Transportation Research Part C 98 (2019) 358–369

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Cordon control with spatially-varying metering rates: A Reinforcement Learning approach

Wei Ni, Michael J. Cassidy*

Department of Civil and Environmental Engineering, University of California, Berkeley, United States

A B S T R A C T

The work explores how Reinforcement Learning can be used to re-time traffic signals around cordoned neighborhoods. An RL-based controller is developed by representing traffic states as graph-structured data and customizing corresponding neural network architectures to handle those data. The customizations enable the controller to: (i) model neighborhood-wide traffic based on directed-graph representations; (ii) use the representations to identify patterns in real-time traffic measurements; and (iii) capture those patterns to a spatial representation needed for selecting optimal cordon-metering rates. Input to the selection process also includes a total inflow to be admitted through a cordon. The rate is optimized in a separate process that is not part of the present work. Our RL-controller distributes that separately-optimized rate across the signalized street links that feed traffic through the cordon. The resulting metering rates vary from one feeder link to the next. The selection process can reoccur at short time intervals in response to changing traffic patterns. Once trained on a few cordons, the RL-controller can be deployed on cordons elsewhere in a city without additional training.

This portability feature is confirmed via simulations of traffic on an idealized street network. The tests also indicate that the controller can reduce the network's vehicle hours traveled well beyond what can be achieved via spatially-uniform cordon metering. The extra reductions in VHT are found to grow larger when traffic exhibits greater in-homogeneities over the network.

1. Introduction

Metering vehicle inflows to cordoned neighborhoods is a promising means of combatting city-street traffic congestion. The task entails re-timing the traffic signals that reside along a cordon and feed traffic to the neighborhood inside (Daganzo, 2007; Geroliminis et al., 2013; Aboudolas and Geroliminis, 2013; Ramezani et al., 2015; Haddad, 2017; Keyvan-Ekbatani et al., 2012, 2015a,b, 2017; Kouvelas et al., 2017). This form of control can reduce the neighborhood's congestion, but can cause considerable queueing on the outlying feeder links, and on other links upstream. In essence, the objective is to determine the traffic-signal metering rates that balance these effects and reduce vehicle hours traveled (VHT) in, and around the neighborhood.

Efforts of this kind have often been pursued using Macroscopic Fundamental Diagrams (MFDs); e.g. see Daganzo (2007), Geroliminis and Daganzo (2008), Daganzo and Geroliminis (2008), Daganzo et al. (2011), Ji et al. (2010), Gayah and Dixit (2013), Du et al. (2016) and Knoop et al. (2013). These provide simple, but physically-realistic descriptions of neighborhood traffic; and enable one to model neighborhood streets as simple, a-spatial queueing systems. In certain cases, MFDs have been combined with flow conservation laws to produce what we term Network Transmission Models (Daganzo, 2007; Geroliminis et al., 2013; Aboudolas and Geroliminis, 2013; Ramezani et al., 2015; Haddad, 2017; Keyvan-Ekbatani et al., 2012, 2015a,b, 2017; Kouvelas et al., 2017). These NTMs forecast a neighborhood's future traffic states over short time horizons. The forecasts are used in combination with PID-, state feedback or model predictive control to generate a cordon's optimal allowable inflows at specified time steps. In most works, each optimized total metering rate is uniformly distributed across the cordon. Each time step is thus characterized by a common

* Corresponding author.
E-mail addresses: weini@berkeley.edu (W. Ni), cassidy@ce.berkeley.edu (M.J. Cassidy).

https://doi.org/10.1016/j.trc.2018.12.007
Received 10 September 2018; Received in revised form 11 December 2018; Accepted 13 December 2018
Available online 21 December 2018
0968-090X/

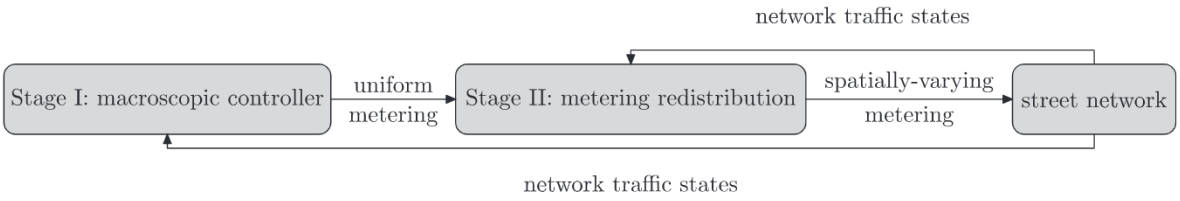

Fig. 1. Two-stage process for cordon control.

metering rate on every feeder link.

Yet when traffic conditions are not homogeneous over a street network, further reduction in VHT might come by varying the metering rates from one feeder link to the next. Feeders with high demands, for example, might be metered less restrictively, to shorten their queues. To compensate for the higher flows that pour into the neighborhood via those links, metering might be made more restrictive on low-demand feeders. Restrictive metering could also reduce the occurrence of queues fully dissipating, and thus wasting green times on those low-demand links. It might even make sense to spatially-vary metering rates in response to traffic inhomogeneities across non-feeder links. These could include physically-remote links both inside and outside a cordon.

Difficulties can arise in the implementation, however. For example, spatially-varying metering rates might induce driver route-choice decisions that are difficult to anticipate. Wasted green times may be similarly difficult to forecast. And impacts of cordon metering on physically remote links would seem especially hard to predict.

Despite the challenges in modeling these sorts of cause-and-effect relations, several analytical-based methods for designing cordon-control systems have been proposed. These are described below.

1.1. Review of earlier methods

The literature includes three particularly relevant efforts; see Ramezani et al. (2015), Keyvan-Ekbatani et al. (2016) and Jusoh and Ampountolas (2017). In each of those works, spatially-varying metering plans were generated and updated over time as per the two-stage hybrid framework diagrammed in Fig. 1. Allowable cordon-wide inflows were optimized in Stage I. This was done in Keyvan-Ekbatani et al. (2016) using a PID controller in combination with a Neighborhood Transmission Model. The works in Ramezani et al. (2015) and Jusoh and Ampountolas (2017) combined the use of NTMs with model predictive control instead.

The process of spatially distributing the allowable inflows occurred in Stage II of the framework. The methods used in that second stage were distinct for each of the three above-cited works. Each sought, moreover, to reduce street-network VHT by targeting distinct proxy measures.

The Stage II process was pursued in Keyvan-Ekbatani et al. (2016) by formulating the task as a continuous quadratic knapsack problem. The objective was to balance the lengths of the feeder-link queues that formed due to the metering. Simulations showed that feeder queues under this balancing scheme spilled-back less often onto upstream links. Network-wide VHT reportedly diminished as a result. In Jusoh and Ampountolas (2017), that same balancing of queue lengths was combined with the minimization of trip-completion rates on the network. Simulations in that work again showed reductions in VHT as a result.

