

MARLens: Understanding Multi-agent Reinforcement Learning for Traffic Signal Control via Visual Analytics

Yutian Zhang, Guohong Zheng, Zhiyuan Liu, Quan Li and Haipeng Zeng

Abstract—The issue of traffic congestion poses a significant obstacle to the development of global cities. One promising solution to tackle this problem is intelligent traffic signal control (TSC). Recently, TSC strategies leveraging reinforcement learning (RL) have garnered attention among researchers. However, the evaluation of these models has primarily relied on fixed metrics like reward and queue length. This limited evaluation approach provides only a narrow view of the model's decision-making process, impeding its practical implementation. Moreover, effective TSC necessitates coordinated actions across multiple intersections. Existing visual analysis solutions fall short when applied in multi-agent settings. In this study, we delve into the challenge of interpretability in multi-agent reinforcement learning (MARL), particularly within the context of TSC. We propose *MARLens*, a visual analytics system tailored to understand MARL-based TSC. Our system serves as a versatile platform for both RL and TSC researchers. It empowers them to explore the model's features from various perspectives, revealing its decision-making processes and shedding light on interactions among different agents. To facilitate quick identification of critical states, we have devised multiple visualization views, complemented by a traffic simulation module that allows users to replay specific training scenarios. To validate the utility of our proposed system, we present three comprehensive case studies, incorporate insights from domain experts through interviews, and conduct a user study. These collective efforts underscore the feasibility and effectiveness of *MARLens* in enhancing our understanding of MARL-based TSC systems and pave the way for more informed and efficient traffic management strategies.

Index Terms—Traffic signal control, multi-agent, reinforcement learning, visual analytics.

1 INTRODUCTION

TRAFFIC congestion is a critical and challenging issue in modern urban development, which leads to significant financial losses and detrimental emissions [1], [2]. The implementation of intelligent traffic signal control (TSC) presents a practical approach to mitigating traffic congestion. This strategy is a fundamental element of smart cities and has garnered considerable attention from both industrial and government sectors globally, with an aim to identify efficient TSC methods.

Despite the widespread adoption of certain adaptive TSC systems, their effectiveness remains heavily reliant on traffic flow models and expert knowledge. This reliance poses limitations on their capacity to dynamically adjust to real-time traffic conditions [3], [4]. In recent times, reinforcement learning (RL) has emerged as a promising approach to resolving TSC issues, owing to its ability to learn optimal TSC strategies through trial-and-error without being constrained by pre-set rules [5]. While prior research has dedicated efforts to evaluating the performance of RL-based TSC techniques for managing traffic at multiple intersections [6], [7], [8], there has been a notable lack of focus on the decision-making process employed by these models. Additionally, previous studies have primarily relied on a limited set of

metrics, such as reward, queue length, and average travel time, to assess the efficacy of TSC models. This approach often lacks the depth required to comprehend the intricate learning mechanisms inherent to RL-based methods. It is essential to note that in real-world scenarios, the trial-and-error nature of TSC models can potentially lead to severe, and in some cases, fatal consequences. Therefore, the development of more interpretable RL models and a comprehensive understanding of the decision-making processes within TSC models are imperative steps towards enhancing their reliability and safety.

The field of interpretability in machine learning (ML) is rapidly evolving within the broader scope of artificial intelligence. Visual analytics has emerged as an invaluable tool to enhance the interpretability of ML models [9]. While there have been extensive studies to unravel the workings of various deep learning models such as Convolutional Neural Networks (CNNs) [10], [11], [12], Recurrent Neural Networks (RNNs) [13], [14] and Generative Adversarial Networks (GANs) [15], [16], [17] using visualization techniques, there have been relatively few attempts to apply these methods to enhance the interpretability of RL. This gap in research primarily stems from the limited exploration of RL models and scenarios [18]. Researchers have thus far predominantly employed video games [19], [20], [21], robot control [22], [23], [24], and simple 2D environments [25], [26] as their primary domains for visualizing RL models. However, one notable omission in this body of work has been the domain of TSC based on RL, which has not received the attention it deserves. Additionally, prior research

- Y. Zhang, G. Zheng, Z. Liu are with Sun Yat-sen University. E-mail: {zhangyt85, zhenggh8, liuzhy253}@mail2.sysu.edu.cn.
- Q. Li is with the School of Information Science and Technology, ShanghaiTech University. E-mail: liquan@shanghaitech.edu.cn.
- H. Zeng is with Sun Yat-sen University and is the corresponding author. E-mail: zenghp5@mail.sysu.edu.cn.

Manuscript received March 31, 2023; revised December 07, 2023.

has primarily leveraged single-agent RL models as the backend for visualization, whereas TSC operates on multi-agent reinforcement learning (MARL) systems. The utilization of multi-agent systems entails more than just an increase in the number of agents; it also necessitates addressing the intricate interactions that can occur between these agents. These complex interactions present significant challenges and can profoundly impact the overall performance of the system. While some visual analytics systems have been proposed to explain MARL [27], [28], these solutions may not be directly applicable to TSC settings. To the best of our knowledge, no visual analytics solutions tailored explicitly to the needs of both **RL** and **TSC researchers** have delved into the intricacies of comparing multiple RL agents within the context of TSC. Closing this research gap requires the development of advanced visualization tools capable of shedding light on the inner workings of RL models and their applications in TSC, especially in multi-agent scenarios.

Nonetheless, three significant challenges must be addressed to gain a comprehensive understanding of RL-based TSC methods. **(1) Providing a Complete Assessment of RL Models.** In transportation research, RL model performance is typically assessed from a holistic road network perspective [5], often employing commonly used metrics like reward and queue length. Yet, these metrics do not reveal the underlying rationale behind each agent's actions within a given state. This underscores the pressing need for a complete assessment of RL models. **(2) Unveiling Dynamic Relationships in Multi-Agent TSC.** In a multi-agent setting, a single agent's actions can significantly impact the performance of other agents [29], consequently influencing the overall model performance. The dynamic interactions among multiple agents make it challenging to examine the intricate relationships among them, including their collaboration and contribution to the model's performance. **(3) Interpreting Complex Decision-making Process of RL-based TSC.** An RL model comprises various components, including agents, states, and actions. Each agent engages with the environment, learning from its experiences. However, the behaviors of these agents can lead to unpredictable collective patterns that have a substantial impact on the overall model performance. As a result, articulating and interpreting the decision-making process of the model at various levels becomes challenging.

To tackle the aforementioned challenges, we propose a visual analytics system designed to enhance the exploration and interpretability of RL-based TSC models. Our approach initiates with a comprehensive requirement analysis, identifying the essential needs and concerns of domain experts. We then carefully select a set of metrics, drawing from both prior research and specific requirements, to extract detailed insights that characterize RL models comprehensively. Subsequently, we apply Shapley values to pinpoint important features for different agents and train interpretable models that unveil decision-making rules governing the relationships between states and the actions of each agent. Additionally, we perform a multi-level analysis (episode level, time-period level and time-step level) to gain a deeper understanding of the decision-making process in RL-based TSC. To validate our approach, we conduct three distinct case studies and expert interviews. In summary, the key

contributions of our work can be succinctly outlined as follows:

- We begin by examining the requirements of domain experts in the context of TSC and identify key factors while also delving into the interpretability of RL-based TSC models.
- We introduce a comprehensive visual analysis system designed to facilitate an in-depth exploration of MARL models within TSC scenarios. This system allows users to evaluate MARL models from a variety of angles, enhancing their understanding.
- We employ a combination of methods including case studies, expert interviews, and a user study to assess the effectiveness and user-friendliness of our system, thereby providing a well-rounded evaluation of our approach.

2 RELATED WORK

This section presents two relevant topics, *Reinforcement Learning for Traffic Signal Control* and *Interpretability of Reinforcement Learning*.

2.1 Reinforcement Learning for Traffic Signal Control

The traditional pre-set signal control scheme struggles to efficiently handle complex traffic situations. As a result, researchers have proposed adaptive signal control methods that can dynamically adjust signal timing to alleviate traffic congestion [30], [31]. RL has emerged as one of the most promising techniques for TSC. It allows control strategies to adapt in real-time based on the current traffic conditions, making it a valuable approach [5].

In early time, the application of RL-based TSC often involved simplifying the modeling process by focusing on a single intersection. For instance, Shashi et al. [32] employed the Deep Q-Network (DQN) algorithm to adjust traffic signals, making decisions based on LSTM-generated suboptimal actions. Wei et al. [6] enhanced the DQN model by incorporating a phase-gated learning model, providing better insights into how the model adapts to real traffic conditions. Many studies have reported favorable results when applying RL control to individual intersections [7], [33], [34]. However, extending this approach to address multi-intersection challenges presents difficulties. It necessitates the coordination of control models for each intersection and involves coping with the high dimensionality resulting from multiple data sources. To tackle this, Wang et al. [35] employed a directional adjacency graph to model the collaborative mechanism among multiple signal lights. Xu et al. [36] introduced graph attention networks for the first time in TSC scenarios, enhancing information exchange among agents.

While prior studies have significantly advanced the field of TSC, a common shortcoming is the insufficient explanation of their models. Bridging the gap between these advancements and practical applications requires researchers to provide compelling justifications for their methodologies. Hence, the primary goal of this study is to not only contribute to the progress in TSC but also to address the need for enhanced user understanding of the model's decision-making process.

2.2 Interpretability of Reinforcement Learning

In recent years, there has been a growing interest in enhancing the interpretability of artificial intelligence (AI) research. Most of this research has been concentrated on improving the interpretability of ML models, particularly in the domains of classification, decision-making, and action selection, as documented in studies such as [18], [37], [38]. However, compared to ML, achieving interpretability in the context of RL poses a more intricate challenge. RL relies on feedback mechanisms to guide agents in learning optimal actions within their environment, rather than solely relying on predefined training data.

Significant progress has been made in enhancing the interpretability of RL models. However, there exists a noticeable gap in research focusing on RL-based TSC scenarios. To address this, we introduced a taxonomy of related work inspired by Stephanie et al. [39], outlining three key approaches to boost the interpretability of RL models: **1) Analyzing Feature Importance:** This approach strives to identify the features influencing agents' actions for a given state. Hayes et al. [40] directly generated explanations by posing questions like "when do you do?" to gain insights into the policies employed by RL agents. Rizzo et al. [41] pioneered the use of SHapley Additive exPlanations (SHAP) [42] in RL-based TSC, examining the relationship between traffic conditions and agent behaviors. **2) Analyzing the Learning Process and Markov Decision Process (MDP):** This approach breaks down rewards, decomposing them to gain insights into an agent's action preferences. For instance, Zhang et al. [43] proposed the Shapley Q-value algorithm, which dissects the global reward into local components, facilitating optimal decision-making in a multi-agent context. **3) Analyzing the Policy Level:** This approach illustrates the long-term behavior of the agent. For instance, Topin et al. [44] introduced Abstract Policy Graphs to summarize policies and explain agents' decisions. Guo et al. [45] focused on generating action advice, using explanations to enhance the transfer of suboptimal strategies during learning in multi-agent scenarios. Wollenstein-Betech et al. [46] utilized Knowledge Compilation and the d-DNNF language to generate the decision process of the traffic light controller. Schreiber et al. [47] also applied SHAP to explain how features of the road network influence the agent's action in a given state. However, it's important to note that these studies still face some limitations. First, they do not support multi-level analysis, which could not provide more comprehensive interpretability solutions. Moreover, there is a need to provide more effective visual techniques and interaction functions to strengthen the scalability and level of comprehension [18].

