Intelligent Transportation System: The Applicability of Reinforcement Learning Algorithms and Models

S. P. Krishnendhu and Prabu Mohandas

Abstract Nowadays, many research works that associate real-time data widely use an unsupervised artificial intelligence (AI) technique, namely reinforcement learning (RL). Its fast adaptiveness to the dynamicity drags the attention of researchers who works in real-time traffic signal control systems. The scope of RL in most of the research problems remains remarkable with its peculiar characteristics. This paper reviews the basic concepts of RL, along with RL algorithms and models with an emphasis on traffic signal control (TSC). TSC is one among the trending applications of RL. Traffic congestion control with less human intervention is a challenging task of the intelligent transportation system (ITS). It not only helps traffic managers to get a grip over the traffic operation situation and analyze congestion, but also assists travelers to avoid congestion. Considering its significance, we have chosen TSC as the basis to explain the RL algorithms and models presented in this paper. In addition to such a comprehensive review, we have also provided a list of open challenges which when addressed can take the research in this area to considerable heights.

Keywords Traffic signal control · Reinforcement learning · Intelligent transportation system · Artificial intelligence · Supervised learning

1 Introduction

Traffic congestion has become an annoying and a complicated issue in most of the urban areas. A smart and efficient traffic controlling mechanism is the solution to this problem. Moreover, such a system can provide abundant advantages such as smooth traffic flow, and reducing unwanted waiting time in traffic junctions. Better managing of traffic at bottleneck junctions is essential as the traffic demands rise, failure in which is sure to cause congestions. Congestion can mostly occur in a junction if most

S. P. Krishnendhu (✉) · P. Mohandas
National Institute of Technology, Calicut, Kerala, India
e-mail: krishnendhusp@gmail.com

S. M. Thampi et al. (eds.), *Advances in Computing and Network Communications*, Lecture Notes in Electrical Engineering 735,
https://doi.org/10.1007/978-981-33-6977-1_41

of the vehicles are waiting for the signal to turn green. Unfortunately, the current traffic systems fail to consider real-time parameters that affect traffic congestion.

Thus, many research works are ongoing in traffic regulatory systems to avoid the challenge of traffic congestion. The automation of traffic regulatory systems is related to many fields such as *image processing (IP)*, *machine learning (ML)*, and *Internet of things (IoT)*. Previously, *traffic signal control (TSC)* models did not significantly address the inconveniences caused by over-saturation, delays due to unexpected events, and climate change. Data collected from traffic networks at different times were used to control green signals based on the Webster formula [1]. However, they were not adequate to control the fast-moving traffic. Scientific and technical studies that are consistent with the fact that queue size plays a vital role in traffic control [2–4] have also failed to address green-signal vegetate, cross-blocking, and occlusion.

Nowadays, the world is witnessing a few exciting research pieces that strive to automate an optimized traffic signal that overcomes the shortcoming of existing ones by considering all the real-time facts learned from the system's surroundings, including the driver behavior [5]. The research in traffic automation is fastened right from the introduction of *Reinforcement Learning (RL)*. However, utilizing RL, the traffic regulatory system can be modified in an effective way such as the green light duration is shortened or lengthened, or even skipped according to the dynamic traffic conditions [6]. RL is highly adaptable to the dynamicity of traffic conditions irrespective of the time. This peculiar property increases the possibility of producing such a system in real.

Figure 1 shows the typical framework of an RL scenario. Here, an agent takes action (say A) in an environment. This action is interpreted into a reward (say R) and a representation of the state (say S). The result is fed back into the agent. The main problem arises in deciding an algorithm that suits the current situation. One must have a clear idea of the algorithms to select an appropriate one for the case under study. Also, when RL algorithms utilize the advantages of other techniques, the results obtained are mindblowing.

This paper focuses on different RL algorithms. The discussion introduces some of the well-known algorithms and familiarizes the environments where these algorithms can be used. The paper also addresses the advantages and disadvantages of algorithms under consideration.

The study is organized as follows. Section 2 focuses on the preliminary knowledge of the RL algorithms. Section 3 provides a foundation to lay a better understanding of the existing systems, which will act as the basis for this paper. A detailed analysis of *RL models (RLMs)* is presented in Sect. 4. Section 5 gives an overview of datasets, simulation platforms and performance metrics used in *RL-based vehicular traffic control models (RL-VTCMs)*. The open challenges and recommendations of the system and the conclusion are given in Sects. 6 and 7, respectively.