The work in Ramezani et al. (2015) is notable in that it sought to balance not the lengths of feeder-link queues, but rather the vehicle loading across all links inside cordoned neighborhoods. It did so by partitioning the neighborhood into multiple sub-regions, with boundaries that remained fixed over time. Metering rates were varied along the encircling cordon so as to balance the traffic densities across all sub-regions.

Simulated outcomes in Ramezani et al. (2015) were favorable, and the work deserves credit for the unique attention given to traffic states on (non-feeder) links residing inside a network. The work raises certain concerns, however. These are described below.

First, the method assumes that a well-defined MFD exists for each sub-region. This may not be the case when sub-regions are small, or when larger-sized sub-regions are not each homogeneously-loaded with traffic; see (Daganzo, 2007).[1] Second, the quantity of input needed in Ramezani et al. (2015) expands to include origin–destination demands disaggregated by sub-region. Those demands can be difficult to obtain, particularly if they change with time. Finally, a driver route-choice model is also required, which may be complex and not fully tested.

1.2. Present approach

Like its predecessors described above, the present work advocates a two-stage, hybrid framework for spatially distributing cordon-metering rates; see again Fig. 1. The present focus is solely on Stage II, however. Allowable cordon inflows generated in a Stage I process would be taken as inputs. These could come from previously-cited sources such as Geroliminis et al. (2013), Haddad (2017) and Ni and Cassidy (2018), which combine Network Transmission Models either with PID-, state feedback or model predictive control.[2] Attention now turns to our Stage-II approach for distributing the allowable inflows in spatially-varying fashion.

[1] These concerns could make the task of delimiting sub-regions difficult in its own right, especially for street networks with irregular geometries.

[2] The method in Ni and Cassidy (2018) will be used for simulation experiments in Section 3, and our reasons for selecting this method will be explained in that section as well.

Unlike earlier approaches, the present one features Reinforcement Learning (RL). RL has been applied on a few traffic control problems like (Zhu and Ukkusuri, 2014; Balakrishna et al., 2010; Cai et al., 2009), but never in the domain of cordon metering. Our work produced an RL-based cordon controller that: (i) automatically identifies patterns in real-time measurements of neighborhood traffic; and (ii) learns through trail-and-error iteration how to spatially distribute the allowable cordon inflows generated in Stage I to obtain the greatest reward. The reward in this case is a proxy measure of VHT, much as in the previous works cited in Section 1.1. To obtain that reward, the RL-controller can spatially-vary metering rates in response to traffic patterns on feeder- and non-feeder links alike, both near and far from a cordon line.[3] It does so despite our limited understanding of relevant cause-and-effect relations. It does so, moreover, without partitioning neighborhoods into sub-regions; without requiring (human) operators to obtain origin–destination data; and without need for route-choice models. The controller is portable to boot: once trained on a few cordons, it can be deployed on other cordons without additional training.

The controller itself is represented by neural networks. Part of the present innovation lies in representing street networks as directed graphs. These representations enable the bundling of (directed!) traffic flows together with street topologies in well-structured fashion. Once trained on data, the controller automatically recognizes patterns on the graph representations through the process of graph convolution (Bruna et al., 2013; Defferrard et al., 2016; Henaff et al., 2015; Levie et al., 2017). Enabling this process required customizations to neural network architectures. Lastly, the controller agglomerates recognized patterns through the process of average pooling (Dhillon et al., 2007). This process generates new directed graphs with fewer vertices, which reduces the size of the inputs needed for decision-making. The convolution and pooling processes produce simpler, more generalizable decision rules, which give the controller its portability.

1.3. Road map

The methodological contributions noted above are detailed in the following section. Payoffs are explored in Section 3 by simulating traffic on an idealized street network. Practical implications and the likelihood of real-world deployments are discussed in Section 4.

Before proceeding further, all symbols used in the present work are defined in Table 1. Each symbol will be defined again when it first appears in the discussion. The redundancy of Table 1 is offered for the convenience of the reader, and in deference to a suggestion by a reviewer.

2. Methods

Reinforcement Learning entails the training of a software-controlled agent to interact with its environment in the manner shown in Fig. 2. At each time step, t, the agent takes control action, $a^{(t)}$, in response to the current system state, $s^{(t)}$. The environment transitions to state $s^{(t+1)}$ and a reward, $r^{(t)}$, is fed-back to the agent. The objective at each t is to maximize the reward that accumulates over an infinite horizon, starting from $t = 0$. This accumulated reward is kept to a finite quantity, and temporally-proximate rewards are more heavily weighted than are distant ones, by means of discounting, such that $\sum_{i\geqslant 0} \gamma^i \cdot r^{(t+i)}$, where $0 < \gamma < 1$ is the discounting factor.

For the present work, reward $r^{(t)}$ was set to be the metered flow that actually passes through the cordon during time step t. That inflow can be lower than the optimal allowable rate generated in Stage I. Differences can occur due to: wasted green times on low-demand feeders; and vehicle blocking from queued links inside or outside a cordon. The point is that maximizing this reward reduces street-network VHT. As a bonus, this chosen reward is directly affected by the metering rates selected by the controller, which simplifies its task of learning the relation between $r^{(t)}$ and $(s^{(t)}, a^{(t)})$.

The reward is achieved via a control policy, π. It is a function that maps system state to control action; i.e. $a^{(t)} = \pi(s^{(t)})$. There is a corresponding Q-function for π. Its value represents the accumulated discounted reward if action $a^{(t)}$ is taken for $s^{(t)}$ and π is followed from $t + 1$ onwards; see Sutton and Barto (1998). This second function is defined as

$$Q^{\pi}(s^t, a^t) = \mathbb{E}\left[r^t + \gamma Q^{\pi}(s^{t+1}, \pi(s^{t+1}))|s^t, a^t\right]. \tag{2.1}$$

An optimal policy, $\pi^{\star}$, is sought, such that $\mathbb{E}\left[Q^{\pi}(s^t, \pi(s^t))\right]$ is maximized.

A directed-graph representation that was customized to transfer data to the RL-controller is described in Section 2.1. The control problem is formulated as an RL task atop that representation in Section 2.2. A customized algorithm that finds $\pi^{\star}$ by modeling network traffic based on the graph representation is presented in Section 2.3.

2.1. Graph representation

The directed-graph representation described below provides means of inputting data to our controller. Data to be input can include both: the static features of a street network's geometry; and the dynamic features of its traffic conditions and directional movements. The representation can thus describe street networks in comprehensive fashion. It is to our knowledge the only means of transferring these particular kinds of data in ways that can be handled by neural networks.

[3] The RL-controller can engage all signals along a cordon in the metering effort, or can decide (from the data) to allow some of those signals to operate in normal, non-metering fashion.

Table 1
Symbols.

