Transportation Research Part C 160 (2024) 104525

Contents lists available at ScienceDirect

Transportation Research Part C

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

On the string stability of neural network-based car-following models: A generic analysis framework

Xiaohui Zhang [a], Jie Sun [a,b,*], Zuduo Zheng [b], Jian Sun [a,*]

[a] Department of Traffic Engineering & Key Laboratory of Road and Traffic Engineering of Ministry of Education, Tongji University, Shanghai, China
[b] School of Civil Engineering, The University of Queensland, St. Lucia 4072, Brisbane, Australia

ARTICLE INFO

Keywords:
String stability
Car-following
Neural network
Numerical experiment
MLP
LSTM

ABSTRACT

String stability plays a crucial role in regulating traffic flow, as traffic oscillation can be triggered by string instability in the car-following (CF) behavior. Although studies over the past decades have provided various methods for analyzing string stability of analytical CF models, no studies have focused on neural network (NN) based CF models despite the fact that these models have exhibited remarkable performance in learning realistic driving behavior in the recent literature. This paper fills this gap by proposing a generic theoretical framework for analyzing the string stability of NN-based CF models through an Estimation-Approximation-Derivation-Calculation process (referred to as EADC framework). Within the framework, we first estimate the steady states of NN-based CF models by solving the corresponding optimization model and obtain the smooth approximation of the NN-based models for linearization, based on which the transfer function is constructed. We then derive the general stability criteria for two commonly used classes of NN in CF modeling, i.e., feedforward NN-based CF models with basic input and recurrent NN-based CF models with multi-step historical information. The string stability is thus obtained by calculating the partial derivatives through the automatic differentiation method. As two case studies, we apply the proposed stability analysis framework on two typical NN-based CF models, the Mo-MLP model (Mo et al., 2021) and the Huang-LSTM model (Huang et al., 2018), and obtained the complete consistency between the theoretical results and the simulation results for both models, which demonstrates the soundness of the proposed EADC framework. Moreover, we discuss the applicability of the proposed EADC stability analysis framework in the emerging era of connected and autonomous vehicles and artificial intelligence.

1. Introduction

String stability is one of the key characteristics of car-following (CF) models, which is closely related to the onset of traffic oscillation, a commonly observed phenomenon in congested traffic (Chandler et al., 1958; Orosz et al., 2010; Sun et al., 2020; Treiber and Kesting, 2013). Small perturbations are dissipated while propagating through a string stable platoon of vehicles, while amplified in a string unstable platoon which has significantly negative effects on traffic operations, e.g., riding comfort, crash risk, and energy consumption (Ahn et al., 2010; Zheng et al., 2010). String stability analysis of CF models, which aims to study how small perturbations

* Corresponding authors at: Department of Traffic Engineering & Key Laboratory of Road and Traffic Engineering of Ministry of Education, Tongji University, Shanghai, China (Jie Sun).
E-mail addresses: sunj7917@gmail.com (J. Sun), sunjian@tongji.edu.cn (J. Sun).

https://doi.org/10.1016/j.trc.2024.104525
Received 13 April 2023; Received in revised form 31 October 2023; Accepted 9 February 2024
Available online 19 February 2024

evolve over a platoon when propagating along the vehicles, has been implemented since the early days of CF model development (Chandler et al., 1958; Herman et al., 1959; Wilson, 2008; Wilson and Ward, 2011). Several theoretical methods have been proposed for string stability analysis of different categories of CF models, i.e., basic CF models, CF models incorporating time delays, and CF models considering multiple leading vehicles (more details in the Literature Review section). By analyzing the string stability, we can understand the dynamics and characteristics of different CF models and how driving behaviors exemplified by such CF models affect the traffic flow. Moreover, new CF models or effective control strategies can be designed and evaluated to improve the string stability and mitigate traffic oscillations and congestions. However, as the concept of stability analysis is originated from the control theory where a dynamic system is usually defined by a set of ordinary differential equations, explicit functions and analytical tractability are required for existing stability analysis of CF models.

On the other hand, with the recently rapid advances in machine learning, many studies adopted neural network (NN) based approaches for modeling CF behavior rather than the traditional analytical approaches (Huang et al., 2018; Ye et al., 2019; Zhu et al., 2018), because NN-based approaches can be more powerful and flexible in modeling driving behaviors (e.g., capturing asymmetric driving behavior, stop/slow and go behavior). They have also been widely used in modeling connected or/and automated vehicles (Kiran et al., 2022; Mozaffari et al., 2022). However, despite the superior performance of NN-based CF models, their underlying stability characteristics are difficult to understand with their black-box nature (Rudin, 2019). Since the traditional stability analysis method requires explicit functions and analytical tractability of CF models, it is not applicable for NN-based CF models due to the fact that they do not have explicit functions. To address this issue, numerical or field experiments can be conducted by simulating a wide range of traffic scenarios (Hart et al., 2021; Ma et al., 2022), which is usually time-consuming and computationally expensive. Therefore, there is a great need to develop an analytical method which is capable of assessing the stability of NN-based CF models, and provide insights for the future development of autonomous driving with NN-based CF models to maintain string stable and thus ease traffic congestion. This is the primary motivation of this study.

The main difficulty of assessing NN-based CF models' stability lies in the fact that: (1) there is a lack of explicit and mathematical formulations of the NN-based CF models, which makes it difficult to obtain the steady states analytically; (2) a typical NN contains the non-differentiable (e.g., ReLU (Rectified Linear Unit) activation function) part, which impedes the linearization of the model for obtaining the stability criterion; and (3) even for the differentiable part, it is highly non-linear because of the multiple hidden layers structure and numerous complicated neurons in each layer, and thus its partial derivatives cannot be directly calculated.

Additionally, one important feature of NN-based models, particularly the prevailing recurrent NN (RNN) based models, is that they can incorporate multiple steps of historical information to embed long memory and thus improve the model performance, unlike traditional time-delayed CF models which only use one step of historical information (Sun et al., 2018). However, this advanced setting of NN-based models adds further complexity to the stability analysis of NN-based models, introducing additional dynamics and frequency dependence in the Laplace transform for transfer function derivation.

Because of these great challenges, the string stability of NN-based CF models has not been rigorously investigated. To the best of our knowledge, this work provides one of the first systematic investigations into the theoretical string stability analysis of NN-based models. Specifically, we aim to develop a generic framework to assess the string stability of different types of NN-based CF models theoretically by addressing the above-mentioned challenges through an ***Estimation*** (of steady states) -***Approximation*** (of non-differentiable functions) -***Derivation*** (of general stability criteria) -***Calculation*** (of partial derivatives) process (EADC framework). The major contributions of this EADC framework are:

- We convert the steady states estimation into an equivalent optimization problem, where the steady states are calculated by solving the optimization model.
- By replacing the non-differentiable activation functions with smooth approximation and linearizing NN-based CF models, we derive a set of general stability criteria for both the basic NN-based model and the advanced NN-based models that incorporate multi-step historical information.
- We employ the automatic differentiation method to facilitate the calculation of the partial derivative coefficients in the stability criteria for estimating the string stability.

As two case studies, we apply the proposed framework to two typical NN-based CF models: one is a basic NN-based CF model (Mo et al., 2021) and the other is an advanced NN-based CF model that incorporates multi-step historical information (Huang et al., 2018), and evaluate their stability under different traffic states (i.e., with different equilibrium speeds). The validity of the framework is then assessed by comparing the theoretical results with the simulation results from a series of numerical experiments. In addition, we discuss the applicability of the proposed EADC stability analysis framework in the emerging era of connected and automated vehicles (CAVs).

The rest of the paper is structured as follows: Section 2 reviews the related literature; Section 3 introduces the stability analysis framework for NN-based CF models, and derives the string stability criteria for two types of NN-based CF models; Section 4 applies the framework to two typical well-trained NN-based CF models; Section 5 assesses the framework's performance with numerical simulations; Section 6 discusses the applicability and applications of the proposed framework; and Section 7 summarizes the main conclusions of this work.

2. Literature review

Given the significant progress of both stability analysis and NN-based CF modeling, in this section, we briefly review representative

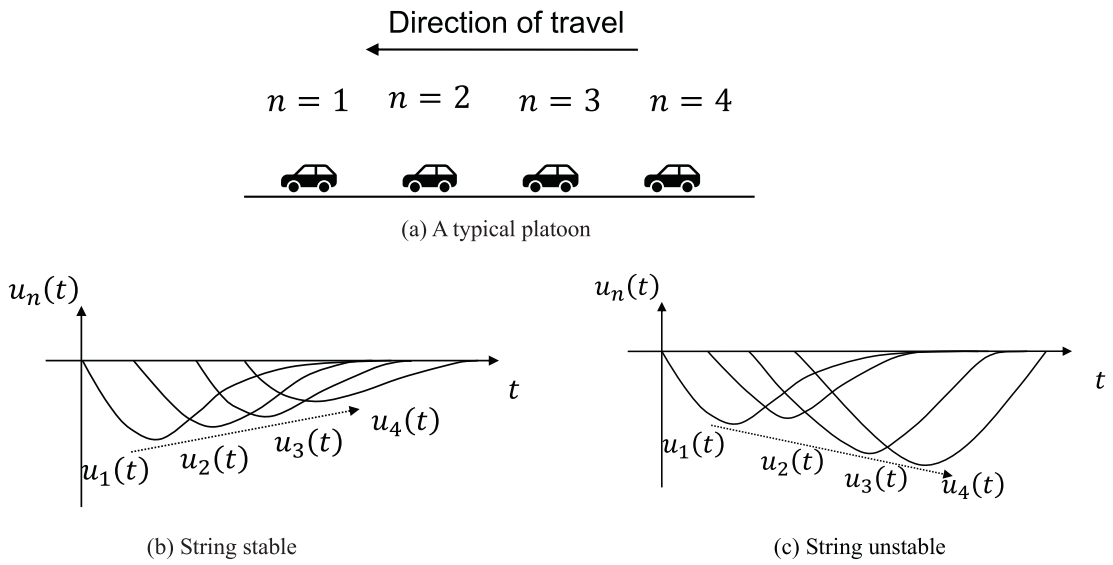


Fig. 1. Intuitive descriptions of string stability, where $u_n(t)$ denotes the state fluctuation of vehicle n at time t in terms of the steady state.

studies on string stability analysis for CF models and on NN-based CF models, respectively.

2.1. String stability analysis for CF models

Due to the close relationship with traffic oscillation, numerous studies have been conducted for string stability since the early days of CF model development. There are several different definitions of string stability in the field literature, here we adopt the commonly used definition of strict string stability for consistency. Intuitively, a platoon of vehicles (Fig. 1(a)) is considered string stable if the disturbance originating from the first vehicle strictly dampens over vehicles (Peppard, 1974; Sun et al., 2023; Treiber and Kesting, 2013), as shown in Fig. 1(b), and otherwise string unstable as shown in Fig. 1(c). Assume that $u_n(t)$ stands for the state fluctuation of vehicle n at time t in terms of the equilibrium state (e.g., $u_n(t) = v_n(t) - v_e$, the difference between velocity and equilibrium velocity), and $\|u_n\|_\infty = max_t|u_n(t)|$ is the maximum magnitude in the time domain for each lead-follower pair. So, the string stability is satisfied if:

$$\|u_1\|_\infty > \|u_2\|_\infty > \cdots > \|u_n\|_\infty \tag{1}$$

As mentioned in the Introduction, several notable methods borrowed from control theory were used to analyze the string stability for different types of analytical CF models (i.e., basic, time-delayed, and multi-anticipative/cooperative CF models) defined in Sun et al. (2018), namely, direct transfer function-based, Laplace transform-based and characteristic equation-based methods. Specifically, by calculating the magnitude of the transfer function using the Fourier ansatz, Chandler et al. (1958) applied the direct transfer function-based technique for the first time in string stability analysis for a relative speed control model and several linear time-delayed CF models. Bando et al. (1998) also employed this method to analyze stability of the time-delayed optimal velocity model (OVM). In addition, by constructing the transfer function between outputs and inputs with Laplace transform, the string stability is assessed by comparing the H-infinity norm of the transfer function with 1. This Laplace transform based method has been widely used for investigating the string stability of time-delayed CF models (Orosz et al., 2011) and cooperative CF models (Liu et al., 2001; Swaroop, 1997). Moreover, using the exponential ansatz to describe perturbations and then induce a characteristic equation, the characteristic equation-based method is a general method in string stability analysis for CF models. For example, Lenz et al. (1999) investigated the string stability of multi-anticipative OVM by combining the characteristic equation-based technique with the Routh-Hurwitz criterion. Sau et al. (2014) employed the root locus of the characteristic equation to illustrate the string stability criteria of basic and cooperative CF models. Monteil et al. (2014) also established the characteristic equation of a cooperative CF model. The mathematical details of these methods are reviewed in Sun et al. (2018).

However, despite significant progress has been made in the stability analysis methods of CF models, all the methods require explicit functions of the CF model, which are only applicable for analytical CF models. To assess the string stability of CF models without explicit functions like NN-based models, one way is to directly conduct numerical simulations or field experiments of a platoon with given disturbances and calculate the amplification rate, which is time-consuming and lacking in analytical tractability. Another way is to utilize the CF data (e.g., Open ACC dataset (Makridis et al., 2021)) that generated from the black-box type CF model to fit an analytical CF model (Gunter et al., 2021) or estimate the empirical transfer function (Zhou et al., 2023). However, this empirical method cannot obtain the explicit expression of stability criteria and can hardly guide the development of control strategies for improving stability. To remedy these limitations, we aim to develop an analytical method which is capable of assessing the stability of NN-based CF models, and provide insights for the future development of autonomous driving with NN-based CF models to maintain

Table 1
Representative studies of NN-based CF models.

Architecture		Historical time step	Input	Output	Study
FNN	MLP	1	$s_n, v_n, \Delta v_n$, reaction delay	a_n	Khodayari et al. (2012)
	MLP	1	$s_n, v_n, \Delta v_n$	v_n	Zheng et al. (2013)
	MLP	1	$s_n, v_n, \Delta v_n$	a_n	Zhu et al. (2018), Zhu et al. (2020)
	MLP	1	$s_n, v_n, \Delta v_n$	a_n	Mo et al. (2021)
RNN	RNN	1	$s_n, v_n, \Delta v_n$	a_n	Zhou et al. (2017)
	GRU	10	$s_n, v_n, \Delta v_n$	v_n	Wang et al. (2018)
	LSTM	50	$s_n, v_n, \Delta v_n$	v_n	Huang et al. (2018)
	Encoder-Decoder LSTM	50	$s_n, v_n, \Delta v_n$	a_n	Ma and Qu (2020)
	Quantile-regression LSTM	100	$s_n, v_n, \Delta v_n$	v_n	Liu et al. (2023)

Note, s_n, v_n, a_n are the spacing, velocity and acceleration of the subject vehicle. Δv_n is the velocity difference between the subject vehicle and its leader. CF models with grey shadow incorporate multiple steps of historical information.

string stable and thus ease traffic congestion.

