Reducing Urban Traffic Congestion Using Deep Learning and Model Predictive Control

Zhun Yin, Tong Liu, *Graduate Student Member, IEEE*, Chieh Wang, *Member, IEEE*, Hong Wang, *Fellow, IEEE*, and Zhong-Ping Jiang, *Fellow, IEEE*

***Abstract*— This article proposes a deep learning (DL)-based control algorithm—DL velocity-based model predictive control (VMPC)—for reducing traffic congestion with slowly time-varying traffic signal controls. This control algorithm consists of system identification using DL and traffic signal control using VMPC. For the training process of DL, we established a modeling error entropy loss as the criteria inspired by the theory of stochastic distribution control (SDC) originated by the fourth author. Simulation results show that the proposed algorithm can reduce traffic congestion with a slowly varying traffic signal control input. Results of an ablation study demonstrate that this algorithm compares favorably to other model-based controllers in terms of prediction error, signal varying speed, and control effectiveness.**

***Index Terms*— Deep learning (DL), gain-scheduling, model predictive control (MPC), traffic signal control, velocity-based linearization (VL).**

I. Introduction

With the increase in traffic congestion, urbanization, and vehicle ownership, the optimal setting and control of traffic lights have become a challenge for urban areas [1], [2]. In recent decades, systematic control-theoretic methods have been applied to tackle this problem [3], [4], [5], [6]. Because of the complexity of traffic systems, an accurate system identification method for a model-based controller design is needed. Compared with least-squares (LS) methods commonly used in system identification, it has been shown that deep neural networks can model the inherent nonlinearity of traffic system. Indeed, deep learning (DL) for traffic system prediction has been studied recently [7], [8], [9]. However, few control schemes have been considered based on the DL prediction models for traffic systems, which have recently been attracting more attention [10].

To control the identified system, researchers often linearize the system at an equilibrium point of interest [5]. This linearization method is accurate only when the system state is near the equilibrium point. However, in the traffic system, operating points are often far away from any specific equilibrium point in the presence of unpredictable, and sometimes large, traffic demands as disturbances.

The gain-scheduling methods (see research from Leith and Leithead [11] and numerous references therein), on the other hand, are powerful approaches to the nonlinear control system design, retaining the advantage of linearizing a nonlinear system at a sequence of operating points. Specially, the linearization method adopted by Leith and Leithead [11], namely, the velocity-based linearization (VL), is a good approximation to the original system, provided that the system has locally bounded first-order derivatives, and the system states and control signals are slowly time-varying [11], [12], [13]. Because of physical limitations from the dynamics of traffic flows, and the rate of change of control signals is not expected to be overly rapid in real-world applications, the requirements of slowly varying states and inputs can usually be met. Thus, such VL is suitable for the traffic system intuitively, and we will validate this intuition by means of numerical simulation results. Note that in this article we approximate the nonlinear dynamics of the traffic system using the fact that the deep neural network can act as a universal approximator [14], which is also a salient feature of fuzzy logic systems [15], to which the gain-scheduling method was recently adopted in the work [16]. More specifically, in [16], the Takagi–Sugeno (T-S) fuzzy system is adopted to approximate nonlinear systems by smoothly merging several linear systems, and a switching-type gain-scheduling control law is proposed to reduce the conservativeness of the alert threshold condition for resilient fuzzy stabilization [17].

To take into account the control effect and the varying speed of the control signals, we designed the controller based on the optimal control methods. Velocity-based optimal control has been studied in the literature [18], [19]. Since infinite-horizon linear quadratic regulation (LQR) is not generally applicable for the VL of nonlinear systems [18], we applied techniques in finite-horizon LQR and model predictive control (MPC) to tackle this problem. Another reason for selecting MPC is that it can handle the state and input constraints that often exist in real-world applications. Recently, a field deployment experiment of the MPC algorithm based on the virtual phase-link (VPL) model has been conducted in downtown Chattanooga, Tennessee [20].

In this article, an MPC controller for the VL of the nonlinear traffic system modeled by DL is proposed. A multilayer

Manuscript received 27 March 2022; revised 25 January 2023; accepted 27 March 2023. This work was supported in part by the U.S. Department of Energy, Vehicle Technologies Office, Energy Efficient Mobility Systems Program, and in part by the National Science Foundation under Grant EPCN-1903781. *(Corresponding author: Zhun Yin.)*

Zhun Yin, Tong Liu, and Zhong-Ping Jiang are with the Department of Electrical and Computer Engineering, New York University, Brooklyn, NY 11201 USA (e-mail: zy1652@nyu.edu; tl3049@nyu.edu; zjiang@nyu.edu).

Chieh Wang and Hong Wang are with the Buildings and Transportation Science Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831 USA (e-mail: cwang@ornl.gov; wangh6@ornl.gov).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2023.3264709.

Digital Object Identifier 10.1109/TNNLS.2023.3264709

perceptron (MLP) was used to model the traffic system as a discrete-time nonlinear system with the number of vehicles in each link as the state variables, and the difference between the green times of the actual traffic signal and a fixed-time traffic signal as the control input. In addition, for the training process of DL, we propose learning error entropy by regarding the training process as a stochastic distribution control (SDC) problem [21]. The VL method is then applied to this nonlinear system model along the operating trajectory. As a consequence, the original nonlinear system can be regarded as a linear time-varying system with the system matrix being the Jacobian of the neural network at each operating point. Linear MPC control is then applied to the linearized system at each operating point to obtain the optimal green time splits for a set of intersections, subject to state and input constraints. Finally, the bounded-input, bounded-output (BIBO) stability of the closed-loop linear time-varying system is proven.

The contributions of this article are threefold.

1) For traffic systems, although DL has been intensively used in traffic congestion prediction [7], [8], this article is the first to apply gain-scheduling control to traffic signal control tasks based on DL prediction.
2) The combination of DL and velocity-based MPC (VMPC) for the controller design has not been investigated, and a new training loss inspired by SDC is proposed.
3) From a theoretical perspective, this article demonstrates the BIBO stability for the proposed method, contrary to other similar traffic-responsive urban control methods [22], [23], which often lack stability guarantees [24].

This article is structured as follows. Section II conceptually introduces the traffic signal control, DL, entropy loss, VL, and VMPC; Section III introduces the proposed DL-VMPC; Section IV introduces two baseline methods and an ablation study; Section V provides the numerical results; Section VI demonstrates the BIBO stability for the proposed method; and Section VII provides conclusions of this research.

Notation: $\mathbb{N}$ ($\mathbb{N}_+$) denotes the set of all (positive) natural numbers. $\mathbb{Z}$ denotes the set of all integers. $\mathbb{R}$ denotes the set of all real numbers. $\mathbb{R}^n$ denotes the n-dimensional Euclidean space. For any vector $x \in \mathbb{R}^n$, we denote $x = [x_1, x_2, \ldots, x_n]^{\mathrm{T}}$. $\mathbb{Z}^n$ denotes set $\{x \in \mathbb{R}^n | \ x_i \in \mathbb{Z} \text{ for all } i\}$, $\mathbb{N}^n$ denotes set $\{x \in \mathbb{R}^n | \ x_i \in \mathbb{N} \text{ for all } i\}$. $\mathbb{R}^n_+$ denotes set $\{x \in \mathbb{R}^n | \ x_i \geq 0 \text{ for all } i\}$, and $\mathbb{N}^n_+$ denotes set $\{x \in \mathbb{R}^n | \ x_i \in \mathbb{N}_+ \text{ for all } i\}$. A partial order "$\preceq$" on $\mathbb{R}^n$ is defined by $x \preceq y$ if and only if $y - x \in \mathbb{R}^n_+$. The Hadamard product between two matrices of the same dimension $A, B \in \mathbb{R}^{m \times n}$ is denoted as $A \odot B$.

II. Problem Formulation and Preliminaries

A. Traffic Signal Control

Generally, traffic signal control consists of four primary aspects: phase specification, signal timing/split control, cycle duration, and cycle offset. Among these four aspects, signal timing/split control, which sets the relative green duration of each phase, has potentially the most significant effect on the operation of traffic flows in a network of intersections [3]. Thus, in this article, we describe the control signal as the green time of all traffic lights, denoted as $u^k \in \mathbb{N}^m$ for time step k, where m is the number of controlled green phases in a particular area and the output as the number of vehicles on the concerned n links in the area denoted by $x^k \in \mathbb{N}^n$ [22]. As a result, the object of the control system design is to select u^k so that x^k is made as small as possible subjected to variable traffic demand.

At present, the fixed-time traffic light signal control scheme has been widely used, especially in urban settings, partly because of the nature of the randomness of traffic conditions in cities, and partly because of its simplicity for implementation. For example, stage-based approaches SIGSET [25] and SIGCAP [26] use the well-known Webster's delay formula to generate fixed-time strategies [24]. In this article, the control input is set as the differences between the green time of all traffic lights and a predetermined fixed-time controller $u \in \mathbb{N}^m$, denoted as $\delta u^k := u^k - u$.

To achieve the aforementioned control objective, it is imperative to develop an effective modeling strategy at first. Considering the unknown and nonlinear nature of the traffic system, neural networks such as MLP will be used here as discussed in the next subsection.

B. Multilayer Perceptron

MLP is a class of feedforward artificial neural networks. We use this terminology to refer to networks comprising multiple layers of neurons.

In the mathematical theory of artificial neural networks, universal approximation theorems ensure that such MLP can closely approximate any continuous function defined on a compact set arbitrarily, provided that the width of the network is sufficiently large [27]. This universal approximation property of MLP was used to learn the underlying nonlinear dynamics of the traffic system, denoted as

$$x^{k+1} = f\left(x^k, \delta u^k\right) \tag{1}$$

where $f: \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a unknown nonlinear function that captures the system dynamics, and we assume that f has locally bounded first-order derivatives.

