Project Title

Enhancing traffic safety and connectivity:
A data-driven multi-step-ahead vehicle headway prediction leveraging high-resolution vehicular trajectories

Investigators

Mohamadhossein Noruzoliaee (https://orcid.org/0000-0003-3860-8911)
Fatemeh Nazari (https://orcid.org/0000-0003-1587-1848)

Institution

The University of Texas Rio Grande Valley (UTRGV)

FINAL REPORT

July 30, 2024

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, under [grant number 69A3552344811] from the U.S. Department of Transportation's University Transportation Centers Program. The U.S. Government assumes no liability for the contents or use thereof.

1. Report No. 470	**2. Government Accession No.**	**3. Recipient's Catalog No.**
4. Title and Subtitle Enhancing traffic safety and connectivity: A data-driven multi-step-ahead vehicle headway prediction leveraging high-resolution vehicular trajectories		**5. Report Date** July 30, 2024 **6. Performing Organization Code**
7. Author(s) Mohamadhossein Noruzoliaee, Ph.D. https://orcid.org/0000-0003-3860-8911 Fatemeh Nazari, Ph.D. https://orcid.org/0000-0003-1587-1848		**8. Performing Organization Report No.**
9. Performing Organization Name and Address The University of Texas Rio Grande Valley (UTRGV) 1201 W University Dr Edinburg, Texas 78539		**10. Work Unit No.** **11. Contract or Grant No.** Federal Grant No. 69A3552344811
12. Sponsoring Agency Name and Address Safety21 University Transportation Center Carnegie Mellon University 5000 Forbes Avenue Pittsburgh, PA 15213		**13. Type of Report and Period Covered** Final Report (July 1, 2023-June 30, 2024) **14. Sponsoring Agency Code** USDOT

15. Supplementary Notes
Conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.

16. Abstract
Vehicle headway, defined as the time elapsed between two successive vehicles passing a roadway point, is a key mesoscopic-scale measure in traffic flow theory with safety-critical transportation applications, such as preemptive collision avoidance warning systems as well as connected and autonomous vehicle (CAV) platoon control. Hence, it is crucial to accurately predict vehicle headway over sufficiently long future horizons (i.e., multi-step-ahead prediction) to be applicable for downstream safety-critical applications. This is a challenging task due to several random factors influencing headway, including inter- and intra-driver heterogeneity, asymmetric car-following driving behavior, and vehicle heterogeneity under mixed traffic of different vehicle classes. This becomes even more complicated under traffic congestion, which results in tangible inter-vehicle interactions and, thus, speed-dependent headways. The complex effects of the above factors on headway, along with the unprecedented amount of high time-resolution vehicle trajectory big data (e.g., datapoints recorded every 0.1 second), call for advanced data-driven headway prediction models. Deep learning architectures, particularly variants of Recurrent Neural Network (RNN), are promising candidates as they can "learn" highly nonlinear relationships from headway time-series data. However, recurrent networks are notorious for the vanishing gradient problem, which precludes learning long-term dependencies in time series data. To tackle, this project employs a state-of-the-art interpretable deep learning model for multi-step-ahead time series forecasting (e.g., next 2-5 seconds), which can accommodate reasonably long prediction horizons that can capture human/vehicle reaction time. Leveraging the vehicle trajectory big data from the USDOT's Next Generation Simulation (NGSIM) dataset, the model is trained and tested to investigate the effects on headway of microscopic traffic measures, macroscopic traffic flow, vehicle class, and lane position.

17. Key Words Vehicle headway time series; multi-step-ahead prediction; interpretable deep learning; vehicle trajectory big data	**18. Distribution Statement** No restrictions.		
19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 23	**22. Price**

Table of Contents

Chapter 1: Problem Statement & Related Work

Defined as the time distance between two successive vehicles as they pass a point on a roadway (Highway Capacity Manual, 2010), vehicle headway is a fundamental measure in traffic flow theory. Accurate prediction of vehicle headway is crucial in various transportation applications, such as enhancing the preemptive collision avoidance warning capability of smart vehicles. This could have significant traffic safety implications, since the main causes of rear-end crashes, which typically constitute almost 30% of all police reported crashes, are the drivers' inattention and following the front vehicle too closely, alone or in combination (Barfield and Dingus, 2014). Additionally, accurate prediction of vehicle headway can enhance the advanced traffic and vehicle control systems, such as connected and autonomous vehicles (CAVs), to maintain a minimum safety distance between platooning CAVs (Bian et al., 2019).

Over the past decades, several statistical models have been proposed to analytically derive *closed-form* distributions for vehicle headway (see Li and Chen (2017) for a comprehensive review). However, accurate **analytical modeling** of inter-vehicle headway distribution is challenging due to a multitude of known and unknown random factors that can potentially shape the vehicle headway distribution, namely:

a) driver heterogeneity, suggesting that different drivers show diverse perceptual reactions and desired headways (i.e., inter-driver heterogeneity) and even the same driver may behave differently over time and space (i.e., intra-driver heterogeneity), leading to distinct vehicle headway distributions with different variances (Taylor et al., 2015);

b) asymmetric car-following driving behavior, which refers to the preference of drivers in maintaining short headways rather than long headways, resulting in rightly-skewed headway distributions (Chen et al., 2010);

c) vehicle heterogeneity under mixed traffic, indicating that different vehicle classes or types (e.g., passenger cars and heavy-duty trucks) have distinct dimensions and performances (e.g., vehicle speed, acceleration, and deceleration), giving rise to irregular headway distributions along a roadway and across traffic lanes; and

d) congested traffic flow conditions, under which vehicle speeds are lower than the free-flow speed and inter-vehicle interactions are not negligible, which causes the headway distribution to be speed-dependent, bridging headway to the (microscopic) instantaneous speeds of vehicles (Zhang and Wang, 2014).

The complex effects of the above known and unknown heterogenous factors on inter-vehicle headway warrant the development of more advanced vehicle headway prediction models that can effectively accommodate them. Deep learning (Goodfellow et al., 2016) can serve as a promising candidate as it can "learn" highly nonlinear relationships governed by inter-vehicle headway "data" rather than imposing *a priori* known closed-form specification for vehicle headway and fitting the data to that specification.

Such a **data-driven approach** to vehicle headway modeling has increasingly become more appealing with the emergence of unprecedented amounts of vehicle trajectory big data acquired from fixed-location sensors (for example, radars, loop detectors, and cameras) and mobile sensors (for instance, probe vehicles, connected vehicles, navigation applications, and unmanned aerial aircraft). The type of collected data have evolved from simple vehicle counts to rich vehicle trajectories, providing continuous time series data of heterogenous vehicle movements, which have been used in several traffic flow studies (see Li et al. (2020) for a recent review).

Vehicle trajectory data are inherently regarded as time series data, where a sequence of kinematic measurements of each vehicle (e.g., vehicle position, velocity, and acceleration) are collected over time. Given the vehicle trajectory time series data, modeling vehicle headway would then become a univariate time series forecasting problem, wherein vehicle headways in the future time steps are predicted based on the sequential vehicle headway data in the past time steps.