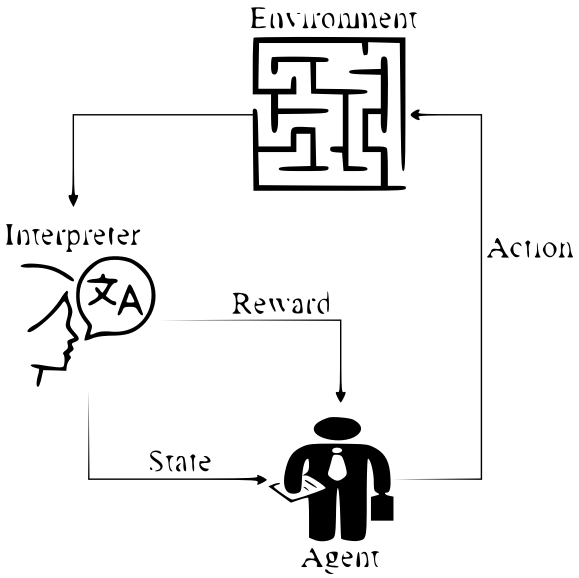

Fig. 1 Framework of the RL scenario

2 Reinforcement Learning

Reinforcement learning (RL) refers to a kind of ML method which analyzes how the software agents are ought to take actions in their environments to maximize the cumulative reward. Nowadays, it is used in various software and machines to find the most suitable path or state it should take while considering the present scenario. This section provides the preliminaries regarding RL.

The environment means the object on which the agent is acting. The agent is the RL algorithm. Initially, without any prior knowledge of how to behave, the agent starts interacting with its environment. The input is sent to the agent by the environment. The input is a state. Then, the agent takes action based on the knowledge it gained as a response to the received state. Via an interpreter, the environment sends a pair of next state and reward back to the agent. The reward, which is either positive or negative, solemnly depends on the agent's action. The negative reward is usually referred to as punishment to the agent. Also, to evaluate its last action, the agent updates its knowledge using the reward obtained. The agent iteratively learns and reaches the optimal condition.

Even now, few are confused with *Supervised Learning (SL)* and RL. Table 1 gives a comparison between RL and SL to bring in more clarity.

The sequential nature of RL can be explained as follows. The output depends on the state of current input, which depends on the previous output. i.e., the input at a particular time always considers the output of the previous cycle. Thus, a chain is formed. To predict future output, the SL algorithms apply the knowledge gained to the new data as labeled examples.

Table 1 Reinforced learning versus supervised learning

Reinforced learning	Supervised learning
Makes decisions sequentially	Decision entirely depends on the initial input
Trial and error search method and delayed reward	Starts functioning by analyzing a known training dataset
The learning algorithm interacts with its environment	The learning algorithm produces an inferred function
Labeling of the sequence of dependent decisions	Labeling of each decision since they are independent
Example: Games	Example: Object recognition

The continuous interaction with the environment benefits software agents and the machines to automatically determine the specific context's quintessential behavior to maximize its performance. After sufficient training, the ideal output is attained for any new input if the system is provided with a suitable dataset.

RL requires a reward feedback method known as the reinforcement signal. The agent learns using this reinforcement signal. The action corresponding to each reward is analyzed to find the best. The learning algorithm compares its obtained output with the correct, intended output and thereby calculate the error. Accordingly, the necessary modifications are made in the model.

2.1 Classification of Reinforcement Learning Algorithms

The RL algorithms can be classified based on different factors such as the reward, model, action space, policy. These classifications are explained below:

Based on Reward The reward-based classification mainly depends on the nature of the reward. In practical cases, RL is categorized into positive RL and negative RL. An RL algorithm is said to be positive reinforcement when an event increases the strength of the behavior; i.e., an event occurs because of a particular behavior of the agent. A reward is assigned to the agent. If that reward helps to maximize performance, then it is positive reinforcement. Alternatively, in other words, in positive RL, the reward said to be a positive effect on the behavior. In negative RL, the system is trained to stop or avoid unfavorable conditions, which reduces strength. Such an action strengthens behavior. It resists the minimum standard of performance.

Based on Model Model-free and model-based are the two classifications of RL algorithms based on the model. The *transition probability distribution (TPD)* is also known as the transition model. The model of the environment contains both TPD and *Reward Function (RF)*. In the model-free algorithm, the TPD and the RF associated with the environment are not utilized.

