AI Routers & Network Mind: A Hybrid Machine Learning Paradigm for Packet Routing

©ISTOCKPHOTO.COM/METAMORWORKS

Haipeng Yao and Tianle Mai
State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, CHINA

Chunxiao Jiang and Linling Kuang
Tsinghua Space Center, Tsinghua University, Beijing, CHINA

Song Guo
Department of Computing, Hong Kong Polytechnic University, HONG KONG SAR

Digital Object Identifier 10.1109/MCI.2019.2937609
Date of current version: 14 October 2019

Abstract—With the increasing complexity of network topologies and architectures, adding intelligence to the network control plane through Artificial Intelligence and Machine Learning (AI&ML) is becoming a trend in network development. For large-scale geo-distributed systems, determining how to appropriately introduce intelligence in networking is the key to high-efficiency operation. In this treatise, we explore two deployment paradigms (centralized vs. distributed) for AI-based networking. To achieve the best results, we propose a hybrid ML paradigm that combines a distributed intelligence, based on units called "AI routers," with a centralized intelligence, called the "network mind", to support different network services. In the proposed paradigm, we deploy centralized AI control for connection-oriented tunneling-based routing protocols (such

Corresponding Author: Haipeng Yao (yaohaipeng@bupt.edu.cn)

as multiprotocol label switching and segment routing) to guarantee a high QoS, whereas for hop-by-hop IP routing, we shift the intelligent control responsibility to each AI router to ease the overhead imposed by centralized control and use the network mind to improve the global convergence.

1. Introduction

Recently, networks throughout the world are undergoing profound restructuring and transformation with the development of Software-Defined Networking (SDN), Network Function Virtualization (NFV), and 5th-generation wireless systems (5G). The new networking paradigms are eroding the dominance of traditional ossified architectures and reducing dependence on proprietary hardware. However, the corresponding improvements in network flexibility and scalability are also presenting unprecedented challenges for network management. In particular, with the emergence and development of new services and scenarios (such as the IoT paradigm and AR/VR), network scales and traffic volumes are exhibiting explosive growth, and the QoS/QoE requirements are becoming increasingly demanding. This ever-increasing network complexity makes effective network control extremely difficult. In particular, current control strategies largely rely on manual processes, which have poor scalability and robustness for the control of complex systems. Therefore, there is an urgent need for more powerful methods of addressing the challenges faced in networking.

In recent years, with the great success of machine learning, applications of Artificial Intelligence and Machine Learning (AI&ML) in networking have received considerable attention [1], [2]. Compared to meticulously manually designed (white-box) strategies, AI&ML (black-box) techniques offer enormous advantages in networking systems. For example, AI&ML provides a generalized model and uniform learning method without prespecified processes for various network scenarios [3]. In addition, such techniques can effectively handle complex problems and high-dimensional situations; indeed, AI&ML methods have already achieved remarkable success in many complex system control domains, including computer games and robotic control [4]. In addition to the enormous advantages of AI&ML for networking, the development of new network techniques is also providing fertile ground for AI&ML deployment. For example, In-band Network Telemetry (INT) enabled end-to-end network visualization at the millisecond scale in 2015, and Cisco published a big data analytics platform for networking, PNDA, in 2017. Therefore, the growing trend of applying AI&ML in networking is being driven by both task requirements (the increasing complexity of networks and increasingly demanding QoS/QoE requirements) and technological developments (new network monitoring technologies and big data analysis techniques) [5].

The AI&ML-driven networking paradigm was first put forward by D. Clark *et al.* in [6], where "A Knowledge Plane for the Internet" for network operations using AI&ML was proposed. However, learning based on distributed nodes with only a partial perspective on the network is a complex task, especially with the goal of global optimization. This fundamental defect has resulted in the stagnation of knowledge plane development. In recent years, benefiting from developments in SDN technology, a centralized intelligent network architecture has become a feasible solution. In [7], Mestres *et al.* proposed a centralized intelligent paradigm for AI-driven networking called Knowledge-Defined Networking (KDN), in which control strategies are generated in a centralized knowledge plane enabled by ML algorithms. However, as the network scale expands, the centralized paradigm incurs excessive overhead in terms of both communication and computation, especially for real-time network control tasks (such as traffic routing). This overhead will certainly introduce large delays that will further degrade the performance of AI-based algorithms.

As discussed above, both the distributed and centralized paradigms are imperfect and have fundamental flaws. Therefore, in this paper, we propose a hybrid AI-driven paradigm for traffic routing control in which we combine a distributed intelligence, based on units called "AI routers," with a centralized intelligence platform, called the "network mind," to support different network services. Specifically, we separately consider centralized intelligent control for tunneling-based routing and distributed intelligence for hop-by-hop routing. In addition, we apply two kinds of ML algorithms to optimize traffic routing control strategies to satisfy network service requirements, such as congestion control and QoS/QoE guarantees.

The main contributions of this paper are briefly summarized below.

- We propose a hybrid ML paradigm for packet routing, in which we combine a distributed intelligence based on AI routers with a centralized intelligence platform called the network mind.
- For tunneling-based routing (with a high-QoS guarantee), we discuss the feasibility and superiority of centralized optimization and deploy a deep-reinforcement-learning-based routing strategy in the network mind for route optimization.
- For hop-by-hop routing, we shift the responsibility for intelligent control to each AI router to ease the overhead imposed by centralized control and use the network mind to improve the global convergence.

The rest of this paper is organized as follows. In Section 2, we review the related work on AI-driven network traffic routing. In Section 3, we discuss the placement of the intelligent control plane and propose a hybrid architecture for various tasks. In Section 4, we propose a centralized AI-based routing algorithm for high-QoS network services. In Section 5, we design a hybrid routing architecture to address the distributed congestion control problem. In Section 6, several challenges and open issues are presented.

2. Related Work

Although AI-driven networking is currently a research area of considerable interest, the idea of applying ML in traffic routing

can be traced back to the 1990s. In this section, we review the related work on AI-driven network routing algorithms.

In recent years, with the great success of machine learning, applications of Artificial Intelligence and Machine Learning (AI&ML) in networking have received considerable attention.

