InitLight: Initial Model Generation for Traffic Signal Control Using Adversarial Inverse Reinforcement Learning

Yutong Ye[1] , **Yingbo Zhou**[1] , **Jiepin Ding**[1] , **Ting Wang**[1] , **Mingsong Chen**[1,2] and **Xiang Lian**[3]
[1]MoE Eng. Research Center of SW/HW Co-Design Tech. and App., East China Normal University
[2]Shanghai Institute of Intelligent Science and Technology, Tongji University
[3]Department of Computer Science, Kent State University
{ytye, ybzhou, jpding}@stu.ecnu.edu.cn, {twang, mschen}@sei.ecnu.edu.cn, xlian@kent.edu

Abstract

Due to repetitive trial-and-error style interactions between agents and a fixed traffic environment during the policy learning, existing Reinforcement Learning (RL)-based Traffic Signal Control (TSC) methods greatly suffer from long RL training time and poor adaptability of RL agents to other complex traffic environments. To address these problems, we propose a novel *Adversarial Inverse Reinforcement Learning* (AIRL)-based pre-training method named InitLight, which enables effective initial model generation for TSC agents. Unlike traditional RL-based TSC approaches that train a large number of agents simultaneously for a specific multi-intersection environment, InitLight pre-trains only one single initial model based on multiple single-intersection environments together with their expert trajectories. Since the reward function learned by InitLight can recover ground-truth TSC rewards for different intersections at optimality, the pre-trained agent can be deployed at intersections of any traffic environments as initial models to accelerate subsequent overall global RL training. Comprehensive experimental results show that, the initial model generated by InitLight can not only significantly accelerate the convergence with much fewer episodes, but also own superior generalization ability to accommodate various kinds of complex traffic environments.

1 Introduction

Along with the rapid growth of urbanization, we are witnessing an increasing number of traffic jams on road networks worldwide, inevitably resulting in both serious environmental pollution (e.g., carbon emissions) and tremendous loss in economic and time costs. To reduce such traffic congestion, various Traffic Signal Control (TSC) methods [Rizzo *et al.*, 2019; Chen *et al.*, 2020b; Zheng *et al.*, 2021] have been extensively studied. So far, the most widely used TSC approaches in practice can be classified into two categories: i) *rule-based methods* (e.g., FixedTime [Koonce and Rodegerdts, 2008], GreenWave [Török and Kertész, 1996], SCOOT [Hunt *et al.*, 1982], and SCATS [PR, 1992]) that make control decisions based on the pre-defined rules of traffic plans; and ii) *adaptive methods* (e.g., MaxPressure [Varaiya, 2013], MaxQueue [Zhang *et al.*, 2021], and SOTL [Cools *et al.*, 2013]) that control the traffic in a heuristic manner.

Thanks to the prosperity of Artificial Intelligence (AI), Reinforcement Learning (RL) has become more and more popular in designing intelligent policies to further improve TSC performance. Unlike traditional TSC methods, RL-based TSC equips each intersection with an agent, which gradually trains its control policy based on repetitive "trial-and-error" style interactions with the corresponding intersection traffic. Although such methods can achieve much better TSC performance than their traditional counterparts, due to the increasing simulation complexity of modern traffics, the training time for agents of RL-based TSC is skyrocketing. Worse still, existing RL-based TSC methods are typically traffic environment-specific, i.e., a set of correlated RL-based agents are trained together within a specific traffic environment. As a result, it is hard for such agents to either learn from or benefit from the training in other complex traffic environments. Therefore, *how to effectively improve both the learning efficiency and generalization ability of RL agents is becoming a major challenge in the design of RL-based TSC.*

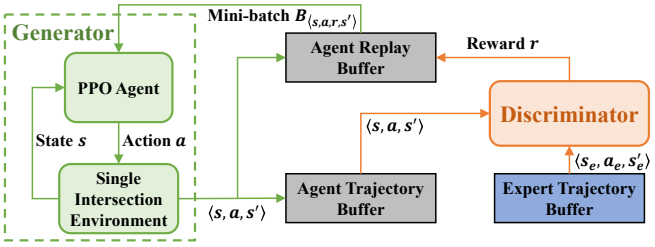

Figure 1: Our InitLight framework.

Inspired by the concept of *Adversarial Inverse Reinforcement Learning* (AIRL), we propose a novel agent pre-training method named InitLight, which can quickly generate an initial model with superior generalization ability to other complex traffic environments. As shown in Figure 1, InitLight is built on top of an adversarial framework consisting of two main components, i.e., generator and discriminator. Unlike traditional RL-based TSC methods that train a large set of correlated agents simultaneously for a specific multi-

intersection environment, the pre-training made by InitLight is based on a variety of single-intersection environments together with their expert trajectories. To form a pre-trained initial model, InitLight conducts the training on a Proximal Policy Optimization (PPO) [Schulman *et al.*, 2017] agent guided by the discriminator, which iteratively distinguishes the expert trajectories (obtained from a given TSC method) from agent trajectories (derived from real-time agent-environment interactions). Eventually, the discriminator learns a reward function that can recover the ground-truth reward function at optimality for TSC of any traffic environment. Therefore, the pre-trained agent by InitLight can be deployed at intersections of any complex traffic environments as the initial models to accelerate their overall global RL training. This paper makes the following three major contributions:

- We propose a novel AIRL-based framework that can quickly derive a general but effective initial model for various multi-intersection environments to accelerate their overall RL-training performance.
- We introduce an adversarial learning mechanism that supports simultaneous learning of the RL model and reward function, where the discriminator can recover the ground-truth reward at optimality.
- We implement InitLight on top of the traffic simulator Cityflow [Zhang *et al.*, 2019], and conduct comprehensive experiments on both complex real-world and synthetic multi-intersection datasets to show the effectiveness of our approach.

2 Related Work

Various RL-based methods [Chen *et al.*, 2022] have been proposed to improve the performance of TSC. For example, inspired by the Max Pressure (MP) control theory [Varaiya, 2013], PressLight [Wei *et al.*, 2019a], MPLight [Chen *et al.*, 2020a], and MetaLight [Zang *et al.*, 2020] conducted RL-based TSC optimization by encoding both system states and reward functions based on pressure. Similarly, Zhao et al. [2022] created a new concept named intensity, which takes extra vehicle dynamics (i.e., speed and position) besides pressure into account. Based on the notion of intensity, Ye et al. [2022] presented the fair index, which takes the travel quality of individual vehicles into account. To further improve TSC performance, Ye et al. [2021] developed FedLight, which adopts federated reinforcement learning to reduce the overall RL training costs. Jiang et al. [2022] proposed a universal communication form to enable high-quality TSC based on cooperation among intersections. Moreover, RL-based methods have been increasingly investigated in specific TSC problems, e.g., multi-agent-based TSC [de Almeida *et al.*, 2022; Koohy *et al.*, 2022], TSC with dynamic lanes [Jiang *et al.*, 2021], and resource-constrained TSC [Xing *et al.*, 2022]. Although the above methods are promising, most of them adopt randomly initialized models, where TSC agents need a long training time to achieve near-optimal control performance.