2.2. NN-based CF models

Artificial neural networks are computing systems that are inspired by the biological neural networks in animal brains (Abiodun et al., 2018). For modeling CF behaviors, there are generally two prevailing architectures of neural networks: feedforward neural networks (FNN) and recurrent neural networks (RNN). Additionally, NN-based models can also be grouped into three types according to the learning paradigm: supervised learning, unsupervised learning, and reinforcement learning (RL). These paradigms differ in the way of learning the NN, and supervised learning is the most frequent case. Given that string stability is more related to the intrinsic characteristic of the CF model and the structure of NN, the former classification is adopted.

FNN was developed as the earliest and most widely used type of artificial neural network. Information in this type of network travels forward from the input nodes to the hidden nodes and to the output nodes. Multilayer Perceptron (MLP), which can represent a wide variety of functions with appropriate weights, is a commonly used network structure of FNN in modeling CF behavior. Particularly, Jia et al. (2003) presented a four-layer NN-based CF model. Chong et al. (2011) demonstrated that it is feasible to reliably predict acceleration using NN with a single hidden layer, while Khodayari et al. (2012) and Zheng et al. (2013) extended the work by including an instantaneous response time delay. Mo et al. (2021) proposed a CF model combining MLP and analytical CF models to obtain better accuracy and interpretability. It is also shown that MLP consisting of only one hidden layer can be effective in developing a human-like autonomous CF model (Zhu et al., 2018) or super-human CF model (Zhu et al., 2020) with the training through a RL framework.

RNN is a subclass of artificial neural networks in which nodes form a cyclic graph along a temporal sequence, including basic RNN, LSTM NN, GRU NN, etc. The memory is kept in the internal states with the recurrent structure and sequences of inputs can be processed, which makes it a powerful tool in handling time series data. Thus, RNN has been widely employed in CF modeling. Zhou et al. (2017) proposed an RNN-based CF model that outperforms a popular analytical CF model, i.e., Intelligent Driver Model (IDM). Wang et al. (2018) later reported that the GRU-based CF model that incorporates 10 s of historical driving information can achieve better results than the MLP-based CF model and IDM. Based on LSTM NN, a CF model was proposed to capture realistic traffic flow characteristics, in particular the asymmetric driving behavior (Huang et al., 2018). An Encoder-Decoder LSTM model was also developed to consider both memory effect and reaction delay (Ma and Qu, 2020). Recently, Liu et al. (2023) integrated a LSTM architecture with the physics-informed computational graph to preserve the high precision in learning CF behaviors.

A summary of representative studies is provided in Table 1. The hidden layer with the non-differentiable ReLU activation function and output layer with tanh (hyperbolic tangent) activation function are the most common configuration of these models. Due to the inherent characteristics of these two different architectures, FNN usually takes instantaneous driving information (i.e., the spacing and velocity of the subject vehicle and its velocity difference with the immediate leading vehicle) into account, while RNN incorporates multiple time steps of historical information to capture the long memory effect, which is proved to be critical in CF modeling (Wang et al., 2019). In this context, we further categorize the NN-based models into basic CF models and historical information incorporated CF models according to how many steps of historical information are used, and develop general stability criteria separately in the next section.

In summary, NN-based CF models are proved to outperform analytical CF models because of their better representation capability from the inherent structure. Meanwhile, the complex structure and non-differentiable activation functions make it difficult to assess

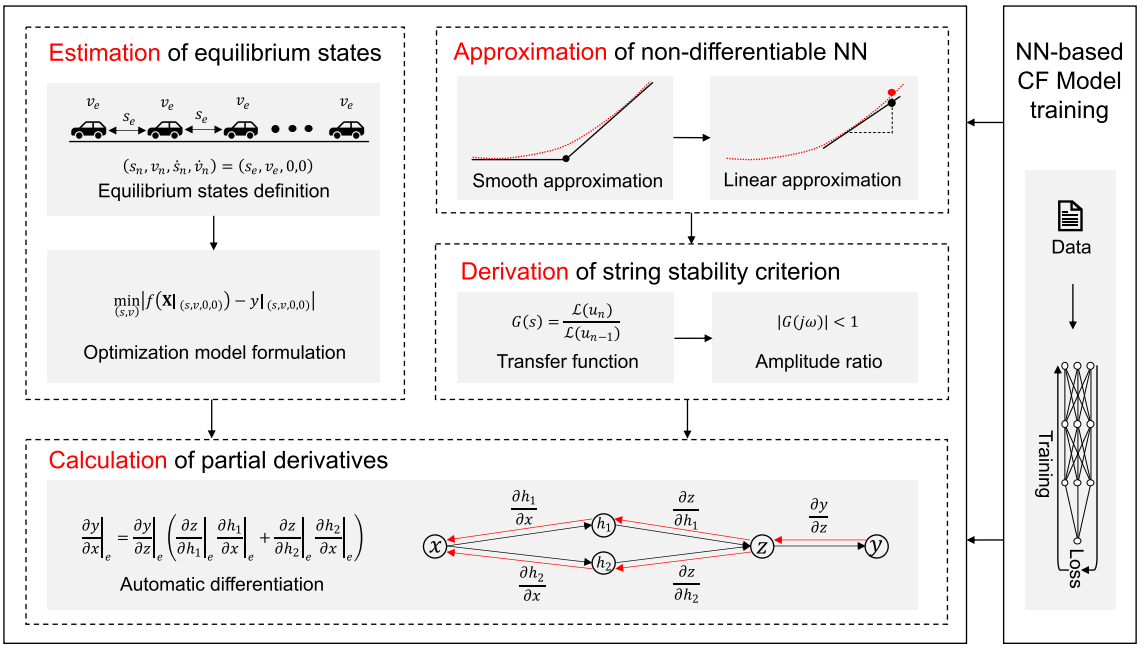


Fig. 2. The EADC framework of string stability analysis for NN-based CF models.

the string stability theoretically. To this end, we aim to develop a generic stability analysis framework for NN-based CF models in this paper. Moreover, as the MLP model developed by Mo et al. (2021) represents for the typical FNN-based model and the LSTM model developed by Huang et al. (2018) represents for the typical RNN-based model, we employ these two models as two examples for validating our proposed stability analysis framework in Section 4.

3. A generic framework for stability analysis of NN-based CF models

3.1. Basic concepts

Given a typical scenario of CF, where identical vehicles are traveling in a single lane without overtaking, and vehicle n follows vehicle $(n-1)$. Generally, a CF model can be formulated as $y = f(\mathbf{X})$, where the output y is longitudinal acceleration or velocity, the input vector $\mathbf{X}$ includes the influence factors of CF behaviors. A widely used basic CF model is $y = a_n = f(s_n, v_n, \Delta v_n)$, where the output a_n is the longitudinal acceleration of vehicle n, s_n denotes the gap between the vehicle $(n-1)$ and the vehicle n, v_n is the velocity of the vehicle n and Δv_n is the velocity difference between vehicle n and its leading vehicle $(n-1)$, i.e., $\Delta v_n = v_n - v_{n-1}$. The function f mapping the input to the output usually constitutes a set of ordinary differential equations for analytical CF models and is an artificial neural network for NN-based CF models.

In conventional stability analysis for analytical CF models, the model is first linearized with the first-order Taylor expansion as:

$$a_n = f_s(s_n - s_e) + f_v(v_n - v_e) + f_{\Delta v}(\Delta v_n - 0) \tag{2}$$

where $f_s = \frac{\partial f}{\partial s}\Big|_e$, $f_v = \frac{\partial f}{\partial v}\Big|_e$, $f_{\Delta v} = \frac{\partial f}{\partial \Delta v}\Big|_e$ are the Taylor expansion coefficients of the function f at the steady states in terms of spacing s_n, velocity v_n and velocity difference Δv_n; s_e and v_e are the equilibrium gap and equilibrium velocity of the homogeneous platoon, which keep the platoon steady (the accelerations are zeros and the gaps and velocities are constant).

The state fluctuation of vehicle n at time t is defined as $u_n(t) = v_n(t) - v_e$, which is the difference between velocity and equilibrium velocity. Using the Laplace transform based method as an example for the stability analysis, the relationship of the perturbation between two consecutive vehicles in the frequency domain is first constructed (Feng et al., 2019; Peppard, 1974; Swaroop and Hedrick, 1996):

$$G(s) = \frac{U_n(s)}{U_{n-1}(s)} = \frac{f_s - s f_{\Delta v}}{s^2 - s(f_v + f_{\Delta v}) + f_s} \tag{3}$$

where $U_n(s)$ and $U_{n-1}(s)$ are the Laplace transform of the $u_n(t)$ and $u_{n-1}(t)$, respectively.

By replacing $s = i\omega$, a widely used stability criterion in frequency domain is $|G(i\omega)| < 1$ for all frequency ω, which is a necessary condition of the time domain-based string stability definition in Eq. (1) (Liu et al., 2001; Sun et al., 2018).

A general string stability criterion (Sun et al., 2018) for basic CF models without considering any time delays is thus deducted as

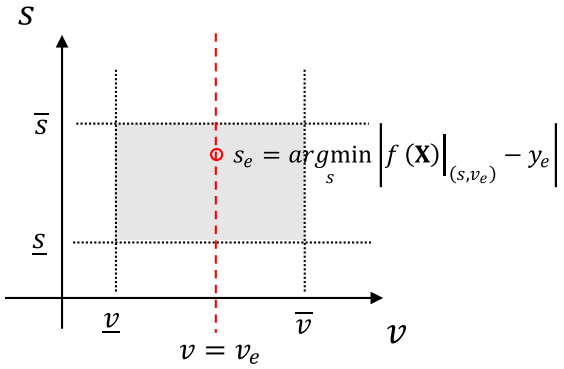


Fig. 3. Feasible region (shadow area) and the search strategy (red dotted line) for Eq. (5). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

below:

$$f_v^2 - 2f_s + 2f_v f_{\Delta v} > 0 \tag{4}$$

However, with respect to the NN-based CF models, the traditional stability analysis methods can no longer be applied mainly for three reasons: 1) it is impossible to calculate the steady states directly without a closed-form expression in the NN-based CF models, 2) non-differentiable functions existing in NN-based CF models impede the linearization for obtaining the stability criterion, and 3) the calculation of partial derivatives is complicated for NNs.

Responding to these challenges, we thus propose a generic EADC framework for theoretically analyzing the string stability of NN-based CF models in this study. The main structure of the framework is presented in Fig. 2, which is composed of four modules: ***Estimation*** (of steady states), ***Approximation*** (of non-differentiable functions), ***Derivation*** (of general stability criteria), and ***Calculation*** (of partial derivatives), as each step in the framework should keep analytical tractability as much as possible. Specifically, we first estimate the steady states of the well-trained NN-based CF model, i.e., the equilibrium spacing-speed relation, by solving an optimization problem formulated for the equilibrium conditions. Then we adopt a continuous function $\widetilde{f}$ for approximating the original function f and linearizing it around the steady states, where the transfer function can be derived with Laplace transform and the string stability criterion can be further determined for two categories of NN-based CF models: basic CF models and historical information incorporated CF models. Lastly, the automatic differentiation method is utilized to compute the values of the partial derivatives in the criterion.

3.2. Estimation of the steady states

The steady-state equilibrium of CF models is reached when vehicles keep the platoon steady with no additional actions needed, i.e., $(s_n, v_n, \Delta v_n, a_n) = (s_e, v_e, 0, 0)$. For analytical CF models where the function f has an explicit mathematical expression, the closed form of $s_e(v_e)$ or $v_e(s_e)$ can be directly derived from $f(\mathbf{X}_e) = y_e$, where $\mathbf{X}_e$ and y_e are the expected input and output under the condition of steady-state equilibrium (for example, regarding the basic CF model $\mathbf{X}_e = (s_e, v_e, 0)$ and $y_e = 0$). However, the equation cannot be analytically solved when the function f is a complicated neural network without explicit formulation. This can be done by carrying out numerical experiments for various speed conditions by simulating a vehicle following a leader for sufficient time. However, the numerical experiment is usually time-consuming and computationally expensive. More importantly, analytical tractability is of great importance in CF stability analysis for providing insights for the future development of autonomous driving with NN-based CF models.

In this study, we propose a generic method to theoretically estimate steady states based on an optimization problem. The idea is to find the steady-state pairs (s_e, v_e) that minimize the absolute difference between the output of the NN-based CF model and the expected output under the condition of steady-state equilibrium, as the difference is zero in an ideal manner. The optimization problem is formulated as below:

$$\begin{aligned} &\min_{(s,v)} \left| f(\mathbf{X})|_{(s,v)} - y_e \right| \\ &s.t. \\ &\underline{s} \le s \le \overline{s} \\ &\underline{v} \le v \le \overline{v} \end{aligned} \tag{5}$$

where $\underline{s}$, $\underline{v}$ are the lower bounds of spacing, velocity respectively, and $\overline{s}$, $\overline{v}$ are the upper bounds. The decision variables are state pairs (s, v). The constraints of spacing s and velocity v imply rational CF behaviors.

The optimization problem in Eq. (5) can be easily solved by traversal search with discretization, since the feasible solutions form a

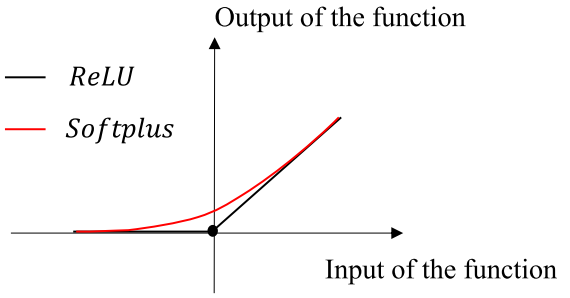


Fig. 4. Illustration of the ReLU function and Softplus function.

simple and limited polygon, as shown in the shadow area in Fig. 3. For each given equilibrium velocity v_e, i.e., $v = v_e$, every feasible s is enumerated in a range of $\left(\underline{s}, \bar{s}\right)$ with a discretization step δ, which corresponds to the red dotted line within the shadow area in Fig. 3. Then for each feasible s, the output of the NN-based CF model $f(\mathbf{X})|_{(s,v_e)}$ and the expected output y_e (if the output of the CF model is acceleration, $y_e = 0$; if the output is velocity, $y_e = v_e$) can be calculated and the candidate s_e can be found with the minimized objective $\left|f(\mathbf{X})|_{(s,v_e)} - y_e\right|$.

Nonetheless, it is very possible that steady states of NN-based CF models will not exist for some traffic conditions, which means local unstable. The obtained candidate steady state pair (s_e, v_e) needs to be further verified with another constraint as shown in Eq. (6). It requires that the objective value is smaller than a tolerance error ε. If no state pair (s_e, v_e) can fulfill this condition, it is assumed that the model is unstable at such traffic state.

$$\left|f(\mathbf{X})|_{(s_e,v_e)} - y_e\right| < \varepsilon \tag{6}$$

To verify the proposed optimization problem for steady state estimation, we apply this method to two widely used CF models, i.e., IDM and OVM, with explicit functions of steady states. The estimated results are totally consistent with the analytical results, which validate the feasibility of the optimization problem for estimating steady states. Details are presented in Appendix A.

