scientific reports

Check for updates

OPEN

Federated deep reinforcement learning-based urban traffic signal optimal control

Mi Li[1,2]✉, Xiaolong Pan[3], Chuhui Liu[1] & Zirui Li[4]

This paper proposes a cross-domain intelligent traffic signal control method based on federated Proximal-Policy Optimization (PPO) for distributed joint training of agents across domains for typical intersections, aiming at solving the problems of slow learning speed and poor model generalization when deep reinforcement learning (RL) is applied to cross-domain multi-intersection traffic signal optimization control. The proposed method improves the model generalization ability of different local models during global cross-region distributed joint training under the premise of ensuring information security and data privacy, solves the problem of non-independent and homogeneous distribution of environmental data faced by different agents in real intersection scenarios, and significantly accelerates the convergence speed of the model training phase. By reasonably designing the state, action and reward functions and determining the optimal values of several key parameters in the federated collaboration mechanism, the RL model could ensure high learning efficiency and fast convergence in the face of the gradual increase of road network size and the exponential increase of state and action space with the number of intersections. In addition, the new state interaction method and the reward function allow the agents to collaborate with each other, which greatly improves the information interaction efficiency between the federated learning local agents and the central coordinator, and improves the access efficiency of the road network and reduces the amount of communication data transmitted. Finally, through experimental comparisons, the proposed method can significantly reduce the average vehicle waiting time by up to 27.34% compared with the existing fixed timing method, and under the same convergence height, the convergence speed is up to 47.69% faster compared with the individual PPO trained in a single local environment, and up to 45.35% faster than the aggregated PPO trained jointly using all local data. The proposed method effectively optimizes intersection access efficiency with excellent robustness under various traffic flow settings.

Keywords Federated deep reinforcement learning, Optimal control of urban traffic signals, Federated proximal-policy optimization, Multi-agents, Distribution

To address the increasing conflict between growing traffic demand and limited transportation resources, intelligent traffic signal control is crucial for mitigating congestion in smart transportation systems. Efficiently utilizing existing traffic infrastructure through optimized signal control can enhance the capacity of critical intersections and reduce congestion on main roads via traffic path guidance and load distribution. Research[1] indicates that delays caused by inefficient traffic signal control at intersections account for 5% to 10% of total urban traffic delays. Effective signal control algorithms not only alleviate road congestion and enhance traffic flow efficiency but also play a positive role in reducing traffic accidents, mitigating environmental pollution, and improving traffic management capabilities[2]. Therefore, in-depth research on traffic signal control is essential for constructing a more comprehensive, scientific, and intelligent system to better address varying traffic demands.

Most current intelligent signal control methods require the establishment of specific mathematical models, followed by solving these models using techniques such as dynamic programming, fuzzy control, and cooperative game theory[3]. While these methods improve traffic flow efficiency to some extent, they demand substantial computational resources in complex scenarios, making real-world application challenging. The advancement of reinforcement learning (RL) and deep reinforcement learning (DRL) aligns well with the complexities of traffic

[1]College of Information Science and Engineering, Jiaxing University, Jiaxing 314000, China. [2]Jiangsu Yikong Intelligent Equipment Co., Ltd, Nantong 226000, China. [3]Faculty of Business, The Hong Kong Polytechnic University, Hong Kong, China. [4]School of Energy and Architecture, Xi'an Aeronautical University, Xi'an 710000, China. ✉email: limi@zjxu.edu.cn

environments, offering model-free approaches that have garnered significant attention in the field of traffic signal optimization[3–9].

The inherent nature of traffic signal control problems makes them well-suited for RL methods due to their dynamic, uncertain, and high-dimensional characteristics. Traffic flow fluctuates over time, influenced by factors such as time of day, weather, and incidents. RL methods, particularly DRL, excel at adapting to such dynamic environments through iterative interactions with the system. Additionally, traffic signal control aims to optimize long-term performance metrics, such as minimizing cumulative vehicle delay and maximizing throughput, which aligns with RL's focus on maximizing cumulative rewards over time. Existing studies[10,11] have shown that RL can adapt to dynamic traffic environments through interactions, effectively reducing vehicle waiting times. Moreover, RL's ability to handle complex, high-dimensional state spaces, such as multi-intersection coordination and multi-agent scenarios, further supports its application in traffic control. For instance, to tackle the curse of dimensionality, research[12] has proposed Q-Learning method[13] utilizing function approximation, yet these approaches still face storage limitations. Li et al.[14] introduced DQN-based signal control for single intersections, employing deep neural networks (DNNs) to learn the Q-function, where the state is defined as queue length and the reward function as the difference in maximum queue lengths in both north-south and east-west directions. However, this definition yielded limited improvements in vehicle waiting times. To enhance performance, Shabestary et al.[15] represented intersection states visually, detailing vehicle positions and speeds while using convolutional neural networks to extract features. The reward was defined as the change in accumulated delay, with actions involving the random selection of one among several signal phases. This approach not only complicates the definitions of states and rewards but also considers the free combination of phases[16], departing from traditional symmetrical signal phases. Such modifications have shown the potential to improve algorithm performance, albeit at the cost of increased computational time and resource consumption. In addition, some progress has been made on the DRL-based collaborative traffic signal control problem[17–20], which contributes to high-quality management of urban transportation, reduction of congestion and energy consumption. Du et al.[21] demonstrated that, during periods of relative congestion, optimal control does not necessitate overly complex states; rather, focusing on vehicles within a small area near the intersection suffices. RL-based traffic signal control methods can effectively learn control experiences from data, yet they typically require training with simulated data in software, followed by fine-tuning using real-world data. Due to the inaccuracies of simulated data in reflecting real-world conditions, the training process for RL models can be exceedingly time-consuming. Despite the remarkable performance of RL and DRL in traffic signal control, their practical applications face challenges, including limited data, low learning efficiency, slow convergence, poor model generalization, and difficulties in parameter fine-tuning[22]. Improving the training and convergence speed of RL-based signal control optimization algorithms, enhancing model generalization, reducing computational burdens from complex choices of state, action, and reward, and simplifying parameter fine-tuning represent significant challenges in the field of traffic signal control. These issues warrant further investigation and constitute the primary motivation for this study.

On the other hand, the effectiveness of training RL models is closely tied to the quality of the data used for training. When the feature space of the state is small and training data is limited, constructing high-quality model policies becomes challenging[23]. A common solution is to employ parallel or distributed training[24–27], where information exchange and data sharing among agents allow for centralized model updates using data from all agents. However, this approach places high demands on the data storage and computational capabilities of the central control unit. Furthermore, there has been an increasing focus on user privacy and data security in recent years. Centralized model training methods that involve direct sharing of local data do not meet these security requirements. Federated Learning (FL) offers a distributed machine learning framework that enhances the effectiveness of artificial intelligence models while ensuring data privacy and compliance[28]. In real world, with the rapid development of smart city and smart transportation, cases of cross-domain cooperation based on FL have appeared[29,30], which effectively addresses the conflict between data privacy and data silos. Integrating FL with RL not only facilitates information exchange while avoiding privacy breaches but also enables agents to adapt to diverse environments. Inspired by the success of FL in supervised learning tasks, researchers have increasingly explored its application to RL, leading to the development of Federated Reinforcement Learning (FRL). Combining FL with RL not only enables agents to train in varied environments without sharing their locally collected data but also enhances the security and overall performance of RL systems. FRL can be broadly categorized into two types: Horizontal Federated Reinforcement Learning (HFRL) and Vertical Federated Reinforcement Learning (VFRL), as illustrated in Fig. 1. In HFRL, agents tackle similar decision-making tasks in independent environments that do not influence each other. Each participant executes actions based on the local environment's state and receives corresponding rewards. Given that agents operate within limited state spaces, sharing experiences among multiple agents interacting with their respective environments can accelerate training and improve model performance. Agents in HFRL typically exchange parameters of policy and value function models via a client-server architecture or a peer-to-peer network. This approach addresses the sample efficiency problem inherent in RL, enabling agents to achieve optimal strategies more rapidly. Additionally, it maximizes cumulative expected rewards for specific tasks while safeguarding data privacy and security. FRL has demonstrated promising applications across various domains, including the Internet of Things, robotics, and autonomous driving, showcasing its potential to drive innovation in these fields.

In the field of robotics, a FRL algorithm based on DQN has been proposed for cloud robot system navigation[31], enabling robots to efficiently learn in new environments and rapidly adapt to changes using prior knowledge. Each robot trains a local model based on its navigation tasks, followed by knowledge fusion on a centralized cloud server to update the global model. In the Internet of Things (IoT) domain, to protect privacy, a FL architecture has been deployed between edge nodes and IoT devices for computational offloading tasks[32]. IoT devices can download RL models from edge nodes and train local models using their own data. The edge

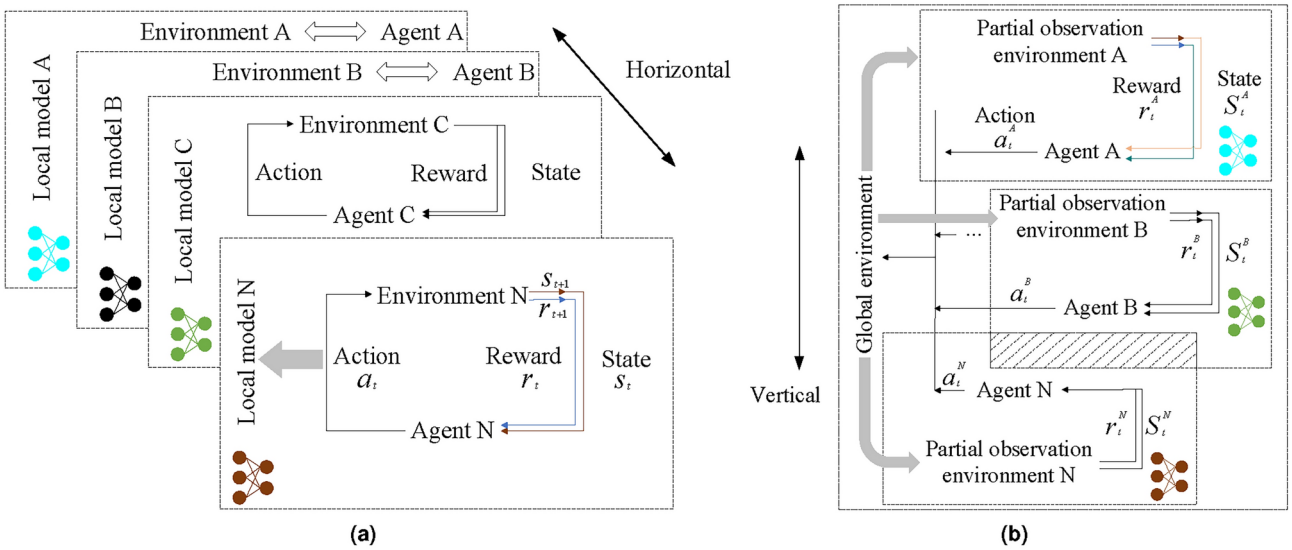

Fig. 1. Schematic comparison of HFRL and VFRL. (**a**) HFRL. (**b**) VFRL.

nodes aggregate the local models from various IoT devices to form a global model; however, the communication resource costs associated with model exchanges have not been further evaluated in the literature. Lim et al.[33] introduced a FRL method based on Proximal Policy Optimization (PPO)[34] for optimizing control of IoT devices, enhancing agent learning speed through shared gradient loss functions. In the autonomous driving sector, a FRL approach utilizing Deep Deterministic Policy Gradient (DDPG)[35] has been proposed[36], which aggregates model parameters learned by agents across multiple environments to jointly train a global model, thereby improving robustness and generality. Research on FRL in traffic signal control is limited. Ye et al.[37] introduced a novel FRL method called FedLight, where each intersection in a road network employs the Advantage Actor-Critic (A2C) algorithm for signal control. Agents engage in joint learning through gradient aggregation and parameter sharing without transmitting local data. This study claims that federated multi-agent RL achieves faster convergence compared to existing multi-agent methods. However, it primarily addresses the cooperation between multiple intersections within a single road network environment and does not consider the data privacy issues that arise when different regions utilize their own local data for joint model training. This challenge represents a significant hurdle for regional cooperation in real-world applications. Therefore, it is crucial to leverage federated deep reinforcement learning (FDRL) to enhance model training convergence and robustness while addressing the challenges posed by non-independent and identically distributed (non-IID) data. Establishing specific federated collaboration mechanisms and aggregation algorithms that ensure information security and data privacy while improving model generalization is of great significance, which serves as the main motivation of this paper.

The objective of this study is to develop a FRL-based cross-domain intelligent traffic signal control architecture, establishing an effective and practical adaptive traffic signal optimization control system to enhance intersection and network throughput while reducing road congestion. The contributions of this paper are summarized as follows:

- A FRL-based cross-domain intelligent traffic signal control architecture and collaborative mechanism are proposed to address the non-IID environmental data issues faced by different agents in real-world scenarios. This framework enhances the generalization capability of various local models during global distributed joint training while ensuring information security and data privacy, significantly accelerating the convergence speed during the model training phase.
- By carefully designing the state, action, and reward functions, as well as determining the optimal values for several key parameters in the federated collaboration mechanism, a federated PPO-based traffic signal control method is introduced. This approach effectively resolves the challenges posed by slow convergence and low learning efficiency in RL models as the road network size increases, leading to an exponential growth in state and action spaces with the number of intersections. Compared to existing RL-based signal optimization methods, the proposed traffic signal control approach demonstrates a faster response speed while maintaining similar convergence levels.
- The study introduces a well-designed state interaction method and reward function that facilitate cooperation among agents, significantly improving the information exchange efficiency between local agents and the central coordinator in FL. This enhancement not only boosts the efficiency of road network throughput but also reduces the volume of data transmitted during communication. Additionally, the specific configurations of local environments in FL are optimized, enabling all local agents to adapt to varying conditions and thoroughly explore the state space.

