Transportation Research Part C 84 (2017) 245–264

Contents lists available at ScienceDirect

Transportation Research Part C

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

A recurrent neural network based microscopic car following model to predict traffic oscillation

Mofan Zhou [a,b], Xiaobo Qu [a,c,*], Xiaopeng Li [d]

[a] School of Civil and Environmental Engineering, University of Technology Sydney, NSW 2007, Australia
[b] Griffith School of Engineering, Gold Coast Campus, Griffith University, QLD 4222, Australia
[c] Department of Architecture and Civil Engineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden
[d] Department of Civil and Environmental Engineering, University of South Florida, FL 33620, United States

ARTICLE INFO

Article history:
Received 15 March 2017
Received in revised form 30 August 2017
Accepted 30 August 2017
Available online 12 September 2017

Keywords:
Recurrent neural networks
Traffic flow dynamics
Car-following
Oscillation

ABSTRACT

This paper proposes a recurrent neural network based microscopic car following model that is able to accurately capture and predict traffic oscillation. Neural network models have gained increasing popularity in many fields and have been applied in modelling microscopic traffic flow dynamics due to their parameter-free and data-driven nature. We investigate the existing neural network based microscopic car following models, and find out that they are generally accurate in predicting traffic flow dynamics under normal traffic operational conditions. However, they do not maintain accuracy under conditions of traffic oscillation. To bridge this research gap, we first propose four neural network based models and evaluate their applicability to predict traffic oscillation. It is found that, with an appropriate structure and objective function, the recurrent neural network based model has the capability of perfectly re-establishing traffic oscillations and distinguish drivers characteristics. We further compare the proposed model with a classical car following model (Intelligent Driver Model). Based on our case study, the proposed model outperforms the classical car following model in predicting traffic oscillations with different driver characteristics.

1. Introduction

Traffic oscillation, or stop-and-go traffic, has raised many concerns to both transport practitioners and researchers. Its negative impacts include deteriorating highway mobility, increasing safety concerns, excessive fuel consumption and greenhouse gas emissions (Bilbao-Ubillos, 2008; Zheng et al., 2010; Song et al., 2013). Various car-following models have been proposed and improved in order to accurately capture the traffic oscillation. See for example Brackstone and McDonald (1999) and Saifuzzaman and Zheng (2014) for reviews.

On the other hand, machine learning techniques, especially artificial neural networks, have been widely applied in many modelling fields, such as speech recognition (Sak et al., 2014), handwriting recognition (Graves et al., 2009), and self-driving cars (Santana and Hotz, 2016). With the increases in data quantity and computational power, the machine learning approach is more efficient and accurate than ever before. Using machine learning to model car-following behaviours has also been studied intensively (Papathanasopoulou and Antoniou, 2015; He et al., 2015; Aghabayk et al., 2014; Chong et al., 2011,

* Corresponding author at: School of Civil and Environmental Engineering, University of Technology Sydney, NSW 2007, Australia.
E-mail address: drxiaoboqu@gmail.com (X. Qu).

http://dx.doi.org/10.1016/j.trc.2017.08.027
0968-090X/

2013; Mathew and Ravishankar, 2012; Khodayari et al., 2012; Panwai and Dia, 2007; Jia et al., 2003). Compared with classical car-following modelling approaches, one unique benefit of machine learning is its nonparametric characteristic. Typically, in classical car-following models, one has to calibrate the parameters in the empirical equation to achieve a high level of accuracy. Whereas neural network based models can learn from field data and automatically generate the car-following model without many artificial parameters. They often produce superior results (Aghabayk et al., 2014; Mathew and Ravishankar, 2012; Khodayari et al., 2012; Chong et al., 2011). Unfortunately, these neural network based models have not been tested under oscillating traffic conditions.

Therefore, we implement different neural network architectures under traffic oscillations, and the results indicate that neural network based models can produce highly accurate results when the tests are based on the inputs from field data. However, when being tested based on the inputs from the previous predictions similar to iterative simulation, the test results yield unsatisfactory performance. The results suggest that the unsatisfactory performance is mainly caused by insufficient data inputs and inaccurate measurements.

This paper aims to bridge the gap using a Recurrent Neural Network (RNN) based car-following model. RNNs are a variation of the neural network family, with greater strength in predicting sequential data (Lipton et al., 2015; Mikolov et al., 2010). The car-following behaviours can be treated as a behaviour sequence due to the fact that the order of driving conditions matters when a driver is going to take their next action. For example, an acceleration phase followed by a deceleration phase is different to the reverse. With this strength, an RNN-based model considers not only the local information (such as short-term or immediate traffic condition used in classical car-following models), but also the global information which is a long-term condition stores in a RNN memory. In other words, RNN model takes action more like human in terms of the memory-based decision-making. In the car-following literature, a memory-based car-following model has also been paid attention to due to the driving history is an essential factor in predicting the next action. Treiber and Helbing (2003) address the historical impact by introducing an additional dynamical variable to a classical model. From this perspective, the memory effect does matter regarding developing a more accurate car-following model. Therefore, we develop an RNN-based model to capture car-following behaviours in oscillating scenarios and extended it to non-oscillating cases. The results illustrate that the RNN-based model can successfully predict both oscillating and non-oscillating car-following rules and increase accuracy by learning driver behaviours.

