allocate routes and wavelengths and compute Q-values (lines 11-15). A ϵ-greedy action selection strategy is devised for selecting action a from the set of possible lightpaths (line 16). The state is transitioned to a new state and a reward signal is sent to the agent (line 21). The current experience tuple (s, a, r, s') is added to the replay memory buffer and replaces a previous one with lowest priority (line 22). The environment awaits the arrival of the successor incoming request and upon arrival, the features of the new state are updated. The algorithm is repeated for a predefined number of episodes. The objective of the learning algorithm is to learn the weights of the feed forward neural vector such that the predicted Q-values accurately such that the expected long-term collected reward is maximized. In training, we batch samples of size 32 from the memory buffer. We obtain a new representation of the input graph-structured samples by applying graph filtering and graph pooling layers in our developed attention-based GNN. The output layer is then forwarded to the structure of a neural network for predictions. Once Q-values are computed and an action is selected, we update the weights of the neural network in order to minimize the loss between the predicted values and the real values (lines 25-29). Since we do not have access to real values, we approximate them following the target network (Sutton and Barto 2018). To this end, Adam optimizer (Kingma and Ba 2015) is used. When the episodes are all finished, we save the weights of the trained model for solving new instances (line 30).

After experimenting with different values for the learning rate during the training phase, a rate of 7×10^{-7} resulted in stable learning. By using larger learning rate in our experiments during the training phase, the performance on the validation set fluctuates after 5000 episodes. This means that the agent cannot reach near optimal solution as the step size in its search strategy is too large, although the search direction could be correct. By setting smaller learning rates, the algorithm converges but very slowly. At each iteration, we use a batch of 32 samples drawn from the experience replay buffer. Prior to the training phase, we store 50000 samples in the experience buffer using Prioritized Experience Replay (Schaul et al. 2016). Samples with the lowest priority are replaced with the new samples during training.

We select actions following the ϵ-greedy strategy. For the initial 200 iterations of training, the agent explores the environment with the exploration rate of $\epsilon = 1$. This rate is then decayed with a rate of 0.995 after every episode until it reaches the minimum value of $\epsilon = 0.01$. We also update the weights of the target network every 50 iteration by cloning the values from the policy network of our DQN agent. The weights of the policy network are initialized following Xavier initialization (Glorot and Bengio 2010).

Algorithm 1 DQN algorithm for the dynamic RWA Problem

memory_buffer $\leftarrow \{\}$
total_reward $\leftarrow 0$
w initial weights of the Neural Network
k-shortest-paths initiated for all source-destination pairs prior to training

for *episode* in $0, \ldots,$ num_episodes **do**
    environment $\leftarrow$ *initialize_environment*
    instance $\leftarrow$ *generate_instance*
    for i in $0, \ldots,$ num_requests **do**
        $s \leftarrow$ *(*$n^1, n^2, \delta, \tau,$ *current_time)*
        for κ in $1, \ldots, k$ **do**
            for ω in $1, \ldots, |\Omega|$ **do**
                q_values$[\kappa][\omega] \leftarrow$ *compute_q_values*
            end for
        end for
        $a \leftarrow$ *epsilon_greedy (q_values, s)* ▷ action a is comprised of a route and a wavelength (lightpath)
        if $a = a_\emptyset$ **then**
            Reject current request
            skip to the next request
        end if
        $s', r \leftarrow$ *make_step* (s, a)
        memory_buffer $\leftarrow (s, a, r, s')$
        total_reward $\leftarrow$ *total_reward* $+ r$
    end for
    for j in $0, \ldots,$ *batch_size* **do**
        $e \leftarrow$ *get_sample*
        $\mathcal{L}_j(w) \leftarrow$ *loss of* e
    end for
    update the weight vector **w**
    save **w**
end for

3.7. Solving Algorithm

Once the model is trained, we can use the saved model to solve new instances following the Algorithm 2. To this end, we take the best model observed during training. This is the model that resulted in the highest reward on the validation set (line 1). We initiate the total cumulative reward (line 2), the environment (line 4), and the instance (line 5). For each incoming connection request, a state graph is generated (line 7) and Q-values are computed for the corresponding action space (line 8-12). We use a greedy strategy to select the lightpath resulting in the largest estimated Q-value (line 13). The state is thereby transitioned into a new state and a positive reward is returned (line 14), and accumulated reward is updated (line 19). Once all the requests arrive, the accumulated reward, i.e. number of admitted incoming traffic connections is returned (line 21).

Algorithm 2 Solving Algorithm.

1: $\mathbf{w} \leftarrow$ best model saved during training
2: *total_reward* $\leftarrow 0$
3:
4: *environment* $\leftarrow$ *initialize_environment*
5: *instance* $\leftarrow$ *generate_instance*
6: **for** i in $0, \ldots,$ num_requests **do**
7: $s \leftarrow$ *(*$n^1, n^2, \delta, \tau,$ *current_time)*
8: **for** κ in $1, \ldots, k$ **do**
9: **for** ω in $1, \ldots, |\Omega|$ **do**
10: *q_values*$[\kappa][\omega] \leftarrow$ *compute_q_values*
11: **end for**
12: **end for**
13: $a \leftarrow \text{argmax}($*q_values,* $s)$
14: $s', r \leftarrow$ *make_step* (s, a)
15: **if** $a = a_\emptyset$ **then**
16: Reject current request
17: skip to the next request
18: **end if**
19: *total_reward* $\leftarrow$ *total_reward* $+ r$
20: **end for**
21: **return** *total_reward*

4. Experimental Results

In this section we evaluate the performance of the proposed algorithm, and corresponding results are discussed here as well.

