Hybrid Recurrent Neural Network Modeling for Traffic Delay Prediction at Signalized Intersections Along an Urban Arterial

Arun Bala Subramaniyan, Chieh Wang, *Member, IEEE*, Yunli Shao, *Member, IEEE*, Wan Li, Hong Wang, *Senior Member, IEEE*, Guohui Zhang, and Tianwei Ma

***Abstract*— This paper studies the traffic delay prediction modeling for multiple signalized intersections along the Ala Moana Boulevard and Nimitz Highway in Hawaii. Several machine learning (ML) based approaches have been studied in the literature, and most of them focused on prediction accuracy rather than the end use of real-time control and implementation. These ML models tend to be very complex and non-linear in nature, making it challenging to achieve fast inferences and are computationally heavy for real-time signal control implementation. In this paper, a simple yet accurate hybrid modeling method is proposed to predict traffic delay one-step ahead with the model made suitable for real-time implementation to control traffic flow. Since real-time road-side measurements are recorded in unstructured form, the paper also discusses other issues related to data extraction and the pre-processing process. Finally, a simple signal control loop is developed to demonstrate the proposed modeling approach, which has shown advantages in model accuracy and computation efficiency compared against several existing modeling methods.**

***Index Terms*— Traffic prediction, hybrid neural networks, recurrent neural networks, signalized intersection, AI-based signal control.**

I. Introduction

ADVANCES in data acquisition systems and wireless communications, such as 5G, have greatly facilitated high-frequency data sharing across roadway networks (both highways and urban streets), bringing the Vehicle-to-Everything (V2X, such as vehicle-to-vehicle and vehicle-to-infrastructure) concept to reality. The information from V2X communication systems can be combined with traffic flow models to help design traffic control decisions to effectively balance traffic congestion across complex intersections and facilitate smoother traffic flow. Improving traffic safety and operational efficiency lies at the heart of traffic flow modeling, which lays the groundwork for emerging smart mobility and intelligent traffic control system operations. In addition, traffic demand and traffic flow can be significantly influenced by uncertainties and randomness in traffic participants' behaviors and environmental factors such as weather, time of day, incidents, events, constructions, thereby making the system a multi-input and multi-output (MIMO) stochastic dynamic system. To capture the stochasticity of MIMO traffic flow systems, various traffic flow models (e.g., travel delay models, vehicle speed prediction models, energy consumption models) have been proposed and tested against road-side sensor data [1], [2], [3], [4]. These methods utilized only a single data source (univariate), and the model predictions can be improved using multivariate models. Some of the well-known modeling methods include time series regression, AutoRegressive Integrated Moving Average (ARIMA) models, artificial neural networks (ANNs), multi-layer perceptrons (MLPs), convolutional neural networks (CNNs), etc. [1], [2], [3], [4], [5], [6]. Most of the models worked well for simple use cases but failed to capture the nonlinear spatio-temporal correlations of traffic data. In such cases, recurrent neural networks (RNNs) and long short-term memory (LSTM) [7], [8] applications have been demonstrated as advantageous in capturing nonlinear spatial and temporal dependencies of traffic features. In addition, graph neural networks (GNNs) have also been shown to be powerful for large-scale traffic systems, since GNNs can extract features from graph-structured data and predict future traffic states in an efficient and effective manner with existing high-performance computing resources [9].

In addition to finding the best model representing the stochastic nature of traffic flow systems, one of the important criteria is the reliability and confidence in the developed models for real-time signal control implementation. Although the accuracy of the above-mentioned modeling approaches was sufficient in predicting the response of the variable of interest,

Manuscript received 30 November 2021; revised 7 June 2022; accepted 27 July 2022. This work was supported by the U.S. Department of Energy (U.S DOE), Office of Energy Efficiency and Renewable Energy, Vehicle Technologies Office's funding DE-LC-000L084, and in part by the National Science Foundation Major Research Instrumentation award 1920304. This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the U.S. DOE. The US government retains and the publisher, by accepting the article for publication, acknowledges that the US 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 US 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 Associate Editor for this article was S. Olariu. *(Corresponding author: Chieh Wang.)*

Arun Bala Subramaniyan and Guohui Zhang are with the Department of Civil and Environmental Engineering, University of Hawaii at Mānoa, Honolulu, HI 96822 USA (e-mail: arunbs@hawaii.edu; guohui@hawaii.edu).

Chieh Wang, Yunli Shao, Wan Li, and Hong Wang are with the Oak Ridge National Laboratory, Buildings and Transportation Science Division, Oak Ridge, TN 37831 USA (e-mail: cwang@ornl.gov; shaoy@ornl.gov; liw2@ornl.gov; wangh6@ornl.gov).

Tianwei Ma is with the College of Engineering, Texas A&M University-Corpus Christi, Corpus Christi, TX 78412 USA (e-mail: david.ma@tamucc.edu).

Digital Object Identifier 10.1109/TITS.2022.3201880

the end use case with real-time signal control implementation is difficult to achieve due to complex model architectures and high-dimensional weight matrices, leading to increased computational burden. Note that the end application of any of these modeling and prediction approaches is to alleviate the existing problems (e.g., travel delay) and increase human comfort on the road. There are several aleatory and epistemic uncertainties affecting the possibilities to achieve this required objective, but one effective controllable factor is the signal timing plan, which can have positive influence on several other factors when executed properly.

Most machine learning (ML) methods use various models to find the optimal model structure for minimizing the error between predicted and actual values. However, the problem is not solved completely by developing sophisticated models to accurately predict factors such as delay, vehicle speed, and travel time. There should be a way to control the factors that affects the response variable of interest (e.g., delay, energy consumption) in real time.

For instance, consider the generic form of the delay prediction model given in the following format

$$Y_{i,k}(t+1) = F\{X_{i,k}(t)\} + \epsilon_{i,k}(t+1), \tag{1}$$

where

i: intersection number

k: phase number

t: the time step representing the progression of sampling instant

$Y(t+1)$: delay at time $t+1$

X: the input vector (e.g., delay, green duration and vehicle volume) observed

$\epsilon(t+1)$: Model error at time $t+1$

F: Non-linear function.

In the above equation, function F can be easily approximated using any ML model (e.g., ANN, RNN, etc.) to predict the required response Y. The model parameters are estimated with the objective of minimizing the error defined by the difference between actual and predicted values. Once the model for predicting the one-step-ahead delay $Y(t+1)$ is available, the optimized signal control plan can be devised [10], [11], [12]. However, the prediction model given in (1) is highly non-linear with a neural network, making the signal control optimization problem non-linear and even NP-hard in most cases.

More recently, deep reinforcement learning (DRL) algorithms have been studied extensively and made significant progress in traffic control domains [13], [14], [15]. DRL algorithms directly provide controls that are rooted in the Markov decision process (MDP) by adapting to real-time changes in traffic environment with a rewarding or penalizing criterion, and they learn sequentially from the collected state-action pairs for each intersection. There are several key challenges, including defining the environment, setting up the reward function, modeling the relationships between actions and future reward, and, most importantly, coordination and information sharing between multiple agents representing the intersections. In addition, DRL-based approaches are approximations of classic dynamic programming methodology, where the approximations (for both value and policy) are performed based mostly on ML models using the observed state vector, which again takes the functional form F given in (1).