Under the guidance of expert trajectories, Inverse Reinforcement Learning (IRL) [Russell, 1998; Ng *et al.*, 2000] has been acknowledged as a promising way to optimize RL policy learning. However, existing IRL methods (e.g., maximum margin approaches [Abbeel and Ng, 2004; Ratliff *et al.*, 2006] and probabilistic approaches [Ziebart *et al.*, 2008; Ramachandran and Amir, 2007; Boularias *et al.*, 2011; Choi and Kim, 2013]) are not good at dealing with complex RL tasks. Although various adversarial IRL methods [Ho and Ermon, 2016; Finn *et al.*, 2016a; Finn *et al.*, 2016b] have been proposed to tackle such a problem, most of them: i) do not support reward function recovery; or ii) have to learn from complete trajectories. Therefore, they are unsuitable for RL scenarios with complex settings or uncertain environments. To enable smooth transfer between environments with notable variations, Fu et al. [2018] proposed an effective adversarial IRL method named AIRL based on a novel adversarial reward learning formulation, whose discriminator can accurately recover reward functions with superior adaptability to various dynamic environments. Inspired by AIRL, we propose a novel pre-training method that can derive initial models for TSC agents to accelerate the overall RL performance of complex traffic environments. Unlike existing RL-based TSC methods [Xiong *et al.*, 2019; Zhang *et al.*, 2020; Zhu *et al.*, 2021] that consider generalization performance, our work can quickly pre-train a general but effective initial model from single-intersection environments, which can be deployed to various kinds of multiple-intersection environments to accelerate their RL training processes.

To the best of our knowledge, InitLight is the first adversarial IRL framework for TSC to enable the pre-training of an initial model for the RL agent, which can be deployed to arbitrary complex traffic environments to accelerate their RL training processes.

3 Our InitLight Approach

To enable training an effective initial RL model with outstanding adaptability to various traffic environments, we design InitLight based on an adversarial architecture, where the discriminator learns a reward function from expert trajectories to guide the RL training. Figure 2 details the major components and workflow of InitLight. In our approach, the adversarial framework consists of a generator, a discriminator, and three buffers. Through interactions between the PPO agent and the corresponding traffic environment, the generator generates agent trajectory data stored in the agent trajectory buffer to confuse the discriminator. By distinguishing between expert and agent trajectories, the discriminator can learn a reward function to guide the learning process of the PPO agent. Based on the adversarial learning framework above, InitLight can improve both the learning efficiency and generalization ability of RL models. The following subsections will give the details of our approach.

3.1 Generator Design

As shown on the left part of Figure 2, the generator consists of a PPO agent and a single-intersection environment. The PPO agent interacts with the environment to generate agent trajectory data in the same way as classic reinforcement learning. We use solid-line arrows in the generator part to show this process. First, the agent obtains the current traffic state s

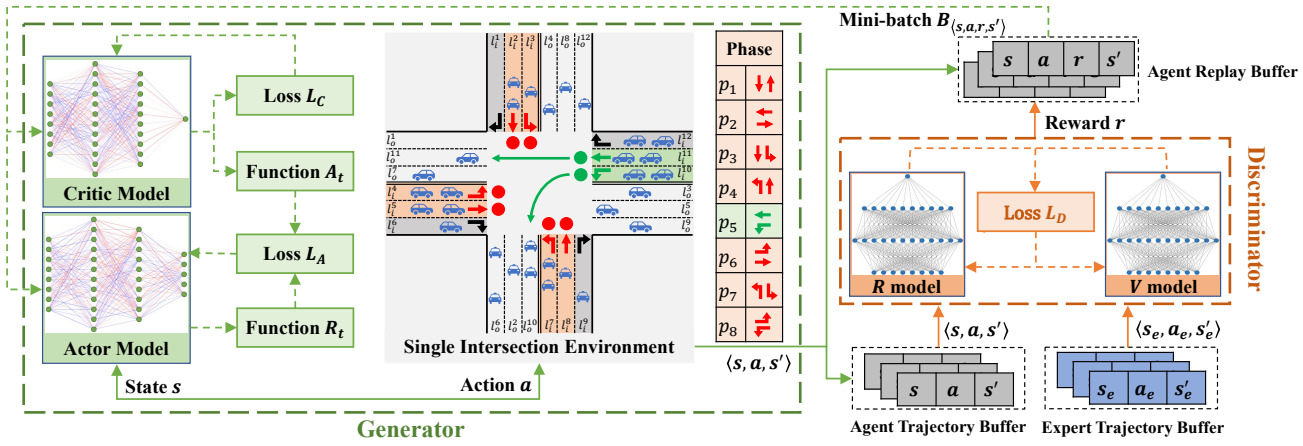

Figure 2: Architecture and workflow of InitLight.

from the intersection environment. Second, a best action a is chosen by the agent based on s. Finally, after applying a in traffic lights, the traffic environment will enter the next state s'. The trajectory data $\langle s, a, r, s' \rangle$ will be saved in the agent replay buffer for future learning by using mini-batch samples $B_{\langle s,a,r,s' \rangle}$, where reward r is obtained from the discriminator based on $\langle s, a, s' \rangle$.

Intersection Modeling. The right part of the generator in Figure 2 shows an intersection example. The intersection consists of a set of arrival lanes $L_a = \{l_i^1, l_i^2, \cdots, l_i^{12}\}$ and a set of departure lanes $L_d = \{l_o^1, l_o^2, \cdots, l_o^{12}\}$. Each road is represented by a 2-tuple (l_i, l_o), where $l_i \in L_a$ and $l_o \in L_d$ are incoming and outgoing lanes, respectively. Based on direction marks on the end of each arrival lane, we define a directed road as (l_i, l_o) if l_o is the departure lane indicated by the direction mark on the ground of l_i. For example, (l_i^{10}, l_o^{10}) and (l_i^{11}, l_o^{11}) are two directed roads. According to common sense, the vehicles turning right are not restricted by traffic signals. Therefore, there is a set of eight feasible control phases $P = \{p_1, \cdots, p_8\}$ indicating the rights-of-way signaled to vehicles by traffic lights. The intersection example shows a scenario with control phase p_5 enabled, where the vehicles on lane l_i^{10} can turn left to enter lane l_o^{10} and the vehicles on lane l_i^{11} can go straight to enter lane l_o^{11}. Note that, when the number of lanes increases, the number of control phases will not change.

Pressure. Inspired by the MP control theory [Varaiya, 2013], we use the concept of *pressure* for the design of RL agents.

Definition 1. ***(Pressure of a Directed Road).*** *For a directed road (l_i, l_o), the pressure of (l_i, l_o) is defined as*

$$P_{(l_i,l_o)} = N(l_i) - N(l_o), \tag{1}$$

where $N(l_i)$ and $N(l_o)$ are the numbers of vehicles on the arrival lane l_i and departure lane l_o, respectively.