Remark 1: Since u is a constant vector, by defining $f_0(x^k, u^k) := f(x^k, u^k - u)$, we can see that learning the traffic system in the form $f(x^k, \delta u^k)$ is equivalent to learning the traffic system in the form $f_0(x^k, u^k)$. Using δu^k instead of using u^k directly is based on the model introduced in [22] which was designed for the traffic-responsive urban control (TUC) strategy. In [22], the traffic model is as follows:

$$x^{k+1} = x^k + B_0 \delta u^k \tag{2}$$

for some constant matrix B_0. One of the motivations of our article is to generalize (2) to a nonlinear model $x^{k+1} = f(x^k, \delta u^k)$ via DL. Thus, we use δu^k as our control inputs.

C. Neural Network Training Using Learning Error Entropy as the Loss Function

1) Stochastic Distribution Control: For a stochastic system subjected to random signals $w(t)$, where $t \in \mathbb{R}_+$ represents

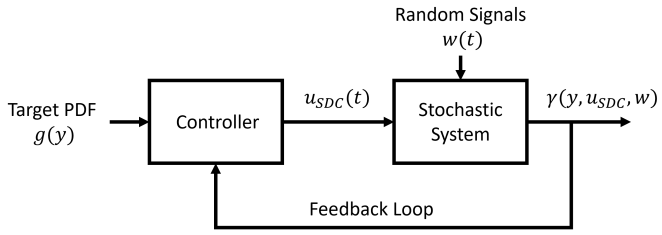

Fig. 1. Feedback structure of SDC.

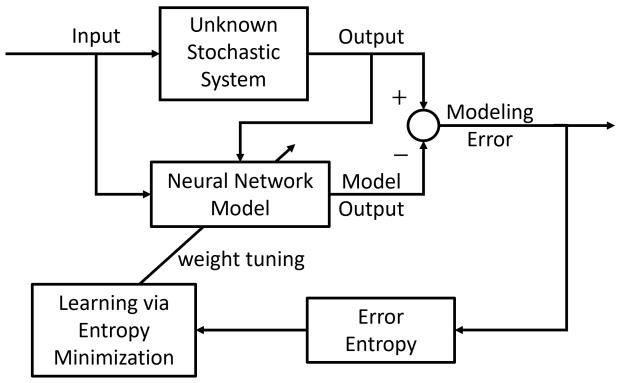

Fig. 2. Training process of a neural network regarded as an SDC problem.

the time, an important aspect of the controller design is to attenuate the uncertainties contained in the controlled system outputs or tracking errors. As shown in Fig. 1, SDC is a control system theory developed to control non-Gaussian and dynamic stochastic systems whose actual controlled output is the shape of its output probability density functions (PDFs) $\gamma(y, u_{\text{SDC}}, w)$, where $y \in [a, b]$ is the distributions' range, and $-\infty < a < b < \infty$ [21].

The objective of the SDC theory is to make the output PDF $\gamma(y, u_{\text{SDC}}, w)$ as close to a target PDF $g(y)$ as possible by formulating the control inputs $u_{\text{SDC}}(t)$ [28].

2) Learning Error Entropy Loss: In this article, the training process of a neural network is regarded as an SDC problem. As shown in Fig. 2, a neural network with parameter θ can be regarded as a stochastic system, and the finite sequence of batches of the training dataset $\{\mathcal{B}^\tau\}_{\tau=0}^{T-1}$ can be regarded as random signals, where $\tau \in \mathbb{N}$ represents the τth iteration. The actual controlled output is the PDF of each component of the neural network's error (label minus the neural network's output). The PDF of the qth component of error is denoted as $\gamma_{e_q}(y, \theta, \mathcal{B}^\tau)$.

Thus, the training process of a neural network can be regarded as a controller design problem of SDC, which formulates parameters $\{\theta^\tau\}_{\tau=0}^{T}$ for the neural network as SDC control inputs $u_{\text{SDC}}(\tau)$. For each q, the aim is to make $\gamma_{e_q}(y, \theta^\tau, \mathcal{B}^\tau)$ as close to a Dirac delta distribution as possible. This is because a narrowly distributed PDF of the modeling error with zero mean would generally indicate that the uncertainties in the error are small and the trained model thus obtained has a high reliability and robustness. In practice, a normal distribution PDF $g(y)$ with zero mean and a very small variance was adopted. Based on this target PDF of the modeling error, we formulate the modeling error entropy loss as

$$\begin{aligned}&\mathcal{L}_{\text{Entropy}}(\theta^\tau, \mathcal{B}^\tau)\\&= \frac{c}{n_{\text{output}}} \sum_{q=1}^{n_{\text{output}}} \int_a^b \gamma_{e_q}(y, \theta^\tau, \mathcal{B}^\tau) \log\left\{\frac{\gamma_{e_q}(y, \theta^\tau, \mathcal{B}^\tau)}{g(y)}\right\} dy\\&\quad + \frac{1}{|\mathcal{B}^\tau|} \sum_{i=1}^{|\mathcal{B}^\tau|} \left\|e^{[i]}\right\|_2 \end{aligned} \tag{3}$$

where $e^{[i]} \in \mathbb{R}^{n_{\text{output}}}$ is the error for the ith sample in batch $\mathcal{B}^\tau$; n_{output} is the dimension of the neural network's output layer; $c > 0$ is a constant; and $|\mathcal{B}^\tau|$ is the size of batch $\mathcal{B}^\tau$.

D. Gain-Scheduling and VL

Gain-scheduling is a nonlinear control approach that aims to achieve desirable performance by scheduling the feedback gains of a family of linear controllers. Each linear controller provides a satisfactory control to a different system operating point. The controller is selected and enabled based on a subset of observable variables, called "scheduling variables."

To apply a linear controller at each operating point, suitable linearization methods should be adopted for the nonlinear system. In this article, VL [11] was used. For nonlinear system (1), using Taylor's theorem [29], we have

$$\begin{aligned}x^{k+1} = f\left(x^k, \delta u^k\right) = f\left(x^{k-1}, \delta u^{k-1}\right) + A_{k-1}\Delta x^k\\ + B_{k-1}\Delta u^k + o\left(\left\|\left[\left(\Delta x^k\right)^{\mathrm{T}}, \left(\Delta u^k\right)^{\mathrm{T}}\right]\right\|_2\right)\end{aligned} \tag{4}$$

where $A_{k-1} := (\partial f)/(\partial x)|_{\substack{x=x^{k-1},\\ u=\delta u^{k-1}}}$, $B_{k-1} := (\partial f)/(\partial u)|_{\substack{x=x^{k-1},\\ u=\delta u^{k-1}}}$, $\Delta x^k := x^k - x^{k-1}$, and $\Delta u^k := \delta u^k - \delta u^{k-1}$. The final term in (4) stands for the approximation error or higher order approximation error.

Based on (1), we have $x^k = f(x^{k-1}, \delta u^{k-1})$. Thus, we can rewrite (4) as

$$\Delta x^{k+1} = A_{k-1}\Delta x^k + B_{k-1}\Delta u^k + o\left(\left\|\left[\left(\Delta x^k\right)^{\mathrm{T}}, \left(\Delta u^k\right)^{\mathrm{T}}\right]\right\|_2\right). \tag{5}$$

If x^k and δu^k are slowly time-varying, we can omit the last term of (5). Then, we can derive the VL at $(x^k, \delta u^k)$ for (1)

$$\begin{bmatrix} x^{k+1} \\ \Delta x^{k+1} \end{bmatrix} = \begin{bmatrix} I & A_{k-1} \\ 0 & A_{k-1} \end{bmatrix} \begin{bmatrix} x^k \\ \Delta x^k \end{bmatrix} + \begin{bmatrix} B_{k-1} \\ B_{k-1} \end{bmatrix} \Delta u^k. \tag{6}$$

Note that at time step k, the Taylor expansion is adopted at $(x^{k-1}, \delta u^{k-1})$ in (6). From the viewpoint of gain-scheduling, the VL takes the operating point of the previous time step as its scheduling variable for the current time step.

E. Velocity-Based MPC

To reduce traffic congestion with slowly time-varying traffic control signals, an MPC controller is applied, whose cost function at each sampling instance k is

$$\begin{aligned}J_k := \sum_{i=1}^{N} \Bigg(\begin{bmatrix} x^{k+i} \\ \Delta x^{k+i} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x^{k+i} \\ \Delta x^{k+i} \end{bmatrix}\\ + \Delta\left(u^{k+i-1}\right)^{\mathrm{T}} R \Delta u^{k+i-1} \Bigg)\end{aligned} \tag{7}$$

where the matrices $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are the prespecified symmetric and positive definite matrices, and $N \in \mathbb{N}_+$ is the prediction horizon. In addition, we constrain the varying speed of green signal duration explicitly

$$\left|\Delta u^{k+i-1}\right|^c \preceq \Delta u_{\max}, \quad i = 1, \ldots, N \tag{8}$$

where $|\Delta u^{k+i-1}|^c$ is the componentwise absolute value of Δu_{k+i-1}, and constant $\Delta u_{\max} \in \mathbb{N}_+^m$ denotes the upper bounds for the green signal duration change.

Considering the constraints of minimum green time for pedestrians and the maximum green time based on the cycle length (summation of all phases' duration time equals cycle length), we impose the following constraints:

$$u_{\min} \preceq u^{k+i-1} \preceq u_{\max}, \quad i = 1, \ldots, N \tag{9}$$

where $u_{\min}, u_{\max} \in \mathbb{N}_+^m$ denote the lower and upper bounds of the green time duration, respectively.