Among deep learning models, variants of recurrent neural network (RNN) architecture, such as the long short-term memory (LSTM) neural network (Hochreiter and Schmidhuber, 1997), are particularly suited for modeling time series data, because the output from previous time steps are used as input for the future time steps. Recurrent networks are popular in modeling various transportation phenomena, including car following and lane changing behavior (Huang et al., 2018; Xie et al., 2019; Zhang et al., 2019), traffic speed (Gu et al., 2019), traffic flow (B. Yang et al., 2019), CAV trajectory planning (Lin et al., 2021), bike-sharing demand (Xu et al., 2018), on-demand ride service demand (Ke et al., 2017), vehicle classification (Simoncini et al., 2018), parking occupancy (Yang et al., 2019), and traffic safety (Zhang et al., 2018).

However, the emergence of high time-resolution vehicle headway time series data (for example, data points collected every 0.1 second), on the one hand, and the need for providing vehicle headway predictions over sufficiently long horizons to be applicable for traffic safety-critical applications (e.g., the next 2-5 seconds to allow for driver perception-reaction time), on the other hand, call for multi-step-ahead (multi-horizon) time series models (e.g., 20 time steps into the future, where each time step is 0.1 second). To address this need, this project presents a state-of-the-art deep learning model for multi-step-ahead time series forecasting, and trains the model

using the vehicle trajectory big data acquired from the USDOT's Next-Generation Simulation (NGSIM) dataset (Traffic Analysis Tools: Next Generation Simulation - FHWA Operations).

The remainder of the report is organized as follows. Chapter 2 presents the multi-step-ahead time series forecasting methodology. Data preparation and description are summarized in Chapter 3, followed by the results of the model implementation on the USDOT's NGSIM vehicle trajectory big data. The report concludes in Chapter 5 with a summary of the project and key findings.

Chapter 2: Methodology

The multi-step-ahead time series prediction problem involves forecasting the future values of a target variable (i.e., vehicle headway) over multiple time steps in the future, given the historical values of the target variable and exogenous input variables. Specifically, considering a finite look-back window k and a finite look-ahead window τ, the τ-step-ahead prediction of the target variable made at time step t, as denoted by $\hat{y}_{t+\tau}$, is based on the historical information about the target variable $y_{t-k:t} = \{y_{t-k}, \dots, y_t\}$ and input variables, the latter encompassing static inputs $\boldsymbol{s}$ (e.g., vehicle length) and time-varying inputs $\boldsymbol{x}_{t-k:t}$ (e.g., vehicle acceleration).

Acknowledging the uncertainty involved in future forecasts, this project seeks prediction intervals rather than point estimates of the target variable. More clearly, the model aims to predict $\hat{y}_{t+\tau}(q)$, which is the τ-step-ahead prediction of the target variable made at time step t for the qth instance (i.e., sample) quantile. Note that, as a direct method of multi-step-ahead prediction, forecasts of $\hat{y}_{t+\tau}(q)$ will be simultaneously made for all future time steps $\tau \in \{1, \dots, \tau_{max}\}$. As shown in Eq. (1), the goal of the deep learning model presented in this chapter is to "learn" a nonlinear mapping $f_q(.)$ between the desired target variable and the historical input variables.

$$\hat{y}_{t+\tau}(q) = f_q\left(y_{t-k:t}, \boldsymbol{x}_{t-k:t}, \boldsymbol{s}, \tau\right) \tag{1}$$

Figure 1 depicts the overarching structure of the presented deep learning model, dubbed Temporal Fusion Transformers (TFT) (Lim et al., 2021), which contains three main components: 1) Variable Selection Network (VSN) using gating mechanisms, 2) short-term temporal processing using LSTM sequence-to-sequence encoder and decoder, and 3) long-term temporal processing using interpretable multi-head attention. What ensues elaborates on these three components.

2.1. Input Variable Selection Using Gating Mechanisms

Whether and how significantly each input variable can contribute to predicting the target variable (i.e., vehicle headway) is unknown *a priori* and should be "learned" from data. In addition, some input variables may be noisy and should be learned to be automatically removed to prevent from harming the model's prediction performance. For these reasons, the Variable Selection Network (VSN) is included as a building block of the TFT model architecture, which leverages gating mechanisms to generate sample-wise variable selection weights, as illustrated below.

Before feeding into the VSN, each raw input variable ($\boldsymbol{x}_t \in \mathbb{R}^{m_x}$ and $\boldsymbol{s} \in \mathbb{R}^{m_s}$) is first transformed into a (d_{model})-dimensional vector representation, where d_{model} corresponds to the

dimensions of the subsequent layers. The input variable transformation is performed using entity embeddings (Gal and Ghahramani, 2016) for categorical variables and linear transformation for continuous variables.

