SOHO-FL: A Fast Reconvergent Intra-domain Routing Scheme Using Federated Learning

Peizhuang Cong, Yuchao Zhang, *Member, IEEE*,
Wendong Wang, *Member, IEEE*, Ke Xu, *Senior Member, IEEE*

***Abstract*—The development of machine learning provides a new paradigm for network optimizations, e.g., reinforcement learning (RL) has brought great improvements in many fields, such as adaptive video streaming, congestion control of TCP. The fundamental mechanism of such RL-based architectures is that the neural network decision model converges to a stable state by continuously interacting with network environment. However, for network routing problem, such RL-based strategies do not work well due to topology change. This is because topological changes would require the existing RL models to be retrained, while these models may stop making routing decisions or provide non-optimal decisions during the slow reconverging process of retraining, seriously affected transmission performance. To solve this problem, we proposed a fast convergent RL-model (SOHO-FL), which can alleviate the performance degradation caused by the slow retraining process by federated learning. The experimental results based on real-world network topologies demonstrate that SOHO-FL outperforms the state-of-the-art algorithms in reconvergence time by 22.3% on average.**

***Index Terms*—routing, reinforcement learning, federated learning**

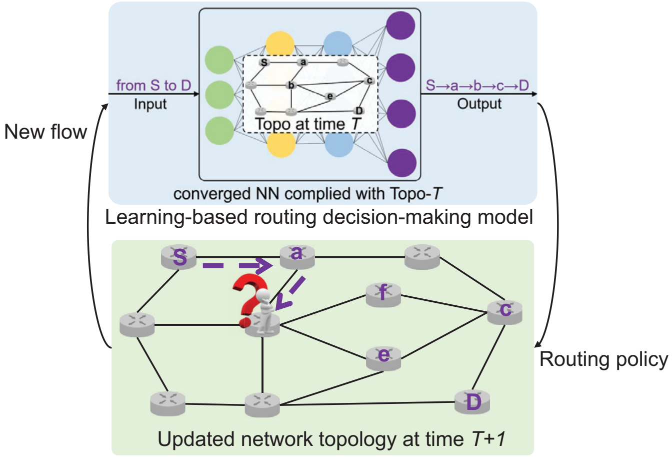

from S to D
Input
S→a→b→c→D
Output
S
a
b
c
e
D
Topo at time *T*
converged NN complied with Topo-*T*
Learning-based routing decision-making model
New flow
Routing policy
S
a
f
c
e
D
Updated network topology at time *T+1*

Fig. 1. Impact of topological changes on policies of learning-based routing decision-making model

I. Introduction

As a highly-regarded technology in recent years, machine learning (ML) has been widely employed in various fields, which provides new ideas for solving increasingly complex network optimization problems [1]–[3]. Reinforcement learning (RL), as an important branch of ML, is well-suited for being applied in decision-making scenarios, such as adaptive video streaming, congestion control of TCP, network resource management, etc [4]. RL-based routing decision architectures are currently a promising paradigm for dealing with sophisticated routing tasks and have shown preliminary performance in terms of accuracy and precision compared with traditional modeling approaches in some recent research works [5]–[7].

Such RL-based routing approaches need to be trained to converge under given network topology and traffic pattern, and only the converged model can provide stable well-performance routing policies. However, it is noted that the dynamicity, a common feature of online networks that can introduce topological changes, may invalidate the converged decision model and consequently result in inappropriate or even wrong routing decisions, as shown in Fig. 1[1]. When network dynamics trigger model retraining, it will take a long time for the model to reconverge through continuous interactions with the latest network environment and traffic [5]. Furthermore, during this process, the transmission performance of the network will be inevitably deteriorated by the unstable routing policies generated by unconverged decision model [6]. Hence, it is valuable and motive to accelerate the reconvergence process to facilitate such learning-based approaches.

The existing works that faces topological changes can be divided into the following categories. From the architecture perspective: 1) For centralized architectures, when there are topological changes, especially adding intermediate nodes, the follows that involved these new nodes can be forwarded timely following the proxy tactics [3], [6]. However, the proxy tactic can only temporarily provide simple reachability but fails to guarantee the routing performance, e.g., it will increase the link load on proxy nodes and thus may cause congestion; 2) For distributed architectures, each forwarding node independently maintains a local model, which involves a smaller scale of state space and action space than the centralized global model. Therefore, the distributed model can converge faster than the centralized global model in the same

Peizhuang Cong and Wendong Wang are with State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT). Yuchao Zhang is with BUPT. Ke Xu is with Tsinghua University. Corresponding authors: Yuchao Zhang (e-mail: yczhang@bupt.edu.cn; orcid: 0000-0002-0135-8915).

[1] For a routing request generated from the network with the updated topology, the routing decision-making model that converged on the previous topology and traffic can only infer it based on the previous knowledge, and thus, the routing policy cannot be applied to the updated network.

network environment [5], [6]. Nevertheless, the time overhead of convergence/reconvergence process is still unsatisfactory. From the structure of decision model perspective: non-tightly coupled neural network structures mainly enables the flexibility of the model in terms of topological changes. Although such the structure may possible to reduce the computation overhead of existing parameters, there is still a necessity for a reconvergence process for parameters of newly added neurons [5], [6]. That is, above schemes only passively adapt to changes, while a time-consuming retraining is still inevitable. In this work, based on existing studies, we propose a general architecture to accelerate the reconvergence process.