Combining (6)–(9), we define the VMPC problem as

$$\min_{u^k,\ldots,u^{k+N-1}} J_k \tag{10a}$$

$$\text{s.t.} \begin{bmatrix} x^{k+i} \\ \Delta x^{k+i} \end{bmatrix} = \begin{bmatrix} I & A_{k-1} \\ 0 & A_{k-1} \end{bmatrix} \begin{bmatrix} x^{k+i-1} \\ \Delta x^{k+i-1} \end{bmatrix} + \begin{bmatrix} B_{k-1} \\ B_{k-1} \end{bmatrix} \Delta u^{k+i-1}, \quad i = 1, \ldots, N \tag{10b}$$

$$\left|\Delta u^{k+i-1}\right|^c \preceq \Delta u_{\max}, \quad i = 1, \ldots, N \tag{10c}$$

$$u_{\min} \preceq u^{k+i-1} \preceq u_{\max}, \quad i = 1, \ldots, N. \tag{10d}$$

Indeed, the MPC controller (10a)–(10d) can be viewed as a linear controller designed to control the traffic system linearized at the last operating point. Note that since the traffic system is slowly varying and the traffic signals generated by our MPC controller are also slowly varying, the linearized system (6) is a good approximation of the original nonlinear traffic system (1).

III. Traffic Signal Controller Design

A. DL for Traffic System Prediction

The dataset $\mathcal{D} = \{((x^k, \delta u^k), x^{k+1})\}_{k=1}^d$ is sampled every cycle length subjected to a set of random control inputs $\{\delta u^k = \lfloor \delta \hat{u}^k + (1/2) \rfloor | \delta \hat{u}^k = 0.1 \times w \odot u, w \in \mathbb{R}^m, w_p \sim U[-0.5, 0.5] \text{ i.i.d. for } p = 1, \ldots, m\}_{k=1}^d$, where $d \in \mathbb{N}_+$ is the size of dataset $\mathcal{D}$ and U stands for uniform distribution.

In the system identification step, we designed an MLP model (see Fig. 3) to identify the dynamics in (1). The MLP model comprises five layers: one input layer, three hidden layers, and one output layer. The number of vehicles at each link at a certain time step together with the difference in green time between the signals at that time step and the fixed time controller are fed into the deep neural network. The output of the neural network is the predicted number of vehicles on each link at the next time step. During the experiments, we found that the activation function combination of Relu, ELU, and Relu produced better prediction results than only using the Relu activation function. We conjecture that this is because the ELU activation function can push mean unit activations closer to zero, which speeds up the learning [30]. Hence, the activation functions for the hidden layers are Relu, ELU, and Relu, successively. The output layer is a linear layer without any activation function. All the outputs are rounded to an integer.

The neural network is denoted as $f_{\text{NN}} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, and the prediction for step $k+1$ generated by the neural network is denoted as $\hat{x}^{k+1}$

$$\hat{x}^{k+1} = f_{\text{NN}}(x^k, \delta u^k). \tag{11}$$

The error e^{k+1} is defined as

$$e^{k+1} = x^{k+1} - \hat{x}^{k+1}. \tag{12}$$

For the τth iteration, batch $\mathcal{B}^\tau$ randomly samples $b^\tau := |\mathcal{B}^\tau|$ instances from $\mathcal{D}$, denoted as $\mathcal{B}^\tau = \{((x^{k_i}, \delta u^{k_i}), x^{k_i+1})\}_{i=1}^{b^\tau}$. Then, the qth component of errors' PDF for this batch can be estimated using kernel distribution estimation [31]

$$\hat{\gamma}_{e_q}(y, \theta^\tau, \mathcal{B}^\tau) = \frac{1}{h \times b^\tau} \sum_{i=1}^{b^\tau} \phi\left(\frac{y - e_q^{k_i+1}}{h}\right) \tag{13}$$

where ϕ is the standard normal distribution kernel, and $0 < h < 1$ is the bandwidth of the kernel distribution estimation.

$\Delta y := (b-a)/p$, where $p \in \mathbb{N}_+$ represents the number of sampling points in the integration interval for computing integral. The modeling error entropy loss in (3) is computed as

$$\begin{aligned} &\mathcal{L}_{\text{Entropy}}(\theta^\tau, \mathcal{B}^\tau) \\ &= \frac{c}{n} \sum_{q=1}^{n} \sum_{j=0}^{p} \hat{\gamma}_{e_q}(a + j\Delta y, \theta^\tau, \mathcal{B}^\tau) \\ &\quad \times \log\left\{\frac{\hat{\gamma}_{e_q}(a + j\Delta y, \theta^\tau, \mathcal{B}^\tau)}{g(a + j\Delta y) + \epsilon} + \epsilon\right\} \Delta y \\ &\quad + \frac{1}{b^\tau} \sum_{i=1}^{b^\tau} \|e^{[i]}\|_2 \end{aligned} \tag{14}$$

where $\epsilon > 0$ is a small constant to prevent numerical problems. Note that since $g(y)$ is always positive, the denominator $g(a + j\Delta y) + \epsilon$ will never be zero.

By performing backpropagation to the loss function $\mathcal{L}_{\text{Entropy}}$, we can update the neural network's parameters to bring f_{NN} close to f. This model f_{NN} is referred to as the "DL model" for convenience.

B. DL-VMPC

Once we obtain a trained DL model, we can identify the underlying dynamics of the traffic system (1) as f_{NN}. Then, we can use VL on the identified nonlinear system and apply VMPC to this linearized system, similar to (10a)–(10d)

$$\min_{u^k,\ldots,u^{k+N-1}} J_k \tag{15a}$$

$$\text{s.t.} \begin{bmatrix} x^{k+i} \\ \Delta x^{k+i} \end{bmatrix} = \begin{bmatrix} I & A_{k-1}^{\text{NN}} \\ 0 & A_{k-1}^{\text{NN}} \end{bmatrix} \begin{bmatrix} x^{k+i-1} \\ \Delta x^{k+i-1} \end{bmatrix} + \begin{bmatrix} B_{k-1}^{\text{NN}} \\ B_{k-1}^{\text{NN}} \end{bmatrix} \Delta u^{k+i-1}, \quad i = 1, \ldots, N \tag{15b}$$

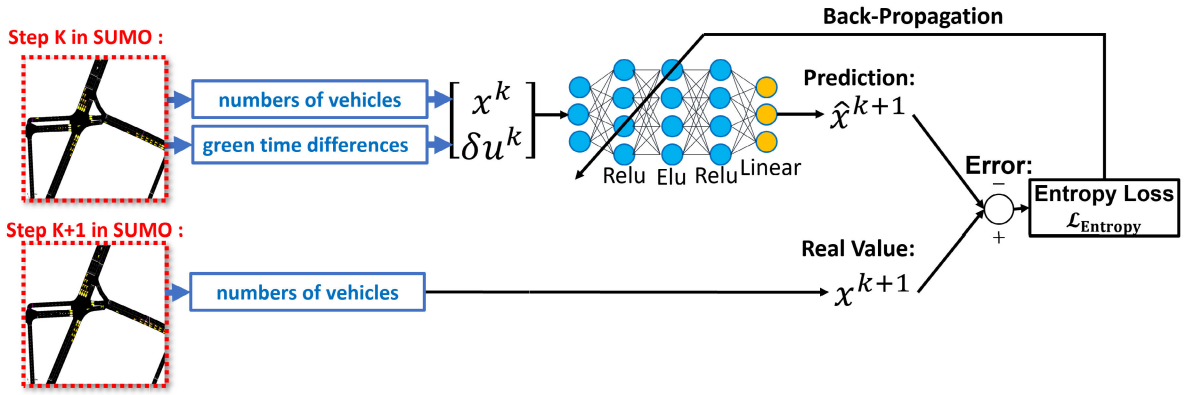

Fig. 3. DL for traffic system prediction.

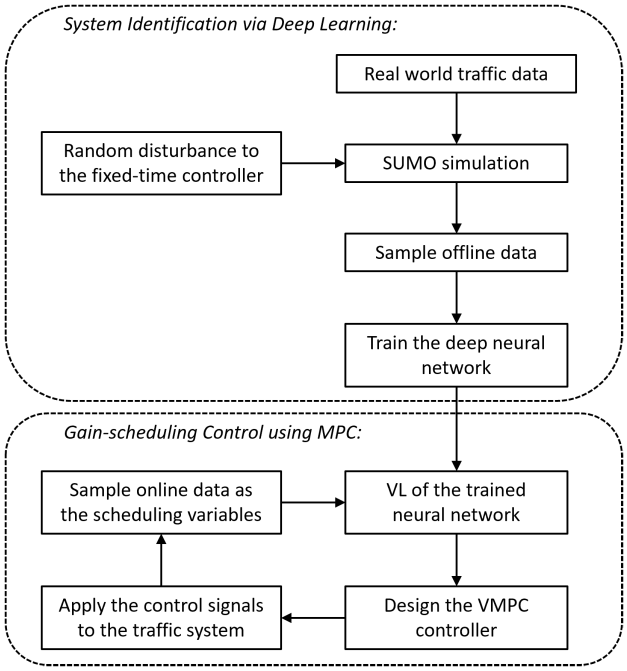

Fig. 4. Design procedure of DL-VMPC.

$$\left|\Delta u^{k+i-1}\right|^c \preceq \Delta u_{\max}, \quad i = 1, \ldots, N \tag{15c}$$

$$u_{\min} \preceq u^{k+i-1} \preceq u_{\max}, \quad i = 1, \ldots, N \tag{15d}$$

where $A_{k-1}^{\text{NN}} := (\partial f_{\text{NN}})/(\partial x)|_{\substack{x=x^{k-1},\\ u=\delta u^{k-1}}}$, and $B_{k-1}^{\text{NN}} := (\partial f_{\text{NN}})/(\partial u)|_{\substack{x=x^{k-1},\\ u=\delta u^{k-1}}}$.

C. Control Scheme Summary

The design procedure can be found in Fig. 4. Essentially, the control scheme consists of three steps.

1) Model the traffic system as a nonlinear system using a deep neural network.
2) Linearize the nonlinear system by applying VL along the operating trajectory.
3) Apply MPC to the linearized system.

From the viewpoint of gain-scheduling, step 2 can be considered as a step for computing scheduling variables, and step 3 can be viewed as selecting a corresponding linear controller based on the described scheduling variables.