2.1. Decentralized Routing

2.1.1. Single-Agent Reinforcement Learning

In [8], Boyan *et al.* proposed the Q-routing algorithm for optimizing packet routing control. In the Q-routing algorithm, each router updates its policy according to its Q-function based on local information and communication. The experiments showed that Q-routing offered more efficient performance than the nonadaptive shortest path algorithm, especially under a high workload. In [9], Choi *et al.* proposed a memory-based Q-learning algorithm called predictive Q-routing to increase the learning rate and convergence speed by retaining past experiences. In addition, in [10], Kumar *et al.* proposed dual reinforcement Q-routing (DRQ-routing), which uses information gained through backward and forward exploration to accelerate the convergence speed. In [11], [12], Reinforcement Learning (RL) was successfully applied in wireless sensor network routing, where the sensors and sink nodes could self-adapt to the network environment. However, in a multiagent system, single-agent RL suffers from severe non-convergence. Instead, applying multiagent RL to improve the cooperation among network nodes is more feasible, and there have been a series of works on ML-driven routing based on multiagent RL.

2.1.2. Multiagent Reinforcement Learning

In [13], [14], Stone *et al.* proposed the Team-Partitioned Opaque-Transition RL (TPOT-RL) routing algorithm, which allows a team of network nodes working together toward a global goal to learn how to perform a collaborative task. In [15], Wolpert *et al.* designed a sparse reinforcement learning algorithm named the Collective Intelligence (COIN) algorithm, in which a global function is applied to modify the behavior of each network agent. In contrast, the author of [16] proposed a Collaborative RL (CRL)-based routing algorithm with no single global state. The CRL approach was also successfully applied for delay-tolerant network routing in [17]. However, in an inherently distributed system, state synchronization among all routers is extremely difficult, especially with increasing network size, speed, and load. With the development of SDN technology, centralized AI-driven routing strategies have received considerable attention.

2.2. Centralized Routing

In [18], Stampa *et al.* proposed a deep RL (DRL) algorithm for optimizing routing in a centralized knowledge plane. Benefiting from the global control perspective, the experimental results showed very promising performance. In [19], Lin *et al.* applied the SARSA algorithm to achieve QoS-aware adaptive routing in multilayer hierarchical software-defined networks. For each flow, the controller updated the optimal routing strategy based on the QoS requirements and issued the forwarding table to each node along the forwarding path. In [20], Wang *et al.* proposed a RL-based routing algorithm for Wireless Sensor Networks (WSNs) named AdaR. In AdaR, Least-Squares Policy Iteration (LSPI) is implemented to achieve the correct tradeoff among multiple optimization goals, such as the routing path length, load balance, and retransmission rate. However, the overhead incurred for centralized AI control is high.

3. AI-Driven Network Routing

In this section, we first propose a three-layer logical functionality architecture for AI-driven networking. Then, we discuss the problem of how far away the intelligent control plane can be located from the forwarding plane ("centralized" or "distributed").

3.1. Closed-Loop Control Paradigm

In a traditional network, the network layer functionality can be divided into the forwarding plane and the control plane. However, with the introduction of AI&ML, this two-layer architecture cannot effectively describe the logic of intelligent system operation. In this paper, inspired by the closed-loop mechanism of the learning process of the human brain ("observation - judgment - action - learning"), we split the functionality of AI-based networking into three layers to

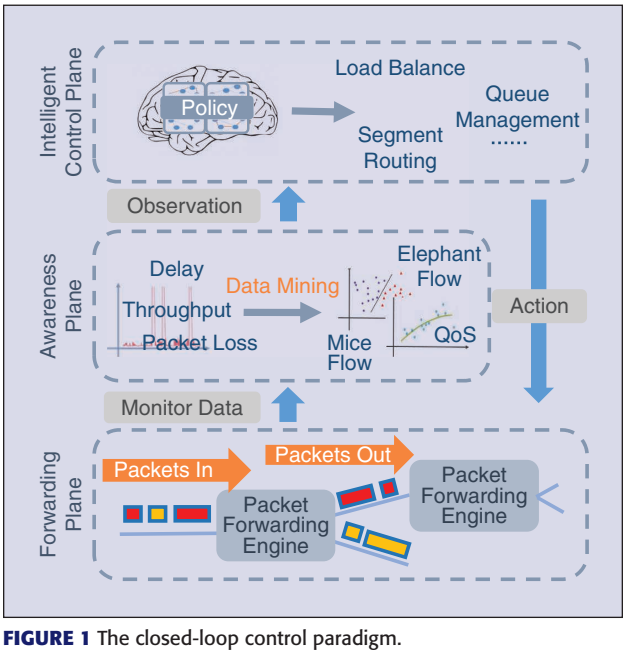

FIGURE 1 The closed-loop control paradigm.

In an inherently distributed system, state synchronization among all routers is extremely difficult, especially with increasing network size, speed, and load. With the development of SDN technology, centralized AI-driven routing strategies have received considerable attention.

construct a closed-loop network control paradigm. As illustrated in Fig. 1, our paradigm consists of three layers, called the forwarding plane, the awareness plane, and the intelligent control plane.

The forwarding plane is responsible for forwarding data packets from one interface to another in distributed network equipment. Its operation logic relies completely on the forwarding table and configuration instructions issued by the control plane.

The purpose of the awareness plane is to monitor the network status and upload the results to the control plane. Network monitoring and awareness are prerequisites for ML-based control and optimization. Therefore, we abstract this new layer called the awareness plane for the collection and processing of monitoring data (for tasks such as network device monitoring and network traffic identification) to provide network status information.

The intelligent control plane is responsible for feeding control decisions to the forwarding plane. The AI&ML-based algorithms are deployed in this plane to transform the current and historical operation data into control policies.

These three abstract planes together constitute a closed-loop framework for AI&ML deployment in networking. In analogy to the human learning process, the forwarding plane acts as the "subject of action," the awareness plane acts as the "subject of observation", and the intelligent control plane acts as the "subject of learning/judgment." Based on these three planes for closed-loop control, an AI&ML agent can continuously learn and optimize network control and management strategies by interacting with the underlying network.