Inspired by Digital Twin Network (DTN) and Federated Learning (FL) technologies [8], [9], we proposed a **S**emi-**O**ffline **H**ybrid **O**nline reinforcement learning-based routing scheme (SOHO-FL), which can accelerate retraining process. When topological changes trigger the retraining process, SOHO-FL will immediately build multiple simulated network environments by extracting the latest network features through DTN, each of the simulated network contains an independent model (*twin model*), which will be trained by individual interactions with different traffic on the corresponding simulated network. Then, in line with FL, the parameters of all *twin models* are aggregated and updated periodically, in a FL way, to update the *master model*. Such offline multi-model federation can take advantage of the DTN to efficiently capture more knowledge for the online *master model*, and thus promotes faster reconvergence of retraining.

Our contributions are summarized as follows:

- We designed a novel neural network structure, SOHO, which is promising when facing frequent changes in network environment.
- We constructed a semi-offline hybrid online RL-based routing architecture, SOHO-FL, which, assisted by FL, can accelerate the reconvergence process of model retraining and provide performance-guaranteed routing policies efficiently.
- We conducted experiments based on real network topologies, and the results demonstrate that the proposed scheme outperforms the state-of-the-art algorithms in reconvergence time by 22.3% in average.

II. Related Work

The rapidly evolving deep learning promotes learning-based technologies to provide new paradigms for sophisticated network modeling and optimization problems, e.g., adaptive video streaming, congestion control of TCP [4]. In this section, we present some existing related studies from two aspects: learning-based routing strategies and federation learning applications.

A. Learning-based Routing

As an important category of machine learning, supervised learning algorithms are applied to routing strategies. Kato et al. [1] constructed a supervised deep learning-based traffic control system with appropriate input and output characterizations of heterogeneous network traffic, which outshines the widely used OSPF routing strategy; they further designed the subtle characterized input and output traffic patterns for the used deep belief network to enhance route computation, which outperforms the benchmark in terms of latency, throughput, and signaling overhead [2]. Zhuang et al. [3] leveraged a centralized graph deep learning model to calculate routing policies in Software Defined Network (SDN) architecture, which shows improvements in latency and efficiency. While such supervised learning-based routing tactics can provide satisfactory performance in specified networks, they still need to be improved regarding adaptation and reliability in dynamic networks.

Given that the features and execution process of reinforcement learning are well suited for solving decision problems with changing environments, it is widely used in network routing optimization. The prior works applied Q-learning to design routing algorithms for some specific optimization objectives and scenarios. However, the Q-learning-based model gets incapable when the network status or routing objectives become complicated. Therefore, recently, some studies start to employ deep reinforcement learning to solve complicated routing optimization tasks. Xu et al. [7] propose two tactics, traffic engineering-aware exploration and actor-critic-based prioritized experience replay, to specialize the general deep reinforcement learning framework, which aims to minimize the maximum link utilization by splitting each flow into different given candidate routing paths. Liu and Cong et al. [5], [6] all decompose the path generation process into hop-by-hop routing decisions. In addition, [6] employs a multi-model fusion strategy to flexibly satisfy multi-type transmission requirements and analyzes the impact of topological changes on model decision performance.

These existing studies, in the respective scenarios, showed promising performance and improvement, which are guaranteed by the convergent models. However, as [5] indicates, the reconvergence process of the models is time-consuming, which will be further exacerbated by the topological dynamics of the network. To this end, in this work, we strive to accelerate the reconvergence process of the decision model in learning-based network routing scenarios.

B. Federated Learning and Digital Twin Network

Federated learning was promoted for mainly addressing the privacy concerns of devices, which was utilized by Google company to deploy a model architecture that can enhance the accuracy of mobile device keyboard search recommendations on a global scale while not directly accessing the data of each device [10]. In recent years, FL is being intensively studied and extensively implemented. To improve the performance of FL, some works try to solve the issue of data heterogeneity of non-independent and identically distribution [11], accelerate parameters integration and models convergence [12]. As well, from application aspect, FL is applied to resource allocation for IoT, communication in connected automated vehicles network, and scheduling of the unmanned aerial vehicles [13], [14]. Chetan et al. [9] proposed a novel FL-based strategy to facilitate each model's personalization efficiency in game scenario.

Digital twin network aims to model a virtual representation of the physical communication network to ease the cost and risk practices on the real network. In DTN, the involved data is mainly generated from simulations, testbeds, or production. The model construction techniques mainly involve classical methods (Causal Bayesian Network, Hidden Markov Model, etc.), neural network-based methods (Convolutional Neural Networks, Graph Convolutional Networks, deep reinforcement learning model, etc.), and so on [8].

In this work, to alleviate the impact caused by network dynamics on learning-based routing decision model, we aim to utilize the federated learning framework to improve the efficiency of decision model reconvergence process combined with DTN technology.

