IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 10 April 2023, accepted 5 May 2023, date of publication 12 May 2023, date of current version 18 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3275883

RESEARCH ARTICLE

Multi-Agent Reinforcement Learning Based on Representational Communication for Large-Scale Traffic Signal Control

ROHIT BOKADE[1], XIAONING JIN[1], (Member, IEEE), AND CHRISTOPHER AMATO[2]
[1]Department of Mechanical and Industrial Engineering, Northeastern University, Boston, MA 02115, USA
[2]Khoury College of Computer Sciences, Northeastern University, Boston, MA 02115, USA

Corresponding author: Rohit Bokade (bokade.r@northeastern.edu)

This work was supported by the Northeastern University Tier 1 Award titled "Multi-Agent Reinforcement Learning Framework for Learning Coordination and Decision-Making."

ABSTRACT Traffic signal control (TSC) is a challenging problem within intelligent transportation systems and has been tackled using multi-agent reinforcement learning (MARL). While centralized approaches are often infeasible for large-scale TSC problems, decentralized approaches provide scalability but introduce new challenges, such as partial observability. Communication plays a critical role in decentralized MARL, as agents must learn to exchange information using messages to better understand the system and achieve effective coordination. Deep MARL has been used to enable inter-agent communication by learning communication protocols in a differentiable manner. However, many deep MARL communication frameworks proposed for TSC allow agents to communicate with all other agents at all times, which can add to the existing noise in the system and degrade overall performance. In this study, we propose a communication-based MARL framework for large-scale TSC. Our framework allows each agent to learn a communication policy that dictates "which" part of the message is sent "to whom". In essence, our framework enables agents to selectively choose the recipients of their messages and exchange variable length messages with them. This results in a decentralized and flexible communication mechanism in which agents can effectively use the communication channel only when necessary. We designed two networks, a synthetic 4×4 grid network and a real-world network based on the Pasubio neighborhood in Bologna. Our framework achieved the lowest network congestion compared to related methods, with agents utilizing $\sim 47-65\%$ of the communication channel. Ablation studies further demonstrated the effectiveness of the communication policies learned within our framework.

INDEX TERMS Multi-agent reinforcement learning, communication, traffic signal control, intelligent transportation systems, deep reinforcement learning.

I. INTRODUCTION

Rapid urbanization in recent years [1] has given rise to a growing problem of traffic congestion [2]. Recent trends also show a huge rise in ride-hailing and e-commerce services, which have contributed significantly towards the increasing number of vehicles on the road [3], [4]. The impacts of traffic congestion include increased delays and wasted fuel in addition to the impact on the environment and public health [5], [6]. Traffic signal control (TSC) is one of the challenging bottlenecks in reducing traffic congestion. The goal of TSC is to dynamically and intelligently control signal timings to reduce the number of vehicles halted on the road.

Recent advances in machine learning have opened up a wide range of opportunities for developing intelligent transportation systems solutions, including traffic signal control. Deep learning based architectures provide flexibility in processing data from various sensory inputs [7] and additionally

The associate editor coordinating the review of this manuscript and approving it for publication was Qiang Li.

serve as a useful tool for multimodal data fusion [8]. Deep reinforcement learning (RL) uses deep neural networks (DNNs) to map inputs to actions. Deep RL frameworks have shown tremendous progress in learning effective policies directly from raw sensory inputs [9]. Following these advances, deep MARL has emerged as one of the promising tools to develop effective frameworks for network-wide TSC, where each traffic light is treated as an agent that learns to select appropriate phases to minimize congestion within the network.

A straightforward way to carry over the framework of deep RL into the MARL setting is to treat all the agents as a collective entity. One can then use a function approximator, such as DNNs, to map the state into joint actions. However, the problem with this approach is that the action space grows exponentially with the number of agents. This kind of centralized control often proves impractical for large-scale applications. Furthermore, centralized approaches require access to the global state of the environment, which may not always be feasible. TSC is a large-scale problem for which decentralized execution becomes crucial. Several deep MARL frameworks have been proposed for independently controlling the traffic signals [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37]. However, to apply these methods to real-world applications, such as TSC, one must consider potential limitations of communication such as bandwidth availability [38]. In addition, allowing such unrestricted communication can be disadvantageous for several reasons. One reason is that the system incurs additional communication overhead when the messages received by an agent are unhelpful and excess communication can increase the overhead and reduce performance by adding unnecessary noise. Another reason is that it leaves the system in a state of vulnerability to adversarial attacks. Potential solutions to these problems are (1) compressing the information into a small number of bits [39], [40], [41], [42], (2) communicating only when necessary [43], [44], [45], [46], [47], [48], [49], and/or (3) communicating with selective agents [43], [45], [48], [49], [50], [51]. The majority of studies that proposed message passing mechanisms focused extensively on the aspect of improving the content of the messages by leveraging techniques from DL (e.g., attention mechanism [52], graph neural networks [53], and variational inference [54]). The lines of work that focused on addressing the problem of deciding *when to communicate* or *whom to communicate with* involved heuristics-based frameworks [43], [45], or use gating mechanisms [44], [46], [49], [50], [51].

A. CONTRIBUTION

In this paper, we propose an alternate framework for learning communication protocols that builds upon the existing Q-MIX [55] and NDQ [45] frameworks, which leverage the paradigm of centralized training and decentralized execution (CTDE) by learning a global action-value function. The global action-value is monotonically decomposed into individual action-values for decentralized execution. Facilitating communication among agents results in better action-value estimates [45]. Within our framework, QRC-TSC, agents learn how to effectively compress their environmental perception and action intentions into a message and determine which part of the message needs to be transmitted to another agent for effective coordination. Also, this decision is made independently for each available recipient, thereby, making the communication framework flexible. We utilize the *variational inference deep learning* framework [54], [56], [57], [58] to maximize the mutual information between the message sent by the sender and the actions taken by the recipient [45], which is an effective metric to measure communication performance [59]. Specifically, we model the message space as a joint distribution of generated message and communication policy (whether to send the bit of message). Through our formulation of the communication objective, we also encourage exploration over the communication policy space.

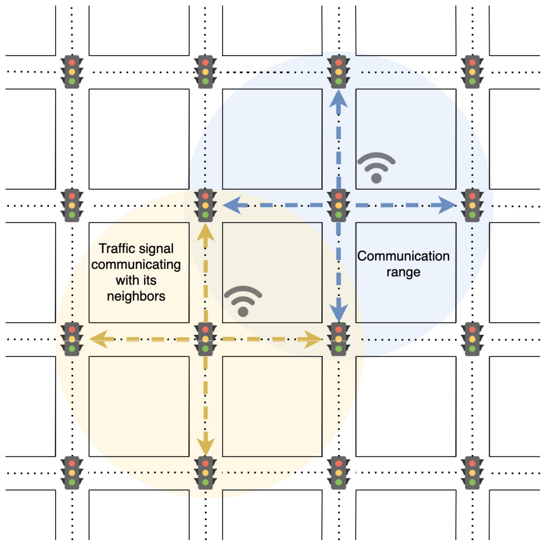


FIGURE 1. The highlighted circles represent communication range for each traffic light, i.e., each traffic light can communicate with its immediate neighbor or within 500 meters of range.

We used the SUMO simulator [60] to design two traffic networks, a 4×4 synthetic grid network with variable traffic flow and a real-world network based on the city of Pasubio. We demonstrated the efficacy of our framework in reducing the congestion level of network-wide traffic by comparing it with some of the leading communication-based MARL frameworks. We also conducted ablation studies on the communication mechanism by comparing the results of our framework with several baseline communication strategies, including full communication, no communication, and random communication. We observed that traffic signals on the network were able to dynamically adjust the number of bits they send in the messages while maximizing performance.

The rest of the paper is organized as follows. Section III provides an overview of the relevant work done in MARL which serves as the basis of our framework. Section IV discusses our framework in detail and also describes the formulation of the TSC problem within our framework. In Section V we provide the experimental setup and compare the results of QRC-TSC with other frameworks and perform ablation studies. Finally, Section VI concludes the paper and discusses potential future research directions.

II. RELATED WORK

A. INTER-AGENT COMMUNICATION IN MARL

Recently proposed algorithms (e.g., DIAL [39] and Comm-Net [40]) have made it possible to learn communication protocols through a feedback mechanism by leveraging DL techniques. DIAL is an extension to Independent Q-learning (IQL), where each agent generates both action-values and a message vector. The message vector is then passed as input to the other agent networks in the next time step, thus obtaining feedback from the receiver agents in the form of gradients.

The most relevant work to our problem is the Nearly Decomposable Q-function (NDQ) [45], which combines the communication framework of DIAL with the general learning framework of Q-MIX by utilizing the variational inference [54] technique from DL. In addition to learning communication through feedback, NDQ proposes an objective function that maximizes the mutual information (MI) between the sender's message and the recipient's action. The main idea is for agents to learn to capture the most relevant information in as few bits of message as possible. A similar metric, causal influence of communication (CIC), was proposed [59], [61] to improve communication performance without impeding the general learning process. However, NDQ uses a threshold-based heuristic to filter out unhelpful messages in its communication framework. In our work, we extend the work done in NDQ and develop a communication framework that learns to effectively select the important bits of messages.