Although artificial intelligence (AI) theory for modeling and signal control is maturing, several challenges remain in terms of field testing and closed-loop control implementation for signalized intersections. Some of the reasons include insufficient data for fast feedback control, complex models, lack of infrastructure, inability of the models to capture the nonlinear and dynamic stochastic nature with high reliability, and robustness for urban road networks. Li *et al.* [16] proposed a hybrid neural network model (HNN) that can overcome the aforementioned challenges and suitable for real-time implementation. The results were promising but have been attempted only for one corridor (seven intersections) with limited data. There can be several challenges and potential improvements when extending to multiple corridors utilizing a large amount of data with various forms of neural network architectures.

In this paper, a hybrid recurrent neural network (HRNN) method for modeling nonlinear dynamic traffic systems is proposed to achieve both modeling accuracy and ease of real-time control implementation. The objective is to predict delay one time step ahead as a function of current delay, traffic volume, and green light duration. The control variable being considered is the green light duration and represented in a linear term, whereas non-linearity of the traffic are captured by the RNN model. This reduces the complexity of the optimization problem to obtain the optimal green light duration for real-time implementation. This hybrid modeling approach was tested for 34 signalized intersections divided into 4 corridors along the Ala Moana Boulevard and Nimitz Highway arterial in Honolulu, as shown in Fig. 1. Real-world event-based detector data were collected and used for model training and validation. The proposed hybrid modeling approach shows better performance in terms of both modeling accuracy (training and testing) and runtime computational burden when compared with several of the most commonly used ML-based modeling approaches.

The rest of this paper is organized as follows. Section II summarizes the real-time sensor data collection, preprocessing, and delay calculation model. Section III describes the modeling approach, formulation of HRNN, training methodology using least squares and gradient approaches for both linear and nonlinear parts to represent the relationship between traffic delays and signal timing plans. Section IV presents modeling results and the performance analysis for the arterial with 34 signalized intersections, followed by the conclusion and discussions of future research in Section V.

II. Data Description

As discussed above, the primary objective of this paper is to model and predict traffic delay at signalized intersections, where the inputs are the current delay, traffic volume, and green light duration of all phases (e.g., left turns, right turns, and through movements) of each intersection. This paper studies a total of 34 signalized intersections in Honolulu, Hawaii, as shown in Fig. 1 where the green dots in the figure along the arterial show the locations of these intersections.

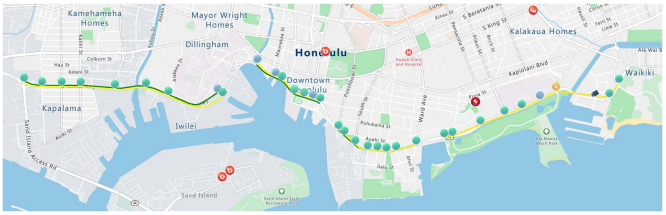


Fig. 1. The 34-intersection study area along Ala Moana Boulevard and Nimitz Highway.

All intersections along this arterial road are equipped with the Centracs Signal Performance Measures (SPM) data collection system by Econolite. This system uses video cameras, magnetic detectors, and the Cobalt advanced traffic control (ATC) controller to collect high-resolution traffic event data for all intersections. Every approach to these intersections is equipped with one camera to capture traffic event data for all signal phases. For every lane and phase, the cameras and/or magnetic sensors act as three types of detectors: (a) advanced detectors that locate a few hundred feet upstream from the stop-bar to record the vehicle arrival events; (b) stop-bar detectors that detect vehicle presence for an area directly upstream of the stop-bar and trigger vehicle calls to the signal controller; and/or (c) pulse detectors located immediately downstream of the stop-bar to detect vehicle departure events.

Note that the current Econolite system implemented in the real world is capable of processing the raw camera video data into detector events data in real time. The data processing presented herein is to take these unstructured detector events data as input to derive the estimation of traffic delay, signal duration, and volume information. The data processing method is designed to be real-time executable and is currently undergoing implementation to the actual signal control system in Hawaii.

A. Data Processing

The Centracs SPM system records historical high-resolution data. The data are collected at one second intervals for every phase along the intersections as high-resolution event data in a format developed based on the research conducted by the Purdue University and the Indiana Department of Transportation (INDOT) [17]. Fig. 2 shows an excerpt of the raw event data.

The three columns in this figure represent timestamps of each event, event code, and event parameter. Detector-triggering events are assigned with unique event parameters (i.e., identification); and each detector on and off event is associated with a unique event code. For example, as shown in Fig. 2, event code 82 indicates the "detector on" event, and event code 81 indicates the "detector off" event. Similarly, different event codes also represent different signal switching events and event parameters denote the corresponding phases of such events. For example, Fig. 2 contains a "green off" event (event code 7) for phase 5, and a "green on" event (code 1) for phase 2 and then phase 6. The event parameters are defined in [17], and event codes are based on the intersection's detector layout from Econolite. All detector and signal events were extracted from the raw high-resolution

Timestamp	Event code	Event Param	
2/1/2021 11:27:30	44	1	
2/1/2021 11:27:31	7	1	
2/1/2021 11:27:31	8	1	
2/1/2021 11:27:31	63	13	
2/1/2021 11:27:33	81	36	
2/1/2021 11:27:33	44	5	
2/1/2021 11:27:33	82	37	Detector on, Detector id 37
2/1/2021 11:27:33	7	5	Green off, phase 5
2/1/2021 11:27:33	8	5	Yellow on, phase 5
2/1/2021 11:27:33	63	15	
2/1/2021 11:27:33	81	37	Detector off, Detector id 37
2/1/2021 11:27:35	10	1	Red clearance on, phase 1
2/1/2021 11:27:35	9	1	
2/1/2021 11:27:35	64	13	
2/1/2021 11:27:35	65	13	
2/1/2021 11:27:36	0	2	
2/1/2021 11:27:36	11	1	Red clearance off, phase 1
2/1/2021 11:27:36	1	2	Green on, phase 2
2/1/2021 11:27:36	2	6	
2/1/2021 11:27:36	12	1	
2/1/2021 11:27:36	21	2	
2/1/2021 11:27:37	10	5	Red clearance on, phase5
2/1/2021 11:27:37	9	5	
2/1/2021 11:27:37	64	15	
2/1/2021 11:27:37	65	15	
2/1/2021 11:27:38	0	6	
2/1/2021 11:27:38	11	5	Red clearance off, phase5
2/1/2021 11:27:38	1	6	Green on, phase6
2/1/2021 11:27:38	12	5	

Fig. 2. Excerpt of raw high-resolution event data.

data. By calculating the timestamp differences between "green on" and "green off" event code, the green light duration can be calculated. The total number of "detector on" events at advanced detectors or pulse detectors indicates the vehicle count of each phase. Technically, the number of advanced detector events should match the number of pulse detectors, but there could be missing data that lead to discrepancies among the detectors. In this study, volume was calculated as the maximum total number of "detector on" events of advanced detectors and pulse detectors.