Remark 2: Since VL adopts $\Delta u^k := \delta u^k - \delta u^{k-1} = u^k - u^{k-1}$ as the control inputs, the constant term u will not be a feedforward part of our proposed control strategy.

IV. Baseline Methods and Ablation Study

A. Baseline Methods

1) Max-Pressure Algorithm: Unlike our setting, the max-pressure traffic signal control method does not have a fixed cycle length [24]. Instead, this algorithm updates the phase duration of each traffic light at a fixed time interval (35 s in this study) between the two phases. If the new phase duration is different from the current one, then a yellow phase is enforced for a predefined duration (4 s in this study).

Inspired by the max-pressure algorithm in the field of wireless networks [32], the max-pressure traffic signal control method updates the phases according to the "pressure" of each traffic signal phase φ, defined as

$$\text{Pressure}(\varphi) = \sum_{l \in L_{\varphi,\text{up}}} |q_l| - \sum_{l \in L_{\varphi,\text{down}}} |q_l| \tag{16}$$

where l is a traffic lane; q_l is the average vehicle queue length in lane l during the last updating interval; $L_{\varphi,\text{up}}$ is the set of upstream lanes controlled by phase φ, and $L_{\varphi,\text{down}}$ is the set of downstream lanes controlled by phase φ. The phase with the greatest pressure is selected as the next phase. Intuitively, the pressure measures the nonuniformity of vehicle distribution by the difference between the upstream and downstream queue lengths. To effectively mitigate such nonuniformity, the max-pressure algorithm aims to release the phase that has the highest pressure.

2) Independent Deep Q-Network: Like the max-pressure algorithm, this method updates the phase of each traffic light once every 35 s, and a 4-s yellow phase is enforced if the new phase is different from the current enabled one. The algorithm we implemented is based on one from the work of Ault and Sharon [33], in which each intersection has a local deep Q-network agent, whose input is the current enabled phase and the current state, and whose output is the next phase. The state, action, reward, and Q-network architecture are defined as follows.

1) *State:* 1) Number of approaching vehicles; 2) stopped vehicles' accumulated waiting time; 3) number of

stopped vehicles; and 4) average speed of approaching vehicles.

2) *Action:* The phase to be enabled.
3) *Reward:* Minus stopped vehicles accumulated waiting time.
4) *Q-Network:* Input is resized as a 3-D tensor. The dimensions of each dimension are 1, the number of coming lanes connected to the intersection is 5, respectively, in which four components of the last dimension store the state information, and the other component stores the current phase information as a one-hot vector. The input layer is a 2-D convolution layer whose kernel size is $(2, 2)$, stride is 1, and the number of output channels is 64. After being flattened, the output of the input layer is transmitted to two hidden layers, each with 64 dimensions. The output layer's dimension is the number of intersection's phases. The activation function of the input and hidden layers is Relu.

B. Ablation Study

To further assess the performance of the proposed control scheme, an ablation study was conducted. Two variations were considered in this study: replacing the deep neural network with a linear model identified by offline LS [34] and replacing the VL method with the linearization along a nominal system trajectory. These two variants are conducted for step 1 and step 2 in Section III-C, respectively.

1) Offline LS Model: According to Diakaki et al. [22], the traffic system can be identified as a linear time-invariant system

$$x^{k+1} = Ax^k + B\delta u^k \tag{17}$$

where the system matrices A and B can be estimated using the well-known LS from historical data offline. Equation (17) is referred to as the "LS model" for convenience. Two controllers can be designed based on this model.

1) Set the cost function as

$$J = \sum_{k=1}^{\infty} \left(x^k\right)^{\mathrm{T}} Q x^k + \left(\delta u^k\right)^{\mathrm{T}} R \delta u^k \tag{18}$$

where the matrices $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are the prespecified symmetric and positive definite matrices. The following LQR controller can be designed for this linear model if (17) is stabilizable

$$\delta u_k = -K x^k \tag{19}$$

where $K = (B^T P B + R)^{-1} B^T P A$, and matrix P is obtained by solving the following Riccati equation:

$$A^T P A - P - A^T P B\left(B^T P B + R\right)^{-1} B^T P A + Q = 0. \tag{20}$$

This controller is referred to as "LS-LQR" for convenience.

2) Adopt VL to (17)

$$\begin{bmatrix} x^{k+1} \\ \Delta x^{k+1} \end{bmatrix} = \begin{bmatrix} I & A \\ 0 & A \end{bmatrix} \begin{bmatrix} x^k \\ \Delta x^k \end{bmatrix} + \begin{bmatrix} B \\ B \end{bmatrix} \Delta u^k. \tag{21}$$

Then, a VMPC controller can be formulated for the given linear time-invariant system. This controller is referred to as "LS-VMPC" for convenience.

2) Linearization Along the Nominal System Trajectory: This linearization method is motivated by the observation that there is a natural equilibrium point for the traffic system $f(0, 0) = 0$. This equilibrium point suggests that if there are no vehicles in the area at time step k and we adopt fixed-time controller u, then in the next time step $k + 1$, the number of vehicles in this area should be 0 [22]. Thus, using the definition of the Fréchet derivative, we can see that

$$f(0,0) = f\left(x^k, \delta u^k\right) + Df\left(x^k, \delta u^k\right)\left\{0 - \left[\left(x^k\right)^{\mathrm{T}}, \left(\delta u^k\right)^{\mathrm{T}}\right]^{\mathrm{T}}\right\} + o\left(\left\|\left[\left(x^k\right)^{\mathrm{T}}, \left(\delta u^k\right)^{\mathrm{T}}\right]\right\|_2\right) \tag{22}$$

where $Df(x^k, \delta u^k)$ denotes the Fréchet derivative of f at $(x^k, \delta u^k)$. Using [29, Th. 1.2], it can be shown that

$$f(0,0) = f\left(x^k, \delta u^k\right) + \left[\left.\frac{\partial f}{\partial x}\right|_{\substack{x=x^k,\\ u=\delta u^k}} \left(0 - x^k\right)\right] + \left[\left.\frac{\partial f}{\partial u}\right|_{\substack{x=x^k,\\ u=\delta u^k}} \left(0 - \delta u^k\right)\right] + o\left(\left\|\left[\left(x^k\right)^{\mathrm{T}}, \left(\delta u^k\right)^{\mathrm{T}}\right]\right\|_2\right). \tag{23}$$

We can rearrange (23) to further obtain the following:

$$f\left(x^k, \delta u^k\right) = \left.\frac{\partial f}{\partial x}\right|_{\substack{x=x^k,\\ u=\delta u^k}} x^k + \left.\frac{\partial f}{\partial u}\right|_{\substack{x=x^k,\\ u=\delta u^k}} \delta u^k + o\left(\left\|\left[\left(x^k\right)^{\mathrm{T}}, \left(\delta u^k\right)^{\mathrm{T}}\right]\right\|_2\right). \tag{24}$$

For easy reference, this linearization method is referred to as "nominal linearization (NL)."

The linearized LS model via NL is the same as the LS model itself. Thus, we only consider NL for the DL model. Since the first-order derivatives of f are locally bounded, we can replace $(\partial f_{\mathrm{NN}})/(\partial x)|_{\substack{x=x^k,\\ u=\delta u^k}}$ and $(\partial f_{\mathrm{NN}})/(\partial u)|_{\substack{x=x^k,\\ u=\delta u^k}}$ with $(\partial f_{\mathrm{NN}})/(\partial x)|_{\substack{x=x^k,\\ u=0}}$ and $(\partial f_{\mathrm{NN}})/(\partial u)|_{\substack{x=x^k,\\ u=0}}$, respectively, if δu^k is small. The prediction obtained by NL of the DL model can be defined as

$$\hat{x}_{\mathrm{NL}}^{k+1} := \left.\frac{\partial f_{\mathrm{NN}}}{\partial x}\right|_{\substack{x=x^k,\\ u=0}} x^k + \left.\frac{\partial f_{\mathrm{NN}}}{\partial u}\right|_{\substack{x=x^k,\\ u=0}} \delta u^k. \tag{25}$$

An MPC controller can be designed for the linear time-varying system

$$x^{k+1} = \left.\frac{\partial f_{\mathrm{NN}}}{\partial x}\right|_{\substack{x=x^k,\\ u=0}} x^k + \left.\frac{\partial f_{\mathrm{NN}}}{\partial u}\right|_{\substack{x=x^k,\\ u=0}} \delta u^k. \tag{26}$$

We formulate such an MPC controller as

$$\min_{u^k,\ldots,u^{k+N-1}} \sum_{i=1}^{N} \left(\left(x^{k+i}\right)^{\mathrm{T}} Q x^{k+i} + \left(\delta u^{k+i-1}\right)^{\mathrm{T}} R \delta u^{k+i-1}\right) \tag{27a}$$

$$\text{s.t. } x^{k+i} = \left.\frac{\partial f_{\mathrm{NN}}}{\partial x}\right|_{\substack{x=x^{k+i-1},\\ u=0}} x^k + \left.\frac{\partial f_{\mathrm{NN}}}{\partial u}\right|_{\substack{x=x^{k+i-1},\\ u=0}} \delta u^k, \quad i = 1, \ldots, N \tag{27b}$$

$$\left|\Delta u^{k+i-1}\right|^c \preceq \Delta u_{\max}, \quad i = 1, \ldots, N \tag{27c}$$

$$u_{\min} \preceq u^{k+i-1} \preceq u_{\max}, \quad i = 1, \ldots, N. \tag{27d}$$

This controller is referred to as "DL-NMPC."

Remark 3: It can be seen that the proposed algorithm does not use a tractable adaptive fuzzy classifier since it is based on the neural network and the task is prediction rather than classification. In summary, the logical procedures of the proposed algorithm are as follows.