To enhance the interpretability of RL models, an increasing number of visualization techniques are being employed. For instance, Wang et al. [20] used an interactive saliency map to elucidate the learned strategies of agents in Atari games. Likewise, Gou et al. [19] conducted a visual analysis of the DQN algorithm, providing detailed insights into DQN models at four different levels to aid comprehension. They also proposed a visual design for representing time series data generated by the model. Moreover, He et al. [22], Jaunet et al. [48], and Wang et al. [20] introduced systems

designed to assist users in exploring, interpreting, and diagnosing RL models based on RNNs. Additionally, Mishra et al. [26] developed PolicyExplainer, a tool aimed at answering common RL policy-related questions. However, these research predominantly focuses on visually interpreting single-agent behavior in 2D games, which is not adequate for analyzing MARL models in complex traffic scenarios.

Recently, there has been an emerging interest in exploring the interpretability of MARL through visual analytics. For instance, Kravaris et al. [27] introduced a visual component designed to facilitate the exploration of MARL-based air traffic flow management. Additionally, Shi et al. [28] presented a visual analytics system aimed at analyzing the training process of the MARL model in a grounded communication environment within a 2D world. However, these studies cannot be directly applied to our specific target scenario. Primarily, these works concentrate on demonstrating data collected directly during the model training phase and lack the in-depth analysis needed to unveil the decision-making process of the model. Furthermore, the training environment in the prior systems is fundamentally distinct from that of TSC. In TSC scenarios, researchers frequently resort to utilizing synthetic road networks [49]. This necessitates the provision for integrating a customizable traffic simulation module into the backend of the visual analytics system.

This study focuses on gaining insights into MARL in the context of TSC. Our approach involves introducing a visual analytics approach designed to clarify the model's behavior across different levels. This spans explanations at the time-step level, extending to the episode level. The approach involves generating visual summaries for individual agents at specific episodes, illustrating relationships among multiple agents, and presenting the decision-making process that links actions and states.

3 PRELIMINARIES

In this section, we clarify how we employ RL to model the TSC problem and introduce the RL algorithm tailored to our particular scenario.

3.1 RL-based TSC Scenario

For an RL-based TSC scenario, each intersection of a road network functions as an agent. Further details, along with specific specifications and parameters, are presented below:

Definition of Episodes. Episodes are composed of sequential time steps that detail the interaction between the agent and the environment. During each time step, the agent receives observations of the current state of the environment, takes a corresponding action, and consequently transitions the environment to a new state. The agent is then rewarded based on its executed action.

Definition of States. In the context of the TSC problem, there have been various proposals for defining the environment state, encompassing factors like waiting time, queue length, and vehicle location. However, recent research, as exemplified in studies such as [7], [50], has suggested that the incorporation of complex state definitions may not necessarily yield substantial improvements in performance.

Instead, it is advisable to employ simpler state definitions, such as queue length, to effectively capture the environmental conditions in the TSC problem. Therefore, in this study, we have chosen to define the state using queue length.

Reward Function. While the fundamental goal of TSC is to reduce the average travel time for all vehicles, using travel time directly as a metric in the reward function is not deemed suitable. This is because travel time doesn't immediately reflect the impact of TSC, and vehicle movements can influence their travel time in complex ways [5]. Instead, a widely adopted factor in the reward function by TSC researchers is queue length [51]. To address this concern, we have incorporated the queue length of the entire road network as a component of the reward function. Larger queue lengths signify more congested road networks, resulting in lower rewards. Furthermore, from a safety standpoint, it is preferable to minimize the frequency of phase changes by traffic lights and extend the duration of the current phase. Therefore, we have also taken into account the frequency of phase changes when designing the reward function. The reward function for the i^{th} intersection (agent) is formulated as follows:

$$\text{Reward}_i = -(\omega_{\text{queue}} \text{Queue}_i + \omega_{\text{phase}} \delta_i) \quad (1)$$

In the reward function, Queue_i represents the queue length of the i^{th} agent, which is the cumulative queue length of its entries. Additionally, δ_i is a binary variable used to penalize phase changes. When an agent switches its current phase, δ_i is set to 1; otherwise, it remains 0. The frequency of phase changes by an agent determines the extent of penalty it incurs over a given time period. The hyperparameters ω_{queue} and ω_{phase} are introduced, with values set at $\omega_{\text{queue}} = 1$ and $\omega_{\text{phase}} = 2$ in this study, based on empirical findings.

Definition of Actions. In TSC, researchers commonly adopt one of two action settings. The first approach involves maintaining or altering the current phase, which defines the allowed traffic movements at the intersection, within a pre-defined and fixed phase sequence [52]. Conversely, the second approach entails selecting a specific phase from a pre-defined yet variable phase set [53]. Given that the first action setting closely mirrors real-world scenarios, we have selected this approach to demonstrate the efficacy of our model. In our designed scenario, the phase sequence consists of two types of phases: "allow North-South (N-S) to pass" and "allow West-East (W-E) to pass". Furthermore, to minimize frequent and abrupt traffic light changes, each green light phase has been set to a duration of 10 seconds. As previously mentioned, to mitigate potential hazards arising from uncertain and frequent traffic light transitions, whenever an agent switches its current phase to a different one, the intersection's phase remains fully red for the initial three seconds of the 10-second interval, allowing the intersection to clear. In summary, the action is limited to a binary value of 0 or 1, corresponding to the index of a phase in the predefined phase set.

3.2 Multi-Agent Deep Deterministic Policy Gradient

The Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [54] is a MARL algorithm developed by Ope-

nAI¹. It serves as an extension of the Deep Deterministic Policy Gradient (DDPG) [55], specifically designed to address scenarios involving multiple agents. Unlike alternative approaches such as Independent Q-learning (IQL) [56] and QMIX [57], MADDPG employs the Centralized Training Decentralized Execution (CTDE) strategy [54], a widely used paradigm for large-scale multi-agent training. This approach leverages the Advantage Actor-Critic (A2C) framework [58]. Each agent has two independent neural networks called actor and critic to make decisions. Specifically, each agent utilizes the actor to observe local states and take action. During the training process, the critic makes use of global information, such as states and actions of other agents, to evaluate the agent's current action, which can guide model training effectively. MADDPG has been widely used in many fields. Regarding TSC, [59] has proved that MADDPG performs better against other traditional RL-based TSC algorithms. Consequently, we have chosen MADDPG as the primary algorithm for our study.

4 OBSERVATIONAL STUDY

4.1 Experts' Current Practices and Bottlenecks

To gain practical insights into the TSC problem, specifically the objective of minimizing average travel time, we conducted interviews with a panel of six domain experts (E1-E6). These experts encompassed a diverse range of expertise, including two professors with over a decade of experience in TSC and RL (E1-E2), three researchers who had been actively incorporating RL models into their daily work for more than two years (E3-E5), and an expert (E6) specializing in multi-agent control and communication for approximately 10 years. The valuable feedback gathered from these experts played a pivotal role in helping us grasp the existing limitations of RL-based TSC. Throughout the task analysis phase, we sought their guidance on various aspects, including setting up the TSC scenario and identifying key priorities in training the RL model.

Challenges in Real-World Traffic Control and Simulation Tools. During our interviews with experts, two significant challenges were emphasized. The first challenge pertains to the complexity of the traffic environment, which makes it difficult to obtain all the necessary data from real-world road networks without advanced roadside equipment. The second challenge relates to concerns regarding traffic safety and order, which make it impractical to test algorithms in real-world scenarios. Consequently, researchers have turned to simulation platforms such as *CityFlow*² and *SUMO*³ as invaluable tools for conducting experiments and evaluations in a controlled environment.

Current Workflow for Researchers in TSC. The experts we consulted view RL as a promising technique for tackling TSC challenges. However, there remains significant potential for improving the model's exploration, evaluation, and interpretability. The current workflow for TSC researchers can be summarized into three key aspects:

(1) *Setting up a Simulation Environment.* Given the inherent complexity and variability of real-world traffic issues,

1. <https://openai.com/>

2. <https://cityflow-project.github.io/>

3. <https://www.eclipse.org/sumo/>

researchers often choose to abstract the research problems and simplify the road networks. A comprehensive review by Noaen et al. [49] highlighted that around 70% of the examined TSC studies incorporated synthetic road networks.

(2) *Model Training*. During the training of RL models, researchers monitor pre-selected critical metrics such as travel time and reward to assess model performance. Typically, they generate basic line charts or output training results directly to the terminal to visualize how these metrics evolve over episodes. As stated by E1, “we attempted to use TensorBoard⁴ to monitor the training process to capture the connection between action and state, but this approach falls short in providing a comprehensive understanding of the model.” Researchers are generally in search of more robust ways to explore the training process and offer detailed insights for each episode.

(3) *Model Evaluation*. Similarly, researchers have primarily focused on enhancing the algorithm’s performance while giving limited attention to elucidating the decision-making process of the model. They have also made efforts to analyze specific states by inspecting agent actions and metrics, particularly rewards and Q -values, in these states. The approaches mentioned above suffer from two main drawbacks. First, they lack convenience in identifying agents and states of interest. While traffic simulation modules provided by platforms like *CityFlow* and *SUMO* allow researchers to directly observe states and agents, they do not provide a comprehensive overview of agent policies or enable the rapid identification of specific states. Second, these methods fall short in delivering an efficient and systematic means of illustrating the model’s decision-making process and the interactions among agents.

4.2 Experts’ Needs and Expectations

Based on the discussion above, we summarize the following four design requirements, each pertaining to different granularities:

[R1] Episode level: Summarizing the RL Model’s Training Process. Monitoring the training process of the RL model is a critical aspect for researchers in assessing its performance. This monitoring typically involves the observation of key metrics like reward and travel time, which assist in refining the model by detecting changes in these metrics. Thus, it is imperative to offer a comprehensive summary of the model’s training process to facilitate a thorough evaluation of its performance.

[R2] Time-period level: Offering a Policy Overview for Each Agent within a Specified Time Frame.

[R2.1] Describing Temporal Variations in Traffic Conditions in the Road Network: Across each episode, the traffic conditions in the road network exhibit temporal changes. Given the numerous time steps within an episode, it becomes cumbersome to inspect all these steps individually to analyze the evolving traffic situation. Hence, it becomes essential to provide an overview that aids researchers in comprehending the fluctuations in traffic conditions. **[R2.2] Providing an Overview of Each Agent’s Policy:** Across different time periods, the traffic conditions within the road network vary, consequently leading to changes in the policies adopted by the agents. As articulated by E2, “It would be useful

if an agent’s policy can be directly displayed.” Additionally, the policy of each agent effectively reflects the model’s training progress. For these reasons, our approach should summarize and present each agent’s policy within a given time frame.

[R3] Time-step level: Investigating States, Actions, Metrics, and Agent Interactions at Each Time Step.