3.3. Approximation of non-differentiable functions

As mentioned before, NN-based CF models are commonly non-differentiable, which is generally contributed by non-differentiable activation functions and largely hampers its linearization around the equilibrium points. Activation functions are mathematical operations that transform the input signals of a neuron into an output signal. They are essential for introducing non-linearity and complexity into the NNs, which enable them to learn and approximate any function. ReLU is one of the most commonly used activation functions nowadays because of its simplicity and effectivity in computation (Nair and Hinton, 2010). This function simply eliminates a negative value by making its value zero and retains the values of positive inputs as formulated in Eq. (7). It also has the advantage of inducing sparsity in NN, meaning that some of the neurons will output zero and have no effect on the next layers. This can reduce the redundancy and improve the generalization ability of the model. As a result, it has been extensively used in NN-based models, including many existing CF models. However, the non-differentiable nature at zero makes the NN-based models non-differentiable. To address this issue, we propose to approximate the original non-differentiable activation functions with a continuous activation function, so that the whole NN can be continuous and differentiable. For example, we can replace the ReLU function with the Softplus function (Dugas et al., 2000), as shown in Fig. 4. Specifically, the Softplus function is defined in Eq. (8) with a parameter β. As proved in Eq. (9), a large β (e.g., 100 used in this study) can lead to closer resemblance to the ReLU function. By replacing the ReLU activation function with the Softplus activation function, the non-differentiable NN-based CF model f becomes a smooth and differentiable CF model $\widetilde{f}$. We further quantify the approximation error between the ReLU and Softplus functions, as well as the errors between the ReLU-based and Softplus-based NN models, which show a high precision of 10^{-4} in terms of RMSE (root mean square error). Details can be found in Appendix B.

Note that, this is an easy yet effective way to obtain a smooth approximation of a trained NN without harming its performance, while retraining the NN with the smooth activation functions is generally cumbersome and tricky since it may lead to degraded performance (Dubey et al., 2022). Also, the ReLU and Softplus activation functions used here are typical practices while extending it to other activation functions should be a straightforward matter. It also should be noted that the approximation of activation function is not necessary for every NN-based model, as some models have already adopted differentiable activation functions, such as tanh, sigmoid function.

$$\mathrm{ReLU}(x) = \max(0, x) \tag{7}$$

$$\mathrm{Softplus}(x) = \frac{1}{\beta}\ln\left(1 + e^{\beta x}\right) \tag{8}$$

$$\lim_{\beta\to+\infty}\frac{1}{\beta}\ln\left(1+e^{\beta x}\right)=\max(0,x) \tag{9}$$

With the smooth approximation $\widetilde{f}$ of the NN-based model, linearization is conducted at steady states. Suppose the input $\mathbf{X}$ can be partitioned into blocks according to the leader $(n-1)$ and the follower n, i.e., $\mathbf{X}=(\mathbf{X}_{n-1},\mathbf{X}_n)^{\mathbf{T}}$. In other words, the reaction of the follower y_n is determined by the states of the leader and itself only, which is widely adopted in modeling CF behaviors. The linearization equation of NN-based CF models is shown in Eq. (10).

$$y_n-y_e=\frac{\partial\widetilde{f}(\mathbf{X})}{\partial\mathbf{X}_{n-1}}\left(\mathbf{X}_{n-1}-\mathbf{X}_{n-1}|_e\right)+\frac{\partial\widetilde{f}(\mathbf{X})}{\partial\mathbf{X}_n}\left(\mathbf{X}_n-\mathbf{X}_n|_e\right) \tag{10}$$

Due to the linear combination of all terms, the transfer function between the output disturbance of the follower n and the input disturbances of the leader $(n-1)$ can be constructed by Laplace transform, which is a critical step for deriving the string stability criterion.

3.4. Derivation of the string stability criteria for typical NN-based CF models

Based on the estimation of steady states and linearization of reconstituted smooth structures for NN-based CF models, theoretical string stability criteria can thus be derived. As mentioned in Section 2.2, FNN and RNN are two commonly used architectures in CF modeling. Meanwhile, since FNN usually use the instantaneous input and RNN can take multiple historical time steps into account representing the driving memory, they can be further distinguished as basic CF models and historical information incorporated CF models. Based on the commonly used Laplace transform based method or so-called s-domain analysis (Feng et al., 2019), we derive string stability criteria for these two classes of CF models respectively due to their distinct structures.

3.4.1. String stability criteria of FNN-based CF models

A typical FNN-based CF model can be formulated as Eq. (11), where the inputs are the spacing, velocity and relative velocity as conventional CF models, the output is the acceleration of the follower, and f_{FNN} denotes the function based on FNN.

$$\dot{v}_n=f_{FNN}(s_n,v_n,\Delta v_n) \tag{11}$$

Reconstituting the network by replacing the non-differentiable activation function with a smooth approximation in Section 3.3, yields a differentiable CF model $\widetilde{f}_{FNN}$. The model can then be linearized as:

$$\dot{v}_n=\widetilde{f}_s(s_n-s_e)+\widetilde{f}_v(v_n-v_e)+\widetilde{f}_{\Delta v}\Delta v_n \tag{12}$$

where $\widetilde{f}_s=\left.\frac{\partial\widetilde{f}_{FNN}}{\partial s}\right|_e,\widetilde{f}_v=\left.\frac{\partial\widetilde{f}_{FNN}}{\partial v}\right|_e,\widetilde{f}_{\Delta v}=\left.\frac{\partial\widetilde{f}_{FNN}}{\partial\Delta v}\right|_e$ are the Taylor expansion coefficients.

Differentiating Eq. (12), we have:

$$\ddot{u}_n=\ddot{v}_n=\widetilde{f}_s\dot{s}_n+\widetilde{f}_v\dot{v}_n+\widetilde{f}_{\Delta v}\Delta\dot{v}_n=\widetilde{f}_s(u_{n-1}-u_n)+\widetilde{f}_v\dot{u}_n+\widetilde{f}_{\Delta v}\left(\dot{u}_n-\dot{u}_{n-1}\right) \tag{13}$$

where $u_n(t)=v_n(t)-v_e$ is the velocity variation with respect to the equilibrium velocity v_e.

Taking Laplace transform of both sides of Eq. (13), it is rearranged as Eq. (14), where U_n,U_{n-1} are the Laplace transform of velocity variation $u_n(t),u_{n-1}(t)$ for the follower and the leader, respectively. The transfer function is further derived as shown in Eq. (15).

$$s^2U_n=\widetilde{f}_s(U_{n-1}-U_n)+\widetilde{f}_vsU_n+\widetilde{f}_{\Delta v}(sU_n-sU_{n-1}) \tag{14}$$

$$G(s)=\frac{U_n}{U_{n-1}}=\frac{\widetilde{f}_s-s\widetilde{f}_{\Delta v}}{s^2-\left(\widetilde{f}_{\Delta v}+\widetilde{f}_v\right)s+\widetilde{f}_s} \tag{15}$$

By substituting $s=i\omega$, the stability criterion for the FNN-based CF model is obtained as Eq. (16) and further simplified as Eq. (17), which is consistent with the criterion for the basic analytical CF model as shown in Eq. (4).

$$|G(i\omega)|=\frac{\sqrt{\widetilde{f}_s^{\,2}+\omega^2\widetilde{f}_{\Delta v}^{\,2}}}{\sqrt{\left(\widetilde{f}_s-\omega^2\right)^2+\omega^2\left(\widetilde{f}_{\Delta v}+\widetilde{f}_v\right)^2}}<1 \tag{16}$$

$$\widetilde{f}_v^{\,2}-2\widetilde{f}_s+2\widetilde{f}_v\widetilde{f}_{\Delta v}>0>-\omega^2 \tag{17}$$

3.4.2. String stability criteria of RNN-based CF models

In general, multiple historical time steps of input data are fed into the RNN-based CF models. Additionally, as per previous studies (Huang et al., 2018; Wang et al., 2018), assuming the velocity variable as the output of RNN-based CF models is a more intuitive way to describe the driving behavior and shows better performance. We thus formulate a typical RNN-based CF model as Eq. (18), where τ denotes the maximum historical time steps, and Δt denotes the updating time step.

$$v_n(t) = f_{RNN}(s_n(t-\tau\Delta t : t-\Delta t), v_n(t-\tau\Delta t : t-\Delta t), \Delta v_n(t-\tau\Delta t : t-\Delta t)) \tag{18}$$

With an approximated smooth function $\widetilde{f}_{RNN}$, we first linearize Eq. (18) into:

$$v_n(t) = \widetilde{\mathbf{f}}_s \bullet (\mathbf{s_n} - \mathbf{s}_e) + \widetilde{\mathbf{f}}_v \bullet (\mathbf{v_n} - \mathbf{v}_e) + \widetilde{\mathbf{f}}_{\Delta v} \bullet \Delta \mathbf{v_n} \tag{19}$$

where $\mathbf{s_n} = s_n(t-\tau\Delta t : t-\Delta t) = (s_n(t-\tau\Delta t), \cdots, s_n(t-2\Delta t), s_n(t-\Delta t))^{\mathrm{T}}$ is a vector of spacing variables, $\widetilde{\mathbf{f}}_s = \left(\widetilde{f}_s(t-\tau\Delta t), \cdots, \widetilde{f}_s(t-2\Delta t), \widetilde{f}_s(t-\Delta t)\right)$ is a set of partial derivatives $\widetilde{f}_s(t-\Delta t) = \frac{\partial \widetilde{f}_{RNN}}{\partial s(t-\Delta t)}\Big|_e$ in terms of various timesteps of spacing. Similarly, $\widetilde{\mathbf{f}}_v$, $\widetilde{\mathbf{f}}_{\Delta v}$ are corresponding partial derivatives for the vector of velocity $\mathbf{v_n}$ and relative velocity $\Delta\mathbf{v_n}$ with multiple historical time steps, respectively.

We take the derivative of Eq. (19) as:

$$\dot{v}_n(t) = \widetilde{\mathbf{f}}_s \bullet \dot{\mathbf{s}}_n + \widetilde{\mathbf{f}}_v \bullet \dot{\mathbf{v}}_n + \widetilde{\mathbf{f}}_{\Delta v} \bullet \dot{\Delta \mathbf{v}}_n = \widetilde{\mathbf{f}}_s \bullet (\mathbf{v}_{n-1} - \mathbf{v}_n) + \widetilde{\mathbf{f}}_v \bullet \dot{\mathbf{v}}_n + \widetilde{\mathbf{f}}_{\Delta v} \bullet (\dot{\mathbf{v}}_n - \dot{\mathbf{v}}_{n-1}) \tag{20}$$

Given that $u_n(t) = v_n(t) - v_e$ and $\mathbf{u_n} = \mathbf{v_n} - \mathbf{v}_e$ we obtain:

$$\dot{u}_n(t) = \widetilde{\mathbf{f}}_s \bullet (\mathbf{u}_{n-1} - \mathbf{u_n}) + \widetilde{\mathbf{f}}_v \bullet \dot{\mathbf{u}}_n + \widetilde{\mathbf{f}}_{\Delta v} \bullet \left(\dot{\mathbf{u}}_n - \dot{\mathbf{u}}_{n-1}\right) \tag{21}$$

Taking Laplace transform with $U_n = \mathscr{L}(u_n(t))$, $U_{n-1} = \mathscr{L}(u_{n-1}(t))$, $\mathbf{A} = (e^{-s\tau\Delta t}, \cdots, e^{-2s\Delta t}, e^{-s\Delta t})^{\mathrm{T}}$, Eq. (21) can be transformed to:

$$sU_n = U_{n-1}\widetilde{\mathbf{f}}_s \bullet \mathbf{A} - U_n\widetilde{\mathbf{f}}_s \bullet \mathbf{A} + sU_n\left(\widetilde{\mathbf{f}}_v + \widetilde{\mathbf{f}}_{\Delta v}\right) \bullet \mathbf{A} - sU_{n-1}\widetilde{\mathbf{f}}_{\Delta v} \bullet \mathbf{A} \tag{22}$$

The transfer function of the RNN-based CF model is thus:

$$G(s) = \frac{U_n}{U_{n-1}} = \frac{\widetilde{\mathbf{f}}_s \bullet \mathbf{A} - s\widetilde{\mathbf{f}}_{\Delta v} \bullet \mathbf{A}}{s - s\left(\widetilde{\mathbf{f}}_v + \widetilde{\mathbf{f}}_{\Delta v}\right) \bullet \mathbf{A} + \widetilde{\mathbf{f}}_s \bullet \mathbf{A}} \tag{23}$$

By inserting $s = i\omega$, vector $\mathbf{A}$ is expanded with Euler's formula:

$$A(i\omega, \tau) = \begin{pmatrix} e^{-i\omega\tau\Delta t} \\ \vdots \\ e^{-2i\omega\Delta t} \\ e^{-i\omega\Delta t} \end{pmatrix} = \begin{pmatrix} \cos\omega\tau\Delta t \\ \vdots \\ \cos 2\omega\Delta t \\ \cos\omega\Delta t \end{pmatrix} - i\begin{pmatrix} \sin\omega\tau\Delta t \\ \vdots \\ \sin 2\omega\Delta t \\ \sin\omega\Delta t \end{pmatrix} \tag{24}$$

Let $\mathbf{B} = (\cos\omega\tau\Delta t, \cdots, \cos 2\omega\Delta t, \cos\omega\Delta t)^{\mathrm{T}}$ and $\mathbf{C} = (\sin\omega\tau\Delta t, \cdots, \sin 2\omega\Delta t, \sin\omega\Delta t)^{\mathrm{T}}$, then $\mathbf{A} = \mathbf{B} - i\mathbf{C}$ and thus the transfer function is rearranged with:

$$G(i\omega) = \frac{\widetilde{\mathbf{f}}_s \bullet \mathbf{B} - \omega\widetilde{\mathbf{f}}_{\Delta v} \bullet \mathbf{C} - i\left(\widetilde{\mathbf{f}}_s \bullet \mathbf{C} + \omega\widetilde{\mathbf{f}}_{\Delta v} \bullet \mathbf{B}\right)}{\widetilde{\mathbf{f}}_s \bullet \mathbf{B} - \omega\left(\widetilde{\mathbf{f}}_v + \widetilde{\mathbf{f}}_{\Delta v}\right) \bullet \mathbf{C} + i\left(\omega - \omega\left(\widetilde{\mathbf{f}}_v + \widetilde{\mathbf{f}}_{\Delta v}\right) \bullet \mathbf{B} - \widetilde{\mathbf{f}}_s \bullet \mathbf{C}\right)} \tag{25}$$