4.1. Setup

Our model is implemented in Python 3.8 using PyTorch. Training is carried out on one GPU (NVIDIA V100 Volta, 32GB memory). Prior to training, we create a fixed *validation set* to keep track of the performance of the training and compare with baselines. This validation set is comprised of 100 randomly created instances. The mathematical programming formulation of the model is solved by Gurobi 9.5.0 with a system containing 16 GBs of RAM with 4 core Intel Core i7 processors.

Random instances are generated by producing random arrival times and service hold time for incoming traffic requests. In this work, we generate arrival times based on a Poisson distribution with a mean of 0.6. Service hold duration for the requests follow an Exponential distribution with the mean of 100. These values are selected after several experiments, in order to maintain the load of the network at an acceptable level. Having a very low load or a very high load will result in situations where the majority of the traffic requests are admitted or rejected. We select these values to keep the problem challenging for the RL agent. Prior to training, we solve these instances with widely used baselines in the literature. For routing, we select Shortest path selection and Alternate Routing methods (Wang et al. 2011). As for the wavelength selection, First fit heuristic is selected. We also developed a simple random selection strategy to evaluate the other methods against it.

In addition to these baselines, we assumed a case where we have perfect information of the arrival time and duration for all the incoming traffic requests. We formulated this problem with a mathematical programming model. Although this problem is in essence different with the dynamic RWA problem as we do not have stochasticity, its solution is an upper-bound for the problem studied in this work.

let $G = (V, E)$ be a graph generated from an underlying topology of the network. V is the set of nodes of the network. $\ell \in E$ denotes a fiber link of the network, joining a pair of nodes. Let D be the set of all incoming traffic requests, and Δ_d be the set of all possible lightpaths for a traffic request $d \in D$. We define a binary variable $x_{d\delta}, \forall d \in D, \forall \delta \in \Delta_d$ that is 1 if traffic request d is assigned to lightpath δ. In addition, a binary variable y_{dt} represents if request d is on the network at period $t \in \mathcal{T}$ where $\mathcal{T}$ is the set of all periods of the horizon. For each request d, we have complete information of the their arrival time arr_d, and duration dur_d. We define a parameter $q_{\ell\delta}$ equals to 1 if fiber link ℓ is assigned to lightpath δ and 0 otherwise. In addition, parameter $r_{\omega\delta}$ is 1 if wavelength $\omega \in \Omega$ is used in lightpath δ and 0 otherwise. The corresponding mathematical programming model for the described problem with perfect information on the requests is illustrated in equations (3) to (8):

$$\max \quad \sum_{d\in D}\sum_{\delta\in\Delta_d} x_{d\delta} \tag{3}$$

$$\sum_{\delta\in\Delta_d} x_{d\delta} \leq 1 \qquad \forall d\in D \tag{4}$$

$$dur_d \sum_{\delta\in\Delta_d} x_{d\delta} = \sum_{t\in\{arr_d,\ldots,arr_d+dur_d+1\}} y_{dt} \qquad \forall d\in D \tag{5}$$

$$\sum_{d:arr_d\in\{t-dur_d,t\}} \sum_{\delta\in\Delta_d} q_{\ell\delta}\, r_{\omega\delta}\, x_{d\delta}\, y_{dt} \leq 1 \qquad \forall t\in\mathcal{T},\ \forall \ell\in E,\ \forall \omega\in\Omega \tag{6}$$

$$x_{d\delta}\in\{0,1\} \qquad \forall d\in D,\ \forall \delta\in\Delta_d \tag{7}$$

$$y_{dt}\in\{0,1\} \qquad \forall d\in D,\ \forall t\in\mathcal{T}. \tag{8}$$

As mentioned before, the objective is to maximize the number of accepted requests which is provided is presented in equation (3). Constraints (4) ensures that at maximum one lightpath is selected for each incoming traffic request. As we consider a dynamic RWA, where the traffic requests are released after their service time is finished, we need to keep track of the times where the requests are utilizing the network resources. Constraints (5) are thereby defined to this end. In constraints (6), we make sure that each fiber link and each wavelength is at maximum used in one lightpath for the requests currently on the network at each time period.