B. DEEP MULTI-AGENT REINFORCEMENT LEARNING IN TRAFFIC SIGNAL CONTROL

The problem of TSC has been studied through the lens of MARL [10], [62], [63] by treating the traffic signal as an agent and rewarding it based on a metric that is inversely proportional to the level of congestion (queue length). Recently, with the advent of Deep MARL, many proposed solutions to the problem of TSC [12], [13], [18], [19], [21], [28], [32], [37], [64] were effective in extracting richer information from more sophisticated sensor inputs for the decision-making process [20], [25]. Communication mechanisms are a part of the progress in applying MARL in TSC domains as well. Several methods proposed for TSC [20], [27], [64], [65], [66] implemented a variety of communication mechanisms to train the traffic signals to send and receive messages from neighboring traffic signals. However, the aforementioned methods fail to avoid the pitfall of unrestricted communication. TSC is a large-scale problem where communication between traffic signals has to be wireless, which comes at the cost of limited bandwidth and requires the utilization of additional resources. Hence, the communication mechanism must be efficient in allowing traffic signals to exchange relevant information only when it is beneficial.

III. BACKGROUND

1) DEEP REINFORCEMENT LEARNING

Reinforcement learning (RL) aims at learning the optimal policy through repeated interaction with the environment. A standard RL problem can be formulated as a Markov Decision Process (MDP). At each time step t agent observes the state of the environment $s_t \in S$ and takes an action $a_t \in A$ according to policy π. Based on this action, the agent receives feedback from the environment in the form of reward r_t and transitions to the next state s_{t+1}. The objective is to maximize the total expected discounted reward $R = \sum_{t=1}^{T} \gamma^t r_t$, where $\gamma \in [0, 1]$ is the discount factor.

Deep Q-Networks (DQN) learns the action-value function

$$Q_\theta = E[R_t | s_t = s, a_t = a],$$

where θ represents the parameters of the Q-network. The action-value function can be trained recursively by minimizing the loss

$$\mathcal{L}(\theta) = E_{s,a,r,s'}[(y - Q_\theta(s, a))^2],$$

where $y = r + \gamma \max_{a'} Q_{\theta'}(s', a')$ and θ' represents the parameters of the target network. The agent selects the action that maximizes the Q-value with the probability $1 - \epsilon$ or acts randomly with probability ϵ. The set of parameters θ^- in the target network are updated in regular time intervals by copying over the parameters θ from the primary network. Double DQN [67] modifies DQN to add stabilization and avoid overestimation. In Double DQN, the target action-value is indexed from the output of the target network based on the greedy action selected by the primary network

$$y = r + Q_{\theta'}(\underset{a'}{\operatorname{argmax}}\, Q_\theta(\cdot|s')|s').$$

Both DQN and Double DQN are based on fully observable MDPs. However, in partially observable settings, an agent conditions its action-value function on the action-observation history. DRQN [68] achieves this by using recurrent neural networks. At each time step, the Q-network takes as input the observation o_t, and the hidden state h_{t-1} to approximate the action values $Q_\theta(o_t, h_{t-1}, a_t)$. This enables the agent to integrate past information to make decisions.

A. COOPERATIVE DEEP MULTI-AGENT DEEP REINFORCEMENT LEARNING

One approach to modeling multi-agent systems as RL problems is to treat the whole system as a single agent. The agent observes the true state of the environment and selects joint-actions for all the agents. This approach, however, scales poorly as the search space for joint-action increases exponentially with the number of agents in the system. A more feasible approach is to enable each agent to act independently. Thus, one could formulate the problem as a decentralized partially observable Markov decision process (Dec-POMDP), which extends the framework of MDP to multi-agent scenarios with partial observability [69]. It is defined by a tuple of $\mathcal{M} =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \Omega, \mathcal{O}, r, \mathcal{N}, \gamma >$, where $s \in \mathcal{S}$ is the global state space and $i \in \mathcal{N} \equiv \{1, \cdots, n\}$ is the finite set of agents. At time step t, each agent a selects an action $a^i \in \mathcal{A}$ resulting in a joint action vector $a \in \mathcal{A} \equiv \mathcal{A}^n$. The transition dynamics of the environment state are given by $P(s'|s, a)$. All agents receive a shared reward according to the reward function $r(s, a)$ and $\gamma \in [0, 1)$ is the discount factor. Each agent receives an observation $o^a \in \Omega$ according to the observation function $O(s, a)$. Each agent has an action-observation history $\tau^i \in \mathcal{T} \equiv (\Omega \times \mathcal{A})^*$ on which it conditions its individual policy $\pi^i(a^i|\tau^i)$. The joint policy $\boldsymbol{\pi} =< \pi^1, \cdots, \pi^n >$ induces a joint-action value function

$$Q^{\pi}(s, \mathbf{a}) = E_{s_{0:\infty};\mathbf{a}_{0:\infty}}[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s; \mathbf{a}_0 = \mathbf{a}, \boldsymbol{\pi}].$$

Some studies propose that each agent learn the global action-value [70]. Recent works have demonstrated better performance with monotonic factorization of the global-action value [55], [71]. Q-MIX [55], specifically, leverages the CTDE paradigm to learn a monotonic mapping between individual utilities and the global action-value by utilizing a mixing network $Q_{total}(\tau, a) = f(Q_1(\tau^1, a^1), \cdots, Q_n(\tau^n, a^n); \theta_{mixer})$. The weights of the mixing network θ_{mixer} are generated by a set of hypernetworks, conditioned on the state s_t, with absolute activation function to ensure monotonicity $\frac{\partial Q_{total}}{\partial Q_i} \geq 0$. The decomposition allows for decentralized action selection during execution, since the mixing network is only used for training. Thus, the mixing network can be conditioned on additional information available during the training. Recent works improved performance on complex multi-agent environments by combining Q-MIX with communication framework [43], [45]. Thus, we utilize Q-MIX as the base framework for our proposed communication mechanism.

$$\underset{a}{\operatorname{argmax}}\, Q_{total}(\tau, \mathbf{a}) = \left(\underset{a^1}{\operatorname{argmax}}\, Q_1(\tau^1, a^1), \cdots, \underset{a^n}{\operatorname{argmax}}\, Q_n(\tau^n, a^n) \right)$$

IV. PROPOSED FRAMEWORK

A. PROBLEM FORMULATION

We extend the framework of Dec-POMDP to incorporate inter-agent communication. We formulate the traffic signal network as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $v_i \in \mathcal{V}$ is the set of nodes and $v_{ij} \in \mathcal{E}$ is the set of edges. Each node represents an agent (traffic signal) and each edge represents the connectivity between agents. The neighborhood for a node v is defined as $\mathcal{N}(v) = \{u \in \mathcal{V}|(u, v)\} \in \mathcal{E}$ and the adjacency matrix $\mathbf{A}$ is a $n \times n$ matrix with $A_{ij} = 1$ if $e_{ij} \in \mathcal{E}$ and $A_{ij} = 0$ if $e_{ij} \notin \mathcal{E}$. We design communication framework

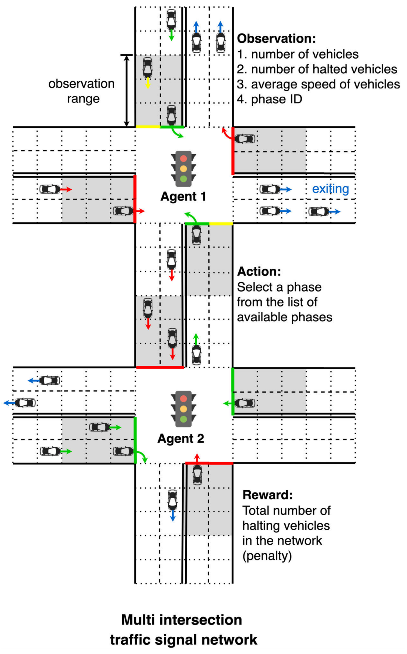

observation range
Observation:
1. number of vehicles
2. number of halted vehicles
3. average speed of vehicles
4. phase ID
Agent 1
exiting
Action:
Select a phase from the list of available phases
Agent 2
Reward:
Total number of halting vehicles in the network (penalty)
Multi intersection traffic signal network

FIGURE 2. Prototype of a traffic signal network with two intersections. The highlighted zones on the incoming lane on each traffic light represent the range within which the traffic light can access information about the vehicles.

such that each agent is only allowed to communicate with its neighbors.

We set up the problem of TSC as a Dec-POMDP, where each traffic signal in the network is treated as an agent and the central goal of the system is to reduce network-wide congestion. The traffic signals make decisions using information about incoming vehicles, which is assumed to be accessible through sensors located near the signals. The traffic signals control the flow of traffic through the intersection by selecting a phase from the available set of phases. We discuss the details of our formulation in detail below.

1) OBSERVATION REPRESENTATION

Each traffic signal has a limited range of vision of 50 meters, within which it can obtain information related to the traffic flow. This is equivalent to the sensory information that can be obtained from practical common sensors. We implement observation collection in the environment using by placing `laneAreaDetector` of length 50 meters on each incoming lane to capture the traffic information which can be seen by the boxes highlighted in grey in Fig. 2. The observation for each traffic signal consists of: the number of vehicles $\{n_l\}_{l=1}^{L_i}$, the average normalized speed of the vehicles $\{s_l\}_{l=1}^{L_i}$, the number of halted vehicles (queue lengths) $\{q_l\}_{l=1}^{L_i}$, and the current `phaseID` of the traffic signal, where $L_i \in L$ are the incoming lanes for a traffic signal i and L is a set of all the lanes in the network.