III. Background and Motivation

A. General workflows of RL-based routing

In this part, we mainly introduce the overall process of RL-based routing decision structure, the neural network model outputs the forwarding path with the maximum reward for the given request under the current network state, i.e., obtains a comprehensive optimal transmission path with respect to delay, throughput, packet loss rate, etc. The three main workflows are as follows: 1) controller gathers the network states, which include but are not limited to links' property, reliability, utilization, and the reward of the transmission performance corresponding to the last round of decision; 2) the parameters of neural network model will be updated based on the reward and the action of the last round decision; 3) the model performs inference and outputs the routing policy based on network states for a new request. The policy will be pushed to the forwarding plane for further implementation. Similarly, the data involved in the new round will be also utilized to update the model.

B. Reconvergence challenge

Although some RL-based routing models can handle dynamicity of link properties including delay, reliability, utilization, etc., and even can temporarily address cancellations of links by setting the weights of attributes to zero or infinity, it is intractable to deal with topological changes when new link/routers/switches are accessed. To be direct, it is because such new nodes/connections will result in dimensional changes in the representation input/output vector of the environment/action space, which consequently fails to match the existing model correctly. It may be feasible to deal with the extensions of environment and action spaces by pre-setting large enough dimensional states (all reserved connections are covered by assigned zero/infinity weights). However, firstly, this approach requires a larger neural network model to extract environment and action features. If the training involves reserved space, it will require more training data and time, which would aggravate the inherently negative convergence issue of RL. If it is simply reserved spare dimensions in representation input/output vector, it still requires retraining the model when topological changes occur because of the lack of the knowledge of the new topology. In addition, large model will increase the computational overhead and latency of online inference when making routing policies. That is, the time-consuming reconvergence process of the decision model is inevitable under topological changes.

During the continuous interaction with the environment, the RL-based model performs exploration and exploitation based on the probabilities of ε and $1 - \varepsilon$, respectively. The ε will decrease continuously as the model converges, i.e., the model will stably choose the action with the maximum reward based on experience with a strong probability. Topological changes will disrupt such the stabilization, i.e., shifts of the state space and action space make model fail to work accurately. Even if the shifted state and action space can simply match the model, the new link/node will be naturally ignored due to the previous stabilization and the absence of up-to-date experience, which may cause non-optimal and even wrong decisions. In this situation, it is necessary to increase the exploration probability of the online model, i.e., the model will conduct extensive interactions with the latest network by the action and reward of online flows to reconverge into a new steady state.

However, the overhead of the reconvergence process cannot be ignored. On one hand, direct offline training is infeasible due to the lack of up-to-date network state and traffic data. On the other hand, either offline training or training the model through interactions with real-world environments can affect the transmission of online traffic. Moreover, this effect will persist throughout the reconvergence process.

C. Motivation and potential solution

As mentioned above, the duration and intensity of impact on flow transmission would be alleviated if the reconvergence process could be shortened. Then, it is valuable and motivating to investigate a scheme that can accelerate the model reconvergence for RL-based routing architectures. How to provide efficient and accurate retraining for the online decision model becomes a feasible approach.

Hence, inspired by federated learning [9] and enabled by DTN technology [8], we propose SOHO-FL, an FL-assisted framework for model reconvergence acceleration. DTN technology can provide additional latest interaction environments and extensive data for retraining, which enables retraining no longer solely depend on the online traffic data from the real-world. In addition, multiple parallel offline retraining can be federated by following the FL way, which can further improve the efficiency and performance of model retraining by fetching additional policy experience.

The workflows of acceleration are as shown in the Fig. 2, when topological changes of the network are detected, or when the transmission performance of routing decision degrades to the set threshold, the FL-assisted acceleration module will be triggered. The accelerator extracts the up-to-date network state via DTN technology and sends it to the control plane. From the implementation perspective, a straightforward solution is to set the network topology involving various properties of the links to the network simulator and generate different simulated traffic accordingly on networks with specified traffic generation patterns. Then, each digital twin emulator network

