NetworkAI: An Intelligent Network Architecture for Self-Learning Control Strategies in Software Defined Networks

Haipeng Yao, Tianle Mai, Xiaobin Xu, Peiying Zhang, Maozhen Li, Yunjie Liu

Abstract—The past few years have witnessed a wide deployment of software defined networks facilitating a separation of the control plane from the forwarding plane. However, the work on the control plane largely relies on a manual process in configuring forwarding strategies. To address this issue, this paper presents NetworkAI, an intelligent architecture for self-learning control strategies in SDN networks. NetworkAI employs deep reinforcement learning and incorporates network monitoring technologies such as the in-band network telemetry to dynamically generate control policies and produces a near optimal decision. Simulation results demonstrated the effectiveness of NetworkAI.

Index Terms—NetworkAI, Software Defined Networks, In-band Network Telemetry, Deep Reinforcement Learning.

1 Introduction

RECENTLY, Software Defined Networking (SDN) has received a large amount of attention from both academia and industry. The SDN paradigm decouples control plane from data plane and provides a logically-centralized control plane, wherein the network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted for the network applications and services [1]. This logically-centralized control mechanism provides the efficient use of network resources for network operators and the programmability brought by SDN simplifies the configuration and management of networks. Therefore, network operators can easily and quickly configure, manage and optimize network resource in SDN architecture [2]. However, even though a separation of the control plane from the forwarding plane facilitates large scale and dynamic networks, a challenge that remains untackled is that the work on the control plane relies heavily on a manual process in configuring forwarding strategies.

Finding the near-optimal control strategy is the most critical and ubiquitous problem in a network. The majority of approaches to solve this problem today usually adopted the white-box approaches [3], [4]. With the expansion of network size and the rapid growth of the number of network applications, current networks have become highly dynamic, complicated, fragmented, and customized. These requirements pose several challenges when applying these traditional white-box algorithms. Specifically, a white-box approach generally requires an idealized abstraction and simplification of the underlying network; however, this idealized model often poses difficulties when dealing with a real complex network environment. In addition, the white-box method presents poor scalability under different scenarios and applications.

Owing to the success of Machine Learning (ML) related applications, such as robotic control, autonomous vehicles, and Go [5], a new approach for network control through ML has emerged. This new networking paradigm using is firstly proposed by A. Mestres et al., which is referred to as Knowledge-Defined Networking (KDN) [6]. However, KDN and some other similar works [7], [8] just proposed some concept, no detail was described in these papers and no actual work has been implemented.

In this paper, we propose NetworkAI, an architecture exploiting software-defined networking, network monitor technologies (e.g., traffic identification, In-band Network Telemetry(INT)), and reinforcement learning technologies for controlling networks in an intelligent way. NetworkAI implements a network state upload link and a decision download link to accomplish a close-loop control of network and builds a centralized intelligent agent aiming at learning the policy by interacting with the whole network. The SDN paradigm decouples control plane from data plane and provides a logically-centralized control to whole underlying network. Some new network monitor technologies, such as In-band Network Telemetry(INT), [9] can achieve millisecond uploading of the network state and provide real-time packet and flow-granularity information to a centralized platform [10]. In addition, a network analytical platform, such as the PNDA [11], provides big data processing services via some technologies such as Spark and Hadoop. SD-

- *Haipeng Yao is with State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China.*
  E-mail: yaohaipeng@bupt.edu.cn
- *Tianle Mai and Xiaobin Xu are with the Beijing Advanced Innovation Center for Future Internet Technology, Beijing University of Technology, Beijing 100124, China.*
- *Peiying Zhang is with the College of Computer & Communication Engineering, China University of Petroleum (East China), Shandong 266580, China.*
- *Maozhen Li is with the Department of Electronic and Computer Engineering, Brunel University London, Uxbridge, UB8 3PH, UK.*
- *Yunjie Liu is with State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China.*

Manuscript received Jan 20, 2018; revised Jun 27, 2018.

N and monitor technologies offer a completely centralized view and control to build the interaction framework of a network, thus enabling the application of ML running in a network environment to address the network control issues.

NetworkAI applies DRL to effectively solve real-time large-scale network control problems without relying on too much manual process and any assumptions of the underlying network. RL involves agents that learn to make better decisions from experiences by interacting with the environment [9]. During training, the intelligent agent begins with no prior knowledge of the network task at hand and learns by reinforcement based on its ability to perform a task. Particularly, with the development of deep learning (DL) techniques, the success of combined application of RL and DL to large-scale system control problems (such as GO [5] and playing games) proves that the deep reinforcement learning (DRL) algorithm can deal with the complicated system control problem. DRL represents its control policy as a neural network that can transfer raw observations (e.g., delay, throughput, jitter) to the decision [12]. Deep learning (DL) can effectively compress the network state space thus enabling RL to solve large-scale network decision-making problems that were previously found difficult in handling high-latitude states and motion space.

In NetworkAI, the SDN and new network monitor technologies are employed herein to construct a completely centralized view and control for geographical distributed network and build a centralized intelligent agent to generate a network control policy via DRL. The NetworkAI can intelligently control and optimize a network to meet the differentiated network requirements in a large-scale dynamic network.