[R3.1] Providing a Visual Summary for Each Agent: Amidst the training of RL models, a substantial volume of data is generated, encompassing actions, states, and rewards for each agent. This abundance of data can pose challenges in comprehending the intricacies of the model. Researchers find value in obtaining a comprehensive overview of an episode, enabling them to swiftly identify anomalous states. Additionally, having access to general information about each agent proves advantageous. Therefore, there’s a requirement to furnish a visual summary for each agent, facilitating efficient analysis of the training data.

[R3.2] Demonstrating the Influence and Relationships Among Multiple Agents: In MARL, a CTDE framework is frequently employed for model training. Within this framework, a centralized controller guides each agent based on global state, action, and reward information. Decisions made by one agent can influence others, and their actions may have repercussions on fellow agents. Consequently, as suggested by E3 and E4, it becomes crucial to consider the interactions among different agents and analyze their impact on the model’s overall performance. Thus, there’s a necessity to demonstrate the influence and relationships among multiple agents, contributing to a deeper comprehension of the model’s behavior. **[R3.3] Presenting the Consistent Rules Governing Actions and States:** Training a MARL model can be a complex undertaking, sometimes resulting in agents exhibiting unpredictable behavior without clear rationales for their actions. As stated by E5, “Sometimes agents become uncontrollable and I don’t know what happened exactly.” In response to this challenge, the experts recommend that our system should assist in exploring the decision-making process of RL models, particularly by facilitating a better understanding of the connection between an agent’s actions and the underlying state. Hence, it becomes imperative to present the consistent rules governing actions and states, aiding in the comprehension of agent behavior.

[R4] Enabling Simulation Progress Replay. Understanding the decision-making process of an RL model based solely on data can be a formidable challenge. In the realm of TSC, researchers frequently rely on simulation platforms for training RL models, as this approach simplifies the replication of specific scenarios post-training. E2 and E5 emphasized the significance of observing agent behavior in specific states, underscoring the need for a replay module to aid researchers in scrutinizing agent behaviors within particular contexts. Consequently, it becomes imperative to grant users the capability to replay the simulation progress, facilitating a more profound comprehension of the model’s decision-making process.

5 OVERVIEW OF MARLens

In the system pipeline, depicted in Fig. 1, we have two primary components: the *back-end engine* and the *front-end*

4. <https://www.tensorflow.org/tensorboard>

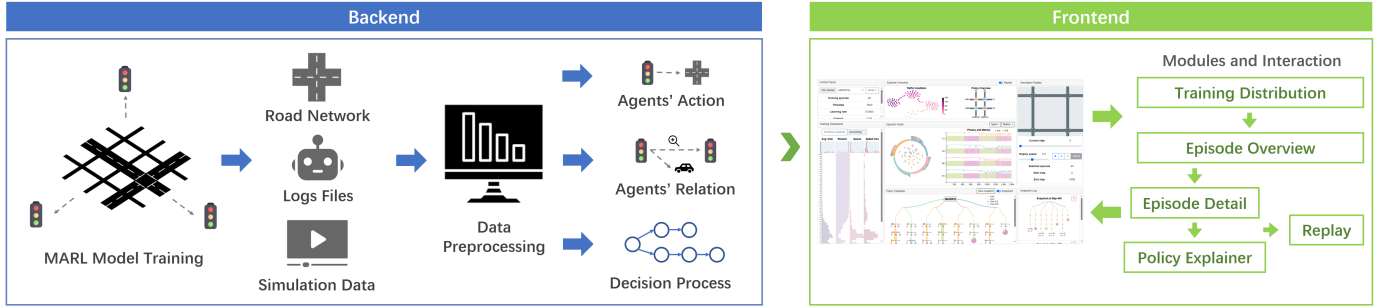


Fig. 1: The system pipeline of *MARLens*. In the back-end engine, we extract critical information about the agents’ behavior, relationships, and decision-making processes from three distinct types of data collected. In the front-end visualization, we offer five coordinated views with rich interactions to facilitate exploration and comprehension of the MARL model.

visualization. Within the back-end engine, our initial steps involve establishing a TSC scenario and training a MARL model. This entails inputting the road network, traffic flow, and optionally, a set of phases, followed by fine-tuning the hyperparameters until the selected model effectively controls the traffic lights and attains a satisfactory performance level. Subsequently, we extract the outputs from the trained networks. This extracted data then undergoes preprocessing to enable analysis of various aspects, including the agents’ actions, their interactions, and the decision-making process. On the front-end visualization side, we have meticulously crafted five coordinated views equipped with rich interactive features, which allow users to comprehend the model’s decision-making process more effectively.

6 BACK-END ENGINE OF *MARLens*

In this section, we will elucidate the data processing procedures, encompassing scene initialization, feature extraction, and the subsequent outputs derived from these processes.

6.1 Scene Initialization

Using the *Libsignal*⁵ [60] platform as a basis, we establish a scenario comprising a 2×2 network, where each intersection serves as an agent denoted by A0, A1, B0, and B1 (Fig. 2C₂). During our expert interviews, researchers typically overlook specific details like turning vehicles when studying general TSC problems of road networks. Consequently, our traffic flows are restricted to straight movement only. Moreover, we assign unique identification numbers to each node. An edge is represented by two nodes. For instance, “A0A1” refers to the edge connecting the intersection from node A0 to A1.

After approximately 50 training episodes using MADDPG with specific training hyperparameters, such as a learning rate of 0.0005 and a target network update frequency of once every 10 episodes, the algorithm demonstrates a tendency toward convergence. Each training episode is segmented into 1600 time steps, with each step corresponding to one second of simulation time. Additionally, we divide the traffic flow into four distinct stages to observe agent behavior under varying traffic conditions. Each stage has a duration of 400 time steps. The four stages, outlined in Table 1, are: (1) **W-E Only**, (2) **N-S Only**, (3) **N-S**

TABLE 1: The traffic flow settings of our cases.

Stage	Direction	Traffic Flow (veh/h)
1	W-E	1800
	N-S	0
2	W-E	0
	N-S	1800
3	W-E	1800
	N-S	600
4	W-E	600
	N-S	1800

lower than W-E, and (4) **W-E lower than N-S**. The traffic flow in the first two stages moves in only one direction, while the last two stages involve traffic flow in all directions, with a larger flow in one specific direction.

Upon completion of each training episode, the trained model will undergo a testing phase to evaluate whether it has entered a state of over-fitting. The testing scenario will be identical to the training scenario.

6.2 Data Description

To provide explanations of the MADDPG algorithm, we have extracted data from the training and testing models as well as logger files. The extracted data can be broadly categorized into three groups: **1) Model Data**. This group includes the outputs of both the training and testing models, which contain various neural network metrics. As described in Section 3.2, the MADDPG algorithm creates a critic network and an actor network for each agent. We have extracted the input and output data from these networks. Specifically, for the critic network, the network requires global observation O and action probability P from each agent to score the corresponding agent’s behavior with Black box neural networks. As for the actor network, the network predicts the probability of each action based on the local observation of the agent. **2) Logger Files**. During the model training or testing phase, a large number of logger files are generated. These files contain various metrics such as mean reward, queue length, delay, and travel time, for each episode. In addition, for each episode, we also obtain all the observations such as queue length of each lane, current/last action, and current/last phase for each of the 10 time steps. **3) Simulation Data**. *LibSignal* can generate traffic simulation data automatically. The phases of intersections and vehicles’ location will be recorded every time step.

5. <https://dar1-libsignal.github.io/>

Simulation data is useful to replay the status of the road network for detailed analysis.

6.3 Feature Extraction

To identify the key features, we perform additional data processing from two different perspectives.

Decision-making Process. In the A2C framework, each agent utilizes neural networks for decision-making. Drawing inspiration from Mishra et al. [26], we independently train decision trees for both the critic and actor networks to gain a deeper understanding of how neural networks make decisions. Specifically, we employ all global information and action probabilities from all agents to train a regression tree. This regression tree establishes a connection between global information and the agent’s evaluation of the current state, providing insights into which features contribute to a specific action. Importantly, this method is model-agnostic, utilizing only state and action information as inputs without introducing new structures inside the model.

Feature Importance. Another model-agnostic approach we employ to glean insights into the model training process is SHAP [42], commonly used for measuring feature importance. Previous studies [41] and [47] have applied SHAP to RL-based TSC. Building on their work, we also use SHAP to analyze the neural networks of each agent and illustrate feature importance.

7 FRONT-END VISUALIZATION OF *MARLens*

In this section, we present the visual design of each view within *MARLens*, accompanied by an exploration of diverse design alternatives that underwent evaluation. Our system comprises six primary components: the Control Panel, Training Distribution, Episode Overview, Episode Detail, Policy Explainer & Snapshot Log, and Simulation Replay.

7.1 Control Panel

The Control Panel (Fig. 2A) is designed to serve as a hub for presenting training information related to various models. Each model’s profile encapsulates details regarding both the training and testing processes. This includes essential information such as the training episode, time step, and crucial hyperparameters like the learning rate. Moreover, the Control Panel affords users the flexibility to seamlessly switch between the display of training and testing data, with the system initially set to showcase the training data as the default view. When a user selects a specific RL model of interest, the relevant information dynamically updates below, while other associated views also receive corresponding updates.

7.2 Training Distribution

To offer a concise overview of the model training process ([R1]), commonly used metrics are employed to evaluate the traffic condition of the road network. These metrics encompass parameters like *speed loss*, *queue length*, *reward*, and *avg. time*, all of which were exclusively derived from the training data. To analyze the distribution of these metrics effectively, we devise the Training Distribution (Fig. 2B),

implementing the lineup design [61]. This design choice enables users to rapidly visualize the distribution of the four metrics and swiftly locate their desired episode within the data. Given the substantial differences in the value ranges of these metrics, we conduct data normalization to ensure meaningful comparisons. Furthermore, users are allowed to arrange and sort specific metrics as per their requirements.

7.3 Episode Overview

To offer a comprehensive policy overview for each agent within a specific time frame ([R2]), the Episode Overview has been intricately designed to facilitate an in-depth exploration spanning various time stages and policies. This component comprises two main sections: *Traffic Condition* and *Policy Overview*.

Traffic Condition. To observe how traffic conditions evolve throughout an episode ([R2.1]), we count the number of vehicles present on each road and employ the *t-SNE* algorithm [62] to project traffic conditions onto a 2D plane. Each time step’s traffic condition is represented as a point in Fig. 2C₁. The color of each point corresponds to its respective time step, with lighter colors denoting earlier times and darker colors representing later times. The use of *t-SNE* ensures that similar traffic conditions are visually clustered together.

Policy Overview. For a comprehensive understanding of each agent’s policy ([R2.2]), we construct a representation of the road network, as depicted in Fig. 2C₂. In this representation, the opacity of the green color is indicative of the number of vehicles on the road. Each intersection in the diagram represents an agent. Within each intersection, we employ four sectors to illustrate the probability of allowing vehicles to pass, corresponding to the four cardinal directions. The size of each sector correlates with the probability it represents, with larger sectors indicating higher probabilities.

Moreover, users can select a group of points of interest with a lasso in Fig. 2C₁. When points are selected, the corresponding sectors in the four directions, as well as the road network, are updated to provide further insights.