Let the current state be $s0$ and action be a. Performing a, the model reaches $s1$ from $s0$. The model analyzes and learns the transition probability function T. In this case, $T(s1|(s0, a))$. By successfully analyzing, the agent determines the chance to enter into a particular state from the current state by taking a specific action. In model-free algorithms, the peculiar trial-and-error method of RL algorithms is used. In each trial, the model gains some knowledge. Correct action helps the model to optimize the output. The wrong action helps the model to update itself to stay away from entering into unfavorable states. Because the model earns some knowledge from all its trials, there is no need to store the transitions.

The absence of state space and action space makes the model-free RL algorithms more demanding than model-based. The cases where the transitions have to be saved uses model-based RL algorithms. As the state-space and action-space grow, the effective utilization of storage space became impractical. Model-based RL algorithms are preferred in scenarios where the system could decide the next move based on a trained model, without interacting with the current environment. Conversely, if the system decision needs a continuous interaction with the environment, such as a real-time traffic regulation system, model-free RL algorithms will perform well.

Based on Action Space Based on the action space, RL agents can have two categories of action spaces, namely discrete and continuous action space. If the agent decides the next action from a finite action set, it is called discrete action space algorithms. Instead, in the continuous action space, a single real-value vector is used to represent the entire action space. The difference in actions cannot be expressed because of the single vector representation. In discrete action space, the fine-tuning of action selection is done. Also, discrete action space is more suited for value-based approaches. Further, a discrete action space approach is engaged in cases where small action space is required. The continuous action space is required when the size of action space grows to infinity.

Based on Policy In the policy-based RL algorithms, the main objective is to maximize the reward. The policy defines the behavior of an agent at a particular time. In other words, it is a mapping from learned states to actions to be taken when the agent reaches those states. These algorithms try to determine the action to be taken at a state to attain the maximum reward in the forthcoming steps.

The algorithm fine-tunes a vector of parameters to attain the objective. For example, to select the best action to be taken under the policy π, a vector of parameters say θ is adjusted. This example is mathematically represented as follows:

$$\pi(a|s, \theta) = Pr\{A_t = a|S_t = s, \theta_t = \theta\} \tag{1}$$

Right-hand side (RHS) of Eq. 1 means that, at time interval t, the best action to be taken is a from state s by tuning the parameter θ. Left-hand side (LHS) implies that the agent learns this knowledge. The entire learning or training phase follows the same policy.

The two types of policy-based RL algorithms are on-policy and off-policy algorithms. The agent learns the Q-function so that the probability of goodness of each action is determined. Among the results, the best one selected stochastically. Such a learning approach is known as on-policy RL algorithms. On the other hand, a

Table 2 Reinforced learning algorithms: comparison

Algorithm	Model	Policy	Action space	State space
Q-learning	Model-Free	Off-policy	Discrete	Discrete
SARSA	Model-Free	On-policy	Discrete	Discrete
Q-learning-λ	Model-Free	Off-policy	Discrete	Discrete
SARSA-λ	Model-Free	On-policy	Discrete	Discrete
DQN	Model-Free	Off-policy	Discrete	Continuous
DDPG	Model-Free	Off-policy	Continuous	Continuous
Actor-critic	Model-Free	On-policy	Continuous	Continuous
A3C	Model-Free	On-policy	Continuous	Continuous
NAF	Model-Free	Off-policy	Continuous	Continuous
TRPO	Model-Free	On-policy	Continuous	Continuous
PPO	Model-Free	On-policy	Continuous	Continuous
TD3	Model-Free	Off-policy	Continuous	Continuous
SAC	Model-Free	Off-policy	Continuous	Continuous

greedy decision is taken to take action with the best Q-value. The Q-value is learned by using other different algorithms. Such algorithms are called off-policy RL algorithms. Based on their nature, these kinds of reinforcement algorithms are sometimes referred to as stochastic and deterministic reinforcement algorithms, respectively.

Policy-based algorithms can exhibit better convergence. These algorithms are suitable even in higher-dimensional action spaces. The more attractive characteristics of these algorithms are their stochastic nature. Though the policy-based algorithms have many advantages, they still possess some disadvantages. Rather than converging into the global optimum, these algorithms converge to the local optimum. In mathematics and computer science, global optimum gives the optimal solution among every possibility. Local optimum is not preferred because it is the best solution to a problem only within a small neighborhood of possible solutions.

Also, the policy-based algorithms have higher variance. But a small variance characterizes an efficient estimation. The important reinforcement algorithms with the properties mentioned above have been compared. The main traits of them are given in Table 2.

3 Related Work

Researchers have proposed several solutions to solve conventional TSC problems. This section discusses the important among such solutions put forward by the researchers.