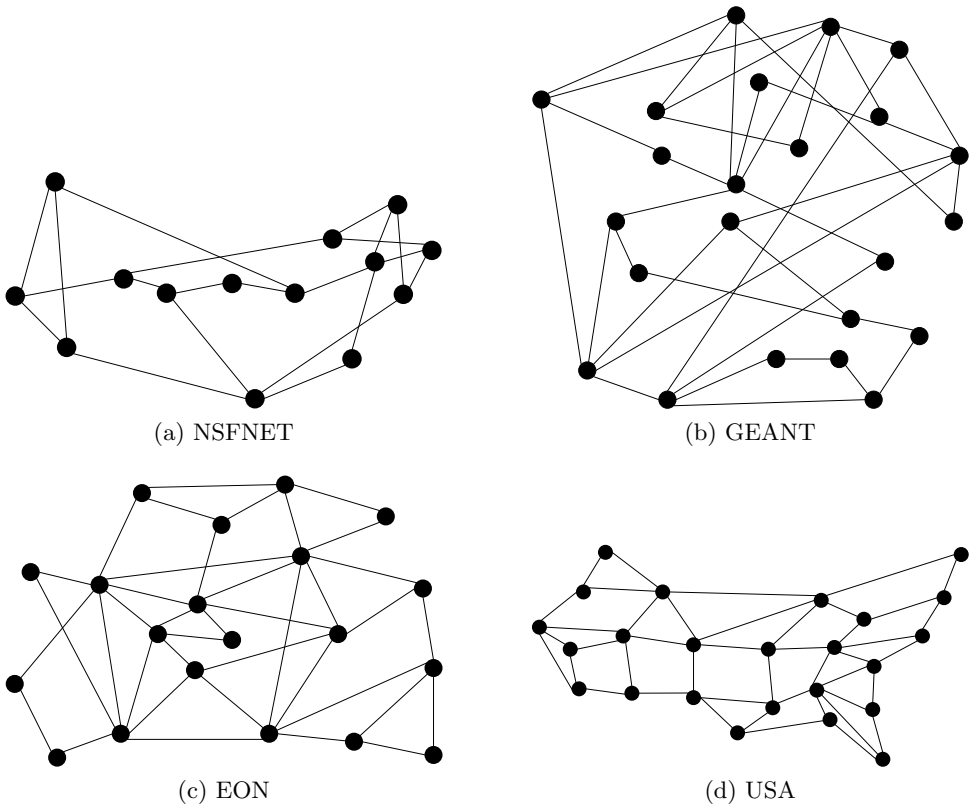
4.2. Experiments

Figure 5 represents the National Science Foundation Network (NSFNET) (Claffy et al. 1994), the Gigabit European Academic Network (GEANT), the European Optical Network (EON), and the USA network. We trained an RL agent for each topology.

For solving new instances, we chose the model that was learned on the NFSNET topology, as the network has a smaller size and lower average node degrees. We apply the learned model on all graph topologies. Table 2 depicts different topologies and their corresponding information which we used in this work. For each incoming traffic connection requests, the source and destination are selected randomly. The incoming traffic connection requests arrive following a Poisson distribution and each request stays on the network for a duration that follows an Exponential distribution.

Table 2 Different topologies

Topology	Nodes	Links	Avg. Degree
NSFNET	14	21	3.0
GEANT	24	37	3.1
EON	20	36	3.6
USA	24	44	3.7



(a) NSFNET (b) GEANT

(c) EON (d) USA

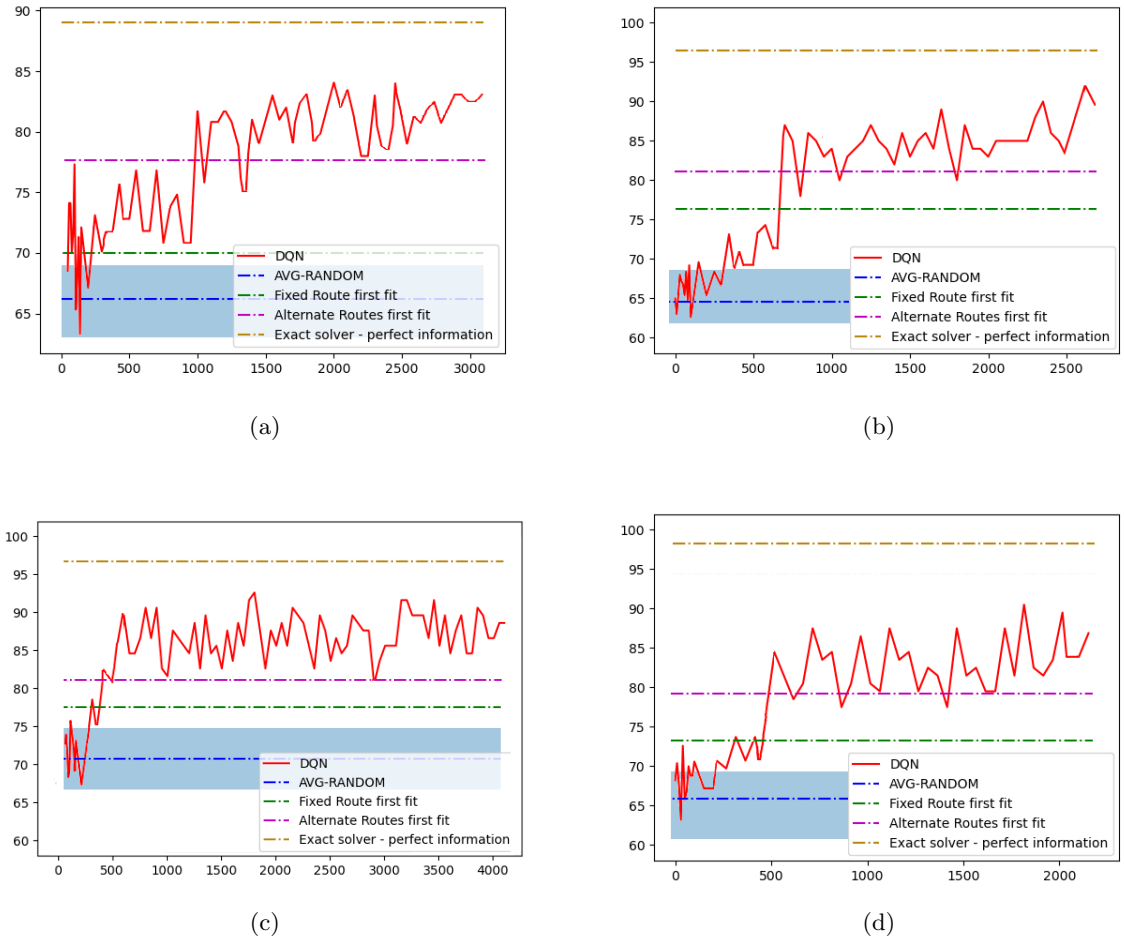
Figure 5 Network topologies.