To analyze data discrepancies, the actual video clips from traffic cameras were manually processed and labelled. The data discrepancies mostly occurred at advanced detector locations (hundreds of feet away from the stop bar), where the detectors failed to detect vehicles when they were further away from the cameras and only appear as very few camera pixels in the images. The advanced detectors' accuracy depends on the configurations (e.g., detectors' distance to intersection, geometry of the road, number of lanes) of each intersection. On the other hand, at pulse detector locations (immediately downstream of the stop bar), the traffic camera system usually can detect vehicles accurately. Because most of the data discrepancies were missing counts at the advanced detectors, whenever the number of advanced detector events and those of pulse detectors did not match, additional artificial detector calls were added. For example, as shown in Fig. 3, there were three advanced detector events during the red light period, but there were four pulse detector events during the queue discharge (i.e., the green light period). This indicates that an advanced detector event was missing. Thus, an artificial detector call (i.e., the orange dot) was added. The timestamp of this artificial detector call was calculated as the average of two other timestamps: (a) the average time of all registered

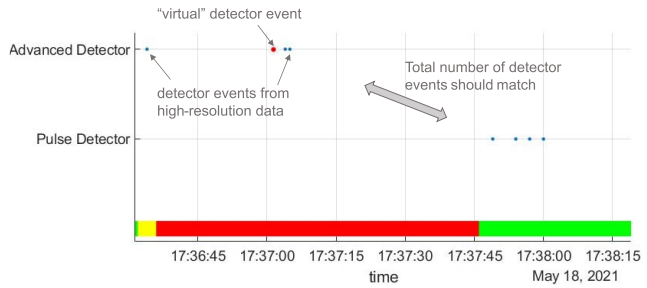


Fig. 3. Handling missing data.

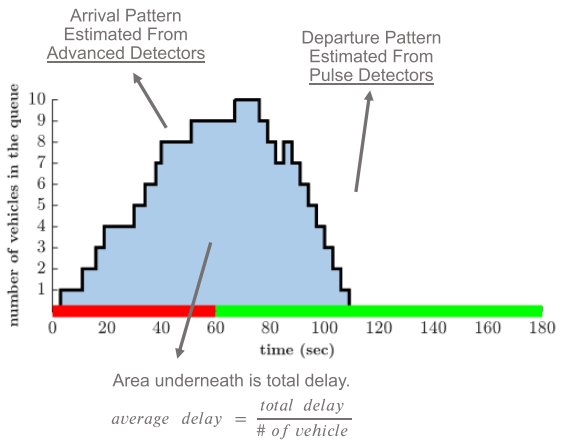


Fig. 4. Illustration of delay calculation.

advanced detector events; and (b) the middle time of the red signal light. Once missing data were filled in based on the aforementioned approach, travel delay was calculated based on the method depicted in the next subsection. This missing data handling approach was developed to help address limitations of the current video-based traffic detection system. Upgrading the current video detection system is outside the scope of this work. That said, the authors are investigating new AI-based traffic monitoring devices and algorithms that could further improve the queue estimation and delay calculation process for future research.

B. Delay Calculation

After all the detector events, volumes, and green light durations were extracted, the traffic delay was calculated. The calculation was based on the queue estimated for each lane of each phase. Essentially, advanced detector events record the vehicle arrival patterns, and pulse detector events measure the vehicle departure patterns. The vehicle queue at each lane can be estimated based on the method depicted in Fig. 4. The total traffic delay is the total wait-in-queue time duration of all vehicles in the queue, which is essentially the blue shaded area in Fig. 4. As such, the average delay was calculated by dividing the total delay by the number of vehicles. The final processed data were obtained in a structured format suitable for supervised learning in this study.

III. Hybrid Modeling Methodology

In this section, development of the hybrid modeling framework for the required data-driven prediction and control implementation is described. Before proceeding directly with implementing ML models, a simple dynamic MIMO linear model was attempted based on (2) to check whether the system could be well represented as a linear system. The output vector from this model groups the one-step ahead travel delays of all the phases while the input vectors to the model are current delay, signal timing plan and traffic volume. Such an MIMO model represents the linear relationship between the input and output vectors and the idea is to predict travel delays at time step $(t+1)$ using the available input data at time step t. This linear model will be easier to solve for deciding optimal signal control plans as those presented in [10] and [11], which were formulated as a linear quadratic optimization problem.

$$y_{i,k}(t+1) = Ay_{i,k}(t) + Bu_{i,k}(t) + Cv_{i,k}(t) + \epsilon_{i,k}(t+1) \quad (2)$$

where,

i: intersection number
k: phase number
$y_{i,k}(t)$: delay at time t
$u_{i,k}(t)$: green light at time t
$v_{i,k}(t)$: vehicle volume at time t
$\epsilon_{i,k}(t+1)$: model error at time $t+1$
A, B, C: parameters to be estimated

The ordinary least squares (OLS) method was used to determine the unknown parameter matrices denoted by $\{A, B, C\}$ with available data. The mean absolute percentage error (MAPE) of the linear model errors turned out to be around 30% and showed a non-linear relationship between input and output variables. This was consistent with the results presented by Li *et al.* [16]. To improve the prediction accuracy and ensure the model's suitability for signal control implementation, a hybrid modeling method that captures the nonlinear dynamics of the system was deemed essential to reflect the relationship between input-output factors. This is described in the following subsection.

A. Hybrid Recurrent Neural Network Formulation

Though the aforementioned linear model in (2) is well-suited for implementing signal control algorithms, the model does not seem to work well for prediction purposes. The use of ML models has outperformed most of the linear modeling approaches over time [3], [4], [5], [6], [8], [9], [12]. Since the dataset under consideration is MIMO time series data, RNNs would be well-suited for prediction purposes. This is because RNN is a type of neural network architecture specialized for processing data with temporal dependencies and nonlinearities. Apart from this, the addition of LSTM cells and gated recurrent units (GRU) will help handle vanishing gradient problems and make the model suitable for prediction purposes. Nevertheless, as discussed earlier, all these models have non-linear activation functions, and the model constraint on the control variable increases the complexity of solving RNNs directly. Therefore, the model in (2) was rewritten as (3). Note that for simplicity, we hereinafter omit i and k notations in this rewritten equation and those similar.

$$y(t+1) = Ay(t) + Bu(t) + Cv(t) + F(y(t), u(t-1), v(t); \pi) + \epsilon(t+1) \quad (3)$$

In (3), the volume of vehicles at some of the intersections were determined not to be significant. But as per expert knowledge, vehicle volume could likely play a major role in determining the travel delay. As a result, in this study, the volume of vehicles was included in the non-linear function F modeled using neural network models. If the volume of vehicles was not significant, neural network weights were automatically adjusted to zero for the corresponding intersection. Additionally, the control variable at time t (ie., the green light duration $u(t)$) was placed outside the non-linear function F so that (3) can be easily optimized with respect to $u(t)$ and the signal control plan can be implemented in a timely manner. The generalized model is given in (4).

$$y(t+1) = Ay(t) + Bu(t) + F(y(t), u(t-1), v(t); \pi) + \epsilon(t+1) \quad (4)$$

where,

$y(t)$: delay at time t

$u(t)$: green light at time t

$v(t)$: vehicle volume at time t

$\epsilon(t+1)$: model error at time $t+1$

A, B, π: parameters to be estimated

Function F in (4) was approximated using the RNN architecture with $\hat{F}(y(t), u(t-1), v(t))$ to estimate the delay $\hat{y}(t+1)$. The advantage of this model in (4) is that the control variable of interest (green light duration) is still represented in a linear form, so the optimization problems proposed in [10] and [11] for optimum controls can be solved efficiently for real-time implementation. In addition, the other variables inside the nonlinear term represented by function F capture the variance left over from the linear model, thereby improving the model accuracy in one-step ahead delay prediction. The following subsection provides the methodology to solve the HRNN model described in (4).