By substituting Eq. (25) into $|G(i\omega)| < 1$, we have:

$$|G(i\omega)| = \frac{\sqrt{\left(\widetilde{\mathbf{f}}_s \bullet \mathbf{B} - \omega\widetilde{\mathbf{f}}_{\Delta v} \bullet \mathbf{C}\right)^2 + \left(\widetilde{\mathbf{f}}_s \bullet \mathbf{C} + \omega\widetilde{\mathbf{f}}_{\Delta v} \bullet \mathbf{B}\right)^2}}{\sqrt{\left(\widetilde{\mathbf{f}}_s \bullet \mathbf{B} - \omega\left(\widetilde{\mathbf{f}}_v + \widetilde{\mathbf{f}}_{\Delta v}\right) \bullet \mathbf{C}\right)^2 5 + \left(\omega - \omega\left(\widetilde{\mathbf{f}}_v + \widetilde{\mathbf{f}}_{\Delta v}\right) \bullet \mathbf{B} - \widetilde{\mathbf{f}}_s \bullet \mathbf{C}\right)^2}} < 1 \tag{26}$$

Thus, the string stability criterion is:

$$\omega^2\begin{bmatrix}1+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}\right)^2+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}\right)\\-2\left(\widetilde{\mathbf{f}}_v+\widetilde{\mathbf{f}}_{\Delta v}\right)\bullet B+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}\right)^2\\+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{C}\right)\end{bmatrix}+2\omega\begin{bmatrix}\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}\right)\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{C}\right)-\\\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{B}\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}\right)-\widetilde{\mathbf{f}}_s\bullet C\end{bmatrix}>0 \tag{27}$$

Since low frequency puts the biggest restriction on stability (long-wavelength instability always happens first), Eq. (27) is transformed with $\omega\to0$, $\sin\omega\tau\Delta t\approx\omega\tau\Delta t$, $\mathbf{C}\approx\omega(\tau\Delta t,\cdots,2\Delta t,\Delta t)^{\mathrm{T}}=\omega\mathbf{C}_\tau$:

$$\omega^2\begin{bmatrix}1+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}\right)^2+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}\right)-2\left(\widetilde{\mathbf{f}}_v+\widetilde{\mathbf{f}}_{\Delta v}\right)\bullet B\\+\omega^2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)^2+2\omega^2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{C}_\tau\right)\\+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}\right)\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{C}_\tau\right)-2\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{B}\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)-2\widetilde{\mathbf{f}}_s\bullet\mathbf{C}_\tau\end{bmatrix}>0 \tag{28}$$

Let $\varphi(\omega,\tau)$ stands for the expression in the square bracket in Eq. (28), the first necessary condition of string stability for RNN-based CF models is:

$$\varphi(\omega,\tau)>0 \tag{29}$$

With $\lim_{\omega\to0}\mathbf{B}=(1,\cdots,1,1)^{\mathrm{T}}=\mathbf{B}_0$, Eq. (29) can be simplified as:

$$\begin{aligned}&1+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)^2+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}_0\right)-2\left(\widetilde{\mathbf{f}}_v+\widetilde{\mathbf{f}}_{\Delta v}\right)\bullet\mathbf{B}_0\\&+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{C}_\tau\right)-2\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)-2\widetilde{\mathbf{f}}_s\bullet\mathbf{C}_\tau>0\end{aligned} \tag{30}$$

Additionally, to guarantee that the inequality in Eq. (29) holds for all $\omega>0$, the derivative of $\varphi(\omega,\tau)$ in terms of ω need to be positive when $\omega\to0$, which results in:

$$\begin{aligned}\dot{\varphi}(\omega,\tau)=&2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}\right)\left(\widetilde{\mathbf{f}}_v\bullet\dot{\mathbf{B}}\right)+2\left(\widetilde{\mathbf{f}}_v\bullet\dot{\mathbf{B}}\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}\right)+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\dot{\mathbf{B}}\right)\\&-2\left(\widetilde{\mathbf{f}}_v+\widetilde{\mathbf{f}}_{\Delta v}\right)\bullet\dot{\mathbf{B}}+2\left(\widetilde{\mathbf{f}}_v\bullet\dot{\mathbf{B}}\right)\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{C}_\tau\right)-2\left(\widetilde{\mathbf{f}}_s\bullet\dot{\mathbf{B}}\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)>0\end{aligned} \tag{31}$$

where $\dot{\mathbf{B}}=(-\tau\Delta t\sin\omega\tau\Delta t,\cdots,-2\Delta t\sin2\omega\Delta t,-\Delta t\sin\omega\Delta t)^{\mathrm{T}}$, and as $\omega\to0$, $\dot{\mathbf{B}}\approx\omega\left(-\tau^2\Delta t^2,\cdots,-4\Delta t^2,-\Delta t^2\right)^{\mathrm{T}}=\omega\mathbf{B}_\tau$, $\mathbf{B}\to\mathbf{B}_0$. Then the second necessary condition of string stability for RNN-based CF models is:

$$\begin{aligned}&\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_\tau\right)+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_\tau\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}_0\right)+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}_\tau\right)\\&-\left(\widetilde{\mathbf{f}}_v+\widetilde{\mathbf{f}}_{\Delta v}\right)\bullet\mathbf{B}_\tau+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_\tau\right)\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{C}_\tau\right)-\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{B}_\tau\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)>0\end{aligned} \tag{32}$$

$$\mathbf{B}_0=(1,\cdots,1,1)^{\mathrm{T}},\mathbf{B}_\tau=\left(-\tau^2\Delta t^2,\cdots,-4\Delta t^2,-\Delta t^2\right)^{\mathrm{T}},\mathbf{C}_\tau=(\tau\Delta t,\cdots,2\Delta t,\Delta t)^{\mathrm{T}}$$

Therefore, Eqs. (30) and (32) jointly constitute the string stability criteria for RNN-based CF models, which is more complex than that for FNN-based CF models because of the incorporation of multiple historical time steps. A similar process can be easily applied for RNN-based CF model with acceleration output, and to save space we directly present its stability criteria in Eqs. (33) and (34).

$$\begin{aligned}&\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)^2+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}_0\right)-2\widetilde{\mathbf{f}}_s\bullet\mathbf{B}_0\\&+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{C}_\tau\right)-2\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)>0\end{aligned} \tag{33}$$

$$\begin{aligned}&\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_\tau\right)+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_\tau\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}_0\right)+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_0\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{B}_\tau\right)+2\left(\widetilde{\mathbf{f}}_v+\widetilde{\mathbf{f}}_{\Delta v}\right)\bullet\mathbf{C}_\tau\\&+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{B}_\tau\right)\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{C}_\tau\right)-\left(\widetilde{\mathbf{f}}_s\bullet\mathbf{B}_\tau\right)\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)+\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)^2+2\left(\widetilde{\mathbf{f}}_v\bullet\mathbf{C}_\tau\right)\left(\widetilde{\mathbf{f}}_{\Delta v}\bullet\mathbf{C}_\tau\right)-\widetilde{\mathbf{f}}_s\bullet\mathbf{B}_\tau>0\end{aligned} \tag{34}$$

3.5. Calculation of partial derivatives with automatic differentiation

Although NN-based CF models become differentiable with approximated continuous activation functions, they are highly nonlinear with complicated hidden layers and have no explicit expression. Thus, how to calculate the partial derivatives within the obtained

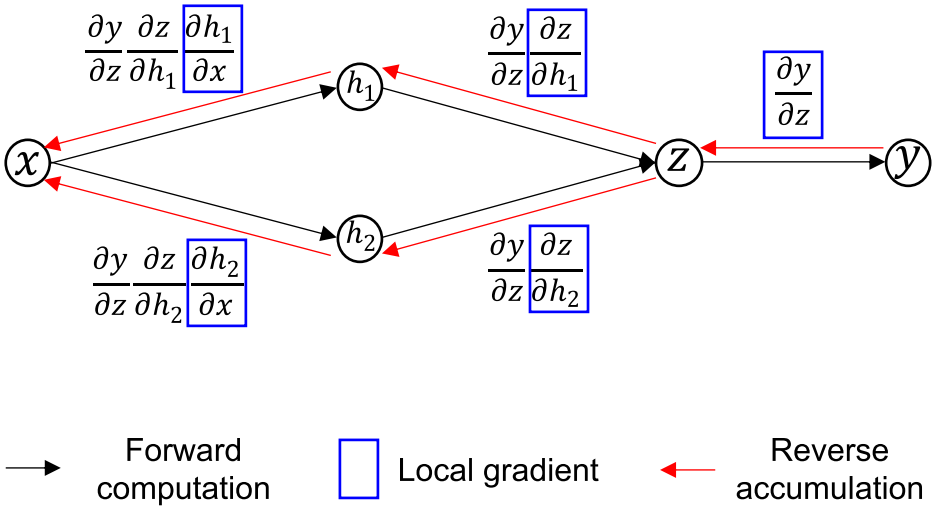


Fig. 5. An example of AD process.

string stability criteria remains unresolved.

There are broadly three approaches for the computation of derivatives, including numerical differentiation (i.e., finite difference approximations (Burden, 2005)), symbolic differentiation (von zur Gathen et al., 2013) and automatic differentiation (AD) method (Güneş Baydin et al., 2015). Numerical differentiation relies on the definition of a derivative (using the simplest example, the derivative of function $f(x)$ is estimated by $\frac{df(x)}{dx} \approx \frac{f(x+h)-f(x)}{h}$, where h is a small deviation), while the approximation error easily occurs by the limited precision of computations and the chosen value of step size h, and it also requires many function evaluations especially for complex models with many inputs. Symbolic differentiation addresses the weakness of finite difference approximations, which manipulates formulas to generate derivative expressions and exact values. However, it requires models to be defined as closed-form expressions and can quickly result in expression swell that is time-consuming to analyze. To overcome these issues, AD method is proposed, which basically first conducts symbolic differentiation at the elementary operation level (e.g., addition, subtraction) and stores the analytical derivative of each elementary function and then repeatedly applies the chain rule to these operations with their numerical values, so that partial derivatives of arbitrary order can be computed automatically. It is thus provided the generality of numerical techniques and the accuracy of analytical derivatives simultaneously.

Therefore, we adopt the AD to evaluate derivatives for NN-based models. For the sake of easier understanding, a typical process of AD is presented here. As shown in Fig. 5, a function $y = f(x)$ is break down to elementary functions, which is possible for most functions and models even for complex NN models. A computational graph is utilized to represent the function with intermediate variables h_1,h_2, z. Meanwhile, the local gradient of each elementary function is calculated and stored, which is the analytical partial derivative of its output with respect to its input. For each input x fed forward, values of intermediate variables and output y are generated along the black arrows in Fig. 5. Then the global derivatives of output y with respect to the input factor x are computed by propagating the numerical results of local gradients backward along the red arrows using the chain rule as in Eq. (35).

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial z}\frac{\partial z}{\partial h_1}\frac{\partial h_1}{\partial x} + \frac{\partial y}{\partial z}\frac{\partial z}{\partial h_2}\frac{\partial h_2}{\partial x} \tag{35}$$

4. Applying the framework to two typical NN-based CF models

To verify the feasibility and performance of the proposed EADC framework, in this section, we apply the framework[1] to two categories of NN-based CF models (i.e., FNN-based CF model and RNN-based CF model, respectively), and compare their theoretical stability characteristics with a widely used analytical CF model IDM.

4.1. Model preparation

4.1.1. The FNN-based CF model

The FNN-based CF model used here was proposed by Mo et al. (2021), which is informed by the physics-based model (IDM) to better learn realistic driving behavior from naturalistic driving data. It leverages the advantage of both physics-based (being data-efficient and interpretable) and deep learning based (being generalizable) models. Despite the use of a physics-informed deep learning paradigm, the CF policy is essentially a MLP network (referred to as Mo-MLP model below) with three layers: an input layer

[1] The implementation of these models is available in https://github.com/tjzxh/EADC.

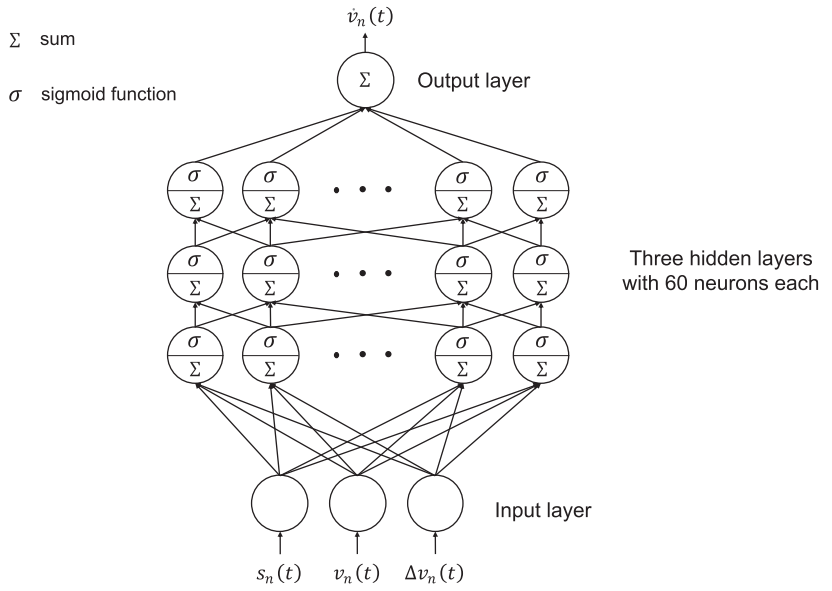


Fig. 6. Schematic illustration of the Mo-MLP model.