In Figure 6, we demonstrate the performance of the learning algorithm for NSFNET (a), GEANT (b), EON (c), and USA (d) . We show the performance of the applied model at every 100 episode on the validation set. The results of applying the other baselines are also illustrated. These training curves suggest that the learning algorithm is able to infer the crucial information from the state (using GNNs), and take actions that result in maximum cumulative rewards. We can observe that after approximately 1000 episodes, the learning remains relatively stable. It should be pointed out that the ϵ parameter started to decay at iteration 200.

We ran experiments to evaluate the performance of the proposed algorithm and the learned model on new instances that did not participate in training. Since the objective is to minimize the number of rejected incoming traffic connection requests, we check the percentage of the blocked requests. In the first set of experiments, the generated instances are from the NSFNET topology. These include the instances used during training, or for solving after the model is learned. In the second set of experiments, we use the same learned model using NSFNET, but we apply this to instances generated from the other topologies (GEANT, EON, USA) that have not been seen during training.

To this end, we run four separate simulation sessions. We randomly create 100 instances of the problem. For each instance, we generate arrival time and service time duration following a Poisson

Figure 6 Evaluating the performance of the DQN algorithm during training for (a) NSFNET , (b) GEANT , (c) EON , and (d) USA topologies.



and an Exponential distribution, respectively. Within all the steps that the model is saved during the training phase, we select the one that revealed to be the best performing on the validation set to solve for the new instances. To be able to compare these results, we solve these instances with the accepted methods which are widely used in practice and in previous studies. For routing, shortest path and alternate routing are chosen. To assign the wavelengths, we opted for the first fit method which favors selecting more wavelengths against resulting in congestion and therefore blockage of the incoming requests. We also solved these new instance using a random strategy. At each decision point for an incoming traffic connection, the lightpath comprising of a route and a wavelength is generated randomly.

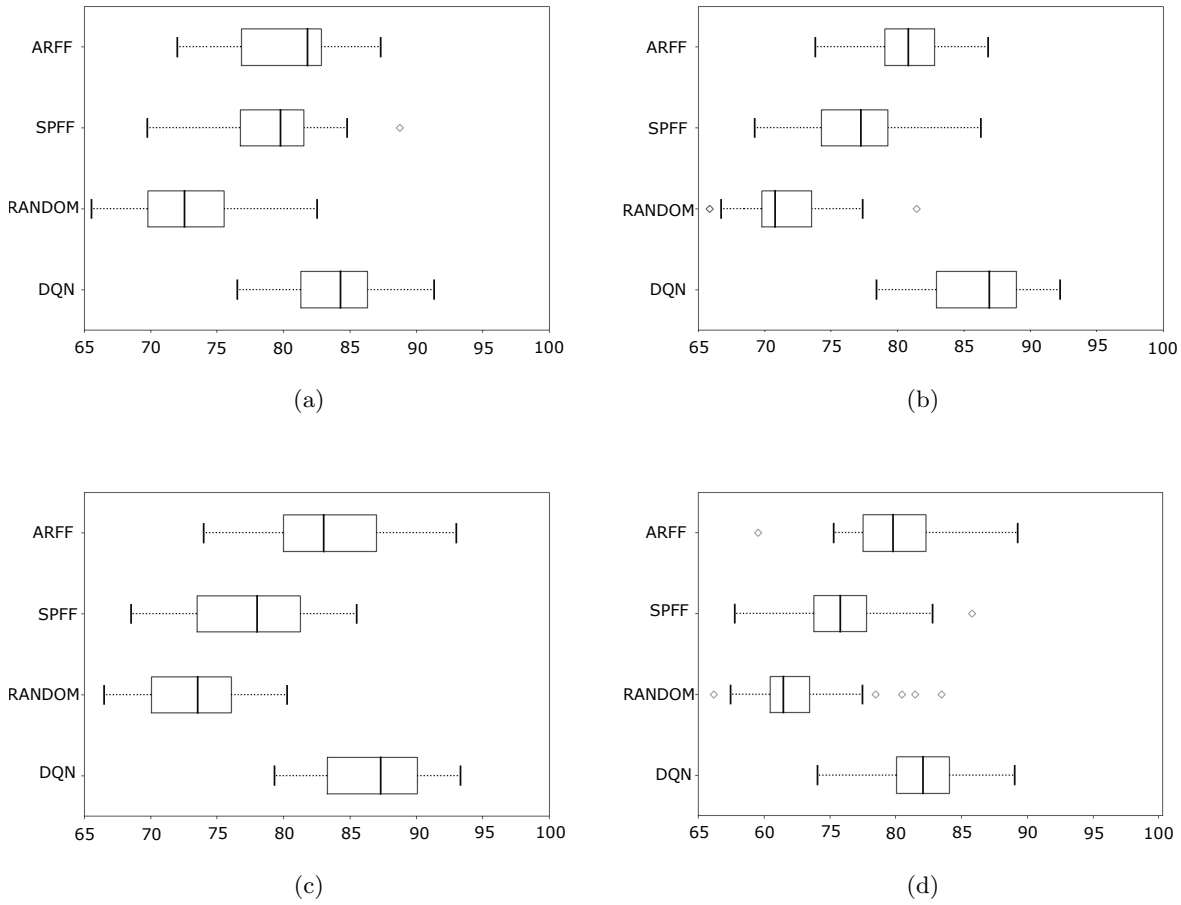
Figure 7 shows the percentage of blocked traffic connection requests for the described simulation scenarios. As demonstrated, the rate of blocking for the proposed method is less than the other baselines which is the goal of this algorithm. The random lightpath selection strategy shows the