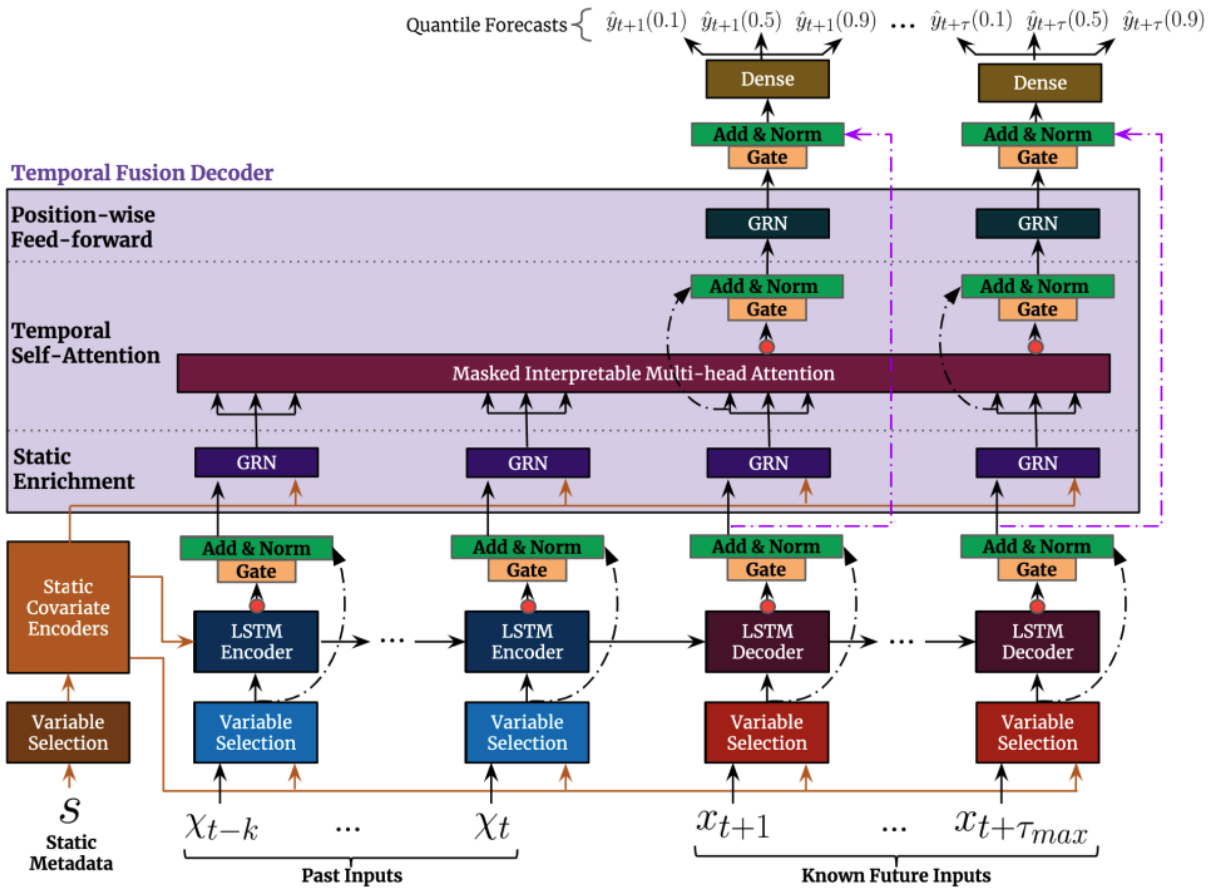

Figure 1. The model architecture (Lim et al., 2021)

The learned input variable transformations are then fed into the VSN to generate variable selection weights. To illustrate, let $\tilde{\boldsymbol{\xi}}_t^{(j)} \in \mathbb{R}^{d_{model}}$ denote the transformed input of the j^{th} raw input variable at time step t, and $\boldsymbol{\Xi}_t = \left[\tilde{\boldsymbol{\xi}}_t^{(1)^T}, \ldots, \tilde{\boldsymbol{\xi}}_t^{(m_x)^T}\right]^T$ be the flattened concatenation of all transformed input variables at time step t. The VSN takes $\boldsymbol{\Xi}_t$ and an external context vector $\boldsymbol{c}_s$ — which is obtained from a static input variable encoder and explained below— as inputs and feeds them through a Gated Residual Network (GRN) followed by a Softmax layer to generate the variable selection weights denoted by $\boldsymbol{v}_{x_t} \in \mathbb{R}^{m_x}$ and $\boldsymbol{v}_s \in \mathbb{R}^{m_s}$. For the time-varying input variables $\boldsymbol{x}_t \in \mathbb{R}^{m_x}$, for example, Eq. (2) generates the corresponding VSN weights $\boldsymbol{v}_{x_t} \in \mathbb{R}^{m_x}$.

$$\boldsymbol{v}_{x_t} = \text{Softmax}\left(\text{GRN}_{v_x}(\boldsymbol{\Xi}_t, \boldsymbol{c}_s)\right) \tag{2}$$

The motivation for using GRN is to provide the model with the flexibility to adaptively control the degree of nonlinear processing as needed. In general, given a primary input $\boldsymbol{a}$ and an optional context vector $\boldsymbol{c}$, which refer respectively to $\boldsymbol{\Xi}_t$ and $\boldsymbol{c}_s$ for the GRN intended for time-varying variable selection in Eq. (2), a GRN is presented in Eqs. (3)-(6). For samples with no optional context vector, the GRN treats the context input as zero, i.e., $\boldsymbol{c} = 0$. The GRN's gating layers are based on Gated Linear Units (GLU), which provide the flexibility to suppress any parts of the model architecture that are not needed for a given input variable. More clearly, GLU allows the model to control the extent to which the GRN contributes to the primary input $\boldsymbol{a}$, potentially skipping over the layer entirely, if necessary, since the GLU outputs could be all close to zero in order to suppress the nonlinear contribution (Lim et al., 2021).

$$\mathrm{GRN}_{\omega}(\boldsymbol{a}, \boldsymbol{c}) = \mathrm{LayerNorm}\big(\boldsymbol{a} + \mathrm{GLU}(\boldsymbol{\eta}_1)\big) \tag{3}$$

$$\boldsymbol{\eta}_1 = \boldsymbol{W}_{1,\omega}\, \boldsymbol{\eta}_2 + \boldsymbol{b}_{1,\omega} \tag{4}$$

$$\boldsymbol{\eta}_2 = \mathrm{ELU}\big(\boldsymbol{W}_{2,\omega}\, \boldsymbol{a} + \boldsymbol{W}_{3,\omega}\, \boldsymbol{c} + \boldsymbol{b}_{2,\omega}\big) \tag{5}$$

$$\mathrm{GLU}_{\omega}(\boldsymbol{\eta}_1) = \sigma\big(\boldsymbol{W}_{4,\omega}\, \boldsymbol{\eta}_1 + \boldsymbol{b}_{4,\omega}\big) \odot \big(\boldsymbol{W}_{5,\omega}\, \boldsymbol{\eta}_1 + \boldsymbol{b}_{5,\omega}\big) \tag{6}$$

wherein $\boldsymbol{\eta}_1 \in \mathbb{R}^{d_{model}}$ and $\boldsymbol{\eta}_2 \in \mathbb{R}^{d_{model}}$ are intermediate layers, $\mathrm{LayerNorm}(.)$ is the standard layer normalization (Ba et al., 2016), and the subscript ω denotes weight sharing. The Exponential Linear Unit, $\mathrm{ELU}(.)$, is used as the activation function in Eq. (5), which would act as an identity function if $(.) \gg 0$ and would generate a constant output when $(.) \ll 0$, leading to linear layer behavior. $\boldsymbol{W}_{(.)} \in \mathbb{R}^{d_{model}} \times \mathbb{R}^{d_{model}}$ and $\boldsymbol{b}_{(.)}$ are respectively the weights and biases, $\odot$ is the element-wise (Hadamard) product, and $\sigma(.)$ is the sigmoid activation function that transfers its input argument $(.)$ into the [0,1] range.

The input variable selection weights $\boldsymbol{v}_{x_t}$ (and $\boldsymbol{v}_s$) generated through Eqs. (2)-(6) are then used to yield the representation vector for each input variable. For this purpose, each transformed input variable $\boldsymbol{\xi}_t^{(j)}$ will first undergo an additional layer of nonlinear processing by feeding $\boldsymbol{\xi}_t^{(j)}$ through its own GRN to yield its corresponding processed variable $\tilde{\boldsymbol{\xi}}_t^{(j)}$, as shown in Eq. (7). The processed input variables are then aggregated in Eq. (8) using their corresponding variable selection weights, yielding $\tilde{\boldsymbol{\xi}}_t$ as the output of the VSN.

$$\tilde{\boldsymbol{\xi}}_t^{(j)} = \mathrm{GRN}_{\tilde{\xi}(j)}\left(\boldsymbol{\xi}_t^{(j)}\right) \tag{7}$$

$$\tilde{\boldsymbol{\xi}}_t = \sum_{j=1}^{m_x} v_{x_t}^{(j)}\, \tilde{\boldsymbol{\xi}}_t^{(j)} \tag{8}$$

where $v_{x_t}^{(j)}$ denotes the jth element of the variable selection weight vector $\boldsymbol{v}_{x_t}$.