For example, in Figure 2, the pressure of directed road (l_i^2, l_0^2) is $P_{(l_i^2,l_0^2)} = 2 - 1 = 1$.

Definition 2. ***(Pressure of an Intersection).*** *The pressure of an intersection I is calculated as:*

$$P_I = \left| \sum_{l_i \in L_a} N(l_i) - \sum_{l_o \in L_a} N(l_o) \right|. \tag{2}$$

In the example of Figure 2, we have $P_I = |0+2+1+2+2+1+2+1+1+2+2+2-0-1-1-1-1-2-1-1-1-2-1-1| = 5$.

Intuitively, the pressures of roads and intersections indicate the degrees of disequilibrium between the numbers of arrival and departure vehicles. In general, we can maximize the throughput of an intersection by minimizing the pressure, P_I, of the intersection. This is because larger P_I implies a more unbalanced distribution of vehicles.

Agent Design. Based on the pressure definition, we design the key elements of PPO agent using the following settings:

- **State:** The PPO agent captures part of the intersection information as the state. For a standard intersection as shown in Figure 2, the state includes the pressure of all the directed roads (i.e., $P_{(l_i^1,l_o^1)}, P_{(l_i^2,l_o^2)}, \cdots, P_{(l_i^{12},l_o^{12})}$) and the current control phase p_{cur} (i.e., p_5).
- **Action:** Based on the observed state, the PPO agent needs to choose one best control phase to maximize the throughput of the intersection. In this paper, the PPO agent has 8 permissible phases (i.e., $p_1, \cdots, p_8$) by default as shown in Figure 2.
- **Reward:** When an action finishes, the environment will return a reward as $r = -P_I$ to reflect the effect of the action, where P_I is the intersection pressure. Note that, in single-intersection environments, we use a discriminator to learn a reward function from expert trajectories and generate the reward of each agent trajectory sample, which is helpful to pre-train a robust initial agent.

In our approach, the PPO agent has two neural networks, namely, Actor and Critic models, with their own model parameters, A_θ and C_θ, respectively. The Actor model is used to learn which action to take under the current state, whereas the Critic model learns to evaluate whether or not the action (chosen by the Actor model) can result in a better state of the

traffic environment. Meanwhile, the feedback from the Critic model is used to optimize the Actor model. On the left part of Figure 2, the dashed-line arrows show the learning process of the PPO agent. To calculate the loss value for optimization, a mini-batch of sequential trajectory samples is collected from the agent replay buffer. Note that, the trajectory samples of the mini-batch must be continuous.

Critic Model. The loss function of the Critic model is:

$$L_C = \mathbb{E}[|C_\theta(s_t)_{target} - C_\theta(s_t)|], \tag{3}$$

where $\mathbb{E}$ is an operator to calculate empirical average over a mini-batch of samples. By using the Temporal-Difference (TD) algorithm [Tesauro and others, 1995] to estimate the target value, $C_\theta(s_t)_{target}$ is calculated by $C_\theta(s_t)_{target} = r_{t+1} + \gamma \cdot C(s_{t+1})$. The parameters of the Critic model can be updated by the stochastic gradient descent algorithm Adam [Kingma and Ba, 2014]:

$$\theta'_C = \theta_C - \eta_C \nabla L_C, \tag{4}$$

where ∇L_C is the gradient of Critic's loss, θ_C is model parameters of C, θ'_C denotes the updated parameters, and η_C is the learning rate for the optimization of the Critic model.

Actor Model. Since PPO is a policy gradient-based RL algorithm, the original loss function of the Actor model L_A is:

$$L_A = \mathbb{E}[log(A_\theta(a_t|s_t))A_t], \tag{5}$$

where A_t is an estimated value of the advantage function at time step t. By using the Generalized Advantage Estimator (GAE) [Schulman *et al.*, 2015], A_t is calculated as follows:

$$\begin{aligned} A_t = \delta_t + (\gamma\lambda)\delta_{t+1} + (\gamma\lambda)^2\delta_{t+2} \\ + \cdots + (\gamma\lambda)^{|B|-t+1}\delta_{|B|-1}, \end{aligned} \tag{6}$$

where $\gamma \in [0, 1]$ is the discount factor of future rewards, $\lambda \in [0, 1]$ is the GAE parameter, $|B|$ is the batch size of the sampled mini-batch B, and $\delta_t = r_t + \gamma C_\theta(s_{t+1}) - C_\theta(s_t)$.

In our PPO agent, instead of using the original loss function in Equation 5, we use the importance sampling to obtain the expectation of samples under the new Actor model A_θ we need to update. Note that, the samples are gathered from an old Actor model A^{old}_θ. The loss function of the Actor model by using the importance sampling is:

$$L_A = \mathbb{E}\left[\frac{A_\theta(a_t|s_t)}{A^{old}_\theta(a_t|s_t)}A_t\right]. \tag{7}$$

By using Kullback–Leibler divergence [Kullback, 1997] with a small value ϵ, we can optimize L_A subject to the constraint on the amount of the Actor model optimization:

$$\mathbb{E}[KL(A_\theta(\cdot|s_t), A^{old}_\theta(\cdot|s_t))] \leq \epsilon. \tag{8}$$

To simplify the implementation and improve the sampling efficiency, the loss function L_A can be given by a clipped surrogate objective function:

$$L_A = \mathbb{E}[min(R_t, clip(R_t, 1-\sigma, 1+\sigma))A_t], \tag{9}$$

where $R_t = \frac{A_\theta(a_t|s_t)}{A^{old}_\theta(a_t|s_t)}$ and σ is the clipping parameter that restricts the upper/lower bounds in the $clip(\cdot)$ function to stabilize the updating process. By using the clipped objective function, the PPO agent can quickly avoid actions with negative advantage values and does not greedily choose actions with positive advantage values.

3.2 Discriminator Design

Inspired by AIRL, we design a discriminator to learn a reward function from expert trajectories, which is robust to changes in dynamics and can guide the pre-training process of the agent. On the right part of Figure 2, the discriminator has two neural networks R and V. The solid-line arrows show the reward generation based on the agent trajectory $\langle s, a, s' \rangle$ and dashed-line arrows show the learned reward function optimization from the expert trajectory $\langle s_e, a_e, s'_e \rangle$.

In a trajectory-centric formulation proposed by [Finn *et al.*, 2016a], the discriminator D_θ is defined as follows:

$$D_\theta(\tau) = \frac{exp\{f_\theta(\tau)\}}{exp\{f_\theta(\tau)\} + \pi(\tau)}, \tag{10}$$

where τ is a sequence of states and actions induced by a policy and dynamics, f_θ is a learned function with its own parameter θ, π is the policy of agent trained to maximize the learned reward $R(\tau) = log(1 - D_\theta(\tau)) - logD_\theta(\tau)$. Note that, updating the discriminator can be viewed as updating the reward function, and updating the policy can be viewed as improving the sampling distribution used to estimate the partition function. If trained to optimality, it has been proved that an optimal reward function can be extracted from the optimal discriminator as $f^*(\tau) = R^*(\tau) + c$ and the policy π can recover the optimal policy, where c is a constant value. However, using the entire trajectory τ can result in high-variance estimates compared to using single state-action pairs.

