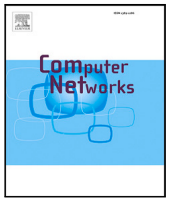
Computer Networks 192 (2021) 108057

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



# A deep reinforcement learning-based multi-optimality routing scheme for dynamic IoT networks☆

Peizhuang Cong [a], Yuchao Zhang [a,*], Zheli Liu [b], Thar Baker [c], Hissam Tawfik [d], Wendong Wang [a,*], Ke Xu [e], Ruidong Li [f], Fuliang Li [g]

[a] Beijing University of Posts and Telecommunications, Beijing, China
[b] Nankai University, Tianjin, China
[c] Unviersity of Sharjah, Sharjah, United Arab Emirates
[d] Leeds Beckett University, Leeds, UK
[e] Tsinghua University, Beijing, China
[f] Kanazawa University, Kanazawa, Japan
[g] Northeastern University, Shenyang, China

ARTICLE INFO

*Keywords:*
Routing
Multi-optimality criteria
Deep reinforcement learning
Deep Q network

ABSTRACT

With the development of Internet of Things (IoT) and 5G technologies, more and more applications, such as autonomous vehicles and tele-medicine, become more sensitive to network latency and accuracy, which require routing schemes to be more flexible and efficient. In order to meet such urgent need, learning-based routing strategies are emerging as strong candidate solutions, with the advantages of high flexibility and accuracy. These strategies can be divided into two categories, centralized and distributed, enjoying the advantages of high precision and high efficiency, respectively. However, routing becomes more complex in dynamic IoT network, where the link connections and access states are time-varying, hence these learning-based routing mechanisms are required to have the capability to adapt to network changes in real time. In this paper, we designed and implemented both centralized and distributed Reinforcement Learning-based Routing schemes combined with Multi-optimality routing criteria (RLR-M). By conducting a series of experiments, we performed a comprehensive analysis of the results and arrived at the conclusion that the centralized is better suited to cope with dynamic networks due to its faster reconvergence (2.2 × over distributed), while the distributed is better positioned to handle with large-scale networks through its high scalability (1.6 × over centralized). Moreover, the multi-optimality routing scheme is implemented through model fusion, which is more flexible than traditional strategies and as such is better placed to meet the needs of IoT.

## 1. Introduction

Along with the burgeoning development of the Internet in recent years, emerging network applications, such as industrial Internet, Internet of Vehicles (IoV), 4K/8K video transmission and edge computing, are requiring routing schemes to be more efficient. Nevertheless, today's network is no longer as stable as the traditional wired network due to the access of a large number of mobile devices, which make network connection status change frequently. The conflict between application requirement and network characteristic brings great challenges to the network in providing efficient and flexible routing decisions. Moreover, the increasing number of service types brings multiple optimization objectives which involve bandwidth, delay, packet loss rate, link utilization and what not, e.g., industrial Internet and IoV require low-delay or deterministic-delay and file transfer requires inverse-bandwidth. Even some services are multi-optimal criteria, like

☆ The work was supported in part by the National Natural Science Foundation of China (NSFC) Youth Science Foundation under Grant 61802024, the Fundamental Research Funds for the Central Universities under Grant 2482020RC36, the NSFC under Grant 62072047, and the National Key R&D Program of China under Grant 2019YFB1802603. Ruidong Li's work was supported in part by the JSPS KAKENHI Grant Number 19H04105.

* Corresponding authors.
*E-mail addresses:* congpeizhuang@bupt.edu.cn (P. Cong), yczhang@bupt.edu.cn (Y. Zhang), liuzheli@nankai.edu.cn (Z. Liu), tshamsa@sharjah.ac.ae (T. Baker), h.tawfik@leedsbeckett.ac.uk (H. Tawfik), wdwang@bupt.edu.cn (W. Wang), xuke@tsinghua.edu.cn (K. Xu), lrd@se.kanazawa-u.ac.jp (R. Li), lifuliang@cse.neu.edu.cn (F. Li).
*URL:* http://yuchaozhang.weebly.com/ (Y. Zhang).

https://doi.org/10.1016/j.comnet.2021.108057
Received 14 November 2020; Received in revised form 3 March 2021; Accepted 23 March 2021
Available online 3 April 2021

video transmission requires minimum delay if bandwidth is sufficiently high [1].

To ensure the Quality of Service (QoS) in current dynamic networks, improving hardware infrastructures not only causes huge cost but also has limitations of performance improvement. Meanwhile, a research of CAIDA [2] shows that the existing network still has a lot of room for optimization. Therefore, many mathematical model-based network optimization schemes have been proposed [3,4]. Most of these works are to simplify specific scenarios through idealized assumptions and many constraints so that network optimization problems can be solved efficiently by mathematical methods. However, there are many uncertainties in the real scene, so the actual effect of this kind method cannot be guaranteed. Besides, there is no single general model to solve multiple routing optimization tasks at the same time, and modeling each task separately would affect the scalability of the network. Therefore, the machine learning (ML) promises to bring a new perspective to solve this problem.

The improvement of ML and hardware devices, like CPU and GPU, have made the artificial intelligence model possess powerful learning capabilities and generalization capabilities. And thanks to Software Defined Network (SDN) and programmable routing devices, intelligent routing based on ML has noble feasibility. Data-driven intelligent routing has features of high accuracy and extreme versatility. Models trained by different data set can solve various network optimization problems without complicated network environment assumptions and modeling. Some existing researches show that the ML-based routing strategies have been successfully applied in many scenarios, such as opportunistic networks [5], wireless networks [6], IoT [7], and improved in accuracy and performance than the traditional routing protocol [7]. The deployment of the ML-based routing scheme can be divided into two categories, centralized and distributed. The centralized gets the globe network state and makes globe routing decisions via a centralized controller which is akin to the SDN controller, and the distributed makes single routing hop by each router. Most of current networks are time-varying, whose connections are can be established and canceled at any time, or nodes be accessed and be eliminated. Both the centralized and distributed require the model to achieve convergence or it will make wrong decision. The aforementioned dynamics lead to the issue of the fitted model non-convergence again. And combined with multiple optimality routing criteria, we complete this paper.

In summary, our contributions are as follows.

- We designed a deep reinforcement learning-based routing scheme and implement it in both centralized and distributed mode.
- We compared and analyzed performances of the two modes via experiments, especially under dynamic conditions.
- Multi-optimal routing schemes are implemented via model fusion, which is more flexible than traditional routing strategies and can better meet the development needs of IoT.

The remainder of this paper is organized as follows. Related works are reviewed in Section 2. Section 3 describes details and specific of RLR-M model. Section 4 presents the evaluations and analyses. Finally, the conclusions and future work are provided in Section 5.

## 2. Background and related work

The traditional transmission service is 'best effort' which only involves reachability of source and destination and does not involve other link attributes. This kind of service principle can no longer meet the current network transmission requirements, then QoS routing has become a critical and necessary guarantee. The guarantee provided by QoS can be divided into two main types. The first is the kind of cumulative constraints, which refers to the final performance is the cumulation of all partial effects of passed links from the source to the destination, such as link delay, loss rate or jitter. And the second is the kind of bottleneck constraints, which means that the transmission performance is determined by the worst limitation of all segment links, the most typical is the bandwidth. And many tasks requires multiple optimization criteria, which makes the mathematical-based optimization model more complicated and inadequate, the ML-based routing strategies have emerged.

The development of network technology makes the IoT popular, being characterized by huge data volume, high proportion of uplink data, stable traffic, et al. [8]. More and more mobile devices are accessed to the network, such as smartphones, automobiles, resulting in quite dynamic IoT networks [9,10]. The dynamic nature of IoT network, e.i. time-varying connections of links and access status, requires routing strategies to vary to adapt the network. It should be noted that the ML-based decision system can fit the optimal result only when the model converges. The dynamic feature will have a great influence on it, making it necessary to retrain the model to refit the current network status. In this paper, only the dynamic characteristics of the IoT are considered, i.e., the time-varying nature of its topology state. Its combination with dynamic network control is estimated as the future work. In the rest of this section, representative works on AI-based network optimization schemes, reinforcement learning, and multiple optimality criteria routing are presented.