An RLM that uses the Q-learning algorithm with action-value approximation has been used to build an online model-free traffic signal controller [7]. This work focuses on both average delay and queue length, rather than considering only the average delay. Also, it utilized the advantage of ANN to train the model according to the temporal difference. However, [7] failed to consider unexpected dynamic scenarios in real time.

The model-free RL algorithm can contribute highly to the traffic signal control problem by combining the prior traffic knowledge with a deep RL approach, which is shown in [8]. Here, Q-learning is used to train another approach named *Mixed Q-network (MQN)*. A model can learn traffic patterns and then find out the most suitable agent. The biggest failure of such a system is finding a suitable traffic pattern detector according to the dynamic traffic structures.

Most of the earlier studies have succeeded in developing traffic signal controllers (with restricted action selection) with the help of peculiar properties of basic RL algorithms. Two RL adaptive traffic signal controllers were designed to analyze their learned policies and compare them to a Webster's controller [9]. The controllers were implemented by using asynchronous Q-learning and advanced adaptive actor-critic algorithms. The neural network function approximation has also been added to the design. Interval became constant due to the fixed green signal duration for the scenario under observation. If the action selection is made dynamic, the agent could control the environment better. Also, in the testing scenario, each intersection was controlled by an isolated RL agent. Hence, the model cannot be considered as a multiagent RL system.

Q-learning techniques maximize the number of vehicles passing a junction and adjust the roads' signals by observing the variation of queue lengths and throughput as the key parameters [10]. However, this system fails to evaluate the accuracy of the model in multiple intersection roads. Also, the data transfer between the traffic island have not been considered in this study.

The time delay, the number of idle vehicles, and the combined saturation were estimated from the experience to learn and determine the optimal actions preserving the traffic signal timing efficiently [11]. The work modularized the actual continuous traffic states for simplification purposes.

The spectacular properties of *Deep Q-Network (DQN)* have a lot to help with TSC models [12]. Further, DQN is used in learning models in modern ride-sharing platforms [13]. The model-free DQN learns the optimal vehicle dispatch policies from its interaction with the environment. However, some crucial detailing is missing in this study. Scalability, fault tolerance, reliability, and availability of shared data also have to be considered.

European countries are well versed with the advantage of group-based signal control that provides flexible phase structures. Most of the existing systems used simple timing logic in implementation. Jin and Ma [14] try to formulate the existing system as an adaptive multi-agent system by incorporating Q-learning and SARSA. Nevertheless, the work lacks the handling of real-time scenarios and the issues associated.

R-Markov average reward technique (RMART) is suited for an environment among signal controllers in a connected vehicle environment [15]. The research took eighteen signalized intersections to implement the idea in a hypothetical network by assuming the learning parameter and discount factor to be arbitrary. Aragon-Gómez and Clempner [16] address a multiagent continuous-time schedule problem and proposes a learning scheme for it. Thus introduced an RLM (based on the temporal difference method) by observing traffic signal control problems as *continuous-time Markov games (CTMG)*. Transition rates and reward points are calculated accordingly. However, some shortcomings in this work include the lack of incorporation of a collaborative approach and the method's robustness when exposed to a real-time environment.

Vehicles are used not only for travel. They are also used for goods transportation. Therefore, traffic control is one of the primary demands for manufacturing companies too. In the future, more emphasis will be given on automation. Therefore, the product's timely delivery to the consumer is also a factor that affects the product's quality and production cost. The *deep reinforcement learning (DRL)* model paves a solution to this via dynamic routing strategy [17]. The traffic states and actions can be predicted using DRL combined with a Q-learning step and a *recurrent neural network (RNN)*. Hence by reducing the delivery time and delay, the different combinations of states, actions, rewards are utilized for the modeling. Still, the model failed to consider a few other dependent factors/causes of traffic congestion.

Lack of proper traffic control not only creates traffic congestion but also adversely affects safety, time, efficiency, and energy. These problems are also heating up with the advent of autonomous cars and electric cars. Therefore, ongoing research work has begun using RL techniques to address these issues [18–20]. RL techniques can also be used intelligently and appropriately to facilitate learning and problem-solving in many other traffic-related areas [21, 22].

4 RL Models in Vehicular Traffic

This section presents some of the RLMs that are used in vehicular traffic for traffic automation purposes. A review of RLMs and their strengths to address traffic control challenges is given in Table 3. Also, Table 4 reports RLMs and the attributes for the vehicular traffic regulation system.