B. HRNN Model Training and Gradient Update Procedure

In this section, the model training and gradient update rule for the proposed hybrid modeling approach are discussed. The overall objective of the training algorithm is to minimize the error defined by the performance function given in (5) and (6).

$$Minimize: E(t+1) = \frac{1}{2}[\hat{y}(t+1) - y(t+1)]^2 \quad (5)$$

$$\hat{y}(t+1) = Ay(t) + Bu(t) + \hat{F}(y(t), u(t-1), v(t); \pi) + \epsilon(t+1) \quad (6)$$

where $\hat{y}(t+1)$ is one-step-ahead predicted delay, and A, B, and π are parameters to be trained using the real-time data from Econolite systems. In terms of the RNN model, the vector π includes the weights for three matrices U, V, W, as shown in the RNN architecture in Fig. 5 (adapted from [18]).

As depicted in Fig. 5, the RNN takes in a multi-input vector (say X_t) as a sequence, and these inputs are processed inside hidden units h_t at time t, which acts as a memory of the network. These hidden units have connections with the input vector parameterized by weight matrix U, as well as a second connection (hidden-to-hidden recurrent connections)

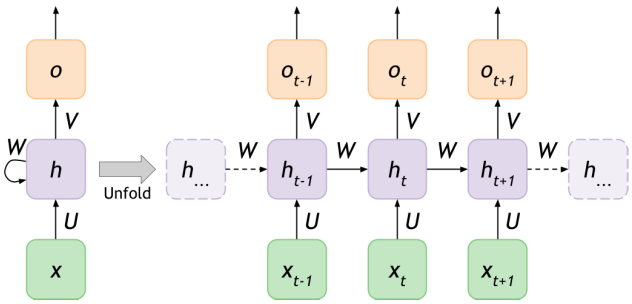


Fig. 5. Recurrent Neural Network (RNN) architecture.

parameterized by a weight matrix W, and a third connection (hidden-to-output connections) parameterized by a weight matrix V. All of these weights in the matrices $\{U, V, W\}$ are shared across time. Thus, the HRNN's forward pass will sweep through all matrices $\{A, B, U, V, W\}$, as represented by (7)–(13).

$$out_1(t) = Ay(t) \quad (7)$$

$$out_2(t) = Bu(t) \quad (8)$$

$$a(t) = b + Wh(t-1) + Ux(t) \quad (9)$$

$$h(t) = \sigma_1 a(t) \quad (10)$$

$$o(t) = c + Vh(t) \quad (11)$$

$$out_3(t) = \sigma_2 o(t) \quad (12)$$

$$\hat{y}(t+1) = out_1(t) + out_2(t) + out_3(t) \quad (13)$$

Equations (7) and (8) represent the linear model, where the outputs $\{out_1(t),\ out_2(t)\}$ are weights $\{A,\ B\}$ multiplied by corresponding input vectors. Equation (9) denotes the start of the RNN layer where the intermediate output $a(t)$ is calculated using the input vector $x(t)$ at time t and the information $h(t-1)$ from the previous time step $t-1$. This vector $a(t)$ is transformed by passing through an activation function σ_1, as given in (10). The output of this RNN sequence is derived using (11) and (12) by passing through another activation function σ_2. The final output of the HRNN model $\hat{y}(t+1)$ is obtained by combining the outputs of all intermediate layers: $\{out_1(t),\ out_2(t),\ out_3(t)\}$.

Once the forward propagation is completed, the loss is calculated using the functional form given in (5). To minimize the error, the model weights must be updated by back-propagating through the layers and calculating the gradients. Considering T as the length of the time series—in other words, the total number of data points—the runtime of HRNN model is $O(T)$ and cannot be reduced using parallelization because the forward propagation process is inherently sequential, and each time step will be computed only after the previous one. The values computed during the forward pass must be stored until they are reused during the backward pass, and thus the memory cost of HRNN is also $O(T)$. This process of gradient calculation using forward and back-propagation completes the back-propagation through time (BPTT) step for HRNN modeling method [7]. Note that in (7)–(13), the parameters were shared throughout all time steps by the network, and thus the gradient at each output depended not only on the calculations of the current time step, but also on the previous

time steps. The weight updates to all the parameter matrices $\{A, B, U, V, W\}$ with corresponding learning rates $(\lambda_i, i = 1, \ldots, 5)$ are given in (14)–(19).

$$\hat{A}(t+1) = \hat{A}(t) - \lambda_1 \frac{\partial E}{\partial A}|_{\hat{A}(t),\hat{B}(t),\hat{U}(t),\hat{V}(t),\hat{W}(t)} \quad (14)$$

$$\hat{B}(t+1) = \hat{B}(t) - \lambda_2 \frac{\partial E}{\partial B}|_{\hat{A}(t),\hat{B}(t),\hat{U}(t),\hat{V}(t),\hat{W}(t)} \quad (15)$$

$$\hat{U}(t+1) = \hat{U}(t) - \lambda_3 \frac{\partial E}{\partial U}|_{\hat{A}(t),\hat{B}(t),\hat{U}(t),\hat{V}(t),\hat{W}(t)} \quad (16)$$

$$\hat{V}(t+1) = \hat{V}(t) - \lambda_4 \frac{\partial E}{\partial V}|_{\hat{A}(t),\hat{B}(t),\hat{U}(t),\hat{V}(t),\hat{W}(t)} \quad (17)$$

$$\hat{W}(t+1) = \hat{W}(t) - \lambda_5 \frac{\partial E}{\partial W}|_{\hat{A}(t),\hat{B}(t),\hat{U}(t),\hat{V}(t),\hat{W}(t)} \quad (18)$$

$$0 \leq \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5 \leq 1 \quad (19)$$

Note that the partial derivatives $\frac{\partial E}{\partial A}$, $\frac{\partial E}{\partial B}$ are straightforward to calculate, and the derivatives belonging to the RNN layer $\frac{\partial E}{\partial U}$, $\frac{\partial E}{\partial V}$, $\frac{\partial E}{\partial W}$ need the chain rule to calculate the gradient. For instance, let E denote the error at a certain time step inside the RNN layer, the partial derivative with respect to weight matrix U, i.e., $\frac{\partial E}{\partial U}$, is derived using the chain rule as given in (20) to update the weights in (14)–(19).

$$\begin{aligned}\frac{\partial E}{\partial U} &= \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial U} \\ &= (\hat{y} - y) \cdot \left(\frac{\partial out_3}{\partial o} \cdot \frac{\partial o}{\partial h} \cdot \frac{\partial h}{\partial a} \cdot \frac{\partial a}{\partial U} \right)\end{aligned} \quad (20)$$

The drawback of RNN model is the vanishing or exploding gradient problem during back-propagation for some cases [19]. Generally, this problem can be solved using gradient clipping and batch normalization methods. However, if the problem persists, then the LSTM or GRU cells can be added to the HRNN architecture, as discussed in the next section.