Intelligent traffic signal control architecture based on FDRL

In the field of intelligent traffic signal control, RL effectively optimizes traffic signals but heavily relies on diverse environmental data. The FL framework ensures data security and privacy while addressing the substantial data requirements of RL. Therefore, integrating FL with RL architectures to enhance model performance represents a viable approach with significant application potential and value. This paper proposes a cross-domain intelligent traffic signal control architecture based on FRL to address the challenges associated with data silos and data protection. This architecture facilitates collaborative traffic signal optimization across cities and regions while enhancing the performance of RL agents.

Federal deep reinforcement learning collaborative

The proposed cross-domain intelligent traffic signal control architecture based on FRL is illustrated in Fig. 2. In this paper, the term 'cross-domain' refers to the ability to perform distributed joint training of traffic signal control models across different intersection domains or environments, where each domain may have distinct traffic flow characteristics, infrastructure setups, and environmental conditions. The goal is to enable the optimization of traffic signal controls by leveraging knowledge shared across these diverse domains while ensuring data privacy and security through federated learning mechanisms. Specifically, in each environment, DRL agents are deployed to optimize traffic signals. These agents can operate as single or multiple entities, employing either centralized or distributed control. Although the environments share the same intersection layout-depicted as a "grid" pattern of four-way intersections-they may exhibit subtle differences, such as varying distances between intersections. This setup satisfies the requirements for horizontal FDRL: (1) The tasks assigned to agents in each environment are similar; (2) Each environment operates independently, without influencing one another; (3) Local data obtained by the agents are not transmitted or shared. The primary objective of the local agents in each environment is to identify the optimal signal control strategy that minimizes the average travel time of all vehicles within the network. The neural network architecture of the agents remains consistent with the global model. The detailed collaborative process of FDRL is outlined as follows:

- Step 1: The federated central coordinator initializes the global model and broadcasts it to the agents in each local environment.

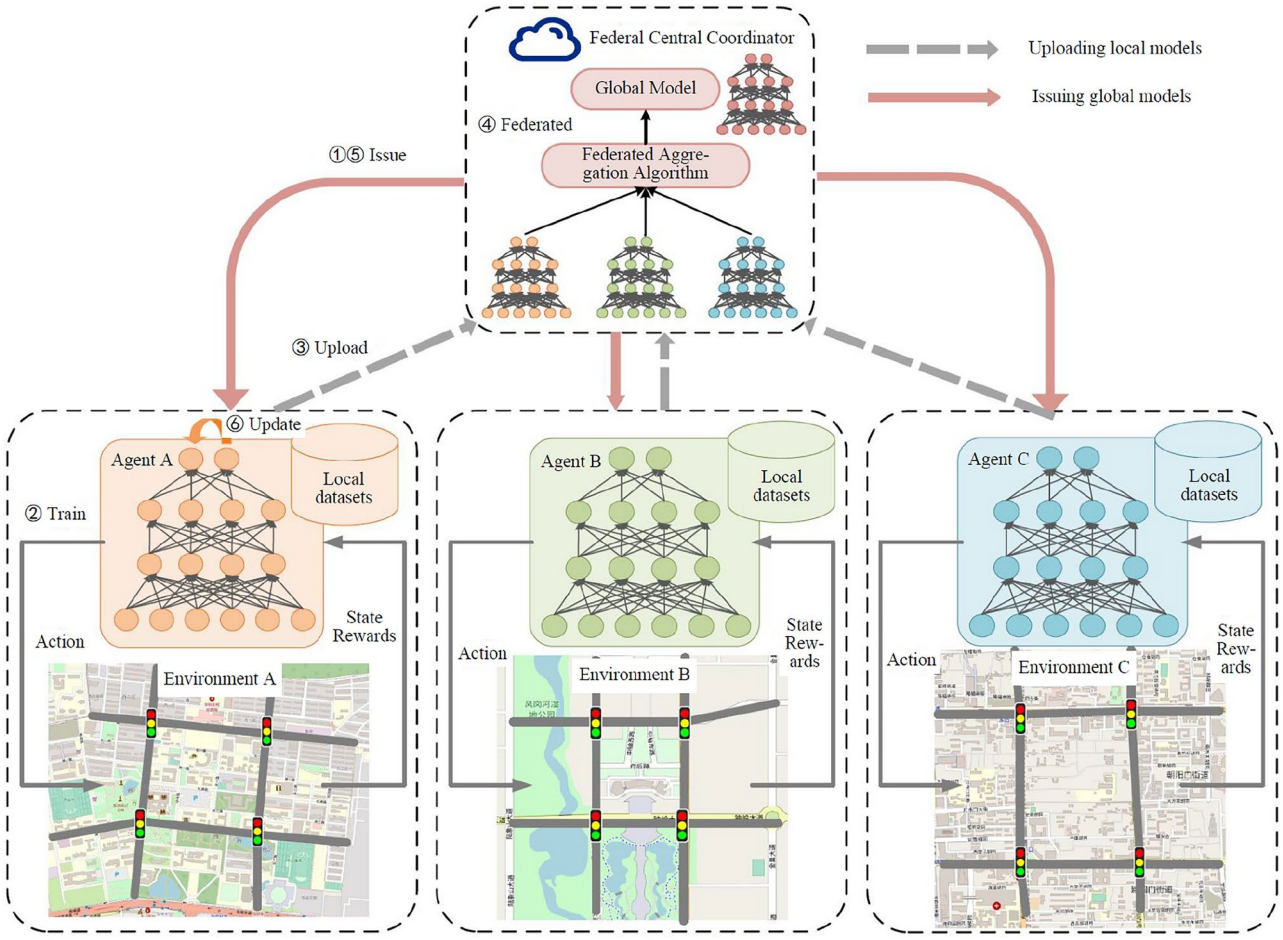

Fig. 2. FDR based cross-domain intelligent traffic signal control architecture.

- Step 2: Agents independently observe the state of their environment and make decisions based on local states. After executing an action, the environment provides feedback in the form of reward values and new states to the agents, which then collect data and update their local model parameters.
- Step 3: After every K local model updates, agents upload their local model parameters to the federated central coordinator. They may apply encryption to these parameters; even without encryption, only the neural network parameters are transmitted, thus preserving sensitive data from the environments.
- Step 4: The federated central coordinator processes the uploaded models using a specific aggregation algorithm to generate the global model. The coordinator does not need to wait for all agents to submit their model parameters and can aggregate a subset of local model parameters based on the situation.
- Step 5: The federated central coordinator distributes the global model to all local agents.
- Step 6: Agents update their local models to align with the global model and may also create personalized local models to derive new local model parameters.
- Step 7: Steps 2 to 6 are repeated until the model converges, the maximum number of iterations is reached, or the longest training time is achieved.

Remark 1 It should be noted that the main challenges faced when applying DRL to traffic signal control in cross-domain scenarios include: (1) Heterogeneous Data: Each intersection may have its own unique traffic flow patterns, road network configurations, and sensor setups, making the data collected from each environment non-IID; (2) Generalization: It is difficult for RL models trained in one domain to generalize effectively to other domains, particularly when data cannot be directly shared; (3) Data Privacy: Ensuring that sensitive traffic data (such as vehicle counts and flow rates) is not exchanged during the training process while still allowing for effective model learning. It is worth emphasizing that the proposed methodology and framework in this paper can effectively address the requirements for an efficient and secure cross-domain learning mechanism that not only enables model generalization across heterogeneous traffic environments but also safeguards data privacy during the training process.

Federated aggregation algorithm

The core function of the federated central coordinator is to aggregate the uploaded local model parameters. The most commonly used aggregation algorithm in FL is the Federated Averaging algorithm proposed by Brendan McMahan et al.[38] at Google. This algorithm is applicable to non-convex loss functions in deep neural network training and is suitable for any finite summation form of the following loss function:

$$\min_{w \in R^d} f(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w) \tag{1}$$

where n represents the number of training data points and $w \in R^d$ denotes the parameters of the deep neural network in d dimensions. In this study, the Federated Averaging algorithm is selected as the aggregation method for the federated central coordinator, as described below.

Assuming there are N agents from different environments participating in the federated model aggregation process, the local model parameters of these agents are represented as $W_t^{(i)}, i = 1, 2, \ldots, N$. Each agent in the local environment can only access the state of its environment and uses local data for model training. The network parameters of the global model maintained by the central coordinator are denoted as $W_t^{(g)}$. The most basic aggregation algorithm can be expressed as:

$$W_{t+1}^{(g)} = \sum_{i=1}^{N} p_i \cdot W_t^{(i)} \tag{2}$$

This represents the direct weighted average of all local models participating in the current round of federated aggregation. Building on this, if the update method for the global model is modified to soft updates, the aggregation algorithm can be expressed as:

$$W_{t+1}^{(g)} = \alpha \cdot W_t^{(g)} + (1 - \alpha) \sum_{i=1}^{N} p_i \cdot W_t^{(i)} \tag{3a}$$

$$\sum_{i=1}^{N} p_i = 1, p_i \geqslant 0 \tag{3b}$$

In this context, α is referred to as the FL rate. When $\alpha = 0$, Eq. (3a) reduces to Eq. (2); when $\alpha = 1$, it indicates that the local model parameters are not incorporated into the global model, rendering the current round of federated aggregation ineffective. Therefore, in general, α should be kept relatively small. The term p_i represents the federated aggregation weight, indicating the influence of each participant on the global model during the aggregation process. Typically, p_i is distributed evenly, i.e., $p_i = \frac{1}{N}$. However, p_i can also be flexibly allocated based on the performance of the agents during training, with the aim of maximizing the performance of the global model and accelerating its convergence.

Remark 2 Compared to traditional DRL algorithms trained in a single environment, FDRL algorithms leverage the FL framework to address the limitations of conventional methods. The primary advantages of FDRL includes: (1) Accelerated training speed; (2) Enhanced generalization and stability; (3) Privacy protection and data security. Specifically, by enabling multiple agents to interact with multiple environments under similar environmental models and target tasks, FDRL facilitates local data storage while sharing experiences to expedite the learning process. This approach requires significantly fewer computational resources compared to aggregating updates using all local data; As the scale of a road network or environment expands, the state space becomes increasingly challenging to fully explore. Training with a single agent often fails to deliver optimal decisions in rare state scenarios. Horizontal FDRL allows agents to collaborate during training, leading to improved performance in low-probability states; FDRL eliminates the need for participants to directly share raw local data. Instead, encrypted model parameters are exchanged, significantly reducing the risk of reconstructing the original local data even in the event of parameter leakage. Compared to other distributed and parallel methods, FDRL ensures secure and legally compliant information exchange while maintaining data privacy.

Problem analysis

A single intersection is the smallest control unit within a traffic signal control system and is also the most common scenario in practical signal control problems. As road network scales expand, it becomes essential for individual intersections to coordinate their signal control with neighboring intersections to effectively improve the overall network traffic efficiency. Thus, optimizing signal control at the single intersection level forms the foundation of traffic signal optimization for the entire network. The objective of this study is to develop a real-time adaptive signal control system for a single intersection, utilizing a Federated PPO algorithm. This system will make real-time control decisions for traffic signals based on the current state of the intersection.

Description of the environment

The subject of this study is a typical intersection, as shown in Fig. 3. The intersection has four directions: north, south, east, and west, with two lanes in each direction. The outer lane is designated for straight-through traffic, and the inner lane is for left turns. These eight incoming lanes are labeled as $\{l_1, l_2, ..., l_8\}$. The intersection operates with four distinct signal phases: north-south through, north-south left-turn, east-west through, and east-west left-turn, corresponding to Phases 1 through 4, respectively. The traffic lights have three states: red, indicating a stop; green, indicating passage; and yellow, signaling caution. By default, the traffic lights cycle through the phases in the order Phase 1 → Phase 2 → Phase 3 → Phase 4 → Phase 1, with a yellow light period between phase transitions.

The traffic signal control problem at the intersection can be modeled as a Markov Decision Process (MDP), represented by a five-tuple $\langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, P the state transition probabilities, R the reward function, and γ the discount factor. As illustrated in Fig. 4, when using DRL algorithms for signal control, it is assumed that each signal phase lasts for ΔT time steps, with each time step corresponding to one second in real time. At the beginning of each phase, the agent receives the current state of the intersection and selects an action. At the end of the phase, the agent receives a reward from the

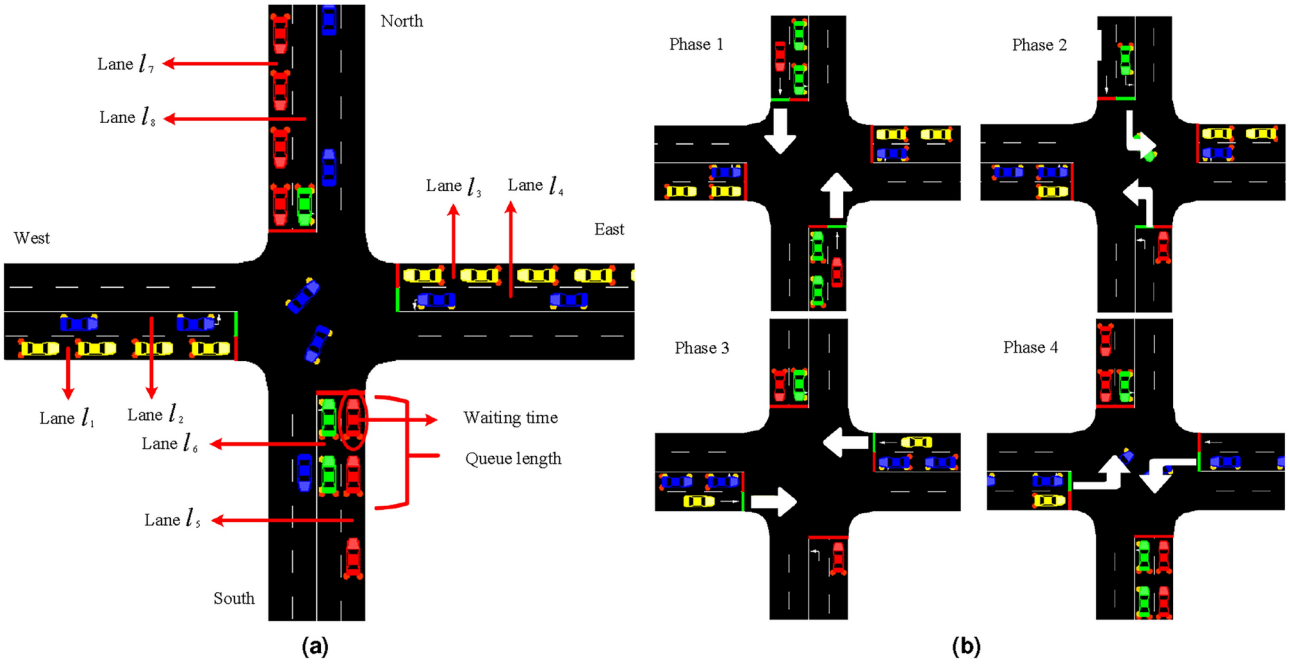

Fig. 3. Schematic diagram for a single intersection and its four phases. (**a**) Typical cross intersection schematic. (**b**) Single intersection four-phase schematic.