Symbol	Meaning
$s^{(t)}$	System state at time step t
$a^{(t)}$	Control action at t
$r^{(t)}$	Reward at t
γ	Discounting factor
$\pi(s)$	Control policy, or "actor" function
$Q(s, a)$	Q function, or "critic" function
$G = (E, V)$	Graph representing street network, with vertex set V, and edge set E
u	A particular vertex (or road link)
U	A set of vertexes (or road links)
W	Adjacent matrix of a graph
L	Laplacian matrix of a graph
Φ	Matrix of the eigen-vectors of a laplacian matrix
Λ	Diagonal matrix of eigen-values of a laplacian matrix
g_θ	A parametrized spectral filter
$n_u^{(t)}$	Vehicle accumulation on vertex u at t
$n^{(t)}$	Accumulations on each vertex in V at t
$d_u^{(t)}$	Indicator of whether vertex u is located on the cordon at t
$d^{(t)}$	Indicator of whether each vertex in V is located on the cordon at t
p_u	Static characteristics of vertex u, including length, capacity, etc
p	Static characteristics of each vertex in V, including length, capacity, etc
c_u	Capacity of vertex u
c	Capacity of each vertex in V
$f_u^{(t)}$	Metering rate on vertex u at step t under a spatially-uniform policy
$f^{(t)}$	Metering rate on each vertex in V at step t under a spatially-uniform policy
θ^Q	Parameters of the "critic" function
θ^π	Parameters of the "actor" function
F_s	Dimension of system state
$F_{(s,a)}$	Dimension of system state and control action
τ	Synchronizing parameter for DDPG algorithm

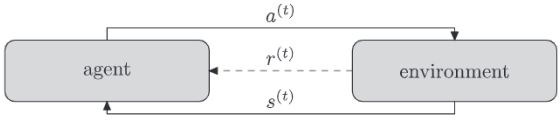

Fig. 2. Reinforcement learning.

The notation for graph representation, G, will be $G = (V, E)$, where: vertex, V, is the collection of all directed street links on a network of interest; and an edge from vertex u to vertex v is assigned to set E if and only if traffic flows directly from u to v.

As per this notation, street links are represented as vertices. This has been done in certain previous works as well; see Saeedmanesh and Geroliminis (2016) and Ji and Geroliminis (2012). Data to be stored in each vertex can include a link's physical length, number of lanes, speed limit and capacity. Each vertex can also store dynamic data concerning time-varying traffic conditions. These can include the link's vehicle accumulation and average vehicle speed at every t.

Distinct traffic movements (through and turning) are represented as edges of a directed graph, and this is more of a novelty.[4] The weight of each edge, $e \in E$, is determined by the time-varying percentages of vertex u's traffic that moves onto v. These weights are often called turning ratios, and are indicators of connectivity strength.

An example of our customized representation is shown for a street intersection in Fig. 3. The left-hand side of the figure shows the intersection with its traffic movements labeled *i-viii*. The corresponding graph-representation is shown to the right.

2.2. Problem formulation

Define the set of street links subjected to cordon control as $U \in V$; i.e. U is the set of feeder links that reside along a cordon. Denote as $n^{(t)}$ the measured vehicle accumulation on a vertex at t, such that $n_u^{(t)}$ is the accumulation on vertex u. Denote $d_u^{(t)}$ as the indicator as to whether u belongs to U; i.e. $\forall\ u \in U$, $d_u^{(t)} = 1$; $\forall\ u \notin U$, $d_u^{(t)} = 0$. Denote as p_u the static attributes of u, including its

[4] Using edges to represent directed traffic movements (and not links or their physical connections) is a departure from custom (Braess et al., 2005; Castillo et al., 2008; Ji and Geroliminis, 2012; Saeedmanesh and Geroliminis, 2016). Though we did not include U-turns in our network, the proposed graph representation can handle U-turns as well.

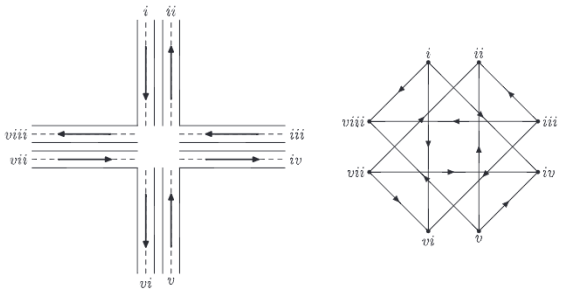

Fig. 3. Directed graph representation of an intersection.

capacity, c_u. Finally, denote as $f^{(t)}$ the time-varying metered link flows under a baseline policy that assigns a single metering rate (determined from Stage I) to all feeder links along a cordon, such that $f_u^{(t)}$ is the metered flow on u. For $u \notin U, f_u^{(t)} = c_u$, since only vertices in U are metered.

Formulating the control problem as an RL task now requires only a straightforward re-definition of the $(s^{(t)}, a^{(t)}, r^{(t)})$. To that end, the system state is defined as $s^{(t)} = (n^{(t)}, f^{(t)}, d^{(t)}, p)$, which stacks all the arguments. Control action, $a^{(t)}$, becomes a redistribution of $f^{(t)}$ for those vertices that are metered, while the total rate allowed through the cordon is unchanged; i.e. $\sum_{u\in U} a_u^{(t)} = \sum_{u\in U} f_u^{(t)}$. Note again that $a_u^{(t)} = c_u, \ \forall\ u \notin U$.

2.3. Optimization algorithm

The optimal control policy, $\pi^{\star}$, and corresponding $Q_{\pi^{\star}}$, were obtained by customizing the actor-critic method (Konda and Tsitsiklis, 2000); where the terms "actor" and "critic" are aliases for the π and Q functions, respectively. The idea is to maintain parameterized actors and critics, $\pi(s|\theta_\pi)$ and $Q(s, a|\theta_Q)$; and to train them in alternating fashion, first by updating θ_Q to satisfy (2.1), and then by updating θ_π with a policy gradient defined by $\mathbb{E}[\nabla_a Q(s, a|\theta^Q)|_{a=\pi(s)} \cdot \nabla_{\theta_\pi} \pi(s|\theta^\pi)]$.

As per recent custom, parameterization was performed in the present work using neural networks (Lillicrap et al., 2015; Mnih et al., 2013, 2015), and this entailed the use of graph convolution and average pooling processes. The former process searches for patterns in data. Searches occur at local levels, meaning that patterns are sought across a graph's neighboring vertices. Average pooling thereafter agglomerates those patterns into a smaller number of composite vertices.

The neural networks were designed in the present work to repeat the convolution and pooling processes three times for each decision generated. In the context of the present work, the reader might envision that convolution and pooling occur first at the layer of neighboring street links; then at neighboring square blocks; and finally across adjacent neighborhoods. This set-up enables the RL-controller to accommodate a street network with hundreds of links.

2.3.1. Convolution on directed graphs

Graph convolution, as originally developed, searches for patterns in data stored in undirected graphs. Customizations were therefore developed to accommodate our directed-graph representations. These customizations occurred at the graph's spectral domain, meaning as matrix representations that are friendly to the computer.