7.4 Episode Detail

In order to provide a visual summary for each agent ([R2]) and demonstrate the influence and relationship among multiple agents ([R3]), the Episode Detail is designed to provide a thorough exploration of each episode, which mainly contains two parts, namely *state projection and feature importance* and *traffic signal phase and metrics*.

State Projection and Feature Importance. On the left part, a circular segment design is adopted to present information about the four agents in our scene (Fig. 2D₁). A ring is partitioned into four segments (arcs) and their colors correspond to different agents. Inside the ring, we utilize the *t-SNE* algorithm to project the state information, which includes the current decision action, rewards, current time step, and the traffic condition of the intersection on the map, into the 2D plane (Fig. 2D₂). This approach ensures that similar states are situated closer to each other. Before zooming in, the color of each state point corresponds to the color of the agent. The opacity of the point encodes

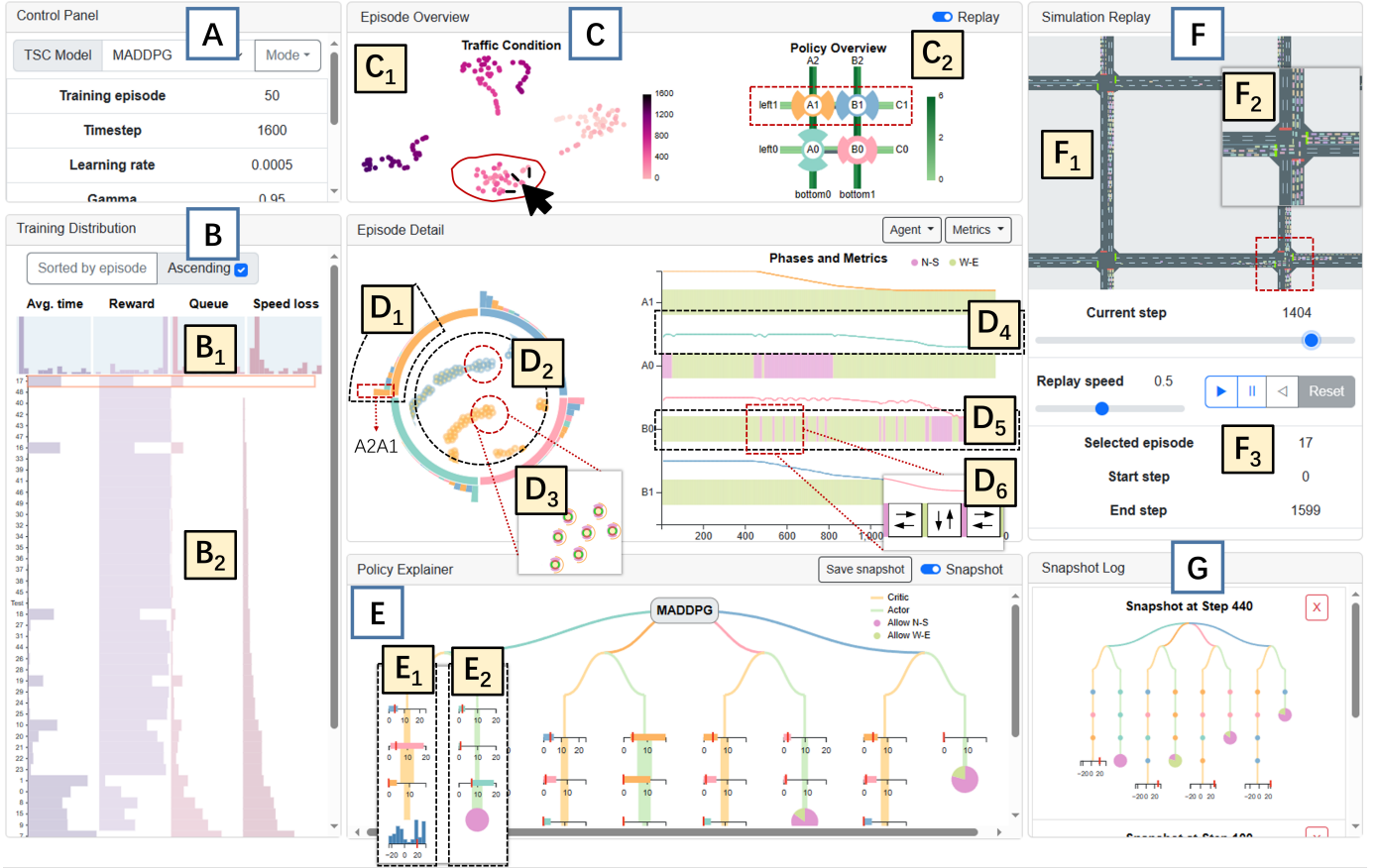


Fig. 2: Our visualization system, *MARLens*, provides an in-depth analysis of MARL models in TSC scenarios. The Control Panel (A) presents parameters in model training and model testing. The Training Distribution (B) provides the distribution of the metrics and ranks the episode based on the metrics. The Episode Overview (C) presents a summary of traffic conditions and each agent’s policy at a certain episode. The Episode Detail (D) provides a visual summary for each agent in an episode, including information of the state, action, and selected metrics, and demonstrates relationships among multiple agents. The Policy Explainer (E) provides explanations between local state and action, global information and critic value. The Simulation Replay (F) supports the replay of an arbitrary episode or time step in the simulation situation. The Snapshot Log (G) saves the snapshots of the Policy Explainer.

the reward. After zooming in, to provide more state information, we design a glyph, as shown in Fig. 3c. The center circle encodes reward, with darker shades representing higher rewards. The inside ring encodes action, with red representing stop and green representing pass. The middle ring encodes vehicles in four directions, with the wider arc representing more vehicles. The outer ring encodes the time step, with the long arc representing later time step. Users can click on these state points (glyphs) in the 2D plane, and other views (e.g., the Policy Explainer) will be updated correspondingly. Moreover, we use a chord diagram to display an overview of the influence among agents (Fig. 3b). A chord from agent A to agent B indicates some features of A affect B’s decision-making. The width of the chord encodes the number of features. To avoid visual clutters, when users hover over an agent, the chord diagram will only display influences related to this agent. Gradient colors are used to strengthen agent information. Outside the arcs, bar charts are used to depict the interplay between agents and Shapley values of the features. To enable better comparisons, these bar charts are encoded with the same color as the corresponding agents and arranged in order of Shapley values (clockwise). As shown in Fig. 3a, when users

hover over a specific feature, the corresponding feature name will be displayed, facilitating a more comprehensive evaluation of each feature’s importance in decision-making.

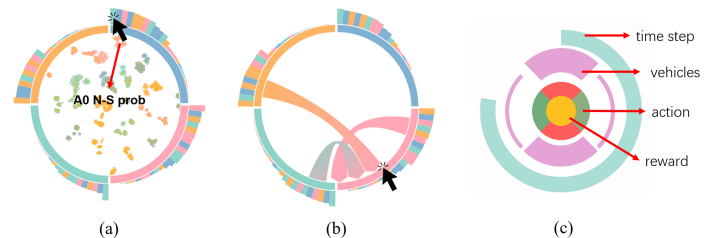


Fig. 3: Glyph design and interaction in the Episode Detail. (a) When hovering the mouse over the bar charts (feature importance), the corresponding feature’s name is displayed in the center. (b) Utilization of a chord diagram to illustrate the relationships among agents. (c) The glyph design showcasing state information, employing distinct rings to represent various aspects of the episode, such as time step, traffic flow, action, and reward.

Traffic Signal Phases and Metrics. On the right side, to better visualize and compare metrics and actions among

different agents, we devise four sets of lines and bands corresponding to each of the four agents. The line chart effectively illustrates the fluctuations in selected metrics (Fig. 2D₄). Bands along the time axis in Fig. 2D₅ represent traffic signal phases, with different colors signifying phases selected by individual agents. Users can zoom in to closely examine metrics and actions at each time step. To facilitate the quick identification of traffic signal phases, when users zoom in to a certain degree, icons appear to directly display traffic signal phases (Fig. 2D₆). Users have the flexibility to select a time step within the bands to update other views.

Design Alternatives. Before settling on the current designs, we explored various alternatives. Initially, Fig. 4a was employed to showcase agents’ traffic signal phases, which is a common practice in TSC. However, it could only display one traffic signal phase per row, leading to significant space consumption when dealing with numerous agents and traffic signal phases. As illustrated in Fig. 4b, an attempt was made to conserve space and facilitate the comparison of actions among different agents by using one row to represent the traffic signal phase information of an agent, with different colors encoding various traffic signal phases. Despite these efforts, this design proved less intuitive as the number of traffic signal phases increased. Subsequently, the final version (Fig. 4c) was developed to address these challenges. In this version, an icon is placed in front of each bar to enable users to quickly identify different traffic signal phases. The use of line charts effectively illustrates how metrics change over time.

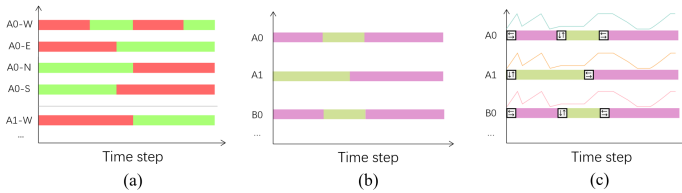


Fig. 4: Various design alternatives were evaluated for the components within the Episode Detail. (a) Represents a commonly employed design in TSC for displaying traffic signal phases. (b) Introduces an alternative design aimed at conserving space and facilitating the comparison of agents’ actions. (c) Depicts the current design, employing icons to directly illustrate different traffic signal phases. Line charts are utilized to concurrently compare metrics for different agents.

7.5 Policy Explainer & Snapshot Log

In order to facilitate a better understanding of the decision-making process of MARL models, we have adopted a tree-based design to present the policies governing the relationship between actions and states ([R3]). As illustrated in Fig. 2E, the root node of the tree represents a selected “MADDPG” model, and four subtrees are used to encode rules for four agents, each designated with its respective agent color. For each agent, a subtree is divided into two branches: 1) The left branch (Fig. 2E₁) elucidates how a critic arrives at a decision. This involves traversing several feature judgments, represented by feature ranges in the corresponding agent’s color, to reach a final critic value encoded with a red bar and distribution. The higher the critic value, the

higher the critic’s evaluation of the agent’s current situation. The current feature value is depicted as a red bar within the feature range. 2) The right branch (Fig. 2E₂) illustrates the action-performing process of the actor. Similar to the critic branch, it involves navigating through feature judgments before executing an action. The final action distribution is visualized as a pie chart. The width of a path within the tree encodes the value range of the associated features. Additional information is revealed when users hover over feature judgments and the corresponding roads will be highlighted in the road map (Fig. 2C₂). To support the comparison of different rules, users can save a snapshot to the Snapshot Log by clicking the “Save snapshot” button. Users can easily reload a snapshot by clicking it. In addition, users can hide the Snapshot Log by clicking the switch button on the top-right of the Policy Explainer.

Design Alternatives: Prior to adopting the current tree-based design, several alternative designs were considered. For instance, as shown in Fig. 5a, one design approach proposed by [26] involved directly displaying feature names and feature value ranges in each branch, accompanied by text explanations. However, this design was found to lack intuitiveness. Another design, depicted in Fig. 5b, utilized distribution representations but did not adequately distinguish between actor and critic branches, used uniform colors for each agent’s branch, and lacked variation in branch thickness. Consequently, the current tree-based design was implemented, introducing distinct colors for agent branches and utilizing branch thickness to visually encode the number of rules, as illustrated in Fig. 5c.