Different from the traditional white-box approaches, this paper proposes a new network paradigm (i.e. NetworkAI) in applying ML to solve network control problem. The main contribution of this paper can be summarized as follows:

- We employ the SDN and INT to implement a network state upload link and a decision download link to accomplish a close-loop control of a network and build a centralized intelligent agent aiming at learning the policy by interaction with a whole network.
- We apply DRL to effectively solve real-time large scale network control problems without too much manual process and any assumptions of underlying network, where the DRL agent can produce a near-optimal decision in real time.

The rest of this paper is organized as follows. In Section 2, we review state-of-the-art works in area of applying ML to network control. In Section 3, we introduce the each plane of NetworkAI architecture briefly. In Section 4, we elaborate the NetworkAI operation mechanism and the policy generation algorithm. In Section 5, a use case that shows the applicability of such a paradigm is described. In Section 6, we conclude the paper by analyzing the open research challenges associated with the NetworkAI paradigm.

2 Related Works

There are a number of works focusing on how SDN networks can be made more intelligent in network design and management. Some new network paradigms are designed in order to achieve intelligent control of SDN. A recent work [6] refers to this paradigm as KDN. In KDN, a knowledge plane is added on top of control plane, which in order to provide automated network control of networking. [7], [8] proposed similar architectures named data-driven network. However, these studies only aim to put forward a concept, but do not describe in detail and thus no actual work has been implemented.

In addition, different machine learning approaches have been applied to achieve intelligent managing network application in the SDN. There is a large amount of literature on applying supervised learning to solve specific problems for a variety of network applications, such as resource allocations [13], web services [14], [15], Internet telephony [16], and video streaming [17]. However, supervised learning methods suffer from data bias and slow reaction problems [18] in a dynamic network control application. At the same time, RL has also been used to solve network control problems, such as routing selection [19], [20], CDN selection [18], and virtual network embedding [21]. However, such proposals use table-based RL agents to solve small sized problems. When the size of a network increases, memory and computation related complexities are the main challenges in using traditional RL.

3 Network Architecture

In this section, we firstly introduce the NetworkAI architecture and elaborate how it operates. The model of the NetworkAI is shown in Fig.1. This model consists of three planes called the forwarding plane, the control plane, and the AI plane. And then, we will give a detailed description of each layer.

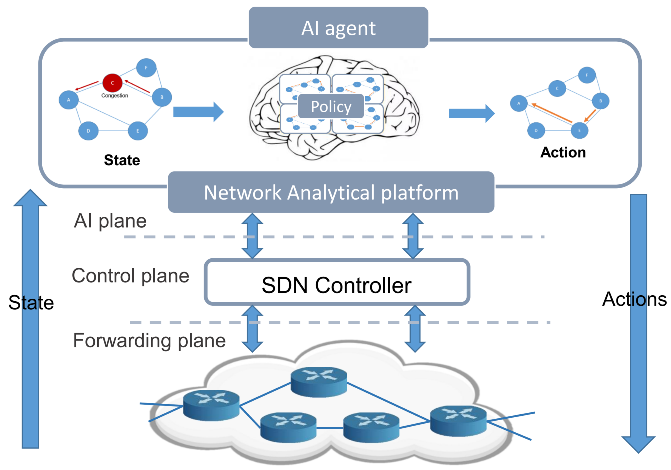


Fig. 1. The NetworkAI architecture

3.1 Forwarding Plane

The function of the forwarding plane is forwarding, processing, and monitoring data packets. The network hardware, which is composed of line-rate programmable forwarding hardware, only focuses on simply data forwarding without embedding any control strategies. The control rules are issued by the SDN controller via southbound protocols such as OpenFlow [22] or P4 [23]. When a packet comes into the node, it will be forwarded and processed according to these rules. Besides, there are some monitoring processes embedded in nodes. The network monitor data will be collected and sent to the analytical platform. Thus, it can offer complete network state information to facilitate the AI plane to make decisions.

3.2 Control Plane

The function of the control plane is to connect the AI plane and the forwarding plane. This plane provides abstractions for accessing lower-level geographical distributed forwarding plane and pools the underlying resources (such as link bandwidth, network adapter, and CPU capacity) to the AI plane. The function of the SDN controller is to manage the network through standard southbound protocols and interact with the AI plane through the northbound interfaces. This logically-centralized plane eases the burden of the network control problem imposed by a geographical distributed network. Thus, the policies generated by the AI plane can be quickly deployed into the network.

3.3 AI Plane

The function of the AI plane is to generate policies. In the NetworkAI paradigm, the AI plane takes advantage of SDN and monitor techniques to obtain a global view and control of the entire network. The AI agent learns the policy through interaction with the network environment. While learning the policy is a slow process, the network analytical platform provides a big data storage and computing capacity. Fundamentally, the AI agent processes the network state collected by the forwarding plane, then transforming the data to a policy through RL and using that policy to make decisions and optimization.

4 Network Control Loop

In the NetworkAI architecture, we design the network state upload link and the decision download link to accomplish a close-loop control of network. The NetworkAI architecture operates with a control loop to provide an interactive framework for a centralized agent to automatically generate strategies. Now, in this section, we will detail how the NetworkAI architecture implement a control loop of the whole network and how the intelligent agent learns the policy by RL approach.

4.1 Action Issue

In traditional distributed networks, the control plane of the network node is closely coupled with the forwarding plane which has only partial control and view over the complete network. This partial view and control can lead to no global convergence of learning result. The AI agent need to continually converge to a new result when network state changed which will lead to a bad performance in real-time control problem. For purpose of achieving global optimum, the controlling and managing of the whole network is premise.