As a starting point, denote as W the adjacent matrix to a directed graph $G = (V, E)$, where W is not symmetric. Define D as a diagonal matrix, where $D_{ii} = \sum_j W_{ij}$. The normalized laplacian matrix is $L = I - D^{-1}W$, where I is the unit matrix. In the case of an undirected graph, the eigen-decomposition of L is $\Phi\Lambda\Phi^T$, where: Λ is a diagonal matrix with diagonal elements that are the real-valued eigenvalues of L; and Φ represents real-valued eigenvectors. Since in the present case the asymmetry of W also makes L asymmetric, the decomposition of L is instead $\Phi\Lambda\Phi^{-1}$, and Λ and Φ are both complex- rather than real-valued. Our customized convolution proceeds as follows.

Start with an attribute z on G's vertexes, and define a polynomial parameterized filter of order K as $g_\theta(\Lambda) = \sum_{k=0}^{K} \theta_k \Lambda^k$, where the set of θ_k are parameters of the filter (Defferrard et al., 2016). The spectral filtering of z using g_θ is the output attribute $\hat{z}$, defined as

$$\hat{z} = \Phi g_\theta(\Lambda)\Phi^{-1} z. \tag{2.2}$$

Eq. (2.2) is also referred to as the application of a convolution operation on attribute z with filter g_θ. If the input, z, and the output, $\hat{z}$, have M and N dimensions respectively, then

$$\forall\ 1 \leqslant n \leqslant N, \hat{z}_n = \sum_m \Phi g_{\theta_{n,m}}(\Lambda)\Phi^{-1} z_m, \tag{2.3}$$

where $g_{\theta_{n,m}}(\Lambda) = \sum_{k=0}^{K} \theta_{n,m,k}\Lambda^k$ with a total number of $M \times N \times K$ parameters. In the terminology of neural networks, the operation of (2.3) is called a graph convolution layer of shape (M, N, K).

2.4. Building the neural networks

The processes of parameterizing the Q-function (the critic) and the π-function (the actor) are separately illustrated in Fig. 4.

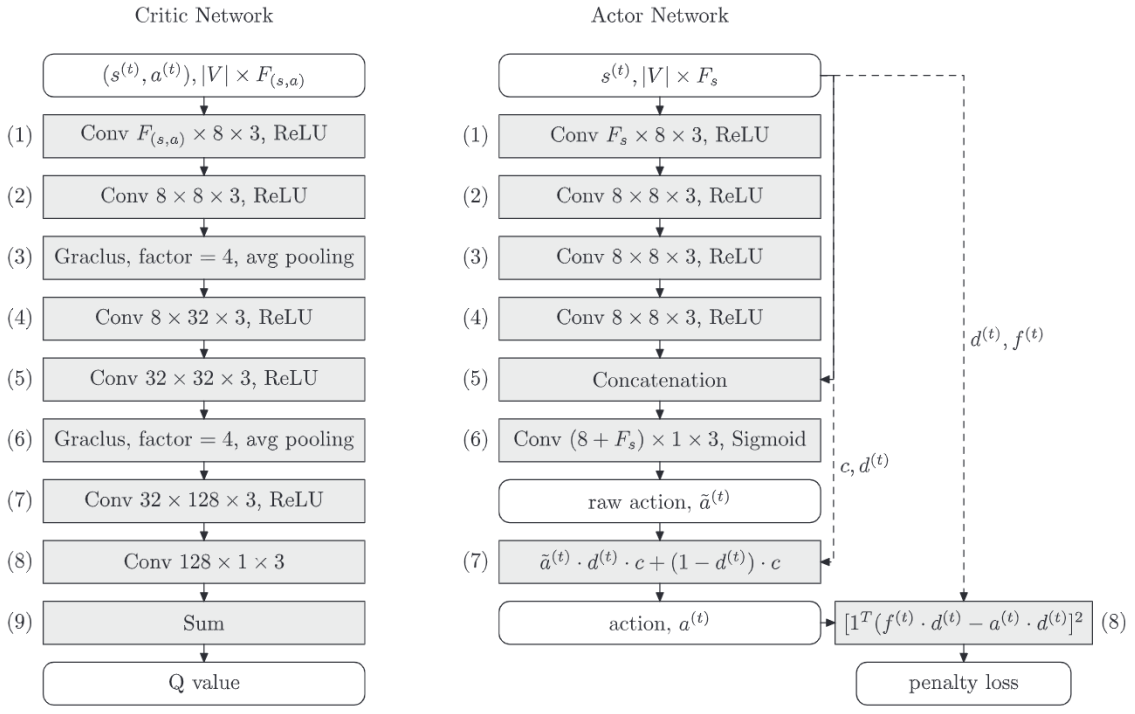

Fig. 4. Critic and actor networks.

Unshaded boxes represent data, including input, output and intermediate variables; and shaded boxes represent operational layers. The layers of each network are numbered to clarify the discussion below.

Note from the top-left box that input to the critic is $(s^{(t)}, a^{(t)})$ and is of dimension $|V| \times F_{(s,a)}$, where $|V|$ is the graph's number of vertices and $F_{(s,a)}$ is the dimension of (s, a) on each vertex. Layers 1–3 form a pipeline consisting of two convolution and one average pooling layer.[5] The convolution in layer 1: is of shape $F_{(s,a)} \times 8 \times 3$; transforms the input; and passes it to layer 2. Layer 2 does convolution again. The third layer pools each group of four neighboring vertices on a directed graph into a single, consolidated vertex with attributes that are the average values of the original four.

Note from the figure that layers 4–6 of the critic network repeat the 2-convolution-1-pooling sequence. This is followed by two additional convolution layers (7 and 8), and a summation operation that consolidates all of the attributes of a graph's remaining vertices into a single Q-value (layer 9).

Turning now to the actor network, note the input as annotated in the top-right box in Fig. 4. It is piped to four consecutive convolution layers (1–4). Layer 5 is a concatenation that pastes the output from layer 4, together with the actor's original input. The concatenation is piped to a convolution operation in layer 6. Its output is a preliminary control action (i.e. a link-specific metering rate) still in need of fine-tuning. The tuning occurs in layer 7 via the customized operation annotated in its shaded box. The final control action, $a^{(t)}$, results.

Recall from Section 2.2 that the $a^{(t)}$ must satisfy three constraints: (i) the inflow from a metered feeder link cannot exceed the link's capacity, $0 \leqslant a_u^{(t)} \leqslant c_u$, $\forall\, u \in U$; (ii) only feeder links to a cordoned neighborhood are controlled, $a_u^{(t)} = c_u$, $\forall\, u \notin U$; and (iii) the link-specific metered inflows must collectively equal the single allowable rate generated by the model-based control (Stage I) of the two-stage framework, i.e. $\sum_{u \in U} a_u^{(t)} = \sum_{u \in U} f_u^{(t)}$. The customized operation in layer 7 guarantees that the output satisfies (i) and (ii). Constraint (iii) is enforced by minimizing the penalty loss introduced by the customized operation annotated in layer 8 of the actor network.