3.2. Centralized vs. Distributed

As described above, a three-tier logical architecture is proposed. However, when this abstract logical concept is deployed in a real-world network, the placement of the intelligent control plane (centralized or distributed) is critical to the efficient operation of AI-driven networking.

How far away the control plane can be located from the data plane has long been a controversial topic. In traditional distributed networking equipment, the control plane and the forwarding plane are closely coupled. Each node has only a partial view of, and partial control over, the complete network. When AI&ML-based algorithms are applied in such a network, the learning process will suffer from severe non-convergence, particularly when a global optimum is sought. In contrast, in an SDN architecture, the control plane is decoupled from the network hardware and acts as a centralized plane in which an AI agent can interact with the whole network to generate an optimal strategy. However, while the advantages of centralized optimization are clear, the overhead of closed-loop control implemented through a centralized AI is high. This overhead includes not only the communication overhead for receiving and sending a large amount of data but also the computational overhead on the AI agent side for training and execution. In the centralized paradigm, all routers need to be programmed to build a single flow-forwarding path. In addition, every time the network status changes, the controller needs to recompute the forwarding logic. For large-scale networks with ultrahigh dynamics (on the millisecond scale), this excessive communication pressure and high computational burden are unacceptable.

As discussed above, the centralized and distributed paradigms, as the two ends of a spectrum, are both imperfect and have corresponding advantages and disadvantages. A distributed architecture carries the risk of non-convergence of the learning process, but it offers faster forwarding and processing speeds for each packet. In contrast, completely centralized learning is advantageous for global optimization but may incur excessive overheads in terms of communication and computation. Therefore, from our perspective, the centralized and distributed approaches should be treated as complementary rather than mutually exclusive. In this paper, as shown in Fig. 2, we propose a hybrid AI-driven control architecture that combines a "network mind" (centralized intelligence) with "AI routers" (distributed intelligence) to support different network services.

Before we detail the operations in our AI-based routing paradigm, let us start by reviewing the current state of development of routing protocols. Early on, the IP protocol won the battle between connectionless and connection-oriented routing and between source routing and distributed routing. As shown in Fig. 3, in the IP protocol, each router establishes a routing table based on its local information and communications. This routing table contains the next-hop node and a cost metric for each destination. Based on this hop-by-hop forwarding paradigm, a data packet needs to carry only its destination address in its header, which is beneficial for network scalability and robustness. However, because of the connectionless and distributed characteristics of the IP protocol, traditional IP routing provides poor support for traffic engineering and QoS guarantees. To support high-QoS (high-bandwidth, delay-sensitive) services, connection-oriented and source routing mechanisms have begun to receive attention once again. For example, as shown in Fig. 4, Multiprotocol Label Switching (MPLS) uses connection-oriented label switching and explicit paths (source routing) to establish temporary network tunnels between senders and receivers. This predetermined temporary tunnel routing strategy provides an easier and more efficient QoS-guarantee mechanism for service providers. However, full-mesh network tunneling is

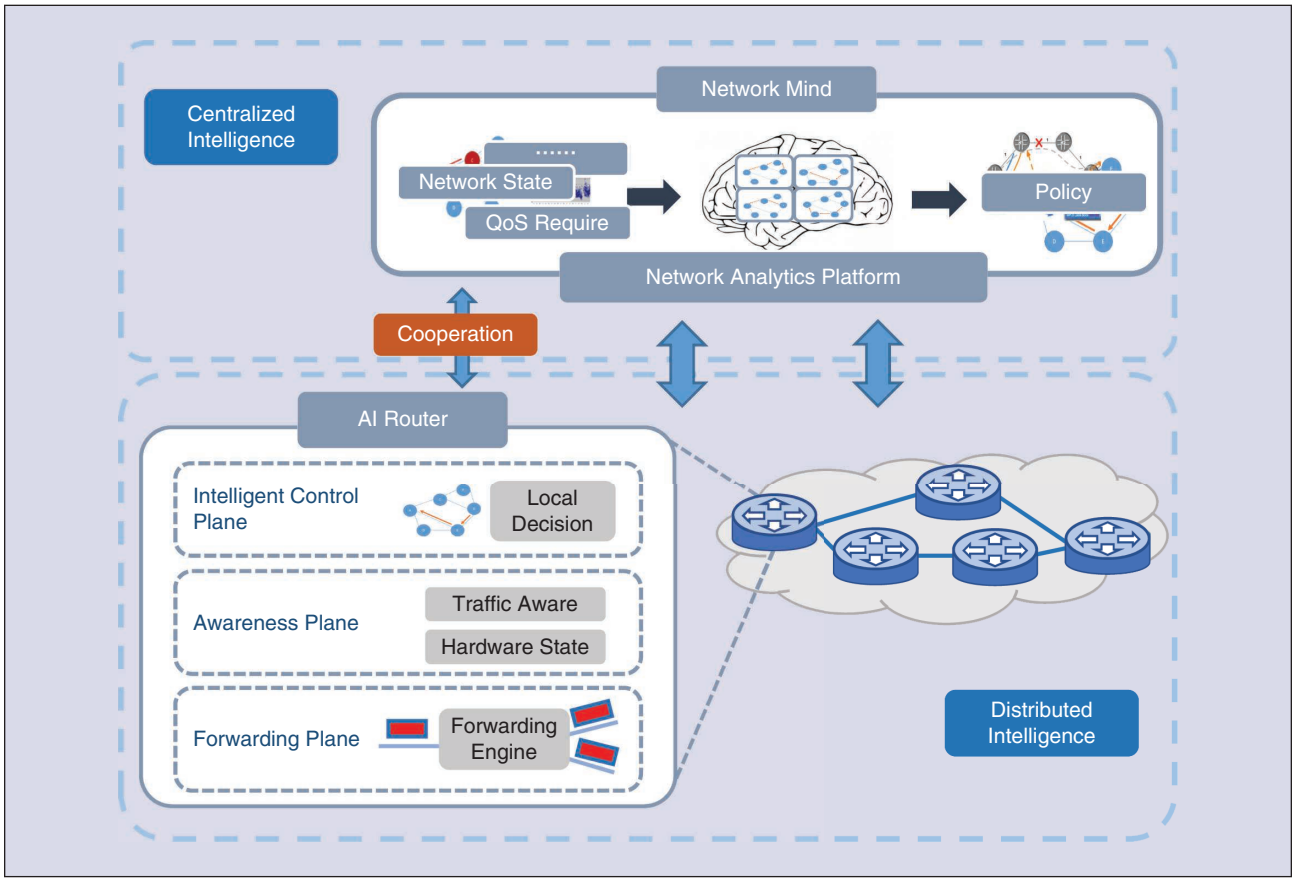

FIGURE 2 The proposed hybrid architecture.

extremely operationally complex and offers limited scalability, usually without any gain.