Finally, the optional context vector used in Eq. (3) is obtained as $\boldsymbol{c}_s = \mathrm{GRN}_{c_s}(\tilde{\boldsymbol{\xi}})$, wherein $\tilde{\boldsymbol{\xi}}$ is the output of the variable selection network for static variable.

2.2. Short-Term Temporal Processing Using LSTM Sequence-To-Sequence Encoders and Decoders

Given the inherent time-ordering of time series data, temporal processing on the output of the Variable Selection Network (VSN) introduced in the previous section (i.e., $\tilde{\boldsymbol{\xi}}_t$) is indispensable to learn the temporal dependencies among $\tilde{\boldsymbol{\xi}}_t$'s in neighboring points in time. To this end, a sequence-to-sequence layer is employed, which is particularly suited for mapping an input sequence (i.e., past input values $\tilde{\boldsymbol{\xi}}_{t-k:t}$) to an output sequence (i.e., future input values $\tilde{\boldsymbol{\xi}}_{t+1:t+\tau_{max}}$) when the lengths of the past and future sequences (i.e., look-back window k and look-ahead window τ_{max}, respectively) are different. The sequence-to-sequence layer involves an encoder, which processes the input sequence $\tilde{\boldsymbol{\xi}}_{t-k:t}$, and a decoder, which usually takes the final hidden state of the encoder in order to generate the output sequence $\tilde{\boldsymbol{\xi}}_{t+1:t+\tau_{max}}$. In this project, the sequence-to-sequence layer employs long short-term memory (LSTM) encoders and decoders.

LSTM has a multi-layer structure encompassing an input layer, recurrent hidden layers, and an output layer. Unlike the standard Recurrent Neural Network (RNN) that simply applies an element-wise nonlinearity to the affine transformation of the input layer and the recurrent hidden layer, LSTM uses "cells" that are connected recurrently to each other, replacing the usual hidden layers of standard RNN. The LSTM cell has three gating units, comprising the forget gate (f_t), input gate (i_t), and output gate (o_t) at time step t, along with an internal "state" unit ($\boldsymbol{z}_t$) that collectively control the flow of information over time. For each cell, the inputs include the current input variables $\tilde{\boldsymbol{\xi}}_t$, previous hidden units $\boldsymbol{h}_{t-1}$, and previous cell state $\boldsymbol{z}_{t-1}$. The cell's output is $\boldsymbol{h}_t$ which is feed forwarded to the next LSTM cell together with the updated state $\boldsymbol{z}_t$. In each cell, the forget gate decides which historical information to pass to the state unit. Through the input gate, new information can be added and replace the outdated ones. Overall, the LSTM model is specified in Eqs. (9)-(13).

$$f_t = \sigma\left(\boldsymbol{W}_{f,\tilde{\xi}}\,\tilde{\boldsymbol{\xi}}_t + \boldsymbol{W}_{f,h}\,\boldsymbol{h}_{t-1} + b_f\right) \tag{9}$$

$$i_t = \sigma\left(\boldsymbol{W}_{i,\tilde{\xi}}\,\tilde{\boldsymbol{\xi}}_t + \boldsymbol{W}_{i,h}\,\boldsymbol{h}_{t-1} + b_i\right) \tag{10}$$

$$o_t = \sigma\left(\boldsymbol{W}_{o,\tilde{\xi}}\,\tilde{\boldsymbol{\xi}}_t + \boldsymbol{W}_{o,h}\,\boldsymbol{h}_{t-1} + b_o\right) \tag{11}$$

$$\boldsymbol{h}_t = o_t \odot \tanh(\boldsymbol{z}_t) \tag{12}$$

$$\boldsymbol{z}_t = f_t \odot \boldsymbol{z}_{t-1} + i_t \odot \tanh\left(\boldsymbol{W}_{z,\tilde{\xi}}\, \tilde{\boldsymbol{\xi}}_t + \boldsymbol{W}_{z,h}\, \boldsymbol{h}_{t-1} + b_z\right) \tag{13}$$

where $\sigma(.)$ is the sigmoid activation function and $\tanh(.)$ is the hyperbolic tangent activation function that squashes input values $(.)$ to the range between -1 and 1.

The output of the LSTM sequence-to-sequence layer is a set of temporal variables $\phi(t,n)$, where $n \in [-k, \tau_{max}]$ is the time position index. As depicted in Figure 1, a gating layer is utilized on top of the LSTM sequence-to-sequence layer, which yields $\tilde{\phi}(t,n)$ in Eq. (14).

$$\tilde{\boldsymbol{\phi}}(t,n) = \text{LayerNorm}\left(\tilde{\boldsymbol{\xi}}_{t+n} + \text{GLU}_{\tilde{\phi}}\left(\boldsymbol{\phi}(t,n)\right)\right) \tag{14}$$

2.3. Long-Term Temporal Processing Using Interpretable Multi-Head Attention

The LSTM sequence-to-sequence layer presented in the previous section can infer temporal dependencies, yet it may still fail to pick up long-term temporal dependencies due to the notorious vanishing gradient problem (Bengio et al., 1994). To tackle, an interpretable multi-head attention layer is placed on top of the LSTM sequence-to-sequence layer in order to integrate information from any time step using dynamically generated weights, allowing the model to focus on significant time steps in the past, regardless of how far back in the look-back window they are (Lim et al., 2021; Vaswani et al., 2017).

Before feeding into the attention layer, the output of the LSTM sequence-to-sequence layer is first fed through a static enrichment layer (see Figure 1) specified in Eq. (15) in order to enhance the learned short-term temporal information (i.e., $\tilde{\boldsymbol{\phi}}(t,n)$ in Eq. (14)) with static metadata.

$$\boldsymbol{\theta}(t,n) = \text{GRN}_{\theta}\left(\tilde{\boldsymbol{\phi}}(t,n), \boldsymbol{c}_e\right) \tag{15}$$

where $\boldsymbol{c}_e$ is an optional context vector obtained in an analogous way to the context vector described in the VSN section.

The learned outputs from the static enrichment layer are then used as input to the interpretable multi-head attention layer. To illustrate, let $\boldsymbol{\Theta}(t) = [\boldsymbol{\theta}(t,-k), \ldots, \boldsymbol{\theta}(t,\tau_{max})]^T$ denote the matrix of static-enriched temporal information $\boldsymbol{\theta}(t,n)$. Given $\boldsymbol{\Theta}(t)$, the interpretable multi-head attention layer yields $\boldsymbol{B}(t) = [\boldsymbol{\beta}(t,-k), \ldots, \boldsymbol{\beta}(t,\tau_{max})]^T$, as shown in Eq. (16) and described below.

$$\boldsymbol{B}(t) = \text{InterpretableMultihead}\left(\boldsymbol{\Theta}(t), \boldsymbol{\Theta}(t), \boldsymbol{\Theta}(t)\right) \tag{16}$$

Conceptually, the standard attention layer (Vaswani et al., 2017) shown in Eq. (17) is a mechanism for a Key $\boldsymbol{K} \in \mathbb{R}^{N \times d_{attn}}$ lookup based on a given Query $\boldsymbol{Q} \in \mathbb{R}^{N \times d_{attn}}$ in order to scale Values (i.e., input variables of the attention layer) $\boldsymbol{V} \in \mathbb{R}^{N \times d_V}$, where N is the total number of past and future time steps fed into the attention layer (i.e., $N = \tau_{max} + k + 1$), $d_{attn} = d_V = d_{model}/m_H$, and m_H is the number of attention heads as illustrated below. In Eq. (17), $\text{Attention}(.,.,.)$ represents the context vector output of the standard attention layer and the normalization function $A(.,.)$ gives the attention weights using Eq. (18).