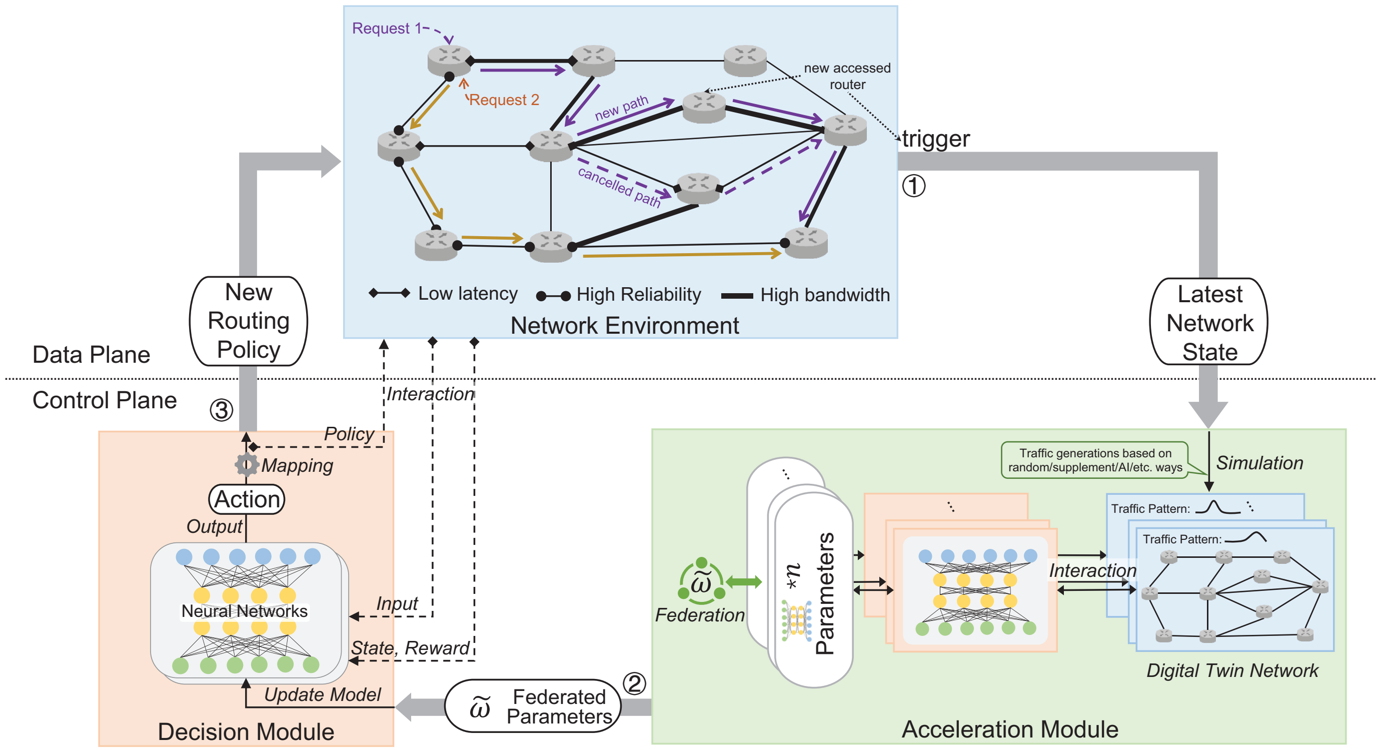


Fig. 2. SOHO-FL framework

trains its associated neural network model (*twin model*) by simulate interactions. The *twin model* has the same structure and the up-to-date state space and action space as the deployed model (*master models*) in the real-world network environment. After several iterations, all *twin models* upload the local parameters to the model controller. The controller aggregates the received parameters depending on specified functions and sends aggregated results to all *twin models* and the *master model* to update the parameters.

By doing so, the *master model* can be benefited from two aspects: all *twin models*' parameters, i.e., more experiences, can be integrated to update *master model*; digital twin network simulators can provide more frequent interaction iterations to update model efficiently. That is, based on SOHO-FL, the online *master model* can no longer depend entirely on the feedback of flows from real-world scenarios, instead, it can be updated more efficiently and faster in a semi-offline way.

IV. Design of SOHO-FL

A. Decision Module

Given the scalability, we designed a distributed architecture, i.e., each forwarding node maintains its decision model. Moreover, the distributed architecture also supports hybrid deployment. For the forwarding node in the network that does not support the RL-based routing structure, it can keep the traditional routing and forwarding way based on classic routing algorithms.

1) Variable definition: The *state space* ($\mathbb{S}$) contains both features of the flow request sub-state and the network sub-state. The information related to the flow request includes the source and destination addresses, the type of transmission of this flow; the information related to the network contains the shortest path distance from all neighbors to other destinations, and the available bandwidth of links, etc. The *action space* ($\mathbb{A}$) is the set of all neighboring nodes. In the employed distributed decision architecture, each action ($A, A \in \mathbb{A}$) is not a routing path, but the next-hop based on the current node. It is possible to obtain a complete path eventually after multiple decision iterations. The *reward* (R) is a quantitative evaluation of the transmission performance to the destination of the decision, and $R^j = \sum_{m=1}^{n} \alpha_i f(e^j_m)$, where e^j_m indicates the performance results of metric m (delay, throughput, etc.) of flow j, $f()$ is the regularization function to transform all metrics' results to a uniform scale (e.g., to ratio the results of metrics), and α_i is the weight of each metric for different transmission types (delay-first (DF), throughput-first (TF), delay-throughput-first (DTF), etc.), which are constrained by $\sum_{i=1}^{n} \alpha_i = 1$ and $\alpha_i \in [0, 1]$.

In this work, the delay (α_d) and throughput (α_t) metrics are considered. Then, for DF, α_d is 1 and α_t is 0; for TF, α_d is 0 and α_t is 1; and for DTF, both α_d and α_t are 0.5. And the Proximal Policy Optimization (PPO) algorithm [15] is adopted for the parameters update.