The connection and arrangement of operations, and the parameters for each layer, were determined for the critic and actor networks by trying different combinations, and selecting the ones that generated the best control in simulated tests. These best outcomes (i.e. outcomes generated by the neural networks in Fig. 4) are presented in the following section.

Before doing so, we conclude this section by noting that the critic and actor networks were trained via a customized version of the Deep Deterministic Policy Gradient (DDPG) method proposed in Lillicrap et al. (2015). Details on that customization are given in the appendix.

[5] Each convolution operation features an activation function that applies a nonlinear transformation to the output. The ReLU (Nair et al., 2010) and Sigmoid function were used in the present work, as annotated in Fig. 4. Average pooling was performed using the Graculus algorithm (Dhillon et al., 2007), also as annotated in the figure.

3. Experiments

The RL-controller's performance was tested against that of spatially-uniform control. Tests were conducted for idealized settings, which served two purposes. First, the simplicity of these settings enabled better understanding of when and why spatially-varying control can outperform uniform metering. (Our understanding of these matters turned out to be limited even with our idealizations, as will be evident in Section 3.4.) Second, the idealizations limited the spatial in-homogeneities that occurred in traffic, and thus the advantages of spatially-varying control.[6] This gave the experiments a conservative quality.

The analyses were made further conservative by generating spatially-uniform metering rates using the methods in Ni and Cassidy (2018). The Network Transmission Model in that reference was developed to remedy shortcomings of its predecessors, and was shown to produce more effective cordon-control actions in idealized settings, like the ones used in the present experiments. The RL-controller was found to improve performance (i.e. reduce VHT) over and above what was achevied by spatially-uniform metering rates produced as per Ni and Cassidy (2018). Evidence will be shown momentarily.

3.1. Set-up

Our tests used the AIMSUN simulation platform (Castillo et al., 2008). It powers its own software for simulating traffic in microscopic fashion, and interacted with the DDPG algorithm described in the appendix.

The test site is shown in Fig. 5(a). It consisted of 15 N-S and 15 E-W streets laid-out in a perfect square grid.[7] Each link was 200 m long with 2 lanes for serving traffic in each direction. Capacity (i.e. queue discharge flow) for each lane was set at 1800 vehicles per hour of green time.

All street intersections were controlled by pre-timed traffic signals with 60-s cycle lengths. Random off-sets were used, just as in Daganzo et al. (2017), to denote the absence of signal progression. When not functioning as a meter, a signal operated with two phases and equal green splits of 28s. (Lost times were 4s per cycle.) Metering was enabled only during the 28s allocated to that inflowing movement each cycle.

Trip origins were uniformly distributed over the entire 15×15 network at the time-varying rates shown in Fig. 5(b). Simulations started with an empty street network, and demand fell to zero at $t = 1.8$ h.

Under these conditions, most or all vehicles could be served within each 3-h period used for simulations. Trip destinations were uniformly distributed, but solely within a cordoned neighborhood, like neighborhood B in Fig. 5(a). This O-D pattern was selected to roughly emulate a morning rush in a mono-centric city. The set-up created severe congestion, which enabled stress-testing of the RL-controller.

Input for the controller came from two (simulated) sources: loop detectors placed in each lane on all feeder links; and onboard information systems placed in 10% of the vehicles served in each simulation. The first source (detectors) measured vehicle inflows to a cordoned neighborhood; e.g. flows from neighborhood A to B in Fig. 5(a). Part of the present experiments entailed moving the cordon to different regions within the larger 15×15 neighborhood (see Section 3.3), and the loop detectors were moved along with the cordon lines. The 10% of the vehicles that comprised the second source for input data served as probes. They were instrumented as if they were connected vehicles. Their locations on the network were determined at every 60-s time step, and were used to estimate the time-varying vehicle accumulations on each link.

3.2. Training

The training process consisted of repeatedly simulating the above conditions over successive 3-h windows, which we refer to as epochs. A distinct random seed was used in each epoch, so as to emulate randomness in trip scheduling, route-choice behavior and the like. A discount factor of $\gamma = 0.96$ was selected for all cases.[8]

Outcomes are shown in Fig. 6. It presents the controller's performance, as measured by the resulting network-wide VHT vs the number of epochs used in the training. Notice the diminishing trend in VHT until around 60 epochs. The training process was therefore a time-consuming one. Yet a city might view this as a one-time cost, as per analysis and discussion to come.

3.3. Performance

Tests of the trained controller entailed 3-h simulations under three cordon-control strategies: (i) a do-nothing strategy in which cordon metering did not occur; (ii) spatially-uniform metering rates deployed around a cordon as determined from stage I of our framework; and (iii) the spatially-varying redistribution of those rates via the RL-controller.

[6] Spatial in-homogeneities still occurred: near the corners of a cordoned neighborhood, since the links there fed traffic from two (perpendicular) directions; and because of the stochastic features of AIMSUN's simulations. Further, but still modest in-homogeneities occurred when a cordoned neighborhood was moved to other areas within the street network used for our tests; see Section 3.3.

[7] Use of a square grid was convenient, because much of the input needed for simulating traffic on this idealized geometry can be coded automatically in AIMSUN. The RL-controller can, however, be applied to almost any street-network geometry, including irregular ones.

[8] The DDPG's hyper parameter, τ, was set at 0.02; see the algorithm in the appendix.

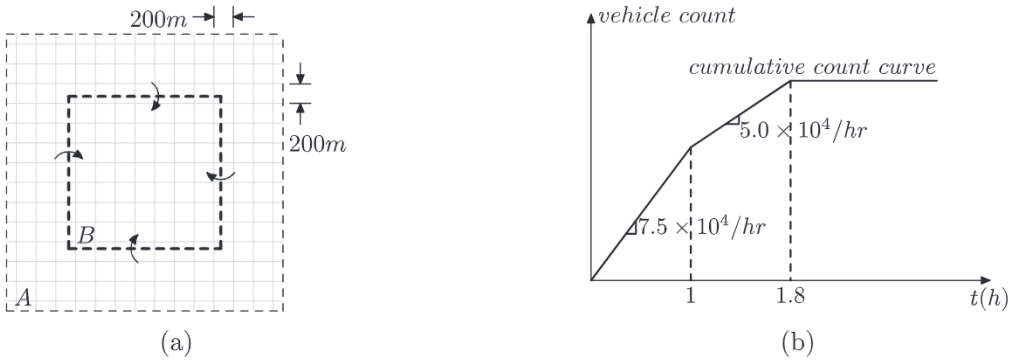

Fig. 5. Simulated urban road network and demand curve.