SDN is a paradigm which separates the control plane from the forwarding plane and therefore breaks the vertical integration. The SDN controller treats the entire network as a whole. In this manner, the SDN acts as a logical-centralized agent to control the whole network. The SDN controller issues a control action through an open and standard interface (e.g., OpenFlow, P4). These open interfaces enable the controller to dynamically control heterogeneous forwarding devices, which is difficult to achieve in traditional distributed networks.

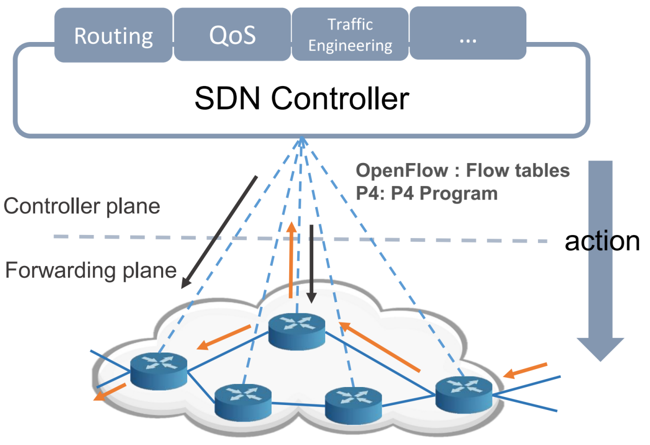


Fig. 2. The action issuing process

As demonstrated in Fig. 2, the agent can issue control actions to the forwarding plane via southbound protocols according to the decisions made at the AI plane, the network node at the forwarding plane operates based on the updated rules imposed by the SDN controller [6]. In this manner, we realized the global controllability of the entire network.

4.2 Network State Upload

In the SDN network architecture, the controller can send an action decision to the underlying network with an aim to acquire a complete network control. Furthermore, obtaining the complete real-time view of the whole network is also crucial to make near-optimal decisions. The most relevant data that should be collected is network state information and traffic information. To this end, we designed the upload link to access fine-grained network and traffic information. In this subsection, we will introduce how the NetworkAI architecture achieves network state upload.

1. Network Information Network information mainly refers to the status of the network device (information below the L2 layer), including the network physical topology, hop latency, and queue occupancy. In our architecture, we borrow in-band telemetry technology to achieve fine-grain network monitoring.

Obtaining fine grained network monitoring data of dynamic networks is a concern of NetworkAI. The traditional monitor technologies are commonly based on out-band approaches. In this way, monitoring traffic is sent as dedicated traffic, independent from the data traffic (”probe traffic”), such as SNMP, synthetic probes. These methods bring to much probe traffic in networking and overhead computation to control plane in large-scale dynamic networking which extremely degrades the performance of real-time controlling.

In-band network telemetry is a framework designed to allow the collection and reporting of network state by the data plane, without requiring intervention or computation by the control plane. The core idea of INT is to write the network status into the header of a data packet to guarantee the granularity of monitoring the network at the packet level [24]. The telemetry data is straightforward adding to the packet. Therefore, the end-to-end monitoring data can be retrieved from the forwarding node through Kafka or IPFIX directly to AI plane's big data platform without intervening by the control plane.

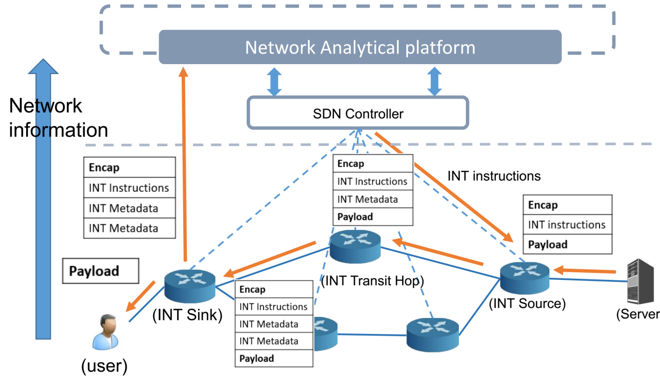


Fig. 3. The collection process of network information

The INT monitoring technology model is illustrated in Fig. 3. From the Fig. 3, we can see that a source node embeds instructions in the packets, listing the types of network information needs to be collected from the network elements (e.g., hop latency, egress port TX link utilization, and queue occupancy). Each network element inserts the requested network state in the packet as it traverses the network. When the packet is sent to the INT Sink, the load data is sent to the user and the telemetry data is sent to the network analytical plane.

Data collection can be realized based on the actual traffic. INT provides the ability to observe and collect real-time and end-to-end network information across physical networks. In addition, the mechanism of INT vanishes the overhead communication of probe traffic and overhead computation of control plane. Due to borrowing from INT technology, the AI plane can obtain millisecond fine grain network telemetry data, which gives the possibility to react network in time.

2. Traffic Information Traffic information mainly include service-level information (e.g., QoS/QoE), anomaly traffic detection information(e.g., elephant flow), etc. In network, different applications produce various traffic types with diverse features and service requirements. In order to better manage and control networking, the identification of network traffic plays a significant role [25].For instance, elephant flow is an extremely large continuous flow established by a TCP (or other protocol) flow [26]. Elephant flows can occupy network bandwidth and bring seriously congestion to the network. Therefore, it is of great significance for AI plane to detect or even predict the elephant flow in time and take the necessary action to avoid network congestion.