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = A(\boldsymbol{Q}, \boldsymbol{K})\, \boldsymbol{V} \tag{17}$$

$$A(\boldsymbol{Q}, \boldsymbol{K}) = \text{Softmax}(\boldsymbol{Q}\, \boldsymbol{K}^T / d_{attn}) \tag{18}$$

The learning capacity of the standard attention mechanism can be enhanced using multi-head attention (Vaswani et al., 2017), which allows for attending to various parts of the input sequence differently. This is accomplished by performing several independent standard attention mechanisms, each referred to as a head $h \in \{1, \dots, m_H\}$, and then concatenating the separate attention outputs from all heads $\left[\boldsymbol{H}_1, \dots, \boldsymbol{H}_{m_H}\right]$, as shown in Eqs (19)-(20).

$$\text{Multihead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \left[\boldsymbol{H}_1, \dots, \boldsymbol{H}_{m_H}\right] \boldsymbol{W}_H \tag{19}$$

$$\boldsymbol{H}_h = \text{Attention}\left(\boldsymbol{Q}\, \boldsymbol{W}_Q^{(h)}, \boldsymbol{K}\, \boldsymbol{W}_K^{(h)}, \boldsymbol{V}\, \boldsymbol{W}_V^{(h)}\right) \tag{20}$$

where $\boldsymbol{W}_Q^{(h)} \in \mathbb{R}^{d_{model} \times d_{attn}}$, $\boldsymbol{W}_K^{(h)} \in \mathbb{R}^{d_{model} \times d_{attn}}$, and $\boldsymbol{W}_V^{(h)} \in \mathbb{R}^{d_{model} \times d_V}$ are respectively the weights for queries, keys, and values corresponding to head h.

The multi-head attention weights $A(.,.)$ can be made interpretable through a slight modification of the multi-head attention mechanism. For this purpose, note that the multi-head attention projects the Values $\boldsymbol{V}$ differently for each head, as signified by the head-specific weights for Values, i.e., $\boldsymbol{W}_V^{(h)}$. In the modified multi-head attention, referred to as the interpretable multi-head attention and shown in Eqs. (21)-(22), the projected Values are shared across all heads, resulting in a common weight for all Values, i.e., $\boldsymbol{W}_V$. With this modification, different heads only capture the interactions between Queries $\boldsymbol{Q}$ and Keys $\boldsymbol{K}$, and the outputs of all heads are concatenated into $\tilde{A}(\boldsymbol{Q}, \boldsymbol{K})$ before multiplying by the Values $\boldsymbol{V}$, according to Eq. (22). In other words, separate heads can learn distinct temporal patterns as suggested by the head-specific modified attention weights $\tilde{A}(\boldsymbol{Q}, \boldsymbol{K})$, while equally attending to the Values $\boldsymbol{V}$. $\tilde{A}(\boldsymbol{Q}, \boldsymbol{K})$ allows for model interpretability through identifying the time steps that most significantly influence the

model's outputs.

$$\text{InterpretableMultihead}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \tilde{\boldsymbol{H}}\,\boldsymbol{W}_H \tag{21}$$

$$\tilde{\boldsymbol{H}} = \tilde{A}(\boldsymbol{Q},\boldsymbol{K})\,\boldsymbol{V}\,\boldsymbol{W}_V$$

$$= \left\{\frac{1}{m_H}\sum_{h=1}^{m_H} A\left(\boldsymbol{Q}\,\boldsymbol{W}_Q^{(h)}, \boldsymbol{K}\,\boldsymbol{W}_K^{(h)}\right)\right\}\boldsymbol{V}\,\boldsymbol{W}_V$$

$$= \frac{1}{m_H}\sum_{h=1}^{m_H} \text{Attention}\left(\boldsymbol{Q}\,\boldsymbol{W}_Q^{(h)}, \boldsymbol{K}\,\boldsymbol{W}_K^{(h)}, \boldsymbol{V}\,\boldsymbol{W}_V^{(h)}\right) \tag{22}$$

To further improve the learning capacity of the model, the outputs of the interpretable multi-head attention (i.e., $\boldsymbol{B}(t) = [\boldsymbol{\beta}(t,-k),\dots,\boldsymbol{\beta}(t,\tau_{max})]^T$ in Eq. (16)) are further processed nonlinearly by feeding them through three additional gating layers, as specified in Eqs. (23)-(25).

$$\boldsymbol{\delta}(t,n) = \text{LayerNorm}\left(\boldsymbol{\theta}(t,n) + \text{GLU}_{\delta}\left(\boldsymbol{\beta}(t,n)\right)\right) \tag{23}$$

$$\boldsymbol{\psi}(t,n) = \text{GRN}_{\psi}\left(\boldsymbol{\delta}(t,n)\right) \tag{24}$$

$$\tilde{\boldsymbol{\psi}}(t,n) = \text{LayerNorm}\left(\tilde{\boldsymbol{\phi}}(t,n) + \text{GLU}_{\tilde{\psi}}\left(\boldsymbol{\psi}(t,n)\right)\right) \tag{25}$$

2.4. Loss Function

At each forecast time t, the target variable's prediction in time step $(t+\tau)$ is generated using a linear transformation of the output of the model's last layer (i.e., $\tilde{\boldsymbol{\psi}}(t,\tau)$ as obtained in Eq. (25)). Note that prediction intervals for different quantiles q (i.e., 5$^{\text{th}}$, 50$^{\text{th}}$, and 95$^{\text{th}}$ percentiles) are simultaneously provided in Eq. (26).

$$\hat{y}_{t+\tau}(q) = \boldsymbol{W}_q\,\tilde{\boldsymbol{\psi}}(t,\tau) + b_q \tag{26}$$

The model is trained by minimizing the following loss function, which roughly measures the difference between the predicted and observed values of the target variable, accumulated over all quantiles.

$$\mathcal{L}(\Omega,\boldsymbol{W}) = \sum_{y_t\in\Omega}\sum_{q\in Q}\sum_{\tau=1}^{\tau_{max}} \frac{QL\left(y_t, \hat{y}_{(t-\tau)+\tau}(q), q\right)}{M\,\tau_{max}} \tag{27}$$

$$QL(y,\hat{y},q) = q\,\max(0, y-\hat{y}) + (1-q)\,\max(0,\hat{y}-y) \tag{28}$$

Chapter 3: Data

This chapter provides a summary description of the vehicle trajectory dataset used for implementing the presented deep learning model. The presented deep learning model is trained using the Next Generation Simulation (NGSIM) vehicle trajectory big data on a portion of eastbound I-80 in San Francisco, California, which were collected by the Federal Highway Administration (FHWA) in April 2005 and updated in June 2020.