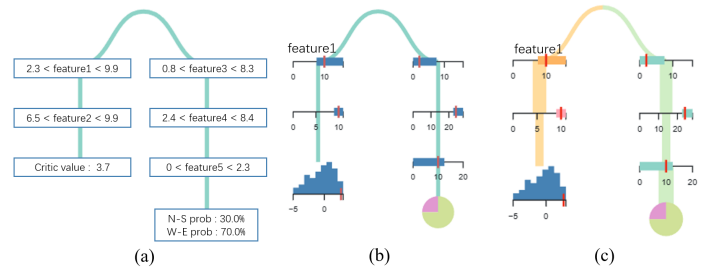


Fig. 5: Design alternatives were considered for the Policy Explainer component. (a) One design approach involved presenting feature names and their value ranges directly within each branch, supplemented by accompanying text. (b) Another design utilized distribution representations but did not differentiate between branches for different agents or vary branch thickness to indicate the number of rules. (c) The current design employs distribution representations with distinct colors for each agent’s branches and varying branch thickness to visually represent the number of rules.

7.6 Simulation Replay

To better understand the model’s decision-making process ([R4]), we have incorporated the Simulation Replay module, making use of the *LibSignal* platform. This module comprises an upper section that displays a road network (Fig. 2F₁) utilized for simulation purposes. When users select an episode in the Training Distribution, the Simulation Replay seamlessly transitions to the corresponding episode.

The lower portion of this module (Fig. 2F₃) allows users to control the simulation. For example, users can pause the replay process at any point and adjust the replay speed within a range from 0.1 to 1, enabling a closer examination of specific moments. Additionally, the Simulation Replay supports the option to restart the current process, restoring the original settings for replay purposes. Furthermore, for quick access to a particular time step, users can select a specific state within the Episode Detail, which will trigger the display of the simulation process, commencing from the specified time step. Similarly, users can close the Simulation Replay by clicking the switch button on the top-right of the Episode Overview.

8 EVALUATION

We perform three case studies and conduct interviews with E1-E6, as introduced in Section 4.1. Section 8.1 details the cases conducted by E2, E1, and E6 respectively, due to the representativeness of these cases. Section 8.2 presents the feedback from all experts. Subsequently, we recruit an additional 12 participants and conduct a user study to assess the performance of *MARLens* in system workflow, visual design and interaction, as well as system usability, as discussed in Section 8.3.

8.1 Case Study

8.1.1 Case I: Overview of Model Training

E2 focuses on crafting RL models and often needs to summarize the performance of model training. His actions are illustrated in Fig. 6I.

To gain a macroscopic understanding of the model training situation, he typically examined the Training Distribution (Fig. 6A₁). Sequentially analyzing the episodes, he observed that the model training was effective, experiencing some fluctuations in the early stages but becoming more stable later on. Conversely, episodes with larger rewards were predominant in the later stages (Fig. 6A₄), providing further evidence of the model's effectiveness.

Then, E2 navigated to the Episode Overview to scrutinize the agents' policies. Observing the road network's traffic conditions in Fig. 6B₁, he identified four distinct clusters corresponding to the four types of traffic flow settings. The red rectangles in Fig. 6B₁ denoted transitions between different traffic flows. E2 chose the first stage of traffic flow (W-E direction only) in the initial episode. In Fig. 6B₂, each sector's size was nearly identical, indicating that all agents assigned almost the same probability to the green light in each direction. E2 remarked, "*Agents appear to take actions randomly, and their strategy for allowing vehicles to pass is confusing and inefficient.*" In contrast, the last episode in Fig. 6B₃ showed larger sectors in the W-E direction than in other directions. All agents permitted only west and east bound vehicles to pass, aligning with the traffic flow's direction.

Subsequently, E2 used the Episode Detail to assess actions and metrics' temporal changes. He chose agent A1 as an example, focusing on the critic value to observe the model's grasp of global information. For actions, he observed frequent color changes in the bands,

signifying continuous policy exploration in the first episode (Fig. 6C₁). Conversely, the strategy stabilized in the last episode (Fig. 6C₃) with A1's actions distinctly segmented into four stages corresponding to the four-stage traffic flow. The line chart in Fig. 6C₁ and C₂ revealed that the critic value in the first episode fluctuates within a narrow range, indicating the model's inability to evaluate the entire road network's traffic condition. Conversely, in the last episode, the critic value accurately reflected the road network's traffic conditions. A higher critic value denoted better traffic conditions. E2 leveraged the Simulation Replay to directly inspect A1's traffic condition. For instance, in the last traffic flow stage where N-S direction is the primary traffic flow direction, the first episode, due to trial-and-error, sometimes leads to severe congestion (Fig. 6C₂). Conversely, in the last episode, the intersection's traffic condition was more orderly. A1 kept the N-S direction green until queues in the W-E direction accumulated, prioritizing the main traffic flow (Fig. 6C₄).

Continuing with the reward and state analysis, E2 explored A1's states during the last traffic flow stage (Fig. 6D₁). Given the model's proficiency in the last episode, his focus shifted accordingly. Notably, he observed two clusters in Fig. 6D₂, indicating high similarities. The left cluster was smaller than the right, prompting his to zoom in for a more detailed examination. It became evident that the left cluster represented states when A1 activated the green light in the W-E direction. As shown in Fig. 6D₄, the larger area of purple sectors in the W-E direction suggested more vehicles waiting in that intersection direction. Conversely, the right cluster denoted states when A1 activated the green light in the N-S direction. The upper part of this cluster indicated lower rewards, signifying congestion in the main traffic flow direction (N-S direction) (Fig. 6D₃).

In summary, E2 concluded that the model underwent effective training with the increasing number of training rounds. All metrics exhibited substantial improvement, indicating that agents had learned to adapt and employ different strategies in response to varied traffic flow settings.

8.1.2 Case II: Decision Process of MADDPG

E1 aimed to delve into the decision-making process of the model. Her specific operations are illustrated in Fig. 6II.

Considering the decision process is stable in the test episode, E1 conducted a comprehensive analysis. Initially, she observed the metrics of the test episode in the Training Distribution (Fig. 6A₄). Despite the test episode not achieving the optimal performance, it remained within the normal fluctuation range.

Then, within the Episode Overview, she selected two time periods, as depicted in Fig. 6E₁ and E₂. These illustrations revealed a correlation between higher traffic flow (darker green on the roads) and a higher probability (larger area of the sectors) of activating green lights in corresponding directions. Fig. 6E₃ provided detailed insights into the agents' actions. For example, in the third stage in Fig. 6E₃, where the main traffic flow was in the W-E direction (Table 1), the area of green bands (indicating turning on the green light in the W-E direction) significantly exceeded that of purple bands. Regarding the inter-agent relationship, E1 observed that higher bars consisted of colors from other

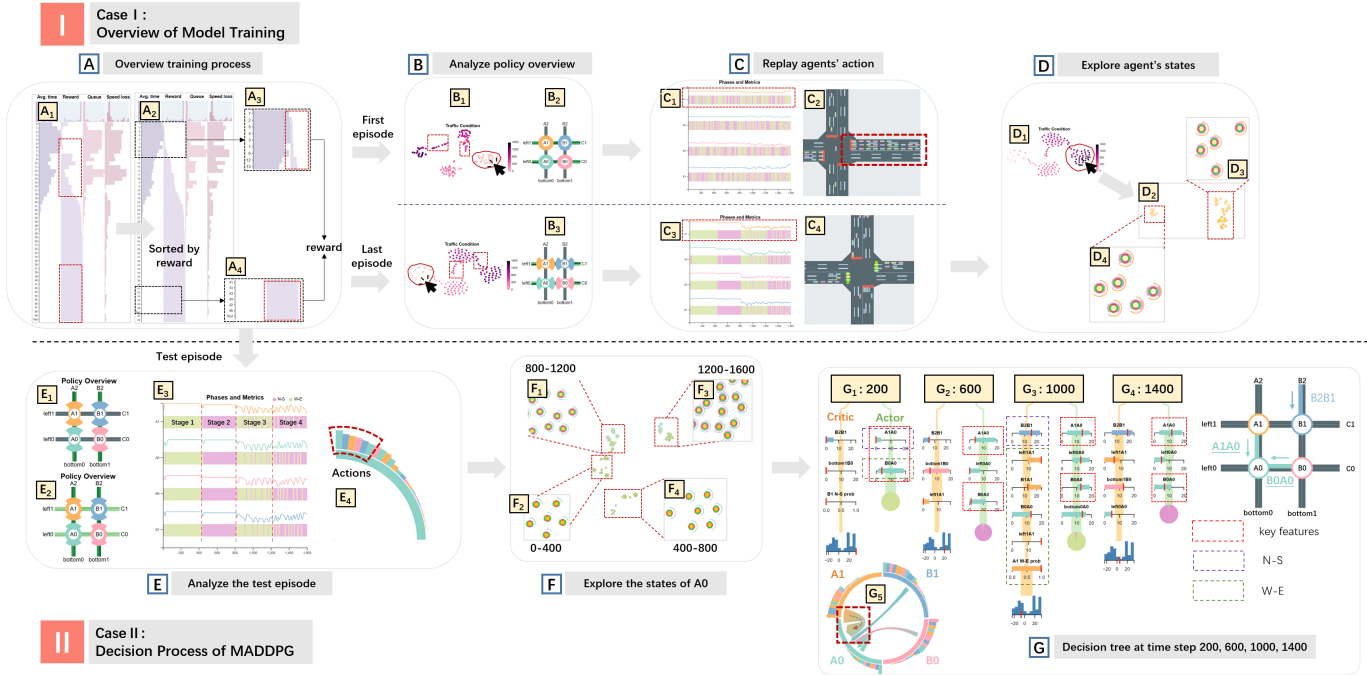


Fig. 6: Experts' operations during the case study I and II. (A) Use the Training Distribution to compare different episodes. (B) Analyze and compare the overview of agents' policy in different episodes. (C) Replay agents' actions with more details. (D) Select a time range and explore the states of an agent. (E) Analyze how agents react to different traffic flows in the test episode. (F) Explore the states of agent A0. (G) Reveal the decision-making process of agent A0.

agents, indicating that an agent's action was significantly influenced by others (Fig. 6E₄). She remarked, "Agents tend to observe each other's actions, reflecting the process of cooperation among agents."

Moving on to the projection of states, E1 examined the projections of each agent separately and found that the results could be roughly divided into four clusters based on locations and rewards. Taking agent A0 as an example, four clusters were highlighted with red rectangles (Fig. 6F). To compare differences within these clusters, she zoomed in on each cluster and observed the glyph in Fig. 6F₁ – F₄. Comparing the time step and traffic flow of the states in each cluster, she discovered that states within the same cluster were highly similar. Considering the traffic flow in the middle of a stage is stable without interference, she chose the middle time steps of the four traffic flow stages (200, 600, 1000, and 1400) for further exploration.