In order to solve this issue, we convert Equation 10 into a single state and action case:

$$D_\theta(s, a) = \frac{exp\{f_\theta(s, a)\}}{exp\{f_\theta(s, a)\} + \pi(a|s)}. \tag{11}$$

Fu et al. [Fu *et al.*, 2018] proved that, at optimality, $f^*_\theta(s, a)$ is equal to the advantage function of the optimal policy $A^*(s, a)$:

$$f^*_\theta(s, a) = log\pi^*(a|s) = A^*(s, a). \tag{12}$$

Although this change results in an efficient algorithm for imitation learning, it is less desirable for the purpose of reward learning. This is because the advantage function is a heavily entangled reward and it supervises each action based on the action of the optimal policy, which is not robust to changes in environmental dynamics.

To decouple the reward function from the advantage, we modify the discriminator with the form:

$$D_\theta(s, a, s') = \frac{exp\{f_{R_\theta, V_\theta}(s, a, s')\}}{exp\{f_{R_\theta, V_\theta}(s, a, s')\} + A_\theta(a|s)}, \tag{13}$$

where the policy π is equivalent to the Actor model A_θ, f_{R_θ,V_θ} is restricted to a reward approximator R_θ and a shaping term S_θ with their own parameters θ:

$$f_{R_\theta,V_\theta}(s,a,s') = R_\theta(s,a) + \gamma V_\theta(s') - V_\theta(s). \quad (14)$$

The additional shaping term V_θ helps mitigate the effects of unwanted shaping on our reward approximator R_θ. The advantage of this approach is that $R_\theta(s)$ can be parameterized as a sole function of the state, which allows us to extract disentangled rewards from the dynamics of the environment. At optimality, it can recover a state-only ground-truth reward in this restricted case:

$$\begin{aligned} R^*_\theta &= R^*(s) + c, \\ V^*_\theta &= V^*(s) + c, \end{aligned} \quad (15)$$

where $R^*(\cdot)$ is the true reward function and $V^*(\cdot)$ is the optimal value function. This is because, by Equation 12, $f^*_{R_\theta,V_\theta}$ must recover to the advantage function:

$$\begin{aligned} f^*(s,a,s') &= A^*(s,a) = Q(s,a) - V(s) \\ &= R^*(s) + \gamma V^*(s') - V^*(s), \end{aligned} \quad (16)$$

where $f(s,a,s')$ can be considered as a single-sample estimate of $A^*(s,a)$ in stochastic environments and $Q(\cdot)$ is the Q-function. Therefore, R_θ and V_θ can recover to the ground-truth reward and optimal value function, respectively.

Since the objective of the discriminator is to classify expert and agent trajectories, we use a binary cross-entropy loss to train the discriminator. The loss function of the discriminator is $L_D = L_e + L_a$, where L_e and L_a are defined as follows:

$$\begin{aligned} L_e &= -\frac{1}{N}\sum_{i=1}^{N} y_{e_i} log(D_{G,H}(x_{e_i})) \\ &\quad +(1-y_{e_i}) log(1 - D_{G,H}(x_{e_i})), \\ L_a &= -\frac{1}{N}\sum_{i=1}^{N} y_{a_i} log(D_{G,H}(x_{a_i})) \\ &\quad +(1-y_{a_i}) log(1 - D_{G,H}(x_{a_i})), \end{aligned} \quad (17)$$

where x_{e_i} and x_{a_i} are expert and agent trajectory samples, respectively, y_{e_i} and y_{a_i} are labels that indicate whether the corresponding sample is from agent or expert trajectories, respectively, and N is the number of samples.

Algorithm 1 The Training Procedure for InitLight

Input: i) the number, E, of episodes, ii) the number, S, of episode steps; iii) expert trajectory buffer B_e; iv) agent trajectory buffer B_a; v) agent replay buffer B_r; vi) Actor model A_θ; vii) Critic model C_θ; viii) R model R_θ; ix) V model V_θ; x) discount factor γ, and; xi) batch size N.
Output: i) C_θ; ii) A_θ.

1: initialize B_a and B_r to empty;
2: **for** $episode = 1, 2, \cdots, E$ **do**
3: **for** $step = 1, 2, \cdots, S$ **do**
4: obtain the current state s;
5: choose the action a based on s;
6: execute action a in environment;
7: observe the next state s';
8: store agent trajectory $\langle s, a, s' \rangle$ in B_a;
9: compute $f \leftarrow R_\theta(s,a) + \gamma V_\theta(s') - V_\theta(s)$;
10: compute $d \leftarrow exp(f)/(exp(f) + A_\theta(a|s))$;
11: compute reward $r \leftarrow log(d) - log(1-d)$;
12: store trajectory $\langle s, a, s', r \rangle$ in B_r;
13: **if** $B_r.size() \geq N$ **then**
14: sample a mini-batch b_e of size N from B_e;
15: sample a mini-batch b_a of size N from B_a;
16: **for** $sample_e$ in b_e and $sample_a$ in b_a **do**
17: compute L_e and L_a by Equation 17;
18: compute $L_D \leftarrow L_e + L_a$;
19: update model parameters of R_θ and V_θ;
20: **end for**
21: sample a mini-batch b_r of size N from B_r;
22: **for** $sample_r$ in b_r **do**
23: compute L_C using Equation 3;
24: compute L_A using Equation 9;
25: update model parameters of C_θ and A_θ;
26: **end for**
27: **end if**
28: **end for**
29: **end for**
30: **return** trained models C_θ and A_θ;

3.3 InitLight Implementation

Algorithm 1 details the training process of InitLight. In lines 4-12, the PPO agent interacts with the environment and stores the trajectory, where the reward is generated by the discriminator. Lines 14-20 show the optimization of the discriminator. After the discriminator update, lines 21-26 update the parameters of the PPO agent. Finally, line 30 returns a trained PPO agent as the initial model to be applied in multi-intersection traffic scenarios.