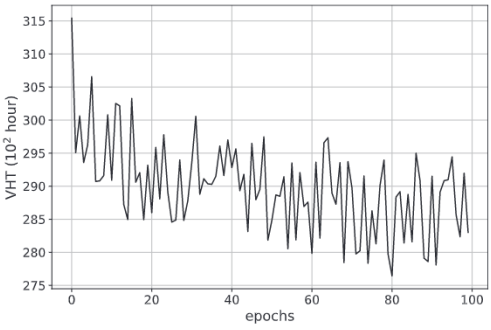

Fig. 6. Training process.

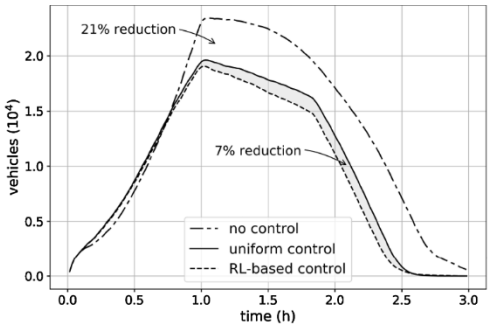

Fig. 7. Time-series of network accumulation.

The first set of experiments were performed for the cordon arrangement previously shown in Fig. 5(a).[9] Outcomes are presented in Fig. 7. Each of its three curves is the time-series of network-wide vehicle accumulation for one of the three control strategies. And each is the average of five simulations with distinct random seeds.[10]

The unshaded area between the dot-dash and solid curves is the network VHT saved via spatially-uniform control. A reduction of 21% was achieved over the do-nothing strategy in which signals did not function as cordon meters. We attribute the large reduction to the high traffic demand and to our use of a Network Transmission Model that was specially developed to improve upon shortcomings of existing models; see again Ni and Cassidy (2018).

The lightly-shaded area between the solid and dashed curves in Fig. 7 indicates that the RL-controller's redistribution of metering rates (i.e. the spatially-varying actions) saved an additional 7% in VHT, above and beyond the impressive reduction achieved via uniform metering. We note for good measure that the extra savings generated in each of the five simulations ranged from 2% to 13% relative to spatially-uniform control. The point being that extra savings occurred via spatially-varying control, without exception, in all five trials.

[9] Eight links fed traffic through each side of the cordon.

[10] A t-test for the samples from 5 simulation runs generated a p-value of 0.023, indicating that the effectiveness of our proposed RL-controller is statistically significant.

Not surprisingly, the added savings grew when demand patterns were altered to create greater traffic inhomogeneities over the network. This was tested in a simple way by transferring part of the demand to cross one of B's perimeters (e.g. the north perimeter) to the opposite (e.g. south) perimeter. With shifts of this kind, the RL-controller saved greater amounts of VHT relative to what was saved under uniform metering. For example, a 15% shift in demand as described above, resulted in 10% additional savings in VHT, up from the 7% shown in Fig. 7.

These sensitivity tests were conducted without re-timing the RL-controller. More is said on this matter below.

3.4. Portability

In light of the time-consuming nature of the training process (see again Section 3.2), we explore how well the trained controller can be used on distinct cordons without additional training. The first experiments along these lines entailed moving the cordon to the right-side of the network, as shown in Fig. 8(a), to surround the neighborhood labeled B'. Trip destinations were likewise moved to fit uniformly across B' (only).[11] Origins continued to be uniformly distributed over the entire network, neighborhoods A and B' in the present case.

The rightward shift of the cordon changed the character of the network's O-D demand pattern. Note that rightward-bound trips from A to B' now double in number relative to rightward trips from A to B in Fig. 5(a). This increased rightward demand (and the longer queues that were now apt to form at the cordon's vertical perimeter) added to traffic's spatial in-homogeneity. Moreover, the limits of what could be achieved via cordon control changed, since now only a 3-sided perimeter could be metered. And a greater number of vehicles per unit length now crossed what remained of the cordon line.

Outcomes are shown in Fig. 8(b). The curves are again the averages of five simulations. By visually comparing the figure's dot-dash curve with its counterpart in Fig. 7, one sees how VHT, when left uncontrolled, increased due to the above-noted changes in the network.[12] Fig. 8(b) also shows that spatially-uniform control continued to ameliorate congestion. In this case, VHT was reduced by over 23%. The figures's lightly-shaded region shows that the RL-controller continued to squeeze-out additional savings in VHT. The 4% improvement in this case is down from the reductions of 7% or more described in Section 3.2. The degraded performance is surely due in part to the controller's lack of case-specific training. Some of the degradation may also be caused by the limits imposed on cordon control by having lost a meter-able perimeter. In contrast, traffic's increased inhomogeneity may have influenced the controller's performance in the opposite (i.e. favorable) direction. Our failure to sort-out the individual effects of these various influences does not change the following observation: once trained in one environment, the controller can be applied elsewhere on the network and still reap benefits.

The same finding came in our second test on portability. In it, the cordon (and the uniformly-distributed trip destinations) were moved to the network's lower-right corner, as shown in Fig. 9(a). The cordon could now be metered on two sides only. Queuing was therefore now more pronounced at the cordon, as can be inferred from the dot-dash curve in Fig. 9(b). Cordon control was especially effective in this more congested environment. Spatially-uniform metering reduced network VHT by more than 28%. Redistribution of those metering rates via the RL-controller was more effective as well: the extra VHT reduction in this case came to 5%, up from the 4% achieved in the previous test.

4. Conclusions

Reinforcement Learning has been used in diverse fields that include robotics, operations research, economics and even game-playing. Extending RL to the cordon-control problem would seem worth exploring, especially in light of the uncertainties that surround the problem, and of the need for real-time (i.e. rapid) decision-making. The present paper is, to our knowledge, the first to pursue this extension. It entailed use of directed-graph representations to embed information on both: link geometries and other static features of a street network; and dynamic elements of the network's directionally-served traffic. These representations required customizations to neural network architectures. The customization enabled the creation of an RL-controller that selects spatially-varying metering rates in optimal fashion.

On a downside, the automated learning style of neural networks produces a "black-box" solution. Our customized convolution process, for example, aggregated static and dynamic information in non-transparent ways. The decision-making rules that produced optimal control actions are similarly unknown.

These concerns are offset by the benefits, at least in part. After all, the controller selects optimal metering actions despite our own limited understanding of how best to respond to traffic in-homogeneities. This would seem to be of much practical value. It bears repeating that selections are made in ways that remedy pitfalls of previous methods. Little wonder perhaps that the RL-controller diminished network-wide VHT in all of the idealized cases tested. The reductions were over and above those achieved by spatially-uniform metering plans. The latter plans were already highly effective on their own.

Other of the controller's useful features includes its customized convolution process. It produces generalizable rules that give the controller its portability. The time- and resource-intensive costs of training the RL-based system thus become a one-time expenditure for a city to bear.

[11] Destinations were moved in this way since it would be counterproductive to meter a cordon that skirts destination-rich areas on a network; see Daganzo (2007).

[12] Visual inspection of that curve also reveals that some vehicles remained on the network when the 3-h simulations ended.