The rest of the paper is organised as follows. Section 2 briefly reviews the literature on traffic oscillation and neural network based car-following models. Section 3 shows the data and the pre-processing method we adopted to regularise data noise. Section 4 compares existing neural network based car-following models under oscillating traffic circumstances, and raises the inaccuracy and other potential issues produced from using these models. At the end of this section, we attempt to fix these issues by proposing a new neural network model that predicts partial gaps. This new model improves the performance but has a shortcoming in handling less informative data. In Section 5, we propose the RNN-based model to further improve modelling capacity in capturing driving behaviours. Section 6 concludes this paper.

2. Literature review

2.1. Oscillation

With the urbanisation of our cities, traffic volume has been increasing continuously. A negative consequence to our daily travel is the notorious phenomenon known as traffic oscillation. The formation and propagation mechanisms of traffic oscillation, also known as stop-and-go traffic, have been intensively investigated during recent years in the literature. A general car-following model may not describe oscillation accurately. Therefore, researchers have proposed or improved upon existing models to better describe and illustrate the oscillating patterns on congested highways.

The measuring and analysing methodologies for traffic oscillation have been developed and extended, which has improved the understanding of the mechanisms of oscillations. The traffic flow on congested highways can be described as three phases including free flow, synchronized flow and wide moving jam. Hence the three-phase traffic theory was proposed to fully explain the nonlinearity of traffic flow on highways (Kerner, 1999, 2012). However, Treiber et al. (2010) discuss the inconsistent use of the term "traffic phases" and show the three-phase traffic theory can be reproduced with simple two-phase models with suitably specified model parameters and a consideration of factors characteristic for real traffic flows. In the term of vehicular speed, Tian et al. (2016) suggests that the standard deviation of speed grows concavely along vehicles in the oscillation, which they find is a universal property of traffic oscillation. Li et al. (2010) propose a frequency spectrum analysis approach that enhanced the measurement of the periodicity and magnitude in a traffic oscillation. The oscillation triggers at the vehicle level, and the propagation features of oscillation, were analysed using the Wavelet Transform (WT) (Zheng et al., 2011a, 2011b), which enables investigation of the oscillation behaviours down to the micro level.

Also, several car-following models have been proposed to capture traffic oscillation. Bando et al. (1998) investigate the properties of congestion and the delay time of car motion using optimal velocity model. Furthermore, the behaviour of each driver differs, especially in a traffic oscillation. An oscillation is more likely to be instigated by an aggressive driver who may maintain a small response time and minimum spacing, which may also lead to capacity drop (Chen et al., 2014). According to this finding, to achieve more accurate car-following simulation, it is necessary to separate drivers depending on their driving behaviours (Chen et al., 2012; Laval and Leclercq, 2010). Laval and Leclercq (2010) introduce an additional parameter η to the

original Newell car-following formula (Newell, 2002) to distinguish driver's behaviours. However, Chen et al. (2012) find the behaviour model proposed in (Laval and Leclercq, 2010) captures one ideal case of driver behaviour but far from sufficient to cover the general patterns of traffic oscillation. Therefore, they further improve this model and group drivers with respect to their different driving behaviours including originally aggressive, originally timid and originally Newell. Their results indicate more precise microscopic car-following patterns. On the other hand, Li and Ouyang (2011) presented the Describing-Function Approach (DFA) to predict the propagation properties in oscillations for nonlinear car-following behaviour, and validated the approach given in (Li et al., 2012). Li et al. (2014) further employ this approach to investigate fuel consumption and emissions and explore ways of using connected autonomous vehicles to dampen traffic oscillation (Li et al., 2014). Built upon the describing function method, Rhoades et al. (2016) propose a method for calibrating nonlinear car-following model to quantitatively reproduce traffic oscillation. Laval et al. (2014) suggest that driver error contributes to oscillation formation and propagation. They improve the kinematic wave model to better reproduce driver behaviours in an oscillation. To capture the time-varying properties of oscillations, Zhao et al. (2014) focus on the analysis of divided short-term windows.

The above approaches and models perform better in oscillating traffic cases, due to their calibrated parameters and model constraints. Most of these existing studies are based on parametric car-following dynamic models constructed based on physics, behaviour and sometimes artificial parameters for tuning model predictions. While these models may yield insightful explanations to traffic oscillation in certain settings, the fixed structure of these models may limit their flexibility and adaptivity to general traffic states and infrastructure settings. It is in general difficult to construct a universal parametric model compatible with various traffic scenarios, infrastructure types and data observations. Recently emerged artificial neural networks have been shown powerful, flexible and adaptable in addressing large-scale nonlinear problems with general settings. They are found suitable to be applied to studies of traffic oscillation. Moreover, neural network based models are parameter-free and data-driven, and thus they are compatible with different traffic states, infrastructure configurations and available data. Therefore, neural network based car-following models hold promise for an alternative solution to predicting car-following and traffic oscillation with higher flexibility and robustness.