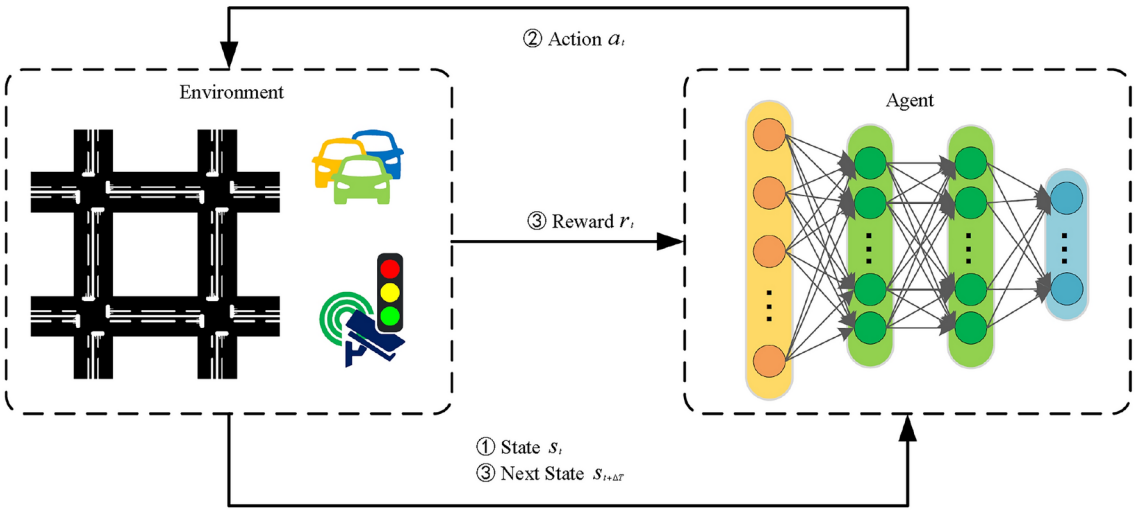

Fig. 4. Schematic of traffic signal control based on DRL.

environment and transitions to the next signal phase. The agent can then update its action selection policy based on the received reward, which corresponds to adjusting the parameters of the agent's neural network. After a period of continuous learning and network updates, the agent will eventually learn the optimal action policy, thereby improving the traffic flow efficiency of the intersection.

Definitions of states, actions, and rewards

This section provides a detailed description of the state and action definitions, as well as the reward function, used in the DRL signal control algorithm for a single intersection.

(1) State Space

The state represents the agent's perception and abstract expression of the environment at a given time step. Previous studies commonly used state representations such as queue length, accumulated waiting time, average vehicle speed, and vehicle count. More complex representations even involve using the positions of vehicles on each lane as matrix inputs[39]. Although these complex states can be obtained with advancements in sensor and vehicular network technologies, they are difficult to collect and require substantial resources. In this study, the state at time step *t*, denoted as s_t, is defined as a 16-dimensional vector: $s_t = \{\text{Queuelength}_t[l], \text{Waitingtime}_t[l]\}$ where $\text{Queuelength}_t[l]$ refers to the vehicle queue length on lane *l* at time step *t*, and $\text{Waitingtime}_t[l]$ refers to the accumulated waiting time of the first vehicle in lane *l* at time step *t*, for $l = l_1, l_2, \ldots, l_8$. The queue length can be measured using sensors deployed on the lanes, while the waiting time can be gathered through vehicular networks. Both types of data are easily accessible and of low complexity, providing a solid foundation for future expansion to road networks.

(2) Action Space

An action is the behavior that the agent may take. In traffic signal control, the agent at the intersection must make appropriate decisions based on the current state of the environment, such as selecting the appropriate signal phase, setting the duration of the current phase, adjusting the green-to-red ratio, or maintaining the current phase/transitioning to the next phase, in order to manage vehicle flow. In this study, the action space consists of the four phases at the intersection. At the start of each phase, at time step *t*, the agent selects one of the four phases shown in Fig. 3b based on the current state s_t.

$$a_t = \begin{cases} \text{North-South straight ahead green}, & \text{phase} = 1 \\ \text{North-South left turn green}, & \text{phase} = 2 \\ \text{East-West straight ahead green}, & \text{phase} = 3 \\ \text{East-West left turn green}, & \text{phase} = 4 \end{cases} \tag{4}$$

In this study, the DRL algorithm allows the agent to switch between phases freely, without necessarily adhering to traditional cyclic phase control. To ensure safe vehicle passage during phase transitions, when the current phase differs from the selected next phase, the system first activates a yellow light for a duration of T_y seconds to signal an impending phase change. Afterward, the signal switches to the selected phase and remains active for $\Delta T - T_y$ seconds. If the current phase and the selected next phase are the same, the phase remains unchanged, and the system continues for the full ΔT seconds.

(3) Reward Function

The reward represents the feedback received from the environment after the agent performs an action. It serves to evaluate the quality of the action. Typically, a positive reward indicates a beneficial outcome, while a negative reward signals a detrimental effect. Based on the received rewards, the agent updates its model to enhance decision-making and maximize long-term rewards. In traffic signal control, the ultimate goal is to

minimize vehicle travel time as much as possible. However, vehicle travel time is difficult to use directly as a reward in DRL for the following reasons: First, vehicle travel time is influenced not only by traffic signals but also by factors such as maximum vehicle speed and route length. Second, in the real world, traffic signal controllers cannot predict the vehicle's final destination in advance, making it challenging to optimize travel time. Existing studies often define the reward function as a weighted sum of multiple indicators (e.g., queue length, waiting time, vehicle speed, intersection pressure, etc.). However, there is no theoretical method for setting the optimal weights for these values to ensure that the reward accurately reflects the quality of the actions. Furthermore, traffic evaluation metrics are often interrelated. For example, with the same route length, shorter waiting times generally lead to shorter vehicle travel times.

Considering the factors mentioned above, the reward in this study is defined as the change in the average vehicle waiting time before and after selecting an action. The reward function at time step t is defined as follows:

$$r_t = W_{t-\Delta T} - W_t \tag{5}$$

where $W_{t-\Delta T}$ represents the average waiting time of all vehicles on the road at time step t, calculated as:

$$W_t = \frac{1}{N_t} \sum_{i_t=1}^{N_t} w_{i_t} \tag{6}$$

Here, N_t denotes the total number of vehicles in the network at time step t, and w_{i_t} represents the accumulated waiting time of the i-th vehicle from its entry into the network until time step t.

Federated PPO-based signal control algorithm design for single intersection

The procedure for the single intersection signal control algorithm based on PPO is outlined in Algorithm 1. Initially, the agent model and experience buffer are initialized, and the environment is set up at the start of each training episode. Every ΔT seconds, an action is selected, after which the environment provides a reward to the agent. The agent then updates its model using the collected tuple. This process is repeated in a loop until the final model is trained. After the agent's training is complete, only the interaction between the Actor and the environment is required, where the agent selects actions based on the current state. This corresponds to lines 6 to 14 in the algorithm.

1 Initialize the agent model, including the Actor network $\pi_\theta(a \mid s)$, the old Actor network $\pi_{\theta_{\text{old}}}(a \mid s)$, and the Critic network $V_\omega(s)$;
2 Initialize the experience buffer $\mathscr{D}$;
3 Set training steps $\leftarrow 0$;
4 **for** *epoch* $= 1, 2, 3, \ldots$ **do**
5 Initialize the environment, obtain the initial state s_0 and the initial intersection signal phase a_0;
6 **for** *time step* $t = 1, 2, 3, \ldots, T$ **do**
7 The agent selects an action $a_t \sim \pi_\theta(\cdot \mid s_t)$ based on the current state at time step t;
8 **if** $a_t =$ *the current signal phase* **then**
9 Execute the action a_t and maintain the phase for ΔT seconds;
10 **else**
11 Set the intersection phase to the yellow light phase and maintain it for T_y seconds;
12 Switch the intersection phase to a_t, maintaining it for $\Delta T - T_y$ seconds;
13 **end**
14 Transition to the new traffic state $s_{t+\Delta T}$, obtain the average waiting time of all vehicles on the road;
15 Compute the reward r_t;
16 Store the tuple $\langle s_t, a_t, r_t, s_{t+\Delta T} \rangle$ in the experience buffer $\mathscr{D}$;
17 Randomly sample N tuples from $\mathscr{D}$ to form a minibatch;
18 Update the Actor network parameters using the objective function:
 $L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}\left(r_t(\theta), 1-\varepsilon, 1+\varepsilon\right)\hat{A}_t \right) \right]$, where $\hat{A}_t = r_t + \gamma V_\omega(s_{t+1}) - V_w(s_t)$;
19 Update the Critic network parameters using the objective function: $L_{\text{critic}}(\omega) = \left(r_t + \gamma V_\omega(s_{t+1}) - V_\omega(s_t)\right)^2$;
20 Copy the Actor network to the old Actor: $\pi_{\theta_{\text{old}}}(a \mid s) \leftarrow \pi_\theta(a \mid s)$;
21 Set $t \leftarrow t + \Delta T$;
22 Increment the training step: training steps $\leftarrow$ training steps + 1;
23 **end**
24 **end**

Algorithm 1. PPO-based signal control algorithm for single intersection.

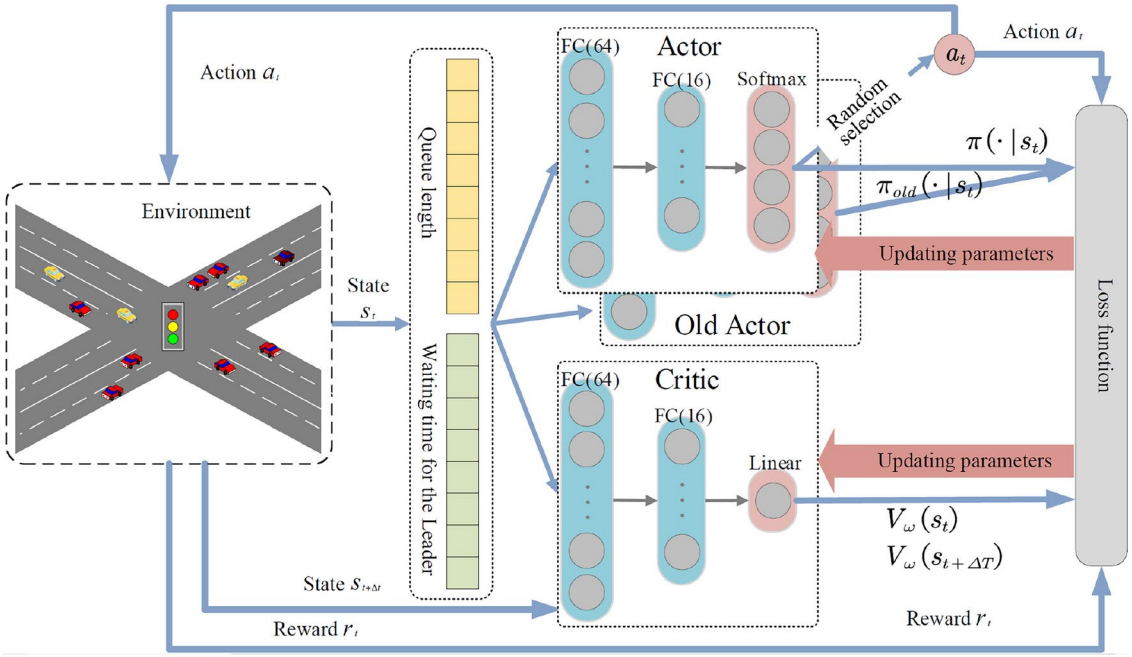

Fig. 5. Schematic diagram of PPO-based signal control algorithm for single intersection.

Fig. 5 illustrates the detailed process of single intersection signal control based on the PPO algorithm. In this model, both the Actor and Old Actor share an identical neural network architecture, which includes an input layer with 16 neurons, two fully connected hidden layers with 64 and 16 neurons, respectively, and a Softmax-activated output layer that produces a 4-dimensional vector. This output represents the probability values for all possible actions, summing to 1. For the Critic, the input and hidden layers are identical to those of the Actor, except that its output layer generates a single scalar value, representing the action evaluation score.