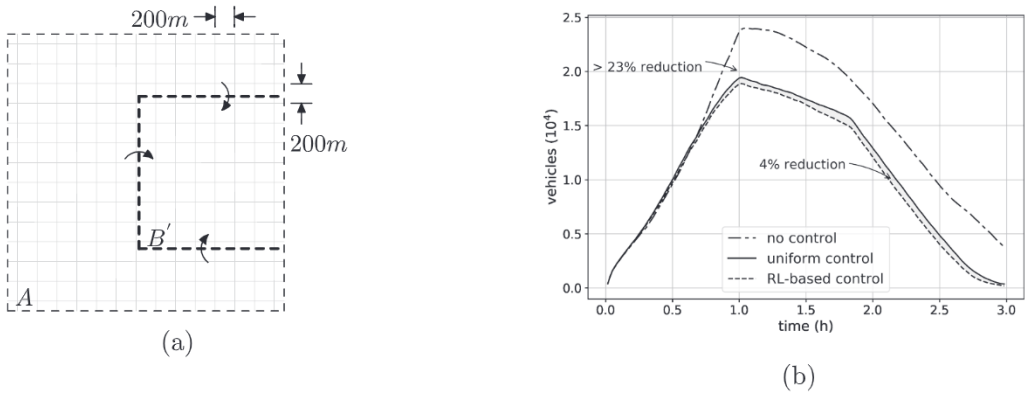

Fig. 8. Portability test 1 (shifted cordon and VHT).

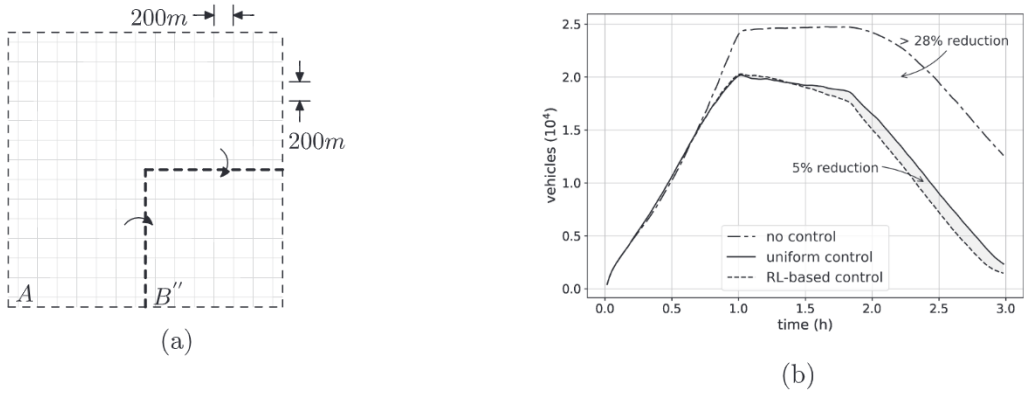

Fig. 9. Portability test 2 (shifted cordon and VHT).

The controller's two-stage framework has practical advantages as well. Within this framework, control actions selected by the black-box approach in Stage II are constrained by the optimal allowable inflows generated via the model-based approach in Stage I. The controller is thus constrained in how far its metering rates can drift from an optimum. This would be the case even during training periods. Any lingering fears about counterproductive actions during training might be eased by pre-training the controller using computer simulation, much like in the present work.

The above consideration could ally a city's fears about black-box control rules, and enhance the likelihood of deploying the controller in real settings. Field tests will be needed first. Our hope is that the present paper might hasten the occurrence of these needed, real-world tests.

Acknowledgment

This work was partially supported with funds from UCCONNECT, the University Transportation Center for federal region 9, and by the National Science Foundation.

Appendix A. Deep deterministic policy gradient

We trained our actor and critic networks together using the Deep Deterministic Policy Gradient (DDPG) method proposed in Lillicrap et al. (2015). Prioritized experience replay (Schaul et al., 2015), Target network (Mnih et al., 2015) and Adaptive $\epsilon-$greedy exploration policy (Tokic, 2010) were also used in the training process. The Adam algorithm (Kingma and Ba, 2014) was used for the optimization. The single change made to the original DDPG algorithm involved updating the actor by minimizing the penalty loss, as was presented in Fig. 4. Psuedo-code for the customized DDPG algorithm is shown below.

Algorithm 1. Deep Deterministic Policy Gradient (Lillicrap et al., 2015)

Initialize critic network $Q(s, a|\theta^Q)$ and actor network $\pi(s|\theta^\pi)$
Initialize target critic network $Q(s, a|\theta^{Q'})$ and target actor network $\pi(s|\theta^{\pi'})$
Initialize prioritized experience replay buffer R
while not converge **do**

Observe system state $s^{(t)}$

Select action $a^{(t)}$ for $s^{(t)}$ following ϵ-greedy exploration policy

Observe transition pair $(s^{(t)}, a^{(t)}, s^{(t+1)}, r^{(t)})$, and store in R

Sample a mini-batch $(s^{(t)}, a^{(t)}, s^{(t+1)}, r^{(t)})$ of size N from R

Set $y^{(i)} = r^{(i)} + \gamma Q(s^{(i+1)}, \pi(s^{(i+1)}|\theta^{\pi'})|\theta^{Q'})$

Update critic by minimizing

$$\sum_i (y^{(i)} - Q(s^{(i)}, a^{(i)}|\theta^Q))^2$$

Update actor with the sampled policy gradient:

$$\nabla_{\theta^\pi} J \sim \frac{1}{N} \sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s^{(i)}, a=\pi(s^{(i)})} \nabla_{\theta_\pi} \pi(s|\theta^\pi)|_{s^{(i)}}$$

Update actor by minimizing the penalty loss

$$\sum_i (1^T (f^{(i)} \cdot d^{(i)} - a^{(i)} \cdot d^{(i)}))^2$$

Update target networks:

$$\theta^{\pi'} \leftarrow \tau \cdot \theta^\pi + (1 - \tau) \cdot \theta^{\pi'}, \ \theta^{Q'} \leftarrow \tau \cdot \theta^Q + (1 - \tau) \cdot \theta^{Q'}$$

References

Aboudolas, K., Geroliminis, N., 2013. Perimeter and boundary flow control in multi-reservoir heterogeneous networks. Transport. Res. Part B: Methodol. 55, 265–281.

Balakrishna, P., Ganesan, R., Sherry, L., 2010. Accuracy of reinforcement learning algorithms for predicting aircraft taxi-out times: a case-study of tampa bay departures. Transport. Res. Part C: Emerg. Technol. 18 (6), 950–962.

Braess, D., Nagurney, A., Wakolbinger, T., 2005. On a paradox of traffic planning. Transport. Sci. 39 (4), 446–450.

Bruna, J., Zaremba, W., Szlam, A., LeCun, Y., 2013. Spectral Networks and Locally Connected Networks on Graphs. Available from: arXiv:1312.6203.