The dataset provides the precise longitudinal and lateral positions of each vehicle relative to other vehicles within the study area at a high temporal resolution every one-tenth of a second (Traffic Analysis Tools: Next Generation Simulation - FHWA Operations). The length of the study area (Figure 2) is approximately 500 meters (1,640 feet), consisting of six freeway lanes. A total of 45 minutes of data are available in the full dataset, originally segmented into three 15-minute periods (16:00-16:15, 17:00-17:15, and 17:15-17:30). The three segments represent the transition periods between different traffic congestion levels. We use the full dataset (45 minutes) for training, validating, and testing the presented deep learning model, which amounts to 4,122,130 data records describing the instantaneous movements of nearly 3,000 vehicles.

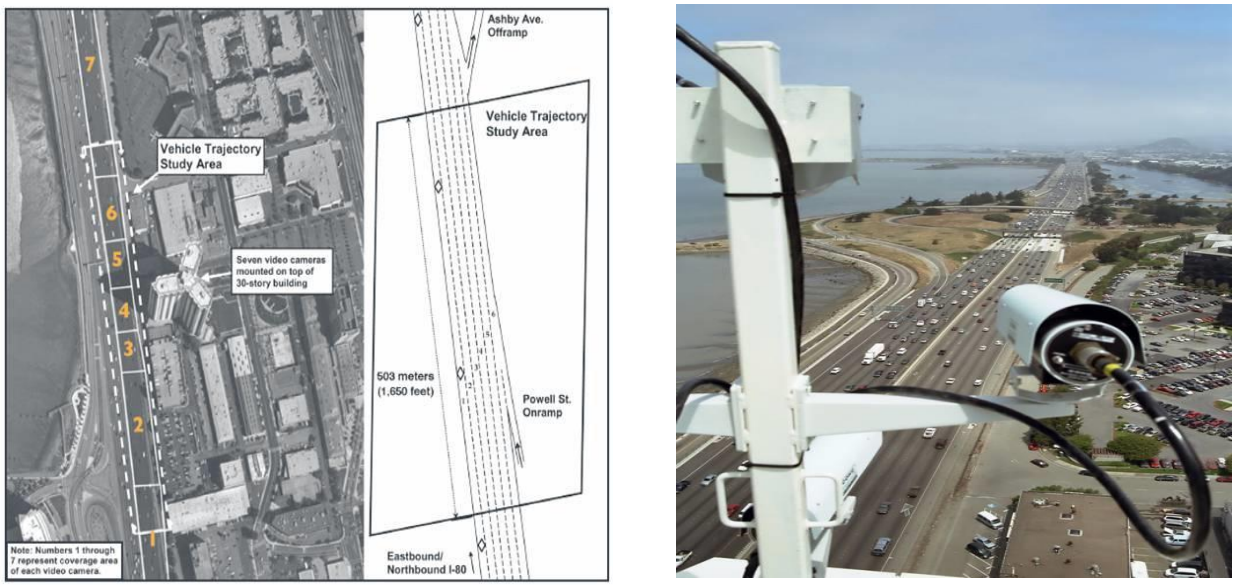

Figure 2. Aerial view of the vehicle trajectory study area (left) and a digital video camera mounted on top of a building recording vehicle trajectory data in I-80 (right) (Traffic Analysis Tools: Next Generation Simulation - FHWA Operations)

Besides using the vehicle trajectory data of highway I-80 for training, validating, and testing the model, data from another highway on southbound US-101 (also known as the Hollywood Freeway) in Los Angeles, California, are also utilized to further test the model fit on unseen data.

This also helps to evaluate the generalizability and transferability of the trained model. The US-101 dataset comprises 4,398,832 records and represents the unique information of approximately 3,500 vehicles.

The raw vehicle trajectory data should be first preprocessed before feeding into the model. This starts with data cleaning, which involves the removal of noises and errors often found in any raw data. Then, a lower bound of 1 second is set for vehicle headway and the data records having shorter time headways are discarded. This is considered as a safety constraint so that the model can be useful in downstream applications of vehicle headway prediction, such as collision avoidance and CAV trajectory planning.

Once data preprocessing is completed, the whole I-80 dataset is split into three groups for model training, validation, and testing purposes. Specifically, 65% of the dataset is used for model training and 15% is used for validation, while the remaining 20% is utilized for testing the prediction accuracy of the trained model. To this end, the trajectories of all vehicles are first sorted based on vehicle ID, reshaping into a multi-index dataset, given the time series nature of data for each recorded vehicle. The unique ID of each vehicle and the time index of each data record corresponding to the vehicle are used as two indices to sort the data. This ensures that the trajectory data of all vehicles are considered in the training, validation, and testing datasets. Once sorted, the data records of each vehicle are split into the training, validation, and test sets.

Chapter 4: Results

This chapter presents the results of model implementation, specifically discussing the model performance, model interpretability, and benchmark comparison

4.1. Model Performance

Figure 3 exhibits the frequency distributions of the ground truth vehicle headways (denoted as "target") and predicted headways (denoted as "p50" for the 50th percentile prediction interval) in highway I-80. The prediction look-ahead window is 20 time steps into the future, which is equivalent to 2 seconds noting that each time step is 0.1 second. As shown, the predicted headway distribution closely matches that of the ground truth headways, indicating a good model fit. Moreover, it can be verified that both the predicted and ground truth headways follow the right-skewed log-normal distribution, which is not surprising as the right-skewness of the headway distribution is known to be rooted in the microscopic car-following behavior (Li and Chen, 2017).

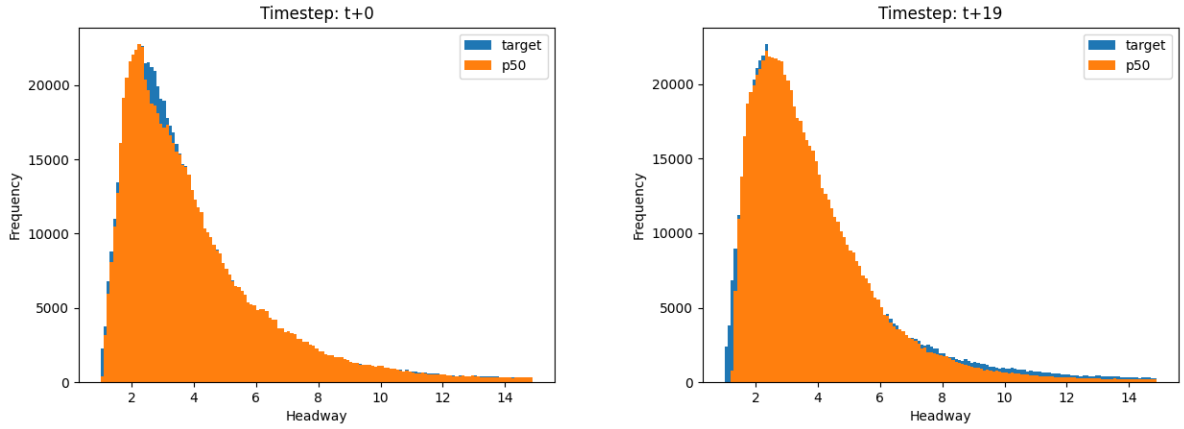

Figure 3. Predicted and target (ground truth) vehicle headway distributions in highway I-80

Figure 4 plots the evolution of the vehicle headway over time for a random vehicle (vehicle ID = 2986). As shown, the predicted headways are close to the ground truth values. In addition, it is seen that the provided prediction intervals (i.e., 5th, 50th, and 95th percentile) can accurately identify the likely best- and worst-case values for the vehicle headway over time.