4.1 Multiagent Reinforcement Learning

Most intelligent systems nowadays highly depend on multiple agents competing with each other to improve the system's overall behavior. Such a process that incorporates RL algorithms is known as *multiagent reinforcement learning (MARL) Algorithm*. The combinational availability of *state-action pairs (SAPs)* increases exponentially with the number of agents. In other words, the number of agents is directly proportional to the number of SAPs. In MARL, the agents exchange information. Based on all the available and received data, the agents coordinate their actions to achieve global Q-value optimization. The most attractive feature of MARL is its scalability (i.e., adding new agents quickly) [12, 23–25].

Table 3 Summary of RL models

RLM	Strengths	Challenges
MARL	Highly scalable, inherently robust, follows top-down approach	Exponential complexity, curse of dimensionality, difficult to state the learning goal
MBRL	Heeds the longer-term effects of an action under a state	Exponential decay in eligibility trace due to trace decay parameter
MPRL	Follows top-down approach, addresses the curse of dimensionality	The algorithms converge to a point after a finite number of iterations
RLFA	Addresses the curse of dimensionality, saves computation time and memory space	May produce an inconvenient result, adjustable weights oscillate within a region

Table 4 Summary of RL models for vehicular traffic regulation systems

Representation	Attributes	MARL [2]	MBRL [27]	MPRL [3]	RLFA [4]
Agent	Traffic signal control	Yes	No	Yes	Yes
	Traffic movement	No	Yes	No	No
State	Queue size	Yes	No	Yes	Yes
	Current traffic phase	No	Yes	Yes	No
	Traffic phase split	No	Yes	Yes	No
Action	Traffic phase type	Yes	Yes	Yes	Yes
	Traffic phase split	No	Yes	No	No
Reward	Variation of vehicular delay	No	Yes	No	No
	Waiting time	Yes	No	No	Yes
	Variation of queue size	Yes	No	Yes	Yes

The main challenge for the agents in a shared dynamic environment lies in learning the situation and making a better decision. The same is the case with traffic also. In vehicular traffic scenario, the action of an agent at an intersection point can affect and vary with the agent's decision at the neighboring intersection point, which may also affect the agent's self-performance. In case of a wrong decision, there is a high probability of having high congestion in the nearby intersections. Hence, each agent should take and communicate optimal actions and coordinate with each other. MARL is a helpful model in such cases [2, 26]. MARL that tries to optimize the global Q-value is used for the traffic regulation system [2]. The inappropriateness of the traffic phase is tackled using a distributed model [26].

4.2 Multistep Backup Reinforcement Learning

The optimal action decided by a typical RL algorithm highly depends on the present state. Usually, an action affects a consecutive series of states. In *multistep backup reinforcement learning (MBRL)*, the average outcomes of the temporal differences are calculated inside an episode (a series of time instants). Based on this data, the agent updates the Q-values. MBRL focuses on the long-term payoff for an action related to a state. The average effects of temporal difference are attained using most fitting traces.

The MBRL model cut down the average hold-up time by considering the phase sequence and phase split of traffic in an intersection with a single lane traffic network [27]. Traffic phases with grouped individual traffic give the traffic phase split for processing. An episode is a duration between activation and termination of the green signals with the combination of traffic movements. Each time a SAP is visited, its value is set to one. This value gets updated for all the visits, and the eligibility trace adds more credit to recent SAPs. The temporal difference is being weighted using the eligibility traces. The Q-value of an episode is updated using this temporal difference. A trace decay parameter exponentially decays the eligibility trace of an unvisited SAP.

4.3 Max-Plus Reinforcement Learning

Agents in a coordination graph are interconnected. A max-plus algorithm calculates and exchanges the local and global payoffs among these agents. As part of the optimal joint action, agents use the payoff values to determine their corresponding action. A *max-plus reinforcement learning (MPRL)* follows a top-down approach. This modularization helps them to confront the challenge of dimensionality. The probability of better results in an oversaturated network is calculated by incorporating MRPL in the reward structure of Q-learning agent in the design of a traffic signal control [3].

The agent *i* sends locally optimized payoffs to its neighbor *j* via the edges connecting them. The action taken by *j* determines the payoff. After a finite number of iterations, the algorithm converges to a fixed point. It is possible to increase the throughput of traffic signal and reduce the number of stops per vehicle to some extent [3].

4.4 Reinforcement Learning with Function Approximation

Commonly, in a shared dynamic space, the number of SAPs can be huge in number. The SAPs increase exponentially when the number of agents increases, leading to a diminishing scalability scope. Thus, RL faces the challenge of dimensionality. This issue can be solved to some extend by introducing *function approximation (FA)* logic in RL. Instead of many SAPs, FA stores and pays attention to an appreciably smaller amount of features. Thus reduces memory/storage capacity, improve scalability, and reduce learning time. In *RL with FA (RLFA)*, Q-values are represented using tunable weight vectors and feature vectors [1, 4, 28, 29].