worst performance. Despite the fact that for some cases the random selection could work better than other baselines, the median and the mean of the results for this strategy is lower than others. Although selecting the shortest paths at each decision point appears to be a logical approach, it only considers a myopic view of the problem. Always performing a greedy approach (by selecting the paths with smaller number of fiber links) may fall in local optimal points. As the results suggest, opting for the alternate routes proves to be better. Comparing the median and the mean of these two methods reveal the slightly better performance of the latter. The DQN-algorithm however, outperforms other approaches. In Figure 7(a), in the best case, almost less than 10% of the requests are blocked. The median of the results of the DQN algorithm is around 3% better than the one of the next best performing approach (alternate routing), and about 12% better than random policy.

For the results presented above, we apply the model learned on the NSFNET topology to the newly created instances that although did not participate in training, have the same underlying topology. To determine the generalization capability of the algorithm, we solve new instances from a different network topology (GEANT Figure 7(b), EON Figure 7(c), USA Figure 7(d)) and compare them with their corresponding baselines. Similar to the previous experiment, we generate 100 new instances following the new topology, and solve them with the proposed DQN agent that is learned over NFSNET. For instance, for GEANT network topology, random selection policy shows the worst performance. For this relatively bigger and more dense network topology, the relative performance of the alternate routing against the shortest path is better than the previous experiment. This is demonstrated via the difference in the median of these two methods across the 100 test instances. The DQN agent trained on NSFNET topology performs well on these instances compared to the best baseline. For EON and USA topologies, the performance of the best cases for the ARFF heuristic is the same as the DQN method (which is trained on the instances from the NSFNET). In both these cases however, the median of the DQN method has an edge over all the baselines.

To observe the characteristic of action selection between the different methods, we evaluate their corresponding action selection behavior. In Figure 8, we represent the average number of admitted incoming requests over the instances created at the test phase. Similar to above, the DQN agent is trained on NSFNET topology and the performance of the algorithm over test instance from NSFNET, GEANT, and EON topologies are illustrated in Figure 8(a), (b), and (c), respectively. The x-axis shows the number of the request and the values of the curves is the average of times a particular request is admitted. As expected, as the time goes on, considering the load, the network gets more congested and less possibilities exist. However, the long-term approach of the DQN method results in more average accepted traffic. The random strategy reveals to be the worst performing one. The alternate routing approach has an edge over the shortest path because of the

Figure 7 Comparison of performance of the DQN-algorithm and the baselines with the model trained on NSFNET. The results are shows for new instances that are generated following the (a) NSFNET , (b) GEANT , (c) EON , and (d) USA topologies.

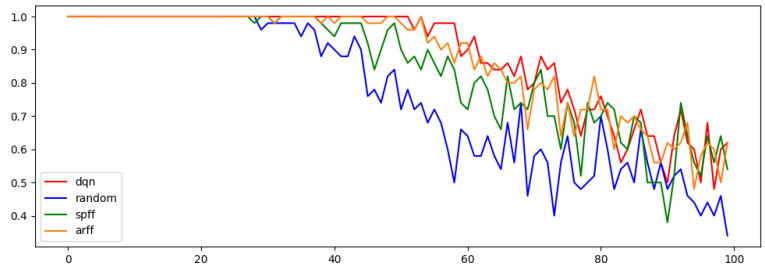


relatively myopic approach of the latter. Figure 9 demonstrates the length of the selected lightpaths for different methods for instances from NSFNET (a), GEANT (b), and EON (c). It could be observed that the random strategy selects lightpaths with relatively longer routes compared to other method; however, after arrival of around half of the incoming traffic, the capacities rapidly depleted and a decrease in admittance is observed. The length of the accepted routes also decrease. On the contrary, always opting to take shortest routes (greedily) results in suboptimal solutions. As this Figure suggests, the rejected connection request rapidly increase for he SPFF method.