A FL framework is then incorporated into the algorithm, where the set of all local environments is denoted as *Env*. The specific process of the federated parameter aggregation method used in this study is shown in Algorithm 2. Each local agent within an environment can perform local learning and updates. When the training steps reach K or the time steps of the current episode reach the maximum T, agents are, by default, included in the federated update (though they may also independently choose whether to participate in the update based on the situation). Additionally, this section introduces a DRL method utilizing data from all local environments for collaborative training, detailed in Algorithm 3. In this approach, a central agent can access the states, actions, and rewards of all local agents, performing centralized aggregation and updates every K training steps.

Remark 3 It is worth pointing out that compared with the industry's popular centralized training distributed execution (CTDE) framework[40], which is widely used in distributed RL scenarios, the proposed method in this paper is significantly superior in these aspects, including (1) the proposed method leverages FRL, where agents collaborate on training across domains without exchanging raw data, thereby avoiding the need for raw data exchange and enhancing data privacy and security; (2) the proposed method allows global model aggregation by introducing a federated collaboration mechanism, while enabling local models to preserve domain-specific nuances. As a result, the approach achieves robust handling of non-IID environment data and ensures better generalization across various transportation scenarios; (3) the proposed approach reduces communication overhead by transmitting only model parameters instead of raw environment data. In addition, the designed efficient state interaction mechanism and reward function optimize the communication process and improve the communication efficiency and scalability while ensuring the model performance, making it suitable for large-scale traffic signal control systems; (4) Subsequent experimental validation also shows that the proposed method has excellent performance in terms of convergence speed, stability and optimization effectiveness.

Central Coordinator: At $t = 0$, initialize the global model parameters $W_0^{(g)}$ randomly, including the Actor network $\pi_{\theta^{(g)}}(a \mid s)$, $\pi_{\theta_{\text{old}}^{(g)}}(a \mid s)$ and the Critic network $V_{\omega^{(g)}}(s)$;
Local Agents in Each Environment:
Initialize local models $W_0^{(i)}$ for each environment $i \in Env$;
Download the global model from the central coordinator and set $W_0^{(i)} = W_0^{(g)}$ for all $i \in Env$;
for *time step* $t = 0$ *to* T **do**
  Local Agents in Each Environment:
  for *In parallel, each local agent in environment* $i \in Env$ **do**
    Download the global model $W_t^{(g)}$ from the central coordinator and set $W_t^{(i)} = W_t^{(g)}$;
    Update local model parameters $W_t^{(i)}$ based on local data, as detailed in Algorithm 1;
    if *the local training step count reaches* $= K$ *or the local time step* $t = T$ **then**
      Send the current local model parameters $W_t^{(i)}$ to the federated central coordinator;
    end
  end
  Central Coordinator:
  Collect all updated local models $W_t^{(i)}$;
  Update the global model according to the formula: $W_{t+1}^{(g)} = \alpha \cdot W_t^{(g)} + (1-\alpha)\sum_{i=1}^{N} p_i \cdot W_t^{(i)}$;
  Broadcast the updated global model parameters $W_{t+1}^{(g)}$ to all participants;
end

Algorithm 2. Federated aggregation algorithm for RL based agent model parameters.

Each local agent in environments $1, 2, \ldots, N$ downloads the global model $W_0^{(g)}$ from the federated central coordinator;
Initialize the training step counter to 0, and initialize the experience buffer $\mathscr{D}^{(i)}$ for each environment $i \in \{1, 2, \ldots, N\}$;
for *each epoch* $= 1, 2, 3, \ldots$ **do**
  Initialize each environment and obtain the initial state $s_0^{(i)}$ and the intersection signal phase $a_0^{(i)}$;
  for *each time step* $t = 1, 2, 3, \ldots, T$ **do**
    for *each agent* $i = 1, 2, \ldots, N$ **do**
      Each agent selects an action $a_t^{(i)} \sim \pi_{\theta^{(i)}}\left(\cdot \mid s_t^{(i)}\right)$ based on the current state;
      Determine if $a_t^{(i)}$ matches the current signal phase in the local environment;
      If the phases do not match, set the intersection phase to yellow for T_y seconds; other intersections retain their current phases;
      Switch the intersection phase to $a_t^{(i)}$ and maintain it for $\Delta T - T_y$ seconds;
      Transition to the new traffic state $s_{t+\Delta T}^{(i)}$ and calculate the reward $r_t^{(i)}$;
      Store the tuple $\langle s_t^{(i)}, a_t^{(i)}, r_t^{(i)}, s_{t+\Delta T}^{(i)} \rangle$ in the experience buffer $\mathscr{D}^{(i)}$;
    end
  end
end
Increment t by ΔT and update the training step counter by one;
if *the training step counter* $= K$ *or* $t = T$ **then**
  Agents upload their data from local experience buffers to the federated central coordinator;
  The central coordinator updates the global model parameters based on the collected data;
  Update $\pi_{\theta_{old}^{(g)}}(a \mid s) \leftarrow \pi_{\theta^{(g)}}(a \mid s)$;
  The central coordinator broadcasts the updated global model parameters $W_{t+1}^{(g)} = \left\{\pi_{\theta^{(g)}}, \pi_{\theta_{old}^{(g)}}, V_{\omega^{(g)}}\right\}$ to all agents;
  Local agents update their models accordingly and clear the experience buffer $\mathscr{D}^{(i)}, i \in \{1, 2, \ldots, N\}$;
  Reset the training step counter to 0;
end

Algorithm 3. Single intersection signal control algorithm based on aggregated PPO.

	Low saturation	Unbalanced 1	Unbalanced 2	Oversaturated
Vehicle arrival rate in the north-south direction (vehicles/second)	0.05	0.05	0.05	0.15
Vehicle arrival rate in the east-west direction (vehicles/second)	0.05	0.15	0.3	0.3

Table 1. Traffic flow pattern settings for each environment.

Symbol	Definition	Value
epoch	Training rounds	200
T	Simulation step size for each round	3600
ΔT	Individual phase time	30 s
γ	Discount factor	0.9
α_{actor}	Actor learning rate	0.0001
α_{critic}	Critic learning rate	0.001
ϵ	Clip parameters	0.2

Table 2. Description of federated PPO training parameters.

Simulation experiment and result analysis

To validate the effectiveness of the proposed single-intersection signal control method based on federated PPO, this section presents a series of training and testing experiments conducted under various traffic flow scenarios. The experimental results are analyzed in detail to assess the method's performance comprehensively.

Experimental design for single intersection simulation

The experimental platform selected for this study is SUMO version 1.7.0, interfaced with the federated PPO-based traffic signal optimization controller via the TraCI API. The algorithm is implemented in Python 3.6.12, using the open-source neural network framework PyTorch 1.7.0. The hardware environment for the experiments includes a 3.7 GHz Intel Core i9-10900X CPU, an NVIDIA GeForce RTX 3090 GPU with 24 GB memory, and 64 GB of RAM.

(1) Single Intersection Setup

A model of a four-way intersection was established on the SUMO platform, with each approach road measuring 200 meters. To replicate realistic conditions, only the directions for entering and exiting the intersection were configured without restricting specific lanes. Vehicle parameters were set as follows: vehicle length of 5 meters, maximum speed of 13.9 m/s (equivalent to 50.4 km/h), minimum spacing of 2.5 meters, acceleration rate of 1 m/s^2, and deceleration rate of 4.5 m/s^2.

(2) Traffic Flow Patterns

Vehicle arrivals in each direction followed a Bernoulli process. With the default signal settings, the phase cycle time ΔT was 30 seconds, including a yellow light duration Ty of 3 seconds. Simulations on the SUMO platform showed that within a 30-second green light, up to 10 vehicles could pass through each lane. In this intersection setting, for all arriving vehicles to clear within the green light duration across four phases (120 seconds), the vehicle arrival rate per lane must remain below 10 vehicles per 120 seconds, leading to a saturation threshold of 0.083 vehicles/second. Arrival rates below this threshold indicate an unsaturated lane, while rates above signify saturation.

In this experimental configuration, vehicles were categorized into two types based on entry direction: one group entering from the north or south, and the other from the west or east. By adjusting the arrival rates for these two groups, distinct traffic flow patterns were defined. Agent training was conducted in environments representing four traffic scenarios, as detailed in Table 1. In the Low Saturation scenario, vehicle arrival rates in all directions remained below 0.083 vehicles/second. In the Unbalanced 1 and Unbalanced 2 scenarios, arrival rates were below 0.083 vehicles/second for north-south lanes and above for east-west lanes. In the Oversaturated scenario, arrival rates greatly exceeded 0.083 vehicles/second in all directions. The trained agents under these conditions are denoted as "PPO-Low," "PPO-Un1," "PPO-Un2," and "PPO-Over" respectively. Agents trained under federated conditions are termed "Federated PPO," while agents trained with data from all local environments collectively are referred to as "Aggregated PPO."

(3) Training Parameters for federated PPO

Training parameters for single-intersection traffic signal control using federated PPO are specified in Table 2. Basic simulation parameters include *epoch count*, T, and ΔT, while γ, α_{actor}, α_{critic}, and ϵ are consistent across independent PPO, federated PPO, and aggregated PPO models.

(4) Performance Metrics

The following performance metrics were used to evaluate the proposed algorithm:

1. **Reward value:** During training, the reward value as defined in (5) was monitored. A positive reward indicates that the average vehicle waiting time decreased compared to the previous phase, suggesting effective agent actions; conversely, a negative reward reflects suboptimal performance.

2. **Average vehicle waiting time:** The average waiting time of all vehicles in the network, where shorter times signify improved traffic conditions.
3. **Average vehicle stops:** The average number of stops per vehicle across the network, with fewer stops indicating smoother traffic flow.
4. **Average travel time:** The mean time for vehicles to traverse the network from entry to exit. Shorter travel times suggest higher intersection efficiency.

This setup facilitates a comprehensive evaluation of the federated PPO model under various traffic conditions, measuring both the direct and system-wide impacts of the traffic signal control strategy.

Analysis of training results

During the training process, four different traffic flow patterns, as outlined in Table 1, were selected for single-intersection environments. Both individual PPO and federated PPO training were conducted, with multiple experiments performed to determine the optimal federated update settings, focusing on key parameters such as update frequency, learning rate, and learning weight. Additionally, to verify whether federated PPO can achieve the same performance as aggregated PPO, which theoretically uses all available data for experiential learning, a comparative experiment between federated PPO and aggregated PPO was also conducted.

(1) Comparison of model performance under different federated update frequencies (K)

In FL, communication overhead is typically higher than computational costs. To reduce the number of communication rounds required during training, multiple local model updates can be performed between two communication rounds. This section explores the impact of different federated update frequencies (K) on model performance and communication overhead in a federated PPO-based signal control method for single-intersection environments. Fig. 6 shows the training curves for federated PPO and individual PPO with K values of 1, 5, and 10. The training curve represents the cumulative reward r_t^{acc}, which is the sum of rewards from the start of training to the current moment, i.e., $r_t^{acc} = r_0 + r_{\Delta T} + \cdots + r_t$. Physically, this value represents the inverse of the average vehicle waiting time. When the reward stabilizes around 0, the cumulative reward stabilizes, indicating model convergence. From Fig. 6, regardless of the value of K, it is evident that federated PPO significantly improves the cumulative reward in the low-saturation traffic flow mode, indicating its ability to optimize model control performance in this scenario. In the other three traffic flow modes, federated PPO achieves nearly the same convergence height as individual PPO, suggesting that individual PPO also performs well in these traffic flow settings.

To quantitatively analyze model convergence speed, a mathematical method was proposed, which uses the moving average of the cumulative reward to determine convergence. The specific criteria are as follows:

- The percentage change in the average cumulative reward over five consecutive windows should not exceed 0.02%;
- The sum of the percentage changes in the average cumulative reward over five consecutive windows should not exceed 0.05%;
- The difference between the cumulative reward in the current window and the final window should not exceed 5%.

The smallest training step count that satisfies these three conditions is considered the point of model convergence. A larger value indicates slower convergence, and vice versa. A window size of 120 was chosen, and the impact of different federated update frequencies (K) on model convergence speed was analyzed. The results are shown in Table 3. It is evident that different values of K accelerate the convergence speed in the Unbalanced 1, Unbalanced 2, and oversaturated traffic flow modes. The effect is most pronounced when $K = 5$, which accelerates the convergence speed of the DRL agent by 29.57% on average. For $K = 1$ and $K = 10$, federated PPO converges 19.19% and 18.85% faster than individual PPO, respectively.

This section also evaluates the additional communication and computation time incurred by the FL framework, with specific data presented in Table 4. In the individual PPO-based signal optimization control method for a single intersection, each agent has three neural networks: two Actor networks and one Critic network. The Actor network consists of layers with 16, 64, 16, and 4 neurons, while the Critic network has 16, 64, 16, and 1 neuron in the respective layers. During each federated aggregation, each Actor network requires the transmission of $(16+1)\times 64+(64+1)\times 16+(16+1)\times 4=2196$ parameters, and each Critic network requires 2145 parameters. Each parameter is encoded as a float64 variable, so the total size of the network parameters transmitted between the agent and the federated central coordinator is $(2196\times 2+2145)\times 8B=51.07KB$. Assuming a communication rate of 5 MB/s between the agent and the central coordinator, the communication time per transmission is approximately 9.97 ms. Computation time was measured during training. The data in Table 4 show that the additional communication and computation time in FL decreases with higher values of K. Balancing training performance and resource requirements, $K = 10$ was selected as the default setting.

(2) Comparison of model effects under different federal learning rates α

This section explores the impact of different values of α in (3) on the training performance of the federated PPO model. A larger α value means that the global model from the previous round has a higher weight in the federated averaging process, which also implies that the impact of the current update on the global model is smaller. The results are shown in Fig. 7. From the figure, it is observed that when $\alpha = 0.9$, federated PPO converges to a lower value than individual PPO, indicating that the inclusion of the FL framework in this case worsens the model's performance. This confirms that the value of α in the federated averaging algorithm should not be too large. When $\alpha = 0$, $\alpha = 0.1$, and $\alpha = 0.5$, the model convergence heights are similar, suggesting that for $\alpha \leq 0.5$, the final convergence performance is almost unaffected.