4 Performance Evaluation

To evaluate the effectiveness of our approach, we conducted experiments on an Ubuntu server equipped with 3.7GHz Intel CPU, 32GB memory, and NVIDIA RTX 3080 GPU. We used the open-source traffic simulator Cityflow [Zhang *et al.*, 2019] to simulate both single- and multi-intersection environments. During the simulation of traffic scenarios, similar to the work in [Wei *et al.*, 2019a; Ye *et al.*, 2021], we set the phase duration of each signal light to 10 seconds. In InitLight, the PPO agent contains two neural networks, i.e., the Actor model with three layers (13, 32, and 8 neurons on each layer, respectively) and the Critic model with three layers (13, 64, and 1 neurons on each layer, respectively). The R model and V model of the discriminator have the same structure, i.e., three layers with 13, 32, and 1 neurons on each layer, respectively. We used the Adam optimizer for parameter updating and set the learning rate η to 0.0003 for all models. The discount factor γ is 0.99, and the GAE parameter λ is 0.95. The batch size N of the data sampled for training is 20, and the clipping parameter ϵ is 0.2. We designed comprehensive experiments to answer the following three research questions.

Type	Method	Average Travel Time (seconds)										
		S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11
Non-RL	FixedTime	385.16	556.05	412.41	209.13	165.86	217.05	186.41	247.63	502.89	458.88	535.41
	MaxPressure	112.31	**125.88**	103.34	93.22	87.20	91.79	93.23	95.32	106.23	98.95	103.63
RL	PressLight	125.24	151.28	**98.21**	**75.04**	**70.41**	78.67	**76.13**	**83.06**	108.11	**88.91**	97.58
	A2C	155.59	213.09	116.89	77.09	**71.34**	**76.00**	**75.91**	**84.55**	113.64	120.86	111.51
	FedLight	155.59	213.09	116.89	77.09	**71.34**	**76.00**	**75.91**	**84.55**	113.64	120.86	111.51
	PPO	**104.98**	134.44	98.87	82.32	73.29	**76.99**	79.28	94.83	**98.06**	92.92	**94.04**
	InitLight	**94.20**	**110.09**	**97.92**	**74.06**	73.75	77.80	80.91	84.85	**101.02**	**87.80**	**94.14**

Table 1: Comparison of average travel time on single-intersection datasets.

Type	Method	Average Travel Time (seconds)								
		Synthetic Datasets				Real-World Datasets				
		Syn1	Syn2	Syn3	Syn4	Hangzhou1	Hangzhou2	Jinan1	Jinan2	Jinan3
Non-RL	FixedTime	380.35	453.73	534.47	606.52	525.28	537.82	444.84	378.41	403.22
	MaxPressure	122.68	162.32	245.26	310.37	404.67	456.11	373.76	371.24	356.30
RL	PressLight	108.23	145.43	186.28	260.41	351.55	425.61	305.21	302.56	294.08
	A2C	117.60	**133.65**	236.58	433.85	**339.24**	416.08	375.12	322.92	288.92
	FedLight	108.71	136.45	**172.45**	**217.58**	341.94	**410.40**	**290.38**	**290.58**	**278.69**
	PPO	**105.86**	138.11	204.76	314.84	358.66	421.89	321.37	304.66	286.43
	InitLight	**102.86**	**126.44**	**172.36**	**237.52**	**333.80**	**374.60**	**297.58**	**293.81**	**284.24**

Table 2: Comparison of average travel time on multi-intersection datasets.

Method	Average Travel Time after the 1st Episode (seconds) / Start Episode # of Converge (#)								
	Synthetic Datasets				Real-World Datasets				
	Syn1	Syn2	Syn3	Syn4	Hangzhou1	Hangzhou2	Jinan1	Jinan2	Jinan3
PressLight	**487.54** (79)	**532.26** (123)	**759.62** (142)	**781.93** (157)	**472.46** (80)	**521.24** (61)	**541.44** (75)	**512.36 (76)**	**543.18** (82)
A2C	871.73 (95)	1306.89 (165)	1032.85 (N/A)	1315.05 (N/A)	1241.92 (122)	765.53 (65)	1298.90 (N/A)	1207.52 (105)	1224.99 (133)
FedLight	916.81 **(53)**	1175.63 **(111)**	1309.26 **(133)**	1357.66 **(59)**	1009.92 **(48)**	807.72 **(52)**	1152.66 **(47)**	1213.16 (110)	1229.11 **(70)**
PPO	873.82 (83)	1056.83 (165)	1127.96 (145)	1297.66 (N/A)	813.43 (111)	694.34 (100)	972.88 (110)	1031.68 (89)	869.87 (80)
InitLight	**115.73 (1)**	**164.63 (2)**	**202.51 (2)**	**262.45 (5)**	**330.78 (1)**	**387.06 (1)**	**300.42 (1)**	**294.18 (1)**	**288.42 (1)**

Table 3: Convergence information on multi-intersection datasets.

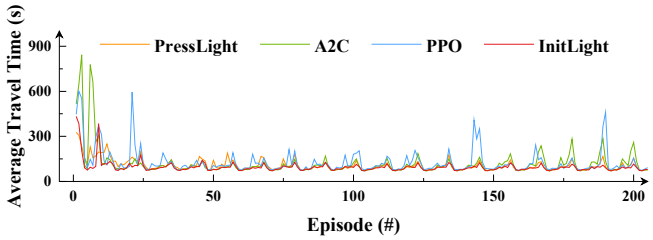

Figure 3: Convergence comparison for single-intersection datasets.

RQ1 (Effectiveness): Can InitLight quickly learn from expert trajectories of single-intersection environments and form a corresponding high-quality TSC policy?

RQ2 (Generalizability): Can the agent pre-trained in RQ1 be deployed in arbitrary complex multi-intersection environments to benefit their RL training processes?

RQ3 (Benefits): Why can our proposed InitLight substantially improve the learning performance and generalization ability of RL models?

4.1 Experimental Settings

Baselines. Since the agent design of our InitLight approach is based on the concept of the pressure, for fair comparison, we chose six representative pressure-based baseline methods, including two classic Non-RL methods and four RL-based methods as follows: i) **FixedTime** [Koonce and Rodegerdts, 2008], a non-RL control method that cyclically selects control phases based on a predefined phase sequence; ii) **MaxPressure** [Varaiya, 2013], a heuristic MP-based control method that greedily selects the phase with the maximum pressure; iii) **PressLight** [Wei *et al.*, 2019a], a deep RL-based control method controlling traffic lights intelligently based on the MP control theory; iv) **A2C** [Ye *et al.*, 2021], a TSC method that controls one intersection by an individual Advantage Actor-Critic (A2C) agent; v) **FedLight** [Ye *et al.*, 2021], a state-of-the-art federated reinforcement learning method that enables the sharing of the knowledge among RL agents; and vi) **PPO** [Schulman *et al.*, 2017], a deep RL-based method that uses the PPO algorithm to control traffic lights.