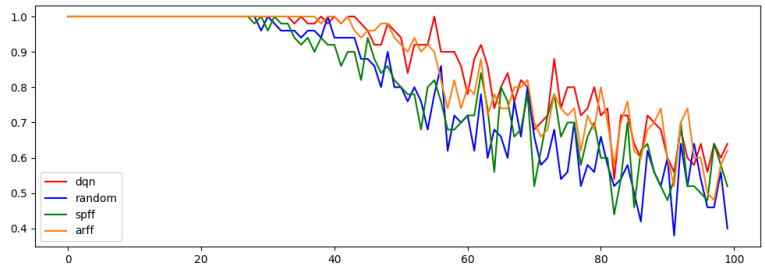
4.3. Conclusion

This paper studies a dynamic RWA algorithm based on Deep Reinforcement Learning that simultaneously selects routes and wavelengths. By inferring the information from the elements and the structure of the network, it devises a strategy to minimize the blockage of the incoming requests in the long term. Simulations show that the proposed algorithm performs superior to widely used

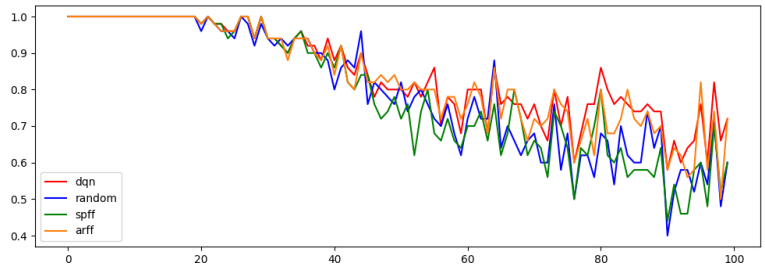
Figure 8 Average admitted traffic for each request over test set instances. (a) NSFNET, (b) GEANT, (c) EON, and (d) USA.



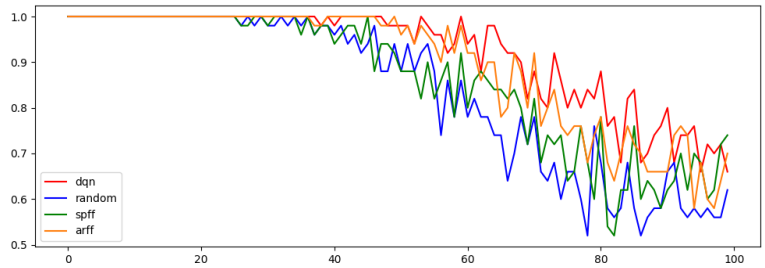
(a)



(b)

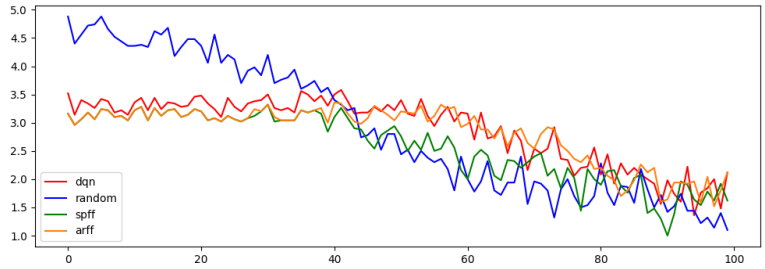


(c)

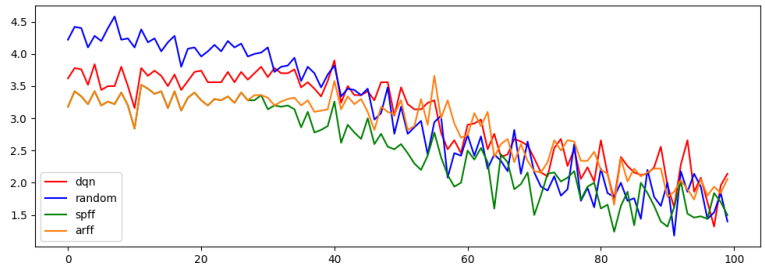


(d)

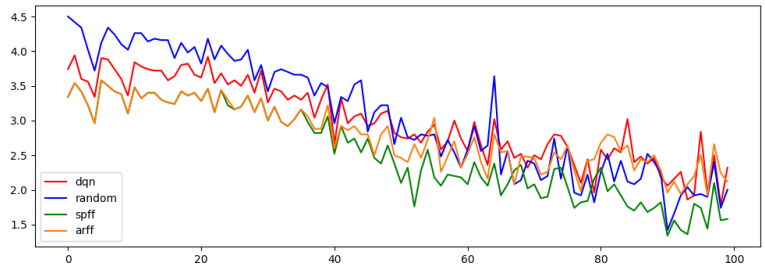
Figure 9 **Average length of the lightpath for each request over test set instances. (a) NSFNET, (b) GEANT, (c) EON, and (d) USA.**



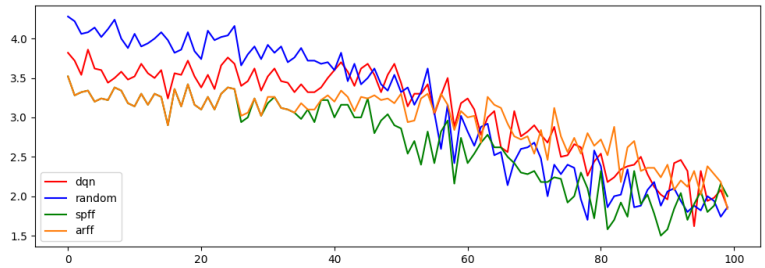
(a)



(b)



(c)



(d)

solution methods to solve the dynamic RWA problem. We demonstrate the generalization capability of the proposed model. The GNN module of the training learns a new representation of the graph-structured input regardless of the topology of the graph. In other words, computations within GNN module only takes into account node and edge features, which is similar in size for any network topology. Therefore, we can train the model using a single topology, and solve instances arising from variety of topologies (not present during training) which works better than the current methods used in practice.