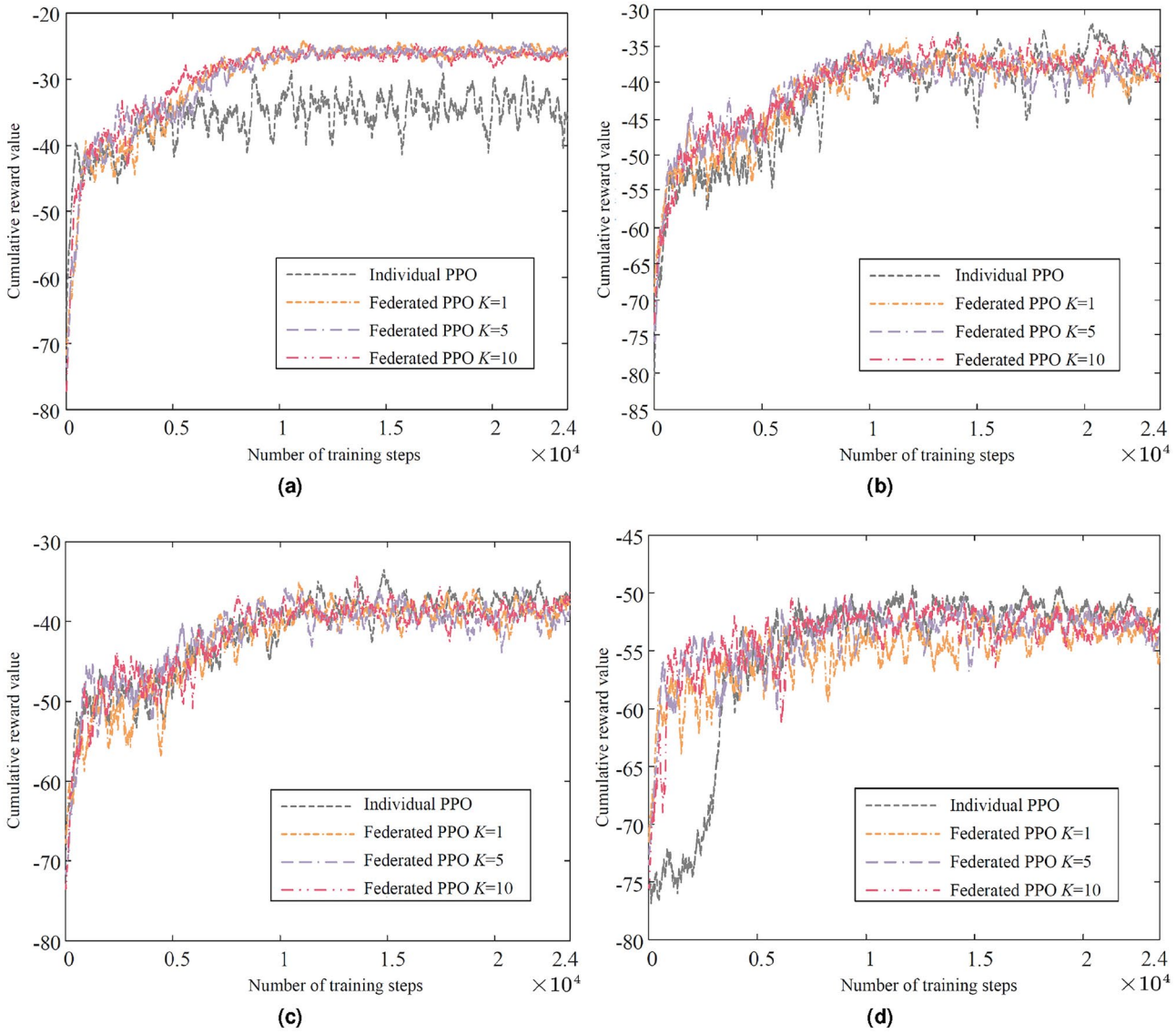

Fig. 6. Training curves for different federal update frequencies K for each traffic pattern. (**a**) Low saturation traffic flow pattern. (**b**) Unbalanced traffic flow pattern 1. (**c**) Unbalanced traffic flow pattern 2. (**d**) Oversaturated traffic pattern.

	Individual PPO	Federated PPO ($K = 1$)	Federated PPO ($K = 5$)	Federated PPO ($K = 10$)
Low saturation	**4679**	8276 (+76.88%)	8735 (+86.69%)	9211 (+96.86%)
Unbalanced 1	8578	7898 (−7.93%)	**6608 (−22.97%)**	8346 (−2.70%)
Unbalanced 2	9253	**7199 (−22.20%)**	7388 (−20.16%)	7729 (−16.47%)
Oversaturated	4655	3073 (−33.98%)	**1840 (−60.47%)**	2173 (−53.32%)

Table 3. Comparison of model convergence speeds for various traffic flow patterns with different federal update frequencies K. Significant values are in bold.

	Federated PPO ($K = 1$)	Federated PPO ($K = 5$)	Federated PPO ($K = 10$)
Communication time (s)	478.65	95.72	47.86
Calculation time (s)	121.84	23.97	12.02

Table 4. Additional communication and computation time required for different federal update frequencies K.

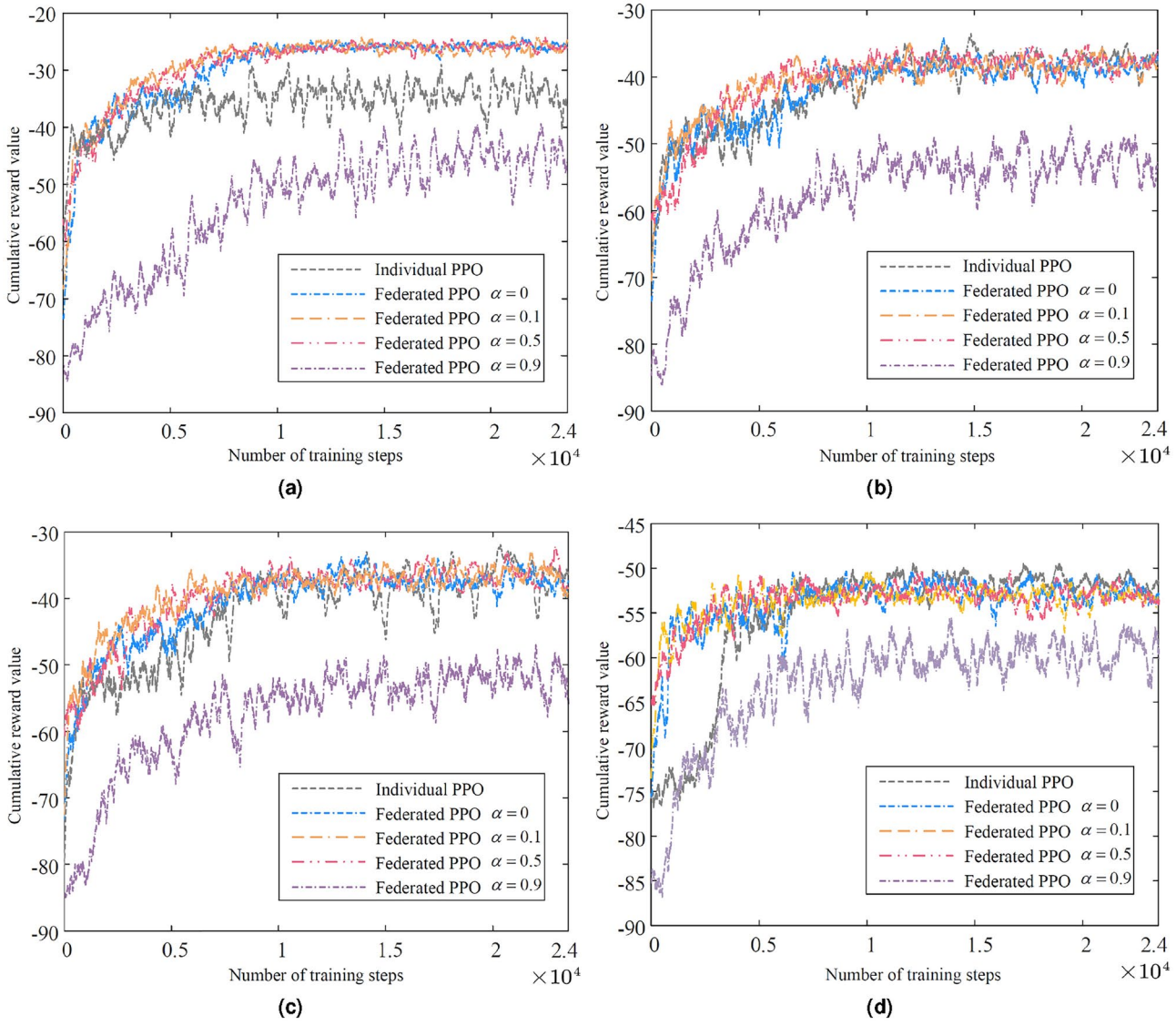

Fig. 7. Training curves for different federal learning rates α for each traffic flow pattern. (**a**) Low saturation traffic flow pattern. (**b**) Unbalanced traffic flow pattern 1. (**c**) Unbalanced traffic flow pattern 2. (**d**) Oversaturated traffic pattern.

	Individual PPO	Federated PPO ($\alpha = 0$)	Federated PPO ($\alpha = 0.1$)	Federated PPO ($\alpha = 0.5$)
Low saturation	**4679**	9211(+96.86%)	**6841(+46.21%)**	7613(+62.71%)
Unbalanced 1	8578	8346(−2.70%)	**5514(−35.72%)**	6360(−25.86%)
Unbalanced 2	9253	7729(−16.47%)	**6122(−33.84%)**	6928(−25.13%)
Oversaturated	4655	2173(−53.32%)	**1637(−64.83%)**	2272(−51.19%)

Table 5. Comparison of model convergence speeds for different federal learning rates α across traffic flow patterns. Significant values are in bold.

Based on the convergence criteria defined earlier, Table 5 provides the convergence speeds for different α values. $\alpha = 0$ indicates that each global model update only uses the current local model parameters, while $\alpha > 0$ retains some parameters from the previous global model. The data show that, compared to the case of $\alpha > 0$, federated PPO with $\alpha = 0$ results in slower convergence, indicating that using a soft update of the global model is an effective way to accelerate training convergence. When $\alpha = 0.1$, the model requires 40.97% fewer training steps to reach convergence in the Unbalanced 1, Unbalanced 2, and oversaturated traffic flow modes compared to individual PPO, significantly improving convergence speed. In the low-saturation traffic

flow mode, federated PPO converges slower than individual PPO, but $\alpha = 0.1$ produces the fastest convergence among all tested α values. Based on these experimental results, $\alpha = 0.1$ is determined to be the optimal FL rate.

(3) Comparison of model effects with different federal update weights p_i

In (3), the default weight $p_i = 1/N$ assumes that each participant has equal influence. This section also explores a flexible weight allocation method, where each agent is assigned a different weight p_i based on its performance during local updates. The ability of an agent is determined by its historical data from local updates, as detailed below: during the transition from the previous federated update to the current one, agent *i* performs *K* local model updates. The number of times the reward is greater than 0 during these updates is counted as the agent's score $score_i$. A higher $score_i$ indicates that the agent's local updates have more effectively improved its policy towards higher rewards. After scoring all agents involved in the current federated update, the scores are normalized to determine the corresponding weights p_i, calculated as follows:

$$p_i = \begin{cases} \frac{score_i}{\sum_{i=1}^{N} score_i}, & \sum_{i=1}^{N} score_i \neq 0 \\ \frac{1}{N}, & \sum_{i=1}^{N} score_i = 0. \end{cases} \tag{7}$$

If the reward is less than 0 for all *K* local updates, the agent's update is considered ineffective, and its weight in the federated aggregation is set to 0, meaning it has no impact on the global model. When all local agents have a score of 0, the weights are allocated evenly.

Fig. 8 shows the model training curves for both the equal weight allocation and the flexible allocation methods. The final convergence heights are nearly identical, indicating that different weight allocation strategies do not significantly affect the model's convergence height. Table 6 illustrates that when using flexible weight allocation, the federated PPO model converges faster than the model using equal weight allocation in the Unbalanced 1 and Unbalanced 2 traffic flow modes. Across all four traffic flow modes, federated PPO with flexible allocation achieves an average convergence speed improvement of 30.98%, compared to 25.96% for equal weight allocation. Therefore, in subsequent experiments, flexible weight allocation will be used for p_i.

(4) Comparison of model effects for federated PPO and aggregated PPO

Federated PPO achieves global model updates by aggregating the network parameters of each agent, whereas aggregated PPO performs updates based on each agent's state, action, and reward data. This section compares the performance of federated PPO and aggregated PPO at update frequencies of $K = 1$ and $K = 10$.

Fig. 9 illustrates the training curves for $K = 1$ and $K = 10$. When $K = 1$, aggregated PPO demonstrates higher convergence than federated PPO, especially under low-saturation traffic conditions. With $K = 10$, the convergence levels of federated PPO and aggregated PPO become comparable. As *K* increases, aggregated PPO shows a decline in convergence, whereas federated PPO remains relatively stable, indicating that the advantage in convergence for aggregated PPO is most evident at higher update frequencies. Table 7 summarizes the convergence speeds for both methods, showing that while federated PPO converges more slowly than aggregated PPO at $K = 1$, it surpasses aggregated PPO in speed at $K = 10$. This difference arises because, despite performing global model updates every 10 training steps, federated PPO still updates local policy models at each step, contributing to a faster relative convergence rate.