2.2. Neural networks and car-following models

As is the case with many well-known classical car-following models, the inputs into the network vary across studies. However, the predicted output from such models is usually determined as acceleration rate, velocity or gap distance for the target vehicle at the next time point. The following subsection reviews the existing research methods predicting each of those outputs.

2.2.1. Predicting acceleration

The acceleration rate has a direct link to the control of engine power. Many classical car-following models have already made great contributions towards predicting acceleration rates, such as the Gazis–Herman–Rothery model, known as the GHR model (Herman et al., 1959; Chandler et al., 1958), the Optimal Velocity model (OVM) (Bando et al., 1995, 1998), the Intelligent Driver Model (IDM) (Treiber et al., 2000, 2007), and the Full Velocity Difference Model (FVDM) (Jiang et al., 2001). Therefore, the acceleration rate would also be an ideal predicted output from a neural network.

Jia et al. (2003) introduce a four-layer neural network (including an input layer and an output layer). This neural network takes four input elements that consist of relevant speed, desired speed, follower speed and gap distance at the current time step and uses them to predict the acceleration at the last time step. The delay in prediction is caused by the consideration of drivers' reaction times. They also defined a nonlinear function for the hidden neural network layers, to improve the predictive accuracy. The data set they adopted was collected using the Five-Wheel System. The trained neural network is particularly capable of simulating human driving behaviours when predicting acceleration rates.

Chong et al. (2011) illustrate that it is still possible to predict acceleration rates accurately using a smaller-scale neural network. In their study, the neural network was built with only one hidden layer, and the number of inputs was reduced to three (speed, gap distance, and relevant speed). They trained the neural network on an episodic data set filtered from the Naturalistic Truck Driving Study (NTDS) (Olson et al., 2009). The predictive accuracy depended greatly on the episodes they selected for training. They declared that the accuracy of the output was restricted by the training episodes and it was not possible to generalise the results to different driving conditions.

To further improve the predictive accuracy, Khodayari et al. (2012) take all the inputs that Chong et al. (2011) considered and added the estimated instantaneous reaction delay. By training the network on the U.S. Federal Highway Administration's noise-filtered Next Generation SIMulation (NGSIM) (FHWA, 2008) data set, they obtained a highly accurate simulated result.

2.2.2. Predicting velocity

Predicting velocity is another option since it is shown directly on the car dashboard for the driver to see and control. Therefore, the car-following model tends to mimic the speed a driver would maintain instead of controlling the acceleration rate directly. To be in line with this idea, many classical car-following models attempt to calibrate velocity-based models, of which the Gipps model (Gipps, 1981) is one of the best-known representatives.

Panwai and Dia (2007) train neural networks for different driving modes to predict speed under various driving conditions. The inputs they selected were spacing headway and leading speed in different driving modes. Their data were collected from a congested single-lane road in Germany (Manstetten et al., 1997). However, the total amount of data used

for training and testing is limited to 2100 samples. This amount may not be enough for neural networks to learn a general model, and can be compared with the NGSIM database that has more than a million samples.

Mathew and Ravishankar (2012) build three different neural network architectures with different inputs to predict vehicle-type-dependent following behaviours. In other words, the first neural network architecture include only two inputs, the leader velocity and the gap distance. The other two neural networks are based on the first one and contained additional inputs, namely the leader and follower vehicle types. They declare that the two neural networks with vehicle type inputs could capture the field velocity data closely.

Analysing car-following models with respect to vehicle type has become a tendency due to the better results it produces. Aghabayk et al. (2014) focus on the modified neural network model only for heavy vehicles. The input vector consists of three elements: gap distance, follower speed and leader speed. Their results show that the modified neural network model fits the NGSIM velocity data better than the Gipps model.

2.2.3. Predicting gap distance

To predict the gap distance that a driver will maintain is another accepted method in the car-following model family. Because the gap is positional information, it can easily be extracted from trajectory data. In line with this idea, Helly (1961) develop a car-following model to describe drivers' desired following distance. Another gap-based model is based on seeking to maintain a minimum safe distance from the vehicle, and is called the Collision Avoidance (CA) model (Kometani and Sasaki, 1961). Moreover, Newell (2002) develop a simple car-following model considering space and time shifts, which also involved gap information.

The aforementioned models have also been widely tested in simulations. However, to the best of our knowledge, this type of model has not yet been transformed into a neural network based model. Using a neural network to predict gap distance can be studied further.

3. Data description and pre-processing

3.1. Data description

To be consistent with most previous research, we adopt the data set from NGSIM (FHWA, 2008) to train and test our models. Specifically, we select the trajectory data on the northbound direction of Interstate 80 (I-80) in Emeryville, California. The data record frequency is 0.1 s per frame. The trajectory data example in study site is shown in Fig. 1.