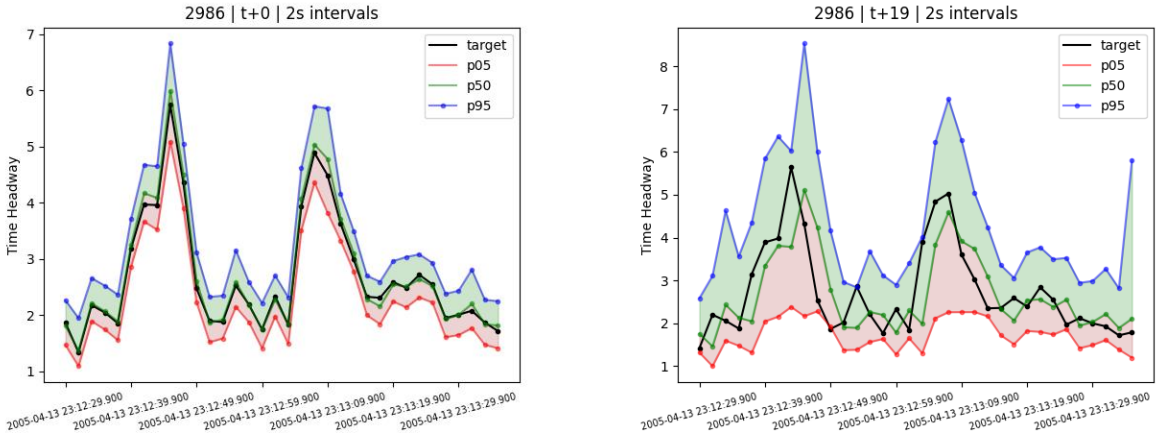

Figure 4. Headway evolution over time for a random vehicle (Notes: "target" refers to the ground truth headways; "p05", "p50", and "p95" denote the predicted headways at the 5th, 50th, and 90th prediction intervals)

4.2. Model Interpretability

In this project, we consider that the mesoscopic-scale vehicle headway at a time step may be affected by the vehicle headway at previous time steps as well as eight input variables encompassing 1) microscopic traffic measures, including the instantaneous speeds of the subject vehicle and preceding vehicle, acceleration of the subject vehicle, and lane position of the subjective vehicle, 2) macroscopic traffic measures, including traffic flow rate, and 3) vehicle class, length, and width. The importance of each of these variables in terms of the extent of contribution to the vehicle headway prediction are calculated using the variable selection weights $\boldsymbol{v}_{x_t}$ described in Eq. (2). Results are shown in TABLE 1, highlighting the significant effects of the vehicle headway in previous time steps and the vehicle length on the future vehicle headway values.

TABLE 1. Model interpretation

Input variable	Variable importance	Input variable	Variable importance
Subject vehicle velocity	0.037	**Subject vehicle length**	**0.431**
Subject vehicle acceleration	0.120	Subject vehicle width	0.091
Preceding vehicle's velocity	0.069	Subject vehicle class 2	0.052
Flow rate	0.010	Subject vehicle class 3	0.426
Lane 2	0.036		
Lane 3	0.044		
Lane 4	0.061		
Lane 5	0.010		
Lane 6	0.012		
Lane 7	0.008		
Subject vehicle headway	**0.569**		

4.3. Benchmark Comparison

To further evaluate the model performance, we compare the goodness-of-fit of the proposed model, in terms of the mean absolute error (MAE) calculated as $MAE = (\sum_{i=1}^{N}|\hat{y} - y|)/N$, with two state-of-the-art benchmark deep learning models, namely a deep autoregressive model (denoted as "DeepAR" (Salinas et al., 2020)) and a sequence-to-sequence model (denoted as "Seq-2-Seq"), using the same training data. Notably, the much smaller MAE values of the presented TFT model serve as a clear indication that it outperforms the other two benchmarks.

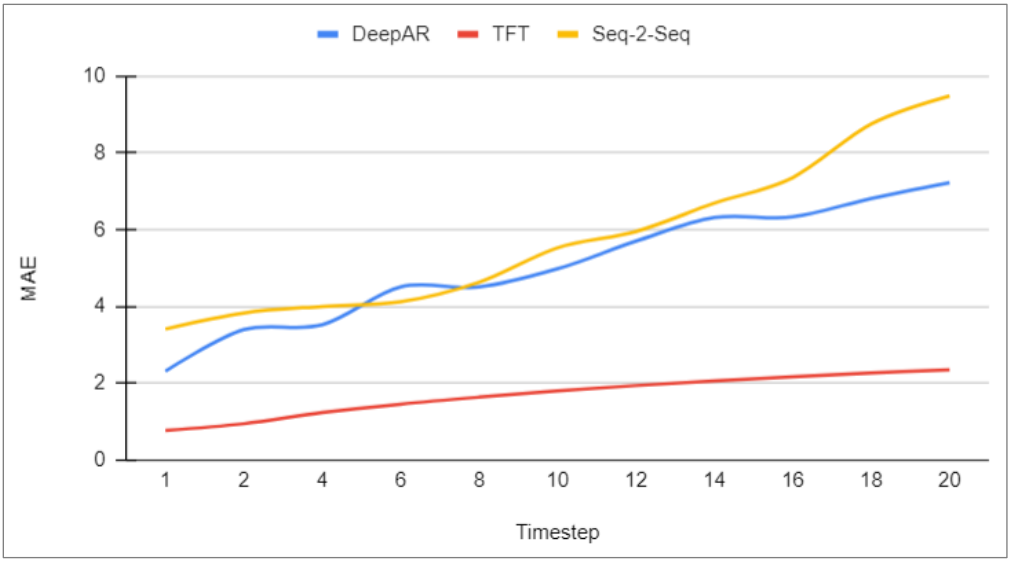

Figure 5. Model comparison against two state-of-the-art benchmark deep learning models for multi-step-ahead time series forecasting

Chapter 5: Conclusions

This project contributes to the traffic flow studies through the application of a novel interpretable deep learning model to empirical vehicle trajectory data in order to predict the vehicle headway over multiple time steps into the future. Compared to the existing statistical probability distribution models of time headway, the presented deep learning model can directly "learn" the complex and highly nonlinear headway distribution from the data, rather than relying on a closed-form mathematical model. This allows to overcome the common issues in headway modeling stemming from the heterogenous influence of various random factors on headway. Of note among such factors is traffic congestion, which causes the (mesoscopic) headway distribution to depend on the (microscopic) instantaneous speeds of vehicles. In addition, mixed traffic conditions comprising multiple vehicle classes (e.g., cars and trucks) give rise to heterogenous headway distributions along a roadway and across traffic lanes. Also, human drivers have distinct perceptual reactions and desired headways (i.e., inter-driver heterogeneity) and even the same driver may behave differently over time and space (i.e., intra-driver heterogeneity), leading to distinct headway distributions with different variances. Last but not least is the asymmetric driving behavior, which refers to the preference of drivers in maintaining short headways rather than long headways, resulting in rightly-skewed headway distributions.