Consider a real-world traffic network based on Bangalore, 2 × 2 and 3 × 3 grids, and sixteen trivial streets traffic network using a centralized model. Here, the RLFA approach addresses the challenge in traffic phase sequence in a two-way intersection by optimizing the global system performance. RLFA helps to increase throughput and reduce waiting time [4, 28].

5 Datasets, Simulation Platforms, and Performance Metrics Analysis of RL-VTCMs

This section includes analyzing performance metrics used in traffic-related research and simulation platforms used in such studies. Also, it investigates the datasets used in RL-VTCMs. Table 5 gives a summary of the performance metrics.

5.1 Benchmarked Datasets for RL-VTCMs

Some of the benchmarked datasets that focus on autonomous navigation are ADE20K [30], *Berkeley Deep Drive (BDD)* [31], Cityscapes [32], Camvid [33], Daimler [34], IDD [35], KITTI [36], Leuven [37], and Mapillary Vistas [38]. The different lighting circumstances and the multiple cameras and sensors in the cities help the Cityscapes provide a large amount of data. The Mapillary Vistas Dataset creates the imagery of street scenes. Images from different angles of the road and its surroundings are present

Table 5 Summary of performance measures

Performance measures	MARL [25]	MBRL [27]	MPRL [3]	RLFA [4]
Lower average waiting time	✓			✓
Lower average delay	✓	✓		
Lower number of stops per vehicle			✓	
Smaller queue size		✓		
Higher throughput	✓		✓	✓

in this dataset, irrespective of the cameras that captured them. They have no video data. The Berkeley Deep Drive Dataset concentrates on autonomous navigation. For ADE20K, the general locale parsing issue is the main area of interest. Dashboard cameras are used on the BDD100K to capture images. The glass in front of the cameras adversely affects the image quality. It can get worse in rainy conditions. IDD can be used to ensure security and reliability in unusual and extreme cases.

5.2 Simulation Platforms

Some discrete-event simulators are developed using programming languages such as C/C++ and tools such as MATLAB. There exist macroscopic and microscopic approaches for traffic simulators with a graphical user interface (GUI). Most traffic simulators embrace the microscopic approach, including VISSIM, SUMO, TSIS, and ITSUMO.

5.3 Performance Measures

Appropriate performance measures are required to assess the merits of any traffic control system. These parameters are essential in RL based TSC; because an agent needs to assess his own performance to learn from experience. Some of the performance measures used in vehicular traffic are reduction of fuel consumption, reduction of emissions, the number of stops in a journey, percentage of stopped vehicles, average delay, *average trip waiting time (ATWT)*, vehicle density at different parts of the network, queue length, and average vehicle speed. Table 5 reports some of the performance measures accomplished by the RLMs and algorithms.

6 Open Challenges and Recommendations

After discussing the major algorithms and models in RL, here we examine various challenges that need to be addressed during their usage. This section throws light into the important hurdles in using RLMs and algorithms in ITS. It also includes suggestions for handling these challenges.

- **Injecting RL in unfitting circumstances-** RL is propitious and fastly advancing technique in a variety of fields such as Resources management in computer clusters, Traffic Light Control, Robotics, Games, and Chemistry. Too much reinforcement leads to states overload, followed by the diminishment of results. The inappropriate parameters and assemblage of payoff messages lead to poor system performance, even during the initial learning phase.

- **Availability of data** When enough data are available, SL methods are preferred. This is due to the fact that when action space is large enough, the RL algorithm becomes time-consuming.
- **Real-time environment** In a shared and dynamic environment like traffic regulation, RL algorithms and models have to include the recent advances in ITS to exhibit their full strength. A better traffic regulatory system comprises almost all the dynamic parameters such as traffic density, road utilization, and vehicles.
- **Self adaptiveness** Aim of the current researches is to build an automated traffic regulatory system that performs self-configuration of the dependent parameters to adapt with the dynamicity of traffic. The interoperability is usually affected by the communication overhead. Hence, the exchange of control messages needs a limit by eliminating unwanted control messages, by which the learning rate of the system also improves. The agent is expected to learn new and unexpected actions and states in the operating environment.
- **External impediments** The weather conditions such as rain, flood, fog are the factors that pull down the hope of a fully automated self-paced traffic regulatory system. Not only this, but also the traffic flow(in and out) and the disturbance in traffic flow make the problem worse. In upcoming traffic regulation proposals, all such situations have to be taken care of.