In order to exclude the correlation between the neural network training process and the test process, we separated the data into a training data set and a test data set. Vehicles travel between 4:00 and 4:15 experience fewer oscillations as shown in Fig. 1. Therefore, we choose the data on lane two collected from 4:00 to 4:15 p.m. on April 13, 2005 as the test data, and leave the rest of lanes in that period and all data from 5:00 to 5:15 p.m. to be the training data.

3.2. Pre-processing

The trajectory data appear unfiltered and exhibited some noise artefacts (Khodayari et al., 2012; Thiemann et al., 2008), as can be seen in Fig. 2. Therefore, we apply a moving-average filter (Thiemann et al., 2008) for a duration of 0.8 s to all raw trajectories, and obtain the numerical velocity/acceleration data from the first-/second-order finite differences of the position respectively. We also attempt to train a neural network with the unfiltered data. The network yield poor predictive

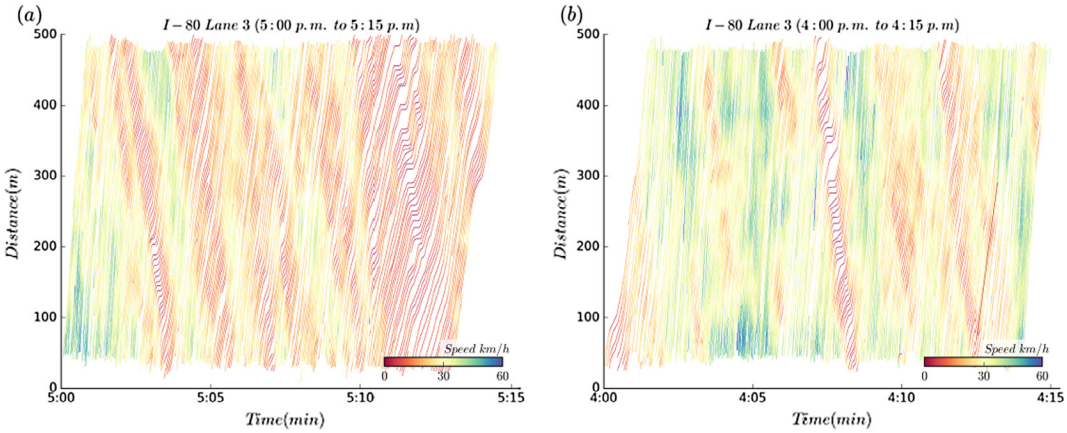

Fig. 1. Illustration of vehicle trajectories.

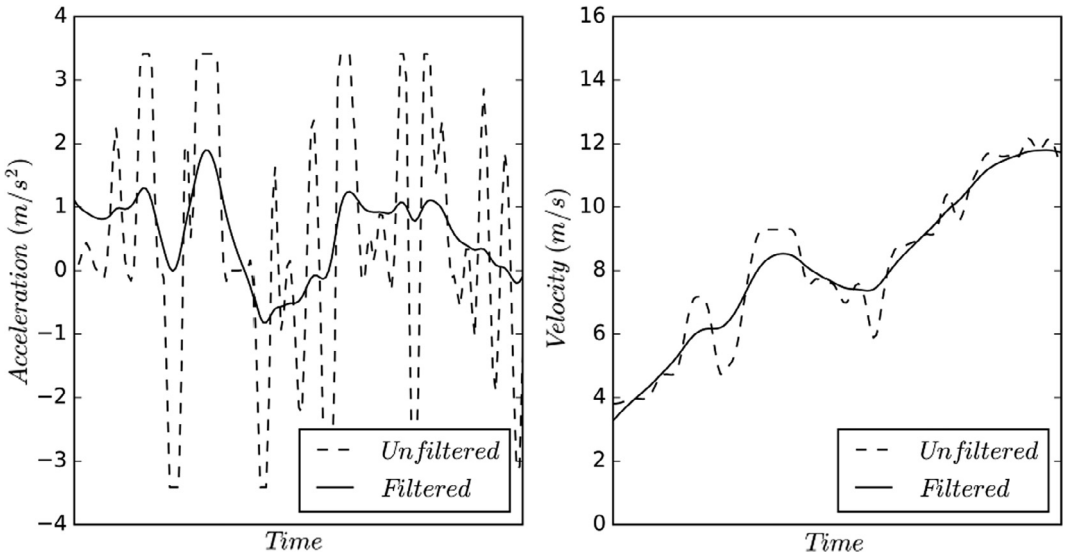

Fig. 2. Comparison of filtered and unfiltered data for acceleration rate and velocity.

performance, especially when predicting the turbulent acceleration rate. This proves that the neural network model is sensitive to the training data used.

Further data pre-processing is applied and listed as follows:

- Filtered trajectory data less than one second;
- Filtered the first (last) two seconds trajectory data for followers who experience a cut-in (move-out) lane-change;
- Filtered fake collision/suddenly jump data segments;
- Set negative vehicle movements to zero which is caused by moving average smoothing; and
- Refined differentiated accelerations to a range of $(-3.41376, 3.41376)$, which is consistent with the range in NGSIM acceleration data.