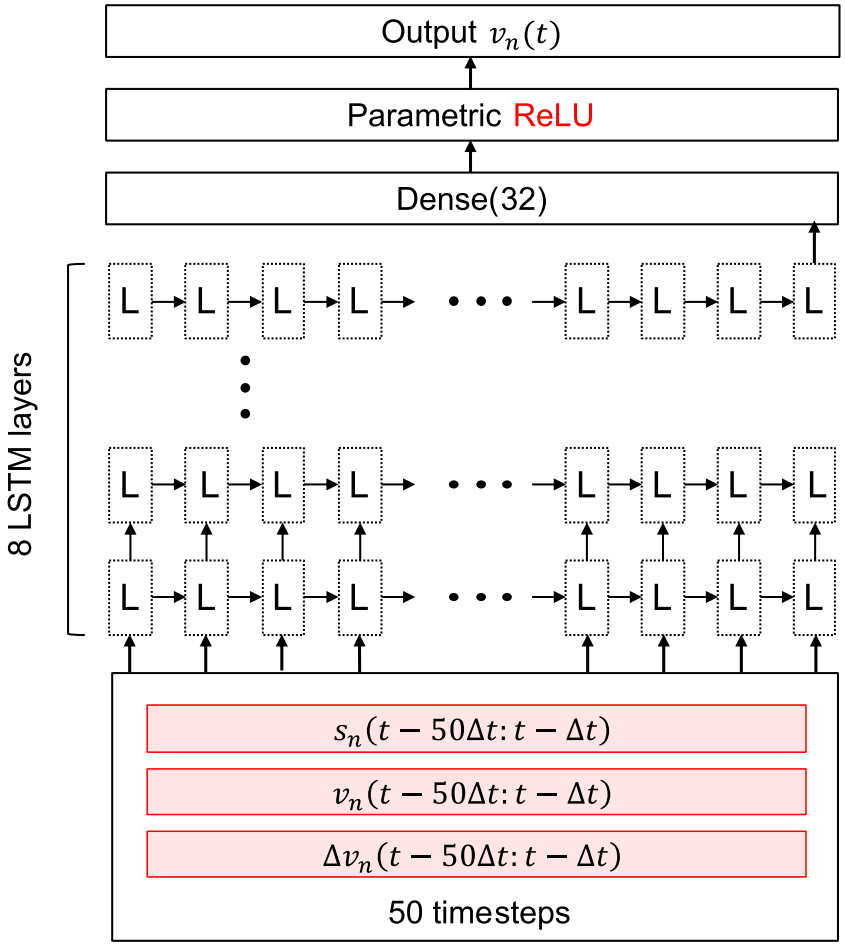


Fig. 7. Schematic illustration of the Huang-LSTM model.

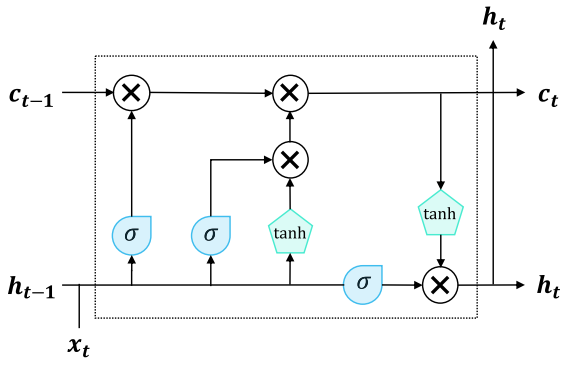


Fig. 8. Schematic diagram of the memory block of LSTM.

with input $\mathbf{X} = (s_n(t), v_n(t), \Delta v_n(t))^{\mathrm{T}}$, an output layer with output $y = \dot{v}_n(t)$, and three hidden layers of 60 neurons with the sigmoid activation function, as shown in Fig. 6. Thus, the Mo-MLP model can be mathematically formulated as Eq. (36), where the vectors $\mathbf{W}_1$, $\mathbf{W}_2, \mathbf{W}_3, \mathbf{W}_4$ are the connection weights of neurons in consecutive layers.

The Mo-MLP model was trained and tested using reconstructed NGSIM dataset (Montanino and Punzo, 2015), and the results demonstrated a decent performance with a RMSE of 0.89 m/s and 5.27 m for speed and gap.

$$\dot{v}_n(t) = \mathbf{W}_4 \bullet \sigma(\mathbf{W}_3 \bullet \sigma(\mathbf{W}_2 \bullet \sigma(\mathbf{W}_1 \bullet (s_n(t), v_n(t), \Delta v_n(t)))))) \tag{36}$$

4.1.2. The RNN-based CF model

The RNN-based CF model adopted in this study was proposed by Huang et al. (2018) based on LSTM (referred to as Huang-LSTM model below), which aims to capture realistic driving behavior by incorporating historical driving information. Specifically, the Huang-LSTM model adopted $s_n, v_n, \Delta v_n$ of last 5 s as inputs, i.e., $\mathbf{X} = (s_n(t-\tau\Delta t : t-\Delta t), v_n(t-\tau\Delta t : t-\Delta t), \Delta v_n(t-\tau\Delta t : t-\Delta t))^{\mathrm{T}}$, where $\tau = 50$, $\Delta t = 0.1s$ and predicted the velocity of the following vehicle at the next time step, i.e., $y = v_n(t)$. The structure of the Huang-LSTM model is shown in Fig. 7, where the number of hidden layers and neurons in each hidden layer are set as 8 and 32, respectively, and the activation function in the output layer is a variant of ReLU called Parametric ReLU ($f(x) = \max(x, bx)$ and $b = 0.9$) in this study.

In the hidden layers, the basic component is a standard LSTM memory block (Hochreiter and Schmidhuber, 1997), as shown in Fig. 8. Each memory block includes three controlling gates, i.e., the input, forget, and output gate, which regulate the flow of information into and out of the cell. The compact forms of the equations are provided as follows:

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \tag{37}$$

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{38}$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{39}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{40}$$

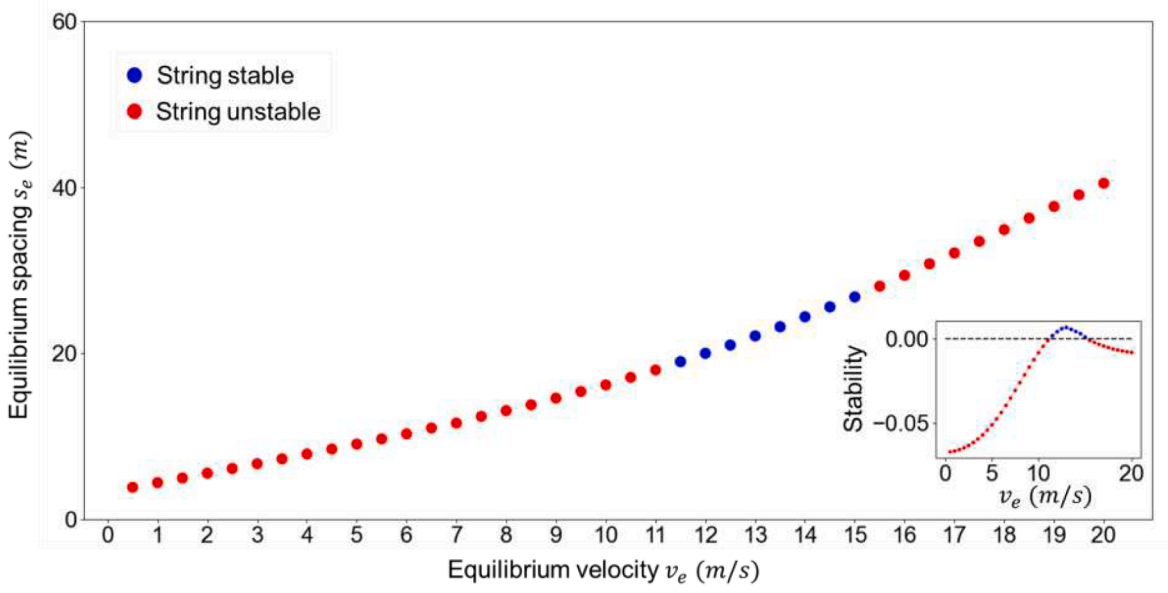
$$h_t = o_t \tanh(c_t) \tag{41}$$

where x_t is the input data of each historical time step t, σ is the logistic sigmoid function, and i, f, c, o denote the input gate, forget gate, cell state vectors and output gate, respectively. W_* ($* = xi,hi,ci,xf,hf,cf,xc,hc,xo,ho,co$) is the weight matrix, and b_* ($* = i,f,c,o$) is the bias vector of each gate.

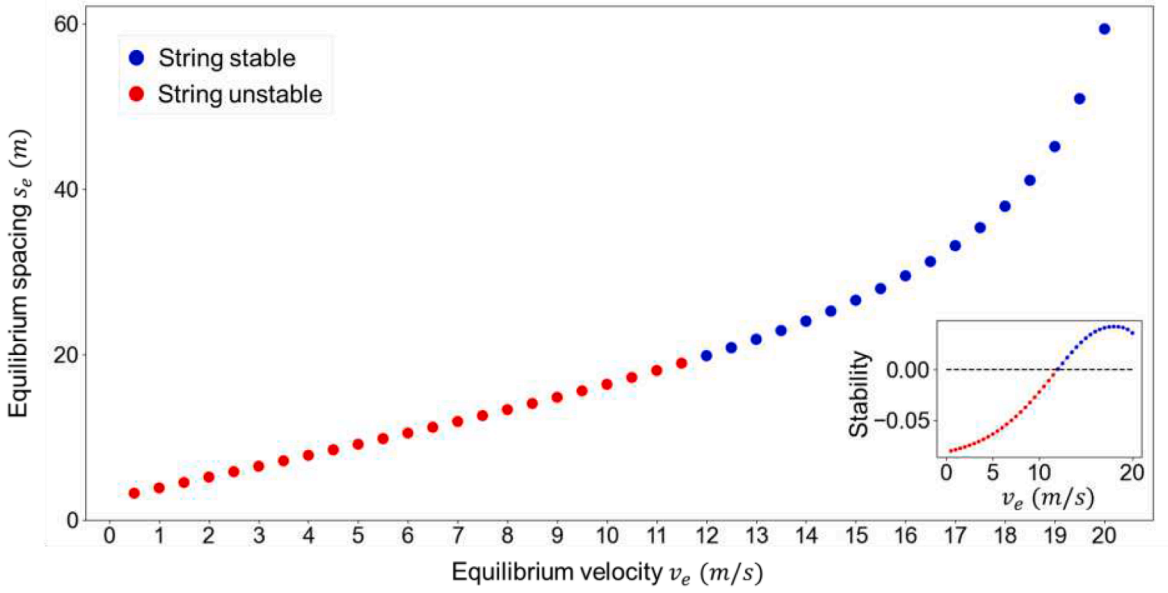
The Huang-LSTM model is calibrated and validated using the reconstructed NGSIM I-80 dataset (the dataset used in the original paper is US-101 dataset). Since the insufficient data in the low-speed range in NGSIM dataset can bias the training, data augmentation (van Dyk and Meng, 2001) is conducted by simulating more CF data in low-speed range with the IDM calibrated with the reconstructed NGSIM dataset. The performance is also evaluated by RMSE of the speed and gap between the simulated trajectories and field data, which is 0.86 m/s and 4.80 m, showing that the proposed CF model is capable of reproducing realistic traffic flow characteristics.

4.2. Results of theoretical stability analysis

Following the proposed EADC framework, we first estimate the steady states of the trained Mo-MLP model and Huang-LSTM model. Specifically, for equilibrium velocities ranging from $\underline{v} = 0.5m/s$ to $\bar{v} = 20m/s$ with a step size of 0.5 m/s, corresponding equilibrium spacings are obtained, based on the optimization problem constructed in Section 3.2 where the feasible spacings range from $\underline{s} = 0.1m$ to $\bar{s} = 100m$; a small step size of discretization $\delta = 0.001$ and a tolerance error $\varepsilon = 0.001$ are used. Replacing the non-differentiable activation function, the linearization of CF models can be implemented and corresponding string stability criteria are derived as Eq.



(a) The Mo-MLP model



(b) The Huang-LSTM model

Fig. 9. Theoretical string stability of NN-based CF models.

(17) for the Mo-MLP model, and Eqs. (30) and (32) for the Huang-LSTM model. Based on the stability criteria, we then determine the string stability of the two CF models for different equilibrium velocities with the implementation of AD method via autograd API in PyTorch. All results are presented in Fig. 9.

As shown in Fig. 9(a), the theoretical string stability of the Mo-MLP model is only maintained for equilibrium velocities from 11.5 m/s to 15 m/s, while it is unstable when the traffic is rather congested (smaller than 11.5 m/s) or in uncongested flow (larger than 15 m/s). Its string stability value in Eq. (17) first increases and then drops as also presented in the subfigure of Fig. 9(a). Regarding the Huang-LSTM model, as we can see from Fig. 9(b), it is unstable with the equilibrium velocity under 12 m/s and stable with the

Table 2
Performance of three CF models.

Model	Architecture	$RMSE_v$(m/s)	$RMSE_s$(m)
Mo-MLP	MLP NN	0.89	5.27
Huang-LSTM	LSTM NN	0.86	4.80
IDM	Analytical functions	0.97	6.37

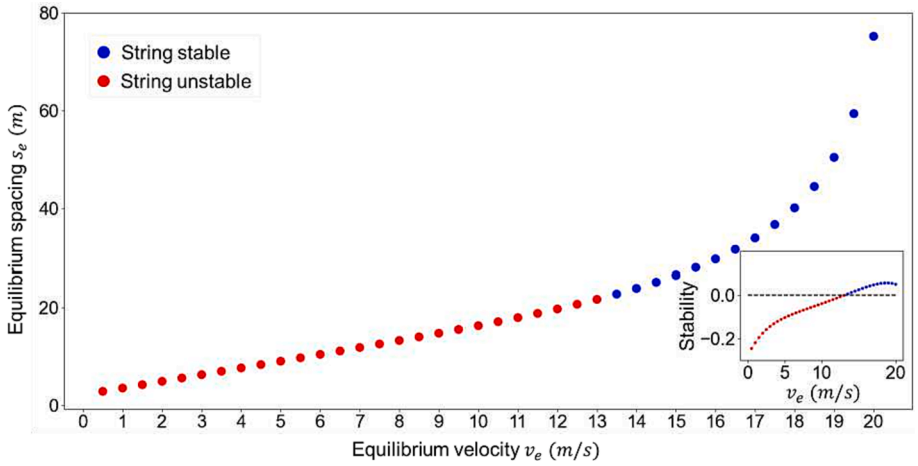


Fig. 10. String stability results of the IDM.

equilibrium velocity equal to or above 12 m/s. Note that the string stability value of the Huang-LSTM model in Eq. (30) exhibit a similar trend that first increases and then decreases, while the values of Eq. (32) in the criteria are all positive and thus omitted.

Although both the Mo-MLP model and the Huang-LSTM model achieve superior performance in learning human behavior (as shown in Table 2), the results of steady states and corresponding string stability are different, largely due to their distinct architectures. However, the string instability in a relatively low-speed range (smaller than 11.5 m/s) is observed in both models, which reflects the potential negative effects of NN-based CF models for AV and necessitates the consideration of string stability in the model training.

4.3. Comparison with analytical CF model

To further demonstrate the theoretical stability results from the proposed framework, we compare them with the string stability result of a widely used analytical CF model–IDM using the conventional string stability analysis method, as the conventional method for analytical CF models has been widely applied. IDM is formulated in Eqs. (42) and (43).

$$\dot{v}_n(t) = \alpha\left(1 - \left(\frac{v_n(t)}{v_0}\right)^4 - \left(\frac{s_n^*(t)}{s_n(t)}\right)^2\right) \tag{42}$$

$$s_n^*(t) = s_0 + v_n(t)T + \frac{v_n(t)\Delta v_n(t)}{2\sqrt{\alpha\beta}} \tag{43}$$

where v_0 is the desired speed, $s_n^*(t)$ is the desired spacing gap, s_0 is the minimum gap at the standstill situation, T is the desired time gap, α is the maximum acceleration and β is the comfortable deceleration. Other variables have the same meanings as those introduced before.