2) SOHO network structure: The neural network structure is shown in Fig. 3(a), which mainly consists of three parts: the

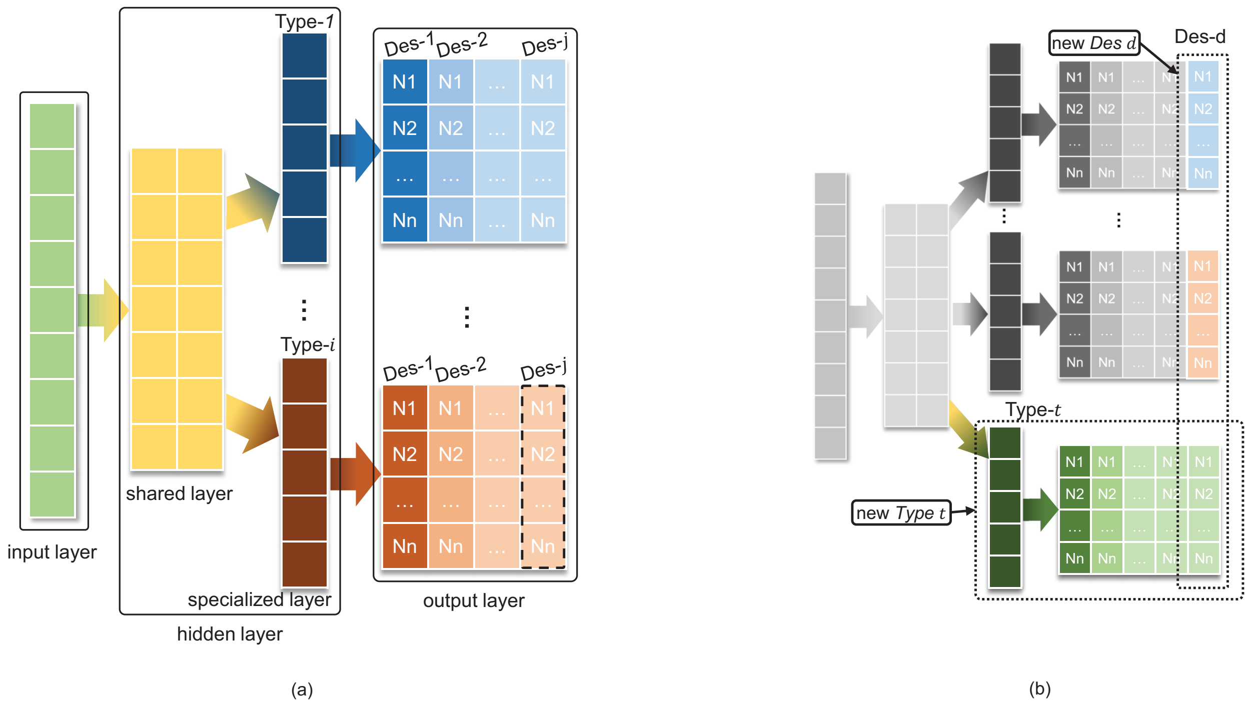


Fig. 3. Policy network structure

input layer, the hidden layer, and the output layer. For ease of observation and understanding, the output layer is shown in a folded form, which essentially is a one-dimensional vector.

- *Input layer:* The input layer is responsible for feeding the input data to the neural network. In line with the general pattern, the input data to the model is also a vector, the required addresses and the type of flow transmission are represented in one-hot mode. Since the decision model employs the distributed structure, combined with the Markovian property of flow forwarding, the source address can be transformed into the current node when making decisions when making the decision, i.e., it is possible to focus only on the destination address. Thus, the size of the input layer is $N_v + N_t$, where N_v is the number of forwarding nodes of the network and N_t is the total number of provided transmission services types.
- *Hidden layer:* The hidden layer includes the shared layer and the specialized layer. The shared layer is the generic fully-connected multi-layer neural network that extracts the features of the input state vectors, which can be viewed as a feature extractor with the network topology. The specialized layer is employed to cope with multi-transmission types, which technicalizes features for different types of transmission. The numbers of neurons and layers can be specified according to the topology scale in implementation.
- *Output layer:* Based on the input of the specialized layer, the output layer produces a probability distribution vector for different types and destinations. Each element in the vector represents the probability of selecting the corresponding indexed neighbor as the next hop to forward the flow. As shown in Figure 3(b), it is possible to directly add the corresponding neurons to cope with the requirement of adding new forwarding nodes. Similarly, the new transmission types correspond to adding new specialized layers and output neurons. Such decoupled output layer can effectively guarantee the flexibility and scalability of the model, which further alleviates the impact of dynamic changes in the network from the perspective of model structure.

3) SOHO Workflow: First, the input layer feeds the flow request vector into the hidden layer. Then, the hidden layer extracts and specializes the features to feed the output layer. Finally, the output layer generates a probability distribution vector. Suppose that, for a flow request (f^i_j) with specified destination and transmission type ($T = i$, $Des = j$), $F(f^i_j|\omega)$ is the output vector of neural network (ω) with input f^i_j. The probability distribution sub-vector, which indicates the probabilities of forwarding the flow to all n neighbors, can be located by features of T and Des. For example, the corresponding action sub-vector $F(f^i_j|\omega)[i][j]$ of f^i_j is circled by the dashed line in Fig. 3(a). Finally, the *action* is to forward this flow to the neighbor corresponding to the index of the maximum probability element, i.e., *action*(f^i_j)$= K$, where $K = argmax_k(F(f^i_j|\omega)[i][j][k] \,|\, k \in [1, n])$.