### 2.1. AI-based network

Given the successful application of machine learning in natural language processing, computer vision and other fields, many scholars try to apply AI technology to network context, including congestion control, resource allocation, security and so on [11]. Zhang et al. optimized the video caching strategy of edge servers through model prediction [12]. Liang et al. optimized the decision tree generation strategy for flow classification by an reinforcement learning model [13]. Hua et al. put forward a reward-clipping mechanism to stabilize GAN-DDQN training against the effects of widely-spanning utility values to solve the problem of several slices in a radio access network with base stations which share the same physical resources [14]. Kasim proposed a robust deep learning based network anomaly detection against distributed denial of service attacks which speeds up training and testing times and performs better classification performance metrics than traditional approaches [15]. In the routing realm, Mao et al. proposed a Deep Belief Network (DBN)-based routing scheme in the backbone of the network, where the domain's border routers calculate the best inter-domain path for packets [16]. Xiao et al. combined a deep learning model and the link reversal theory to generate a Decision Directed Acyclic Graph (DDAG) and assign some weights to all links, then choosing an optimal path via a greedy algorithm when making routing decisions [17].

### 2.2. Reinforcement learning

Reinforcement Learning (RL) learns the optimal strategy by maximizing accumulated rewards which is suitable for decision-making problems [18]. The overall process of RL is as follows: (1) The agent chooses the action $A_t$ according to the strategy $\pi$ in the current state $S_t$. (2) The environment transfers to the next state $S_{t+1}$ according to $A_t$. (3) The agent receives the feed back reward $R_t$ by the environment and chooses the next action $A_{t+1}$ according to strategy $\pi$. In the above process, the strategy $\pi$ is constantly adjusted so that $\pi$ can fit the optimal decision for states $S_i$. Traditional RL usually solves the decision strategy function by iterating the Bellman optimality equation, but this method is too costly or even infeasible in the large state space. Then, the Deep Reinforcement Learning (DRL) emerged [19], which combined the advantages of deep learning and RL, and solved the excessive data dimensions problem. So far, DRL can be divided into three categories [20]: value function-based (e.g., DQN, DDQN, DRQN [21–24]), policy gradient-based (DDPG, A3C [25,26]), and search and supervision-based (AlphaGo [27]). A classic DQN model structure is

shown as Fig. 1, which fits the decision function through a deep neural network. The experience playback mechanism is used in model training process, and data sample $E = (S_t, A_t, R_t, S_{t+1})$ obtained from online processing, which will be stored in the Experience Pool ($EP$). During training, a batch of $E$ is selected from $EP$, and the stochastic gradient descent algorithm is devoted to updating the network parameters $\theta$. The deep neural network requires training data to be independent of each other. This random sampling method reduces the correlation between $E$ and thus improves the stability of the algorithm.

In the Internet, DRL is widely used in traffic engineering, traffic prediction and routing configuration, etc. Xu et al. used DRL for intra-domain traffic engineering optimization, and proposed the DRL-based traffic engineering solution, DRL-TE, which uses traditional methods to generate paths and uses a DRL unit to adjust the routing path split ratio online [28]. Valadarsky et al. tried to predict the future network traffic based on historical traffic data through the DRL unit and calculated the appropriate routing configuration based on the predicted results [29]. Ramy et al. developed a hierarchical cluster-oriented adaptive per-flow path calculation mechanism by leveraging the Deep Double Q-Network (DDQN) algorithm, where the end-to-end paths are calculated by the source nodes with the assistance of cluster leaders at different hierarchical levels [30,31]. Recent researches devoted to applying DRL to IoT involved task scheduling, energy-saving, and routing. Wei et al. proposed a Q-learning based task scheduling algorithm for wireless sensor networks, ISVM-Q, to achieve better application performance with less energy consumption [32]. In terms of routing application of DRL in IoT, Gagandeep et al. proposed a DRL-based intelligent routing scheme for IoT-enabled WSNs, which utilizes a novel unequal clustering strategy to improve energy efficiency and number of alive nodes [33]. Sergio et al. designed a simple RL algorithm based on epsilon-greedy, Epsilon Multi-Hop (EMH), which enabled reliable and low consumption low-power wide area networks multi-hop topologies and achieved significant energy savings with respect to the default single-hop approach [34]. To enhance the throughput of IoT, Ding et al. re-modeled routing selection problem as a Markov decision process and used DRL to make decisions. The simulation results show that the proposed method can significantly reduce the congestion probability [35]. The DRL-based routing decision algorithms for IoT in the aforementioned are all centralized and devoted to the optimization of goals such as energy-saving or throughput. These researches do not involve the influence of the dynamic characteristics of IoT on learning-based routing decision strategies. Therefore, we emphasis modeling and experimental analysis on this aspect.

RL can solve the problem of sequence decision optimization through a reward feedback mechanism, and does not require labeled data. The difficulty of data annotation in the network environment and the fact that network transmission performance can be used as natural decision feedback make DRL more suitable for routing decisions than AI/ML.

### 2.3. Routing on multiple optimality criteria

The multi-objective path problem is a classical issue of the Opsearch that have been studied by the research community for a long time [36–38]. In concrete algebraic terms, this problem can be described as that attributes are tuples of the Cartesian product of elementary metrics, each of which either extends with + and is ordered by $\leq$ or extends with min/max and is ordered by $\geq$ or $\leq$. Tuples extend termwise and are partially ordered by the product order of their termwise total orders. The goal is to find sets of dominant tuples from source to destination in a network and is attained with generalizations of Dijkstra's and Bellman-Ford algorithms [39].

Sobrinho's series early works proposed the algebraic framework based on total order of attributes for a unified treatment of routing problems and protocols, which abstracting away the specificity of performance metrics and protocol parameters [40–42]. And he proposed the path constraints model with a set of attributes and the corresponding protocol to provide a fully distributed solution for addressing routing on multiple optimality criteria problems [39].

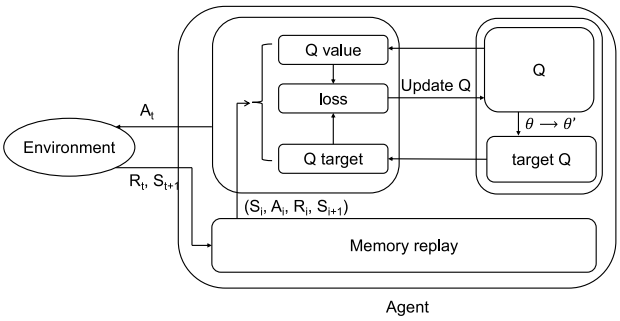



**Fig. 1.** DQN structure.

## 3. Models design

In this section, we propose both two of centralized and distributed deep reinforcement learning-based single-optimality routing algorithms and realize multi-optimality routing through model fusion, and all details are presented.

### 3.1. Variable definition

A network topology is defined as a directed graph, $\langle N, E\rangle$. All routers are defined as a node set $N$, and router $i$ is $N_i$. And all links are denoted as edge set $E$, $E_{ij} \in E$ means there is a direct link between $N_i$ and $N_j$. All declarations are shown in Table 1.

### 3.2. MDP of routing decision

The routing decision can be transformed into a Markov Decision Process (MDP), which is a Markov Process (MP) containing rewards and decisions and can be represented by the tuple, $\langle S, A, \pi, R\rangle$. $S$, $A$ and $R$ are state, action and reward respectively, which are described in the Table 1. And $\pi$ is the state transition probability parameter. The routing decision is only involved in current state, then the Markov property of decisions can be expressed by Eq. (1).

$$Pr(s_{n+1}|s_0, a_0, \ldots, s_n, a_n) = Pr(s_{n+1}|s_n, a_n) \tag{1}$$

In a task, an action is selected according to the current state. After the action is executed, the routing network will feedback the next state and corresponding rewards. This process is repeated until the task is complete.