RL enhances system performance in scenarios with fewer data, such as in traffic regulatory systems. Hence, in developing countries with very few publicly available traffic datasets, RL has a huge impact in developing better VTCMs. Integrating RL with advanced technologies such as fuzzy logic, game theory, and AI; fastens the ride towards an extremely self-paced traffic regulatory system. These technologies help to include prior knowledge and obtain optimal actions. The analysis of prior traffic data, gained knowledge, approximation, and conventional control systems are required for a better traffic control model. Agents in the model use the traffic observer's information for increasing the learning rate in the (re)learning phase to achieve enhanced system performance.

7 Conclusion

In this paper, we have reviewed the RLMs and algorithms with an emphasis on the applicability in traffic regulation systems. The ability of RL algorithms to determine actions that yield highest rewards can be regarded as the prime reason for their wide acceptability. Consequently, a study on the RL algorithms can reveal the intrinsic features which in turn can be utilized effectively for handling traffic regulation issues. The paper provides such a detailed review of the RL algorithms, but it is not limited to that.

In addition to providing an in-depth analysis of various RL algorithms, the paper also discusses the issues that need to be rectified for its hurdle-free application in traffic regulation systems. These issues demand immediate attention of researchers, especially considering the fact that we are fast progressing towards a world which is 'smart' in all aspects.

References

1. B. Yin, Traffic network micro-simulation model and control algorithm based on approximate dynamic programming. IET Intell. Transport Syst. **10**, 186–196 (2016). https://digital-library.theiet.org/content/journals/10.1049/iet-its.2015.0108
2. K.J. Prabuchandran, A.N. Hemanth Kumar, S. Bhatnagar, Decentralized learning for traffic signal control, in *2015 7th International Conference on Communication Systems and Networks (COMSNETS)*, pp. 1–6 (2015)
3. J.C. Medina, R.F. Benekohal, Traffic signal control using reinforcement learning and the max-plus algorithm as a coordinating strategy, in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pp. 596–601 (2012)
4. L.A. Prashanth, S. Bhatnagar, Threshold tuning using stochastic optimization for graded signal control **61**, 3865–3880 (2012)
5. Y. Matsumoto, K. Nishio, Reinforcement learning of driver receiving traffic signal information for passing through signalized intersection at arterial road. Transp. Res. Proc. **37**, 449–456 (2019)
6. M.A. Wiering, Multi-agent reinforcement learning for traffic light control, in *Machine Learning: Proceedings of the Seventeenth International Conference (ICML'2000)*, pp. 1151–1158 (2000)
7. G. Tolebi, N.S. Dairbekov, D. Kurmankhojayev, R. Mussabayev, Reinforcement learning intersection controller, in *2018 14th International Conference on Electronics Computer and Computation (ICECCO)*, pp. 206–212 (2018)
8. J. Zeng, J. Hu, Y. Zhang, Training reinforcement learning agent for traffic signal control under different traffic conditions, in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 4248–4254 (2019)
9. W. Genders, S. Razavi, Policy analysis of adaptive traffic signal control using reinforcement learning. J. Comput. Civ. Eng. **34**(1), 04019046 (2020)
10. H. Joo, S.H. Ahmed, Y. Lim, Traffic signal control for smart cities using reinforcement learning. Comput. Commun. (2020)
11. X. Zhou, F. Zhu, Q. Liu, Y. Fu, W. Huang, A Sarsa (λ)-based control model for real-time traffic light coordination. Sci. World J. vol. 2014, (2014)
12. Y. Gong, M. Abdel-Aty, Q. Cai, M.S. Rahman, Decentralized network level adaptive signal control by multi-agent deep reinforcement learning. Transp. Res. Interdisciplinary Perspect. **1**, 100020 (2019)
13. A.O. Al-Abbasi, A. Ghosh, V. Aggarwal, DeepPool: distributed model-free algorithm for ride-sharing using deep reinforcement learning. IEEE Trans. Intell. Transp. Syst. **20**(12), 4714–4727 (2019)
14. J. Jin, X. Ma, Adaptive group-based signal control by reinforcement learning. Transp. Res. Proc. **10**, 207–216 (2015). http://www.sciencedirect.com/science/article/pii/S2352146515002574, 18th Euro Working Group on Transportation, EWGT, 14–16 July 2015 (Delft, The Netherlands, 2015)
15. H.A. Aziz, F. Zhu, S.V. Ukkusuri, Learning-based traffic signal control algorithms with neighborhood information sharing: an application for sustainable mobility. J. Intell. Transp. Syst. **22**(1), 40–52 (2018)