1) *Step 1:* Generate the dataset by applying the real-world traffic demands and random traffic signals to a simulation platform and sampling the number of vehicles on each link at each time step.
2) *Step 2:* For system identification, identify the traffic system via DL by minimizing the neural network's prediction error for the number of vehicles on each link at the next time step. The inputs of the neural network are the current number of vehicles on each link and the current traffic signals. The trained neural network model is stored for control signal calculation.
3) *Step 3:* For control, at each time step, linearize the trained neural network around the operating point at the previous time step, which can be regarded as the scheduling variable for the current time step.
4) *Step 4:* At each time step, design the MPC controller for the linearized system to generate the optimal variants of the traffic signals subjected to constraints and apply them to the controlled traffic system. Save the applied control signals and current number of vehicles on each link as the scheduling variable for the next time step.

V. Numerical Experiment

A. Simulation of Urban Mobility Simulation

Simulation of urban mobility (SUMO) [35] is an open-source, space-continuous, and time-discrete traffic flow simulation platform developed in 2001. In the past two decades, SUMO has been developed into a simulation platform mainly used in large-scale traffic network simulation, which integrates network generation, demand generation, and simulation.

1) Network Generation: SUMO uses graphs to represent real-world traffic networks, where nodes represent intersections and edges represent links. The traffic lights are contained in the nodes to set right-of-way rules. An edge connecting two different nodes consists of a fixed number of lanes that contain information about the vehicle classes and maximum speeds allowed.

The graph based on a specific real-world network can be generated by the road importer *netconvert*, which can convert networks from other traffic simulators. As shown in Fig. 5, we converted a part of the traffic network in downtown Chattanooga, Tennessee, from OpenStreetMap [36] into SUMO using *netconvert*. This network contains 38 links and 11 intersections, which consist of a total of 15 controlled phases at 140-s cycle length. The predetermined control inputs for the fixed-timed controller are $u = [98, 107, 28, 55, 34, 99, 102, 28, 96, 115, 36, 57, 15, 47, 36]$.

2) Demand Generation: Typically, the traffic demands in SUMO can be generated using one of the three embedded functions: *od2trips*, *jtrrouter*, and *dfrouter*. Among these functions, *jtrrouter* is often adopted for a part of a city's road network with a few intersections, which defines the turn percentages at each intersection to compute the routes for vehicles based on the traffic flow data. Therefore, we adopted this function to generate routes based on real-world traffic flow data for this region from 1 October to 15 October and from 23 October to 31 October 2020. The data were collected from 7 A.M. to 7 P.M. each day.

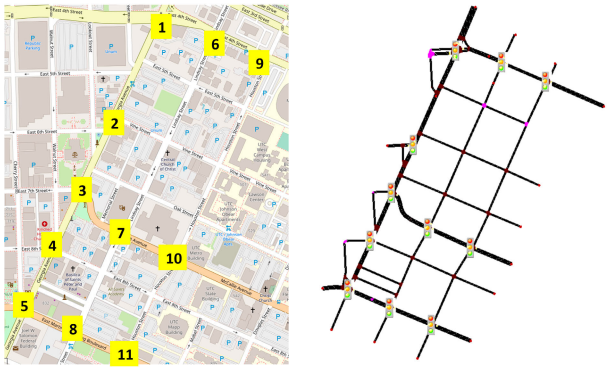

Fig. 5. Downtown Chattanooga model from (left) OpenStreetMap and (right) SUMO simulation network generated using the *netconvert* module.

3) Simulation: The SUMO simulation is discrete-time, and we set the step length to be 1 s. The simulation model is space-continuous, and the positions of vehicles are determined by SUMO's built-in car-following models [37]. Such models are designed to compute the speed of each vehicle by investigating a vehicle's own speed, the leading vehicle's speed, and the distance between the two vehicles.

B. Training Process

According to Fig. 5, the dimension of the input layer is 53, and the dimension of the output layer is 38. We set the dimensions of the three hidden layers as 54, 60, and 100, respectively. A total of 5200 instances were sampled from SUMO as a dataset with the sampling interval being 140 s, using the random control inputs described in Section III-A. We divided the dataset into a training set (4800 instances), a validation set (200 instances), and a test set (200 instances). The optimizer was Adam. We adopted a multistep learning rate: 0.005, 0.0025, and 0.0005 each for 16, 8, and 8 epochs, and a multistep batch size: 32, 48, and 64 each for 16, 8, and 8 epochs. For the experiments, we have used Alienware-17-R4 with an Intel i9-8950HK CPU at 2.90 GHz, 32-GB memory, and a single GeForce GTX1080 GPU. The training took approximately 90 min, and the computation was mainly conducted on the GPU.

For kernel distribution estimation, we set $h = 0.3$, and $\phi(y) = 1/((2\pi)^{1/2})e^{-(1/2)y^2}$. For entropy loss, we set $g(y) = 1/(0.01 \times (2\pi)^{1/2})e^{-(1/2)(y/0.01)^2}$, $-a = b = 20$, $\Delta y = 0.05$, and $c = 1.9$.

Fig. 6 shows the training loss and validation loss curves during the training process. The training losses are plotted once every five iterations, and the validation loss are plotted after every epoch, with the green (yellow) line representing the varying of the l_2 loss [the second term on the right-hand

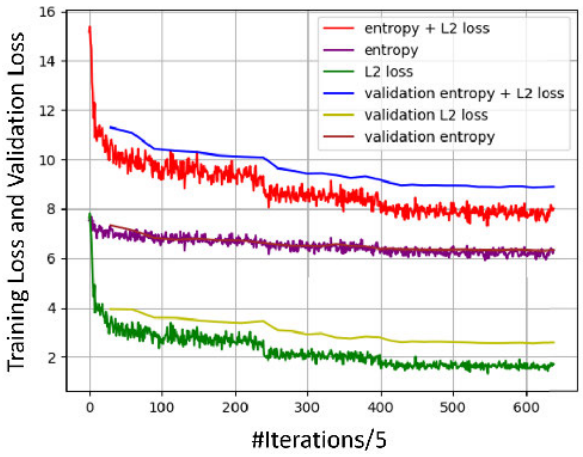

Fig. 6. Training process of the neural network. The training losses are plotted every five iterations, and the validation losses are plotted after every epoch.

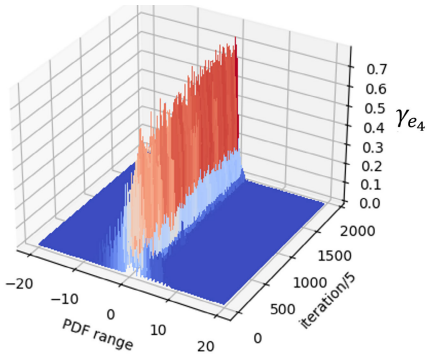

Fig. 7. Varying PDF of e_4.

side of (14)], the purple (brown) line representing the varying of the entropy of the error [the first term on the right-hand side of (14)], and the red (blue) line representing the varying of the entropy loss [the left-hand side of (14)] on the training (validation) set, respectively.

Fig. 7 shows that the error's PDF becomes more and more similar to the target distribution $g(y)$ as the training process progresses.

C. Comparison With Baseline

Fig. 8 presents the control performance of the proposed method against the max-pressure and IDQN methods. We tested the algorithms on the real-world traffic flow data collected from 7 A.M. to 7 P.M., 1 October to 5 October 2020. It can be seen that the proposed method outperformed both the baseline algorithms by reducing the mean value of the number of vehicles in the area by 19.8% when compared with the max-pressure algorithm, and by 35.5% when compared with the IDQN algorithm. Furthermore, the proposed DL-VMPC algorithm performed better than both the baseline methods especially when the traffic demands are high, which is desirable for traffic signal control.

D. Ablation Study

The prediction error generated by the LS and DL models on the test set was compared. Fig. 9 and Table I show that

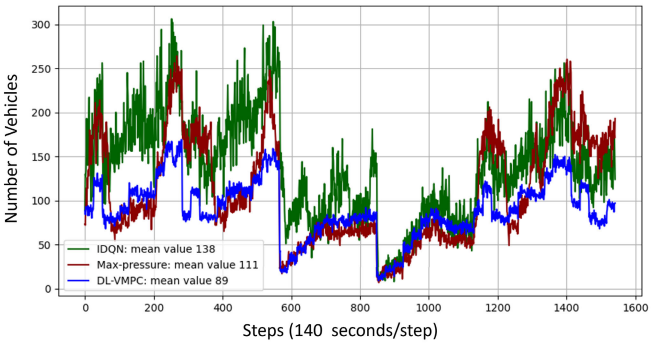

Fig. 8. Control performance comparison with the max-pressure algorithm and IDQN.

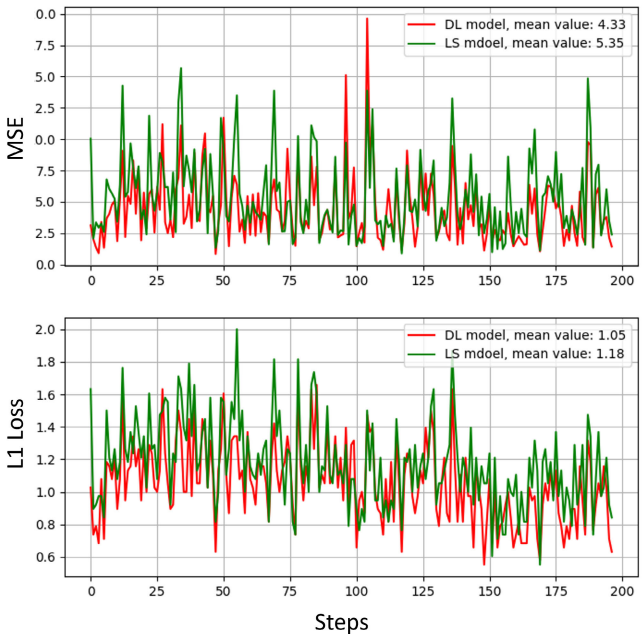

Fig. 9. Prediction error for test set.