4) Loop-free guarantee scheme: Neural network decision-based routing policies cannot provide guaranteed reliability as rule-based scheme, e.g., the *exploration & exploitation* principle of reinforcement learning may produce an unacceptable action, which may cause loop routing problems. To solve this problem, inspired by the anti-loop policy of inter-domain routing protocol (*AS-path* attribute), we set a tag vector to

TABLE I
PERFORMANCE COMPARISON

Metric / Method	Average delay (ms)		Average throughput rate (%)	
	DF	DTF	TF	DTF
SPF	19.06	20.95	96.59	94.54
DRL-OR	**11.81**	11.96	99.85	**96.62**
SOHO-FL	11.82	**11.90**	**99.93**	96.58

record the passed nodes. When a loop decision is detected, i.e., the next hop already existed in the tag vector. Then, this inappropriate decision will be canceled and the new next hop will be selected according to a deterministic policy such as the shortest path algorithm. Meanwhile, a harsh punishment will be fed back to the neural network as the *reward* to reduce the probability of the same wrong decision next time.

B. Acceleration Module

The aforementioned decision model design involves specialization to handle the network dynamic changes from the perspective of neural network structure. However, there is still an issue of excessive overhead when reconvergence occurs. To this end, inspired by federal learning and based on the above design, we propose a FL-assisted model reconvergence acceleration mechanism. When detecting the requirement of converge the model, such as the number of added nodes exceeding a threshold or a severe drop in the performance of the model decision, then the acceleration process will be triggered. The ideal way is to emulate the latest network, including the corresponding traffic, by sophisticated digital twin network technologies [8]. In the current scenario and demands, from the implementation and deployment perspectives in this work, it can be achieved with the existing network simulation platforms.

Firstly, the network state, including network topology and all links' properties (e.g., delay, bandwidth, packet loss rate, jitters), are collected and imported to the network simulator. Then, setting the flow generation mode, which can be the supplemented the historical traffic with some random traffic, or generated simulated random traffic by the learning-based approach, e.g., Generative Adversarial Network (GAN). Moreover, the traffic patterns of each twin network are required to be different. That is, even if the same approach is employed to generate simulated traffic for all DTNs, it is stipulated to set different parameters under the generation characteristics to ensure the discrepancy of data.

Based on the simulators, we set up several *twin models*, which remain the same structure as the *master model*, to independently interact with different traffic on corresponding simulated network to update the decision model. After a period of interactions, or a time interval, the parameters of all *twin models* will be uploaded to the federated parameter controller, which aggregates the received parameters by the specified function. If the parameter upload is triggered at each fixed time interval, the number of interactions of each client could be different due to differences in computing power or simulation traffic. Then, the number of interactions will be used as a weight for parameters aggregation, e.g., $\omega' = \frac{1}{N}\sum_{i=1}^{n}\alpha_i\omega_i$, where $N = \sum_{i=1}^{n}\alpha_i$, α_i is the interaction frequency of twin network client i, and n is the number of twin network clients. After getting the new round aggregated parameters, the federated parameter controller will send new parameters to all client to update the decision models. These steps will be iterated until the routing performance of the *mater model* satisfies the requirements. We would like to explain here, in contrast to distributed training, which divides the complete dataset into independent and homogeneously distributed subsets, we regard the way that each client generates traffic internally on the simulator to individually train the model as a federated learning-based approach.

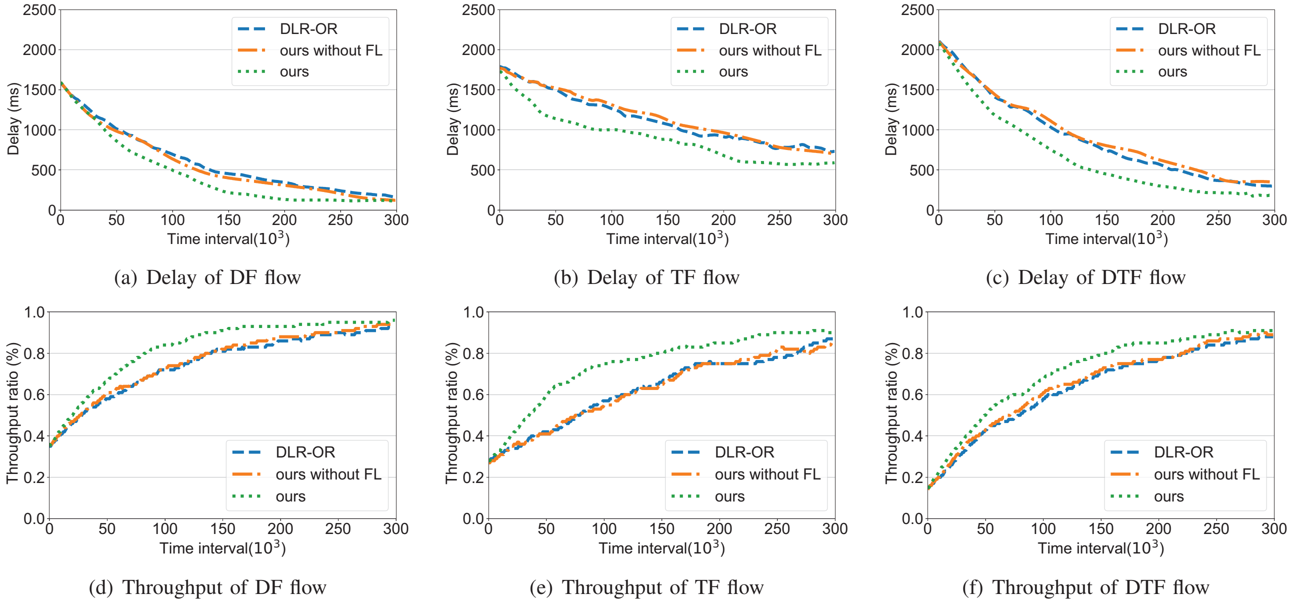
Simulation of networks presents much more diversified traffic and faster state update than real-world networks. Leveraged by multiple *twin models* to obtain traffic features and network state more quickly and extensively, the *mater model* can be updated more effectively than it based simply on interactions with real-world networks. And the *master model* can reach the stable reconvergence state efficiently. Finally, it is possible to alleviate the impact of inaccurate, even unavailable, decisions caused by dynamic changes in the network.