2) ACTION REPRESENTATION

For each traffic signal i, we define its action a_i as choosing one green phase from a list of available phases. As an

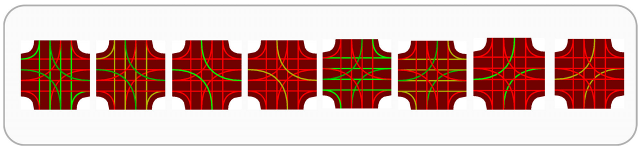


FIGURE 3. An example of the phases available for an intersection in a 4 × 4 grid network from SUMO simulator. The colored lines (red, yellow, and green) together indicate the phase of the traffic signal. The first phase (from the left) indicates an all green phase, where the vehicles are allowed to go straight and/or make turns. Each agent controls the traffic signals by selecting one of these phases.

example, Fig. 3 shows the list of phases that are available for a traffic signal in a 4×4 grid network. A traffic signal can select any green phase from its list or keep its current one, but it must then follow the next yellow phase, which is enforced by the environment. The action selection interval and the yellow phases are fixed for a duration of 5 simulation seconds.

3) REWARD

Various metrics are used for rewards in traffic signal control settings. In our study, we chose queue length q_l as the performance metric of the traffic signal controller due to its simplistic nature and its property of representing an instantaneous feedback signal. We define the objective function as minimizing the number of vehicles stopped throughout the network where $r_t \in \mathbb{R}$ is the global reward and $l \in L$ represents the lanes in the network.

B. OVERALL FRAMEWORK

In this section, we present a detailed design of QRC-TSC in the context of multi-agent Q-learning, Fig. 5. We adopt the CTDE paradigm and use Q-MIX [55] as a base learning framework. The training takes place in a centralized manner, assuming that the global state information is available. Each agent i has access to an agent network with parameters shared across all agents. This approach has been shown to accelerate learning and enhance scalability in deep MARL settings. The agent network takes as inputs the action-observation history of the agent and the incoming messages from other agents to generate action-values. The agent uses its own action values to select an action during decentralized execution. Each agent also has a communication network that takes in the agent's action-observation history and generates the message vector m_{ij} and a communication policy c_{ij} for each available recipient agent $j \in \mathcal{N}(i)$. This can be seen in the communication module in Fig. 4. The message is then gated

$$\hat{m}_{ij} = (m \odot c)_{ij}$$

[1] based on the communication action c_{ij}. The parameters of the communication network are also shared across agents. The mixing network combines the individual action-values of the agents $Q_i(\tau_i, a_i, \hat{m}_{ij}; \theta_{agent})$ to compute the join-action value function Q_{total}. The weights of the mixing network are generated by a set of hypernetworks conditioned on the

[1] $\odot$ represents elementwise multiplication.

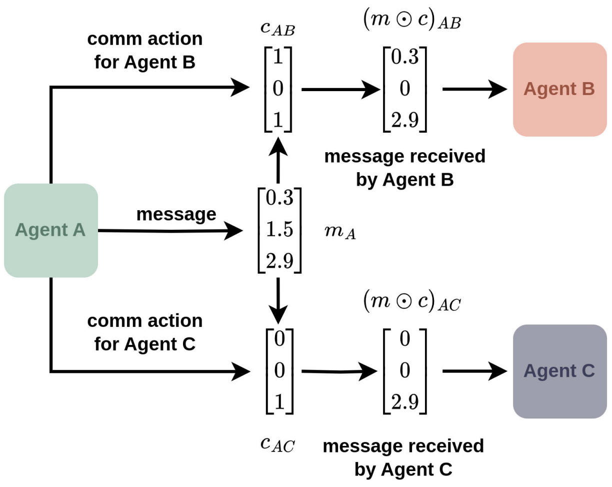


FIGURE 4. Example of the proposed communication framework. Agent A generates a message space m_A and communication action c_{AB} and c_{AC} for agents B and agent C respectively. The message is then gated based on the communication action and sent to respective agents.

state s. We use DIAL [39] as the base communication framework and we improve on it in the following ways:

1) We use variational inference to maximize the mutual information between the sent messages (including the communication action) and the recipient's action.
2) We introduce an entropy regularization term for the communication policies, enabling controlled exploration in the communication action space.
3) Communication policies are differentiable, allowing for end-to-end training.

C. COMMUNICATION IN QRC-TSC

In our framework, each agent learns communication protocols through feedback from the recipient agents. Feedback is received in the form of gradients during backpropagation [39], [40]. Thus, the entire network architecture can be trained from a single objective function. Our goal in this work is to train agents to quickly and effectively learn communication protocols. Therefore, agents must learn the communication policy and send messages that reduce the uncertainty in the recipient's policy. To this end, we aim to maximize the mutual information between the sender's message and the recipient's policy, similar to NDQ [45]. This metric was previously proposed [59] as one of the key metrics to measure communication performance. Therefore, it makes sense to integrate such a metric into the objective function and explicitly maximize it.

First, we model outgoing messages as a joint distribution $p(m_{ij}, c_{ij})$ of the message generated m_{ij} and its communication actions c_{ij} by agent i for agent j.

$$p(m_{ij}, c_{ij}|\tau_{ij}) = p(m_{ij}|\tau_{ij})p(c_{ij}|\tau_{ij}) \tag{1}$$

[58]. Specifically, each agent i generates a shared latent message distribution (a multivariate Gaussian) of size from

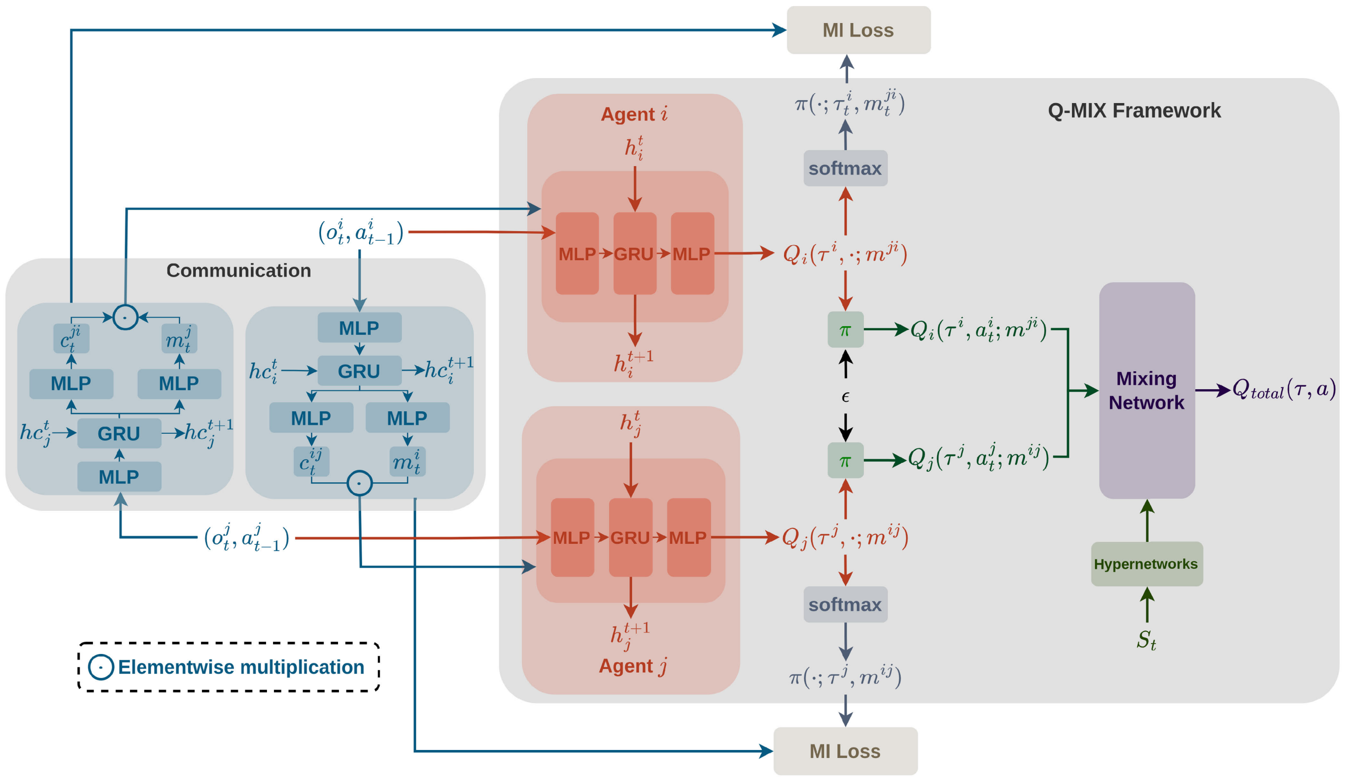


FIGURE 5. **Architecture of QRC-TSC with two agents. Each agent uses a communication network (shown in the communication block) in addition to the agent network. The communication network takes the action-observation history (o_t^i, a_{t-1}^i) of the agent i as input and outputs both the message m_t^{ij} and a communication action c_t^{ij} for the recipient j at time t.**

which a message vector $\mathbf{m}_i$ is sampled and a discrete communication policy distribution (encoded as Bernoulli) which decides which bits of the messages are to be sent to agent j.

This decision c_{ij} is made independently for each agent $j \in \mathcal{N}(i)$ in the neighborhood. Similar to the approach proposed in [56] and [57], we use Gumbel-sigmoid as a continuous approximation of the categorical variables. The Gumbel-max trick allows for differential sampling and does not suffer from high variance like the REINFORCE algorithm [56]. Thus, our framework is end-to-end differentiable.