In our architecture, as illustrated in Fig. 4, several monitor processes embedded in some nodes to transfer the raw traffic data(e.g., flow granularity data, relevant traffic feature, and Deep Packet Inspection (DPI) information) to the traffic information via data mining methods, such as traffic classification and traffic anomaly detection [27], [28]. Then, the traffic information will be upload to the network analytical plane to assist AI plane in decision making.

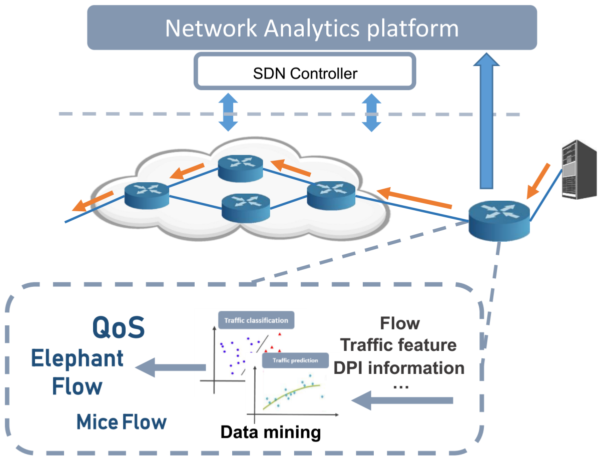


Fig. 4. The collection process of traffic information

4.3 Policy Generation

With the aim to apply ML method to realize an intelligent network control, we have already constructed the interaction framework between an AI agent and the network environment. In this section, we will describe how to use ML method to generate the network policy. The machine learning methods contain three approaches, supervised learning, unsupervised learning and reinforcement learning. Compared to supervised and unsupervised learning, reinforcement learning is more suitable for close-loop control problems. In particular, with the development of DL, the success of combining DL and RL for applications in decision-making domains (Playing Atari with Deep Reinforcement Learning by DeepMind at NIPS 2013 and the 2016 Google AlphaGo success on Go) demonstrates that DRL can effectively solve large-scale system control problems. Thus, we apply RL method to solve the large-scale network control problem.

RL based learning tasks are usually described as a Markov decision process, as shown in Fig. 5. At each step, an agent observes the current state s_t from the network environment and the agent takes an action at according to a policy $\pi(a|s)$. Following the action, the network environment transfer to state s_t+1 and the agent observes a reward signal r_t from environment. The goal of the reinforcement learning is to obtain the optimal behavior policy that maximizes the expected long-term reward. Specifically, in the network scenario, the state is represented by the network state and flow information, the action is by network behaviors (e.g.,

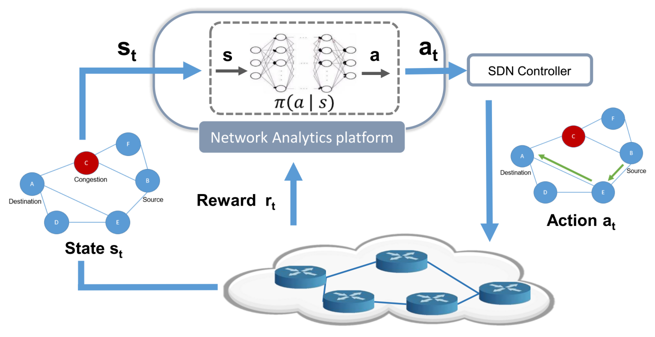


Fig. 5. The reinforcement learning process

CDN selection, routing selection) and the reward is based on the optimal target.

The RL agent uses a state-action value function $Q(s, a)$ to measure the actions expected for a long-term reward on the state s. Starting from a random Q-function, in q-learning algorithm, the agent continuously updates its Q-values by:

$$Q(s_t, a_t) \overset{\alpha}{\leftarrow} r_{t+1} + \lambda Q(s_{t+1}, a_{t+1}) \tag{1}$$

where $x \overset{\alpha}{\leftarrow} y \equiv x \leftarrow x + \alpha(y - x)$ and λ is the discount parameter. Using these evolving Q-values, the agent chooses the action with the highest $Q(s, a)$ to maximize its expected future rewards.

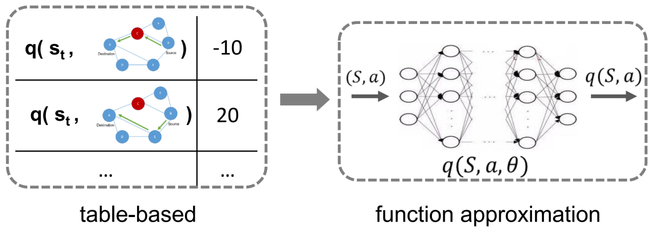


Fig. 6. The compress process of mass states

Particularly, the combination of DL and RL takes a step further in complex system control. The traditional RL algorithm records the reward of $(state, action)$ through the table-based method, which will lead to complexity issues that the RL is not designed for, namely memory complexity, computational complexity, and sample complexity, as its use was inherently limited to low-dimensional problems [29]. Specifically, in a large-scale and highly dynamic network, there are too many $(state, action)$ pairs. It is often impractical to maintain the Q-value for all possible $(state, action)$ pairs. Hence, it is common to use a parameterized function $Q(s, a; \theta)$ to approximate $Q(s, a)$. Deep neural networks have powerful function approximation and representation learning capabilities [30]. The DL algorithm can automatically extract low-dimensional features from high-dimensional data. Therefore, DL can effectively compress the network state space, as illustrated in Fig. 6, thus enabling RL to solve large-scale network decision-making problems that were previously found difficult in handling high-latitude states and motion space.