The parameters of IDM[2] are also calibrated using the reconstructed NGSIM dataset and genetic algorithm, with the recommended Normalized Root Mean Squared Error (NRMSE) adopted as the objective function (Punzo et al., 2021). The average parameter values are: $v_0 = 20.9m/s$, $T = 1.37s$, $\alpha = 0.97m/s^2$, $\beta = 1.85m/s^2$, and $s_0 = 2.14m$. We also assess the performance by RMSE of the speed and gap between the simulated trajectories and field data, which is 0.97 m/s and 6.37 m. As summarized in Table 2, we can see that the two NN-based CF models achieve better results in capturing realistic driving behavior, compared to the result of IDM.

The relationship between equilibrium spacing s_e and equilibrium velocity v_e is derived as:

[2] This calibrated IDM is used as the physics-based model in the physics-informed deep learning CF model, i.e., Mo-MLP, and also used to simulate more data for the training of the Huang-LSTM.

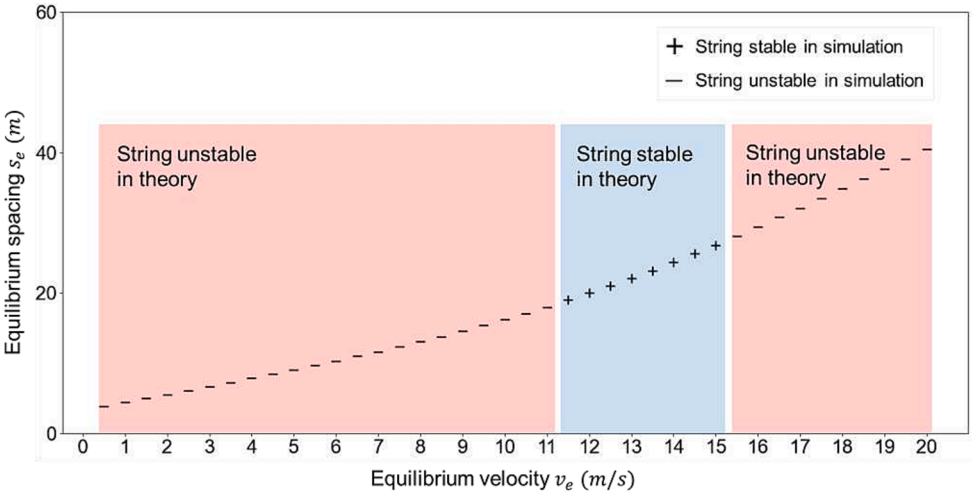


Fig. 11. Simulation results and theoretical results of the Mo-MLP model.

$$s_e = \frac{(s_0 + v_e T)(v_0)^2}{\sqrt{(v_0)^4 - (v_e)^4}} \tag{44}$$

Then, the string stability of IDM can be directly calculated using the traditional string stability method for analytical CF models with the stability criterion in Eq. (4), and the stability results are shown in Fig. 10. We can see that IDM is string unstable when the equilibrium velocity below 13 m/s and string stable with larger equilibrium velocity. The string stability value of IDM in Eq. (4), as shown in the subfigure, first increases with the equilibrium velocity and tends to decrease when the equilibrium velocity is larger than 19 m/s. This is fairly consistent with the previous two NN based CF models, indicating the potential stability deterioration in uncongested traffic states.

In summary, although the steady states and corresponding string stability varies with different models, we can conclude several remarks as follows. First, different architectures of NNs lead to different stability results. Specifically, the Mo-MLP shows string instability beyond 15 m/s while the Huang-LSTM is string stable, although both NNs are well-trained and can reproduce human behavior accurately. This underscores the importance of investigating the string stability of NN based CF models. Additionally, all the three models, including the NN-based models and IDM that trained or calibrated using naturalistic driving data, show string instability in the low-speed range (smaller than 11.5 m/s, i.e., 41.4 km/h). Human drivers are more string unstable under these traffic states and traffic oscillations could be propagated easily, which accords with the previous study (Jiang et al., 2017). Moreover, under uncongested traffic states, unstable driving behavior is very possible with decreased string stability values, which should be carefully investigated when designing new CF models.

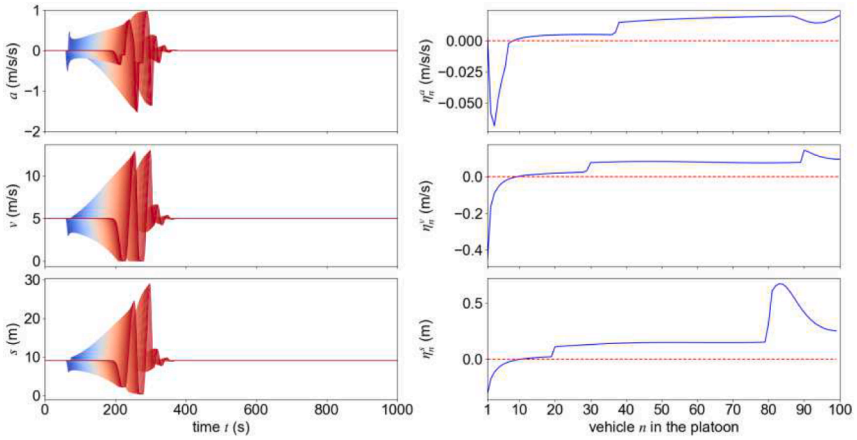
5. Validating the framework with numerical simulations

Due to the unrealistic assumptions adopted in the theoretical stability analysis, such as infinite platoon and long-wavelength perturbation, discrepancies exist between the theoretical stability results and the real traffic, which can significantly harm the applicability of the theoretical stability criteria. One simple yet efficient method to validate the soundness and performance of the proposed EADC framework is to evaluate the consistency between the theoretical results and simulation results. For this purpose, a series of numerical simulation experiments with the Mo-MLP model and the Huang-LSTM model are implemented. We focus on the microscopic fundamental diagram (Treiber and Kesting, 2013), i.e., the equilibrium speed with respect to equilibrium gap or vice versa, and corresponding string stability of these two well-trained NN-based CF models.

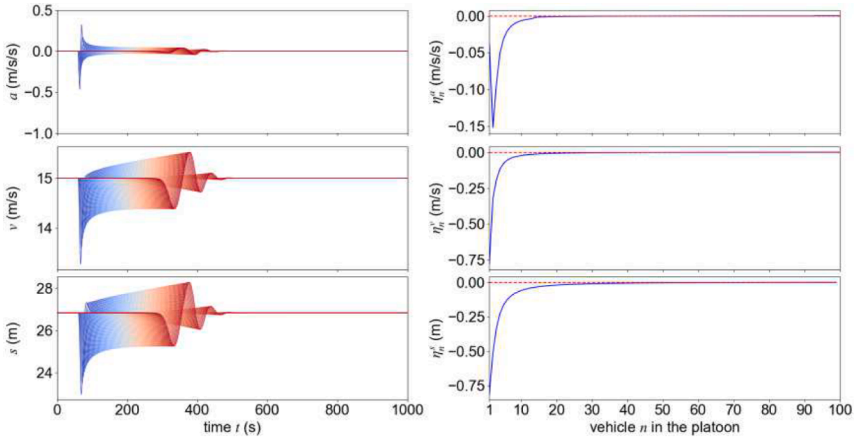
5.1. Simulation setup

Following the setup in Sun et al. (2018), a homogeneous platoon of 100 identical vehicles drive on a very long, single-lane straight road with predefined velocities and corresponding equilibrium spacings calculated before. The vehicles then reach the equilibrium state rapidly. At $t = 60s$, a perturbation is introduced to the first leading vehicle, forcing it to first decelerate at -0.5 m/s^2 for 5 s and then accelerate at 0.5 m/s^2 for another 5 s. After the perturbation, the first leading vehicle continues to cruise at the original equilibrium velocity. The states of the following vehicles are collected from the beginning to 1000 s, which contains the data before the perturbation and after the perturbation.

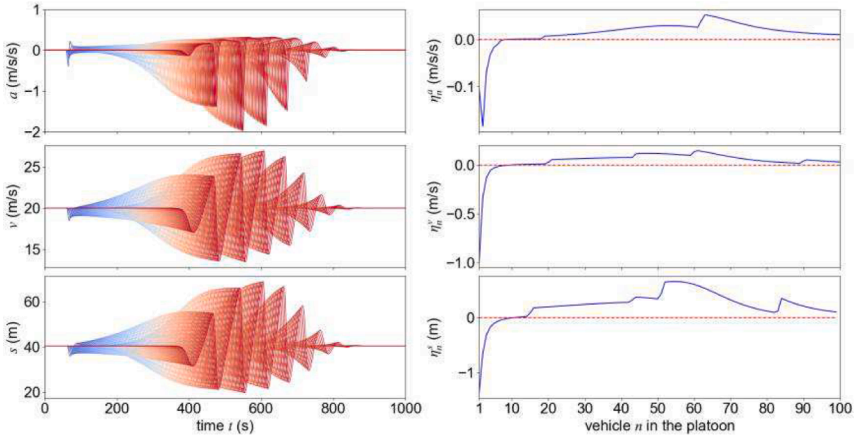
In the simulation, for the Mo-MLP model, the acceleration is first calculated and the velocity and position are then updated with the kinematic equations as shown in Eqs. (45) and (46), and for the Huang-LSTM model, as the velocity is the direct output of the model, only the position is updated with Eq. (46).



(a) $v_e = 5\,m/s$



(b) $v_e = 15\,m/s$



(c) $v_e = 20\,m/s$

Fig. 12. Simulation results of the Mo-MLP model.

$$v_n(t+\Delta t) = v_n(t) + a_n(t)\Delta t \tag{45}$$

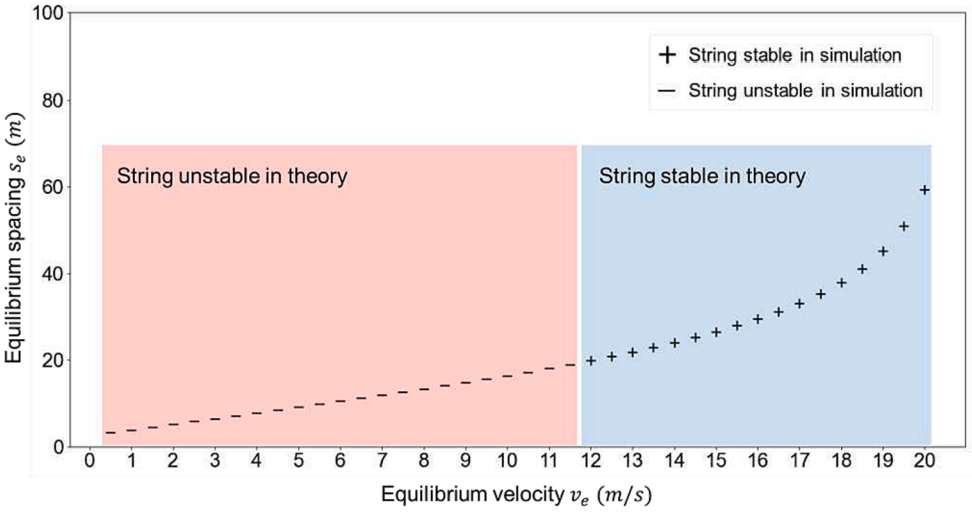


Fig. 13. Simulation results and theoretical results of the Huang-LSTM model.

$$x_n(t+\Delta t) = x_n(t) + v_n(t)\Delta t + \frac{1}{2}a_n(t)(\Delta t)^2 \tag{46}$$

To evaluate the simulation results, we use three variables as the indicator of perturbation, including acceleration, velocity and spacing deviation from the steady states. More specifically, the CF model is identified as string stable if a monotonical decrease is achieved along the platoon for the infinity norm of all three variables; otherwise, it is string unstable.

5.2. Simulation results

For the Mo-MLP model, the simulation results of string stability for each steady state are presented in Fig. 11, which is totally consistent with the theoretical results computed before and thus verifies the proposed EADC framework.

Examples of the simulation results for the Mo-MLP model is shown in Fig. 12, where the equilibrium speed are 5, 15, 20 m/s. For the sake of clarity, other simulation results are not provided here. The left three subfigures are the profiles of acceleration, velocity, and spacing, respectively for all the following vehicles in the platoon, while the right three subfigures show the evolutions of perturbation along the platoon to facilitate the analysis of string stability with $\eta_n^k = \|u_n^k\|_\infty - \|u_{n-1}^k\|_\infty$ (k denotes acceleration, velocity, or spacing), where $\|u_n^k\|_\infty = max_t|u_n^k(t)|$ is the maximum magnitude of u_n^k in the time domain. For example, η_n^v represents the difference between the speed deviation of vehicle n and vehicle $(n-1)$. When $\eta_n^k < 0$ holds for all three states and each vehicle n, i.e., $\forall n, \eta_n^a < 0, \eta_n^v < 0, \eta_n^s < 0$ (the blue solid lines remain below zero, the red dashed line), it means that the perturbation attenuates as it propagates away from the first leading vehicle and the string stability is satisfied; otherwise, it is string unstable. Apparently, the Mo-MLP is string unstable at $v_e = 5m/s$ and $v_e = 20m/s$, and string stable at $v_e = 15m/s$, which validates the theoretical results.

For the Huang-LSTM model, the simulation results are shown in Fig. 13 in conjunction with the theoretical results. The estimated theoretical string stability results accord with the simulated results for all the steady states as well. The soundness of the proposed stability analysis framework for NN-based CF models is thus further verified.

In addition, two typical simulation scenarios of the Huang-LSTM model are illustrated in Fig. 14 with equilibrium velocity $v_e = 5m/s, 15m/s$. Fig. 14(a) shows that the disturbance is amplified at $v_e = 5m/s$, which is string unstable, while it is string stable at $v_e = 15m/s$ as the perturbations decay along the platoon in Fig. 14(b).