For training neural network models, the fitting procedure is data-point-based. We find previous 20-time steps for each data point and filter those points that do not have entire 20 previous steps. Therefore, the total training data contains 2,117,015 data points and randomly select 90% (2,011,164) of them for training, and the rest 10% (105,851) of them are used for validation.

To calibrate classical model, the calibration is trajectory-based. We further filter the trajectories less than five seconds to reflect a clear accumulative impact at the trajectory level. Hence, the training and test data for calibrating classical models contain 5725 and 477 trajectories respectively. Randomly selected 90% (5153) of the training data is for training and the rest 10% (572) is for validation. For RNN models, the sequence length is one of the key element that affects the model's performance. We set the sequence length to 600 steps (60 s) in order to continuously pass the hidden state through long sequences. Therefore, the number of total training trajectory for RNN models is reduced to 917 (826 for training and 91 for validation) as we only keep those trajectories longer than 60 s, while the number in the test set remains as 477.

4. Neural network based model

In the above literature review, we see that neural networks have the capacity to predict the acceleration rate and velocity. However, these highly accurate prediction methods have rarely been applied to real traffic data including oscillating traffic flow. We do not know what car-following behaviours will be exhibited when using those neural network models. Especially, there is a need to examine how a car follows another when encountering traffic oscillations. Therefore, in this section, we describe what happened when we applied different neural network architectures to test the performance of car-following manoeuvres at the trajectory level.

4.1. Architecture

The efficient back-propagation algorithm (Rumelhart et al., 1988) is the foundation for today's neural networks. A standard neural network feeds forward the input values for calculating the predicted outputs, and then uses the objective function to compute predicted errors from the real values. Lastly, it applies these errors as the gradients passing through the

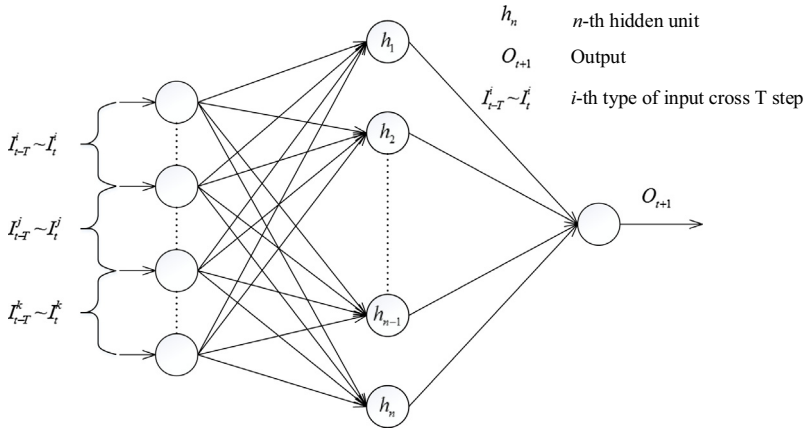

Fig. 3. General neural network architecture with time-series inputs.

network to update the network parameters. Through this procedure, the neural network with updated parameters can produce more accurate predictions.

To develop a general neural network model that is compatible with most of the neural network models mentioned in the literature review, we decide to adopt the neural network architecture shown in Fig. 3. I^i_t, I^j_t and I^k_t are the input type i, j and k at time t, and the input types can include leader speed, gap distance and the relevant speed of a following vehicle. Additionally, the inputs take time-series data into account in order to capture the driver's reaction time. Therefore, the inputs not only consist of one data point for each input type, but also batch data across the previous time steps (T denotes the number of time steps).

In the extreme case where $T = 1$, the neural network model outputs O_{t+1} only depend on the data of the last time step. The model, in this case, has the same architecture as instant models (Panwai and Dia, 2007; Chong et al., 2011; Mathew and Ravishankar, 2012). On the other hand, when taking T as a value greater than 1, the model has the same functionality as the models mentioned in other papers (Khodayari et al., 2012; Aghabayk et al., 2014; Jia et al., 2003), which consider drivers' reaction time or delay.

So as to model the nonlinearity of car-following behaviours, we chose the Rectified Linear Unit (ReLU) (Nair and Hinton, 2010) as the activation function in this neural network model. The equation is as follows:

$$h_n = \mathrm{ReLU}(W_n x + b_n) = \begin{cases} W_n x + b_n; & x > 0 \\ 0; & x \leqslant 0 \end{cases} \tag{1}$$

where h_n denotes the nth hidden unit in the hidden layer, W_n and b_n are the weight matrix and bias for h_n, respectively, and x is a vector of all inputs from the last layer. In general, if more than one hidden layer, the output vector of the last layer becomes the input vector of the next layer. The output O_{t+1} in Fig. 3 is computed as follows:

$$O_{t+1} = W_o h + b_o \tag{2}$$

This equation is a linear function of h. W_o and b_o are the parameters in the output layer, and h denotes the vector of all hidden units calculated using Eq. (1). The predicted output O_{t+1} is related to all inputs in vector x.