V. EVALUATION

A. Setup

Based on the real-world network topology *AbileneTM*, which has 11 forwarding nodes and 14 links, we leverage Ryu (a Python-based network controller), Mininet (a common network simulation platform), and the proposed reinforcement learning algorithm which is implemented by PyTorch to construct the experiments. In our experiments, we set up three transmission types, delay-first (DF), throughput-first (TF), and delay-throughput-first (DTF), corresponding to web services, file downloads, and streaming media services, respectively. In *master*, we scale the link rate and the send rate of the flows. The general link rate is 10 Mbps, the specialized bottleneck link rate is 2 Mbps, and the send rates of flows of three types are 0.1 Mbps, 1.5Mbps, and 1.5Mbps. The delay of all links is 5 ms. Correspondingly, the employed 3 *twin*s are also scaled, whose parameters will be federated per 10^3 time intervals. The three types of flows, DF (35%), TF (35%), and DTF (30%), are generated proportionally, and the corresponding source and destination nodes of each flow are specified randomly on each network individually.

Moreover, although SOHO-FL can cope with multiple topological changes by triggering the acceleration module multiple times, to better demonstrate the experimental results, it is assumed that the current network state can remain stable in topology for a period of time until this reconvergence is completed.



(a) Delay of DF flow

(b) Delay of TF flow

(c) Delay of DTF flow

(d) Throughput of DF flow

(e) Throughput of TF flow

(f) Throughput of DTF flow

Fig. 4. Convergence Performance

B. Routing policies performance

The performance is evaluated by two metrics: one is the delay, i.e., the flow completion time; the other is the throughput, which is converted into a rate format for easier comparison, i.e., the throughput divided by the maximum rate demand of the flow. We select two classical routing strategies as comparison algorithms: the first is a traditional graph-based intradomain routing strategy, the shortest path first (SPF) algorithm, which selects the path with the least forwarding hops and overlooks links' states; the other is a state-of-the-art learning-based algorithm, DRL-OR [5].

The experiment results are shown in Table I, which indicates that SOHO can fully utilize network resources and provide satisfactory transmission performance for multi-type flows. For this set of comparison experiments, we aim to state that the SOHO-FL can outperform the SPF and provide comparable performances to the state-of-the-art learning-based algorithm. Then, in this context, we further demonstrate the convergence improvements of SOHO-FL in the next subsection. Additionally, as for the performance enhancements in Fig. 4(b) (4(d)), it is because that the integrated optimization for all types of flows enables the network to transfer all flows more appropriately and efficiently, so DF (TF) flows can also be improved in throughput (delay) aspect ultimately.

C. Convergence improvement

The target of the RL model is to converge to the state where it can consistently provide satisfied routing policies. Given that, we specify that the convergence of the model is represented by the routing performance of its routing policies, which is also consistent with some existing works [5], [6]. That is, the better the model's routing decision performance over a period of time interval, the more it converges.

To demonstrate the convergence of the models, we compared SOHO-FL and DRL-OR, where SOHO-FL featured two architectures, one is the basic model (SOHO) and the other is equipped with FL acceleration. All three models are pre-trained based on SPF, and we separately measured the routing decisions performance of three types of flows during model convergence.

Fig. 4(a) to Fig. 4(c) illustrate the statistics about the delays of DF, TF, and DTF type flows. With the acceleration of FL, SOHO-FL can reach the convergence state more quickly. During the convergence process, compared to DRL-OR (SOHO), SOHO-FL reduces the delay of DF, TF, and DTF type flows by 24.3% (19.4%), 19.4% (20.5%), and 24.5% (27.6%), respectively.

Along the same lines, Fig. 4(d) to Fig. 4(f) depict the performance of the three types of flows regarding throughput during the convergence process. The results are represented in rate form. Since the DF flows are set up with a small volume, the routing decision can obtain a promising throughput rate as the utilization of the network improves, even without specified optimization for DF flows. Correspondingly, SOHO-FL improves the throughput over DRL-OR (SOHO) for DF, TF, and DTF type flows by 11.7% (10.6%), 21.4% (19.2%), and 16.8% (13.0%).