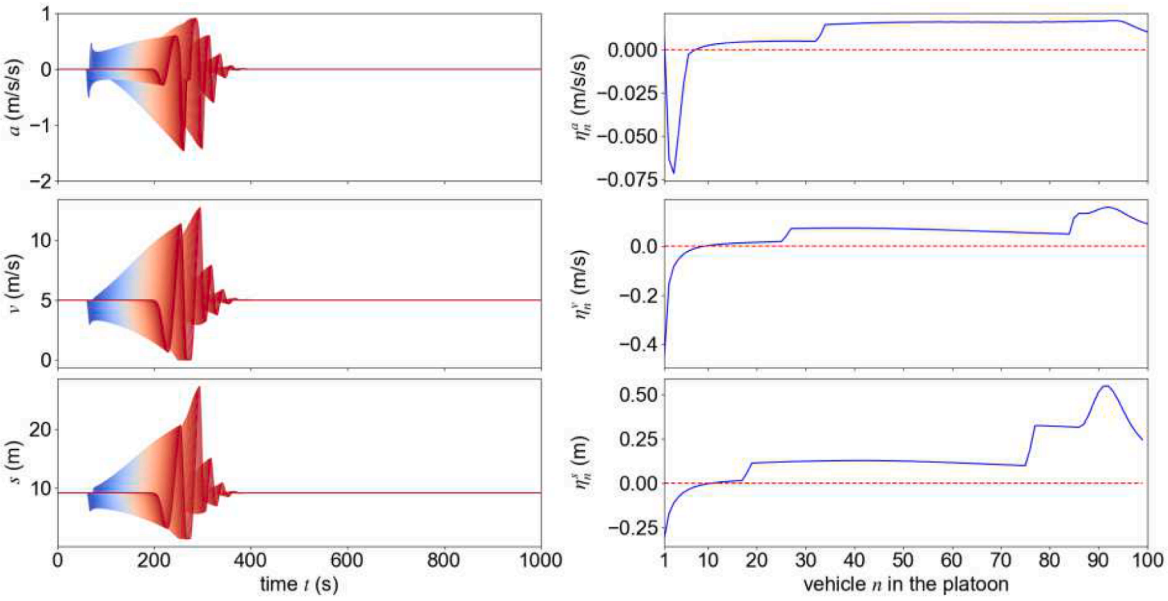
6. Discussion

6.1. Applicability of the framework for complex neural networks

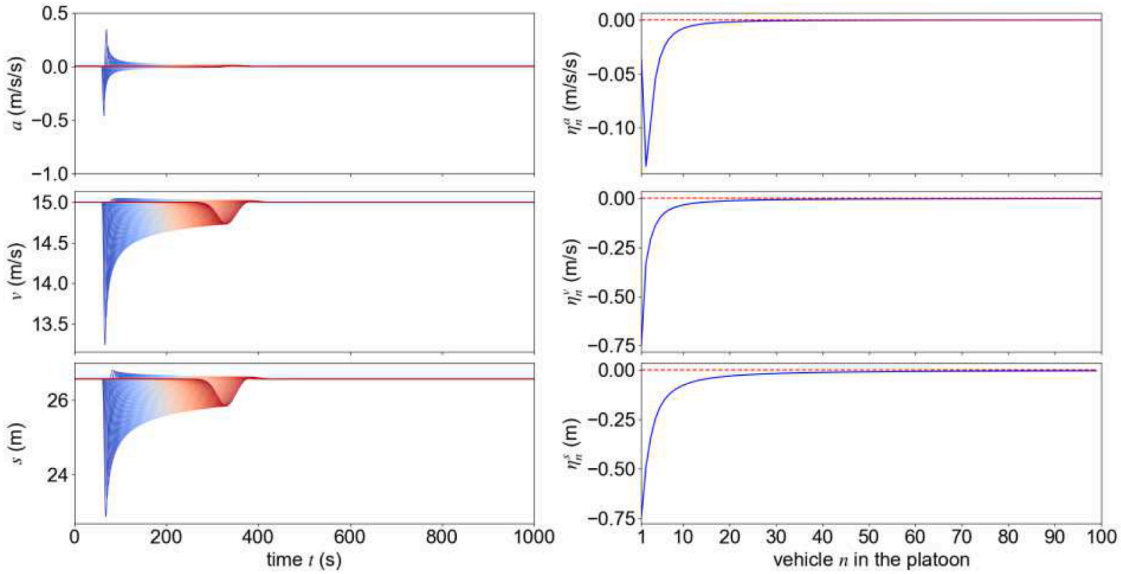
Compared to the conventional method of string stability analysis, we carefully accommodate each step as the EADC framework, including estimation of steady states, approximation of non-differentiable functions, derivation of string stability criteria, and calculation of partial derivatives. These proposed general methods make it applicable for various neural networks even with complex architectures and may be transferred to analyze the stability of other NN-based dynamic systems, which is discussed below.

Specifically, the estimation of steady states is first transferred to a corresponding optimization problem that can be applied for any trained NN-based models. Although it is a brute-force search to solve this problem, the computation cost is negligible thanks to the limited size of candidate solutions.

Regarding the non-differentiability of NN-based CF models, which arises from the non-differentiable activation functions, the aim of the second step is to replace the functions (e.g., the ReLU) with corresponding smooth approximation (e.g., the Softplus). However,



(a) $v_e = 5\,m/s$



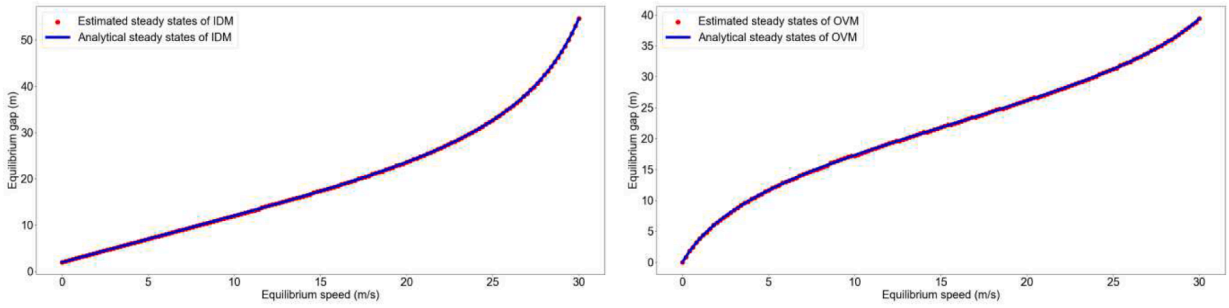
(b) $v_e = 15\,m/s$

Fig. 14. Simulation results of the Huang-LSTM model.

recent studies in deep learning with NN have developed other smooth activation functions such as Swish and Mish (Dubey et al., 2022), as differentiability is also desirable for activation functions to enable gradient-based optimization in training. The non-differentiability can be easily removed from NNs with the popularity of these continuous alternatives.

As for the third step, the derived string stability criteria are generic for other NN-based CF models with similar input and output, despite the complexly internal structures, because the linearized expressions remain unchanged. Meanwhile, the basic input and historical information incorporated input have covered the majority configurations of NN-based CF models, as the basic input considering the subject vehicle and the preceding vehicle's attributes is sufficient to capture the essence of CF behavior (Vasebi et al., 2021) and additional temporal information can enhance the precision with memory effect (Wang et al., 2019).

To circumvent the complexity of NN models, the last step is the calculation of partial derivatives in the stability criteria using AD. As NN models ultimately execute a sequence of elementary arithmetic operations and elementary functions regardless of their structures,



(a) IDM (b) OVM

Fig. A1. Estimated and analytical results of the steady states of IDM and OVM.

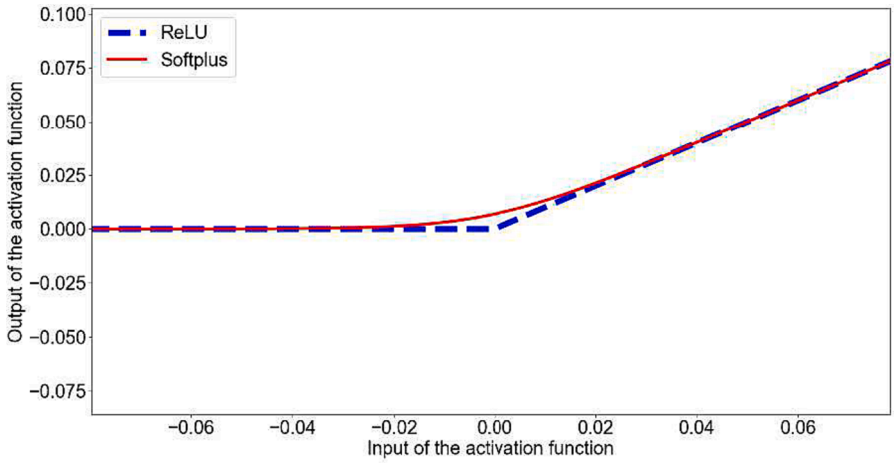


Fig. B1. Plots zoomed in zero for ReLU and Softplus activation functions.

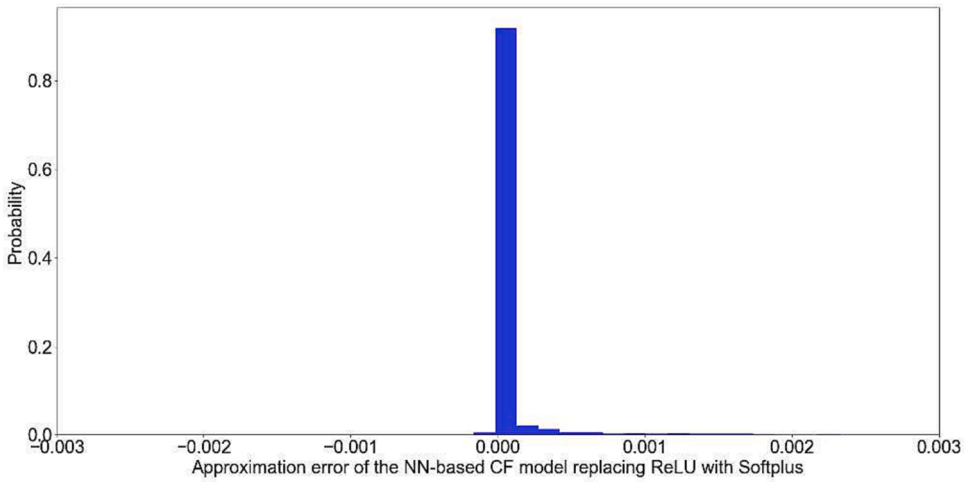


Fig. B2. The distribution of model-level approximation error.

e.g., the convolution and pooling layer in CNN can be seen as the combination of addition and multiplication (Li et al., 2022), by applying the chain rule repeatedly to these operations, partial derivatives of arbitrary order can be computed automatically via AD. That is also the reason that, backpropagation, a special case of AD, is usually used in the training of NN.

6.2. Applications of the stability analysis framework

Recently, CAVs have attracted lots of attention from both researchers and practitioners due to their potential to revolutionize the driving behaviors and traffic flow operations, and ultimately resolve many transportation-related issues, e.g., road safety, traffic congestion, energy consumption and vehicle emissions. Due to the superior performance in complex decision making, NN-based methods have been extensively employed in modeling CAVs (Yavas et al., 2022). However, the stability characteristics of these models are routinely ignored in either design or evaluation stage, despite the close relationship between CF stability and traffic oscillations, riding comfort, etc. The disregard is mainly rooted in the fact that conventional string stability analysis method is not applicable for the non-differentiable and complex NNs. With the proposed EADC framework, the theoretical assessment of NN-based CAV models in terms of CF stability is achievable as well as the design of stable CAV controller. For example, the calculated string stability can be added as a component of the reward function in RL and punish the unstable CF policy to learn to maintain stability. Another way is to embed the derived stability criteria into the loss function of the NN models as a soft constraint, which is a typical method of physics-informed neural network by introducing learning bias (Karniadakis et al., 2021). Moreover, the stability analysis results bring more insights for the training of NN-based CF models. As the real-world vehicle trajectories include both stable and unstable vehicles, deliberated selection of training dataset based on the stability analysis for the development of more stable CAV models thus becomes possible.

7. Conclusions

Despite the recent significant progress in modeling CF behaviors using deep learning, the inherent characteristics of the NN-based CF models remain unclear, particularly the string stability that is closely associated with the traffic oscillation has not been investigated yet because of their extreme non-linearity and complexity. To fill this gap, this study developed a generic EADC framework for the theoretical analysis of NN-based CF models' string stability, which mainly consists of ***Estimation*** (of steady states), ***Approximation*** (of non-differentiable functions), ***Derivation*** (of general stability criteria), and ***Calculation*** (of partial derivatives).

Specifically, we converted the steady states estimation issue of NN-based CF models into an equivalent optimization problem. We then approximated the non-differentiable NN-based CF models by replacing the non-differentiable part, i.e., the activation function with its smooth approximation, and derived the general stability criteria for both the basic NN-based model and the multi-step historical information incorporated NN-based models with the Laplace transform based method. Additionally, we used the automatic differentiation method to facilitate the calculation of partial derivatives coefficients in the stability criteria.

The EADC framework was applied to the Mo-MLP model for the basic model, and the Huang-LSTM model for the multi-step historical information incorporated model. We evaluated the string stability of the two models for different steady states and compared their stability features with IDM. The results reveal that the stability of NN-based CF models vary with different traffic states which is largely due to their complex architectures.

In addition, we conducted a series of numerical simulations to examine the validity of the EADC framework. The simulation results are completely consistent with the theoretical results, which indicates the capability of the framework for assessing the stability of newly developed NN-based CF models.

Finally, the applicability and future applications of the EADC framework were discussed. We believe that the proposed EADC stability analysis method can be a generic and powerful tool in assessing the stability of NN-based CF models and guiding the development of traffic flow optimizing CAV controllers.

CRediT authorship contribution statement

Xiaohui Zhang: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Writing – original draft. **Jie Sun:** Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Validation, Writing – review & editing. **Zuduo Zheng:** Formal analysis, Investigation, Validation, Writing – review & editing. **Jian Sun:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Supervision, Validation, Project administration, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research is jointly sponsored by the National Natural Science Foundation of China (52125208) and the Shanghai Municipal Science and Technology Major Project (No. 2021SHZDZX0100).

Appendix A. . Validation of the steady state estimation

Two widely used CF models with explicit functions of steady states are adopted to validate the proposed optimization problem for steady state estimation, including IDM and OVM. The IDM is defined by Eqs. (42) and (43) with typical parameters as follows: desired

velocity $v_0 = 33.3m/s$, desired time headway $T = 1s$, maximum acceleration $\alpha = 1m/s^2$, comfortable deceleration $\beta = 1.5m/s^2$, minimum distance $s_0 = 2m$. The OVM with typical parameters is also adopted for verification, where its acceleration equation is given by:

$$\dot{v} = \frac{v_{opt}(s) - v}{\tau} \tag{A1}$$

$$v_{opt}(s) = v_0 \frac{\tanh\left(\frac{s}{\Delta s} - \beta\right) + \tanh\beta}{1 + \tanh\beta} \tag{A2}$$

where the adaptation time $\tau = 0.65s$, desired speed $v_0 = 33.3m/s$, transition width $\Delta s = 15m$ and form factor $\beta = 1.5$. The analytical function of steady states is shown in Eq. (A2). The estimated results in conjunction with the analytical function of steady states $s_e(v)$ for both IDM and OVM are shown in Fig. A1. It shows that the estimated results totally accord with the analytical results, which demonstrates the capability of the proposed optimization problem for estimating steady states.