Thus, connection-oriented tunneling-based protocols for reliable service delivery and connectionless distributed protocols coexist in current networks. For these two kinds of routing mechanisms, intelligent control should be deployed in different ways. A tunneling-based protocol is essentially a centralized routing protocol, in which the source node maintains an understanding of the state of the whole network and calculates an appropriate forwarding path. Therefore, in our paradigm, we place the responsibility of intelligent control for routing optimization with the network mind. In contrast, for hop-by-hop routing, we shift the responsibility for intelligent control to the AI routers and use the network mind to facilitate cooperation among multiple AI routers. We will discuss these issues in detail in the following sections.

4. Network Mind

High-QoS delivery is crucial to the success of current business models for many network applications, such as online gaming (which is sensitive to delays) and AR/VR (which requires high bandwidths). Methods of guaranteeing QoS over tunneling-based protocols (such as MPLS and segment routing) have been discussed and developed for more than a decade. However, traditional distributed signaling solutions based on RSVP-TE/

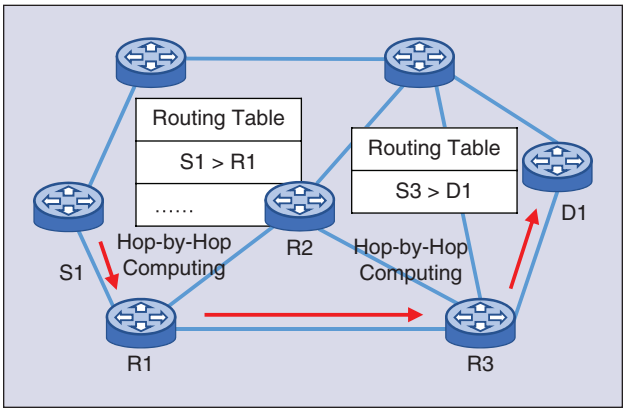

FIGURE 3 Hop-by-hop protocol.

MPLS-TE will result in a lack of coordination and competition for network resources, which in turn will lead to a lack of optimality, a lack of predictability and slow convergence. Therefore, in our paradigm, to guarantee a high QoS for services that require it, the AI-based intelligent control of tunneling-based routing is deployed in a centralized way.

As shown in Fig. 5, the proposed network mind is responsible for centralized intelligent traffic control and optimization. The network mind accesses the fine-grained network state through an upload link and issues actions via a download link.

Thus, connection-oriented tunneling-based protocols for reliable service delivery and connectionless distributed protocols coexist in current networks. For these two kinds of routing mechanisms, intelligent control should be deployed in different ways.

The upload link relies on a network monitoring protocol, such as INT, Kafka, or IPFIX, to gather device states, traffic characteristics, configuration data, and service-level information; the download link relies on a standard southbound interface, such as OpenFlow or P4, to facilitate efficient control over the network. The upload and download links constitute an interaction framework that provides the network mind with a global perspective and global control capabilities, and the current and historical data provided by the closed-loop operations are fed to AI&ML algorithms for generating and learning knowledge.

However, learning control policies from complex and high-dimensional system states is a challenging task. In particular, as the network scale and granularity of monitoring increase, a dimensional explosion of the network state will occur. Recently, the success achieved in applying RL to many challenging decision-making domains, such as GO and video games, suggests that this idea may not be impossible to realize [21]. RL provides a paradigm of learning through trial and error to generate an optimal behavior policy. In particular, benefiting from the representation learning abilities of deep learning, DRL can be applied to directly construct and learn knowledge from raw high-dimensional data [22]. Therefore, in this paper, we apply DRL for effective routing policy generation.

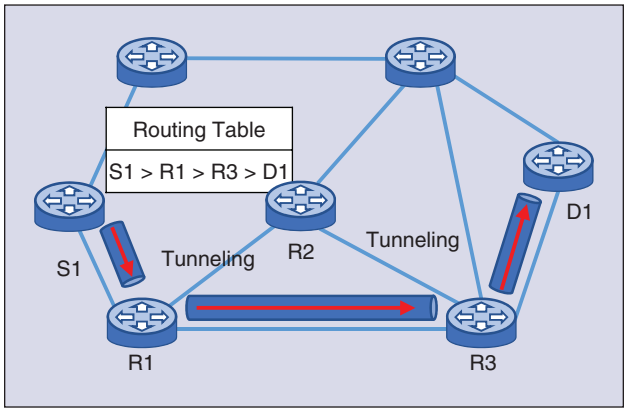

FIGURE 4 Tunneling-based protocol.

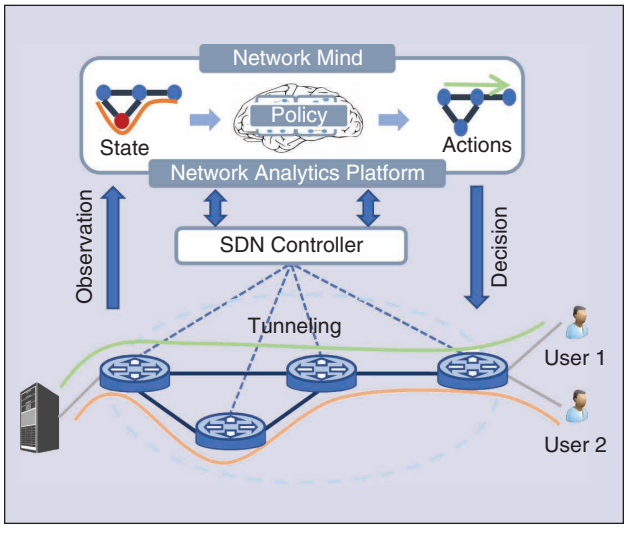

FIGURE 5 The centralized intelligent control scheme.