Based on such reinforcement learning framework, the data flow in NetworkAI is described as follow. The network monitor data and traffic information will be collected by the upload link, the decision for each flow calculated in AI plane send to SDN controller via northbound Interface. The SDN control then issue the rule through southbound interface. Thus, this data flow achieves RL agent through interaction with the underlying network. Different applications just need craft the reward signal to guide the agent toward good policy to meet their objectives.

In our architecture, we apply deep reinforcement learning to generate network policy. Combining RL with DL leads to a general artificial intelligence solution for solving complex network control problems. We believe that introducing DRL for network decision making presents two main advantages.

First, the DRL algorithm is a black-box approach. The DRL agent only need to have different network decision tasks and optimization goals in designing action spaces and rewards without changing the mathematical model. In addition, because an artificial neural network has the characteristic of expressing arbitrary nonlinear mappings, the DRL agent can understand a nonlinear, complex, multi-dimensional network control problem without the need of any simplifications. On the contrary, traditional white-box approaches require assumptions and simplifications of the underlying network aiming at building the equivalent mathematical model and tailoring for a problem that has to be optimized.

Second, the DRL agent does not need to converge again when network state changed. Once the DRL agent trained, an approximate optimal solution can be calculated in single step through matrix multiplication where the time complexity is only approximately $O(n^2)$, where n is represented by number of network nodes. In contrast, the heuristic algorithms need take many steps to converge to a new result, where leads to high computational time cost. For example, the time complexity of ant colony algorithm is $O(n \times (n-1) \times mt)$, where n is represented by number of network nodes , m is number of ants , t is number of iterations. Therefore, DRL offers a tremendous advantage for the real-time control of a dynamic network.

Above all, the NetworkAI achieves applying RL approach for the real-time control of the network. The SDN, INT and traffic identification technologies are used to implement the network state upload link and the decision download link respectively with an aim to obtain a centralized view and control of a complex network systems. In addition, DRL agent in AI plane can effectively solve complexity network control problem without the need of any simplifications of real network environment. Furthermore, that a near-optimal solution can quickly be calculated once it is trained represents an important advantage for the real-time control of the network. Thus, the NetworkAI facilitates an automated control of a large-scale network.

5 Use Case

The main objective of this use case is to demonstrate that it is possible to model the behavior of a network with the proposed NetworkAI architecture. In particular, we present a simple example in the context of QoS routing, where the NetworkAI was used to make intelligent decisions to select the best routing path aiming at satisfying the QoS requirements.

The traditional Internet design is based on end-to-end arguments with an aim to minimize the network supports. This type of architecture is perfectly suited for data transmission where the primary requirement is reliability [7]. However, with the proliferation of various applications (such as multimedia transmission application, where timely delivery is preferred over reliability), the demands of each application are different. Thus, the network should support QoS in a multi-application traffic scenario. However, the question of how to support end-to-end QoS is an on-going problem.

QoS routing mainly involves path selection that meets the QoS requirements arising from different service flows. It is a mechanism of routing based on QoS requests of a data flow and the available network resources. The typical QoS indicators for applications in the network are different, as demonstrated in Table. 1 in which we list the QoS requirements for several applications.

TABLE 1
The QoS indicator for several applications

QoS / Application	Delay	Throughput	Jitter	Losses
realtime multimedia	√	√	√	
Augmented Reality or Virtual Reality	√	√	√	√
VoIP	√		√	
scheduling in datacenters	√	√	√	√
internet of vehicles	√			√

The dynamic QoS routing can be seen as a Constrained Shortest Path (CSP) problem, which is an NP-complete problem [31]. Although researchers from both academia and industry have proposed many solutions to solve the QoS limitations of the current networking technologies [19], [20], [31], many of these solutions either failed or were not implemented because these approaches come with many challenges. The tradition heuristic methods bring high computational time cost in network control, which is difficult to apply to real-time dynamic network environment.

The NetworkAI paradigm can address many of the challenges posed by the QoS-Routing problem. RL approach can calculate an near-heuristic solution in one step, which can benefit for real-time control to large-scale network. And the closed-loop architecture in NetworkAI provide an interaction framework which achieves applying DRL in geo-distributed network. In our experiment, we applied the Deep Q-learning algorithm in a central agent, aimed to select the best routing path to minimize the network delay [32].

Methodology We simulated a network with 36 nodes and 50 full-duplex links, with uniform link capacities and different transmission delay. One of these nodes applied

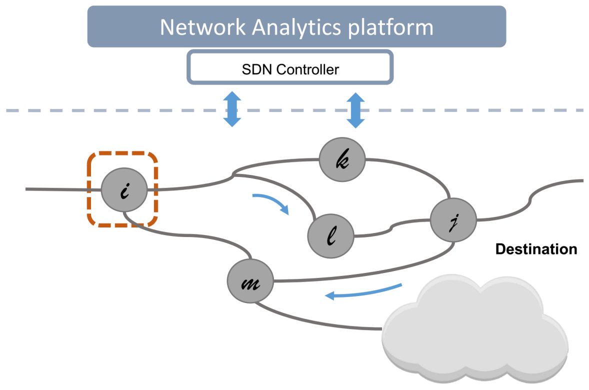


Fig. 7. The experiment example

DQN and the others run short path algorithm. In addition, we devised 10 traffic intensity (TI) levels, ranging from 1 to 10, which represent the volumes of the network traffic at different times. The traffic generated randomly in each node and volumes is subject to poisson distribution.