### 3.3. Accumulative constrain type

In this subsection, we take the link delay as a typical cumulative constraint type and implement the generic routing decision model based on reinforcement learning in two modes: centralized and distributed.

#### 3.3.1. Centralized routing scheme

*(a) Overall Structure*

This fashion needs a central controller that akin to the SDN controller to make all routing decisions. It can train the DRL-based model offline and push forwarding rules to all routers via OpenFlow protocol. It is different from the traditional routing protocol in this mode. All routers no longer need to interact with neighbors, but the state of the entire network is maintained by the controller. The centralized mode can be directly upgraded in the existing network running the OpenFlow protocol and does not require to do much work on the underlying equipment.

As shown in Fig. 2, the environment generates a reward based on change of state after performing the last action. The reward is

**Table 1**
Declaration of notations.

| Notation | Declaration |
|---|---|
| $N$ | the set of all nodes |
| $N_i$ | means router i, and $N_i \in N$ |
| $E$ | the set of all links |
| $E_{ij}$ | means the link of $N_i$ to $N_j$, if it is exist then $E_{ij} \in E$, otherwise, $E_{ij} \notin E$ |
| $des$ | destination of a task |
| $S$ | state of the model environment |
| $A$ | action |
| $R$ | reward of state change, $R_{S_i \to S_j}$ means reward of change from $S_i$ to $S_j$. $R_{max}$ means the biggest prize when the task is done and $R_{min}$ means the harshest punishment which usually a small negative number |
| $C_{ij}$ | the cost of transform from $N_i$ to $N_j$, such as time consumption. |
| $B_{ij}$ | the link constraint of $N_i$ to $N_j$, such as link bandwidth. |
| $q\_target$ | a network trained by data of state change and related reward |
| $q\_value$ | a network evaluates the action and updated by $q\_target$ |

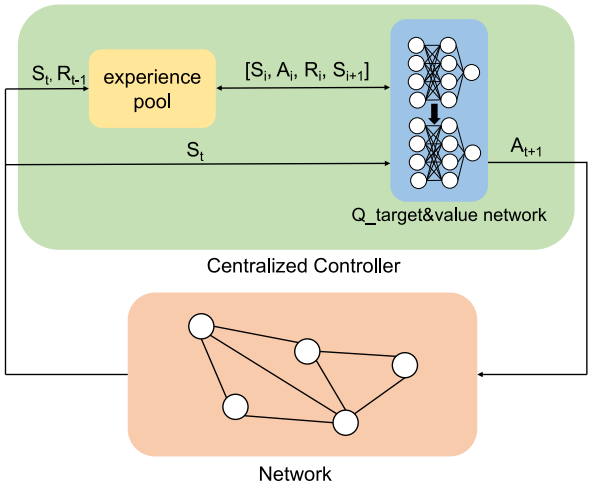



**Fig. 2.** Centralized RL-based routing model.

**Algorithm 1** Centralized Process.

**Initial:** $S_{current}$ = environment.state()
1: # Interaction Process:
2: **while TRUE do**
3: $A$ = q_value($S_{current}$)
4: $S_{next}$, $R$ = environment.execute($A$)
5: experience_pool.save([$S_{current}$, $A$, $R$, $S_{next}$])
6: $S_{current} = S_{next}$
7: **end while**
8: # Training Process:
9: **while TRUE do**
10: $replay\_data$ = experience_pool.sample(batch_size)
11: q_target.train($replay\_data$)
12: **if** isupdate == TRUE **then**
13: q_value.copy_parameters(q_target)
14: **end if**
15: **end while**

usually related to the optimal criteria, if the transmission target is to minimize delay, the smaller the delay of routing decision, the greater the reward, and vice versa. In a period, it involves the state before action, action, state after the action, reward, and whether the task is done. This five-tuple will be stored in the experience pool of model to make the previous five-tuples used for training the $q\_target$ network independently. The state after action will be taken as the input of $q\_value$ which will output an action with biggest reward of this state. The environment performs this action and feedbacks a reward, then a new period is completed. The pseudocode of the centralized process is shown in the Algorithm 1. The $q\_target$ obtains previous data from the experience pool and trains the model. The $q\_value$ has the same network structure as $q\_target$, and its parameters are updated periodically by the $q\_target$.

*(b) Detailed design*

If routing decisions on all transport tasks need to be made using a model, then there must be at least two essential properties for a packet forwarding, the source address, and the destination address. Besides, because forwarding satisfies the MP, the address of the router that processes the packet is the source address of the current state, which is independent of all previous states. Therefore, we define that the state has two attributes, the current and the destination, as shown in the Eq. (2). After performing the forwarding action, only the current will be modified, and the destination address remains to be consistent with the entire transmission process.

$$S = [N_{current}, N_{des}] \tag{2}$$

The reward is a significant part of DRL, which needs to be designed with different calculation rules according to diverse tasks and the transition of environmental states. Before that, it was explained that the action space is nodes set $N$, and the action output by the model is $N_i, \in N$, such as Eq. (3).

$$A = [N_{next}] \tag{3}$$

For a transmission task, the evaluation of the distance to the destination will change after each routing decision is executed: better or not. And combining the reachability of routes, rewards can be divided into the following three categories. First, if $N_{next}$ is not directly connected, the reward is the harshest punishment, $R_{min}$, which is usually a small negative number. Second, if $N_{next}$ is $N_{des}$, the reward is the maximum reward, $R_{max}$, which is usually a large positive number. For the third case, $N_{next}$ is not $N_{des}$ and it can be reached directly by $N_{current}$, the reward is a positive coefficient the $\alpha$ multiply the minimum cost of from $N_{next}$'s neighbors to $N_{des}$ minus the cost of from $N_{next}$ to $N_{des}$. As shown in formula Eq. (4). Due to the iterative computation, the value of $N_{current} = N_{next} = N_{des}$ is required, which is already included in the above case because the cost of from one to itself is 0. It needs to be explained that the absolute value of $R_{min}$ and $R_{max}$ is much larger than the weight of the link. For example, in the experiment of Section 4,

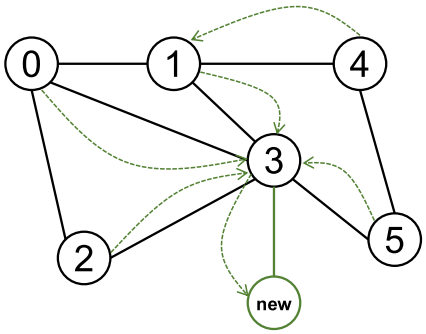



**Fig. 3.** New node accessed.

each link weight is a random number less than 20, and $R_{min}$ and $R_{max}$ are −100 and 100 respectively. $R_{[i,des]\to[j,des]} =$

$$\begin{cases} R_{min}, & E_{ij} \notin E \\ R_{max}, & E_{ij} \in E \ \& \ j = des \\ \alpha(argmin_{k\in neighbor(i)}(C(k,des)) - C(j,des), & \\ \qquad E_{ij} \in E \ \& \ j \neq des) & \end{cases} \tag{4}$$

Since the controller has the global perspective of the network, it can calculate the cumulative reward value for all nodes that made forwarding decisions along the routing path based on historical optimal routing decision and the final routing decision performance. Then, it can put the reward value as the label to train the *q_target*, as shown in the Eq. (5).

$$q\_target_{[i,des]\to[j,des]} = R_{[i,des]\to[j,des]} \tag{5}$$

The *q_value* network is periodically updated by *q_target* using the copy scheme.

*(c) Optimizations*

For centralized mode, as the controller can obtain global status information of network, then it is not necessary to retrain the decision model for any network topology changes, such as, when a new node only accesses network via only one of existed nodes (we named this new node as one-path node, and the connected node as accessed node). The accessed node can be regarded as one-path node's agency, because it must go through accessed node when other nodes communicate with one-path, as shown in Fig. 3. Therefore, the decision model can ensure the accuracy by a mapping operation.