4.1. Modeling and Formulation

In the context of RL, the Markov Decision Process (MDP) is a useful mathematical framework for tackling related problems. The MDP is an abstract framing of the problem of learning via interaction to achieve a certain control and optimization goal. In our scenario, the network mind and the underlying network environment construct an MDP environment and continually interact to generate control strategies. In each step, the centralized AI agent observes the network state s_t from the underlying network and makes a routing decision in accordance with the current strategy $\pi(a|s)$. Following this decision, the controller issues the corresponding policy to the network nodes along the forwarding path. Then, the network transitions into the next state s_{t+1}, and the AI agent obtains an immediate reward R from the environment. Specifically, the network state can be represented by network device information and traffic characteristic information, and the actions can be represented by the forwarding path. The reward function evaluates the effectiveness of the actions taken with respect to the optimization target (such as a delay requirement or throughput guarantee).

In this paper, we apply the Deep Deterministic Policy Gradient (DDPG) approach for policy generation [23]. A DDPG agent consists of two components: the deterministic policy network (actor) $\mu(s|\theta^{\mu})$ and the Q-network (critic) $Q(s,a|\theta^{Q})$. The actor attempts to improve the current policy $\mu(s|\theta^{\mu})$ based on the policy gradient, and the critic evaluates the quality of the current policy with the parameters θ^{μ}. The DDPG agent implements an iterative policy mechanism that alternates between policy improvement (actor) and policy evaluation (critic).

During the learning process, the DDPG agent first selects an action based on the current strategy:

$$a_t = \mu(s_t|\theta^{\mu}) + \mathcal{N}_t. \tag{1}$$

Then, the agent executes the action a_t and observes the reward r_t and the new state s_{t+1} of the underlying network. During training, a replay memory R is used to eliminate the temporal correlations between data. The transition data (s_t, a_t, r_t, s_{t+1}) for the current step are stored in R, and then, a random minibatch

of N transitions (s_i, a_i, r_i, s_{i+1}) is sampled from the replay memory to update the critic network by minimizing the following loss based on the ADAM optimizer [24]:

$$L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i \mid \theta^Q))^2, \tag{2}$$

where we set y_i to

$$y_i = r_i + \gamma Q(s_{i+1}, \mu(s_{i+1} \mid \theta^\mu) \mid \theta^Q). \tag{3}$$

Furthermore, the actor policy is updated using the sampled policy gradient with the aim of maximizing the discounted cumulative reward, which can be described as follows:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a \mid \theta^Q)\big|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s \mid \theta^\mu)\big|_{s_i}. \tag{4}$$

Compared to traditional heuristic-based algorithms, DRL possesses several advantages for networking control. First, due to the strong generalization ability of neural networks, DRL can generate knowledge directly from nonlinear, complex, high-dimensional network systems without requiring assumptions and simplifications. Second, as a black-box optimization approach, DRL allows reward functions to be redesigned to adapt to different network targets without modifying the algorithm model. Third, a DRL agent, once trained, can calculate a near-optimal forward path in one step. In contrast, heuristic-based algorithms require a large number of steps to converge to a new optimal solution whenever the network state changes. Especially in large-scale, highly dynamic networks, the resulting computational complexity will lead to serious non-convergence.

4.2. Simulation

We present simulation results to demonstrate the feasibility and correctness of our algorithm. In our experiment, we simulated a network with 12 nodes and 20 full-duplex links. To evaluate the algorithm performance under various network congestion conditions, we set 10 different levels of traffic load intensity. The traffic was generated subject to a Poisson distribution, and we set different traffic load intensities by means of the parameter λ. We used a neural network with two connected hidden layers, where the first layer had 50 hidden units and the second had 40 hidden units. In our experiment, we applied OMNet++ for network traffic simulation and Keras and TensorFlow for DDPG agent construction.

In our experiment, the network state was represented by the transmission delay and node processing delay, each action was represented by the set of nodes defining the forwarding path from the source node to the destination node, and the reward was represented by the total delay for forwarding from the source to the destination.

The learning process of the DDPG agent is illustrated in Fig. 6. With an increasing number of training steps, the DDPG agent gradually converges to the optimal strategy. In addition, in our experiment, we compared our algorithm with the shortest path routing algorithm. As shown in Fig. 7, when the traffic load is low, congestion does not occur in the network. Therefore, the shortest path routing performs as well as the AI-based algorithm. However, with increasing load intensity, the network congestion becomes severe on the shortest forwarding path, and the AI-based routing achieves better performance than the shortest path routing. Therefore, we can conclude that the AI-based routing is effective, especially in the presence of network congestion.

5. AI Routers & Network Mind

Although tunneling-based protocols have advantages in terms of traffic engineering and QoS guarantees, full-mesh tunneling across the whole network will result in operational complexity and limited scalability. Therefore, as shown in Fig. 8, we propose a hybrid AI-based hop-by-hop routing paradigm. In our architecture, for easing the overhead imposed by centralized

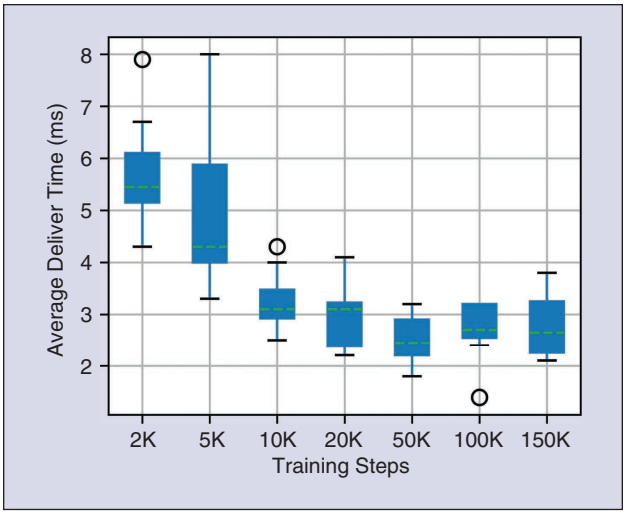

FIGURE 6 The learning process of the DDPG agent.