TABLE I
PREDICTION ERROR FOR TEST SET

	LS Model	DL Model
MSE[a]	5.35	**4.33**
L1 Norm[b]	1.18	**1.05**
MAPE[c]	26.20%	**23.44%**

[a] MSE $:= \frac{1}{n \times b_{\text{test}}} \sum_{i=1}^{b_{\text{test}}} \|\hat{x}^{k_i+1} - x^{k_i+1}\|_2$.
[b] L1 Norm $:= \frac{1}{n \times b_{\text{test}}} \sum_{i=1}^{b^{\text{test}}} \|\hat{x}^{k_i+1} - x^{k_i+1}\|_1$.
[c] MAPE $:= \frac{1}{b_{\text{test}}} \sum_{i=1}^{b_{\text{test}}} \frac{\|\hat{x}^{k_i+1} - x^{k_i+1}\|_2}{\|x^{k_i+1}\|_2}$.
(b_{test} stands for number of data, $\hat{x}^{k_i+1}$ stands for prediction. The best result for each performance index is boldfaced.)

in the test set, the DL model provides better prediction than the LS model. Although LS can be performed within 1 s on the same device as the DL model was trained on, those training processes can be realized offline using historical data according to Fig. 4, for which the much longer training time of the DL model will not affect the application of our proposed method.

However, the control input difference δu^k of the test set (see Section III-A) is much smaller than real-world controllers'

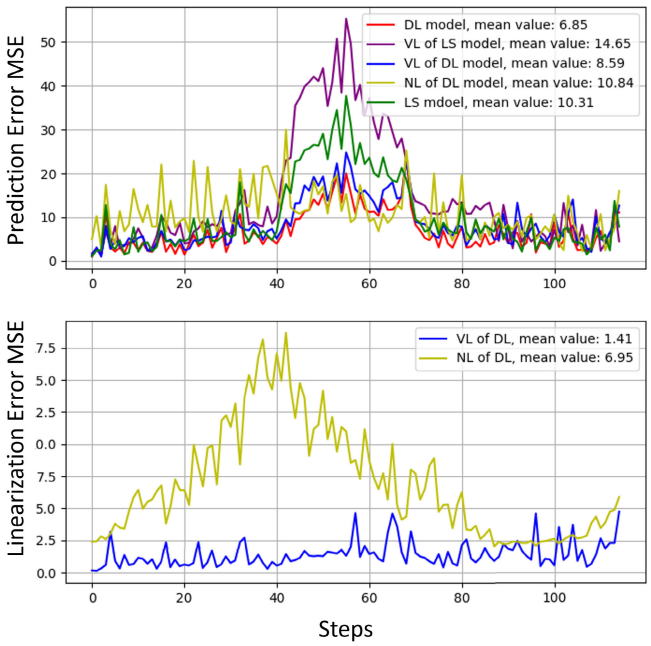

Fig. 10. (Top) Prediction error for real-scale input dataset using (4). (Bottom) Linearization error comparison between (4) and NL.

TABLE II
PREDICTION ERROR FOR REAL-SCALE INPUTS

	LS	DL	VL of LS	NL of DL	VL of DL
MSE	10.31	6.85	8.10	10.84	**6.63**
L1 Norm	1.68	1.45	1.25	1.84	**1.19**
MAPE	28.84%	**24.52%**	26.91%	32.05%	24.55%

control input differences. Thus, we further tested the prediction error on the dataset generated by "real-scale" inputs $\{\delta u^k = \lfloor \delta\hat{u}^k + (1/2) \rfloor | \delta\hat{u}^k = 0.1u \times \mathcal{U} + 0.4\sin((\pi k)/100), \mathcal{U} \sim \text{Uniform}[-0.5, 0.5]\}_{k=1}^{120}$. The prediction was obtained from the LS and DL models, and from the other four models derived by linearizing the given two models with two different linearization methods, NL (see Section IV-B) and VL (see Section II-C).

For the real-scale input dataset, as shown in Fig. 10, it can be seen that the DL methods provided the best prediction, which reflects the advantage of using the nonlinear model to predict the traffic. Compared with NL, velocity-based linearization using (4) approximated the DL model better, which reflects the advantage of this method being able to linearize nonlinear system models at any operating point. Furthermore, the DL model provides better prediction than that of the LS model for the models themselves and their velocity-based linearized models.

The comparison shown in Fig. 10 is only given to demonstrate the advantage of using a nonlinear model and VL for traffic systems. For the controller design, we can use (6), which replaces $f(x^{k-1}, \delta u^{k-1})$ with x^k. As shown in Fig. 11 and Table II, we can conclude that using this equation, we can further improve the prediction of the VL of the DL model at some operating points.

Next, we compared the proposed controller with the other three model-based controllers introduced in Section IV-B. All

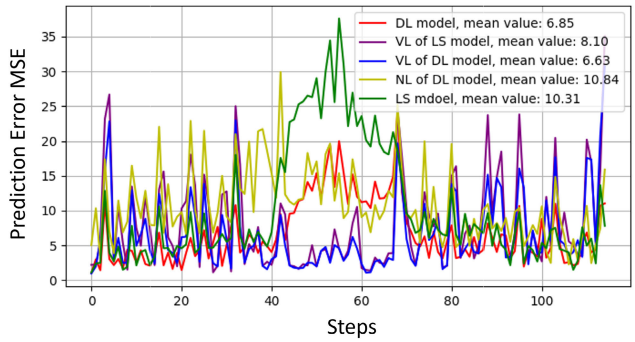

Fig. 11. Prediction error for real-scale input dataset using (6).

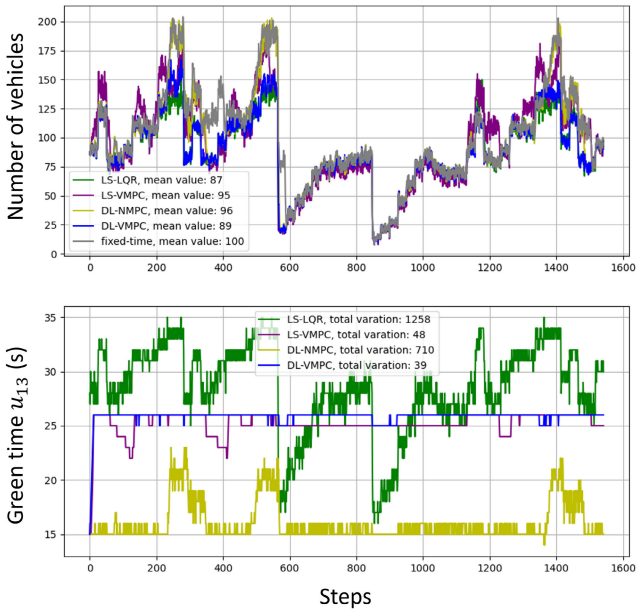

Fig. 12. (Top) Control performance, and (bottom) 13th component of control signals.

the control signals were rounded to an integer before being applied to the SUMO simulation, and the parameters were set to achieve their best performance.

Fig. 12 indicates that even though LS-LQR has the best control effectiveness, its control signals vary rapidly, which is not desirable in real-world applications. To achieve effective control, the control signals of LS-LQR significantly surpass the explicit constraints of model predictive controllers, which is unrealistic. LS-VMPC has slowly varying control signals, but the control effectiveness is worse than the proposed DL-VMPC controller especially when the traffic demands are high. DL-NMPC has neither slowly varying control signals nor good control effectiveness. However, the proposed controller can achieve almost as effective of control as LS-LQR with much smaller control signal variation, and at the same time the control signals strictly satisfy the predefined constraints.

Remark 4: The proposed controller requires full observability of the data. However, data from some traffic system intersections may not be available because those intersections do not have enough sensors in the real-world environment. Indeed, how to deal with the data's partial observability could be one direction of our future research.

VI. Stability Analysis

For the closed-loop system, to show a desired property that the change in the number of vehicles on each link will be bounded by the varying speed of the traffic signals, the BIBO stability property of the velocity-based system introduced in Section III-B is analyzed

$$\Delta x^{k+1} = A_k^{\mathrm{NN}} \Delta x^k + B_k^{\mathrm{NN}} \Delta u^k. \tag{28}$$

First, we recall a few basic results from the past literature, especially from Desoer [38].

Assumption 1: The sequence of matrices $\{A_k^{\mathrm{NN}}\}_{k\in\mathbb{N}_+}$ is uniformly bounded

$$\sup_{k\in\mathbb{N}_+} \left\| A_k^{\mathrm{NN}} \right\|_2 = a_M < \infty \tag{29}$$

and $\exists \varepsilon > 0$ such that

$$\max_j \left\{ \left| \lambda_j \left(A_k^{\mathrm{NN}} \right) \right| \right\} \le 1 - 2\varepsilon \tag{30}$$

for all $k \in \mathbb{N}_+$, where $\lambda_j(A_k^{\mathrm{NN}})$ represents the jth eigenvalue of A_k^{NN}.

Lemma 1 [38]: If Assumption 1 holds, then

$$\left\| \left(A_k^{\mathrm{NN}} \right)^p \right\|_2 \le (1-\varepsilon)^{p+1} \frac{(1-\varepsilon) + (a_M)^{n-1}}{\varepsilon^n} \tag{31}$$

for all $k, p \in \mathbb{N}_+$.

Lemma 2 [38]: Under Assumption 1, define matrix

$$P_{k+1} = I + \sum_{p=1}^{\infty} \left[\left(A_k^{\mathrm{NN}} \right)^{\mathrm{T}} \right]^p \left(A_k^{\mathrm{NN}} \right)^p \tag{32}$$

and define function $V : \mathbb{N}_+ \times \mathbb{R}^n \to \mathbb{R}_+$ as

$$V(k, x) = x^T P_k x. \tag{33}$$

Then, we have

$$\left(A_k^{\mathrm{NN}} \right)^{\mathrm{T}} P_{k+1} A_k^{\mathrm{NN}} - P_{k+1} = -I \quad \forall k \in \mathbb{N}_+ \tag{34}$$

and

$$1 \le \| P_k \| \le M \quad \forall k \in \mathbb{N}_+ \tag{35}$$

where $M = 1/(1-(1-\varepsilon)^2) \max\{1, ((1-\varepsilon)^2 + (1-\varepsilon)(a_M)^{n-1})/\varepsilon^n\}^2$.