Datasets. We tested twenty public datasets (i.e., environments) provided by [Wei *et al.*, 2019b], including eleven real-world single-intersection datasets (i.e., S1-S11), four synthetic multi-intersection datasets (i.e., Syn1-Syn4), and five real-world multi-intersection datasets (i.e., Hangzhou1, Hangzhou2, and Jinan1-Jinan3). For all the investigated datasets, each intersection of all the road networks has four incoming roads and four outgoing roads, where each road has three lanes, i.e., turning left, going straight, and turning right. The details of the datasets are as follows:

- **Real-world single-intersection datasets**: We considered 11 real-world single-intersection datasets (i.e., S1-S11), which are captured by cameras deployed in the city of Hangzhou. Based on the collected taxi statistics in Hangzhou, we set the turning ratios of these datasets to 10%, 60%, and 30% for turning left, going straight, and turning right, respectively.
- **Synthetic multi-intersection datasets**: Synthetic

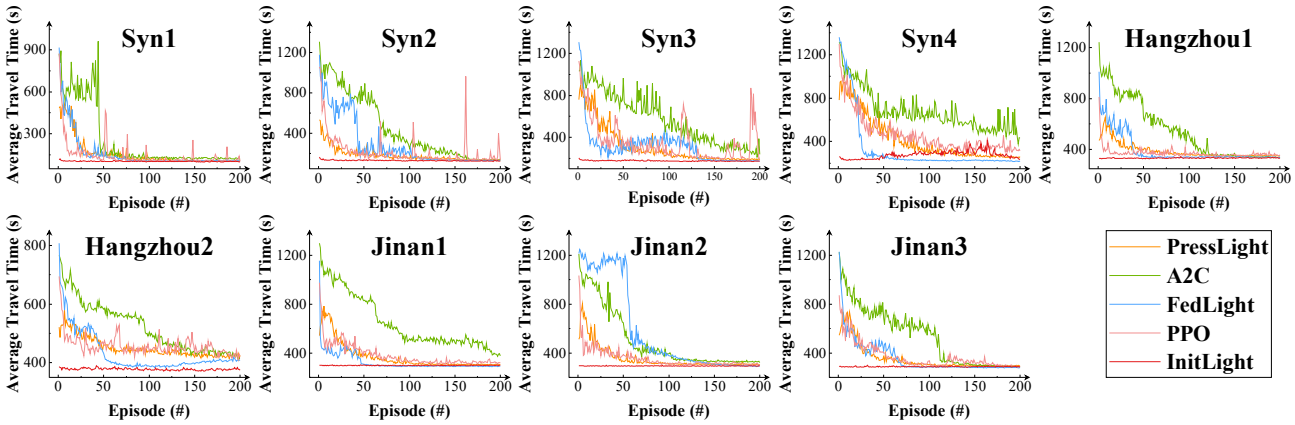

Figure 4: Convergence rates in multi-intersection datasets.

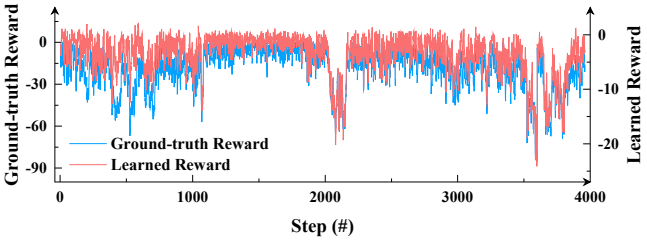

Figure 5: Learned rewards on single-intersection datasets.

datasets, Syn1-Syn4, contain 1×3, 2×2, 3×3, and 4×4 intersections, respectively. The vehicle arrival rates of each dataset follow a Gaussian distribution with an average of 500 vehicles per hour for each entry lane.

- **Real-world multi-intersection datasets**: We investigated five real-world datasets (i.e., Hangzhou1, Hangzhou2, Jinan1, Jinan2, and Jinan3), which are collected by the cameras deployed in the cities of Hangzhou (Gudang sub-district) and Jinan (Dongfeng sub-district). Note that each multi-intersection dataset collected from Hangzhou has 12 (3×4) intersections, and each multi-intersection dataset collected from Jinan has 16 (4×4) intersections.

Since the diversity of different datasets is crucial for our experiments of generalization ability, we used eleven single-intersection datasets, S1-S11, to pre-train the initial PPO agent and nine multi-intersection datasets, Syn1-Syn4, Hangzhou1-Hangzhou2, and Jinan1-Jinan3, to evaluate the transfer performance of the initial model generated by our InitLight approach.

4.2 Performance of Imitation Learning (RQ1)

To answer RQ1, we considered InitLight as an imitation learning method and compared it against the six baseline methods on single-intersection datasets. For each RL method, we trained its model on the eleven single-intersection datasets (i.e., S1-S11) in a round-robin fashion, where the training lasted 20 times. Therefore, the whole training process involved 220 episodes in total. We collected the expert trajectories for each single-intersection dataset, which were generated by the TSC method MaxPressure [Wei *et al.*, 2019a]. The reason why we chose MaxPressure here is that MaxPressure can achieve competitive results compared with state-of-the-art pressure-based methods.

Table 1 shows the comparison results for different control methods in terms of average travel time over S1-S11. For each dataset, the TSC methods with the best or second-best performance are highlighted in bold. From this table, we can find that InitLight achieves the best performance, since it achieves top-2 best average time in 7 out of 11 datasets. Since InitLight adopts the same RL model as PPO, due to the guidance of expert trajectories, we can find that InitLight significantly outperforms PPO by more than 10% in four datasets (i.e., S1, S2, S4, and S8). Note that FedLight and A2C have the same results. This is because the clients of FedLight are implemented on top of A2C models, where FedLight is equivalent to A2C when dealing with only one single intersection.

Figure 3 compares the convergence performance of all the RL-based methods on datasets S1-S11. Since the training is in a round-robin fashion, we can observe periodic fluctuations along with the convergence curves. Since FedLight and A2C are the same for single-intersections, we omit the result for FedLight here. From this figure, we can find that, compared with the three baselines, our InitLight approach can achieve the quickest convergence and lowest average travel time for the given 11 single-intersection datasets. According to the convergence criterion used by [Zhao *et al.*, 2022; Fang *et al.*, 2022], the three baselines need more training episodes to converge (e.g., PressLight converges at the 88^{th} episode, whereas InitLight needs only 22 episodes to converge) and have more significant fluctuations (without the guidance of expert trajectories).