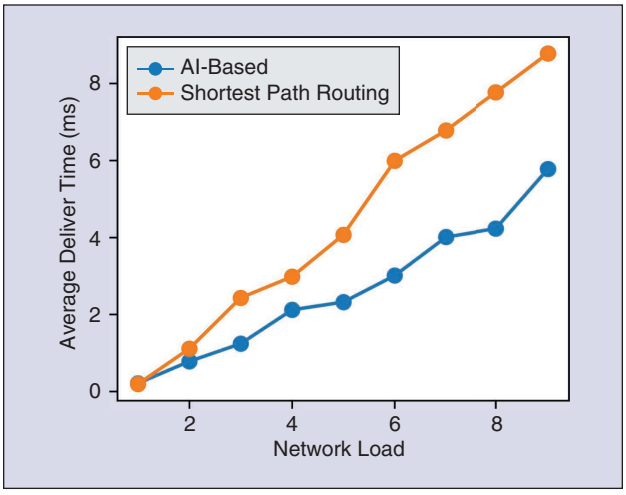

FIGURE 7 The average delivery time versus different network loads.

In our architecture, for easing the overhead imposed by centralized control, we shift the responsibility for intelligent control to the AI routers and use the network mind to improve the global convergence.

control, we shift the responsibility for intelligent control to the AI routers and use the network mind to improve the global convergence. In this section, we will detail the operations of this architecture.

With the intelligent control responsibility shifted to each router, each router acts as an independent intelligent agent and the distributed AI agents constitute a Multi-Agent System (MAS). Each AI agent attempts to optimize its local policy by interacting with its uncertain environment with the aim of maximizing the expected cumulative reward. Compared to a single-agent system, in which the state transitions of the environment depend solely on the actions of the single agent, the state transitions of an MAS are subject to the joint actions of all agents. In other words, although each AI router makes decisions based on its own local network information, these individual decisions affect each agent's transitions and the global reward.

To improve the global utility of this MAS, the ability to share experiences among the AI routers is significant. However, the question of how to achieve such information sharing in this geo-distributed system is a key problem for high-efficiency operation. In our architecture, the centralized network mind is introduced to serves as a point of global knowledge convergence for experience sharing. The centralized network mind can access global network information via the network monitoring system and share knowledge via the download link. This centralized architecture improves the efficiency of knowledge sharing compared to that in a peer-to-peer architecture. In addition, the question of which information should be transferred is another important factor to consider; the more information is transferred, the faster the convergence speed will be, but more communication overhead will also be incurred. In this paper, we use a "difference reward" as a modified reward signal to improve the collective behavior of the AI routers, as we will describe in detail below.

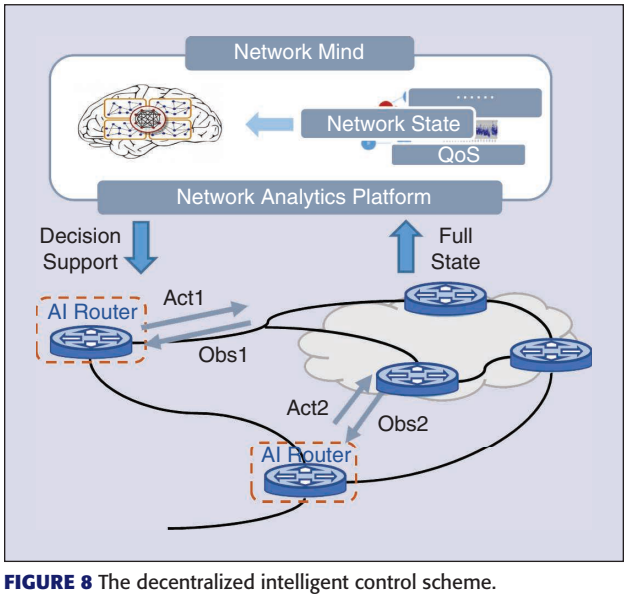

FIGURE 8 The decentralized intelligent control scheme.

5.1. Modeling and Formulation

Coordination among multiple agents can be formulated as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP). This Dec-POMDP can be described as a 5-tuple $< I, S, A, O, R >$, where I is the set of agents, S is the set of states, A is the set of actions, O is the set of local observations of each agent, and R is the set of rewards. In our scenario, in each step, each AI router takes an action a_t in accordance with its local observation o_t and current policy $\pi(a_t \mid o_t)$. Then, an immediate local reward L is obtained from the network environment, and the network state s_t transitions to the new state s_{t+1}.

In the MAS, the AI routers need to both cooperate with each other and compete with each other for the limited network resources. In this paper, we define that n resources (such as bandwidth, cache, and computation power) exist in each router. For router i, its observation o_i can be represented by $o_i = (< \omega_1, Cap_1, Csu_{1,t} >, \ldots, < \omega_k, Cap_k, Csu_{k,t} >, \ldots)$, where ω_k is the weight of resource k, Cap_k is the capacity of that resource, and $Csu_{k,t}$ is the amount of that resource consumed [25]. The action a_i is represented by the next-hop router, and the immediate local reward L is described as follows:

$$L(o_i, t) = f(o_i) = \sum_{k \in n} \omega_k e^{\frac{-Cap_k}{Csu_{k,t}}}. \tag{5}$$

However, with the objective of achieving the maximum cumulative reward, this local reward signal encourages only selfish behavior. Therefore, to facilitate cooperation among the AI routers, we implement a difference reward to modify the reward signal by removing much of the noise introduced by other routers. The difference reward is defined as follows:

$$D_i(s, a) = G(s, a) - G(s, a_{-i}). \tag{6}$$

Here, $G(s, a)$ is the global reward, which reflects the global utility of the whole system based on the joint actions executed by the multiple AI routers. The global reward is defined as the sum of all the local rewards:

$$G(t) = \sum_{o_i \in O} L(o_i, t). \tag{7}$$

Based on this, the update of the Q-value in AI router i can be rewritten as follows:

$$Q_i(o_t, a_t) \leftarrow D_i(s, a) + \lambda Q_i(o_{t+1}, a_{t+1}). \quad (8)$$

Based on the difference reward single, the centralized network mind can continuously revise the strategy of each router. The underlying distributed intelligence can be trusted to adapt correctly to changes in the network state, thereby reducing the need for reaction, recomputation and updating of the centralized AI platform.