In addition, each aggregated PPO update requires transferring each agent's state, action, reward, and next state data for *K* local updates to the aggregator, amounting to $(8 + 1 + 1 + 8) \times 4 \times K$ float64 variables per update. Following the neural network update, network parameters are distributed, requiring the transmission of $2196 \times 2 + 2145 = 6537$ float64 variables, amounting to approximately 11.34 ms per aggregation update when $K = 10$. Over 200 training rounds, aggregated PPO requires about 27.22 seconds of communication time-shorter than federated PPO's additional 47.86 seconds-yet it demands substantially higher computation time (227.68 s), nearly 19 times that of federated PPO (12.02 s). In summary, federated PPO offers improved model convergence in terms of both speed and accuracy with minimal additional communication and computational costs while maintaining the privacy and security of local data.

Analysis of test results

This section evaluates the performance of federated PPO agents trained under settings of $K = 10$, $\alpha = 0.1$, and flexible allocation of p_i, aggregated PPO agents trained with $K = 10$, and individually trained PPO agents tailored for specific traffic patterns (Low saturation, Unbalance 1, Unbalance 2, and Oversaturated). Each approach is tested on a single intersection control task and benchmarked against a fixed-timing method with a default phase duration of 30 seconds. To account for the variability due to PPO agents' action selection based on probabilistic distributions, each scenario was tested 50 times, with average values reported to better reflect each agent's performance.

(1) Comparison of the control effect of each intelligence under four typical traffic flow patterns

As shown in Fig. 10, under low-saturation traffic conditions, the performance of all intelligent control methods closely matches that of the fixed-timing approach. Interestingly, the PPO-Low agent, trained specifically for low-saturation conditions, exhibited increased vehicle waiting times, travel times, and stop counts compared to fixed timing. This can be attributed to the low arrival rate (below 0.083 vehicles/second) in each direction, where fixed timing ensures all arriving vehicles can pass within the green light phase, minimizing congestion, while the inherent randomness in PPO introduces occasional delays.

Under Unbalanced 1, Unbalanced 2, and Oversaturated traffic conditions, intelligent control methods demonstrated varying degrees of optimization for vehicle metrics, including average waiting time, travel time, and stop count, with PPO-Low performing the least effectively. federated PPO consistently ranked among the top performers, reducing average waiting times by 40.80%, 60.48%, and 26.55% compared to fixed timing across these scenarios. Average travel times were reduced by 29.78%, 45.88%, and 21.27%, and average stop counts

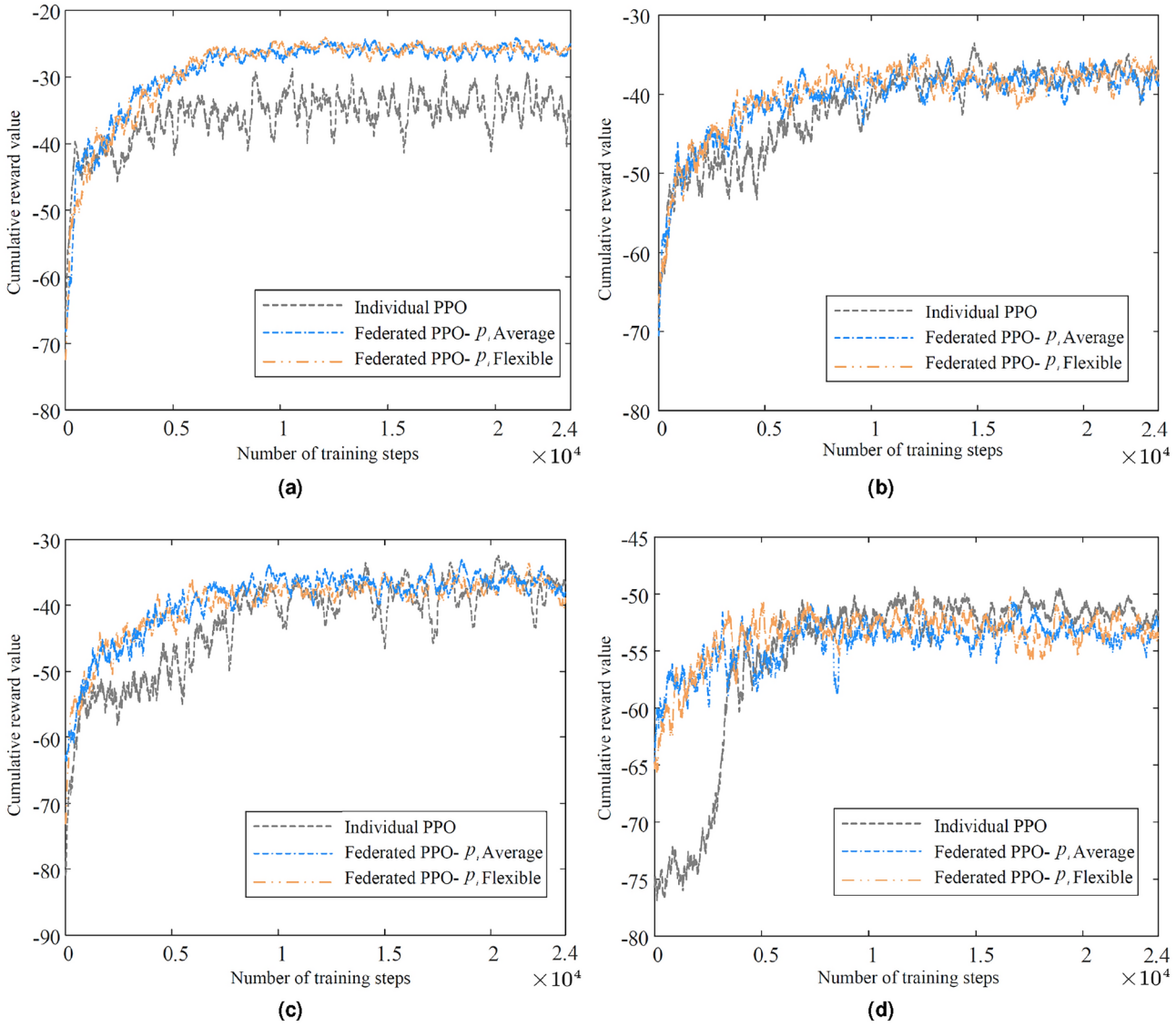

Fig. 8. Training curves for different federated update weight p_i allocation methods for each traffic flow pattern. (**a**) Low saturation traffic flow pattern. (**b**) Unbalanced traffic flow pattern 1. (**c**) Unbalanced traffic flow pattern 2. (**d**) Oversaturated traffic pattern.

	Individual PPO	Federated PPO (p_i average allocation)	Federated PPO (p_i flexible allocation)
Low saturation	**4679**	**6841(+46.21%)**	6987(+49.33%)
Unbalanced 1	8578	5514(−35.72%)	**4919(−42.66%)**
Unbalanced 2	9253	6122(−33.84%)	**5163(−44.20%)**
Oversaturated	4655	**1637(−64.83%)**	1681(−63.89%)

Table 6. Comparison of model convergence speeds for different federal update weight p_i assignments for each traffic flow pattern. Significant values are in bold.

were reduced by 37.37%, 44.49%, and 25.40%. federated PPO's performance approached that of aggregated PPO across these traffic scenarios and even surpassed it under oversaturated conditions.

On average, federated PPO achieved a 28.89% reduction in vehicle waiting time, a 22.90% reduction in travel time, and a 26.52% reduction in stop count. Aggregated PPO displayed the best overall performance, reducing waiting time, travel time, and stop count by 32.65%, 24.61%, and 24.90%, respectively. Among the individual PPO agents, PPO-Over provided the most notable optimizations with reductions of 31.40% in waiting time, 24.26% in travel time, and 26.00% in stop count, while PPO-Low showed the weakest improvements, at 9.17%,

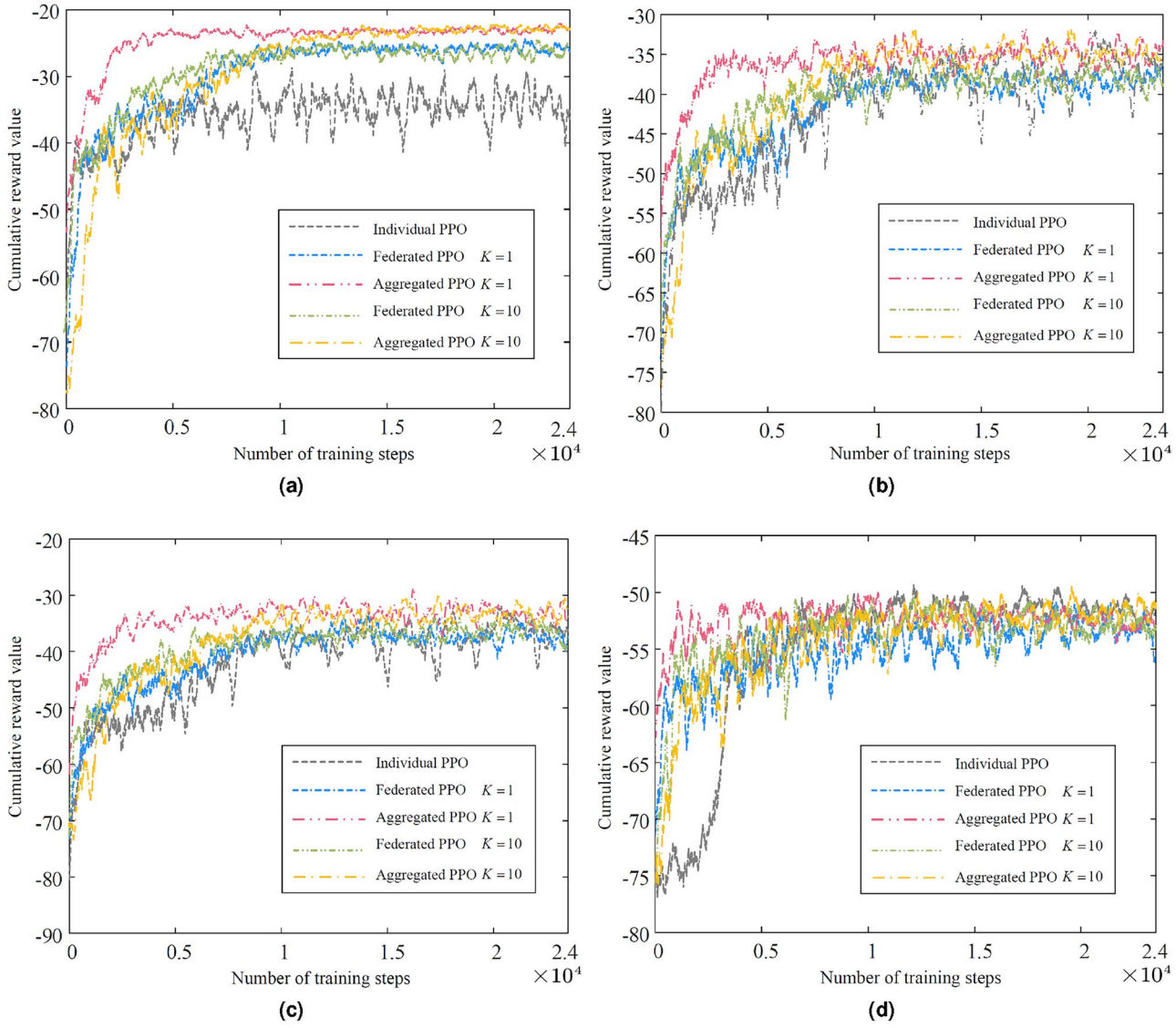

Fig. 9. Training curves for federated PPO and aggregated PPO at $K = 1$ and $K = 10$. (**a**) Low saturation traffic flow pattern. (**b**) Unbalanced traffic flow pattern 1. (**c**) Unbalanced traffic flow pattern 2. (**d**) Oversaturated traffic pattern.

	$K = 1$		$K = 10$	
	Federated PPO	**Aggregated PPO**	**Federated PPO**	**Aggregated PPO**
Low saturation	8276	4563	6987	12482
Unbalanced 1	7898	4245	4919	8076
Unbalanced 2	7199	4319	5163	9089
Oversaturated	3073	1560	1681	4665

Table 7. Model convergence speed comparison between federated PPO and aggregated PPO.

12.00%, and 4.67%, respectively. The other two individual PPO agents achieved moderate improvements, yet all fell short of federated PPO's performance.