Appendix B. . Validation of the approximation of non-differentiable functions

To validate the feasibility of approximation, we comprehensively evaluate the approximation error between the ReLU and Softplus functions, as well as the errors between the ReLU-based and Softplus-based NN models. The maximum absolute error for the function-level approximation in zero input is around 0.007 as shown in Fig. B1, and the RMSE for the input range from −10 to 10 is 1.4×10^{-4}, which shows high precision. Regarding the model-level approximation, we utilize a simple MLP model with ReLU activation function and evaluate the approximation errors when replacing the ReLU with Softplus function in the model without other changes. We compare the output of these two NN models with the same inputs. As shown in Fig. B2, the distribution of errors predominantly converges around zero, with an RMSE of 3.7×10^{-4}. The high-precision of both function- and model-level approximation lay a foundation for the high consistency between theoretical string stability and simulation results and further verify the practicability of the proposed EADC framework.

References

Abiodun, O.I., Jantan, A., Omolara, A.E., Dada, K.V., Mohamed, N.A., Arshad, H., 2018. State-of-the-art in artificial neural network applications: A survey. Heliyon 4, e00938.

Ahn, S., Laval, J., Cassidy, M.J., 2010. Effects of Merging and Diverging on Freeway Traffic Oscillations. Transp. Res. Rec. 2188, 1–8. https://doi.org/10.3141/2188-01.

Bando, M., Hasebe, K., Nakanishi, K., Nakayama, A., 1998. Analysis of optimal velocity model with explicit delay. Phys. Rev. E 58, 5429–5435. https://doi.org/10.1103/PhysRevE.58.5429.

Burden, R.L., 2005. Numerical analysis, 8th ed. Thomson Brooks/Cole, Belmont, Calif.

Chandler, R.E., Herman, R., Montroll, E.W., 1958. Traffic Dynamics: Studies in Car Following. Oper. Res. 6, 165–184. https://doi.org/10.1287/opre.6.2.165.

Chong, L., Abbas, M.M., Medina, A., 2011. Simulation of driver behavior with agent-based back-propagation neural network. Transp. Res. Rec. 44–51 https://doi.org/10.3141/2249-07.

Dubey, S.R., Singh, S.K., Chaudhuri, B.B., 2022. Activation functions in deep learning: A comprehensive survey and benchmark. Neurocomputing 503, 92–108. https://doi.org/10.1016/j.neucom.2022.06.111.

Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C., Garcia, R., 2000. Incorporating Second-Order Functional Knowledge for Better Option Pricing. In: Proc. of the 13th Int. Conf. on Neural Inf. Process. Syst., NIPS'00. MIT Press, Cambridge, MA, USA, pp. 451–457.

Feng, S., Zhang, Y., Li, S.E., Cao, Z., Liu, H.X., Li, L., 2019. String stability for vehicular platoon control: Definitions and analysis methods. Annu. Rev. Control 47, 81–97. https://doi.org/10.1016/j.arcontrol.2019.03.001.

Güneş Baydin, A., Pearlmutter, B.A., Andreyevich Radul, A., Mark Siskind, J., 2015. Automatic differentiation in machine learning: a survey. J. Mach. Learn. Res. 18, 1–43. 10.48550/arxiv.1502.05767.

Gunter, G., Gloudemans, D., Stern, R.E., McQuade, S., Bhadani, R., Bunting, M., Delle Monache, M.L., Lysecky, R., Seibold, B., Sprinkle, J., Piccoli, B., Work, D.B., 2021. Are Commercially Implemented Adaptive Cruise Control Systems String Stable? IEEE Trans Intell. Transp. Syst. 22, 6992–7003. https://doi.org/10.1109/TITS.2020.3000682.

Hart, F., Okhrin, O., Treiber, M., 2021. Formulation and validation of a car-following model based on deep reinforcement learning. ArXiv. http://arxiv.org/abs/2109.14268.

Herman, R., Montroll, E.W., Potts, R.B., Rothery, R.W., 1959. Traffic Dynamics: Analysis of Stability in Car Following. Oper. Res. 7, 86–106. https://doi.org/10.1287/opre.7.1.86.

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

Huang, X., Sun, J., Sun, J., 2018. A car-following model considering asymmetric driving behavior based on long short-term memory neural networks. Transp. Res. Part C Emerg. Technol. 95, 346–362. https://doi.org/10.1016/j.trc.2018.07.022.

Jia, H., Juan, Z., Ni, A., 2003. Develop a car-following model using data collected by "five-wheel system," in: Proc. of the 2003 IEEE Intell. Transp. Syst. Conf. IEEE, pp. 346–351. 10.1109/ITSC.2003.1251975.

Jiang, R., Jin, C., Zhang, H., Huang, Y., Tian, J., Wang, W., Hu, M., Wang, H., Jia, B., 2017. Experimental and Empirical Investigations of Traffic Flow Instability. Transp. Res. Procedia 23, 157–173. https://doi.org/10.1016/j.trpro.2017.05.010.

Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L., 2021. Physics-Informed Machine Learning. Nat. Rev. Phys. 3, 422–440. https://doi.org/10.1038/s42254-021-00314-5.

Khodayari, A., Ghaffari, A., Kazemi, R., Braunstingl, R., 2012. A Modified Car-Following Model Based on a Neural Network Model of the Human Driver Effects. IEEE Trans. Syst. Man Cybern.: Syst. 42, 1440–1449. https://doi.org/10.1109/TSMCA.2012.2192262.

Kiran, B.R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A.A.A., Yogamani, S., Perez, P., 2022. Deep Reinforcement Learning for Autonomous Driving: A Survey. IEEE Trans. Intell. Transp. Syst. 23, 4909–4926. https://doi.org/10.1109/TITS.2021.3054625.

Lenz, H., Wagner, C.K., Sollacher, R., 1999. Multi-anticipative car-following model. Eur. Phys. J. B 7, 331–335. https://doi.org/10.1007/s100510050618.

Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J., 2022. A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. IEEE Trans. Neural Netw. Learn. Syst. 33, 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827.
Liu, X., Goldsmith, A., Mahal, S.S., Hedrick, J.K., 2001. Effects of communication delay on string stability in vehicle platoons, in: 2001 IEEE Intell. Transp. Syst. Conf. IEEE, pp. 625–630. 10.1109/ITSC.2001.948732.
Liu, J., Jiang, R., Zhao, J., Shen, W., 2023. A quantile-regression physics-informed deep learning for car-following model. Transp. Res. Part C Emerg. Technol. 154, 104275 https://doi.org/10.1016/j.trc.2023.104275.
Ma, L., Qu, S., 2020. A sequence to sequence learning based car-following model for multi-step predictions considering reaction delay. Transp. Res. Part C Emerg. Technol. 120, 102785 https://doi.org/10.1016/j.trc.2020.102785.
Ma, K., Wang, H., Zuo, Z., Hou, Y., Li, X., Jiang, R., 2022. String stability of automated vehicles based on experimental analysis of feedback delay and parasitic lag. Transp. Res. Part C Emerg. Technol. 145, 103927 https://doi.org/10.1016/j.trc.2022.103927.
Makridis, M., Mattas, K., Anesiadou, A., Ciuffo, B., 2021. OpenACC. An open database of car-following experiments to study the properties of commercial ACC systems. Transp. Res. Part C Emerg. Technol. 125, 103047 https://doi.org/10.1016/j.trc.2021.103047.
Mo, Z., Shi, R., Di, X., 2021. A physics-informed deep learning paradigm for car-following models. Transp. Res. Part C Emerg. Technol. 130, 103240 https://doi.org/10.1016/j.trc.2021.103240.
Montanino, M., Punzo, V., 2015. Trajectory data reconstruction and simulation-based validation against macroscopic traffic patterns. Transp. Res. Part B Methodol. 80, 82–106. https://doi.org/10.1016/J.TRB.2015.06.010.
Monteil, J., Billot, R., Sau, J., El Faouzi, N.-E., 2014. Linear and Weakly Nonlinear Stability Analyses of Cooperative Car-Following Models. IEEE Trans. Intell. Transp. Syst. 15, 2001–2013. https://doi.org/10.1109/TITS.2014.2308435.
Mozaffari, S., Al-Jarrah, O.Y., Dianati, M., Jennings, P., Mouzakitis, A., 2022. Deep Learning-Based Vehicle Behavior Prediction for Autonomous Driving Applications: A Review. IEEE Trans. Intell. Transp. Syst. 23, 33–47. https://doi.org/10.1109/TITS.2020.3012034.
Nair, V., Hinton, G.E., 2010. Rectified Linear Units Improve Restricted Boltzmann Machines, in: Proc. of the 27th Int. Conf. on Mach. Learn., ICML'10. Omnipress, Madison, WI, USA, pp. 807–814.
Orosz, G., Wilson, R.E., Stépán, G., 2010. Traffic jams: dynamics and control. Phil. Trans. r. Soc. A 368, 4455–4479. https://doi.org/10.1098/rsta.2010.0205.
Orosz, Gábor, Moehlis, J., Bullo, F., 2011. Delayed Car-Following Dynamics for Human and Robotic Drivers, in: Int. Conf. Multibody Sys. Nonlinear Dyn. Control. ASMEDC, pp. 529–538. 10.1115/DETC2011-48829.
Peppard, L.E., 1974. String Stability of Relative-Motion PID Vehicle Control Systems. IEEE Trans. Automat. Contr. 19, 579–581. https://doi.org/10.1109/TAC.1974.1100652.
Punzo, V., Zheng, Z., Montanino, M., 2021. About calibration of car-following dynamics of automated and human-driven vehicles: Methodology, guidelines and codes. Transp. Res. Part C Emerg. Technol. 128, 103165 https://doi.org/10.1016/J.TRC.2021.103165.
Rudin, C., 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. 2019 1:5 1, 206–215. 10.1038/s42256-019-0048-x.
Sau, J., Monteil, J., Billot, R., El Faouzi, N.-E., 2014. The root locus method: application to linear stability analysis and design of cooperative car-following models. Transportmetrica b: Transport Dyn. 2, 60–82. https://doi.org/10.1080/21680566.2014.893416.
Sun, J., Zheng, Z., Sharma, A., Sun, J., 2023. Stability and extension of a car-following model for human-driven connected vehicles. Transportation Research Part C: Emerging Technologies 155, 104317.
Sun, J., Zheng, Z., Sun, J., 2018. Stability analysis methods and their applicability to car-following models in conventional and connected environments. Transp. Res. Part B Methodol. 109, 212–237. https://doi.org/10.1016/J.TRB.2018.01.013.
Sun, J., Zheng, Z., Sun, J., 2020. The relationship between car following string instability and traffic oscillations in finite-sized platoons and its use in easing congestion via connected and automated vehicles with IDM based controller. Transp. Res. Part B Methodol. 142, 58–83. https://doi.org/10.1016/j.trb.2020.10.004.
Swaroop, D., 1997. String stability of interconnected systems: An application to platooning in automated highway systems. Transp. Res. Part A Policy Pract. 31, 65. https://doi.org/10.1016/S0965-8564(97)88297-3.
Swaroop, D., Hedrick, J.K., 1996. String stability of interconnected systems. IEEE Trans. Automat. Contr. 41, 349–357. https://doi.org/10.1109/9.486636.
Treiber, M., Kesting, A., 2013. Traffic Flow Dynamics, Traffic Flow Dynamics: Data, Models and Simulation. Springer Berlin Heidelberg, Berlin, Heidelberg. 10.1007/978-3-642-32460-4.
van Dyk, D.A., Meng, X., 2001. The Art of Data Augmentation. J. Comput. Graphical Stat. 10, 1–50. https://doi.org/10.1198/10618600152418584.
Vasebi, S., Hayeri, Y.M., Jin, P.J., 2021. Surrounding Vehicles' Contribution to Car-Following Models: Deep-Learning-Based Analysis. Transp. Res. Rec. 2675, 623–640. https://doi.org/10.1177/03611981211018693.
von zur Gathen, J., Gerhard, Jürgen, 2013. Modern Computer Algebra, Modern Computer Algebra. Cambridge University Press. 10.1017/CBO9781139856065.
Wang, X., Jiang, R., Li, L., Lin, Y., Zheng, X., Wang, F., 2018. Capturing Car-Following Behaviors by Deep Learning. IEEE Trans. Intell. Transp. Syst. 19, 910–920. https://doi.org/10.1109/TITS.2017.2706963.
Wang, X., Jiang, R., Li, L., Lin, Y., Wang, F., 2019. Long memory is important: A test study on deep-learning based car-following model. Physica A 514, 786–795. https://doi.org/10.1016/j.physa.2018.09.136.
Wilson, R.E., 2008. Mechanisms for spatio-temporal pattern formation in highway traffic models. Phil. Trans. r. Soc. A 366, 2017–2032. https://doi.org/10.1098/rsta.2008.0018.
Wilson, R.E., Ward, J.A., 2011. Car-following models: fifty years of linear stability analysis – a mathematical perspective. Transp. Plann. Technol. 34, 3–18. https://doi.org/10.1080/03081060.2011.530826.
Yavas, U., Kumbasar, T., Ure, N.K., 2022. Model-Based Reinforcement Learning for Advanced Adaptive Cruise Control: A Hybrid Car Following Policy, in: 2022 IEEE Intell. Veh. Symp. (IV). IEEE, pp. 1466–1472. 10.1109/IV51971.2022.9827279.
Ye, Y., Zhang, X., Sun, J., 2019. Automated vehicle's behavior decision making using deep reinforcement learning and high-fidelity simulation environment. Transp Res Part C Emerg Technol 107, 155–170. https://doi.org/10.1016/j.trc.2019.08.011.
Zheng, Z., Ahn, S., Monsere, C.M., 2010. Impact of traffic oscillations on freeway crash occurrences. Accid. Anal. Prev. 42, 626–636. https://doi.org/10.1016/J.AAP.2009.10.009.
Zheng, J., Suzuki, K., Fujita, M., 2013. Car-following behavior with instantaneous driver-vehicle reaction delay: A neural-network-based methodology. Transp. Res. Part C Emerging Technol. 36, 339–351. https://doi.org/10.1016/j.trc.2013.09.010.
Zhou, M., Qu, X., Li, X., 2017. A recurrent neural network based microscopic car following model to predict traffic oscillation. Transp. Res. Part C Emerging Technol. 84, 245–264. https://doi.org/10.1016/j.trc.2017.08.027.
Zhou, Y., Zhong, X., Chen, Q., Ahn, S., Jiang, J., Jafarsalehi, G., 2023. Data-driven analysis for disturbance amplification in car-following behavior of automated vehicles. Transp. Res. Part B Methodol. 174, 102768 https://doi.org/10.1016/j.trb.2023.05.005.
Zhu, M., Wang, X., Wang, Y., 2018. Human-like autonomous car-following model with deep reinforcement learning. Transp. Res. Part C Emerging Technol. 97, 348–368. https://doi.org/10.1016/j.trc.2018.10.024.
Zhu, M., Wang, Y., Pu, Z., Hu, J., Wang, X., Ke, R., 2020. Safe, efficient, and comfortable velocity control based on reinforcement learning for autonomous driving. Transp. Res. Part C Emerging Technol. 117, 102662 https://doi.org/10.1016/J.TRC.2020.102662.