Cai, C., Wong, C.K., Heydecker, B.G., 2009. Adaptive traffic signal control using approximate dynamic programming. Transport. Res. Part C: Emerg. Technol. 17 (5), 456–474.

Castillo, E., Conejo, A.J., Menéndez, J.M., Jiménez, P., 2008. The observability problem in traffic network models. Comput.-Aid. Civil Infrastruct. Eng. 23 (3), 208–222.

Daganzo, C.F., 2007. Urban gridlock: macroscopic modeling and mitigation approaches. Transport. Res. Part B: Methodol. 41 (1), 49–62.

Daganzo, C.F., Gayah, V.V., Gonzales, E.J., 2011. Macroscopic relations of urban traffic variables: bifurcations, multivaluedness and instability. Transport. Res. Part B: Methodol. 45 (1), 278–288.

Daganzo, C.F., Geroliminis, N., 2008. An analytical approximation for the macroscopic fundamental diagram of urban traffic. Transport. Res. Part B: Methodol. 42 (9), 771–781.

Daganzo, C.F., Lehe, L.J., Argote-Cabanero, J., 2017. Adaptive offsets for signalized streets. Transport. Res. Part B: Methodol.

Defferrard, M., Bresson, X., Vandergheynst, P., 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems, pp. 3844–3852.

Dhillon, I.S., Guan, Y., Kulis, B., 2007. Weighted graph cuts without eigenvectors a multilevel approach. IEEE Trans. Pattern Anal. Mach. Intell. 29 (11).

Du, J., Rakha, H., Gayah, V.V., 2016. Deriving macroscopic fundamental diagrams from probe data: Issues and proposed solutions. Transport. Res. Part C: Emerg. Technol. 66, 136–149.

Gayah, V., Dixit, V., 2013. Using mobile probe data and the macroscopic fundamental diagram to estimate network densities: tests using microsimulation. Transport. Res. Rec.: J. Transport. Res. Board (2390), 76–86.

Geroliminis, N., Daganzo, C.F., 2008. Existence of urban-scale macroscopic fundamental diagrams: some experimental findings. Transport. Res. Part B: Methodol. 42 (9), 759–770.

Geroliminis, N., Haddad, J., Ramezani, M., 2013. Optimal perimeter control for two urban regions with macroscopic fundamental diagrams: a model predictive approach. IEEE Trans. Intell. Transport. Syst. 14 (1), 348–359.

Haddad, J., 2017. Optimal perimeter control synthesis for two urban regions with aggregate boundary queue dynamics. Transport. Res. Part B: Methodol. 96, 1–25.

Henaff, M., Bruna, J., LeCun, Y., 2015. Deep convolutional networks on graph-structured data. Available from: arXiv:1506.05163.

Ji, Y., Daamen, W., Hoogendoorn, S., Hoogendoorn-Lanser, S., Qian, X., 2010. Investigating the shape of the macroscopic fundamental diagram using simulation data. Transport. Res. Rec.: J. Transport. Res. Board (2161), 40–48.

Ji, Y., Geroliminis, N., 2012. On the spatial partitioning of urban transportation networks. Transport. Res. Part B: Methodol. 46 (10), 1639–1656.

Jusoh, R.M., Ampountolas, K., 2017. Multi-gated perimeter flow control of transport networks. In: 2017 25th Mediterranean Conference on Control and Automation (MED). IEEE, pp. 731–736.

Keyvan-Ekbatani, M., Carlson, R.C., Knoop, V.L., Hoogendoorn, S.P., Papageorgiou, M., 2016. Queuing under perimeter control: analysis and control strategy. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 1502–1507.

Keyvan-Ekbatani, M., Carlson, R.C., Knoop, V.L., Papageorgiou, M., 2017. Balancing delays and relative queues at the urban network periphery under perimeter control, Technical report.

Keyvan-Ekbatani, M., Kouvelas, A., Papamichail, I., Papageorgiou, M., 2012. Congestion control in urban networks via feedback gating. Proc.-Soc. Behav. Sci. 48, 1599–1610.

Keyvan-Ekbatani, M., Papageorgiou, M., Knoop, V.L., 2015a. Controller design for gating traffic control in presence of time-delay in urban road networks. Transport. Res. Part C: Emerg. Technol. 59, 308–322.

Keyvan-Ekbatani, M., Yildirimoglu, M., Geroliminis, N., Papageorgiou, M., 2015b. Multiple concentric gating traffic control in large-scale urban networks. IEEE Trans. Intell. Transport. Syst. 16 (4), 2141–2154.

Kingma, D., Ba, J., 2014. Adam: a method for stochastic optimization. Available from: arXiv:1412.6980.

Knoop, V.L., Hoogendoorn, S., Van Lint, J., 2013. The impact of traffic dynamics on macroscopic fundamental diagram. In: 92nd Annual Meeting Transportation Research Board, Washington, USA, 13–17 January 2013; Authors version. Transportation Research Board.

Konda, V.R., Tsitsiklis, J.N., 2000. Actor-critic algorithms. In: Advances in Neural Information Processing Systems, pp. 1008–1014.

Kouvelas, A., Saeedmanesh, M., Geroliminis, N., 2017. Enhancing model-based feedback perimeter control with data-driven online adaptive optimization. Transport. Res. Part B: Methodol. 96, 26–45.

Levie, R., Monti, F., Bresson, X., Bronstein, M.M., 2017. Cayleynets: graph convolutional neural networks with complex rational spectral filters. Available from: arXiv:1705.07664.

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D., 2015. Continuous control with deep reinforcement learning. Available from: arXiv:1509.02971.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M., 2013. Playing atari with deep reinforcement learning. Available from: arXiv:1312.5602.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.

Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814.
Ni, W., Cassidy, M.J., 2018. City-Wide Traffic Control: Modeling Impacts of Cordon Queues. Institute of Transportation Studies, UC Berkeley.
Ramezani, M., Haddad, J., Geroliminis, N., 2015. Dynamics of heterogeneity in urban networks: aggregated traffic modeling and hierarchical control. Transport. Res. Part B: Methodol. 74, 1–19.
Saeedmanesh, M., Geroliminis, N., 2016. Clustering of heterogeneous networks with directional flows based on snake similarities. Transport. Res. Part B: Methodol. 91, 250–269.
Schaul, T., Quan, J., Antonoglou, I., Silver, D., 2015. Prioritized experience replay. Available from: arXiv:1511.05952.
Sutton, R.S., Barto, A.G., 1998. Reinforcement Learning: An Introduction, vol. 1 MIT Press, Cambridge.
Tokic, M., 2010. Adaptive ε-greedy exploration in reinforcement learning based on value differences. In: Annual Conference on Artificial Intelligence. Springer, pp. 203–210.
Zhu, F., Ukkusuri, S.V., 2014. Accounting for dynamic speed limit control in a stochastic traffic environment: a reinforcement learning approach. Transport. Res. Part C: Emerg. Technol. 41, 30–47.