(2) Control effect comparison of agents in diverse traffic flow patterns

This section further examines agent performance by defining 24 traffic flow patterns through varying arrival rates for north-south and east-west traffic, covering nearly all real-world single-intersection scenarios. Federated PPO, aggregated PPO, and individual PPO agents are used to control traffic signals, and their performance across these 24 patterns is compared. As indicated by Fig. 10, reduced vehicle waiting times correlate closely

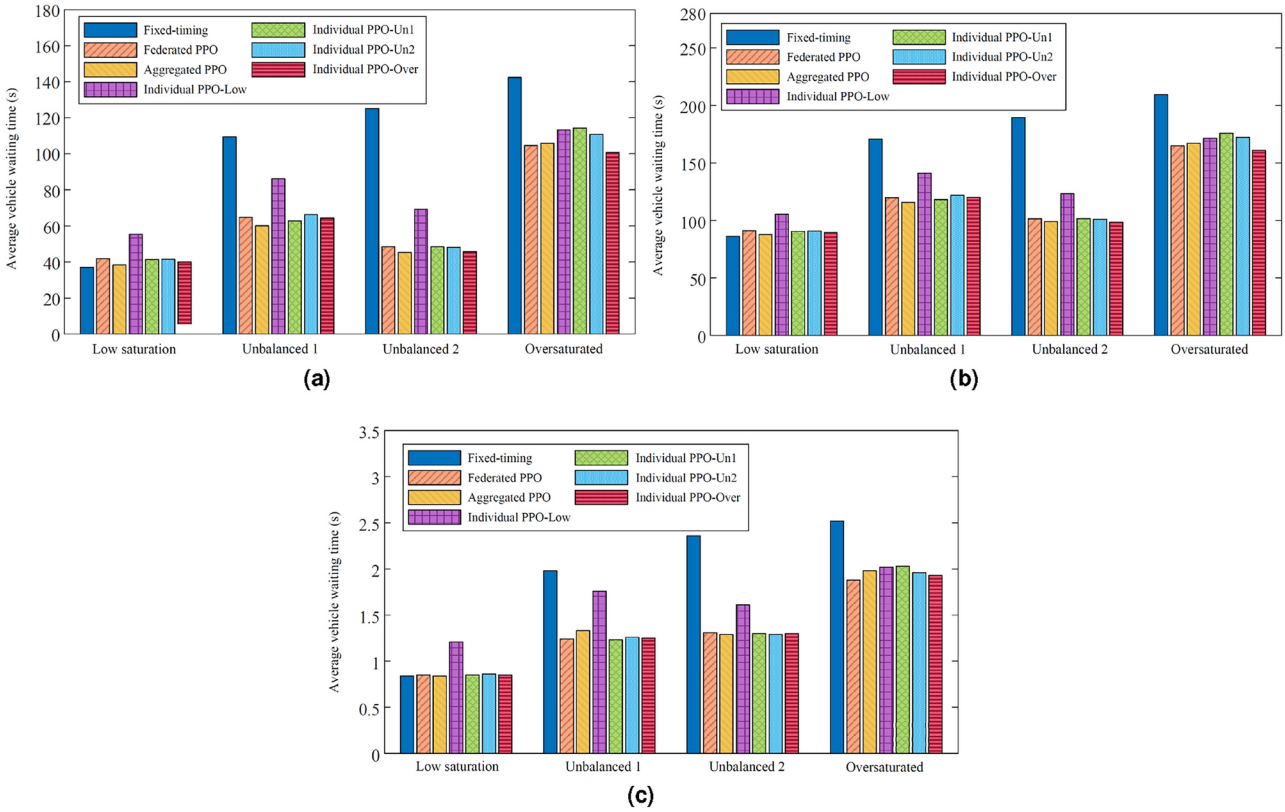

Fig. 10. Comparison of federated PPO, aggregated PPO and individual PPO performance in single intersection tasks. (**a**) Average vehicle waiting time. (**b**) Average vehicle travel time. (**c**) Average number of vehicle stops.

with lower travel times and stop counts; thus, only waiting time is used as the performance metric in subsequent experiments.

Fig. 11 displays the average vehicle waiting times under different traffic conditions using federated PPO and individual PPO agents. When the north-south arrival rate is 0.05 vehicles/second, federated PPO performs comparably to individual PPO agents (with the exception of PPO-Low), and fixed-timing control is optimal when the east-west arrival rate is below 0.083 vehicles/second. As the east-west rate exceeds 0.083 vehicles/second, both federated PPO and individual PPO significantly reduce waiting times. At a north-south arrival rate of 0.1 vehicles/second, all agents exhibit less effective control when the east-west arrival rates are 0.075 and 0.1 vehicles/second, with waiting times exceeding those of fixed timing. federated PPO achieves the best control performance when the east-west rate is 0.15 or 0.2 vehicles/second, maintaining a close second otherwise. When the north-south arrival rate reaches 0.3 vehicles/second and the east-west rate remains below 0.15 vehicles/second, PPO-Over performs best; as the east-west rate increases beyond 0.15 vehicles/second, PPO-Un1 excels. federated PPO consistently ranks in the top two positions, indicating stable performance across various settings.

These findings reveal each algorithm's optimization capabilities across diverse traffic scenarios. federated PPO consistently achieves near-optimal results across all 24 scenarios, avoiding any scenario-specific performance "collapse." It reduces vehicle waiting time by an average of 27.34%, compared to 6.37%, 27.15%, 24.59%, and 27.24% reductions for the individual PPO agents. The variance in federated PPO's performance across these patterns is 0.0625, lower than the variances for the individual PPO agents (0.1170, 0.0582, 0.0679, and 0.0628), highlighting federated PPO's stability and generalizability. In contrast, PPO agents trained on a single traffic pattern exhibit performance volatility; for instance, PPO-Un1 achieves a greater than 10% reduction in waiting time when the north-south rate is 0.3 vehicles/second and the east-west rate is over 0.2 vehicles/second but ranks lower under different conditions, such as a north-south rate of 0.1 vehicles/second and an east-west rate above 0.15 vehicles/second. The results from PPO-Low further indicate that an agent trained solely on low-saturation data would yield suboptimal control under fluctuating conditions. In contrast, the federated PPO-based approach for single-intersection signal control, enhanced by the FL framework, achieves improved control outcomes without requiring privacy-sensitive data exchange.

Fig. 12 presents average waiting times under federated PPO and aggregated PPO across the 24 patterns. While aggregated PPO generally outperforms federated PPO in reducing waiting times, federated PPO demonstrates superior control under high-traffic conditions in all directions. Aggregated PPO reduces waiting time by an average of 29.97%, yet its performance variance is higher than that of federated PPO, indicating that federated

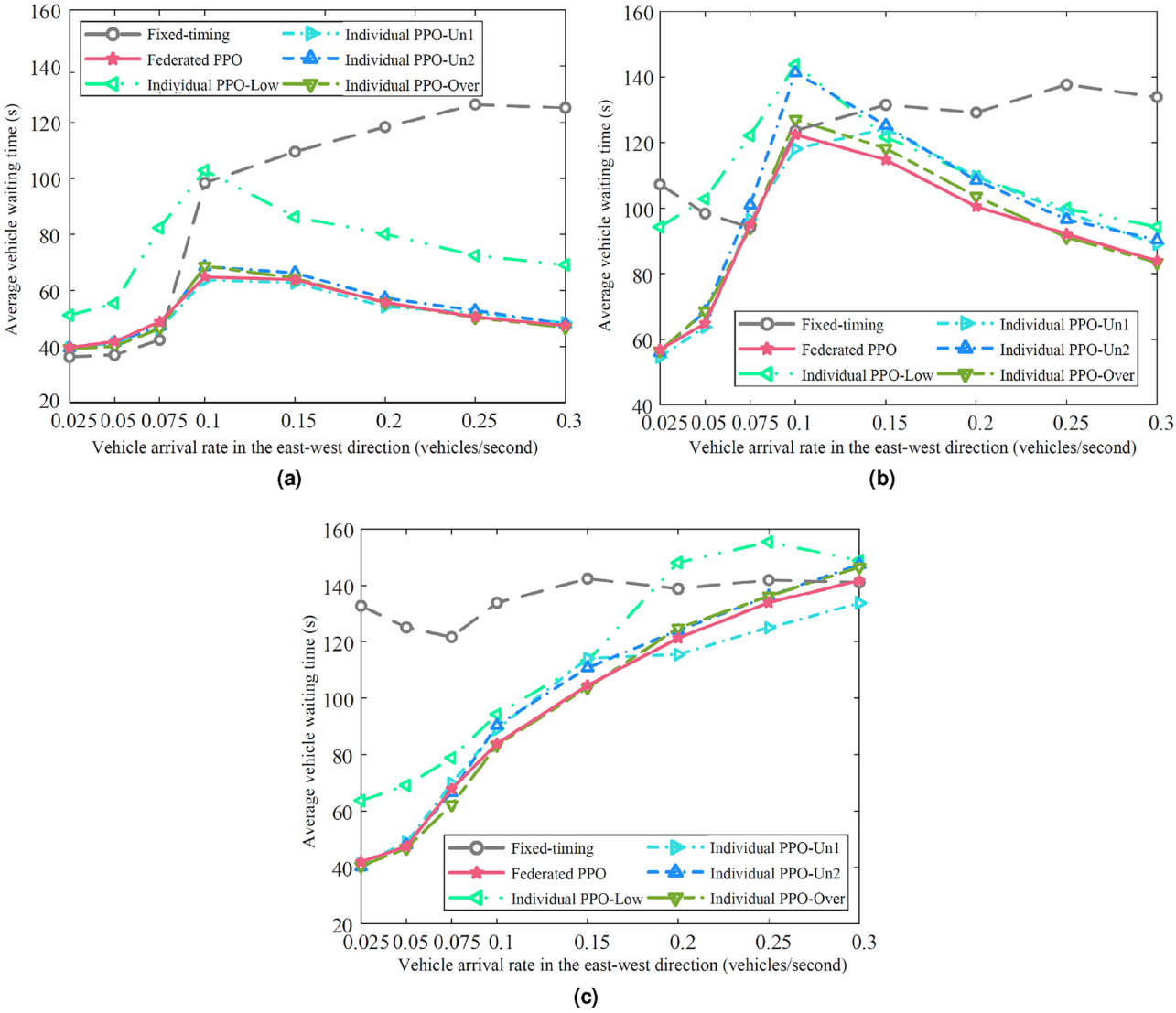

Fig. 11. Comparison of federated PPO and individual PPO performance in single intersection tasks. (**a**) Vehicle arrival rate in the north-south direction 0.05 vehicles/second. (**b**) Vehicle arrival rate in the north-south direction 0.1 vehicles/second. (**c**) Vehicle arrival rate in the north-south direction 0.3 vehicles/second.

PPO not only approaches aggregated PPO's performance but also provides more stable control across various traffic patterns.

Table 8 summarizes the specific optimization effects of the federated PPO agent on average vehicle waiting time across various traffic patterns compared to the fixed-timing method. The observations reveal two traffic conditions under which DRL does not significantly improve performance:

- When vehicle arrival rates in both north-south and east-west directions are below 0.083 vehicles per second, all directions remain unsaturated, making the fixed-timing method the optimal control approach.
- When arrival rates in the east-west and north-south directions are approximately equal, vehicle volumes in each direction are similar, and the agent's inherently probabilistic action selection strategy tends to increase waiting times.

Under all other conditions, federated PPO enhances intersection throughput. However, as the difference between east-west and north-south arrival rates narrows, federated PPO's effectiveness in reducing waiting times diminishes. In conditions of extreme unbalanced (e.g., a north-south arrival rate of 0.025 vehicles per second versus an east-west rate of 0.3 vehicles per second), federated PPO yields the highest improvement over fixed timing, reducing average vehicle waiting time by 68.40%.

(3) Control effectiveness of the algorithms in the three-lane single intersection environment

The previous tests were conducted within a standard four-way intersection, varying only the vehicle arrival rates in each direction. In this section, fixed-timing and intelligent signal control methods are extended to a single intersection with two straight lanes and one left-turn lane. Using the four traffic flow settings listed in

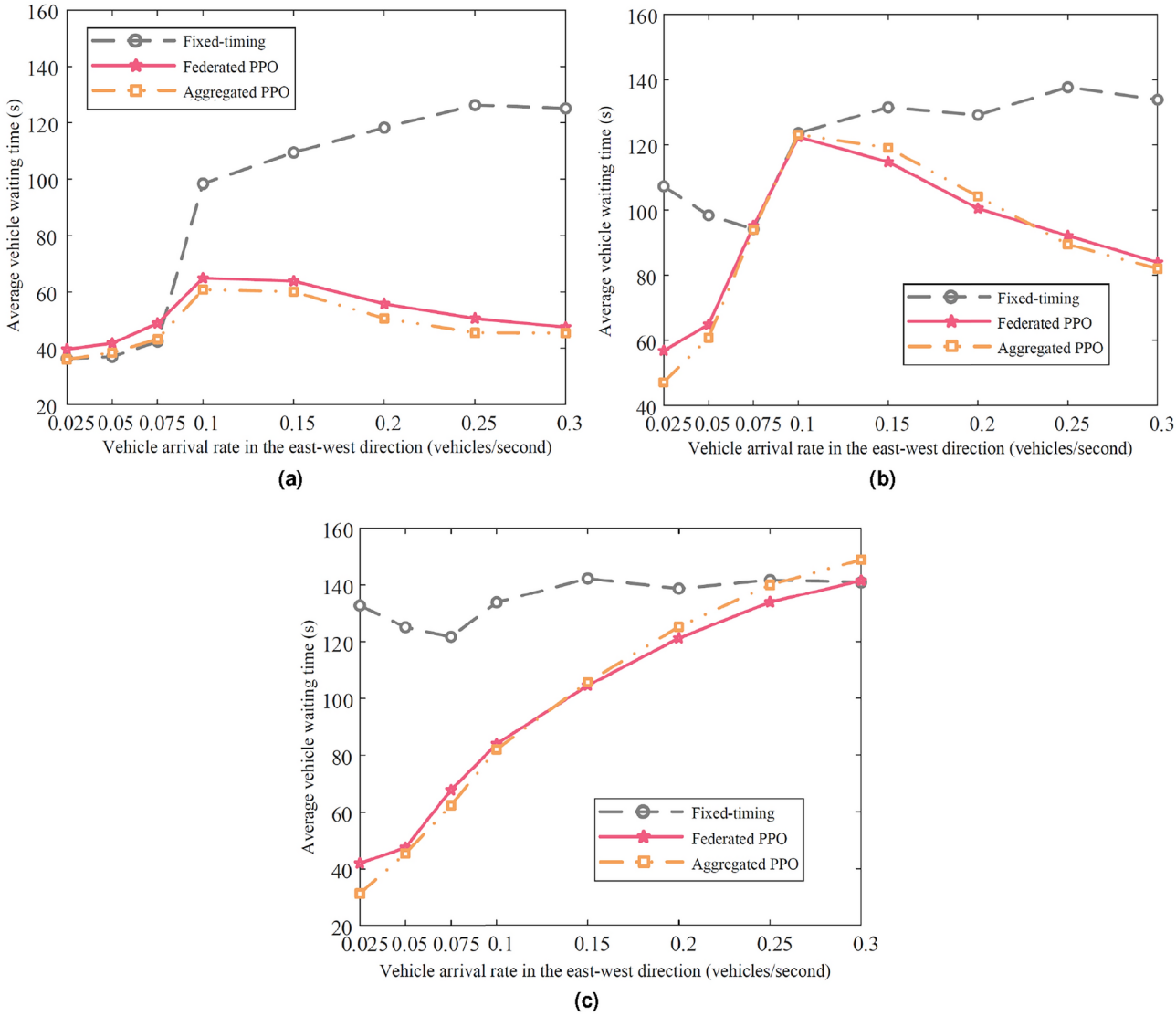

Fig. 12. Comparison of federated PPO and aggregated PPO performance for more traffic modes in single intersection tasks. (**a**) Vehicle arrival rate in the north-south direction 0.05 vehicles/second. (**b**) Vehicle arrival rate in the north-south direction 0.1 vehicles/second. (**c**) Vehicle arrival rate in the north-south direction 0.3 vehicles/second.