E1 conducted a detailed analysis of A0's decision process at four representative time steps, utilizing the Snapshot Log to save snapshots for easy comparison of agents' decision processes under different states. She identified *A1A0* and *B0A0* as key features influencing the agents' decisions at each traffic flow stage (Fig. 6G). Upon comparing the values of these features, she observed that the agent's policy was influenced by the number of vehicles in different directions. For instance, at time step 200 (Fig. 6G₁), the actor branch considered the number of vehicles on *A1A0* and *B0A0*, with the value of *B0A0* significantly higher than *A1A0*. Consequently, the agent chose to allow vehicles in the W-E direction to pass through the intersection. When the number of vehicles in two directions was similar (time step 1000 in Fig. 6G₃), E1 delved deeper into how the critic guided the model's decision process.

E1 noted the chord diagram in Fig. 6G₅, where the chord from agent A1 to agent A0 was larger than other chords. She remarked, "It seems that A1 has a greater impact on A0 at this time." Examining the critic tree in Fig. 6G₃, she found an evaluation of the agent's action based on features mainly related to agent A1 in the W-E direction (highlighted with a green dashed rectangle), such as *left1A1* and *A1 W-E prob*. At this time step, A1 turned on the green light in the W-E direction (*A1 W-E prob*) due to a higher number of vehicles in the W-E direction (*left1A1*, *B1A1*). For agent A0, maintaining the N-S direction green might interrupt A1 and lead to congestion in A1's N-S direction. Therefore, agent A0 opted to turn on the green light in the W-E direction.

In conclusion, E1 concluded that the agent prioritizes the direction with heavier traffic flow. In cases where traffic situations are similar, the critic serves as a valuable aid in decision-making based on other features. E1 remarked, "This looks like priority control in TSC, where traffic lights stay green for the main road until there are too many vehicles waiting in the side road."

8.1.3 Case III: Anomaly Detection of the MARL model

Identification of Abnormal Episode. When E6 explored the training process in the Training Distribution, he found an unusual episode 17. As shown in Fig. 2B, the speed loss of episode 17 was very low, but the reward has not reached the optimal level. He sorted episodes immediately according to speed loss in the training distribution (Fig. 2B₁). He noticed that the speed loss of episode 17 was the lowest in the training process. To understand what caused this phenomenon, he selected episode 17 and explored it further in the Episode Overview (Fig. 2C) and the Episode Detail (Fig. 2D).

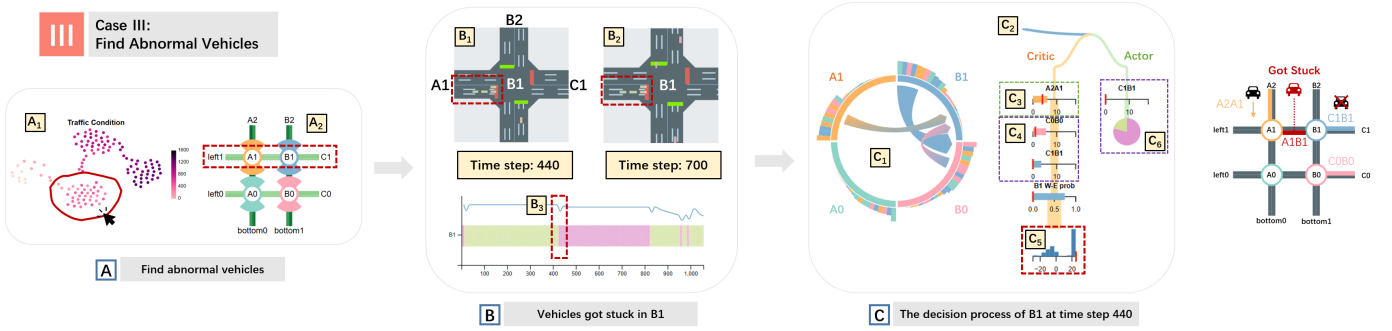


Fig. 7: Experts' operations to explore abnormal vehicles. (A) Identify abnormal vehicles during exploration. (B) Vehicles got stuck in intersection B1 and the expert selected the time step when the agent changed the traffic signal. (C) The decision-making process of agent B1.

At first, for some agents, the size of sectors in the W-E direction is much larger than sectors in the N-S direction (Fig. 2C₂), which means those agents keep the green light in the W-E direction on, even the main traffic flows are in the N-S direction. In the Episode Detail, he focused on the metrics and traffic signal phases in Fig. 2D. As shown in the right part of Fig. 2D, both agents A1 and B1 had a long green band. They all adopted the strategy of only allowing vehicles in W-E directions throughout the episode. Except for A0, the reward and critic value of the agents went down rapidly over time. Then, he checked the states of A1 and B1 in Fig. 2D₂, the projection was distributed in a chain. There were many light circles in Fig. 2D₂, which indicated low reward. Since the state's distribution of A1 and B1 were similar, E6 took A1 as an example for analysis. After zooming in, those light circles turned into glyphs in Fig. 2D₃. The large purple sectors in the N-S direction demonstrated there were more vehicles in the N-S direction. Considering the Shapley values in Fig. 2D₁, he realized that agents focused on its entry in the N-S direction (A2A1). Combined with the Simulation Replay (Fig. 2F₂), E6 noticed that the N-S direction was extremely congested because the W-E direction is always allowed to pass. Due to the congestion in the N-S direction, many vehicles were even blocked inside the intersection. This also caused traffic congestion in the W-E direction framed in Fig. 2F₂. He thought that the long-term congestion in the N-S direction greatly reduces the reward, making this N-S feature the most important one. To answer why the 17 episode has the lowest speed loss with poor performance, he concluded, "In this episode, most of the traffic flow in the W-E direction maintained smooth. The speed loss of them was 0 at most time steps. This greatly reduced the speed loss of the entire road network." He added, "Compared with earlier episodes, the metrics of episode 17 are better but its strategy is very extreme. In the past, we could only write programs to judge by ourselves, but now we can find out more quickly with the help of the system."

Find Abnormal Vehicles. In the test episode, E6 wanted to explore how agents' strategy changes with the traffic condition according to the Episode Overview. After he selected a time period in Fig. 7A₁, he found there were a few vehicles in the W-E direction of the road network (highlighted by the red rectangle in Fig. 7A₂). "At first, I thought these vehicles were remnants of the previous stages and would leave the

road network soon, since the current stages only had traffic flows in the N-S direction." However, after he checked each intersection one by one. He found some vehicles got stuck in agent B1's intersection. Those abnormal vehicles are demonstrated in Fig. 7B₁ and B₂. Next, he used the Episode Detail to select the time step when B1 changed its traffic signal phase (Fig. 7B₃). In the chord diagram in Fig. 7C₁, a chord from A1 to B1 and a chord from B0 to B1 show that B1's surrounding intersections A1 and B0 had an influence on its decision. Then he moved to the decision process with more detail in Fig. 7C₂. For the critic of B1 in Fig. 7C₃, he observed there were many vehicles in the N-S directions (A2A1). In the W-E direction in Fig. 7C₄, he noticed two roads (C0B0 and C1B1). Especially, C1B1 is one of the entries of B1's intersection and there was no vehicle on C1B1 at this time step. Although the critic of B1 got global information including one of its entries (C1B1) as well as its surrounding intersections, it still ignored there were a few vehicles got stuck in A1B1 (Fig. 7B₁) and thought highly of B1's action in Fig. 7C₅. Similarly, for the actor of B1, it only focused on the vehicles on C1B1 to make the decision Fig. 7C₆. "That's an interesting result, which shows why the model made a mistake at this time. One agent tends to pay more attention to the main traffic flows. As a result, vehicles in other traffic flows may be neglected. This may lead to unreasonable waiting time in the real world." To avoid this problem, E6 recommended adding more rules, such as considering the waiting time of the vehicle in the reward function or setting a time limit for each traffic signal phase according to traffic flow.

To sum up, case III proves: (1) the limitations of judging the model training effects from certain metrics and the effectiveness of *MARLens* for identifying the abnormal situation. (2) *MARLens* has the potential to enable users to quickly find abnormal situations and explain why the model made the decision in a given state, which can provide suggestions for model improvement in the future.

8.2 Expert Interview

We organized individual interviews with experts E1-E6 (their background has been introduced at the beginning of Section 4.1). First, we briefly introduced our system and provided a simple tutorial to demonstrate the visual design and interactions of *MARLens*. Next, experts could explore the system for about an hour. Then we carried out an

individual interview for each expert. Each interview lasted about 30 minutes and we collected their feedback as follows:

System Designs. Generally, all experts agreed that our system is useful to explore and strengthen the interpretability of RL-based TSC problems. After the simple tutorial, they could easily understand the purpose and meaning of our visual designs. They thought our system provides enough information to track the RL model’s training process and agents’ behaviors. *“I can easily observe agents’ reactions to different traffic conditions and directly compare how agents’ strategies change over episodes.”*, said E3. Some experts mentioned that the existing tools, such as *TensorBoard*, cannot provide friendly support to combine traffic simulation platforms, which makes it hard to analyze RL models’ decision-making process under a certain state. Besides, E1 said that *“sometimes we need to design a segmented reward function according to traffic condition. The Policy Explainer Module can provide suggestions on setting the threshold of segmentation.”*

Usability and Suggestions. Generally, experts approved that the modules of *MARLens* are user-friendly. It may be not easy to understand some visual designs at first glance, but after the simple tutorial, they could easily understand the purpose and meaning of our visual designs. Moreover, they agreed that *MARLens* provides a multi-level analysis for a MARL-based TSC model. *“It’s insufficient to evaluate the model only based on metrics. MARLens is useful to understand the model more comprehensively.”*, said E5. Many experts approved that the findings at different levels are interesting, especially in exploring abnormal situations. For the Episode Detail, E6 can easily get the main idea of the visual design of the Episode Detail, but the middle part can be more intuitive (Fig. 2D₂). *“The state glyph can be turned into a real intersection.”* In addition, he said that we can add more connections between the Episode Detail and the Simulation Replay, such as showing reward in traffic simulation directly. E1 suggested that the system can take more factors into account. *“The mixed traffic flow of different vehicles and traffic rules can affect how traffic conditions change. Further, she valued the potential of the system to guide RL model design in the future. She said that “The relation between agent and environment can guide us to design action and reward function. How the reward functions influence an agent’s behaviors is worthy of further exploration.”*

8.3 User Study

A user study is conducted to further evaluate *MARLens* in terms of *system workflow*, *visual design*, and *system usability*.

Participants. We invited 10 participants (3 females, 7 males, $age_{mean}=24.1$, $age_{std}=1.59$) majoring in traffic engineering from a local university. All participants possessed basic knowledge in TSC and RL. Specifically, 9 of them were master’s students, while 1 participant was pursuing a doctoral degree. Their familiarity with RL and traffic simulation software, applied in their respective research areas such as TSC and autonomous driving, made them ideal candidates as target users for our system. The selection of these participants was strategic, as they could offer valuable insights into the practice of RL models. The user study was conducted in a face-to-face setting. We presented and demonstrated our system to the participants, who then

actively engaged with the system to complete various tasks. Following the interactions, we utilized questionnaires to systematically gather feedback and suggestions from the participants, aiming to capture their perspectives on the usability and functionality of our system.