To quantify the quality of the prediction based on current network parameters, an objective function or so-called cost function is chosen, as follows:

$$C(W, b) = (O_{t+1} - y)^2 \tag{3}$$

where y denotes the real value corresponding to the outputs. In our case, the cost function is also known as the mean squared error. This cost is then used as the source for updating all network parameters through back-propagation.

4.2. Results, applicability and critiques of existing neural network model types

We conduct four simulations under different neural network architectures to test the predictive performance for the acceleration rate and velocity. For every simulation, the prediction is based only on the preceding trajectory from field data and the previously generated trajectory from the simulated vehicle. The neural network settings can be found in Table 1. T is fixed to one time step (or 0.1 s) for the neural networks, so that they predict based only on the information from the previous time step. The second scenario is used to consider drivers' reaction delay, and here T is extended to the previous ten time steps (or 1 s). In the second case, the neural network considers more than a single time step so as to weight the inputs according to their importance to the output. Based on our expectations, the one-second scenario should perform better than

Table 1
Neural network settings.

Model	T (0.1 s)	Prediction	Number of hidden units
NNa(0.1)	1	Acceleration	$3 \times 3 = 9$
NNv(0.1)	1	Velocity	$3 \times 3 = 9$
NNa(1.0)	10	Acceleration	$3 \times 10 \times 3 = 90$
NNv(1.0)	10	Velocity	$3 \times 10 \times 3 = 90$
NNa(2.0)	20	Acceleration	$3 \times 20 \times 3 = 180$
NNv(2.0)	20	Velocity	$3 \times 20 \times 3 = 180$

the 0.1-s case, because it involves more input information. To completely reflect the impact of a driver's reaction delay, various steps T are tested and compared.

In this study, the number of hidden units is based on the number of inputs. Normally, when we increase the number of hidden units, the network capacity for describing the nonlinearity will also improve. In the literature review, almost all neural network models have a relatively smaller number of hidden units. In order to increase the network capacity, we choose the number of hidden units to be three times greater than the number of inputs. For example, NNa(0.1) has three inputs, namely leader speed, gap distance and relevant speed, and the number of hidden units is nine. Moreover, as T increases to 10 steps, the total number of inputs increases to 30. Therefore, NNs with $T = 10$ and $T = 20$ have 90 and 180 hidden units respectively. In the following subsections, these neural network models are tested and compared in detail.

A cross-validation is further conducted to verify the performance of all Neural Network (NN) models. We again randomly select 90% of the training data to train the model and the rest of 10% of the training data to validate the cost. All models are trained on mini-batches of 128, and the average cost of the validation data at training step t is shown in Fig. 4 by using Eq. (4).

$$\overline{C}_t = \frac{\sum_{i=1}^{n} C_{i,t}(W_t, b_t)}{n} \tag{4}$$

where $C_{i,t}(W_t, b_t)$ denotes the cost for validation data i with weights and biases at time step t. The average cost $\overline{C}_t$ at time t is to average all costs in the validation data.

After averaging several runs for each model, it is clear that for all NN model types, the number of historical inputs has its influence on the model performance. Generally, involving more historical inputs increases the accuracy the model predicts. However, the results also indicate that models with T greater than 1.0 s in general converges faster than others but will have a similar cost with whose T equals 1.0 s at the end of training.

4.2.1. Predicting acceleration

The above-mentioned neural network structures NNa(0.1), NNa(1.0) and NNa(2.0) belong to the family of the neural networks that predicting acceleration (Jia et al., 2003; Chong et al., 2011; Khodayari et al., 2012). We select a road segment from the test data set to test the accuracy of these models. The predicted acceleration and velocity (obtained by integrating the predicted acceleration with the initial velocity from the field data) results are shown in Fig. 5.

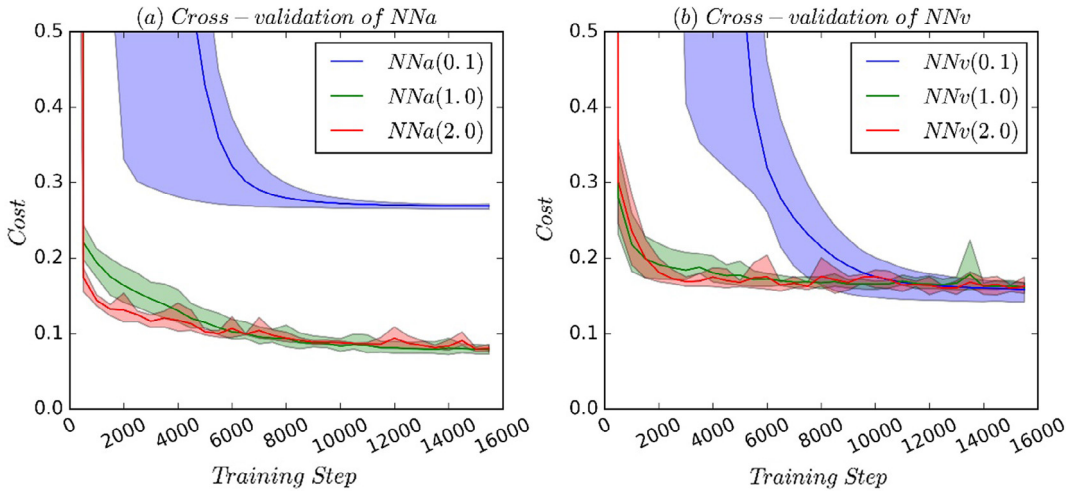

Fig. 4. Cross-validation of NNa and NNv.