IV. Results and Discussion

Results of the proposed hybrid modeling methodology are discussed in this section. Prior to model training and evaluation stage, the raw data were pre-processed to remove outliers. Any missing values were imputed using the K-nearest neighbors (KNN) approach with $k = 5$. Since the multivariate data were measured at different scales, such as delay (in seconds) and the number of vehicles (discrete units), the dataset was normalized to scale the data between zero and one. For traffic delay data, after normalization, simple exponential smoothing was applied to further filter the data to remove noise, as shown in (21) and (22), where $l(t)$ is the filtered delay, $y(t)$ is the normalized delay, and α is the smoothing factor between zero and one. As α decreases, the observation of delay at time t has less impact on the output $y(t)$, indicating that the randomness of the delay measurements is reduced.

$$l(t) = \alpha y(t) + (1 - \alpha) l(t-1) \quad (21)$$

$$0 < \alpha < 1 \quad (22)$$

In this study, the 34 intersections consisting of 133 phases along the arterial in Fig. 1 were divided into 4 corridors. The data were collected for the period of March 1, 2021, to June 30, 2021, and models were fitted to evening peak hours from 4:00 pm to 7:00 pm on weekdays. The training and testing data were split in the ratio of 80 to 20. In total, four corridor-level HRNN models were trained using 80% of the total data (March–May, 2021) and validated with 20% of the remaining total data (June 2021). The HRNN structure used for this study is shown in Fig. 6.

The loss function used for the delay prediction model was MAPE, as given in (23), where $y_k(t)$ is the true delay at time t of phase k, and $\hat{y}_k(t)$ is the predicted delay at time t of phase k.

$$MAPE = \frac{1}{KT} \sum_{t=1}^{T} \sum_{k=1}^{K} \left| \frac{y_k(t) - \hat{y}_k(t)}{y_k(t)} \right| \quad (23)$$

In addition to splitting up the corridors and using the HRNN model, other variants of RNNs were also utilized inside the proposed framework—namely, hybrid long short term memory (HLSTM) and hybrid gated recurrent units (HGRU), in addition to the original HNN-MLP model developed in [16]. The equations in the previous subsection remain the same for HLSTM and HGRU versions, except for the hidden layer (h_t), as discussed in the following subsections.

A. Long Short-Term Memory (LSTM)

The LSTM unit was initially proposed in 1997 [7], and since then, a number of minor modifications to the original LSTM architecture have been made [20]. In our HRNN model, the hidden unit, which usually computes a weighted sum of the input signal and applies a nonlinear function, was modified to maintain a memory c_t (input, forget, and output) at time t for each LSTM block m as given in (24) and (25).

$$h_t^m = o_t^m \ tanh(c_t^m) \quad (24)$$

$$o_t^m = \sigma(W_o x_t + U_o h_{t-1} + V_o c_t)^m \quad (25)$$

The value c_t is the value of memory cell at time t, and it was updated by partially forgetting the current memory and adding a new memory $\hat{c}_t$. The extent to which this existing memory was forgotten is modulated by a forget gate f_t, and the degree to which the new memory content is added to the memory cell was modulated by an input gate q_t as given in (26)–(29).

$$c_t^m = f_t^m c_{t-1}^m + q_t^m \hat{c}_t^m \quad (26)$$

$$\hat{c}_t^m = tanh(W_c x_t + U_c h_{t-1})^m \quad (27)$$

$$f_t^m = \sigma(W_f x_t + U_f h_{t-1} + V_f c_{t-1})^m \quad (28)$$

$$q_t^m = \sigma(W_q x_t + U_q h_{t-1} + V_q c_{t-1})^m \quad (29)$$

In the HRNN, the hidden (recurrent) unit overwrites its content at each time step. However, in the HLSTM, the gates in the hidden unit will help decide whether the existing memory can be kept or removed. The advantage of the LSTM cell is that it helps remember long-term dependencies in the data.

B. Gated Recurrent Units (GRU)

As discussed in the previous subsection, LSTM is most suitable for long sequences with temporal dependence. However, in some cases, the dataset with short-term dependencies

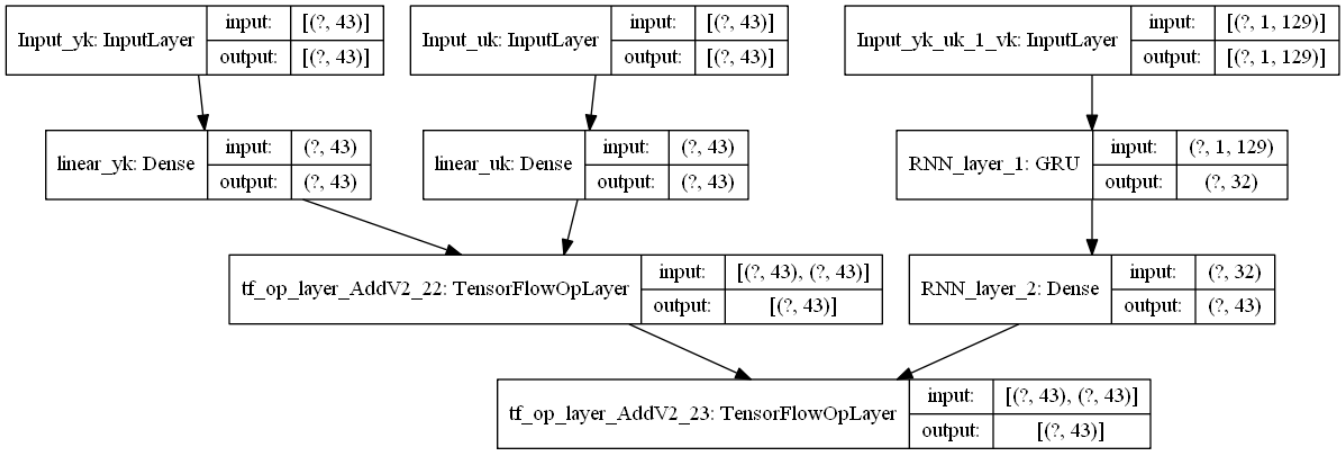


Fig. 6. Hybrid recurrent neural network (HRNN) architecture.

might not make LSTM a good candidate. Therefore, the GRU version, proposed by [20] and [21] was explored to allow each hidden unit to adaptively capture short-term temporal dependencies. Unlike LSTM, the GRU has only two gates (update z_t and reset r_t gates) that modulate the flow of information inside the hidden unit, h_t, given in (30) and (31).

$$h_t^m = (1 - z_t^m)h_{t-1}^m + z_t^m \hat{h}_t^m \tag{30}$$

$$z_t^m = \sigma(W_z x_t + U_z h_{t-1})^m \tag{31}$$

$$\hat{h}_t^m = \tanh(W x_t + U(r_t \odot h_{t-1}))^m \tag{32}$$

$$r_t^m = \sigma(W_r x_t + U_r h_{t-1})^m \tag{33}$$

The update gate z_t will decide on the extent to which the unit updates its activation or content in addition to candidate activation unit $\hat{h}_t^m$ with reset gate for each GRU block m as given in (32) and (33).