When a new node connects multiple old node when accesses network (similarly, we named this new node as multi-path node), it has multiple accessed nodes and may affect optimal paths of initial network. However, the controller can make quasi-optimal routing decisions in agent strategy that mentioned above. Therefore, it is unnecessary to retrain decision model when existing few multi-path nodes if the network system is not extremely strict with the accuracy of the routing decision. In Evaluation section, we made statistics on the influence of the number of one-path nodes and multi-path nodes on the accuracy of decision model in different scale networks, which can be used as a guide for when to start retraining.

Based on the proposed agent strategy, we also make optimization on the network, which is called pruning. Nodes with 1 degree in the topology can be regarded as one-path nodes and can be proxied. These nodes on a non-loop with 2 degrees in topology can also be proxied because the router first to judge whether the destination of the received packet is itself, and if not, the packet can be forwarded directly to its another port, the operation is in an unconscious state for proxied node. In the Evaluation section, we performed pruning operations on more than 20 real network topologies.

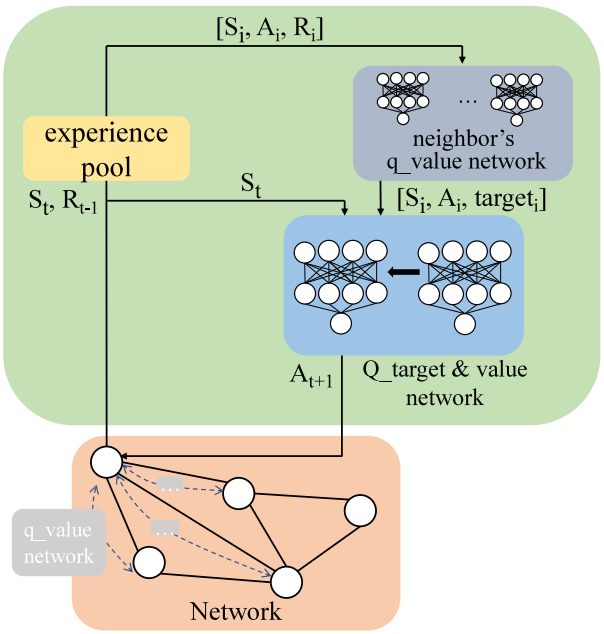



**Fig. 4.** Distributed RL-based routing model.

*3.3.2. Distributed routing scheme*

*(a) Overall structure*

Different from the above-mentioned centralized mode, the distributed mode does not require a centralized controller. Each router only needs to maintain its link states to its neighbors and interact *q_value* network with them. It is similar to the traditional routing protocol, but the optimal path is calculated through the neural network model. And information in the routing table is no longer distance metrics but some parameters of *q_value* network.

Each router in the network needs to maintain its model. Take one router as an example, as shown in Fig. 4. The router quantifies environmental states and rewards and stores them in the experience pool. When training the *q_target* network, it is necessary to combine the *q_value* network of the maintained neighbors and the reward to calculate the target value. This process takes into account the cumulative reward to prevent the decision model from only selecting the optimal link based on the immediate reward. The parameters of *q_value* are maintained and updated by *q_target*. Taking the transmission task label as the input of *q_value* and selecting the maximum value action according to the calculation result. This action will be sent to the router forwarding module or as an entry updated to the local routing table. When the action result of *q_value* changes, the latest network parameters need to be diffused to neighbors. The pseudocode of the distributed process is shown in the Algorithm 2.

*(b) Detail design*

Since the distributed mode does not have a global perspective and all routers only maintain their state and neighbor *q_values*, considering that the forwarding of data packets satisfies the Markov random process, the state in this mode only needs to have a destination parameter, such as the Eq. (6) shown. The action of the model is the same as the centralized mode as Eq. (3).

$$S = [N_{des}] \tag{6}$$

The same as explanation of the Eq. (4) descriptions above, the reward calculation of distributed mode can be divided into the following two situations. The first, if $N_{next}$ is not $N_{des}$, whether it can be reached directly by $N_{current}$ or not, the reward is the harshest punishment, $R_{min}$. The second is that the $N_{next}$ can be reached directly by $N_{current}$ and it

**Algorithm 2** Distributed Process.

**Take $N_i$ as example:**
**Initial:** $S_{current}$ = get_des(packet)
1: # Interaction Process:
2: **while TRUE do**
3: $A$ = q_value($S_{current}$)
4: $R$ = environment.execute($A$)
5: experience_pool.save([$S_{current}$, $A$, $R$])
6: $S_{current}$ = get_des(packet)
7: **end while**
8: # Training Process:
9: **while TRUE do**
10: $replay\_data$ = experience_pool.sample(batch_size)
11: $q\_target\_label$ = $replay\_data$ + cumulative_reward()
12: q_target.train($replay\_data$)
13: **if** isupdate == TRUE **then**
14: q_value.copy_parameters(q_target)
15: **end if**
16: **end while**
17: # Diffusion and Reception of $q\_value$:
18: **while TRUE do**
19: **if** ischange(q_value) == TRUE **then**
20: send_neighbors(q_value)
21: **end if**
22: **if** isreceive(q_value) == TRUE **then**
23: $N_i$.update_neighbors(q_value)
24: **end if**
25: **end while**

is $N_{des}$, then the reward is the maximum reward, $R_{max}$, minus the cost of from $N_{current}$ to $N_{next}$ multiply the positive coefficient, $\alpha$. Again, it includes the value of $N_{current} = N_{next} = N_{des}$ due to the cost of from one to itself. As shown in the Eq. (7). Similarly, the settings of $R_{min}$ and $R_{max}$ follow the above constraints.

$$R_{\rightarrow j}^{i,[des]} = \begin{cases} R_{min}, & E_{ij} \notin E \text{ or } j \neq des \\ R_{max} - \alpha C_{ij}, & E_{ij} \in E \ \& \ j = des \end{cases} \tag{7}$$

To prevent the model from causing local optimum by choosing actions based only on a one-step reward, the $q\_target$ network update takes into account two parts. One is the reward immediately obtained by the action, and the other is the long-term cumulative reward, which is calculated by neighbor's $q\_value$ network according to the action. The formula is as Eq. (8), which means that the target value from $[N_i]$ to $[N_j]$ under destination node is $N_{des}$ should be the direct reward from $[N_i]$ to $[N_j]$ plus the max value calculated by $N_j$'s $q\_value$ to $[N_{des}]$. This is an iterative process, and it will finally converge to a stable state.

$$q\_target_{\rightarrow j}^{i,[des]} = R_{\rightarrow j}^{i,[des]} + \gamma max(q\_value_{\rightarrow k}^{j,[des]} | N_k \in N) \tag{8}$$

In the same way, the $q\_value$ is updated by $q\_target$ periodically as mentioned above.

*(c) Discussions*

It is important to note that in the distributed mode, each node needs to complete local training before federated training. That is, it need to be familiar with all of its neighbors. The experiment results show that the federated model will converge slowly or even nonconvergent if the model cannot fit routing decisions to their neighbors.

### 3.4. Bottleneck constraint type

Since the proposed routing decision model is generic, taking the link bandwidth as an example of the bottleneck constraint type, we only need to redefine the reward and retrain the model. It should be noted that different from the cumulative constraint type, the bottleneck constraint type routing model needs to feedback a "long view" reward for each decision, otherwise, it may miss the global optimum because of the local optimum. The centralized mode has global state information that can be converted into reward values according to the final transmission performance, while the distributed mode needs to transfer the reward values from the destination to the source in the reverse direction. The details of the network model and the main process are basically same as the cumulative type that will not be repeated.