Based on the difference reward single, the centralized network mind can continuously revise the strategy of each router. The underlying distributed intelligence can be trusted to adapt correctly to changes in the network state, thereby reducing the need for reaction, recomputation and updating of the centralized AI platform.

5.2. Simulation

In this section, we present simulation results to demonstrate the feasibility and performance of our architecture and algorithm. In our experiment, we focused on the congestion control problem for the distributed routing paradigm, which is difficult for traditional routing algorithms to address.

Our experimental environment was developed based on [26]; we simulated a network with 4 nodes and 6 unidirectional links and generated 400 data packets to be routed through the network. For simplicity, all these packets started at the same source node and were sent to the same destination node. Each packet was routed in a distributed manner by the AI routers.

As shown in Fig. 9, we compared our algorithm with a deterministic routing strategy and a single-agent RL algorithm. For the deterministic routing strategy, all data packets were routed along the same path. This strategy cannot respond to the network state in a timely manner; thus, it will lead to serious congestion problems and achieve an extremely low global utility. In contrast, RL can dynamically adapt to the congestion state of the network. However, due to the nonstationary environment of the MAS, the learning process for single-agent RL suffers from severe non-convergence, also resulting in a relatively low global score, as shown in Fig. 9. By contrast, in our architecture, a difference reward is introduced to modify the reward signal to enhance the collective behavior of the AI routers, thereby improving the global utility of the whole system.

6. Challenges and Open Issues

AI&ML-driven networking control is a promising paradigm for future networks, but many challenges still remain, and much more work needs to be done. In this section, we will discuss the major challenges and open issues regarding AI&ML-driven networking.

6.1. New Hardware Architectures

Every innovation with regard to upper-level services is based on significant advances in the performance of the underlying hardware, such as the Central Processing Unit (CPU) for general-purpose computations, the Digital Signal Processor (DSP) for a communication system, and the Graphics Processing Unit (GPU) for image processing. Similarly, to meet the requirements of the AI-driven networking age, there is an urgent need for a specific AI networking processor [27].

Current networks generate millions of different types of flows every millisecond. Running AI algorithms on such massive volumes of data is extremely challenging. The computing power of current routers is far from being able to satisfy the requirements for AI&ML deployment. Recently, as highly parallel, multicore, multithreaded processors, GPU and Tensor Processing Unit (TPU) chips have become a cornerstone of the AI age. Some studies have already shown that a GPU can offer improved packet processing capabilities [28]. However, due to the need for high-speed processing of massive amounts of data (more than 10 Gb/s) and the stringent response delay requirements (less than 1 ms) for future networks, there is still a large gap between universal AI processing chips and their actual deployment prospects in the networking field.

6.2. Advanced Software Systems

Currently, the handling of network data is posing challenges typical of big data; recent years have seen a 3-fold increase in total IP traffic and a >60% increase in the number of devices deployed and the amount of telemetry data streamed in near

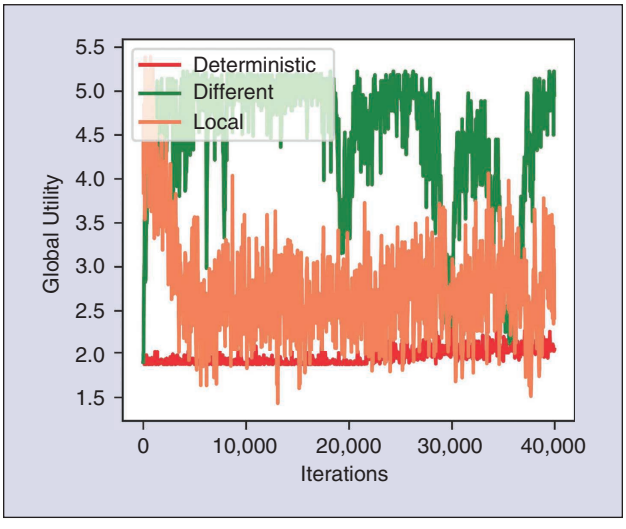

FIGURE 9 The global utility of the whole system.

real time. Meanwhile, the geo-distributed nature of networking is further increasing the difficulty of the widespread deployment of platforms for network data analytics. For example, challenges arise in determining how to aggregate data such as log data, metric data, and network telemetry data; how to scale up to the consumption of millions of flows per millisecond; and how to efficiently share knowledge among distributed network nodes. The current end-to-end solutions, which combine multiple technologies such as Apache Spark and Hadoop MapReduce, are extremely complex and time-consuming. Therefore, a powerful, scalable, big data analytics platform for networks and network services is needed. [27]

In addition, software libraries for ML networking tasks are another important enabler for AI-based networking. ML frameworks offer high-level programming interfaces for designing, training and validating ML algorithms. However, current ML frameworks, such as TensorFlow, Caffe, and Theano, are designed for general-purpose tasks and impose too heavy a burden for the networking domain. They need to be further optimized to satisfy the requirements for networking applications, such as high processing speed, low complexity, and light weight.

6.3. Promoting ML Algorithms

While myriad ML algorithms have been developed, current ML algorithms are typically driven by the needs of specific existing applications, such as Computer Vision (CV) and Natural Language Processing (NLP). For example, convolutional neural networks are fascinating and powerful tools for image and audio recognition that can even achieve superhuman performance on many tasks. However, the networking domain involves completely different theoretical mathematical models compared to those found in the fields of computer vision and NLP. Convolutional layers or recurrent layers may not work effectively in the networking domain. In addition, networks involve far more data and stringent response time demands, which pose great challenges for ML deployment. Therefore, the demanding requirements and specific characteristics of the networking domain will require both the adaptation of existing algorithms and the development of new ones [7]. Thus, efforts to meet the needs of the networking domain, as a new application domain for ML, will drive advances in both the ML and networking domains to a new level. [27]

7. Conclusion

In this article, we first explored two deployment models for an intelligent control plane in a network and discussed the unique advantages and disadvantages of the centralized and distributed paradigms. Then, we proposed a hybrid ML paradigm for packet routing, in which we combine distributed AI routers with a centralized network mind to address the needs of different network services. In our paradigm, we deploy a centralized AI control plane for tunneling-based routing and a hybrid AI architecture for hop-by-hop routing. In addition, we apply two kinds of RL algorithms to optimize the routing strategies.