Once all abovementioned hybrid models were constructed, their performance varied drastically during the initial evaluation stage for individual phases at each intersection. There was a common pattern that one model seemed to work well, in terms of testing errors, for certain intersections but performed poorly for others in the same corridor. Thus, it was not possible to conclude that a single model would work well for all intersections, posing a model selection problem. This problem was a result of the model hyperparameters being improperly tuned. As a result, we devised a hyperparameter tuning process to address this issue. The following subsection provides details about this hyperparameter tuning process.

C. Hyperparameter Tuning

The hyperparameters under consideration are the activation function (Sigmoid, ReLu, Exp, SeLu, Linear, etc.,), learning rate (between 0 and 1), and number of neurons in hidden layers (0 to 64). Note that matrices A and B in (4) must be a square matrix for optimal signal timing plan calculations. Therefore, the number of neurons in non-linear function F in (4) is the variable that must be tuned along with the learning rate and activation function. Several methodologies are available to tune all the hyperparameters and select the best model before deciding on the real-time implementation [22]. Some of the primary methods for tuning model hyperparameters, such as the grid search method and random search method, involve an exhaustive searching procedure using a specified subset of parameter space and suffer from the curse of dimensionality. In addition, the parameter space of ML algorithms usually includes real values or unbounded values for certain parameters. Thus, it is necessary to discretize some parameters and manually limit the bounds based on the prior knowledge to decrease convergence time and function evaluations, which will affect the model accuracy—especially when there are a large number of hyperparameters. There are huge matrix calculations involved in each epoch of the HNN models, which further increase the computational burden of hyperparameters tuning.

To overcome the aforementioned shortcomings, a Bayesian hyperparameter tuning approach was used in this paper. The Bayesian optimization method can provide the global optimum, especially with noisy black-box functions such as neural networks. In practice, Bayesian optimization has been shown to obtain better results in fewer evaluations compared to both grid search and random search methods, due to its ability to adaptively sample data points for experimentation before evaluation [23], [24], [25], [26].

The process of hyperparameter tuning began by randomly initializing two sample points from the given parameter space and evaluated on the objective function. A probabilistic surrogate model using a Gaussian kernel (termed as *acquisition/utility function*) was built from the original function mapping from these initial promising hyperparameter samples (i.e., the functional form of HNN models was approximated to be Gaussian). The expected improvement (EI) criterion [27] was used to select the next promising sample point for evaluation. This new best guess that minimized the loss was evaluated using the objective function, and the process continued by iteratively evaluating new samples of hyperparameter configurations. This sequentially updated the current surrogate model in turn, and the best possible combination of hyperparameters that minimized the model loss was selected.

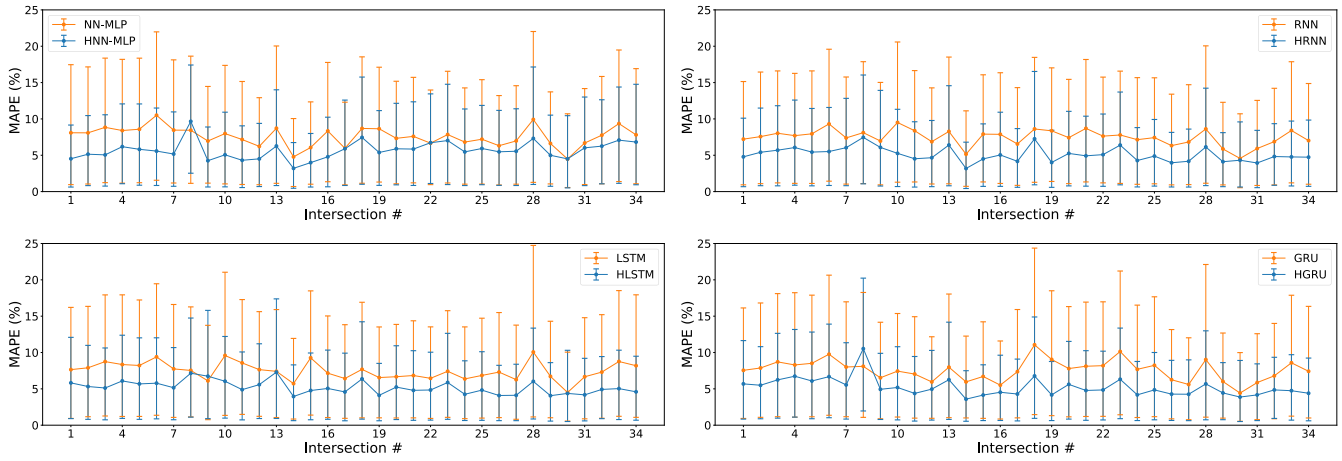


Fig. 7. Comparison of MAPE (%) along with error bars for different modeling frameworks.

TABLE I
COMPARISON OF MEAN ABSOLUTE PERCENTAGE ERROR (MAPE) FOR BASELINE MODELS (A) AND HYBRID MODELS (B)

Model Number	Description	Training MAPE (%)	Testing MAPE (%)
1a	NN-MLP	7.60	11.37
2a	RNN	7.47	11.38
3a	LSTM	7.47	11.08
4a	GRU	7.57	11.21
1b	HNN-MLP	5.67	8.68
2b	HRNN	5.09	7.96
3b	HLSTM	5.19	7.66
4b	HGRU	5.22	7.20

After tuning the hyperparameters for all individual models in each corridor, each of the models was trained using the corresponding set of optimum hyperparameters, and the MAPE was used to compare the performance. Table I shows the modeling results for all 34 intersections. The learning rate for all the models was tuned and trained with a decay factor of 0.00001. The proposed HRNN, HLSTM, and HGRU models were compared with the existing HNN-MLP model, as well as their counterpart ML architectures. Results in terms of training and testing MAPEs of these models are tabulated in Table I. As shown in the results, the hybrid models show consistently better performance similar than most traditional ML models for all intersections with an MAPE around 10%.

The MAPE evaluates the mean model performance, but there could be extreme values hidden in some cases. To further compare, the model performance for individual intersections along with error bars, which provide the range of model variation with respect to the mean value, were plotted. As shown in Fig. 7, the proposed hybrid models (blue) are more robust as they show smaller MAPE variations than those of the traditional baseline models (orange). In addition, the HGRU model slightly outperforms other models because of the presence of update and reset gates, making it suitable for short-term sequential data. In addition to better modeling results, as demonstrated in the next subsection, the proposed hybrid modeling approach leads to better efficiency for control optimization.