The communication action c_{ij} acts as a mask over the messages during execution. To achieve this, we use differential relaxation of categorical/discrete variables [56], [57]. Gumbel-sigmoid can be considered as a continuous relaxation of the Bernoulli distribution and can be written as

$$\sigma(\alpha_l) = \text{sigmoid}((\alpha_i + g_l - g_m)/\lambda) \tag{2}$$

where g_l and g_m are samples from $Gumbel(0, 1)$ distribution and λ is the temperature parameter.

Next, we discuss the objective function $J_c(\theta_c)$ for learning communication. We maximize the mutual information between the sender's message and the recipient's policy.

$$I_{\theta_c}(\pi_j(\cdot|\tau_j); \hat{m}_{ij}|\tau_j, \hat{m}_{(-i)j}), \tag{3}$$

where $\pi_j(\cdot|\tau_j, \hat{m}_j^{in}) = \text{softmax}(Q(\cdot; \tau_j, \hat{m}_j^{in}))$ represents the policy of the agent conditioned on its action-observation history and incoming messages, $\hat{m}_{ij}$ is the resulting outgoing message from agent i to agent j, and θ_c is the set of parameters of the communication network.

$$J_c(\theta_c) = \sum_{j=1}^{n} [\underbrace{I_{\theta_c}(\pi_j(\cdot|\tau_j); \hat{m}_{ij}|\tau_j, \hat{m}_{(-i)j})}_{(1)} - \underbrace{\beta I_{\theta c}(\hat{m}_{ij}; \tau_i)}_{(2)}], \tag{4}$$

where β is the scaling factor that controls the tradeoff between the expressiveness and compressiveness of the messages. Since our objective is to maximize the mutual information it is sufficient to derive the objective as the lower bound for the term. The lower bound [45], [54] for the mutual information objective, the first term in (4) can be given as

$$I_{\theta_c}(\pi_j(\cdot|\tau_j); \hat{m}_{ij}|\tau_j, \hat{m}_{(-i)j}) \\ \geq \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}, m_{ij}^{in}, c_{ij}^{in} \sim f_c(\tau;\theta_c)}[-\mathcal{CE}(\pi_j(\cdot; \tau_j, \hat{m}_j^{in}) \| q_{\theta_r}(\cdot; \tau_j, \hat{m}_j^{in})] \tag{5}$$

where $\boldsymbol{\tau}$ is the joint local action-observation history of the agents sampled from the replay memory $\mathcal{D}$ and $\mathcal{CE}$ is the cross-entropy. The posterior estimates are given by $q_{\theta_r}(\cdot; \tau_j, \hat{m}_j^{in}) = q_{\theta_r}(\cdot; \tau_j, (m \odot c)_{-(j)j}^{in}) = q_{\theta_r}(\cdot; \tau_j, m_{-(j)j}^{in}, c_{-(j)j}^{in})$ and parameters θ_r are shared across all the agents.

The second term, analogous to the variational bottleneck objective in [54], is the mutual information between the agent's action-observation history τ^i and the messages generated m^i.

$$\beta I_{\theta c}(\hat{m}_{ij};\tau_i) = \beta D_{KL}(p(m_{ij},c_{ij}|\tau_i)\|q_{\theta_r}(m_{ij},c_{ij}|\tau_i)) \\ = \underbrace{\beta_m D_{KL}(p(m_{ij}|\tau_i)\|q_{\theta_r}(m_{ij}|\tau_i))}_{(1)} \\ + \underbrace{\beta_c D_{KL}(p(c_{ij}|\tau_i)\|q_{\theta_r}(c_{ij}|\tau_i))}_{(2)}, \qquad (6)$$

The first term in (6) controls the tradeoff between maximizing the mutual information between the message m_{ij} and agent j's policy $\pi_j(\cdot;\tau_j,\hat{m}_{ij})$ and being compressive about the action-observation history τ_i. The second term in (6) regularizes the communication policy. This encourages exploration of varied communication policies, which can be controlled by β_c.

Combining equations (5) and (6), we can write the loss function for the communication objective as:

$$\mathcal{L}(\theta_r,\theta_c) = \\ \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D},m_{ij}^{in},c_{ij}^{in}\sim f_c(\tau;\theta_c)}[\mathcal{CE}(\pi_j(\cdot;\tau_j,\hat{m}_j^{in})\|q_{\theta_r}(\cdot;\tau_j,\hat{m}_j^{in})] \\ + \beta_m D_{KL}(p(m_{ij}|\tau_i)\|q_{\theta_r}(m_{ij}|\tau_i)) \\ + \beta_c D_{KL}(p(c_{ij}|\tau_i)\|q_{\theta_r}(c_{ij}|\tau_i)) \qquad (7)$$

Thus, the final loss function for training can be given as:

$$\mathcal{L}(\theta) = \mathcal{L}_{TD}(\theta) + \mathcal{L}_C(\theta_r,\theta_c), \qquad (8)$$

where

$$\mathcal{L}_{TD} = [r + \gamma \max_{a'} Q_{total}(s',a';\theta') - Q_{total}(s,a;\theta)]^2 \qquad (9)$$

is the TD loss, θ^- is the set of parameters of the target network, θ is a set of parameters for all the networks combined and $\mathcal{L}_C(\theta_r,\theta_c)$ is the total communication loss.

V. EXPERIMENTS

A. EXPERIMENTAL SETUP

We built a synthetic 4×4 grid network and a real-world network of Pasubio, Bologna as proposed by Bieker et al. [72]. Trips are generated with origin-destination pairs of the fringe edges. For both the networks, we generated variable hourly traffic, similar to [18], as shown in Fig. 6d, where the solid lines represent the high flow rates and the dotted lines represent the low flow rates. Flow rates are varied in 5-minute intervals within which the vehicles are inserted uniformly into the network with the specified flow rate. The peak flow rate is 900 veh/hr. For convenience and representation purposes, we broke down the traffic flow into two types: (1) from east-west/west-east (red lines), which starts at the beginning of the hour and (2) north-south/south-north (blue lines), which starts after 15 minutes. Both flows last for 35 minutes. Flows from the opposite direction, represented by dotted lines in Fig. 6d, are scaled down by a factor of 0.6. Every hour, a random direction is selected as the opposite direction.

1) 4×4 **grid network:** We built a two-lane synthetic 4×4 grid network of homogeneous agents. We simulated two traffic flow scenarios, one of which is selected randomly at the beginning of each simulation hour. For the first scenario, Fig. 6a, we simulated high traffic on the external edges of the network, whereas in the second scenario Fig. 6b internal edges of the network received a higher bulk of traffic flow. To induce a level of randomness in the traffic flow, a random direction was selected at the beginning of each simulation hour to have a high flow rate. Traffic flow settings in the synthetic version were not tethered to reality but were designed to test the robustness of the learning algorithm. The speed limit on all the lanes was around 14 m/s.
2) **Pasubio network:** We used the real-world network of Pasubio, Bologna. The neighborhood has a hospital and includes common routes to the football stadium, and therefor is prone to congestion. The network has 7 traffic lights, some of which control multiple junctions. 3 traffic signals have 8 phases and the rest have 4, 10, 14, and 16 phases. The heterogeneity of the real-world network made it a more challenging environment than the synthetic network. We tried to replicate the traffic flow settings from [72]. The maximum allowable speed on each lane was set to 14 m/s.

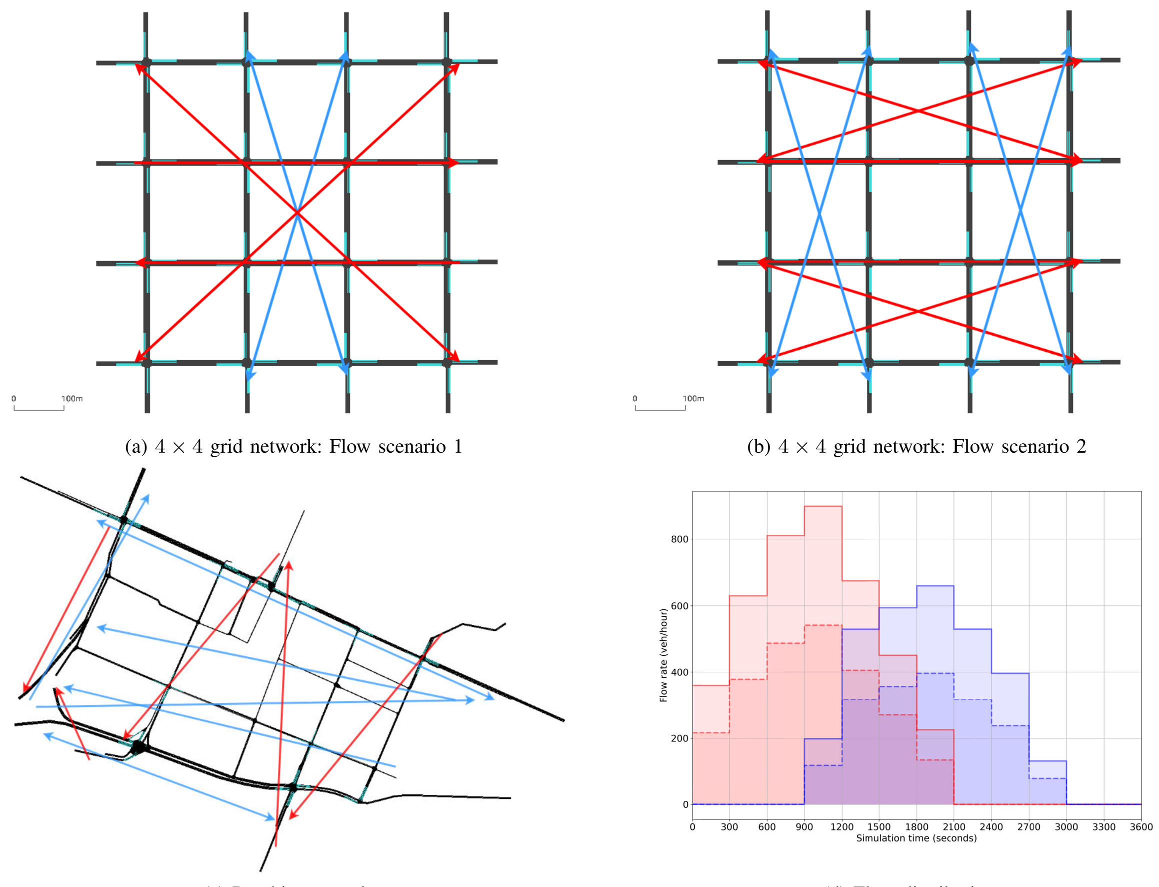
Further, we adopt the metric average number of stops or queue length to measure the performance of the algorithms on the traffic network.