As shown in Fig . 7, it is a part of the whole experiment topology. We applied DQN to guide node i's routing action. In our experiment, state is represented by the link delay, node processing delay, which is up to 86^{20} states. Action is represented by the tuple of node which univocally determine the paths for the source-destination node pairs. As shown in Fig . 7, the action space for the node i to destination j is (i, k, j), (i, l, j) and (i, m, j). In addition, reward is represent by the delay from the source to the destination node.

In our experiment, we employed a neural network that has five fully connected hidden layers with a sigmoid activation function as well as a trained DQN on the gym platform [33] and Keras [34]. In addition, we devised 10 traffic intensity (TI) levels, ranging from 1 to 10, which represent the volumes of the network traffic at different times. The DRL agent was trained for 200K steps for each TI.

Experimental Results and Analysis The DRL learning process is demonstrated in Fig. 8. This relevant outcome is that the DRL performance increases with training steps and the DRL agent be convergence when the training step more than 200K. The reason is that the DRL learning is a process that the result is close to a near-optimal strategy by interacting with the environment. The second simulation results are demonstrated in Fig. 9. It can be seen that the average transmission time of the network increases with the increase of network traffic load. When the load level is low, with the increase of network load, the increase of average transmission time is stable. But when the network load continues to increase, the average transmission time increases sharply, due to the fact that the network capacity is close to saturation.

In our experiment, the benchmark algorithm is the shortest path algorithm. When the network load is low, there is no congestion in the network and the shortest path is the optimal path. So, the benchmark result is better. With the increase of network load, congestion occurs on the shortest

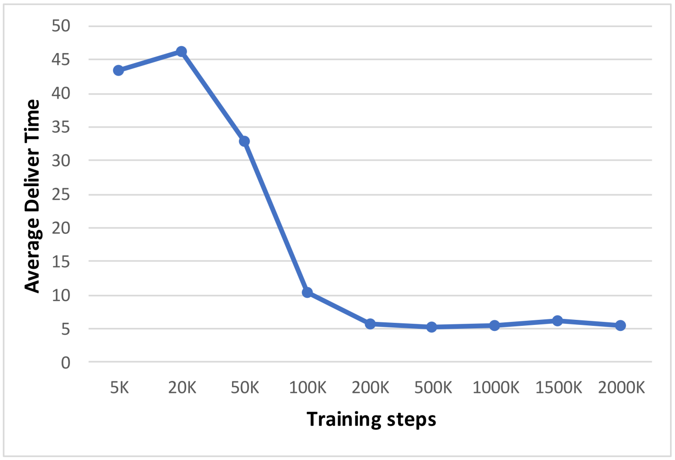


Fig. 8. The DRL learning process

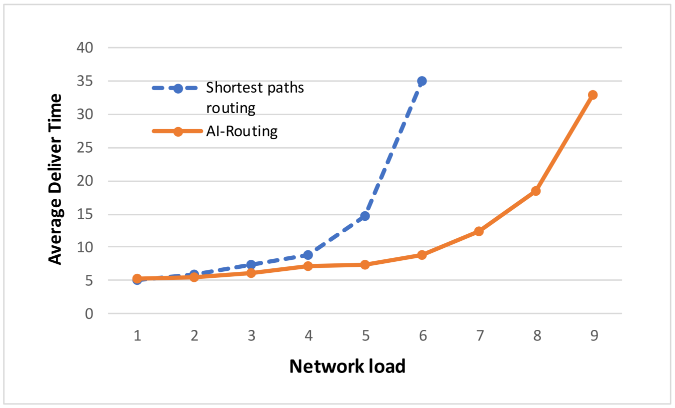


Fig. 9. The average delivery time over different network loads

path of the network, and the network agent will have to choose non-congested links for transmission. Thus, in this situation, the DRL performance performs much better than the benchmark.

6 Challenges and Discussions

In this paper, we proposed a new network paradigm named NetworkAI. We introduce this architecture to effectively solve the two main problems when apply ML in network, how to run ML on distributed network and how to generate complex strategies. Through the technologies of SDN, monitor technologies, and RL, the intelligent agent can automatically generate network strategies for network control and management. Moreover, through a use case, we have demonstrated that a fully automated DRL agent can provide routing policy that tends to minimize the network delay.

Compared with the traditional network architectures, we apply SDN and monitor technologies to implement a completely centralized view and control for distributed network systems and build a closed control loop for real-time upload of network state info and download of actions.

At the same time, DRL is introduced to effectively generate control policy in network systems. Benefit from the DL method with powerful function approximation and representation learning capabilities, the massive state space of the network can be effectively compressed. In addition, due to the black-box feature of RL approach, for different network decision tasks and optimization goals, only the action space and reward have to be designed. Furthermore, the near-optimal solution can be calculated in single step once the DRL trained. Compared with the traditional heuristic algorithms, this feature presents a huge advantage for real-time network applications.

The NetworkAI paradigm brings significant advantages to networking. However, it also meets some challenges that need to be addressed further.

6.1 Communication Overhead

The communication overhead for retrieving and issuing data is a serious problem in SDN architecture. While the convenience brought by centralized framework, it leads to too much interaction between centralized controller and distributed forwarding unit. The performance of NetworkAI will be degraded as a result of the rapid flow table update to all forwarding unit. To address it, NetworkAI can borrow some technologies from SDN. One possible solution is segment routing technology, which implements the source routing and tunnel method to effectively reduce the flow table update. Another way to alleviate this problem is to employ a cluster of controller to handle larger flow tables [35].