References

Almasan P, Suárez-Varela J, Badia-Sampera A, Rusek K, Barlet-Ros P, Cabellos-Aparicio A (2019) Deep Reinforcement Learning meets Graph Neural Networks: exploring a routing optimization use case URL http://arxiv.org/abs/1910.07421.

Amin R, Rojas E, Aqdus A, Ramzan S, Casillas-Perez D, Arco JM (2021) A Survey on Machine Learning Techniques for Routing Optimization in SDN. *IEEE Access* 9:104582–104611, ISSN 2169-3536 VO - 9, URL http://dx.doi.org/10.1109/ACCESS.2021.3099092.

Cappart Q, Chételat D, Khalil E, Lodi A, Morris C, Veličković P (2021) Combinatorial optimization and reasoning with graph neural networks URL http://arxiv.org/abs/2102.09544.

Chen X, Li B, Proietti R, Lu H, Zhu Z, Yoo SJB (2019) DeepRMSA: A Deep Reinforcement Learning Framework for Routing, Modulation and Spectrum Assignment in Elastic Optical Networks. *J. Lightwave Technol.* 37(16):4155–4163, URL http://jlt.osa.org/abstract.cfm?URI=jlt-37-16-4155.

Chu X, Li B, Zhang Z (2003) A dynamic RWA algorithm in a wavelength-routed all-optical network with wavelength converters. *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, volume 3, 1795–1804, ISBN 0743-166X VO - 3, URL http://dx.doi.org/10.1109/INFCOM.2003.1209202.

Claffy KC, Braun HW, Polyzos GC (1994) Tracking long-term growth of the NSFNET. *Communications of the ACM* 37(8):34–45, ISSN 0001-0782.

Daryalal M, Bodur M (2020) Stochastic RWA and Lightpath Rerouting in WDM Networks URL http://arxiv.org/abs/2012.10084.

Di Ianni M (1998) Efficient delay routing. *Theoretical Computer Science* 196(1-2):131–151, ISSN 0304-3975, URL http://dx.doi.org/10.1016/S0304-3975(97)00198-9.

Duhovnikov S, Schupke DA, Lehmann G, Fischer T, Rambach F (2006) Dynamic RWA for All Optical Networks Using Linear Constraints for Optical Path Feasibility Assessment. *2006 European Conference on Optical Communications*, 1–2, ISBN 1550-381X VO -, URL http://dx.doi.org/10.1109/ECOC.2006.4801328.

Freeman LC (1977) A Set of Measures of Centrality Based on Betweenness. *Sociometry* 40(1):35–41, ISSN 00380431, URL http://dx.doi.org/10.2307/3033543.

Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 249–256 (JMLR Workshop and Conference Proceedings).

Gu R, Yang Z, Ji Y (2020) Machine learning for intelligent optical networks: A comprehensive survey. *Journal of Network and Computer Applications* 157:102576, ISSN 1084-8045, URL http://dx.doi.org/https://doi.org/10.1016/j.jnca.2020.102576.

Huimin G, Wen Y, Liang W, Zhengtang L (2017) Research on dynamic routing and wavelength assignment algorithm for optical networks. *2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 221–224, ISBN VO -, URL http://dx.doi.org/10.1109/IAEAC.2017.8054010.

Jaumard B, Daryalal M (2017) Efficient Spectrum Utilization in Large Scale RWA Problems. *IEEE/ACM Transactions on Networking* 25(2):1263–1278, URL http://dx.doi.org/10.1109/TNET.2016.2628838.

Jaumard B, Meyer C, Yu X (2005) When is wavelength conversion contributing to reducing the blocking rate? *GLOBECOM '05. IEEE Global Telecommunications Conference, 2005.*, volume 4, 2078–2083, URL http://dx.doi.org/10.1109/GLOCOM.2005.1578031.

Jaumard B, Meyer C, Yu X (2006) How much wavelength conversion allows a reduction in the blocking rate? *J. Opt. Netw.* 5(12):881–900, URL http://dx.doi.org/10.1364/JON.5.000881.

Kingma DP, Ba JL (2015) Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Kiran YV, Venkatesh T, Ram Murthy CS (2007) A Reinforcement Learning Framework for Path Selection and Wavelength Selection in Optical Burst Switched Networks. *IEEE Journal on Selected Areas in Communications* 25(9):18–26, URL http://dx.doi.org/10.1109/JSAC-OCN.2007.028806.

Martin I, Troia S, Hernandez JA, Rodriguez A, Musumeci F, Maier G, Alvizu R, de Dios O (2019) Machine Learning-Based Routing and Wavelength Assignment in Software-Defined Optical Networks. *IEEE Transactions on Network and Service Management* 16(3):871–883, URL http://dx.doi.org/10.1109/TNSM.2019.2927867.

Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing Atari with Deep Reinforcement Learning URL http://dx.doi.org/10.48550/arxiv.1312.5602.

Mohan N, Kaushal P (2020) Dynamic routing and wavelength assignment for efficient traffic grooming. *Journal of Optical Communications* URL http://dx.doi.org/doi:10.1515/joc-2019-0309.