B. BASELINES

In this work, we are interested in teaching the agents efficient communication policies. Specifically, our goal is to show that agents do not need to communicate all the time to be able to coordinate. Instead, agents can establish an optimal communication policy that tells the agent which parts of the message are worth sending and to which agent. To this end, we set Q-MIX [55] as the baseline framework for learning the action-value function and DIAL [39] as a baseline framework for communication. To make fair comparisons, we implemented DIAL by extending Q-MIX. We also compared our framework to NDQ [45], a state-of-the-art method to learn communication, which uses thresholds to filter out unnecessary messages. Thus, all the methods we compared our framework to only differed in the type of communication mechanism: (i) Q-MIX can be seen as a base method without communication, (ii) Q-MIX + DIAL enables learning communication via a feedback mechanism, (iii) Q+MIX + TarMAC, adds attention mechanism to messages, and (iv) NDQ can be seen as an extension to Q-MIX + DIAL, which maximizes the mutual information between the sender's message and the recipient's policy.

C. TRAINING SETTINGS

We trained all the algorithms on the grid network and Pasubio environment for 1.8 million and 3 million simulation steps, respectively. At the end of each episode, which lasted for 90 steps or 360 simulation seconds, we ran a training



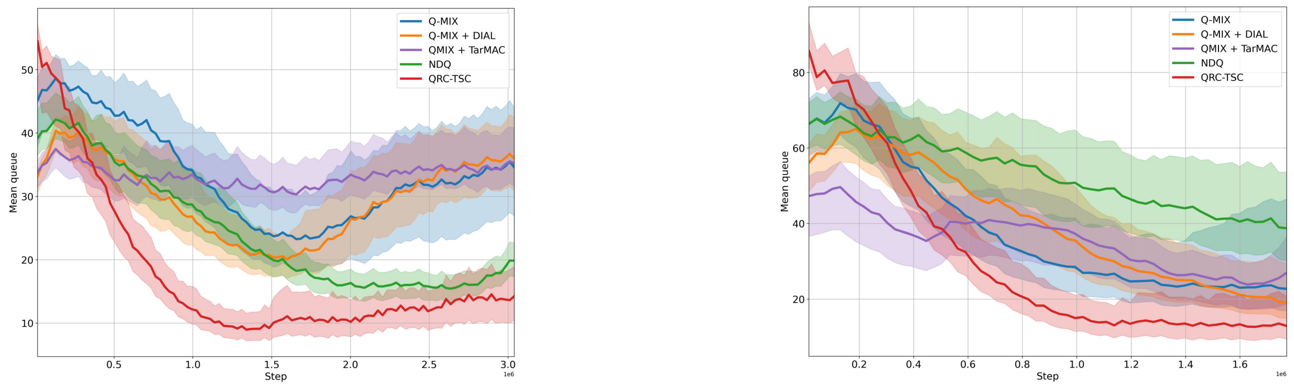
(a) 4 × 4 grid network: Flow scenario 1

(b) 4 × 4 grid network: Flow scenario 2

(c) Pasubio network

(d) Flow distribution

FIGURE 6. (a) and (b) represent the flow scenarios for the 4 × 4 grid network. (c) shows the flow in Pasubio network and (d) shows the hourly flow distribution for both the networks. The dotted lines represent flow from opposite direction whenever bidirectional flows are simulated. The red and the blue lines represent the outer and inner network flow respectively.



(a) Average queue length in Pasubio network

(b) Average queue length in 4 × 4 grid network

FIGURE 7. The plot shows average queue length throughout training (lower the better). The x-axis represents simulation steps (in millions). The solid lines show mean over 5 runs and the shaded region represents 95% CI.

iteration. To evaluate the robustness of the algorithm, we ran 10 evaluation episodes with each agent selecting its actions greedily after every 200 training episodes.

D. RESULTS

To ensure a fair comparison, we used Q-MIX as a baseline centralized training algorithm for all the algorithms based on communication. The learning curves of the algorithms are illustrated in Fig. 7. The solid lines represent the hourly average queue length of an intersection for each scenario. Queue length, which represents the number of vehicles stopped in the incoming lanes of the traffic signal, is a key metric in evaluating the performance of a traffic signal network. Evaluations were conducted after every 200 training episodes, and

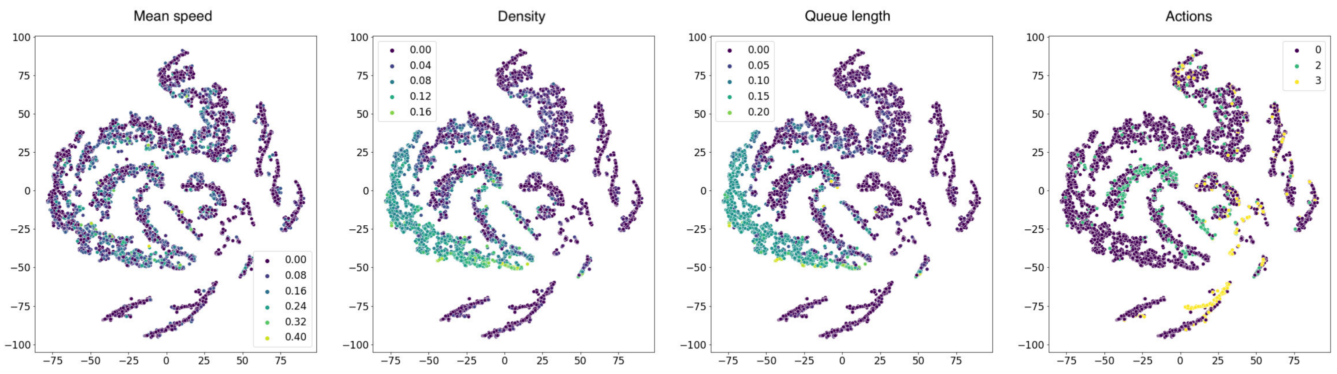


FIGURE 8. Message representation: The figure shows a t-SNE plot of the learned messages representations by an agent in the 4 × 4 grid network. The color scale in the first three plots, starting from left, represents a feature (averaged across all incoming lanes) of observations received by the agent. The color scale in the final plot represents the actions taken by the agent. We can see that the agent learns to embed messages in the latent space based on its inputs and action intentions.

the results were averaged over 15 independent runs. Additionally, we compare our algorithm to some traditional traffic signal control approaches (Fixed time [73], Self Organizing Traffic Lights (SOTL) [74], Max pressure [75]). For the fixed time algorithm, the phase duration for green phases was set to 30 seconds.

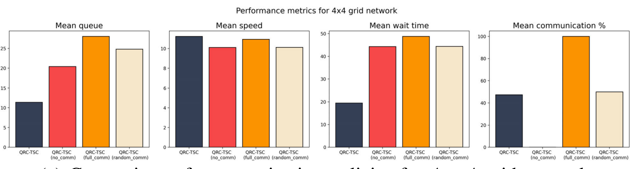
In both network scenarios, QRC-TSC performed consistently better than the other frameworks. While Q-MIX uses a centralized training mechanism to factorize the action-values, the agents operate in a completely decentralized way during execution. Purely decentralized policies can hinder the performance of systems, since traffic flow can be highly dynamic at times. On the other hand, in DIAL, the agents communicate all the time, which can decrease performance, as communication is often unnecessary and acts as additional noise. The performance of Q-MIX + DIAL, Q-MIX + TarMAC, and Q-MIX was relatively similar and significantly underperformed in the Pasubio scenario. The performance of NDQ and QRC-TSC was similar in the Pasubio network (Fig. 7a), however, NDQ performed poorly in the grid network (Fig. 7b). When considering average queue length, QRC-TSC consistently outperformed the other frameworks in both network scenarios and learned relatively stable policies, as can be seen in Fig. 7.

E. COMMUNICATION

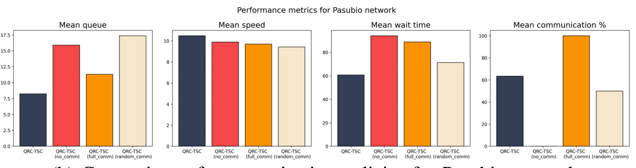
1) LEARNED MESSAGE REPRESENTATIONS

Within our framework, each agent learns to generate messages conditioned on its action-observation history. Thus, messages can be interpreted as compressed representations of the agent's inputs and its action intentions. The message space is analogous to latent space in variational autoencoders, where each variable in the latent space is independent of the other. Thus, each bit in the message represents a unique information from the sender's action-observation history.