Thus,

$$\|x\|^2 \le V(k, x) \le M \|x\|^2 \quad \forall k \in \mathbb{N}_+. \tag{36}$$

Lemma 3 [38]: If Assumption 1 holds, and

$$\sup_{k\in\mathbb{N}_+} \left\| A_{k+1}^{\mathrm{NN}} - A_k^{\mathrm{NN}} \right\|_2 \le \frac{1-\eta}{2M^2 a_M} \tag{37}$$

for some $\eta \in (0, 1)$, then

$$\left(\Delta x^k \right)^{\mathrm{T}} \left[\left(A_k^{\mathrm{NN}} \right)^{\mathrm{T}} P_{k+1} A_k^{\mathrm{NN}} - P_k \right] \Delta x^k \le -\eta \left\| \Delta x^k \right\|^2 \tag{38}$$

for all $k \in \mathbb{N}_+$.

Assumption 2: The sequence of matrices $\{B_k^{\mathrm{NN}}\}_{k\in\mathbb{N}_+}$ is uniformly bounded

$$\sup_{k\in\mathbb{N}_+} \left\| B_k^{\mathrm{NN}} \right\|_2 = b_M < \infty \tag{39}$$

and $\exists \beta > 0$ such that

$$\sup_{k\in\mathbb{N}_+} \left\| \Delta u^k \right\| \le \beta. \tag{40}$$

Proposition 1: If Assumption 1, Assumption 2, and (37) hold for some $\eta \in (0, 1)$, then there exist $\rho \in (0, 1)$, $\lambda > 0$, and a $\mathcal{K}$-function α such that

$$\left\| \Delta x^{k+1} \right\| \le \rho^k \sqrt{M} \left\| \Delta x^1 \right\| + \alpha(\beta) \quad \forall k \in \mathbb{N}_+ \tag{41}$$

provided that $\|\Delta x^1\| < \infty$.

Proof: Direct computation yields

$$\begin{aligned} &V\left(k+1, \Delta x^{k+1}\right) - V\left(k, \Delta x^k\right) \\ &= \left(\Delta x^k\right)^{\mathrm{T}} \left[\left(A_k^{\mathrm{NN}} \right)^{\mathrm{T}} P_{k+1} A_k^{\mathrm{NN}} - P_k \right] \Delta x^k \\ &\quad + 2\left(\Delta u^k\right)^{\mathrm{T}} \left(B_k^{\mathrm{NN}} \right)^{\mathrm{T}} P_{k+1} A_k^{\mathrm{NN}} \Delta x^k \\ &\quad + \left(\Delta u^k\right)^{\mathrm{T}} \left(B_k^{\mathrm{NN}} \right)^{\mathrm{T}} P_{k+1} B_k^{\mathrm{NN}} \Delta u^k. \end{aligned} \tag{42}$$

By Assumptions 1 and 2, Lemma 3, and (35), we have

$$\begin{aligned} &V\left(k+1, \Delta x^{k+1}\right) - V\left(k, \Delta x^k\right) \\ &\quad \le -\eta \left\| \Delta x^k \right\|^2 + 2 a_M b_M M \beta \left\| \Delta x^k \right\| + b_M^2 M \beta^2. \end{aligned} \tag{43}$$

Letting $W_k := (V(k, \Delta x^k))^{1/2}$, with (36), we have

$$W_{k+1}^2 \le \left(1 - \frac{\eta}{M}\right) W_k^2 + 2 a_M b_M M \beta W_k + b_M^2 M \beta^2. \tag{44}$$

Because the right-hand side of (44) monotonically increases with respect to $W_k \in \mathbb{R}_+$, we can deduce that if $\exists \lambda > 0$ such that

$$\lambda^2 = \left(1 - \frac{\eta}{M}\right) \lambda^2 + 2 a_M b_M M \beta \lambda + b_M^2 M \beta^2 \tag{45}$$

then $W_{k+1} \le \lambda$, provided $W_k \le \lambda$. Such a λ can be obtained by solving (45)

$$\lambda = \frac{M b_M \beta}{\eta} \left(M a_M + \sqrt{M a_M + \eta} \right) > 0. \tag{46}$$

Thus, according to (44), we have

$$W_{k+1}^2 \le \rho^2 W_k^2 + \alpha_1'(\beta)^2 \tag{47}$$

provided that $W_k \le \lambda$, where $\rho := (1 - (\eta/M))^{1/2}$ and $\alpha_1'(\beta) := (2 a_M b_M M \beta \lambda + b_M^2 M \beta^2)^{1/2}$. Thus,

$$W_{k+1} \le \rho W_k + \alpha_1'(\beta) \tag{48}$$

for all $k \in \mathbb{N}_+$ such that $W_k \le \lambda$.

Also

$$\begin{aligned} W_{k+1}^2 &\le \left(1 - \frac{\eta}{M}\right) W_k^2 + 2 a_M b_M M \beta W_k + b_M^2 M \beta^2 \\ &\le W_k^2 \end{aligned} \tag{49}$$

provided that $W_k \ge \lambda$. Such $W_k \ge \lambda$ exists if and only if $W_1 \ge \lambda$. By (49), if $W_k \ge \lambda$, then $W_k \le W_1$. Thus, according to (44), we can derive

$$W_{k+1} \le \rho W_k + \alpha_2'(\beta) \tag{50}$$

where $\alpha_2'(\beta) := (2 a_M b_M M^{(3/2)} \beta \|\Delta x^1\| + b_M^2 M \beta^2)^{1/2}$, provided that $W_k \ge \lambda$.

Using (48) and (50), we have

$$W_{k+1} \le \rho W_k + \alpha'(\beta) \tag{51}$$

where $\alpha'(\beta) := \max\{\alpha_1'(\beta), \alpha_2'(\beta)\}$.

As a consequence

$$\begin{aligned}\left\|\Delta x^{k+1}\right\| \leq W_{k+1} &\leq \rho^k W_1 + \left(1+\rho+\cdots+\rho^{k-1}\right)\alpha'(\beta) \\ &< \rho^k W_1 + \frac{1}{1-\rho}\alpha'(\beta) \\ &\leq \rho^k\sqrt{M}\left\|\Delta x^1\right\| + \alpha(\beta) \end{aligned} \tag{52}$$

where $\alpha(\beta) := 1/(1-\rho)\alpha'(\beta)$.

The given proposition states that if the demands are slowly varying and the duration's variation in every traffic light phase between each successive two cycle length is bounded, then the varying number of vehicles on each link will be bounded. The same conclusion can also be derived by constructing the ISS-Lyapunov function discussed by Jiang and Wang [39]. ■

VII. Conclusion

In this article, a new approach to reducing traffic congestion is proposed based on DL and dynamical system techniques. The proposed algorithm first models the traffic system via DL as a nonlinear system and then applies a gain-scheduling method, DL-VMPC, to design an appropriate controller to solve the control task. Besides, a modeling error entropy loss inspired by the SDC theory was proposed for the training process. Tested on real-world traffic data, the SUMO simulation showed that the proposed controller can achieve favorable control effectiveness with slowly varying traffic signals. Furthermore, robustness in the sense of BIBO stability was assessed for the closed-loop traffic control system. The desired results have been obtained.

For future work, one can: 1) use a higher order approximation to system (1) and 2) apply PDF shaping-based learning to train the neural network, where weight tuning is performed by making the modeling error PDF follow the narrowest Gaussian PDF centered at zero.

Acknowledgment

This manuscript has been authored in part by UT-Battelle LLC under Contract DE-AC05-00OR22725 with the U.S. Department of Energy (DOE). The U.S. government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for U.S. government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan). The authors would like to thank the city of Chattanooga for providing sensor data. The authors gratefully appreciate the input and assistance received from members of the joint research team from the Oak Ridge National Laboratory, the National Renewable Energy Laboratory, and the University of Tennessee, Knoxville, especially Jibonananda Sanyal, Andy Berres, Joseph Severino, Juliette Ugirumurera, and Airton Gustavo Kohls.

References

[1] K. Triantis, S. Sarangi, D. Teodorović, and L. Razzolini, "Traffic congestion mitigation: Combining engineering and economic perspectives," *Transp. Planning Technol.*, vol. 34, no. 7, pp. 637–645, Oct. 2011.

[2] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, "Review of road traffic control strategies," *Proc. IEEE*, vol. 91, no. 12, pp. 2043–2067, Dec. 2003.

[3] L. B. de Oliveira and E. Camponogara, "Multi-agent model predictive control of signaling split in urban traffic networks," *Transp. Res. C, Emerg. Technol.*, vol. 18, no. 1, pp. 120–139, Feb. 2010.

[4] H. Wang et al., "Network-wide traffic signal control using bilinear system modeling and adaptive optimization," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 1, pp. 79–91, Jan. 2022.

[5] H. Wang, M. Zhu, W. Hong, C. Wang, G. Tao, and Y. Wang, "Optimizing signal timing control for large urban traffic networks using an adaptive linear quadratic regulator control strategy," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 333–343, Jan. 2022.

[6] J. Park et al., "Adaptive urban traffic signal control for multiple intersections: An LQR approach," in *Proc. IEEE 25th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2022, pp. 2240–2245.

[7] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. AAAI*, vol. 34, no. 1, 2020, pp. 1234–1241.

[8] H. Zheng, F. Lin, X. Feng, and Y. Chen, "A hybrid deep learning model with attention-based conv-LSTM networks for short-term traffic flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 11, pp. 6910–6920, Nov. 2021.

[9] A. B. Subramaniyan et al., "Hybrid recurrent neural network modeling for traffic delay prediction at signalized intersections along an urban arterial," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 1, pp. 1384–1394, Jan. 2022.