D. Signal Control Optimization

In order to demonstrate the efficacy of the proposed modeling methodology, a simple signal timing plan for one signal cycle was generated for the 9 intersections in first corridor of the study area. The control objective here is to minimize one-step-ahead delay, $Y_{i,k}(t+1)$, as given in (34) and (35) [10], [11].

$$u^*_{i,k}(t) = \underset{u_{i,k}(t)}{\arg\min} \left[Y_{i,k}(t+1) \right] \tag{34}$$

$$\text{s.t. } L_{i,k} \leq u_{i,k}(t) \leq U_{i,k} \tag{35}$$

where

$u_{i,k}$: Control variable (green light duration for intersection i and phase k)

$L_{i,k}$: Lower bound for green light duration

$U_{i,k}$: Upper bound for green light duration

As mentioned previously, the delay at time $t+1$ in (34) can be represented as a function of current delay, volume, and green light duration in (4). The unknown parameters in (4) were already estimated using a hybrid modeling method, as discussed in the previous sections. Since the model prediction power has already been shown to be close to the actual value (Table I), the one-step-ahead delay, $Y_{i,k}(t+1)$, in (34) is replaced using the estimated delay from the hybrid modeling method, $\hat{Y}_{i,k}(t+1)$, as shown in (36).

$$u^*_{i,k}(t) = \underset{u_{i,k}(t)}{\arg\min} \left[\hat{Y}_{i,k}(t+1) \right] \tag{36}$$

Several optimization algorithms—such as Trust Region, Interior Point, and Nelder-Mead, etc.—are readily available to solve the optimization problem and arrive at optimal signal control values. However, as mentioned previously, the main advantage of our proposed modeling method is that the control variables (i.e., the green light duration) are represented using a linear form, as shown in (4). Therefore, it is easy to derive an analytical solution for the signal control optimization problem.

TABLE II
RUN-TIME (IN SECONDS) COMPARISON FOR ONE SIGNAL CYCLE FOR BASELINE MODELS (a) AND HYBRID MODELS (b)

Model Number	Description	Analytical Method (s)	Trust-Region Optimization (s)
1a	NN-MLP	N/A	27.06
2a	RNN	N/A	27.30
3a	LSTM	N/A	26.67
4a	GRU	N/A	26.83
1b	HNN-MLP	0.030	1.34
2b	HRNN	0.031	1.26
3b	HLSTM	0.030	1.25
4b	HGRU	0.030	1.26

Because the overall objective is to minimize the delay, the prediction model output is set to be equal to zero.

$$0 = \hat{A}y(t) + \hat{B}u(t) + \hat{F}(y(t), u(t-1), v(t); \hat{\pi}) \quad (37)$$

The matrix $\hat{B}$ may not be a full column rank matrix, and it might not be invertible; thus, it is necessary to multiply $\hat{B}^T$ to both sides of (37), leading to (38).

$$0 = \hat{B}^T\hat{A}y(t) + \hat{B}^T\hat{B}u(t) + \hat{B}^T\hat{F}(y(t), u(t-1), v(t); \hat{\pi}) \quad (38)$$

Rearranging the terms will provide an analytical solution to the optimal signal timing variable $u(t)$ given by (39).

$$\hat{u(t)} = -(\hat{B}^T\hat{B})^{-1}[\hat{B}^T\hat{A}y(t) + \hat{B}^T\hat{F}(y(t), u(t-1), v(t); \hat{\pi})] \quad (39)$$

Because the values of all unknown parameters and variables are known until time t, the optimal control values $u(t)$ for the current cycle that minimize the future delay at time $t+1$ can be obtained using the analytical form in (39). All the matrix calculations were performed using a PC with a 10-core CPU with 16 GB memory, and the runtime results of signal time optimization are provided in Table II.

The proposed analytical approach is almost 10 times faster than solving the signal timing optimization problem using any available optimizers. This analytical solution was possible only because of the hybrid method of modeling with linear and nonlinear terms; it cannot be derived easily by other traditional ML approaches. The solution time can also vary depending on placing additional constraints on cycle time and boundary space. In addition, the proposed approach was demonstrated on real-time data for one signal cycle for the first corridor at 5:30 pm to determine the signal timing plan for next cycle. To measure the impact of the proposed solution, it is essential to incorporate this optimal signal timing plan into either simulated signal controllers in a simulation environment or real signal controllers and continue with closed-loop control, which is beyond the scope of this paper.

V. CONCLUSION

In this paper, a new hybrid modeling method for traffic delay prediction suitable for real-time implementation was developed to eliminate drawbacks of current ML-based methods for signalized intersections. The advantage of the developed hybrid modeling method is that it can capture both linear and nonlinear stochastic characteristics of multiple traffic features—traffic signal timings, traffic volume, travel delays, etc., and it is computationally efficient for real-time implementation. The proposed method was validated by developing MIMO neural-network-based models for multiple intersections along an arterial in Hawaii with event-based data extracted from the Econolite systems. Also, a sample control plan was generated for one signal cycle to demonstrate the effectiveness and advantage of the proposed approach. During actual implementation, this signal timing plan will be iteratively generated and fed to the field signal controller using a closed-loop feedback system configured for all the intersections.

The quality of the results depends mainly on the quality of data collected—and the delay calculations from the event-based data. Due to limitations of the current system, there could be data discrepancies and missing data. New AI-based traffic monitoring devices and algorithms are under investigation to further improve the queue estimation and delay calculation process for future research.

Nevertheless, the overall MAPE in delay prediction for all the intersections was shown to be less than 10%. These results represent evening peak hours traffic data (4pm–7pm), but similar performance is expected for other times of the day. As mentioned earlier, this paper focuses only on modeling methodology to predict the one-step-ahead delay given the raw traffic data. The effective use case of this approach can be fully understood during the implementation phase by utilizing the closed-loop control strategy, which is the authors' ongoing work. Since the control variable of interest is in a linear form, this prediction model can be easily implemented to achieve a real-time closed-loop control system that uses traffic flow state as feedback to control signal timing 24/7 intelligently along a multi-intersection corridor.

At present, this hybrid modeling method is being used to develop a real-time, AI-based optimal traffic control system in Hawaii to achieve smoother traffic flow with minimized delay and energy consumption. In our approach, though the model is well-trained with historical data, it will be continuously updated based on the data collected, and the control parameters will be updated to ensure the model's robustness to uncertainties. Also, a microscopic traffic simulation model is under development to build a digital twin that can reflect the real traffic environment for the area under study using PTV VISSIM. The developed delay prediction model and the corresponding control algorithm will be implemented in the simulation environment for testing and validation before real-world implementation.

ACKNOWLEDGMENT

The authors would like to thank Econolite Systems for providing high resolution event-based data that were used in this study. In addition, the technical support and advanced computing resources from Transportation Technology Laboratory (TTL, UH Manoa) as well as University of Hawaii Information Technology Services–Cyberinfrastructure.

REFERENCES

[1] B. M. Williams, P. K. Durvasula, and D. E. Brown, "Urban freeway traffic flow prediction: Application of seasonal autoregressive integrated moving average and exponential smoothing models," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1644, no. 1, pp. 132–141, Jan. 1998.

[2] D. Srinivasan, M. C. Choy, and R. L. Cheu, "Neural networks for real-time traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 3, pp. 261–272, Sep. 2006.

[3] R. Ke, W. Li, Z. Cui, and Y. Wang, "Two-stream multi-channel convolutional neural network for multi-lane traffic speed prediction considering traffic volume impact," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2674, no. 4, pp. 459–470, Mar. 2020.