Fig. 8 shows an example t-SNE plot [76] of message embeddings learned by our algorithm collected over



(a) Comparison of communication policies for 4 × 4 grid network



(b) Comparison of communication policies for Pasubio network

FIGURE 9. Comparison of performance of communication policies averaged across 100 test episodes. QRC-TSC (in blue) represents the performance of the communication policies learned by our framework.

100 evaluation episodes. In the first three plots from left to right, the color gradients represent features of agents inputs averaged over the number of incoming lanes: mean speed $(\frac{1}{L}\sum_l s_l)$, mean density $(\frac{1}{L}\sum_l n_l)$, and mean queue length $(\frac{1}{L}\sum_l q_l)$, respectively. These images show that the message distribution learned by the agents was correlated with its inputs, confirming that the agents learned to send meaningful information from their observations. The color labels in the fourth plot represent the actions taken by the agents, which indicates that the agents were able to effectively convey their action intentions through the message space. A key observation from this figure is that mean density and mean queue length are often correlated with each other, and hence the agent can eliminate information.

2) LEARNED COMMUNICATION POLICIES

In our framework, the agents are allowed to send 5 bit messages at each time step. Therefore, the communication policy can be seen as an action of selecting the bit of message for each recipient. This makes visualizing the communication policy for each agent in a reduced space almost

TABLE 1. Performance results of various algorithms on 4 × 4 grid and pasubio network.

				4×4 grid network				
Metrics	Q-MIX	DIAL + Q-MIX	TarMAC + Q-MIX	NDQ	**QRC-TSC**	Fixed time	SOTL	Max pressure
Mean queue length	29.62	22.17	23.53	47.91	**7.81**	62.57	32.42	24.27
Mean wait time (s/veh)	53.83	44.15	46.48	37.20	**16.97**	78.81	50.96	50.42
Mean speed (m/s)	10.60	10.50	10.97	8.23	**11.18**	8.92	10.70	10.43
% communication	0	100	100	90.63	**47.37**	0	0	0
				Pasubio network				
Mean queue length	35.66	35.06	34.29	19.88	**14.74**	47.10	38.33	39.48
Mean wait time (s/veh)	112.49	79.83	82.18	61.9	**60.62**	107.48	104.94	98.48
Mean speed (m/s)	9.76	9.67	9.70	10.39	**10.48**	9.21	9.45	9.98
% communication	0	100	100	93.75	**63.41**	0	0	0

infeasible. To evaluate the effectiveness of communication policy, we compared the communication policy learned by QRC-TSC with (1) random policy, (2) full communication, and (3) no communication. During the evaluation stage, we ran 3 additional independent tests where we manually altered the communication policies. Since this was done during the execution stage, we can be sure that altering the communication policies did not affect the training of QRC-TSC.

Fig. 9 illustrates the performance of the communication policies learned by our framework. We selected a few key metrics (queue length, wait time, and mean speed) from traffic signal control theory to showcase the effectiveness of the learned policies. All metrics were averaged over 100 test episodes and across five runs. The performance of QRC-TSC (in blue) was the best across all metrics in both network scenarios. By choosing which bits to send, the agents were effectively able to balance the performance between no communication and full communication. It is interesting to note that the performance in the Pasubio network with full communication is the worst, which can also be seen in Fig. 7(a) where DIAL performs the worst among all the algorithms. This strongly indicates that constant communication can impede the performance of the system, likely caused due to redundancy in input information (from incoming messages). This leads us to conclude that the agents only need limited information about the action-observation history of the other agent to take optimal actions.

F. HYPERPARAMETERS

We based our framework on the PyMARL library [77] and used the default parameters for all experiments. We experimented with different values for the message size and found that the message of length 5 performed the best. For the additional hyperparameters within the QRC-TSC framework, we conducted a coarse grid search to find the best set of hyperparameters. We set the value of both β_m and β_c to 10^{-5} across all environments. We tried linearly annealing the values of β_m and β_c over 50k iterations, but the overall performance change was negligible. We trained our models on an NVIDIA GeForce RTX 2080 using experience sampled from 8 parallel environments.

VI. CONCLUSION

In this paper, we propose a novel communication mechanism enabling agents to effectively learn (i) *which part of the message* is worth sending (ii) *when* to send a message, and (iii) *to whom* the message should be sent. This can be especially beneficial for problems where there exist constraints on communication (e.g. limited bandwidth). Further, our proposed framework is differentiable which allows for end-to-end training. The advantage of this framework is that the agents can act in a completely decentralized manner but exchange necessary bits of information to maintain coordination between agents. The framework is versatile and could be extended to a large number of applications. We tested our framework, QRC-TSC, on the real-world problem of traffic signal control by building two different traffic signal network scenarios (a synthetic and a real-world network). We compared QRC-TSC with several state-of-the-art frameworks involving communication, and demonstrate that it is able to maintain the least amount of congestion throughout the network while keeping the utility of the communication channel within $\sim 47 - 65$ percent.

Some real-world problems have constraints, for example the cost of communication. Although this study did not address a constrained problem, we believe that QRC-TSC can be extended to include additional parameters, such as cost. One of the drawbacks QRC-TSC is that the maximum length of the message needs to be set a priori. One solution to this problem could be to allow for multiple communication passes. Future work will address how to establish the maximum message length.

APPENDIX

A. ALGORITHM FOR QRC-TSC

Algorithm 1 Training Procedure for QRC-TSC

```
1: Initialize the agent network with parameters $\theta$, the target network with parameters $\theta^-$, replay buffer $\mathcal{D}$ with capacity $N_{\mathcal{D}}$, and batch size $N_B$
2: for each training episode $e$ do
3:     for each episode do
4:         $t = 0$ and $h_0^i = 0,\ hc_0^i = 0$ for each agent $i = \{1, \cdots, n\}$
5:         while $s_t \neq$ terminal and $t < T$ do
6:             $t = t + 1$
7:             Obtain observation $o_t = \{o_t^1, \cdots, o_t^n\}$ and global state $S_t$
8:             Get message vector $\hat{m}_t^i$ and communication action $c_t^{i'}$ from agents $\triangleright\ \hat{m}_t^i, c_t^{i'} = \text{CNet}_i(o_t^i, m_{t-1}^{-i}, hc_{t-1}^i, a_{t-1}^i; \theta_c^i)$
9:             Set outgoing messages as: $m_t^{i'} = \hat{m}_t^i \odot c_t^{i'}$
10:            Select action $a_t^i$ according to $\epsilon$-greedy policy w.r.t. agent $i$'s decentralized action value $Q(o_t^i, m_{t-1}^{-i}, h_{t-1}^i, a_{t-1}^i; \theta^i)$
11:            Execute joint action $a_t = \{a_t^1, \cdots, a_t^n\}$ in the environment
12:            Obtain the global reward $r_{t+1}$, next observation $o_{t+1}^i$ for each agent $i$ and next global state $s_{t+1}$
13:        end while
14:        Store the episode in the buffer $\mathcal{D}$ such that the oldest episode is replaced if $|\mathcal{D}| \geq N_{\mathcal{D}}$
15:    end for
16:    Sample a batch of $N_B$ episodes $\sim$ Uniform($\mathcal{D}$)
17:    Calculate the communication loss $\mathcal{L}_C(\theta_r, \theta_c)$ according to (7) and TD loss $\mathcal{L}_{TD}(\theta)$ as in (9) and set total loss as in (8)
18:    Update $\theta$ by minimizing the total loss $\mathcal{L}(\theta)$
19:    Replace target parameters $\theta^- \leftarrow \theta$ every $K$ episodes
20: end for
```

B. VARIATIONAL BOUND ON MUTUAL INFORMATION

The posterior for the mutual information objective based on information bottleneck [54] can be written as

$$I_{\theta_c}(\pi_j(\cdot|\tau_j); \hat{m}_{ij}|\tau_j) = \int p(\tau_j)\pi_j(\cdot|\tau_j)p(\hat{m}_{ij}|\tau_j, \hat{m}_{(-i)j}) \log \pi_j(\cdot|\tau_j, \hat{m}_{ij}) d\tau_j d\pi_j d\hat{m}_{ij}$$

Next, $q_{\theta_r}(\cdot; \tau_j, \hat{m}_j^{in})$ can be written as variational approximation to $\pi_j(\cdot|\tau_j, \hat{m}_{ij})$ and since $D_{KL}(\pi_j(\cdot; \tau_j, \hat{m}_j^{in}) \| q_{\theta_r}(\cdot; \tau_j, \hat{m}_j^{in}) \geq 0$, we obtain the upper bound for mutual information term.

$$\geq \int p(\tau_j)\pi(\cdot|\tau_j)p(\hat{m}_{ij}|\tau_j, \hat{m}_{(-i)j}) \log q_{\theta_r}(\cdot|\tau_j, \hat{m}_{ij}) d\tau_j d\pi_j d\hat{m}_{ij}$$

We approximate $p(\tau_j, a_j) = p(\tau_j)\pi(\cdot|\tau_j)$ using Monte Carlo sampling.

$$\geq \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}, \hat{m}_j^{in}\sim f_c(\tau;\theta_c)}[\int \pi(\cdot|\tau_j) \log q_{\theta_r}(\cdot|\tau_j, \hat{m}_j^{in}) d\pi_j] + \mathcal{H}(\cdot|\tau_j, \hat{m}_{(-i)j})$$
$$\geq \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}, m_{ij}^{in}, c_{ij}^{in}\sim f_c(\tau;\theta_c)}[-\mathcal{CE}(\pi_j(\cdot; \tau_j, \hat{m}_j^{in}) \| q_{\theta_r}(\cdot; \tau_j, \hat{m}_j^{in})],$$

where entropy term $\mathcal{H}(\cdot|\tau_j, \hat{m}_{(-i)j})$ is independent of optimization.