References

[1] C. Jiang, H. Zhang, Y. Ren, Z. Han, K.-C. Chen, and L. Hanzo, "Machine learning paradigms for next-generation wireless networks," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 98–105, Apr. 2017.

[2] H. Yao, T. Mai, X. Xu, P. Zhang, M. Li, and Y. Liu, "NetworkAI: An intelligent network architecture for self-learning control strategies in software defined networks," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4319–4327, Dec. 2018.

[3] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine learning for networking: Workflow, advances and opportunities," *IEEE Netw.*, vol. 32, no. 2, pp. 92–99, Apr. 2018.

[4] H. Yang, J. Wen, X.-J. Wu, L. He, and S. G. Mumtaz, "An efficient edge artificial intelligence multi-pedestrian tracking method with rank constraint," *IEEE Trans. Ind. Informat.*, Feb. 2019. doi: 10.1109/TII.2019.2897128.

[5] S. Goudarzi, N. Kama, M. H. Anisi, S. Zeadally, and S. Mumtaz, "Data collection using unmanned aerial vehicles for internet of things platforms," *Comput. Electr. Eng.*, vol. 75, pp. 1–15, May 2019.

[6] D. D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski, "A knowledge plane for the internet," in *Proc. Conf. Applications, Technologies, Architectures, and Protocols for Computer Communications*, Karlsruhe, Aug. 25–29, 2003, pp. 3–10.

[7] A. Mestres et al., "Knowledge-defined networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 47, no. 3, pp. 2–10, July 2017.

[8] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," in *Proc. Int. Conf. Neural Information Processing Systems*, Denver, 1993, pp. 671–678.

[9] S. P. Choi and D.-Y. Yeung, "Predictive Q-routing: A memory-based reinforcement learning approach to adaptive traffic control," in *Advances in Neural Information Processing Systems*, Denver, Dec. 2–5, 1996, pp. 945–951.

[10] S. Kumar and R. Miikkulainen, "Dual reinforcement Q-routing: An on-line adaptive routing algorithm," in *Proc. Artificial Neural Networks in Engineering Conf.*, 1997, pp. 231–238.

[11] A. A. Bhorkar, M. Naghshvar, T. Javidi, and B. D. Rao, "Adaptive opportunistic routing for wireless ad hoc networks," *IEEE Trans. Netw.*, vol. 20, no. 1, pp. 243–256, Feb. 2012.

[12] R. Arroyo-Valles, R. Alaiz-Rodriguez, A. Guerrero-Curieses, and J. Cid-Sueiro, "Q-probabilistic routing in wireless sensor networks," in *Proc. Int. Conf. Intelligent Sensors, Sensor Networks and Information*, Melbourne, Dec. 3–6, 2007, pp. 1–6.

[13] P. Stone, "TPOT-RL applied to network routing," in *Proc. Int. Conf. Machine Learning*, California, June 29–July 2, 2000, pp. 935–942.

[14] P. Stone and M. Veloso, "Team-partitioned, opaque-transition reinforcement learning," in *Robot Soccer World Cup*, Springer, 1998, pp. 261–272.

[15] D. Wolpert, K. Tumer, and J. Frank, "Using collective intelligence to route internet traffic," in *Advances in Neural Information Processing Systems*, Denver, Nov. 29–Dec. 4, 1999, pp. 952–960.

[16] J. Dowling, E. Curran, R. Cunningham, and V. Cahill, "Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing," *IEEE Trans. Syst., Man, Cybern.*, vol. 35, no. 3, pp. 360–372, May 2005.

[17] A. Elwhishi, P. H. Ho, K. Naik, and B. Shihada, "ARBR: Adaptive reinforcement-based routing for DTN," in *Proc. IEEE Int. Conf. Wireless and Mobile Computing, Networking and Communications*, Ontario, Oct. 10–13, 2010, pp. 376–385.

[18] G. Stampa et al., "A deep-reinforcement learning approach for software-defined networking routing optimization," arXiv Preprint, arXiv:1709.07080, Sept. 2017.

[19] 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 *Proc. IEEE Int. Conf. Services Computing*, San Francisco, June 27–July 2, 2016, pp. 25–33.

[20] P. Wang and T. Wang, "Adaptive routing for sensor networks using reinforcement learning," in *Proc. IEEE Int. Conf. Computer and Information Technology*, Seoul, Sept. 20–22, 2006, pp. 219–225.

[21] H. Yao, X. Chen, M. Li, P. Zhang, and L. Wang, "A novel reinforcement learning algorithm for virtual network embedding," *Neurocomputing*, vol. 284, pp. 1–9, Apr. 2018.

[22] X. Liang, Y. Li, G. Han, H. Dai, and H. V. Poor, "A secure mobile crowdsensing game with deep reinforcement learning," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 1, pp. 35–47, Jan. 2018.

[23] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," arXiv Preprint, arXiv:1509.02971, Sept. 2015.

[24] S. Bock, J. Goppold, and M. Wei, "An improvement of the convergence proof of the ADAM-optimizer," arXiv Preprint, arXiv:1804.10587, Apr. 2018.

[25] K. Malialis, S. Devlin, and D. Kudenko, "Resource abstraction for reinforcement learning in multiagent congestion problems," in *Proc. 2016 Int. Conf. Autonomous Agents and Multiagent Systems*, Singapore, May 9–13, 2016, pp. 503–511.

[26] "Congestion-problem," [Online]. Available: https://github.com/rradules/congestion-problems

[27] H. Yao, C. Jiang, and Y. Qian, *Developing Networks Using Artificial Intelligence*. Springer, 2019.

[28] Y. Go, M. A. Jamshed, Y. Moon, C. Hwang, and K. Park, "APUNet: Revitalizing GPU as packet processing accelerator," in *Proc. USENIX Symp. Networked System Design and Implementation*, Boston, Mar. 27–29, 2017, pp. 83–96.