16. R. Aragon-Gómez, J.B. Clempner, Traffic-signal control reinforcement learning approach for continuous-time markov games. Eng. Appl. Artif. Intell. **89**, 103415 (2020). http://www.sciencedirect.com/science/article/pii/S0952197619303239
17. Y. Kang, S. Lyu, J. Kim, B. Park, S. Cho, Dynamic vehicle traffic control using deep reinforcement learning in automated material handling system, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 9949–9950 (2019)
18. X. Qi, Y. Luo, G. Wu, K. Boriboonsomsin, M. Barth, Deep reinforcement learning enabled self-learning control for energy efficient driving. Transp. Res. Part C: Emerg. Technol. **99**, 67–81 (2019)
19. Y. Wu, H. Tan, J. Peng, H. Zhang, H. He, Deep reinforcement learning of energy management with continuous control strategy and traffic information for a series-parallel plug-in hybrid electric bus. Appl. Energy **247**, 454–466 (2019)
20. C. You, J. Lu, D. Filev, P. Tsiotras, Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning. Robot. Autonomous Syst. **114**, 1–18 (2019)
21. D.M. Vlachogiannis, E.I. Vlahogianni, J. Golias, A reinforcement learning model for personalized driving policies identification. Int. J. Transp. Sci. Technol. (2020)
22. Y. Ye, X. Zhang, J. Sun, Automated vehicle's behavior decision making using deep reinforcement learning and high-fidelity simulation environment. Transp. Res. Part C: Emerg. Technol. **107**, 155–170 (2019)
23. L. Busoniu, R. Babuska, B. De Schutter, Multi-agent reinforcement learning: a survey, in *2006 9th International Conference on Control, Automation, Robotics and Vision* (IEEE, New York, 2006), pp. 1–6
24. L.L. Lemos, A.L. Bazzan, Combining adaptation at supply and demand levels in microscopic traffic simulation: a multiagent learning approach. Transp. Res. Proc. **37**, 465–472 (2019)
25. S. El-Tantawy, B. Abdulhai, H. Abdelgawad, Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): methodology and large-scale application on downtown Toronto. IEEE Trans. Intell. Transp. Syst. **14**(3), 1140–1150 (2013)
26. K. Zhang, Z. Yang, T. Başar, Multi-agent reinforcement learning: a selective overview of theories and algorithms. arXiv preprint arXiv:1911.10635 (2019)
27. J. Jin, X. Ma, Adaptive group-based signal control using reinforcement learning with eligibility traces, in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pp. 2412–2417 (2015)
28. L.A. Prashanth, S. Bhatnagar, Reinforcement learning with function approximation for traffic signal control **12**, 412–421 (2011)
29. T. Chu, J. Wang, Traffic signal control with macroscopic fundamental diagrams, in *2015 American Control Conference (ACC)*, pp. 4380–4385 (2015)
30. B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, A. Torralba, Scene parsing through ADE20K dataset, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5122–5130 (2017)
31. F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, T. Darrell, Bdd100k: a diverse driving video database with scalable annotation tooling. arXiv preprint arXiv:1805.04687 2(5), 6 (2018)
32. M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding. CoRR http://arxiv.org/abs/1604.01685 (2016)
33. G.J. Brostow, J. Fauqueur, R. Cipolla, Semantic object classes in video: a high-definition ground truth database. Pattern Recogn. Lett. **30**(2), 88–97 (2009)
34. T. Scharwächter, M. Enzweiler, U. Franke, S. Roth, Efficient multi-cue scene segmentation, in *German Conference on Pattern Recognition* (Springer, Berlin, 2013), pp. 435–445
35. G. Varma, A. Subramanian, A. Namboodiri, M. Chandraker, C. Jawahar, IDD: a dataset for exploring problems of autonomous navigation in unconstrained environments, in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)* (IEEE, New York, 2019), pp. 1743–1751

36. M. Menze, A. Geiger, Object scene flow for autonomous vehicles, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3061–3070 (2015)
37. B. Leibe, N. Cornelis, K. Cornelis, L. Van Gool, Dynamic 3D scene analysis from a moving vehicle, in *2007 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, New York, 2007), pp. 1–8
38. G. Neuhold, T. Ollmann, S. Rota Bulo, P. Kontschieder, The mapillary vistas dataset for semantic understanding of street scenes, in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4990–4999 (2017)