Compared to recurrent deep networks, the proposed deep learning model can overcome the issue of back-propagated error decay (a.k.a. vanishing gradient problem), thus can exhibit superior capability for multi-step-ahead vehicle headway time series prediction with long temporal dependency. The model is trained and tested using the vehicle trajectory data from the USDOT's Next Generation Simulation (NGSIM) dataset. We investigate eight input features influencing vehicle headway, including both microscopic traffic measures (i.e., instantaneous speeds of the subject and leading vehicles, acceleration of the subject vehicle, and lane position of the subjective vehicle) and macroscopic traffic flow rate, as well as vehicle information (i.e., vehicle class, length, and width).

References

Ba, J.L., Kiros, J.R., Hinton, G.E., 2016. Layer Normalization.

Barfield, W., Dingus, T.A., 2014. Human factors in intelligent transportation systems. Psychology Press.

Bengio, Y., Simard, P., Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks 5, 157–166. https://doi.org/10.1109/72.279181

Bian, Y., Zheng, Y., Ren, W., Li, S.E., Wang, J., Li, K., 2019. Reducing time headway for platooning of connected vehicles via V2V communication. Transportation Research Part C: Emerging Technologies 102, 87–105. https://doi.org/10.1016/j.trc.2019.03.002

Chen, X., Li, L., Zhang, Y., 2010. A Markov Model for Headway/Spacing Distribution of Road Traffic. IEEE Transactions on Intelligent Transportation Systems 11, 773–785. https://doi.org/10.1109/TITS.2010.2050141

Gal, Y., Ghahramani, Z., 2016. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks, in: Advances in Neural Information Processing Systems. Curran Associates, Inc.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Gu, Y., Lu, W., Qin, L., Li, M., Shao, Z., 2019. Short-term prediction of lane-level traffic speeds: A fusion deep learning model. Transportation Research Part C: Emerging Technologies 106, 1–16. https://doi.org/10.1016/j.trc.2019.07.003

Highway Capacity Manual, 2010. Transportation Research Board of the National Academies.

Hochreiter, S., Schmidhuber, J., 1997. Long Short-Term Memory. Neural Computation 9, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Huang, X., Sun, Jie, Sun, Jian, 2018. A car-following model considering asymmetric driving behavior based on long short-term memory neural networks. Transportation Research Part C: Emerging Technologies 95, 346–362. https://doi.org/10.1016/j.trc.2018.07.022

Ke, J., Zheng, H., Yang, H., Chen, X. (Michael), 2017. Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. Transportation Research Part C: Emerging Technologies 85, 591–608. https://doi.org/10.1016/j.trc.2017.10.016

Li, L., Chen, X. (Michael), 2017. Vehicle headway modeling and its inferences in macroscopic/microscopic traffic flow theory: A survey. Transportation Research Part C: Emerging Technologies 76, 170–188. https://doi.org/10.1016/j.trc.2017.01.007

Li, L., Jiang, R., He, Z., Chen, X. (Michael), Zhou, X., 2020. Trajectory data-based traffic flow studies: A revisit. Transportation Research Part C: Emerging Technologies 114, 225–240. https://doi.org/10.1016/j.trc.2020.02.016

Lim, B., Arık, S.Ö., Loeff, N., Pfister, T., 2021. Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. International Journal of Forecasting 37, 1748–1764. https://doi.org/10.1016/j.ijforecast.2021.03.012

Lin, L., Gong, S., Peeta, S., Wu, X., 2021. Long Short-Term Memory-Based Human-Driven Vehicle Longitudinal Trajectory Prediction in a Connected and Autonomous Vehicle Environment. Transportation Research Record 2675, 380–390. https://doi.org/10.1177/0361198121993471

Salinas, D., Flunkert, V., Gasthaus, J., Januschowski, T., 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. International Journal of Forecasting 36, 1181–1191. https://doi.org/10.1016/j.ijforecast.2019.07.001

Simoncini, M., Taccari, L., Sambo, F., Bravi, L., Salti, S., Lori, A., 2018. Vehicle classification from low-frequency GPS data with recurrent neural networks. Transportation Research Part C: Emerging Technologies 91, 176–191. https://doi.org/10.1016/j.trc.2018.03.024

Taylor, J., Zhou, X., Rouphail, N.M., Porter, R.J., 2015. Method for investigating intradriver heterogeneity using vehicle trajectory data: A Dynamic Time Warping approach. Transportation Research Part B: Methodological 73, 59–80. https://doi.org/10.1016/j.trb.2014.12.009

Traffic Analysis Tools: Next Generation Simulation - FHWA Operations [WWW Document], n.d. URL https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm (accessed 7.31.21).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is All you Need, in: Advances in Neural Information Processing Systems. Curran Associates, Inc.

Xie, D.-F., Fang, Z.-Z., Jia, B., He, Z., 2019. A data-driven lane-changing model based on deep learning. Transportation Research Part C: Emerging Technologies 106, 41–60. https://doi.org/10.1016/j.trc.2019.07.002

Xu, C., Ji, J., Liu, P., 2018. The station-free sharing bike demand forecasting with a deep learning approach and large-scale datasets. Transportation Research Part C: Emerging Technologies 95, 47–60. https://doi.org/10.1016/j.trc.2018.07.013

Yang, B., Sun, S., Li, J., Lin, X., Tian, Y., 2019. Traffic flow prediction using LSTM with feature enhancement. Neurocomputing 332, 320–327. https://doi.org/10.1016/j.neucom.2018.12.016

Yang, S., Ma, W., Pi, X., Qian, S., 2019. A deep learning approach to real-time parking occupancy prediction in transportation networks incorporating multiple spatio-temporal data sources.

Transportation Research Part C: Emerging Technologies 107, 248–265. https://doi.org/10.1016/j.trc.2019.08.010

Zhang, G., Wang, Y., 2014. A Gaussian Kernel-Based Approach for Modeling Vehicle Headway Distributions. Transportation Science 48, 206–216. https://doi.org/10.1287/trsc.1120.0451

Zhang, X., Sun, Jie, Qi, X., Sun, Jian, 2019. Simultaneous modeling of car-following and lane-changing behaviors using deep learning. Transportation Research Part C: Emerging Technologies 104, 287–304. https://doi.org/10.1016/j.trc.2019.05.021

Zhang, Z., He, Q., Gao, J., Ni, M., 2018. A deep learning approach for detecting traffic accidents from social media data. Transportation Research Part C: Emerging Technologies 86, 580–596. https://doi.org/10.1016/j.trc.2017.11.027