C. COMMUNICATION LOSS FUNCTION FOR JOINT DISTRIBUTIONS

$$J_c[\hat{m}_i|\tau_j, \hat{m}_{(-i)j}]$$
$$\geq \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}}[-\mathcal{CE}[p(a_j|\boldsymbol{\tau})||q_{\theta_r}(a_j|\boldsymbol{\tau}, \hat{m}_j^{in})]] - E_{m_j^{in}, c_j^{in}\sim f_c(\boldsymbol{\tau};\theta_c)}[D_{KL}(p(\hat{m}_{ij}|\boldsymbol{\tau}_i)||r(\hat{m}_{ij}))]$$
$$\geq \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}}[-\mathcal{CE}[p(a_j|\boldsymbol{\tau})||q_{\theta_r}(a_j|\boldsymbol{\tau}, (m \odot c)_j^{in})]] - E_{m_j^{in}, c_j^{in}\sim f_c(\boldsymbol{\tau};\theta_c)}\left[D_{KL}\left(\log \frac{p(m_{ij}, c_{ij}|\boldsymbol{\tau}_i)}{r(m_{ij}, c_{ij})}\right)\right]$$
$$\geq \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}}[-\mathcal{CE}[p(a_j|\boldsymbol{\tau})||q_{\theta_r}(a_j|\boldsymbol{\tau}, (m \odot c)_j^{in})]] - E_{m_j^{in}, c_j^{in}\sim f_c(\boldsymbol{\tau};\theta_c)}\left[D_{KL}\left(\log \frac{p(m_{ij}|\boldsymbol{\tau}_i)p(c_{ij}|\boldsymbol{\tau}_i)}{r(m_{ij})r(c_{ij})}\right)\right]$$
$$\geq \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}}[-\mathcal{CE}[p(a_j|\boldsymbol{\tau})||q_{\theta_r}(a_j|\boldsymbol{\tau}, (m \odot c)_j^{in})]][t] - E_{m_j^{in}\sim f_c(\boldsymbol{\tau};\theta_c)}\left[D_{KL}\left(\log \frac{p(m_{ij}|\boldsymbol{\tau}_i)}{r(m_{ij})}\right)\right] - E_{c_j^{in}\sim f_c(\boldsymbol{\tau};\theta_c)}\left[D_{KL}\left(\log \frac{p(c_{ij}|\boldsymbol{\tau}_i)}{r(c_{ij})}\right)\right]$$

D. GUMBEL-SIGMOID FOR DISCRETE COMMUNICATION VARIABLE

We consider the communication action c_{ijk} as a Bernoulli random variable. We drop the subscripts ijk for the ease of notation. Let $c \sim Bernoulli(\alpha)$ be the communication action, where $\alpha \in (0, \infty)$ is the location parameter. We can write the Gumbel-softmax [56], [78] function as:

$$c_l = \frac{\exp((\log \alpha_l + g_l)/\lambda)}{\sum_l \exp((\log \alpha_l + g_l)/\lambda)}$$

where $\lambda \in (0, \infty)$ is the temparature parameter and l is the dimension over the softmax vector. And we can rewrite the softmax function for two variables α_l and 0 as:

$$\sigma(\alpha_l) = \frac{\exp((\log \alpha_l + g_l)/\lambda)}{\exp((\log \alpha_l + g_l)/\lambda) + \exp(g_m/\lambda)}$$

$$= \frac{1}{1 + (\exp(g_m/\lambda)/\exp((\alpha_i + g_l)/\lambda))}$$
$$= \frac{1}{1 + \exp(-(\log \alpha_i + g_l - g_m)/\lambda)}$$
$$= sigmoid((\log \alpha_i + g_l - g_m)/\lambda)$$

The difference between two Gumbel distributions $g_l - g_m$ is given as Logistic distribution and can be sampled as $\log U - \log(1 - U)$, where $U \sim Uniform(0, 1)$ [57]. We set the value of λ to 0.67.

REFERENCES

[1] *World Urbanization Prospects: The 2014 Revision*, United Nations Dept. Econ. Social Affairs/Population Division, New York, NY, USA, 2014.

[2] S. Çolak, A. Lima, and M. C. González, "Understanding congested travel in urban areas," *Nature Commun.*, vol. 7, no. 1, p. 10793, Mar. 2016.

[3] *E-Stats 2014: Measuring the Electronic Economy*, U.S. Census Bureau, Suitland-Silver Hill, MD, USA, 2016.

[4] B. Schaller, "The new automobility: Lyft, Uber and the future of American cities," Schaller Consulting, Brooklyn, NY, USA, Tech. Rep. SC-2018-07, 2018. [Online]. Available: http://www.schallerconsult.com/rideservices/automobility.pdf

[5] D. L. Schrank and T. J. Lomax, "TTI's 2012 urban mobility report: Powered by INRIX traffic data," Texas Transp. Inst., Texas A&M Univ., Bryan, TX, USA, 2009.

[6] J. I. Levy, J. J. Buonocore, and K. von Stackelberg, "Evaluation of the public health impacts of traffic congestion: A health risk assessment," *Environ. Health*, vol. 9, no. 1, pp. 1–12, Dec. 2010.

[7] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. S. Awwal, and V. K. Asari, "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, no. 3, p. 292, Mar. 2019.

[8] R. Bokade, A. Navato, R. Ouyang, X. Jin, C.-A. Chou, S. Ostadabbas, and A. V. Mueller, "A cross-disciplinary comparison of multimodal data fusion approaches and applications: Accelerating learning through trans-disciplinary information sharing," *Expert Syst. Appl.*, vol. 165, Mar. 2021, Art. no. 113885.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[10] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis, "Multiagent reinforcement learning for urban traffic control using coordination graphs," in *Machine Learning and Knowledge Discovery in Databases*. Antwerp, Belgium: Springer, Sep. 2008, pp. 656–671.

[11] D. Zhao, Y. Dai, and Z. Zhang, "Computational intelligence in urban traffic signal control: A survey," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 4, pp. 485–494, Jul. 2012.

[12] W. Genders and S. Razavi, "Using a deep reinforcement learning agent for traffic signal control," 2016, *arXiv:1611.01142*.

[13] E. Van der Pol and F. A. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," in *Proc. Learn., Inference Control Multi-Agent Syst. (NIPS)*, vol. 8, 2016, pp. 21–38.

[14] J. A. Calvo and I. Dusparic, "Heterogeneous multi-agent deep reinforcement learning for traffic lights control," in *Proc. AICS*, 2018, pp. 2–13.

[15] X. Liang, X. Du, G. Wang, and Z. Han, "Deep reinforcement learning for traffic light control in vehicular networks," 2018, *arXiv:1803.11115*.

[16] Y. Lin, X. Dai, L. Li, and F.-Y. Wang, "An efficient deep reinforcement learning model for urban traffic control," 2018, *arXiv:1808.01876*.

[17] M. Camelo, M. Claeys, and S. Latré, "Parallel reinforcement learning with minimal communication overhead for IoT environments," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1387–1400, Feb. 2020.

[18] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1086–1095, Mar. 2020.

[19] 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 Trans. Cybern.*, vol. 50, no. 6, pp. 2687–2700, Jun. 2020.

[20] H. Wei, G. Zheng, V. Gayah, and Z. Li, "A survey on traffic signal control methods," 2019, *arXiv:1904.08117*.

[21] 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 *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 1913–1922.

[22] 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 *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 1290–1298.

[23] G. Zheng, Y. Xiong, X. Zang, J. Feng, H. Wei, H. Zhang, Y. Li, K. Xu, and Z. Li, "Learning phase competition for traffic signal control," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 1963–1972.

[24] S. Gupta, R. Hazra, and A. Dukkipati, "Networked multi-agent reinforcement learning with emergent communication," 2020, *arXiv:2004.02780*.

[25] A. Haydari and Y. Yılmaz, "Deep reinforcement learning for intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 11–32, Jan. 2022.

[26] A. Jaleel, M. A. Hassan, T. Mahmood, M. U. Ghani, and A. U. Rehman, "Reducing congestion in an intelligent traffic system with collaborative and adaptive signaling on the edge," *IEEE Access*, vol. 8, pp. 205396–205410, 2020.

[27] J. Ma and F. Wu, "Feudal multi-agent deep reinforcement learning for traffic signal control," in *Proc. 19th Int. Conf. Auto. Agents Multiagent Syst. (AAMAS)*, 2020, pp. 816–824.

[28] T. Tan, T. Chu, and J. Wang, "Multi-agent bootstrapped deep *Q*-network for large-scale traffic signal control," in *Proc. IEEE Conf. Control Technol. Appl. (CCTA)*, Aug. 2020, pp. 358–365.

[29] 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 Trans. Mobile Comput.*, vol. 21, no. 6, pp. 2228–2242, Jun. 2022.

[30] Q. Wu, J. Wu, J. Shen, B. Yong, and Q. Zhou, "An edge based multi-agent auto communication method for traffic light control," *Sensors*, vol. 20, no. 15, p. 4291, Jul. 2020.

[31] D. Xie, Z. Wang, C. Chen, and D. Dong, "IEDQN: Information exchange DQN with a centralized coordinator for traffic signal control," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[32] X. Zang, H. Yao, G. Zheng, N. Xu, K. Xu, and Z. Li, "Metalight: Value-based meta-reinforcement learning for traffic signal control," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 1153–1160.