6.2 Training Cost

RL approach provides flexible tools to address network control problems. Nonetheless, RL method involve a large amount of training cost, especially in network scenario where exists mass applications and services. When new businesses or services appears, the agent requires a significant level of training cost which weaken the flexibility of NetworkAI. In this context, the NetworkAI paradigm requires mechanisms to reduce the training cost to satisfy mass of new network applications. A notable approach to address this problem is adding prior knowledge to accelerate the training process, such as transfer learning, imitation learning [36]. In this sense, these trick may reduce the training cost when new businesses or services appear and essentially in taking a step further in performance of NetworkAI.

6.3 Testbeds

To evaluate the performance of new network designs and algorithms, testbeds are more convincing for network experiments compared with simulators and emulation platforms, because testbeds can incorporate real traffic and real network facilities [37]. Building a complex experimental environment will be the most critical issue for applying AI in a network. In particular, due to the fact that the NetworkAI architecture is aimed at a complex, highly dynamic multi-application network environment, it is difficult to obtain convincing experiments through the network simulator. Therefore, in the immediate future, we plan to build a large-scale real NetworkAI testbed to expand our experiments.

7 Conclusions

In this paper, we introduced the concept of NetworkAI, a novel paradigm that combines SDN, INT, and DRL to automatically control and optimize a network. We also presented a QoS-Routing use case and preliminary experimental evidence with an aim to demonstrate the feasibility of the proposed paradigm. Finally, we discussed some important challenges that need to be addressed. We advocate that addressing such challenges requires a truly interdisciplinary effort between the research fields of artificial intelligence, network science, and computer networks.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant No. 61471056), ”the Fundamental Research Funds for the Central Universities” of China University of Petroleum (East China) (Grant No. 18CX02139A), the Shandong Provincial Natural Science Foundation, China (Grant No. ZR2014FQ018), the National Basic Research Program (973) of China (Grant No. 2012CB315801), and the China research project on key technology strategy of infrastructure security for information network development. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

References

[1] O. N. Foundation, “Software-defined networking: The new norm for networks,” 2012.

[2] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, “Are we ready for sdn? implementation challenges for software-defined networks,” *IEEE Communications Magazine*, vol. 51, no. 7, pp. 36–43, 2013.

[3] H. E. Egilmez, S. T. Dane, K. T. Bagci, and A. M. Tekalp, “Openqos: An openflow controller design for multimedia delivery with end-to-end quality of service over software-defined networks,” in *Signal & Information Processing Association Summit and Conference*, pp. 1–8, 2012.

[4] S. Yussof and O. H. See, “The effect of ga parameters on the performance of ga-based qos routing algorithm,” in *International Symposium on Information Technology*, pp. 1–7, 2008.

[5] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. V. D. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, and M. Lanctot, “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, p. 484, 2016.

[6] A. Mestres, A. Rodrigueznatal, J. Carner, P. Barletros, E. Alarcn, M. Sol, V. Munts, D. Meyer, S. Barkai, and M. J. Hibbett, “Knowledge-defined networking,” *Acm Sigcomm Computer Communication Review*, vol. 47, no. 3, pp. 2–10, 2016.

[7] H. Yao, C. Qiu, C. Fang, X. Chen, and F. R. Yu, “A novel framework of data-driven networking,” *IEEE Access*, vol. 4, no. 99, pp. 9066–9072, 2017.

[8] J. Jiang, V. Sekar, I. Stoica, and H. Zhang, “Unleashing the potential of data-driven networking,” 2017.

[9] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, L. J. Wobker, and B. Networks, “In-band network telemetry via programmable dataplanes,” 2015.

[10] D. Kreutz, F. M. V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proceedings of the IEEE*, vol. 103, no. 1, pp. 10–13, 2014.

[11] “Pnda.” http://www.pnda.io/.

[12] H. Mao, R. Netravali, and M. Alizadeh, “Neural adaptive video streaming with pensieve,” in *Conference of the ACM Special Interest Group on Data Communication*, pp. 197–210, 2017.

[13] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, “Resource management with deep reinforcement learning,” in *ACM Workshop on Hot Topics in Networks*, pp. 50–56, 2016.

[14] H. H. Liu, R. Viswanathan, M. Calder, A. Akella, R. Mahajan, J. Padhye, and M. Zhang, “Efficiently delivering online services over integrated infrastructure,” in *Usenix Conference on Networked Systems Design and Implementation*, pp. 77–90, 2016.

[15] M. Stemm, R. Katz, and S. Seshan, “A network measurement architecture for adaptive applications,” in *INFOCOM 2000. Nineteenth Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, pp. 285–294 vol.1, 2002.

[16] J. Jiang, R. Das, G. Ananthanarayanan, P. A. Chou, V. Padmanabhan, V. Sekar, E. Dominique, M. Goliszewski, D. Kukoleca, and R. Vafin, “Via: Improving internet telephony call quality using predictive relay selection,” in *Acm Sigcomm Conference*, pp. 286–299, 2016.

[17] A. Ganjam, F. Siddiqui, J. Zhan, X. Liu, I. Stoica, J. Jiang, V. Sekar, and H. Zhang, “C3: Internet-scale control plane for video quality optimization,” *IEEE Transactions on Visualization & Computer Graphics*, vol. 20, no. 7, p. 1035, 2015.