4.3 Performance of Transfer Learning (RQ2)

To evaluate the generalization ability of InitLight, we considered nine transfer tasks that adopt the pre-trained PPO agent in various multi-intersection datasets, where each intersection is equipped with one pre-trained PPO agent. For a specific multi-intersection dataset, we trained each RL with 200 episodes. Table 2 presents the TSC performance of all

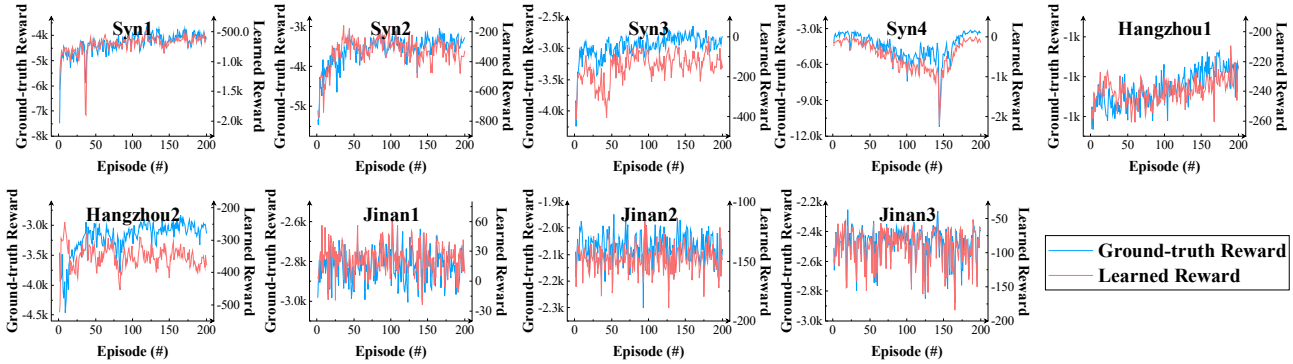

Figure 6: Learned rewards on multi-intersection datasets.

the investigated TSC methods on multi-intersection datasets. Similar to Table 1, we highlight the best and second-best results in bold for each dataset. From this table, we can find that RL-based methods can achieve much better performance than non-RL methods. For example, although MaxPressure and PressLight are based on MP control theory, PressLight outperforms MaxPressure on all datasets as it adopts RL to control traffic signals more intelligently. Compared with the four RL-based baselines, our InitLight method can achieve top-2 performance on all nine datasets.

Figure 4 evaluates the convergence performance of the RL-based methods on the nine multi-intersection datasets. Note that here only InitLight adopted the same pre-trained model at intersections for all the multi-intersection datasets, while all the baselines adopt randomly initialized models at intersections. Compared to all the four baseline methods, our InitLight approach achieves almost the lowest average travel time at the beginning of RL training for all the datasets with much fewer fluctuations. Note that, for all the investigated datasets, InitLight can converge within 5 episodes, which is much quicker than all the baselines. Moreover, InitLight can always achieve the top-2 lowest average travel time for all the multi-intersection datasets eventually. The above facts evidently reveal the generalization ability of the pre-trained agent by InitLight.

To better understand the merits of InitLight, Table 3 shows the detailed convergence information of different RL-based TSC methods from the perspective of jumpstart performance (i.e., average travel time after the first episode) and the starting episode of the convergence, where the best and second-best results are marked in bold. Here, we used the criterion of convergence following [Zhao *et al.*, 2022; Fang *et al.*, 2022]. From this table, we can find that InitLight achieves the best jumpstart performance and the fastest convergence on all datasets. After the first episode, the average travel time of InitLight over all the datasets approximates the final results as shown in Table 2. It means that InitLight can almost converge in the first episode, which again confirms the generalization ability of pre-trained agents.

4.4 Quality of Learned Reward Function (RQ3)

To understand why InitLight can improve both the effectiveness and generalization ability of RL models, we compared the rewards learned by our discriminator and the ground-truth rewards calculated by $r = -P_I$ according to the MP control theory. Figure 5 compares the ground-truth rewards with the learned rewards for single-intersection datasets in the last round (i.e., the 20^{th} round) of pre-training. Note that here the horizontal axis indicates the step indices, where each episode (with one-hour simulation) of a dataset involves 360 steps, since a reward is calculated at the end of each control phase (with 10 seconds). From this figure, we can find that the learned reward trend approximates the ground-truth reward trend. The reason for such consistency is mainly because InitLight can learn an accurate reward function from expert trajectories to guide the pre-training process of the agent.

Figure 6 compares the average accumulative learned rewards with the average accumulative ground-truth rewards of all the intersections obtained within each episode. Note that, for a dataset, a reward of an episode is the summation of 360 rewards calculated within the same episode for all the dataset intersections. From this figure, we can find that the learned reward trends are similar to the ground-truth reward trends for all the datasets. It means that the discriminator learned by InitLight can recover the ground-truth reward function from expert trajectories, which is robust to various kinds of complex environments.

5 Conclusions

To address the problems of slow convergence and poor model generalization ability existing in Reinforcement Learning (RL)-based Traffic Signal Control (TSC), this paper proposed a novel AIRL-based method named InitLight, which enables the pre-training of an initial model to accelerate the global training process of RL-based TSC methods. Since InitLight trains the initial model only on a limited set of single-intersection environments, the generation time of the initial model is negligible. Meanwhile, since the reward function learned by InitLight can recover the ground-truth TSC rewards at optimality for different intersections, the obtained agent by InitLight can be deployed on different kinds of complex traffic environments to accelerate their overall RL processes. Comprehensive experimental results validate the effectiveness of InitLight.

Acknowledgments

This work was supported by the Natural Science Foundation of China (62272170), Shanghai Trusted Industry Internet Software Collaborative Innovation Center, "Digital Silk Road" Shanghai International Joint Lab of Trustworthy Intelligent Software (22510750100), and Natural Science Foundation (NSF CCF-2217104). Mingsong Chen is the corresponding author (mschen@sei.ecnu.edu.cn).

References

[Abbeel and Ng, 2004] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1–8, 2004.

[Boularias *et al.*, 2011] Abdeslam Boularias, Jens Kober, and Jan Peters. Relative entropy inverse reinforcement learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (ICAIS)*, pages 182–189, 2011.

[Chen *et al.*, 2020a] Chacha Chen, Hua Wei, Nan Xu, Guanjie Zheng, Ming Yang, Yuanhao Xiong, Kai Xu, and Zhenhui Li. Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3414–3421, 2020.

[Chen *et al.*, 2020b] Lisi Chen, Shuo Shang, Bin Yao, and Jing Li. Pay your trip for traffic congestion: Dynamic pricing in traffic-aware road networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 582–589, 2020.

[Chen *et al.*, 2022] Rex Chen, Fei Fang, and Norman Sadeh. The real deal: A review of challenges and opportunities in moving reinforcement learning-based traffic signal control systems towards reality. In *Proceedings of the IJCAI Workshop on Agents in Traffic and Transportation Co-located*, pages 14–31, 2022.

[Choi and Kim, 2013] Jaedeug Choi and Kee-Eung Kim. Bayesian nonparametric feature construction for inverse reinforcement learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1287–1293, 2013.

[Cools *et al.*, 2013] Seung-Bae Cools, Carlos Gershenson, and Bart D'Hooghe. Self-organizing traffic lights: A realistic simulation. In *Advances in applied self-organizing systems*, pages 45–55. Springer, 2013.

[de Almeida *et al.*, 2022] Vicente N de Almeida, Ana LC Bazzan, and Monireh Abdoos. Multiagent reinforcement learning for traffic signal control: a k-nearest neighbors based approach. In *Proceedings of the IJCAI Workshop on Agents in Traffic and Transportation Co-located*, pages 32–46, 2022.