[33] Y. Zhao, G. Xu, Y. Duy, and M. Fangz, "Learning multi-agent communication with policy fingerprints for adaptive traffic signal control," in *Proc. IEEE 16th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2020, pp. 266–273.

[34] F.-X. Devailly, D. Larocque, and L. Charlin, "IG-RL: Inductive graph reinforcement learning for massive-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 7496–7507, Jul. 2022.

[35] J. Liu, H. Zhang, Z. Fu, and Y. Wang, "Learning scalable multi-agent coordination by spatial differentiation for traffic signal control," *Eng. Appl. Artif. Intell.*, vol. 100, Apr. 2021, Art. no. 104165.

[36] M. Wang, L. Wu, J. Li, and L. He, "Traffic signal control with reinforcement learning based on region-aware cooperative strategy," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6774–6785, Jul. 2022.

[37] L. Zhu, P. Peng, Z. Lu, X. Wang, and Y. Tian, "Variationally and intrinsically motivated reinforcement learning for decentralized traffic signal control," 2021, *arXiv:2101.00746*.

[38] D. V. Pynadath and M. Tambe, "The communicative multiagent team decision problem: Analyzing teamwork theories and models," *J. Artif. Intell. Res.*, vol. 16, pp. 389–423, Jun. 2002.

[39] J. Foerster, I. A. Assael, N. De Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1–9.

[40] S. Sukhbaatar, A. Szlam, and R. Fergus, "Learning multiagent communication with backpropagation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1–9.

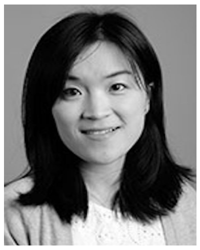
[41] Y. Hoshen, "VAIN: Attentional multi-agent predictive modeling," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[42] E. Pesce and G. Montana, "Improving coordination in small-scale multi-agent deep reinforcement learning through memory-driven communication," *Mach. Learn.*, vol. 109, nos. 9–10, pp. 1727–1747, Sep. 2020.

[43] S. Q. Zhang, Q. Zhang, and J. Lin, "Efficient communication in multi-agent reinforcement learning via variance based control," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–10.

[44] A. Singh, T. Jain, and S. Sukhbaatar, "Learning when to communicate at scale in multiagent cooperative and competitive tasks," 2018, *arXiv:1812.09755*.
[45] T. Wang, J. Wang, C. Zheng, and C. Zhang, "Learning nearly decomposable value functions via communication minimization," 2019, *arXiv:1910.05366*.
[46] D. Kim, S. Moon, D. Hostallero, W. Ju Kang, T. Lee, K. Son, and Y. Yi, "Learning to schedule communication in multi-agent reinforcement learning," 2019, *arXiv:1902.01554*.
[47] Y. Niu, R. R. Paleja, and M. C. Gombolay, "Multi-agent graph-attention communication and teaming," in *Proc. AAMAS*, 2021, pp. 964–973.
[48] A. Agarwal, S. Kumar, and K. Sycara, "Learning transferable cooperative behavior in multi-agent teams," 2019, *arXiv:1906.01202*.
[49] Y. Du, B. Liu, V. Moens, Z. Liu, Z. Ren, J. Wang, X. Chen, and H. Zhang, "Learning correlated communication topology in multi-agent reinforcement learning," in *Proc. 20th Int. Conf. Auto. Agents MultiAgent Syst.*, 2021, pp. 456–464.
[50] J. Jiang, C. Dun, T. Huang, and Z. Lu, "Graph convolutional reinforcement learning," 2018, *arXiv:1810.09202*.
[51] Y. Liu, W. Wang, Y. Hu, J. Hao, X. Chen, and Y. Gao, "Multi-agent game abstraction via graph attention neural network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 7211–7218.
[52] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.
[53] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2008.
[54] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," 2016, *arXiv:1612.00410*.
[55] 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," *J. Mach. Learn. Res.*, vol. 21, no. 1, pp. 7234–7284, 2020.
[56] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with Gumbel–Softmax," 2016, *arXiv:1611.01144*.
[57] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," 2016, *arXiv:1611.00712*.
[58] E. Dupont, "Learning disentangled joint continuous and discrete representations," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–11.
[59] R. Lowe, J. Foerster, Y.-L. Boureau, J. Pineau, and Y. Dauphin, "On the pitfalls of measuring emergent communication," 2019, *arXiv:1903.05168*.
[60] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of sumo-simulation of urban mobility," *Int. J. Adv. Syst. Meas.*, vol. 5, nos. 3–4, p. 164, 2012.
[61] N. Jaques, A. Lazaridou, E. Hughes, C. Gulcehre, P. Ortega, D. Strouse, J. Z. Leibo, and N. De Freitas, "Social influence as intrinsic motivation for multi-agent deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3040–3049.
[62] M. A. Wiering, "Multi-agent reinforcement learning for traffic light control," in *Proc. 17th Int. Conf. Mach. Learn. (ICML)*, 2000, pp. 1151–1158.
[63] M. Wiering, J. van Veenen, J. Vreeken, and A. Koopman, "Intelligent traffic light control," Inst. Inf. Comput. Sci., Utrecht Univ., Utrecht, The Netherlands, Tech. Rep. UU-CS-2004-029, 2004.
[64] X. Wang, L. Ke, Z. Qiao, and X. Chai, "Large-scale traffic signal control using a novel multiagent reinforcement learning," *IEEE Trans. Cybern.*, vol. 51, no. 1, pp. 174–187, Jan. 2021.
[65] S. El-Tantawy, B. Abdulhai, and 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.*, vol. 14, no. 3, pp. 1140–1150, Sep. 2013.
[66] T. Chu, S. Chinchali, and S. Katti, "Multi-agent reinforcement learning for networked system control," 2020, *arXiv:2004.01339*.
[67] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double *Q*-learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, 2016, pp. 1–7.
[68] M. Hausknecht and P. Stone, "Deep recurrent *Q*-learning for partially observable MDPS," in *Proc. AAAI Fall Symp. Ser.*, 2015, pp. 1–9.
[69] F. A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*. Cham, Switzerland: Springer, 2016, doi: 10.1007/978-3-319-28929-8.
[70] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proc. 10th Int. Conf. Mach. Learn.*, 1993, pp. 330–337.
[71] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel, "Value-decomposition networks for cooperative multi-agent learning," 2017, *arXiv:1706.05296*.
[72] L. Bieker, D. Krajzewicz, A. Morra, C. Michelacci, and F. Cartolano, "Traffic simulation for all: A real world traffic scenario from the city of Bologna," in *Modeling Mobility With Open Data*. Berlin, Germany: Springer, 2015, pp. 47–60.
[73] R. P. Roess, E. S. Prassas, and W. R. McShane, *Traffic Engineering*. London, U.K.: Pearson, 2004.
[74] C. Gershenson, "Self-organizing traffic lights," 2004, *arXiv:nlin/0411066*.
[75] P. Varaiya, "Max pressure control of a network of signalized intersections," *Transp. Res. C, Emerg. Technol.*, vol. 36, pp. 177–195, Nov. 2013.
[76] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.
[77] M. Samvelyan, T. Rashid, C. S. de Witt, G. Farquhar, N. Nardelli, T. G. J. Rudner, C.-M. Hung, P. H. S. Torr, J. Foerster, and S. Whiteson, "The StarCraft multi-agent challenge," 2019, *arXiv:1902.04043*.
[78] E. J. Gumbel and J. Lieblein, "Some applications of extreme-value methods," *Amer. Statistician*, vol. 8, no. 5, pp. 14–17, Dec. 1954.

ROHIT BOKADE received the master's degree in operations research from Northeastern University, where he is currently pursuing the Ph.D. degree in industrial engineering. His current research interests include exploring the potential of advanced machine learning techniques, such as reinforcement learning, deep learning, and optimization techniques to improve industrial engineering practices, and solve real-world problems.



XIAONING JIN (Member, IEEE) received the Ph.D. degree in industrial and systems engineering from the University of Michigan, Ann Arbor, MI, USA, in 2012. She is currently an Assistant Professor in mechanical and industrial engineering with the College of Engineering, Northeastern University, Boston, USA. She has over 50 papers in fully refereed international journals and conferences. Her research interests include predictive analytics and decision making, data analytics, fault diagnostics and prognostics, and artificial intelligence in various engineering applications. She was a recipient of the National Science Foundation Career Award, in 2020. She received the 2016 Outstanding Young Manufacturing Engineer Award from the Society of Manufacturing Engineers (SME). She also serves as the Vice-Chair for the Manufacturing Systems Technical Committee with the ASME Manufacturing Science and Engineering Division.

CHRISTOPHER AMATO is currently an Assistant Professor with Northeastern University, where he leads the Laboratory for Learning and Planning in Robotics. Before joining Northeastern University, he was a Research Scientist with Aptima Inc., and a Postdoctoral Researcher and a Research Scientist with MIT and an Assistant Professor with the University of New Hampshire. He has published many papers in leading artificial intelligence, machine learning, and robotics conferences (including winning a Best Paper Prize at AAMAS-14 and being nominated for the Best Paper Prize at RSS-15, AAAI-19, AAMAS-21, and MRS-21). He has also won several awards, such as Amazon Research Awards and an NSF CAREER Award. His research interests include reinforcement learning and planning in partially observable and multi-agent/multi-robot systems.