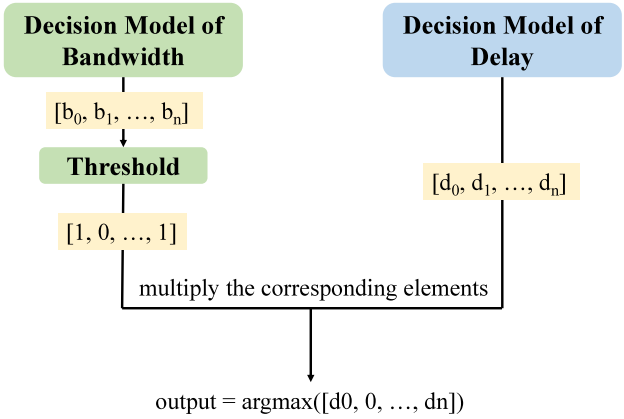



**Fig. 5.** Model fusion for $\langle bandwidth, delay \rangle$ criterion.

### 3.5. Multi-optimality routing scheme

If the multi-optimality criteria are integrated into a model through the reward value, it will not only reduce the flexibility of the model but also need to retrain the model when there is an attribute that needs to be refitted. If many attributes are maintained, the model will be retrained frequently. Therefore, we complete multi-optimality criteria routing decision requirements through model fusion. Taking the principle of bandwidth ($b$) over delay ($d$) as an example, if $b_1 > b_2$, then the link with $\langle b_1, d_1 \rangle$ takes precedence over the link with $\langle b_2, d_2 \rangle$ for any $d_1$ and $d_2$, and if $b_1 = b_2$ or $b_1, b_2 > threshold$, then $\langle b_1, d_1 \rangle$ takes precedence over $\langle b_2, d_2 \rangle$ when $d_1 < d_2$.

As shown in Fig. 5, the bandwidth-based model outputs a decision vector, and each element is compared with the threshold to generate a new vector $\vec{B}$, where the element is 1 if the corresponding element in $\vec{B}$ is greater than the threshold, otherwise, it is 0. The delay-based model outputs decision vector $\vec{D}$ and multiplies the corresponding elements of $\vec{D}$ and $\vec{B}$ to generate $\vec{T}$. The index of the maximum value of $\vec{T}$ is the final decision result.

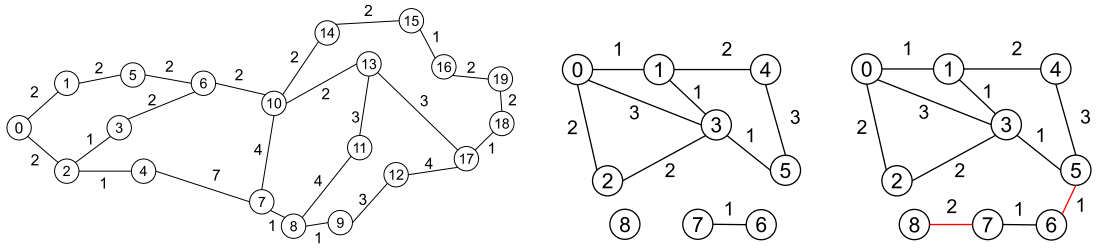
## 4. Evaluation

In this section, we explain the experimental setup and analyzed the RLR-M in convergence time and reconvergence time when network state changes under different network scales.

### 4.1. Experiment setup

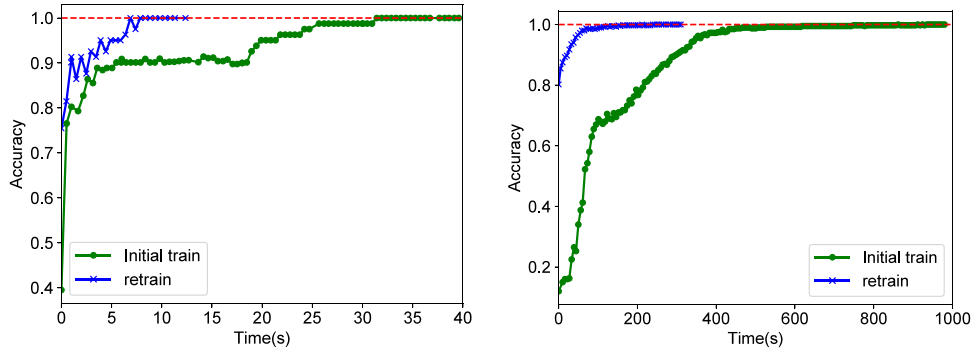
The two modes of centralized and distributed RL-based models are implemented by TensorFlow 1.3.0 and on the Ubuntu 16.04-LTS operating system. The GPU is GeForce GTX 970 and the CPU is Intel Xeon 3.30 GHz × 8.

The neural network in DQN adopts the mode of full connection of a 10 layers neural network. By minimizing the value of loss function, the mean square error of network output and input reward, to adjust model weights and bias. The optimizer is stochastic gradient descent (SGD).



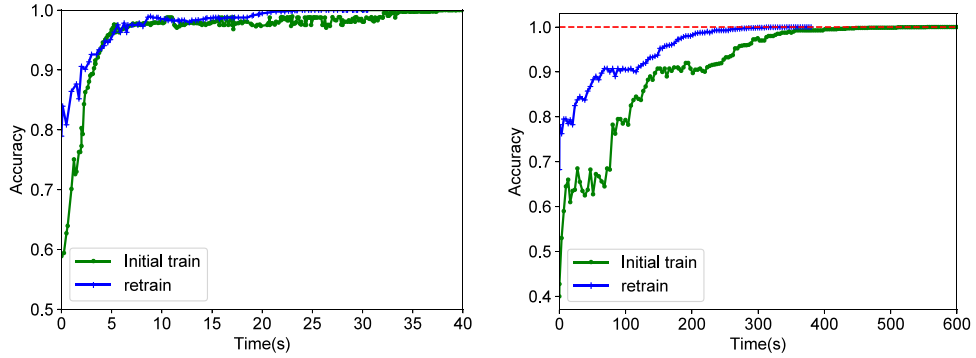
(a) Topology of 20 nodes (b) Topology of 8 nodes (c) New topology of 8 nodes