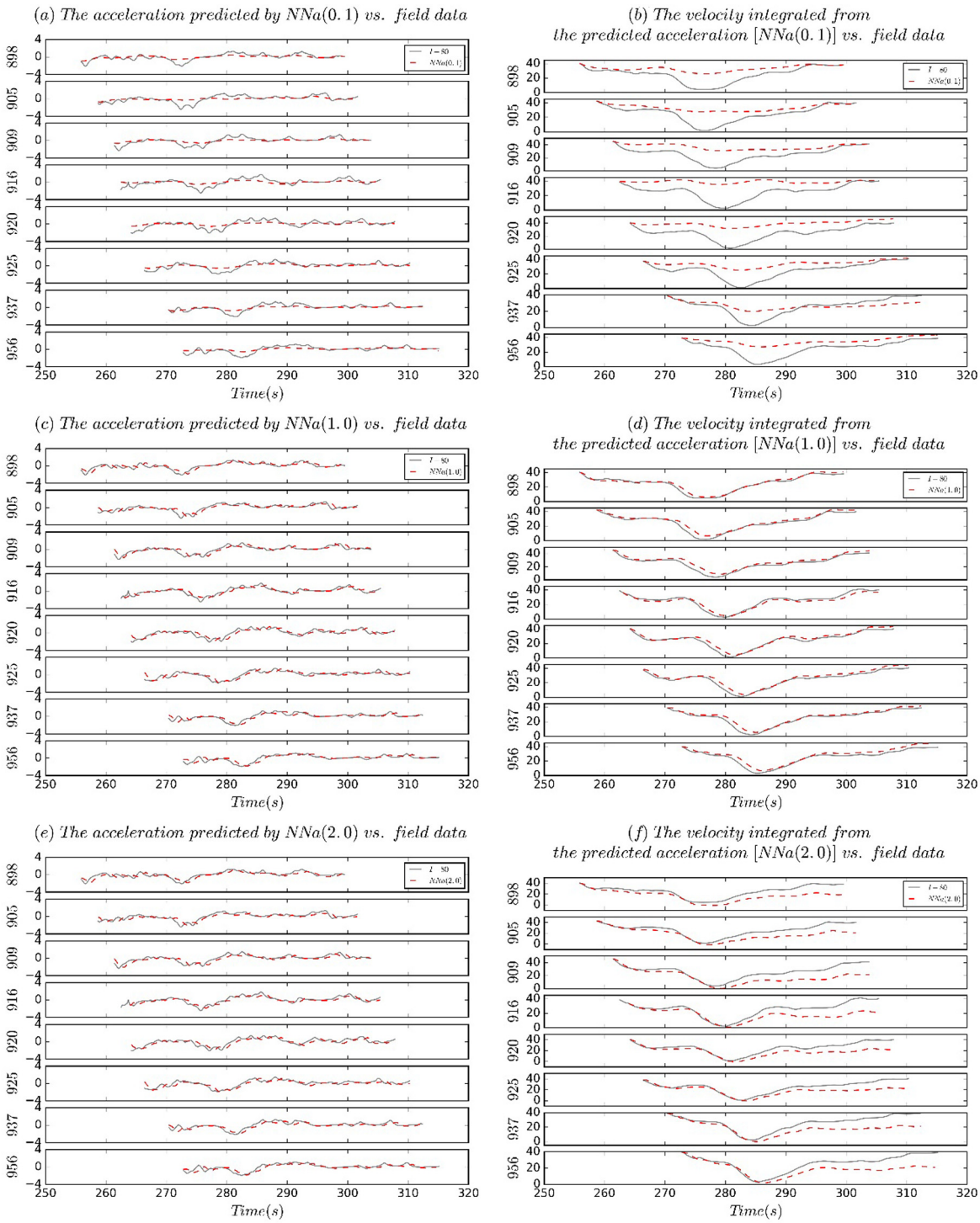

Fig. 5. Comparison of predicted and real accelerations. Note: the number next to y axis represents the ID of a vehicle in NGSIM.

From Fig. 5, it indicates that NNa(0.1) under-fits the real acceleration rate and velocity curve due to the less informative inputs. NNa(1.0) and NNa(2.0) with more historical data inputs, on the other hand, successfully predict the acceleration rate. However, a small difference in predicted acceleration may lead to an unacceptable integrated velocity through time. This accumulative integration error becomes clear while we compare the velocity data. Therefore, training by this method fails to obtain an accurate car-following model.