[18] J. Jiang, S. Sun, V. Sekar, and Z. Hui, “Pytheas: Enabling data-driven quality of experience optimization using group-based exploration-exploitation,” *Usenix Symposium on Networked Systems Design and Implementation*, pp. 393–406, 2017.

[19] J. A. Boyan and M. L. Littman, “Packet routing in dynamically changing networks: a reinforcement learning approach,” in *International Conference on Neural Information Processing Systems*, pp. 671–678, 1993.

[20] S. C. Lin, I. F. Akyildiz, P. Wang, and M. Luo, “Qos-aware adaptive routing in multi-layer hierarchical software defined networks: A reinforcement learning approach,” in *IEEE International Conference on Services Computing*, pp. 25–33, 2016.

[21] S. Haeri and L. Trajkovic, “Virtual network embedding via monte carlo tree search.,” *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–12, 2017.

[22] N. Mckeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “Openflow:enabling innovation in campus networks,” *Acm Sigcomm Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[23] P. Bosshart, M. Izzard, M. Izzard, M. Izzard, N. Mckeown, J. Rexford, T. Dan, T. Dan, A. Vahdat, and G. Varghese, “P4: programming protocol-independent packet processors,” *Acm Sigcomm Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2013.

[24] “barefootnetworks.” https://www.barefootnetworks.com/.

[25] A. W. Moore and D. Zuev, “Internet traffic classification using bayesian analysis techniques,” *Acm Sigmetrics Performance Evaluation Review*, vol. 33, no. 1, pp. 50–60, 2005.

[26] H. Zhang, L. Chen, B. Yi, K. Chen, Y. Geng, and Y. Geng, “Coda: Toward automatically identifying and scheduling coflows in the dark,” in *Conference on ACM SIGCOMM 2016 Conference*, pp. 160–173, 2016.

[27] T. T. T. Nguyen and G. Armitage, “A survey of techniques for internet traffic classification using machine learning,” *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, pp. 56–76, 2009.

[28] C. Li and C. Yang, “The research on traffic sign recognition based on deep learning,” in *International Symposium on Communications and Information Technologies*, pp. 156–161, 2016.

[29] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, “A brief survey of deep reinforcement learning,” 2017.

[30] J. Schmidhuber, “Deep learning in neural networks: An overview.,” *Neural Networks the Official Journal of the International Neural Network Society*, vol. 61, p. 85, 2015.

[31] M. Karakus and A. Durresi, “Quality of service (qos) in software defined networking (sdn): A survey,” *Journal of Network & Computer Applications*, vol. 80, pp. 200–218, 2017.

[32] S. Levine, S. Levine, S. Levine, and S. Levine, “Continuous deep q-learning with model-based acceleration,” in *International Conference on International Conference on Machine Learning*, pp. 2829–2838, 2016.

[33] “gym.” https://gym.openai.com/.

[34] “keras.” https://keras.io/.

[35] L. Cui, F. R. Yu, and Q. Yan, “When big data meets software-defined networking: Sdn for big data and big data for sdn,” vol. 30, pp. 58–65, 2016.

[36] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on Knowledge & Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

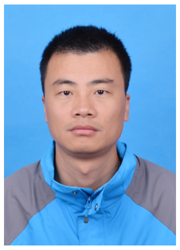
[37] T. Huang, F. R. Yu, C. Zhang, J. Liu, J. Zhang, and J. Liu, “A survey on large-scale software defined networking (sdn) testbed-

s: Approaches and challenges," *IEEE Communications Surveys & Tutorials*, vol. PP, no. 99, pp. 1–1, 2017.

Haipeng Yao is currently an Associate Professor with the School of Information and Communication Engineering, and also with the State Key Laboratory of Networking and Switching Technology in Beijing University of Posts and Telecommunications. His main research interests include future network architecture, artificial intelligence for networking, the architecture and next generation mobile communication systems.

Tianle Mai is currently a graduate student at Beijing University of Technology. He received his bachelors degree in the Beijing University of Posts and Telecommunications in 2016. His main research interests are in the areas of artificial Intelligence, and future internet architecture.

Xiaobin Xu is a Lecturer at Beijing University of Technology. He obtained his PhD at Beijing University of Posts and Telecommunications in 2014. He joined Beijing University of Technology as a Lecturer in 2017 and works at the Beijing Advanced Innovation Center for Future Internet Technology in 2017. His current research areas include data prediction algorithms for wireless sensor networks, routing algorithms for space-terrestrial integrated networks.



Peiying Zhang is currently an Associate Professor with the college of computer and communication engineering from China University of Petroleum (East China). He is a PhD candidate in Information and Communication Engineering, from the State Key Laboratory of Networking and Switching Technology in Beijing University of Posts and Telecommunications. His research interests include deep reinforcement learning, artificial intelligence for networking, network virtualization, and future Internet architecture.

Maozhen Li is currently a Professor in the Department of Electronic and Computer Engineering, Brunel University London. His main research interests include high performance computing, big data analytics, and knowledge and data engineering. He is a Fellow of both BCS and IET.

Yunjie Liu received his B.S degree in technical physics from Peking University, beijing, China, in 1968. He is currently an Academician of China Academy of Engineering, the Chief of the science and technology committee of China Unicom, and the Dean of the School of Information and Communications at Beijing University of Posts and Telecommunications. His current research interests include next generation networks, network architecture and management.