Tasks. In this user study, participants were tasked with completing three assignments. The outlined tasks are as follows: (1)**Task 1: Describe Model Training.** In this task, participants were instructed to articulate the trend of the training process based on metrics and evaluate the performance of the model. (2)**Task 2: Episode Exploration.** Participants were asked to select the last episode and describe the characteristics (action sequences, reactions to different traffic flows, states, et al.) and relationships among different agents. They were also encouraged to compare different episodes and synthesize observations on how agents evolve throughout episodes. (3)**Task 3: Understand Decision Process.** In this task, participants were tasked with choosing a state and elucidating the decision-making process of the RL model according to features and the value of the features.

Procedure. Initially, we provided participants with an overview of the traffic simulation settings, essential definitions of the example RL model, visual design components, workflow, and the system’s overarching framework. Following this introduction, we conducted a comprehensive 20-minute demonstration of the system, delving into detailed explanations of each view’s functions and visual design elements. Subsequently, participants had the opportunity to operate the system themselves and seek assistance from us as needed. Once participants felt comfortable, they were tasked with completing the aforementioned assignments using our system. Upon the completion of all tasks, participants were required to fill out a questionnaire featuring Likert scale questions. The Likert scale employed for the questionnaire ranges from 1 to 7, where 1 denotes “strongly disagree”, and 7 indicates “strongly agree”.

Results. The results of the questionnaires are depicted in Fig. 8, indicating that participants held a favorable opinion of *MARLens*. Notably, the average score for each question surpassed 5.7. Regarding the system workflow, participants expressed satisfaction with their ability to comprehend the RL model’s training process swiftly and efficiently select target episodes. The system was lauded for facilitating exploration of different states and comparisons among agents. In terms of visual design, the system was praised for furnishing ample information to support multi-level analysis, enabling participants to gain a comprehensive understanding of the RL model and identify patterns during training. Specific commendation was directed towards the design in Fig. 4c. One participant remarked, *“Once I used the traditional diagram to compare traffic signal phases of different intersections in a road network. The diagram became too long at last, which made it hard to read and could easily lead to misunderstandings. While participants generally found the system’s modules well-organized for usability, there were suggestions for improvement. Some participants recommended providing an overview of each state cluster. In terms of system usability, participants broadly agreed that the different modules were well-organized, enhancing their understanding of the MARL model in the TSC scenario. Additionally, some participants expressed a desire for more typical TSC scenarios*

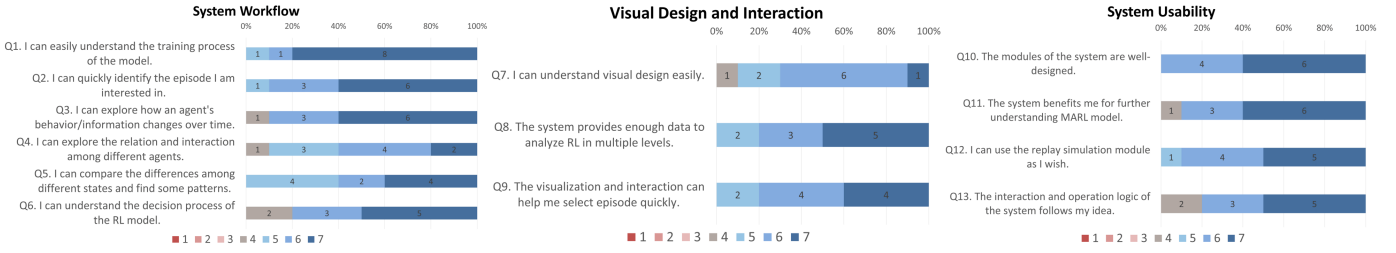


Fig. 8: The result of questionnaires. 1-7 represents “strongly disagree” to “strongly agree” for each statement.

as examples and requested support for additional user-customized options.

9 DISCUSSION AND LIMITATION

Improvements on the Current Workflow. Based on the case study and interviews with domain experts, *MARLens* enhances researchers’ workflow in three aspects. First, it offers a multi-level analysis of the model, which stands in contrast to the prevalent reliance on metrics and line charts for evaluating model performance. This expanded perspective allows researchers to comprehensively explore and analyze the model across varying time granularities, thereby enhancing their understanding. Second, *MARLens* significantly reduces the time required for researchers to unearth pertinent information. Handling high-dimensional data, such as states and traffic conditions, often presents challenges for researchers seeking to leverage them effectively for model exploration and evaluation. Researchers now can easily compare time series data generated by each agent within both the Episode Overview and Episode Detail. Last, the interpretability of the MARL model in the domain of TSC has received less attention in comparison to model performance. *MARLens* addresses this by striving to reveal the decision process of the MARL model, offering an intuitive means for researchers to comprehend the model and provide valuable guidance for future enhancements.

Effective Communication with Domain Experts. Despite the extensive research conducted by experts E1-E6 in RL-based TSC, their understanding of interpretability and visualization has been limited, leading to communication challenges. To address this, we have implemented strategies to enhance communication efficiency. First, in terms of visualization, experts typically utilized simplistic diagrams like line charts and traffic light phase diagrams during research, as depicted in Fig. 4a. However, obtaining direct visual design requirements proved challenging. To overcome this, we introduced visualization cases from previous studies upfront, aiding experts in grasping the utility of visualization. Second, there existed a gap in the comprehension of interpretability. Initially, experts perceived interpretability as the integration of RL models with traditional theories in the TSC domain, lacking familiarity with commonly used model-agnostic techniques for explainable ML. Consequently, we provided an accessible introduction to interpretable techniques, with a primary focus on their practical usage. Last, in system development, experts faced challenges in specifying their visual design and interaction preferences. To address this issue, we adopted a rapid iteration approach, engaging in frequent communication with experts to optimize the system iteratively.

Scalability and Generalizability. Currently, our experiments employ a 2×2 grid network and a straightforward traffic flow setting, a configuration widely accepted in the traffic domain and endorsed by our domain experts. Realizing the necessity to incorporate more intersections (agents) for real-life applications, we acknowledge that while our existing designs can handle additional agents to some extent - such as the seamless addition of new agent information to the radical and top-down layout in the Episode Detail - more effective strategies merit exploration. Consequently, given that *MARLens* is centered on TSC, its visual design could potentially be adapted for other traffic-related tasks. For example, the design in Fig. 4c could facilitate the examination of traffic congestion and the coordination of different traffic signals. In terms of the utility of our system in other RL problems, the methods we used to interpret MARL models are model-agnostic, eliminating the need for users to integrate new structures into their MARL models. Although *MARLens* focuses on TSC, the tree-based design in the Policy Explainer has less connection with TSC scenario, which can be used in other RL-based tasks.

Real-world Deployment of *MARLens*. In contrast to computer vision domains, RL-based TSC models do not require extensive processing of complex visual data, thereby reducing the computational resource requirements significantly. Since training RL-based TSC models can be time-consuming, *MARLens* collects data throughout the model training process, eliminating the need for separate training or testing phases in our system. For the expert interviews and the user study, we have already deployed the system on various desktops and laptops, and generally, the overall operation is quite smooth.

Learning Curve and Evaluation. *MARLens* is designed for RL and TSC researchers aiming to enhance their understanding of MARL-based TSC. Through expert interviews and the user study, we dedicated more time to explaining the Episode Detail and the Policy Explainer due to their involvement with algorithmic interpretations of the model. Overall, users already familiar with RL-based TSC find it easier to grasp *MARLens*, while others may require a bit more time. Following a brief tutorial lasting approximately 20 minutes, users can comprehend the design objectives of different components and believe that *MARLens* facilitates a more comprehensive understanding of the model. As for evaluating our system, existing tools commonly used by researchers, such as *CityFlow* and *SUMO*, primarily serve as programming frameworks for RL model development and traffic simulation. To the best of our knowledge, there is no directly comparable system to *MARLens*, which limits our ability to provide a more extensive evaluation of our system.

10 CONCLUSION AND FUTURE WORK

In this study, we present *MARLens* to aid traffic and RL researchers in gaining a deeper, more comprehensible understanding of RL-based TSC issues. We identify a range of metrics for extracting key traffic information from the RL model. Our approach involves presenting and scrutinizing the RL model from various perspectives and dimensions, incorporating a traffic simulation module. We employed SHAP and decision tree techniques to demonstrate the relationships among different agents, shedding light on the decision process of the RL model. To assess the efficacy and user-friendliness of our system, we conducted three case studies, interviewed domain experts, and conducted a user study. The results validate that *MARLens* significantly enhances users' ability to explore the interpretability of RL-based TSC models. Looking ahead, we plan to involve more end users and evaluate the long-term utility of *MARLens*. In addition, we aim to offer more customization options and add introduce a model comparison module to enhance direct comparisons between various RL-based TSC models.

ACKNOWLEDGMENTS

This work is supported by grants from the National Natural Science Foundation of China (No. 62302531) and the Science and Technology Planning Project of Guangdong Province (No. 2023B1212060029).