[10] W. Hong, G. Tao, H. Wang, and C. Wang, "Traffic signal control with adaptive online-learning scheme using multiple-model neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Feb. 9, 2022, doi: 10.1109/TNNLS.2022.3146811.

[11] D. J. Leith and W. E. Leithead, "Survey of gain-scheduling analysis and design," *Int. J. Control*, vol. 73, no. 11, pp. 1001–1025, 2000.

[12] D. J. Leith and W. E. Leithead, "Gain-scheduled and nonlinear systems: Dynamic analysis by velocity-based linearization families," *Int. J. Control*, vol. 70, no. 2, pp. 289–317, Jan. 1998.

[13] D. J. Leith and W. E. Leithead, "Gain-scheduled control: Relaxing slow variation requirements by velocity-based design," *J. Guid., Control, Dyn.*, vol. 23, no. 6, pp. 988–1000, Nov. 2000.

[14] J. S. R. Jang, C. T. Sun, and E. Mizutani, "Neuro-fuzzy and soft computing—A computational approach to learning and machine intelligence," *IEEE Trans. Autom. Control*, vol. 42, no. 10, pp. 1482–1484, Oct. 1997.

[15] B. Kosko, "Fuzzy systems as universal approximators," *IEEE Trans. Comput.*, vol. 43, no. 11, pp. 1329–1333, Nov. 1994.

[16] X. Xie, C. Wei, Z. Gu, and K. Shi, "Relaxed resilient fuzzy stabilization of discrete-time Takagi–Sugeno systems via a higher order time-variant balanced matrix method," *IEEE Trans. Fuzzy Syst.*, vol. 30, no. 11, pp. 5044–5050, Nov. 2022.

[17] X. Xie, J. Lu, and D. Yue, "Resilient stabilization of discrete-time Takagi–Sugeno fuzzy systems: Dynamic trade-off between conservatism and complexity," *Inf. Sci.*, vol. 582, pp. 181–197, Jan. 2022.

[18] G. Pannocchia and J. B. Rawlings, "The velocity algorithm LQR: A survey," Dept. Chem. Eng., Univ. Wisconsin-Madison, Madison, WI, USA, Tech. Rep. 2001-01, 2001.

[19] P. S. G. Cisneros and H. Werner, "A velocity algorithm for nonlinear model predictive control," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 3, pp. 1310–1315, May 2021.

[20] Q. Wang et al., "Deploying a model predictive traffic signal control algorithm—A field deployment experiment case study," in *Proc. IEEE 25th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2022, pp. 3564–3570.

[21] H. Wang, *Bounded Dynamic Stochastic Systems: Modelling and Control*. London, U.K.: Springer-Verlag, 2000.

[22] C. Diakaki, M. Papageorgiou, and K. Aboudolas, "A multivariable regulator approach to traffic-responsive network-wide signal control," *Control Eng. Pract.*, vol. 10, no. 2, pp. 183–195, Feb. 2002.

[23] K. Aboudolas, M. Papageorgiou, A. Kouvelas, and E. Kosmatopoulos, "A rolling-horizon quadratic-programming approach to the signal control problem in large-scale congested urban road networks," *Transp. Res. C, Emerg. Technol.*, vol. 18, no. 5, pp. 680–694, Oct. 2010.

[24] P. Varaiya, "Max pressure control of a network of signalized intersections," *Transp. Res. C, Emerg. Technol.*, vol. 36, pp. 177–195, Nov. 2013.
[25] R. Allsop, "SIGSET: A computer program for calculating traffic signal settings," *Traffic Eng. Control*, vol. 12, no. 2, pp. 58–60, 1971.
[26] R. E. Allsop, "SIGCAP: A computer program for assessing the traffic capacity of signal-controlled road junctions," *Traffic Eng. Control*, vol. 17, pp. 338–341, Aug. 1976.
[27] P. Kidger and T. Lyons, "Universal approximation with deep narrow networks," in *Proc. Conf. Learn. Theory*, 2020, pp. 2306–2327.
[28] A. Wang and H. Wang, "Survey on stochastic distribution systems: A full probability density function control theory with potential applications," *Optim. Control Appl. Methods*, vol. 42, no. 6, pp. 1812–1839, Nov. 2021.
[29] A. Avez, *Differential Calculus*. New York, NY, USA: Dover, 2020.
[30] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," 2015, *arXiv:1511.07289*.
[31] R. A. Davis, K.-S. Lii, and D. N. Politis, "Remarks on some nonparametric estimates of a density function," in *Selected Works of Murray Rosenblatt*. New York, NY, USA: Springer, 2011, pp. 95–100.
[32] J. G. Dai and W. Lin, "Maximum pressure policies in stochastic processing networks," *Oper. Res.*, vol. 53, no. 2, pp. 197–218, 2005.
[33] J. Ault and G. Sharon, "Reinforcement learning benchmarks for traffic signal control," in *Proc. 35th Conf. Neural Inf. Process. Syst. Datasets Benchmarks Track (Round 1)*, 2021, pp. 1–11.
[34] G. C. Goodwin and K. S. Sin, *Adaptive Filtering Prediction and Control*. Chelmsford, MA, USA: Courier Corporation, 2014.
[35] P. A. Lopez et al., "Microscopic traffic simulation using SUMO," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2575–2582.
[36] M. Haklay and P. Weber, "OpenStreetMap: User-generated street maps," *IEEE Pervasive Comput.*, vol. 7, no. 4, pp. 12–18, Oct. 2008.
[37] Q. Sun et al., "Research on optimization and evaluation method of the car following model based on SUMO application test scenario," in *Proc. IEEE Intell. Vehicles Symp. Workshops (IV Workshops)*, Jul. 2021, pp. 102–107.
[38] C. A. Desoer, "Slowly varying discrete system $x_{i+1} = A_i x_i$," *Electron. Lett.*, vol. 6, no. 11, pp. 339–340, May 1970.
[39] Z.-P. Jiang and Y. Wang, "Input-to-state stability for discrete-time nonlinear systems," *Automatica*, vol. 37, no. 6, pp. 857–869, Jun. 2001.

Zhun Yin received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2018, and the M.S. degree from the Department of Electrical and Computer Engineering, Tandon School of Engineering, New York University, Brooklyn, New York, NY, USA, in 2022, where he is currently pursuing the Ph.D. degree with the Control and Networks Laboratory.

His research interests include deep learning, gain-scheduling control, and traffic signal control.

Tong Liu (Graduate Student Member, IEEE) received the B.S. and M.S. degrees from Beijing Jiaotong University, Beijing, China, in 2017 and 2020, respectively. He is currently pursuing the Ph.D. degree with the Control and Networks Laboratory, Department of Electrical and Computer Engineering, Tandon School of Engineering, New York University, Brooklyn, New York, NY, USA.

His research interests include adaptive dynamic programming, cooperative control of connected vehicles, and adaptive traffic signal control.

Chieh (Ross) Wang (Member, IEEE) received the B.S. and M.S. degrees in civil engineering from the National Taiwan University, New Taipei, Taiwan, in 2005 and 2007, respectively, and the Ph.D. degree in civil and environmental engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2017.

He is currently a Research and Development Associate Staff Member with the Buildings and Transportation Science Division, Oak Ridge National Laboratory, Oak Ridge, TN, USA. His research interests include transportation system modeling and simulation, data analytics and visualization, smart mobility, and infrastructure management.

Hong Wang (Fellow, IEEE) received the M.S. and Ph.D. degrees from the Huazhong University of Science and Technology, Wuhan, China, in 1984 and 1987, respectively.

He was a Research Fellow with Salford University, Salford, U.K., Brunel University, London, U.K., and Southampton University, Southampton, U.K. He was with the University of Manchester Institute of Science and Technology (UMIST), Manchester, U.K., in 1992. He was a Chair Professor in process control from 2002 to 2016, and he was the Deputy Head of the Paper Science Department and the Director of the UMIST Control Systems Centre (which was established in 1966 and is the birthplace of modern control theory) from 2004 to 2007. He was a Member of the University Senate and General Assembly. From 2016 to 2018, he was with the Pacific Northwest National Laboratory as a Lab Fellow and Chief Scientist and was the Co-Leader for the Control of Complex Systems. He joined the Oak Ridge National Laboratory, Oak Ridge, TN, USA in January 2019 and is also an Emeritus Professor with the University of Manchester, Manchester. His research interests focus on stochastic distribution control, fault diagnosis and tolerant control, and intelligent controls with applications to transportation systems.

Dr. Wang is a member of three IFAC committees. He was an Associate Editor for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, the IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, and the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING.

Zhong-Ping Jiang (Fellow, IEEE) received the M.Sc. degree in statistics from the University of Paris XI, Paris France, in 1989, and the Ph.D. degree in automatic control and mathematics from ParisTech-Mines, Paris, in 1993, under the direction of Prof. Laurent Praly.

He is currently a Professor of electrical and computer engineering with the Tandon School of Engineering, New York University, New York, NY, USA. His main research interests include stability theory, robust/adaptive/distributed nonlinear control, robust adaptive dynamic programming, reinforcement learning, and their applications to information, mechanical, and biological systems. In these fields, he has written six books and is the author/coauthor of more than 500 peer-reviewed journal and conference papers.

Prof. Jiang is a fellow of IFAC, CAA, and AAIA, a Foreign Member of the Academia Europaea (Academy of Europe), and is among the Clarivate Analytics Highly Cited Researchers. In 2022, he received the Excellence in Research Award from the NYU Tandon School of Engineering. He was a recipient of the prestigious Queen Elizabeth II Fellowship Award from the Australian Research Council, the CAREER Award from the U.S. National Science Foundation, the JSPS Invitation Fellowship from the Japan Society for the Promotion of Science, the Distinguished Overseas Chinese Scholar Award from the NSF of China, and several best paper awards. He has served as Deputy Editor-in-Chief, Senior Editor, and Associate Editor for numerous journals.