**Fig. 6.** Topologies.



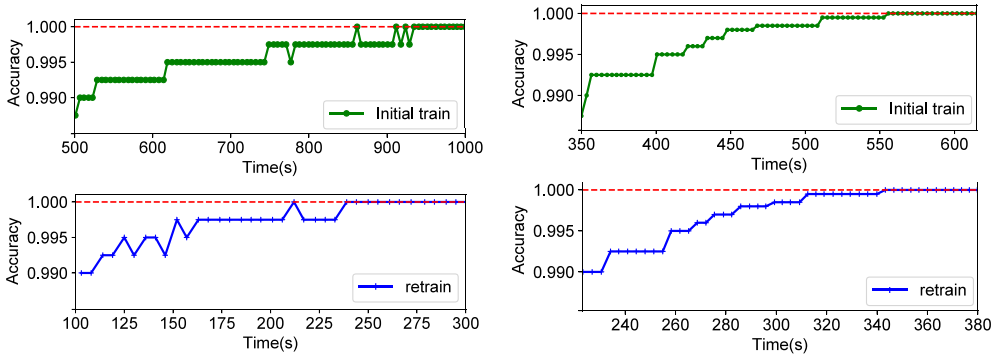

(a) Performance in 8 nodes topology (b) Performance in 20 nodes topology

**Fig. 7.** Accuracy performance of centralized mode.



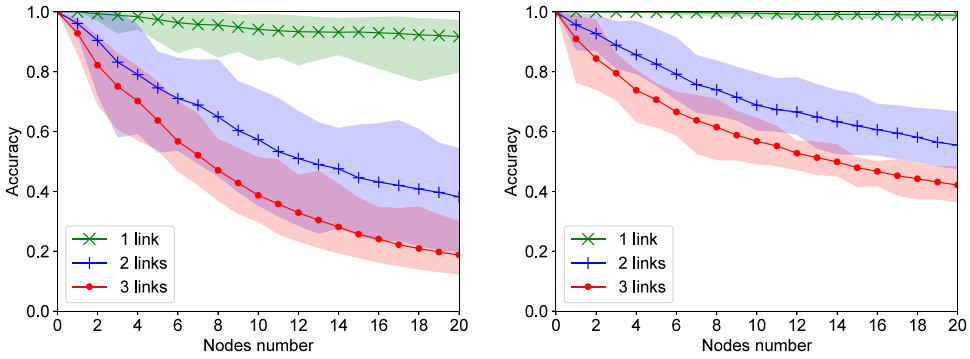

(a) Performance in 8 nodes topology (b) Performance in 20 nodes topology

**Fig. 8.** Accuracy performance of distributed mode.



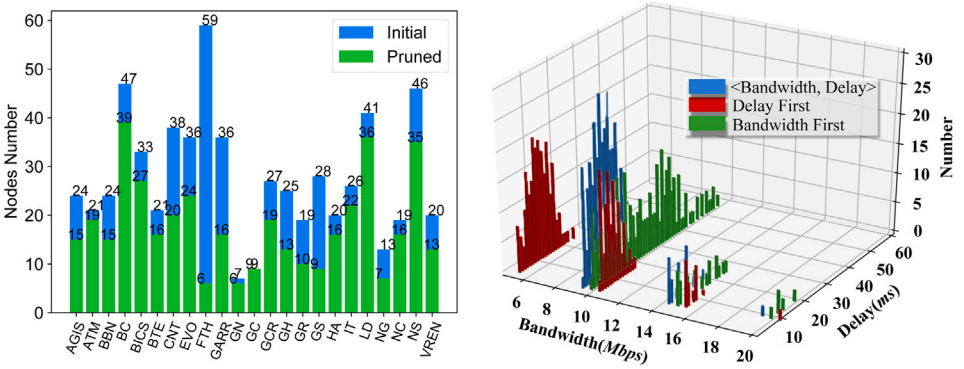

(a) Performance of centralized mode (b) Performance of distributed mode

**Fig. 9.** Tail accuracy performance of 20 nodes topology.




(a) Performance of 6 nodes topology (b) Performance of 20 nodes topology

**Fig. 10.** Accuracy performance of agency strategy.




(a) Node scale comparison of pruned and initial networks (b) Distribution of bandwidth&delay and corresponding task number

**Fig. 11.** Pruning and Routing decision performance distribution.

**Topologies:** Two topologies, with 20 nodes and 8 nodes, were set up in the experiment shown in the Figs. 6 and 6(b). The 20-nodes one is based on Savvis 2011 USA topology [43]. To increase the number of alternative paths in this topology, three routers are added to form some loops. Link weights (delays in ms) are assigned. And setting acnodes ($N_8$) and unconnected subgraphs ($N_6$ and $N_7$) in 8-nodes one is to verify the validity of setting edge weights to represent node exits. In the process of model retraining, topology changes as following rules, the 20-nodes changes the weight of some links randomly, and 8-nodes is connected to all subgraphs, as shown in Fig. 6(c).

### 4.2. Experimental results

In this subsection, we analyze the routing decision model performance, effects of optimization for centralized mode and the performance of model fusion-based multi-optimality routing scheme.

#### 4.2.1. Model performance

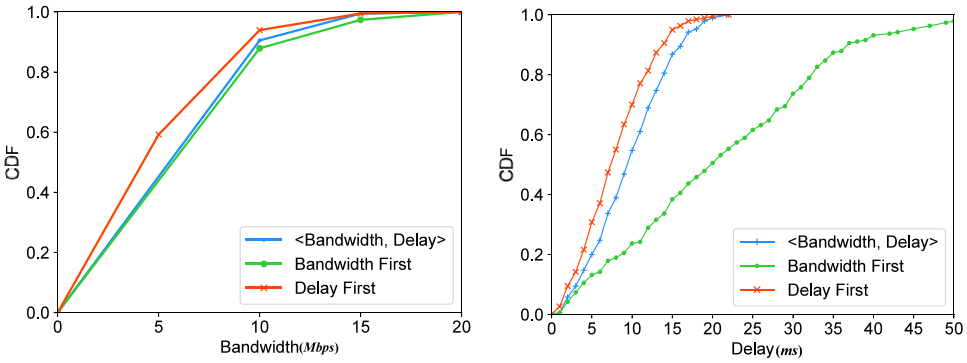
Based on the experimental setup above, for single-optimality routing decision model, we take delay as example and use centralized and distributed mode respectively in the two topologies. The verification label of accuracy is the globally optimal decision calculated according to the shortest path algorithm. The accuracy of each calculation is to verify whether the forwarding port of all points to other points except itself conforms to the label.

*(a) Centralized mode*

In this mode, the relationship between accuracy and training time in the two topologies is shown in Figs. 7(a) and 7(b). The retraining process is to save the convergent model of the initial training first, and then modify the network state, such as modifying links weight randomly or adding or deleting nodes, and then load the saved model parameters for retraining. On the basis of some fitting ability, the accuracy of the model at the beginning time has reached a relatively high, so it can reconverge faster than the initial training. In the 8-N topology, initial training time, as the line marked by '•' in Fig. 7(a), takes about 30 s to stabilize at 100% accuracy. The retraining time, as the line marked by '|' in Fig. 7(b), only takes 10 s to reach 100% accuracy state. In the topology with 20 nodes, the accuracy goes up quickly, but it also takes a long time to reach 100% accuracy. Similarly, the time of retraining is significantly shorter than its the initial training.

*(b) Distributed mode*

In this mode, each point needs to process distribute training first, which means the model should fit neighbors states, and then completes the federal training of the entire network via the neighbor-to-neighbor *q_value* exchange. In the topology with 8 nodes, due to the small scale, the neighborhood fitting process makes it achieve relatively high accuracy quickly, reaching 90% accuracy in less than 5 s and 100% in less than 40 s, as shown in Fig. 8(a). The time to reach 90% accuracy of retraining is similar to the initial training, but the time to reach 100% is slightly shorter. In the topology with 20 nodes, the fluctuation in the beginning stage of initial training is severe, because the updating of neighbor *q_value* has a great influence on the decision making, and the global *q_value* convergence is a conjunct process. The early stage of




(a) Cumulative distribution of task bandwidth (b) Cumulative distribution of task delay

**Fig. 12.** Cumulative Distribution.

retraining is more stable, and the convergence state with high accuracy can be achieved more quickly, as shown in Fig. 8(b).

*(c) From excellent to perfect*

For the model, it is time-consuming to converge to 100% accuracy. The tail accuracy performance from 99% to 100% in the topology with 20 nodes as shown in Figs. 9(a) and 9(b). Tail convergence accounts for a large proportion of the overall situation. We analyze the decision results of the tail stage. The most are suffered fitting fluctuation between the sub-optimal and the optimal. Only a tiny learning rate and multiple iterations can complete the final fitting. And it will be more obvious when the sub-optimal and the optimal are close, but this is not necessary. One is that it is acceptable to choose a sub-optimal path approaching optimal when there is only one optimal criterion. The second is that multi-optimal criteria are usually considered in the current network, such as selecting the comprehensive optimal path subjected by delay and bandwidth, so using top N optimal sets intersection to relax the model fitting accuracy, and then avoiding the tail convergence time.

*(d) Analysis*

According to the above results, the distributed mode requires a process akin to route switching, so the reconvergence time is not as good as the centralized mode. However, when the scale gradually increases, the distributed can improve accuracy and converge faster than the centralized. On large scale, directional diffusion optimization can further improve the performance of the distributed mode, but the centralized mode will be greatly affected.

#### 4.2.2. Optimization performance

*(a) Agency strategy*

Under the two topologies, we counted the relationship between the number of new connected nodes and the accuracy of the initial model for the changed network. New access nodes establish 1 to 3 connections with the network randomly. The comparison of accuracy is the shortest path obtained by the Dijkstra algorithm. The final result, as shown in Figs. 10 and 10(b), is the average of 50 random repetitions.

*(b) Pruning performance*

We performed pruning operations on more than 20 real network topologies provided by Zoo [43] and obtained a node scale comparison between the pruned network and its initial, as shown in Fig. 11. The results show that, especially in a network containing star topology, pruning can effectively reduce the scale of network nodes.