[4] H. Yuan and G. Li, "A survey of traffic prediction: From spatio-temporal data to intelligent transportation," *Data Sci. Eng.*, vol. 6, no. 1, pp. 63–85, Mar. 2021.

[5] W. Li *et al.*, "Short-term traffic state prediction from latent structures: Accuracy vs. efficiency," *Transp. Res. C, Emerg. Technol.*, vol. 111, pp. 72–90. Feb. 2020.

[6] J. Guo, W. Huang, and B. M. Williams, "Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 50–64, Jun. 2014.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[8] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values," *Transp. Res. C, Emerg. Technol.*, vol. 118, Sep. 2020, Art. no. 102674.

[9] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4883–4894, Nov. 2019.

[10] 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.

[11] H. Wang, S. V. Patil, H. M. A. Aziz, and S. Young, "Modeling and control using stochastic distribution control theory for intersection traffic flow," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 1885–1898, Mar. 2022.

[12] 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.

[13] H. Wei, G. Zheng, V. Gayah, and Z. Li, "Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation," *ACM SIGKDD Explor. Newslett.*, vol. 22, no. 2, pp. 12–18, Jan. 2021.

[14] Z. Li, H. Yu, G. Zhang, S. Dong, and C.-Z. Xu, "Network-wide traffic signal control optimization using a multi-agent deep reinforcement learning," *Transp. Res. C, Emerg. Technol.*, vol. 125, Apr. 2021, Art. no. 103059.

[15] C. Chen *et al.*, "Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, 2020, pp. 3414–3421.

[16] W. Li *et al.*, "Hybrid neural network modeling for multiple intersections along signalized arterials—Current situation and some new results," in *Proc. Veh. 10th Int. Conf. Adv. Veh. Syst., Technol. Appl.*, 2021, pp. 10–20.

[17] H. Li *et al.*, "Indiana traffic signal hi resolution data logger enumerations," Indiana Dept. Transp., Purdue Univ., West Lafayette, IN, USA, Tech. Rep., 2019, doi: 10.5703/1288284316998.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[19] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[20] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.

[21] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2014, *arXiv:1409.1259*.

[22] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, no. 2, pp. 281–305, 2012.

[23] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Proc. Int. Conf. Learn. Intell. Optim.* Cham, Switzerland: Springer, 2011, pp. 507–523.

[24] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2013, pp. 847–855.

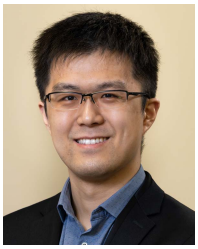
[25] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1–9.

[26] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 24, 2011, pp. 1–9.

[27] F. Nogueira. (2014). *Bayesian Optimization: Open Source Constrained Global Optimization Tool for Python*. [Online]. Available: https://github.com/fmfn/BayesianOptimization

Arun Bala Subramaniyan received the Ph.D. degree in industrial engineering from the School of Computing, Informatics and Decision Systems Engineering, Arizona State University. He is currently a Post-Doctoral Research Associate at the University of Hawaii at Mānoa. His expertise is in the area of statistical modeling, machine learning, and reinforcement learning. His current research interests include developing artificial intelligence-based methodologies for smart mobility applications.

Chieh Wang (Member, IEEE) received the B.S. and M.S. degrees in civil engineering from the National Taiwan University, and the Ph.D. degree in civil and environmental engineering from the Georgia Institute of Technology. He is currently a Research and Development Associate Staff Member of the Oak Ridge National Laboratory, Buildings and Transportation Science Division. His research interests include transportation data analytics and visualization, intelligent vehicles, smart cities, and infrastructure management.



Yunli Shao (Member, IEEE) received the B.S. degree in mechanical engineering from Shanghai Jiao Tong University, China, the M.S. degree in mechanical engineering from the University of Michigan, and the Ph.D. degree in mechanical engineering from the University of Minnesota. He is currently a Research and Development Associate Staff Member of the Oak Ridge National Laboratory, Buildings and Transportation Science Division. His research interests include intelligent transportation systems and control, optimization, and evaluation of connected and autonomous vehicles.

Wan Li received the M.S. degree in transportation engineering from Louisiana State University in 2014 and the Ph.D. degree in transportation engineering from the University of Washington in 2019. She is currently a Research and Development Associate with the Oak Ridge National Laboratory, Buildings and Transportation Science Division. Her research interests mainly focus on urban traffic network modeling and simulation, traffic control system design and optimization, shared mobility technologies, and data-driven spatiotemporal traffic forecasting.

Hong Wang (Senior Member, IEEE) received the master'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, Uxbridge, U.K., and Southampton University, Southampton, U.K., before joining the University of Manchester Institute of Science and Technology (UMIST), Manchester, U.K., in 1992. He was a Chair Professor in process control of complex industrial systems at the University of Manchester, U.K., from 2002 to 2016, where he was the Deputy Head of the Paper Science Department, the Director of the UMIST Control Systems Centre (which is the birthplace of Modern Control Theory established in 1966) between 2004 and 2007. He was a University Senate Member and a member of general assembly during his time in Manchester. Between 2016 and 2018, he was with the Pacific Northwest National Laboratory (PNNL), Richland, WA, USA, as a Laboratory Fellow and a Chief Scientist, and was the Co-Leader and a Chief Scientist for the Control of Complex Systems Initiative (https://controls.pnnl.gov/). He joined the Oak Ridge National Laboratory in January 2019. His research interests focuses on stochastic distribution control, fault diagnosis and tolerant control, and intelligent controls with applications to the transportation system area. He is a member of three IFAC committees. He was an Associate Editor of IEEE TRANSACTIONS ON AUTOMATIC CONTROL, IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, and IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING. At present, he is an Associate Editor of IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.

Guohui Zhang received the Ph.D. degree from the University of Washington (UW), Seattle, in 2000. He is currently an Associate Professor with the Department of Civil and Environmental Engineering, University of Hawaii at Mānoa, Honolulu. His primary research interests include transportation system resilience analysis, large-scale transportation systems modeling, sustainable traffic network infrastructure design, planning, and operation, traffic sensing and senor data analytics, artificial intelligence in transportation, connected and autonomous vehicle systems, traffic-impacted public health, cyber-transportation systems, and transportation safety and security. He has been serving as a member for the American Society of Civil Engineers (ASCE) Connected and Autonomous Vehicle (CAV) Impact Committee, the Transportation Research Board (TRB) Information Systems and Technology Committee, and as a panelist for multiple National Collaborative Highway Research Program (NCHRP) projects.

Tianwei Ma received the Ph.D. degree in mechanical engineering from the University of California, Santa Barbara. He is currently the Dean and a Professor of the College of Engineering, Texas A&M University-Corpus Christi. His research interests spans a wide spectrum of engineering fields, ranging from structural control, smart sensing and bridge monitoring to renewable energy, and smart transportation. In the past decades, his research has been focused on energy harvesting, blue energy technology, and autonomous vehicle technologies. In addition to his academic appointments, he serves as the Chairperson of Board of Directors of Hawaii Autonomous Vehicle Institute, a non-profit organization aiming for safely bringing autonomous vehicle technology to Hawaii. He has also been a reviewer for numerous technical journals and frequently served on NSF review panels.