	North-South vehicle arrivals		
East-west vehicle arrivals	**0.05**	**0.1**	**0.3**
0.025	**+11.96%**	−47.04%	−68.40%
0.05	**+13.05%**	−32.04%	−60.48%
0.075	**+15.23%**	**+3.11%**	−44.28%
0.1	−32.04%	**+0.66%**	−35.07%
0.15	−40.08%	−12.76%	−26.55%
0.2	−52.03%	−21.83%	−12.68%
0.25	−58.40%	−32.96%	−5.60%
0.3	−60.48%	−35.07%	**+0.40%**

Table 8. Optimization effect of federated PPO algorithm on average vehicle waiting time compared to fixed timing method. Significant values are in bold.

	Low saturation	Unbalanced 1	Unbalanced 2	Oversaturated
Fixed-timing method	36.01	164.90	196.72	214.65
Federated PPO	60.95	83.40	61.81	134.58
Aggregated PPO	39.24	76.81	55.57	116.78
Individual PPO-Low	59.03	136.50	86.24	157.08
Individual PPO-Un1	106.55	160.24	124.37	205.91
Individual PPO-Un2	65.15	85.46	63.57	138.75
Individual PPO-Over	54.45	125.72	64.48	124.67

Table 9. Average vehicle waiting time (s) for a three-lane single intersection under the control of each algorithm.

Table 1, new simulation data was generated for this modified three-lane setup. During testing, agents receive the queue length and waiting time for each incoming lane. To adapt the state inputs, the two straight lanes are treated as a single unit: the queue length is taken as the average queue length across both lanes, while the waiting time is the maximum waiting time of the lead vehicle on either lane. Thus, the agent receives a 16-dimensional state vector, allowing it to apply previously trained models directly to this intersection setup. Detailed test results are shown in Table 9.

In low-saturation traffic flow settings, none of the intelligent algorithms were effective in significantly reducing the average vehicle waiting time. However, under other traffic flow patterns, all intelligent methods achieved varying degrees of improvement. Notably, unlike the results from the standard four-way intersection, the single-agent PPO-Un1 performed poorly in this environment, indicating limited generalization ability. Although the single-agent PPO-Low previously performed the worst in the four-way tests, it outperformed PPO-Un1 in this three-lane intersection setup. Aggregated PPO showed the highest effectiveness, reducing the average waiting time by 40.45%, followed by federated PPO with an average reduction of 21.51%. Among the single-agent PPO models, PPO-Over performed best, decreasing waiting times by 20.42%, while PPO-Un1 performed the worst, increasing average waiting times by 38.05%. These results underscore the robust generalization capability of federated PPO and suggest a feasible approach to applying trained agents across diverse intersection configurations for signal control, further validating the scalability of the federated PPO framework.

Remark 4 It is worth pointing out that despite both using a PPO-based FRL approach, the proposed method in this paper still offers significant superiority in terms of cross-domain applicability, improved generalization capabilities, and increased efficiency in processing non-IID data across distributed traffic intersections compared to the works of Lim et al. in[33,41]. Moreover, in contrast to some recent work on DRL-based collaborative traffic signal control[17–19], our approach transmits only model parameters, ensuring that private data is neither transmitted nor aggregated. Second , by exploiting federated learning, the proposed method has better adaptability to different intersection configurations and traffic conditions, surpassing the generalization ability of traditional DRL methods. By optimizing state interactions and reward functions, our approach significantly accelerates convergence while minimizing communication overhead compared to centralized or sparse DRL frameworks.

Conclusion

This study introduces a FRL-based approach for cross-domain traffic signal optimization, enabling joint training of RL models across multiple environments without the need for private data exchange. This approach enhances model learning efficiency and improves stability and generalization. By defining the state as queue length and waiting time, the method significantly reduces the dimensionality of the state space, while also ensuring that the state information is easily accessible. Extensive experiments were conducted to evaluate the effectiveness of individual PPO, federated PPO, and aggregated PPO in controlling traffic flow efficiency at single intersections. Overall, the proposed Federated PPO-based traffic signal control algorithm offers three primary advantages: (1) Faster Convergence: By integrating the FL framework, Federated PPO achieves an average convergence speed 47.69% faster than single-agent PPO while maintaining comparable convergence levels. Additionally, Federated PPO's convergence speed is 45.35% faster than that of Aggregated PPO, demonstrating a clear advantage in accelerating model convergence; (2) Enhanced Model Stability and Generalization: Federated PPO consistently demonstrates near-optimal or optimal control effectiveness across nearly all traffic flow patterns, achieving an average reduction of 27.34% in vehicle waiting time, and up to a 68.40% reduction in highly unbalanced traffic conditions. Furthermore, Federated PPO is effectively adaptable across different single-intersection configurations, showing more stable optimization results and better generalization than single-agent PPO; (3) Data Privacy and Security: During model training, Federated PPO only transmits model parameters without involving the transfer or computation of private data, thereby safeguarding data privacy and security.

The experimental results validate that the proposed algorithm improves traffic flow efficiency at single intersections, demonstrating the effectiveness of the Federated Reinforcement Learning-based framework for cross-domain intelligent traffic signal control in single-intersection scenarios.

Data availability

All data used and generated during the current study are included in this paper.

Received: 12 November 2024; Accepted: 24 February 2025
Published online: 05 April 2025

References

1. Aziz, H. A., Zhu, F. & Ukkusuri, S. V. Learning-based traffic signal control algorithms with neighborhood information sharing: An application for sustainable mobility. *Journal of Intelligent Transportation Systems* **22**, 40–52 (2018).
2. Zhao, D., Dai, Y. & Zhang, Z. Computational intelligence in urban traffic signal control: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* **42**, 485–494 (2011).
3. Shaikh, P. W., El-Abd, M., Khanafer, M. & Gao, K. A review on swarm intelligence and evolutionary algorithms for solving the traffic signal control problem. *IEEE transactions on intelligent transportation systems* **23**, 48–63 (2020).
4. Eom, M. & Kim, B.-I. The traffic signal control problem for intersections: a review. *European transport research review* **12**, 1–20 (2020).
5. Wei, H., Zheng, G., Gayah, V. & Li, Z. Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation. *ACM SIGKDD Explorations Newsletter* **22**, 12–18 (2021).
6. Chen, C. et al. Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control. *In Proceedings of the AAAI conference on artificial intelligence* **34**, 3414–3421 (2020).
7. Rasheed, F., Yau, K.-L.A., Noor, R. M., Wu, C. & Low, Y.-C. Deep reinforcement learning for traffic signal control: A review. *IEEE Access* **8**, 208016–208044 (2020).
8. Li, D., Zhu, F., Wu, J., Wong, Y. D. & Chen, T. Managing mixed traffic at signalized intersections: An adaptive signal control and cav coordination system based on deep reinforcement learning. *Expert Systems with Applications* **238**, 121959 (2024).
9. Liu, M., Yu, J. & Zhao, K. Dynamic event-triggered asynchronous fault detection via zonotopic threshold analysis for fuzzy hidden markov jump systems subject to generally hybrid probabilities. *IEEE Transactions on Fuzzy Systems* **32**, 6363–6377 (2024).
10. Abdulhai, B., Pringle, R. & Karakoulas, G. J. Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering* **129**, 278–285 (2003).
11. Medina, J. C. & Benekohal, R. F. Traffic signal control using reinforcement learning and the max-plus algorithm as a coordinating strategy. In *2012 15th international IEEE conference on intelligent transportation systems*, 596–601 (IEEE, 2012).
12. Prashanth, L. & Bhatnagar, S. Reinforcement learning with function approximation for traffic signal control. *IEEE Transactions on Intelligent Transportation Systems* **12**, 412–421 (2010).
13. Watkins, C. J. & Dayan, P. *Q-learning. Machine learning* **8**, 279–292 (1992).
14. Li, L., Lv, Y. & Wang, F.-Y. Traffic signal timing via deep reinforcement learning. *IEEE/CAA Journal of Automatica Sinica* **3**, 247–254 (2016).
15. Shabestary, S. M. A. & Abdulhai, B. Deep learning vs. discrete reinforcement learning for adaptive traffic signal control. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 286–293 (IEEE, 2018).
16. Zheng, G. et al. Learning phase competition for traffic signal control. In *Proceedings of the 28th ACM international conference on information and knowledge management*, 1963–1972 (2019).
17. Bie, Y., Ji, Y. & Ma, D. Multi-agent deep reinforcement learning collaborative traffic signal control method considering intersection heterogeneity. *Transportation Research Part C: Emerging Technologies* **164**, 104663 (2024).
18. Fan, L., Yang, Y., Ji, H. & Xiong, S. Optimization of traffic signal cooperative control with sparse deep reinforcement learning based on knowledge sharing. *Electronics* **14** (2025).
19. Hassan, M. A., Elhadef, M. & Khan, M. U. G. Collaborative traffic signal automation using deep q-learning. *IEEE Access* **11**, 136015–136032 (2023).
20. Liu, M., Yu, J. & Sun, Y. Dissipativity-based asynchronous fault detection for ts fuzzy nonlinear markov jump systems subject to uncertain probabilities. *Journal of the Franklin Institute* **360**, 12500–12534 (2023).
21. Du, Y., ShangGuan, W., Rong, D. & Chai, L. Ra-tsc: Learning adaptive traffic signal control strategy via deep reinforcement learning. In *2019 ieee intelligent transportation systems conference (itsc)*, 3275–3280 (IEEE, 2019).
22. Qi, J., Zhou, Q., Lei, L. & Zheng, K. Federated reinforcement learning: Techniques, applications, and open challenges. arXiv preprint arXiv:2108.11887 (2021).
23. Zhuo, H. H., Feng, W., Lin, Y., Xu, Q. & Yang, Q. Federated deep reinforcement learning. arXiv preprint arXiv:1901.08277 (2019).
24. Liu, T. et al. Parallel reinforcement learning: A framework and case study. *IEEE/CAA Journal of Automatica Sinica* **5**, 827–835 (2018).
25. Clemente, A. V., Castejón, H. N. & Chandra, A. Efficient parallel methods for deep reinforcement learning. arXiv preprint arXiv:1705.04862 (2017).
26. Nair, A. et al. Massively parallel methods for deep reinforcement learning. arXiv preprint arXiv:1507.04296 (2015).
27. Liu, M., Yu, J. & Rodríguez-Andina, J. J. Adaptive event-triggered asynchronous fault detection for nonlinear markov jump systems with its application: A zonotopic residual evaluation approach. *IEEE Transactions on Network Science and Engineering* **10**, 1792–1808 (2023).
28. Li, L., Fan, Y., Tse, M. & Lin, K.-Y. A review of applications in federated learning. *Computers & Industrial Engineering* **149**, 106854 (2020).
29. Pandya, S. et al. Federated learning for smart cities: A comprehensive survey. *Sustainable Energy Technologies and Assessments* **55**, 102987 (2023).
30. Ramu, S. P. et al. Federated learning enabled digital twins for smart cities: Concepts, recent advances, and future directions. *Sustainable Cities and Society* **79**, 103663 (2022).
31. Liu, B., Wang, L. & Liu, M. Lifelong federated reinforcement learning: a learning architecture for navigation in cloud robotic systems. *IEEE Robotics and Automation Letters* **4**, 4555–4562 (2019).
32. Han, Y., Li, D., Qi, H., Ren, J. & Wang, X. Federated learning-based computation offloading optimization in edge computing-supported internet of things. In *Proceedings of the ACM Turing Celebration Conference-China*, 1–5 (2019).
33. Lim, H.-K., Kim, J.-B., Heo, J.-S. & Han, Y.-H. Federated reinforcement learning for training control policies on multiple iot devices. *Sensors* **20**, 1359 (2020).
34. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017).
35. Lillicrap, T. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971 (2015).
36. Liang, X., Liu, Y., Chen, T., Liu, M. & Yang, Q. Federated transfer reinforcement learning for autonomous driving. In *Federated and Transfer Learning*, 357–371 (Springer, 2022).
37. Ye, Y., Zhao, W., Wei, T., Hu, S. & Chen, M. Fedlight: Federated reinforcement learning for autonomous multi-intersection traffic signal control. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 847–852 (IEEE, 2021).
38. McMahan, B., Moore, E., Ramage, D., Hampson, S. & y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282 (PMLR, 2017).

39. Haydari, A. & Yılmaz, Y. Deep reinforcement learning for intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems* **23**, 11–32 (2020).
40. Lowe, R. et al. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* **30** (2017).
41. Lim, H.-K., Kim, J.-B., Ullah, I., Heo, J.-S. & Han, Y.-H. Federated reinforcement learning acceleration method for precise control of multiple devices. *IEEE Access* **9**, 76296–76306 (2021).

Author contributions

Mi Li: conceptualization, funding acquisition, investigation, methodology, supervision, validation, writing-review and editing, and writing-original draft. Xiaolong Pan: conceptualization, data curation, formal analysis, investigation, methodology, validation, visualization, resources, software, and writing-review and editing. Chuhui Liu: data curation, formal analysis, methodology, validation. Zirui Li: data curation, formal analysis, investigation, methodology, validation, resources, software.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to M.L.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.