REFERENCES

- [1] Z. Deng, D. Weng, S. Liu, Y. Tian, M. Xu, and Y. Wu, "A survey of urban visual analytics: Advances and future directions," *Computational Visual Media*, vol. 9, no. 1, pp. 3–39, 2023.
- [2] J. I. Levy, J. J. Buonocore, and K. Von Stackelberg, "Evaluation of the public health impacts of traffic congestion: a health risk assessment," *Environmental Health*, vol. 9, pp. 1–12, 2010.
- [3] T. Tan, F. Bao, Y. Deng, A. Jin, Q. Dai, and J. Wang, "Cooperative deep reinforcement learning for large-scale traffic grid signal control," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2687–2700, 2019.
- [4] F. Mao, Z. Li, and L. Li, "A comparison of deep reinforcement learning models for isolated traffic signal control," *IEEE Intelligent Transportation Systems Magazine*, 2022.
- [5] H. Wei, G. Zheng, V. Gayah, and Z. Li, "Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation," *ACM SIGKDD Explorations Newsletter*, vol. 22, no. 2, pp. 12–18, 2021.
- [6] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," in *Proceedings of SIGKDD*, 2018, pp. 2496–2505.
- [7] H. Wei, C. Chen, G. Zheng, K. Wu, V. Gayah, K. Xu, and Z. Li, "Presslight: Learning max pressure control to coordinate traffic signals in arterial network," in *Proceedings of SIGKDD*, 2019, pp. 1290–1298.
- [8] B. Xu, Y. Wang, Z. Wang, H. Jia, and Z. Lu, "Hierarchically and cooperatively learning traffic signal control," in *Proceedings of AAAI*, vol. 35, no. 1, 2021, pp. 669–677.
- [9] J. Yuan, C. Chen, W. Yang, M. Liu, J. Xia, and S. Liu, "A survey of visual analytics techniques for machine learning," *Computational Visual Media*, vol. 7, pp. 3–36, 2021.
- [10] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards better analysis of deep convolutional neural networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 91–100, 2016.
- [11] Z. J. Wang, R. Turko, O. Shaikh, H. Park, N. Das, F. Hohman, M. Kahng, and D. H. P. Chau, "Cnn explainer: learning convolutional neural networks with interactive visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1396–1406, 2020.
- [12] L. Gou, L. Zou, N. Li, M. Hofmann, A. K. Shekar, A. Wendt, and L. Ren, "Vatld: A visual analytics system to assess, understand and improve traffic light detection," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 261–271, 2020.
- [13] Y. Ming, S. Cao, R. Zhang, Z. Li, Y. Chen, Y. Song, and H. Qu, "Understanding hidden memories of recurrent neural networks," in *Proceedings of VAST*. IEEE, 2017, pp. 13–24.
- [14] H. Strobel, S. Gehrmann, H. Pfister, and A. M. Rush, "Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 667–676, 2017.
- [15] D. Bau, J.-Y. Zhu, H. Strobel, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "Gan dissection: Visualizing and understanding generative adversarial networks," *arXiv preprint arXiv:1811.10597*, 2018.
- [16] M. Kahng, N. Thorat, D. H. Chau, F. B. Viégas, and M. Wattenberg, "Gan lab: Understanding complex deep generative models using interactive visual experimentation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 310–320, 2018.
- [17] J. Wang, L. Gou, H. Yang, and H.-W. Shen, "Ganviz: A visual analytics approach to understand the adversarial game," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 6, pp. 1905–1917, 2018.
- [18] L. Wells and T. Bednarz, "Explainable ai and reinforcement learning—a systematic review of current approaches and trends," *Frontiers in Artificial Intelligence*, vol. 4, p. 550030, 2021.
- [19] J. Wang, L. Gou, H.-W. Shen, and H. Yang, "Dqnviz: A visual analytics approach to understand deep q-networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 288–298, 2018.
- [20] J. Wang, W. Zhang, H. Yang, C.-C. M. Yeh, and L. Wang, "Visual analytics for rnn-based deep reinforcement learning," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 12, pp. 4141–4155, 2021.
- [21] Y. Metz, E. Bykovets, L. Joos, D. Keim, and M. El-Assady, "Visitor: Visual interactive state sequence exploration for reinforcement learning," in *Computer Graphics Forum*, vol. 42, no. 3. Wiley Online Library, 2023, pp. 397–408.
- [22] W. He, T.-Y. Lee, J. van Baar, K. Wittenburg, and H.-W. Shen, "Dynamicexplorer: Visual analytics for robot control tasks involving dynamics and lstm-based control policies," in *Proceedings of IEEE Pacific Visualization Symposium*. IEEE, 2020, pp. 36–45.
- [23] R. Luo, W. Ni, and H. Tian, "Visualizing multi-agent reinforcement learning for robotic communication in industrial iot networks," in *Proceedings of IEEE Conference on Computer Communications Workshops*. IEEE, 2022, pp. 1–2.
- [24] S. Cheng, X. Li, G. Shan, B. Niu, Y. Wang, and M. Luo, "Acmviz: a visual analytics approach to understand drl-based autonomous control model," *Journal of Visualization*, pp. 1–16, 2022.
- [25] E. Saldanha, B. Praggastis, T. Billow, and D. L. Arendt, "Relvis: Visual analytics for situational awareness during reinforcement learning experimentation." in *Proceedings of EuroVis (Short Papers)*, 2019, pp. 43–47.
- [26] A. Mishra, U. Soni, J. Huang, and C. Bryan, "Why? why not? when? visual explanations of agent behaviour in reinforcement learning," in *Proceedings of PacificVis*. IEEE, 2022, pp. 111–120.
- [27] T. Kravaris, K. Lentzos, G. Santipantakis, G. A. Vouros, G. Andrienko, N. Andrienko, I. Crook, J. M. C. Garcia, and E. I. Martinez, "Explaining deep reinforcement learning decisions in complex multiagent settings: towards enabling automation in air traffic flow management," *Applied Intelligence*, vol. 53, no. 4, pp. 4063–4098, 2023.
- [28] X. Shi, J. Zhang, Z. Liang, and D. Seng, "Maddpgviz: a visual analytics approach to understand multi-agent deep reinforcement learning," *Journal of Visualization*, pp. 1–17, 2023.
- [29] A. L. Bazzan, "Opportunities for multiagent systems and multiagent reinforcement learning in traffic control," *Autonomous Agents and Multi-Agent Systems*, vol. 18, pp. 342–375, 2009.
- [30] P. Hunt, D. Robertson, R. Bretherton, and R. Winton, "Scoot-a traffic responsive method of coordinating signals," *Tech. Rep.*, 1981. [Online]. Available: <https://cir.nii.ac.jp/crid/1570009749344348928>
- [31] P. Lowrie, "Scats, sydney co-ordinated adaptive traffic system: A traffic responsive method of controlling urban traffic," 1990.
- [32] F. I. Shashi, S. M. Sultan, A. Khatun, T. Sultana, and T. Alam, "A study on deep reinforcement learning based traffic signal control

- for mitigating traffic congestion," in *Proceedings of ECBIOS*. IEEE, 2021, pp. 288–291.
- [33] S. Zhancheng, "Research on application of deep reinforcement learning in traffic signal control," in *Proceedings of ICFSP*, 2021, pp. 17–21.
- [34] C. Chouiekh, A. Yahyaouy, A. Aarab, and A. Sabri, "Road traffic: Deep q-learning agent control traffic lights in the intersection," in *Proceedings of ISCV*, 2022, pp. 1–5.
- [35] Y. Wang, T. Xu, X. Niu, C. Tan, E. Chen, and H. Xiong, "Stmarl: A spatio-temporal multi-agent reinforcement learning approach for cooperative traffic light control," *IEEE Transactions on Mobile Computing*, vol. 21, no. 6, pp. 2228–2242, 2022.
- [36] H. Wei, N. Xu, H. Zhang, G. Zheng, X. Zang, C. Chen, W. Zhang, Y. Zhu, K. Xu, and Z. Li, "Colight: Learning network-level cooperation for traffic signal control," in *Proceedings of CIKM*, 2019, pp. 1913–1922.
- [37] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.
- [38] E. Tjoa and C. Guan, "A survey on explainable artificial intelligence (xai): Toward medical xai," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 11, pp. 4793–4813, 2021.
- [39] S. Milani, N. Topin, M. Veloso, and F. Fang, "A survey of explainable reinforcement learning," *arXiv preprint arXiv:2202.08434*, 2022.
- [40] B. Hayes and J. A. Shah, "Improving robot controller transparency through autonomous policy explanation," in *Proceedings of HRI*, 2017, pp. 303–312.
- [41] S. G. Rizzo, G. Vantini, and S. Chawla, "Reinforcement learning with explainability for traffic signal control," in *Proceedings of IEEE Intelligent Transportation Systems Conference*. IEEE, 2019, pp. 3567–3572.
- [42] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [43] J. Wang, Y. Zhang, T.-K. Kim, and Y. Gu, "Shapley q-value: A local reward approach to solve global reward games," in *Proceedings of AAAI*, vol. 34, no. 05, 2020, pp. 7285–7292.
- [44] N. Topin and M. Veloso, "Generation of policy-level explanations for reinforcement learning," in *Proceedings of AAAI*, vol. 33, no. 01, 2019, pp. 2514–2521.
- [45] Y. Guo, J. Campbell, S. Stepputtis, R. Li, D. Hughes, F. Fang, and K. Sycara, "Explainable action advising for multi-agent reinforcement learning," *arXiv preprint arXiv:2211.07882*, 2022.
- [46] S. Wollenstein-Betch, C. Muise, C. G. Cassandras, I. C. Paschalis, and Y. Khazaeni, "Explainability of intelligent transportation systems using knowledge compilation: a traffic light controller case," in *Proceedings of IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2020, pp. 1–6.
- [47] L. Schreiber, G. d. O. Ramos, and A. L. Bazzan, "Towards explainable deep reinforcement learning for traffic signal control," in *Proceedings of LatinX in AI Workshop@ ICML 2021, LXAI. LXIA*, 2021.
- [48] T. Jaunet, R. Vuillemot, and C. Wolf, "Drlviz: Understanding decisions and memory in deep reinforcement learning," in *Computer Graphics Forum*, vol. 39, no. 3. Wiley Online Library, 2020, pp. 49–61.
- [49] M. Noaen, A. Naik, L. Goodman, J. Crebo, T. Abrar, Z. S. H. Abad, A. L. Bazzan, and B. Far, "Reinforcement learning in urban network traffic signal control: A systematic literature review," *Expert Systems with Applications*, vol. 199, p. 116830, 2022.
- [50] G. Zheng, X. Zang, N. Xu, H. Wei, Z. Yu, V. Gayah, K. Xu, and Z. Li, "Diagnosing reinforcement learning for traffic signal control," *arXiv preprint arXiv:1905.04716*, 2019.
- [51] H. Wei, G. Zheng, V. Gayah, and Z. Li, "A survey on traffic signal control methods," *arXiv preprint arXiv:1904.08117*, 2019.
- [52] P. Mannion, J. Duggan, and E. Howley, "An experimental review of reinforcement learning algorithms for adaptive traffic signal control," *Autonomic Road Transport Support Systems*, pp. 47–66, 2016.
- [53] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 128–135, 2010.
- [54] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mor-datch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [55] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.
- [56] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, "Multiagent cooperation and competition with deep reinforcement learning," *PloS One*, vol. 12, no. 4, p. e0172395, 2017.
- [57] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.
- [58] G. Paczoly and I. Harmati, "A new advantage actor-critic algorithm for multi-agent environments," in *Proceedings of ISMCR*, 2020, pp. 1–6.
- [59] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1086–1095, 2019.
- [60] H. Mei, X. Lei, L. Da, B. Shi, and H. Wei, "Libsignal: An open library for traffic signal control," *Machine Learning*, pp. 1–37, 2023.
- [61] S. Gratzl, A. Lex, N. Gehlenborg, H. Pfister, and M. Streit, "Lineup: Visual analysis of multi-attribute rankings," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2277–2286, 2013.
- [62] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of Machine Learning Research*, vol. 9, no. 11, 2008.

Yutian Zhang is currently working towards the M.S. degree in the School of Intelligent Systems Engineering at Sun Yat-sen University. He received a B.S. degree in transportation engineering from Sun Yat-Sen University. His research interests include interpretable machine learning, visual analytics and transportation big data.



Guohong Zheng is currently working towards the M.S. degree in the School of Intelligent Systems Engineering at Sun Yat-sen University. He received a B.S. degree in transportation engineering from Sun Yat-Sen University. His research interests include interpretable reinforcement learning, visual analytics and transportation video analysis.



Zhiyuan Liu is currently a senior undergraduate student in the School of Intelligent Systems Engineering at Sun Yat-sen University, majoring in transportation engineering. His research interests are in vehicle control, intelligent self-driving, and reinforcement learning.



Quan Li is currently a tenure-track assistant professor at School of Information Science and Technology, ShanghaiTech University. He received his Ph.D. from the Hong Kong University of Science and Technology. His research interests lie in visual analytics, interpretable machine learning, and human-computer interaction. For more details, please refer to <https://faculty.sist.shanghaitech.edu.cn/liquan/>.



Haipeng Zeng is currently an assistant professor in School of Intelligent Systems Engineering at Sun Yat-sen University (SYSU). He obtained his Ph.D. in Computer Science from the Hong Kong University of Science and Technology. His research interests include data visualization, visual analytics, machine learning and intelligent transportation. For more details, please refer to <http://www.zenghp.org/>.