#### 4.2.3. Multi-optimality routing scheme performance

The bandwidth values of 5, 10, 15, or 20 ($Mbps$) are randomly assigned to each link of the topology with 20 nodes and train the delay-based and bandwidth-based models respectively. In the topology, 20 nodes transmit to each other corresponding to a total of $20 \times 19 = 380$ tasks. We use bandwidth-first, delay-first, and multi-optimality routing strategies to make routing decisions for these all tasks. And suppose the bandwidth threshold is 10, that is, the link with bandwidth 5 will be bypassed in the multi-optimality routing decision. Count the delay and path bandwidth required by all tasks. The relationship between bandwidth&delay and the number of corresponding tasks is shown in Fig. 11(b). For the delay first algorithm, there will be many cases where the transmission path bandwidth is 5. And for the bandwidth first, detours occur to select the maximum bandwidth path, which increases the transmission delay. The proposed multi-optimality routing model makes a trade-off on the premise of satisfying the bandwidth and most of its tasks are in the range of the satisfying bandwidth and the lower delay.

We respectively count the number of tasks corresponding to bandwidth and delay, and cumulative distributions are shown in Figs. 12 and 12(b). For bandwidth, although the value category of the overall network bandwidth is small, it can still be seen that multi-optimality routing model is closer to bandwidth first in bandwidth performance. For delay, the multi-optimality routing model is close to that of the delay first algorithm in delay performance, which is far better than the bandwidth first algorithm.

## 5. Conclusion

This work adopts a DRL approach for routing optimization for dynamic IoT network. Although conventional machine learning can break through some bottlenecks of traditional methods in the network routing, it still has some limitations, such as limited applicabilities to some special scenarios. This paper designed a generic RL-based routing decision model, implemented it in two modes, centralized and distributed, compared, analyzed time of model convergence and reconvergence, and revealed the advantages and characteristics of each mode in different networks. The centralized mode can achieve 2.2 × speedup over distributed in reconvergence time, which is suitable to cope with dynamic networks. While the distributed mode can achieve 1.6 × over centralized in scalability, which is better to handle with large-scale networks. Moreover, the multi-optimality routing scheme is realized through model fusion, which is more flexible and efficient than traditional strategies to meet the development needs of the Internet of Things.

Based on our research foundation on dynamic network control and driving node selection, calculating which routers are suitable as key nodes to hierarchize the network topology to improve the model's scalability and deploy the proposed system to the real network environment are identified as main avenues for future works.

## CRediT authorship contribution statement

**Peizhuang Cong:** Conceptualization, Design of study, Acquisition of data, Analysis and/or Interpretation of data, Writing - original draft, Writing - review & editing. **Yuchao Zhang:** Conceptualization, Design of study, Acquisition of data, Analysis and/or Interpretation of data, Writing - review & editing. **Zheli Liu:** Writing - review & editing. **Thar Baker:** Writing - review & editing. **Hissam Tawfik:** Writing - review & editing. **Wendong Wang:** Writing - review & editing. **Ke Xu:** Writing - review & editing. **Ruidong Li:** Writing - review & editing. **Fuliang Li:** Writing - review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References


[1] K.-F. Hsu, R. Beckett, A. Chen, J. Rexford, D. Walker, Contra: A programmable system for performance-aware routing, in: 17th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 20), 2020, pp. 701–721.

[2] CAIDA, The cooperative asociate for internet data analysis, 2019, http://www.caida.org/data.

[3] S. Ghorbani, Z. Yang, P.B. Godfrey, Y. Ganjali, A. Firoozshahian, Drill: Micro load balancing for low-latency data center networks, in: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, 2017, pp. 225–238.

[4] P. Kumar, Y. Yuan, C. Yu, N. Foster, R. Kleinberg, P. Lapukhov, C.L. Lim, R. Soulé, Semi-oblivious traffic engineering: The road not taken, in: 15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18), 2018, pp. 157–170.

[5] S.K. Dhurandher, J. Singh, M.S. Obaidat, I. Woungang, S. Srivastava, J.J. Rodrigues, Reinforcement learning-based routing protocol for opportunistic networks, in: ICC 2020-2020 IEEE International Conference on Communications (ICC), IEEE, 2020, pp. 1–6.

[6] X. Guo, H. Lin, Z. Li, M. Peng, Deep reinforcement learning based QoS-aware secure routing for SDN-IoT, IEEE Internet Things J. (2019).

[7] Z. Zhuang, J. Wang, Q. Qi, H. Sun, J. Liao, Toward greater intelligence in route planning: A graph-aware deep learning approach, IEEE Syst. J. (2019).

[8] M. Mohammadi, A. Al-Fuqaha, S. Sorour, M. Guizani, Deep learning for IoT big data and streaming analytics: A survey, IEEE Commun. Surv. Tutor. 20 (4) (2018) 2923–2960.

[9] P. Cong, Y. Zhang, W. Wang, B. Bai, Dnd: The controllability of dynamic temporal network in smart transportations, in: 2019 IEEE Globecom Workshops (GC Wkshps), IEEE, 2019, pp. 1–6.

[10] P. Cong, Y. Zhang, W. Wang, N. Zhang, Dnd: Driver node detection for control message diffusion in smart transportations, IEEE Trans. Netw. Serv. Manag. (2021) 1–12.

[11] Z. Chen, J. Hu, G. Min, A.Y. Zomaya, T. El-Ghazawi, Towards accurate prediction for high-dimensional and highly-variable cloud workloads with deep learning, IEEE Trans. Parallel Distrib. Syst. 31 (4) (2019) 923–934.

[12] Y. Zhang, P. Li, Z. Zhang, B. Bai, G. Zhang, W. Wang, B. Lian, K. Xu, Autosight: Distributed edge caching in short video network, IEEE Network (2020) 194–199.

[13] E. Liang, H. Zhu, X. Jin, I. Stoica, Neural packet classification, in: Proceedings of the ACM Special Interest Group on Data Communication, 2019, pp. 256–269.

[14] Y. Hua, R. Li, Z. Zhao, X. Chen, H. Zhang, Gan-powered deep distributional reinforcement learning for resource management in network slicing, IEEE J. Sel. Areas Commun. 38 (2) (2019) 334–349.

[15] Ö. Kasim, An efficient and robust deep learning based network anomaly detection against distributed denial of service attacks, Comput. Netw. 180 (2020) 107390.

[16] B. Mao, Z.M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, K. Mizutani, Routing or computing? The paradigm shift towards intelligent computer network packet transmission based on deep learning, IEEE Trans. Comput. 66 (11) (2017) 1946–1960.

[17] S. Xiao, H. Mao, B. Wu, W. Liu, F. Li, Neural packet routing, in: Proceedings of the Workshop on Network Meets AI & ML, 2020, pp. 28–34.

[18] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT press, 2018.

[19] Y. Li, Deep reinforcement learning: An overview, 2017, arXiv preprint arXiv:1701.07274.

[20] Q. Liu, J. Zhai, Z. Zhang, S. Zhong, Q. Zhou, P. Zhang, X. Jin, A survey on deep reinforcement learning, Chinese J. Comput. 41 (1) (2018) 1–27.

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, 2013, arXiv preprint arXiv:1312.5602.

[22] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.

[23] H. Van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (1), 2016.

[24] M. Hausknecht, P. Stone, Deep recurrent q-learning for partially observable mdps, 2015, arXiv preprint arXiv:1507.06527.

[25] N. Heess, G. Wayne, D. Silver, T. Lillicrap, Y. Tassa, T. Erez, Learning continuous control policies by stochastic value gradients, 2015, arXiv preprint arXiv:1510.09142.

[26] V. Mnih, A.P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, in: International Conference on Machine Learning, PMLR, 2016, pp. 1928–1937.

[27] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.

[28] Y. Zhan, P. Li, S. Guo, Experience-driven computational resource allocation of federated learning by deep reinforcement learning, in: Proc. of IPDPS, 2020.