D. Additional performance analysis

In this subsection, we demonstrate the performance of the proposed scheme from a model perspective.

The number of parameters of SOHO is proportional to its layer scale, and in this experiment, the model has 0.06M parameters. At this parameter scale, the time overhead of federating the parameters of the two twin models is about 0.2 ms. The time complexity of the total federation is $O(logN)$, where N is the number of twin models. It is worth noting that the federated process can be performed in parallel with the model training. Additionally, each model takes about 0.8 ms to infer the routing policy given the input.

VI. Conclusion and Discussion

Reinforcement learning, as a promising solution to cope with increasingly complex transmission requirements, can provide sophisticated routing decisions, however, its overall performance will be affected by network topological changes. To this end, we proposed an FL-assisted semi-offline hybrid online RL-based routing architecture, SOHO-FL, whose novel neural network structure has high flexibility for multi-transmission types and network changes. In addition, SOHO-FL can alleviate the impact of network topological changes by accelerating the model reconvergence process. The experimental results demonstrate that SOHO-FL achieves 22.3% improvement in reconvergence time over the state-of-the-art approaches in average.

The proposed FL-assisted acceleration scheme can be applied not only to the reconvergence scenario of the specified RL-based routing model in this paper, but its idea can also be employed to facilitate other learning-based models' convergence/reconvergence or improve the accuracy of routing policy by merging additional experience simulated by digital twin networks. Moreover, such performance improvements are based on basic federation mechanisms, which can be further boosted by more elaborate tuning strategies. In the future, we will continue to optimize SOHO-FL in terms of the more flexible neural network structure, and the efficiency or privacy issue of federation strategy, and conduct more experiments to demonstrate the performance improvements.

Acknowledgement

The work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62072047 and 62172054, the National Key R&D Program of China under Grant 2019YFB1802603, and the Key Project of Beijing Natural Science Foundation under M21030. Peizhuang Cong's work was supported in part by BUPT Excellent Ph.D. Students Foundation under Grant CX2021232.

References

[1] N. Kato, Z. M. Fadlullah, B. Mao, F. Tang, O. Akashi, T. Inoue, and K. Mizutani, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE wireless communications*, vol. 24, no. 3, pp. 146–153, 2016.

[2] B. Mao, Z. M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE Transactions on Computers*, vol. 66, no. 11, pp. 1946–1960, 2017.

[3] Z. Zhuang, J. Wang, Q. Qi, H. Sun, and J. Liao, "Graph-aware deep learning based intelligent routing strategy," in *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*. IEEE, 2018, pp. 441–444.

[4] X. Zuo, J. Yang, M. Wang, and Y. Cui, "Adaptive bitrate with user-level qoe preference for video streaming," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1279–1288.

[5] C. Liu, M. Xu, Y. Yang, and N. Geng, "Drl-or: Deep reinforcement learning-based online routing for multi-type service requirements," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[6] P. Cong, Y. Zhang, Z. Liu, T. Baker, H. Tawfik, W. Wang, K. Xu, R. Li, and F. Li, "A deep reinforcement learning-based multi-optimality routing scheme for dynamic iot networks," *Computer Networks*, vol. 192, p. 108057, 2021.

[7] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," in *IEEE INFOCOM 2018-IEEE conference on computer communications*. IEEE, 2018, pp. 1871–1879.

[8] L. Hui, M. Wang, L. Zhang, L. Lu, and Y. Cui, "Digital twin for networking: A data-driven performance modeling perspective," *IEEE Network*, pp. 1–8, 2022.

[9] C. Nadiger, A. Kumar, and S. Abdelhak, "Federated reinforcement learning for fast personalization," in *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. IEEE, 2019, pp. 123–127.

[10] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Generation Computer Systems*, vol. 115, pp. 619–640, 2021.

[11] X. Li, M. JIANG, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-iid features via local batch normalization," in *International Conference on Learning Representations*, 2020.

[12] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 8, pp. 1754–1766, 2020.

[13] V. Balasubramanian, M. Aloqaily, M. Reisslein, and A. Scaglione, "Intelligent resource management at the edge for ubiquitous iot: an sdn-based federated learning approach," *IEEE network*, vol. 35, no. 5, pp. 114–121, 2021.

[14] B. Yang, H. Shi, and X. Xia, "Federated imitation learning for uav swarm coordination in urban traffic monitoring," *IEEE Transactions on Industrial Informatics*, 2022.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

Peizhuang Cong is currently a Ph.D. student in State Key Laboratory of Network and Switching Technology, Beijing University of Posts and Telecommunications (BUPT), Beijing, China. He has published papers in IEEE JSAC, IEEE/ACM IWQoS and so on. His research interests include the next generation network architecture, data driven networks, routing protocols 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. She is now an Associate Professor in Beijing University of Posts and Telecommunications, and a Visiting Scholar in the University of Cambridge, where she is also a Research Associate in the Wolfson College. Her research interests include large scale datacenter networks, federated learning, data-driven networks and edge computing. She is a member of IEEE and ACM.

Wendong Wang (M'05) 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 research interests are the next generation network architecture, network resources management and QoS, and mobile Internet. He is a member of IEEE.

Ke Xu (M'02-SM'09) 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.