Pinto A (2002) Optical Networks: A Practical Perspective, 2nd Edition. *Journal of Optical Networking* 1(6):219–220, URL http://opg.optica.org/jon/abstract.cfm?URI=jon-1-6-219.

Pointurier Y, Heidari F (2007) Reinforcement learning based routing in all-optical networks with physical impairments. *2007 Fourth International Conference on Broadband Communications, Networks and Systems (BROADNETS '07)*, 928–930, URL http://dx.doi.org/10.1109/BROADNETS.2007.4550535.

Pouya H (2018) *New Models and Algorithms in Telecommunication Networks*. Ph.D. thesis, Concordia University, URL https://spectrum.library.concordia.ca/id/eprint/984939/.

Ramaswami R, Sivarajan KN (1995) Routing and wavelength assignment in all-optical networks. *IEEE/ACM Transactions on Networking* 3(5):489–500, URL http://dx.doi.org/10.1109/90.469957.

Randhawa R, Sohal JS (2010) Blocking probability analysis of survivable routing in WDM optical networks with/without sparse-wavelength conversion. *Optik* 121(5):462–466, ISSN 0030-4026, URL http://dx.doi.org/https://doi.org/10.1016/j.ijleo.2008.08.007.

Rodriguez A, Fernández W, Ramírez L (2016) RWA: Novel Heuristic Algorithm for Optical Networks with Dynamic Traffic BT - Advanced Computer and Communication Engineering Technology. 1–10 (Cham: Springer International Publishing), ISBN 978-3-319-24584-3.

Rusek K, Suárez-Varela J, Almasan P, Barlet-Ros P, Cabellos-Aparicio A (2020) RouteNet: Leveraging Graph Neural Networks for Network Modeling and Optimization in SDN. *IEEE Journal on Selected Areas in Communications* 38(10):2260–2270, ISSN 1558-0008 VO - 38, URL http://dx.doi.org/10.1109/JSAC.2020.3000405.

Rusek K, Suárez-Varela J, Mestres A, Barlet-Ros P, Cabellos-Aparicio A (2019) Unveiling the Potential of Graph Neural Networks for Network Modeling and Optimization in SDN. *Proceedings of the 2019 ACM Symposium on SDN Research*, 140–151, SOSR '19 (New York, NY, USA: Association for Computing Machinery), ISBN 9781450367103, URL http://dx.doi.org/10.1145/3314148.3314357.

Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G (2009) The Graph Neural Network Model. *IEEE Transactions on Neural Networks* 20(1):61–80, URL http://dx.doi.org/10.1109/TNN.2008.2005605.

Schaul T, Quan J, Antonoglou I, Silver D (2016) Prioritized experience replay. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*.

Suárez-Varela J, Mestres A, Yu J, Kuang L, Feng H, Cabellos-Aparicio A, Barlet-Ros P (2019) Routing in optical transport networks with deep reinforcement learning. *J. Opt. Commun. Netw.* 11(11):547–558, URL http://dx.doi.org/10.1364/JOCN.11.000547.

Sutton RS, Barto AG (2018) *Reinforcement learning: An introduction* (MIT press), ISBN 0262352702.

Swaminathan A, Chaba M, Sharma DK, Ghosh U (2021) GraphNET: Graph Neural Networks for routing optimization in Software Defined Networks. *Computer Communications* 178:169–182, ISSN 0140-3664, URL http://dx.doi.org/https://doi.org/10.1016/j.comcom.2021.07.025.

Veličković P, Casanova A, Liò P, Cucurull G, Romero A, Bengio Y (2018) Graph attention networks. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*.

Wang Y, Cao X, Pan Y (2011) A study of the routing and spectrum allocation in spectrum-sliced Elastic Optical Path networks. *2011 Proceedings IEEE INFOCOM*, 1503–1511, ISBN 0743-166X, URL http://dx.doi.org/10.1109/INFCOM.2011.5934939.

Watkins CJCH, Dayan P (1992) \cal Q-Learning. *Machine Learning* 8(3-4):279–292, ISSN 0885-6125.

Xu Z, Tang J, Meng J, Zhang W, Wang Y, Liu CH, Yang D (2018) Experience-driven Networking: A Deep Reinforcement Learning based Approach. *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 1871–1879, URL http://dx.doi.org/10.1109/INFOCOM.2018.8485853.

Zang H, Jue JP, Mukherjee B (2000) A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks. *Optical networks magazine* 1(1):47–60.

Zang H, Jue JP, Sahasrabuddhe L, Ramamurthy R, Mukherjee B (2001) Dynamic lightpath establishment in wavelength routed WDM networks. *IEEE Communications Magazine* 39(9):100–108, ISSN 1558-1896 VO - 39, URL http://dx.doi.org/10.1109/35.948897.

Zhou J, Yuan X (2002) A study of dynamic routing and wavelength assignment with imprecise network state information. *Proceedings. International Conference on Parallel Processing Workshop*, 207–213, ISBN 1530-2016 VO -, URL http://dx.doi.org/10.1109/ICPPW.2002.1039732.

Zhou X, Lu W, Gong L, Zhu Z (2012) Dynamic RMSA in elastic optical networks with an adaptive genetic algorithm. *2012 IEEE Global Communications Conference (GLOBECOM)*, 2912–2917, URL http://dx.doi.org/10.1109/GLOCOM.2012.6503559.