[29] A. Valadarsky, M. Schapira, D. Shahaf, A. Tamar, Learning to route, in: Proceedings of the 16th ACM Workshop on Hot Topics in Networks, 2017, pp. 185–191.

[30] R.E. Ali, B. Erman, E. Baştuğ, B. Cilli, Hierarchical deep double Q-routing, in: ICC 2020-2020 IEEE International Conference on Communications (ICC), IEEE, 2020, pp. 1–7.

[31] J. Wang, J. Hu, G. Min, A.Y. Zomaya, N. Georgalas, Fast adaptive task offloading in edge computing based on meta reinforcement learning, IEEE Trans. Parallel Distrib. Syst. 32 (1) (2020) 242–253.

[32] Z. Wei, F. Liu, Y. Zhang, J. Xu, J. Ji, Z. Lyu, A Q-learning algorithm for task scheduling based on improved SVM in wireless sensor networks, Comput. Netw. 161 (2019) 138–149.

[33] G. Kaur, P. Chanak, M. Bhattacharya, Energy efficient intelligent routing scheme for IoT-enabled WSNs, IEEE Internet Things J. (2021).

[34] S. Barrachina-Muñoz, T. Adame, A. Bel, B. Bellalta, Towards energy efficient LPWANs through learning-based multi-hop routing, in: 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), IEEE, 2019, pp. 644–649.

[35] R. Ding, F. Gao, L. Xing, Intelligent routing strategy in the internet of things based on deep reinforcement learning, Chin. J. Internet Things 3 (2) (2019) 56–63.

[36] E.Q.V. Martins, On a multicriteria shortest path problem, European J. Oper. Res. 16 (2) (1984) 236–245.

[37] J. Brumbaugh-Smith, D. Shier, An empirical investigation of some bicriterion shortest path algorithms, European J. Oper. Res. 43 (2) (1989) 216–224.

[38] G. Fandel, T. Gal, Multiple Criteria Decision Making Theory and Application: Proceedings of the Third Conference Hagen/Königswinter, West Germany, August 20–24, 1979, 177, Springer Science & Business Media, 2012.

[39] J.L. Sobrinho, M.A. Ferreira, Routing on multiple optimality criteria, in: Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication, 2020, pp. 211–225.

[40] J.L. Sobrinho, Algebra and algorithms for qos path computation and hop-by-hop routing in the internet, in: Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No. 01CH37213), 2, IEEE, 2001, pp. 727–735.

[41] J.L. Sobrinho, An algebraic theory of dynamic network routing, IEEE/ACM Trans. Netw. 13 (5) (2005) 1160–1173.

[42] T.G. Griffin, J.L. Sobrinho, Metarouting, in: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, 2005, pp. 1–12.

[43] Zoo, The internet topology zoo, 2020, http://www.topology-zoo.org/.


**Peizhuang Cong** is currently a Ph.D. candidate in State Key Laboratory of Network and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China. His current research interests include the next generation network architecture, data-driven networks and mobile Internet.

**Yuchao Zhang** received her Ph.D. degree from Computer Science Department at Tsinghua University in 2017. Before that, she received the B.S. degree in computer science and technology from Jilin University in 2012. Her research interests include large scale datacenter networks, content delivery networks, data-driven networks and edge computing. She is currently with the Beijing University of Posts and Telecommunications as an associate professor.

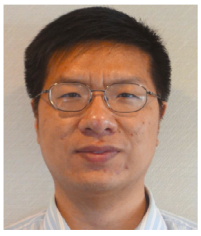
**Zheli Liu** received the BSc and MSc degrees in computer science from Jilin University, China, in 2002 and 2005, respectively. He received the PhD degree in computer application from Jilin University in 2009. After a postdoctoral fellowship in Nankai University, he joined the College of Computer and Control Engineering of Nankai University in 2011. Currently, he works at Nankai University as an Associate Professor. His current research interests include applied cryptography and data privacy protection.

**Thar Baker** received the Ph.D. degree in autonomic cloud applications from Liverpool John Moores University (LJMU), Liverpool, U.K., in 2010. In 2018, he became a Senior Fellow of Higher Education Academy. He has authored/coauthored numerous refereed research papers in multidisciplinary research areas including big data, algorithm design, green and sustainable computing, and energy routing protocols. Dr. Baker has been actively involved as a member of editorial board and review committee for a number of peer-reviewed international journals, and is on program committee for a number of international conferences. He is an Associate Editor for Future Generation Computer Systems. He is an Expert Evaluator of European Horizon 2020 (EU H2020), Information and Communication Technology (ICT) Fund, and British Council.

**Hissam Tawfik** received the M.Sc. and Ph.D. degrees in computer engineering from the University of Manchester, Manchester, U.K. He currently leads the Data Science and AI research at the school of Computing, Creative Technologies and Engineering. Before joining Leeds Beckett University in 2015, Hissam worked for University of Salford and Liverpool Hope University. He is an editor of the International Journal of Future Generation Computer Systems (Elsevier), the International Journal of Neural Computing and Applications (Springer), and International Journal of Medical Engineering and Informatics and a guest Editor of the International Journal of Medical Informatics (Elsevier). He is a visiting Professor at the University of Seville (Spain), and is a chair of the International Conference Series on Developments in eSystems Engineering (DESE). His research interests include the areas of intelligent systems, human–computer interaction and virtual reality applications, traffic and urban simulation, e-health and e-collaboration, and user-centered system design.

**Wendong Wang** received his B.E. and M.E. degrees both from the Beijing University of Posts and Telecommunications, China, in 1985 and 1991, respectively, where he is currently a Full Professor in State Key Laboratory of Networking and Switching Technology. He has published over 200 of papers in various journals and conference proceedings. His current research interests are the next generation network architecture, network resources management and QoS, and mobile Internet.

**Ke Xu** received his Ph.D. from the Department of Computer Science and Technology at Tsinghua University, where he serves as full professor. He serves as an associate editor for IEEE Internet of Things Journal and has guest edited several special issues in IEEE and Springer Journals. His research interests include next generation Internet, P2P systems, Internet of Things, network virtualization, and network economics. He is a member of ACM.



**Ruidong Li** received his Bachelor's degree in engineering from Zhejiang University, China, in 2001, and his Doctor of Engineering degree from the University of Tsukuba in 2008. His current research interests include future networks, big data networking, blockchain, the Internet of Things, network security, and wireless networks. He is the Secretary of the IEEE ComSoC Internet Technical Committee and the Founder and Chair of the IEEE SIG on Big Data Intelligent Networking and the IEEE SIG on Intelligent Internet Edge. He serves as the Co-Chair for the Young Professionals of the IEEE Tokyo Section and the Young Researcher Group in the Asia Future Internet Forum. He has been a Guest Editor of prestigious journals, such as IEEE Communications Magazine, IEEE Network, and IEEE Transactions on Network Science and Engineering, and serves/has served as Chair for international conferences and workshops, such as BRAINS 2020, IEEE INFOCOM ICCN Workshop 2019/2020, ICDCS NMIC Workshop 2019/2020, the IEEE GLOBECOM ICSTO Workshop, and ICCSSE 2019. He is a member of IEICE.

**Fuliang Li** received the B.S. degree in computer science from Northeastern University, Shenyang, China, in 2009, and the Ph.D. degree in computer science from Tsinghua University, Beijing, China, in 2015. He is currently an Associate Professor with the School of Computer Science and Engineering, Northeastern University. He was a Postdoctoral Fellow with the Department of Computing, Hong Kong Polytechnic University, Hong Kong, from 2016 to 2017. He published 50 journal/conference papers, including journal papers, such as IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON CLOUD COMPUTING, Computer Networks, COMPUTER COMMUNICATIONS, Journal of Network and Computer Applications, and mainstream conferences, such as IEEE INFOCOM, IEEE ICDCS, IEEE/ACM IWQoS, IEEE GLOBECOM, IEEE LCN, IEEE CLOUD, and IFIP/IEEE IM. His research interests include network management and measurement, mobile computing, software-defined networking, and network security.