[Fang *et al.*, 2022] Zekuan Fang, Fan Zhang, Ting Wang, Xiang Lian, and Mingsong Chen. Monitorlight: Reinforcement learning-based traffic signal control using mixed pressure monitoring. In *Proceedings of International Conference on Information & Knowledge Management (CIKM)*, pages 478–487, 2022.

[Finn *et al.*, 2016a] Chelsea Finn, Paul Christiano, Pieter Abbeel, and Sergey Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*, 2016.

[Finn *et al.*, 2016b] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 49–58, 2016.

[Fu *et al.*, 2018] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adverserial inverse reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, pages 1–15, 2018.

[Ho and Ermon, 2016] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 4565–4573, 2016.

[Hunt *et al.*, 1982] PB Hunt, DI Robertson, RD Bretherton, and M Cr Royle. The scoot on-line traffic signal optimisation technique. *Traffic Engineering & Control*, 23(4), 1982.

[Jiang *et al.*, 2021] Qize Jiang, Jingze Li, Weiwei Sun SUN, and Baihua Zheng. Dynamic lane traffic signal control with group attention and multi-timescale reinforcement learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3642–3648, 2021.

[Jiang *et al.*, 2022] Qize Jiang, Minhao Qin, Shengmin Shi, Weiwei Sun, and Baihua Zheng. Multi-agent reinforcement learning for traffic signal control through universal communication method. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3854–3860, 2022.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations (ICLR)*, pages 1–15, 2014.

[Koohy *et al.*, 2022] Behrad Koohy, Sebastian Stein, Enrico Gerding, and Ghaithaa Manla. Reward function design in multi-agent reinforcement learning for traffic signal control. In *Proceedings of the IJCAI Workshop on Agents in Traffic and Transportation Co-located*, pages 1–13, 2022.

[Koonce and Rodegerdts, 2008] Peter Koonce and Lee Rodegerdts. Traffic signal timing manual. Technical report, United States. Federal Highway Administration, 2008.

[Kullback, 1997] Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.

[Ng *et al.*, 2000] Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 663–670, 2000.

[PR, 1992] Lowrie PR. Scats: A traffic responsive method of controlling urban traffic control/pr lowrie. *Roads and Traffic Authority*, 1992.

[Ramachandran and Amir, 2007] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2586–2591, 2007.

[Ratliff *et al.*, 2006] Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. Maximum margin planning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 729–736, 2006.

[Rizzo *et al.*, 2019] Stefano Giovanni Rizzo, Giovanna Vantini, and Sanjay Chawla. Time critic policy gradient methods for traffic signal control in complex and congested scenarios. In *Proceedings of the International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, pages 1654–1664, 2019.

[Russell, 1998] Stuart Russell. Learning agents for uncertain environments. In *Proceedings of the Annual Conference on Computational Learning Theory (ACCLT)*, pages 101–103, 1998.

[Schulman *et al.*, 2015] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *Proceedings of the International Conference on Learning Representations (ICLR)*, pages 1–14, 2015.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[Tesauro and others, 1995] Gerald Tesauro et al. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

[Török and Kertész, 1996] János Török and János Kertész. The green wave model of two-dimensional traffic: Transitions in the flow properties and in the geometry of the traffic jam. *Physica A: Statistical Mechanics and its Applications*, 231(4):515–533, 1996.

[Varaiya, 2013] Pravin Varaiya. The max-pressure controller for arbitrary networks of signalized intersections. In *Advances in dynamic network modeling in complex transportation systems*, pages 27–66. Springer, 2013.

[Wei *et al.*, 2019a] Hua Wei, Chacha Chen, Guanjie Zheng, Kan Wu, Vikash Gayah, Kai Xu, and Zhenhui Li. Presslight: Learning max pressure control to coordinate traffic signals in arterial network. In *Proceedings of the International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, pages 1290–1298, 2019.

[Wei *et al.*, 2019b] Hua Wei, Guanjie Zheng, Vikash Gayah, and Zhenhui Li. A survey on traffic signal control methods. *arXiv preprint arXiv:1904.08117*, 2019.

[Xing *et al.*, 2022] Dong Xing, Qian Zheng, Qianhui Liu, and Gang Pan. Tinylight: Adaptive traffic signal control on devices with extremely limited resources. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3999–4005, 2022.

[Xiong *et al.*, 2019] Yuanhao Xiong, Guanjie Zheng, Kai Xu, and Zhenhui Li. Learning traffic signal control from demonstrations. In *Proceedings of the International Conference on Information & Knowledge Management (CIKM)*, pages 2289–2292, 2019.

[Ye *et al.*, 2021] Yutong Ye, Wupan Zhao, Tongquan Wei, Shiyan Hu, and Mingsong Chen. Fedlight: Federated reinforcement learning for autonomous multi-intersection traffic signal control. In *Proceedings of the Design Automation Conference (DAC)*, pages 847–852, 2021.

[Ye *et al.*, 2022] Yutong Ye, Jiepin Ding, Ting Wang, Junlong Zhou, Xian Wei, and Mingsong Chen. Fairlight: Fairness-aware autonomous traffic signal control with hierarchical action space. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022.

[Zang *et al.*, 2020] Xinshi Zang, Huaxiu Yao, Guanjie Zheng, Nan Xu, Kai Xu, and Zhenhui Li. Metalight: Value-based meta-reinforcement learning for traffic signal control. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 1153–1160, 2020.

[Zhang *et al.*, 2019] Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin, and Zhenhui Li. Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In *Proceedings of the World Wide Web Conference (WWW)*, pages 3620–3624, 2019.

[Zhang *et al.*, 2020] Huichu Zhang, Chang Liu, Weinan Zhang, Guanjie Zheng, and Yong Yu. Generalight: Improving environment generalization of traffic signal control via meta reinforcement learning. In *Proceedings of the International Conference on Information & Knowledge Management (CIKM)*, pages 1783–1792, 2020.

[Zhang *et al.*, 2021] Liang Zhang, Qiang Wu, and Jianming Deng. Knowledge intensive state design for traffic signal control. *arXiv preprint arXiv:2201.00006*, 2021.

[Zhao *et al.*, 2022] Wupan Zhao, Yutong Ye, Jiepin Ding, Ting Wang, Tongquan Wei, and Mingsong Chen. Ipdalight: Intensity-and phase duration-aware traffic signal control based on reinforcement learning. *Journal of Systems Architecture*, 123:102374, 2022.

[Zheng *et al.*, 2021] Guanjie Zheng, Hanyang Liu, Kai Xu, and Zhenhui Li. Objective-aware traffic simulation via inverse reinforcement learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3771–3777, 2021.

[Zhu *et al.*, 2021] Liwen Zhu, Peixi Peng, Zongqing Lu, Xiangqian Wang, and Yonghong Tian. Variationally and intrinsically motivated reinforcement learning for decentralized traffic signal control. *arXiv preprint arXiv:2101.00746*, 2021.

[Ziebart *et al.*, 2008] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 1433–1438, 2008.