4.2.2. Predicting velocity

The architecture of neural networks that predicts velocity is another architecture family (Panwai and Dia, 2007; Mathew and Ravishankar, 2012; Aghabayk et al., 2014). Similar to the prediction of acceleration, the predicted velocity and acceleration rate (obtained by differentiating the predicted velocity) are compared in Fig. 6. Because the inputs already contain a

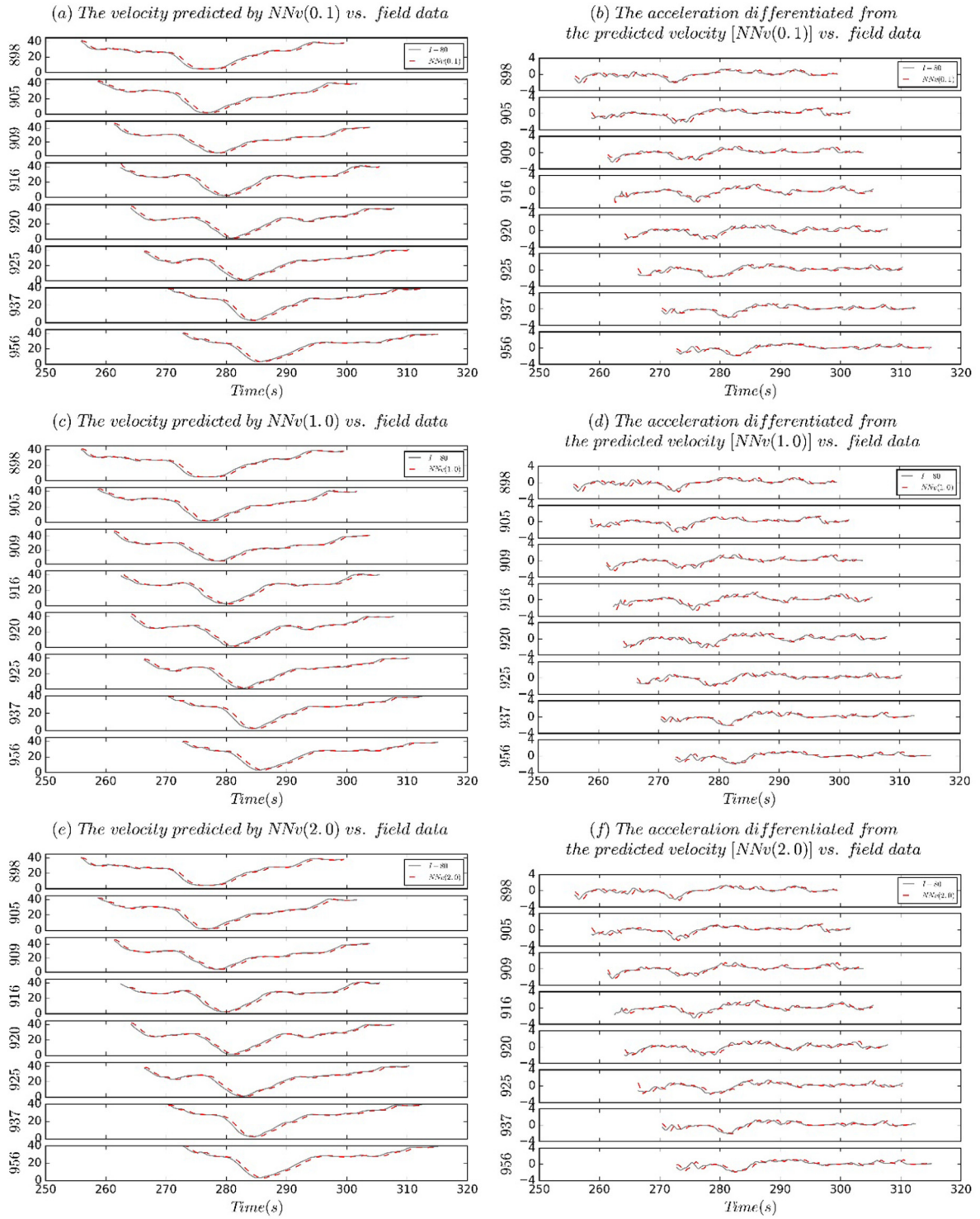

Fig. 6. Comparison of predicted and real velocities. Note: the number next to y axis represents the ID of a vehicle in NGSIM.

velocity factor, which makes them more informative for predicting velocity, the accuracy of the velocity prediction is higher than that of the pure acceleration rate. Based upon the real input data, all types of NNv output very accurate results. To some extent, the number of time steps T does not greatly affect this case.

4.2.3. Shortcomings of existing neural network based model types

The results from NNa(1.0), NNa(2.0), NNv(0.1), NNv(1.0) and NNv(2.0) reflect a good predictive accuracy. However, the model performance under oscillated traffic flow has not been examined. The traffic oscillations have more uncertainty and worth analysing. A model consuming fewer inputs has advantages on increasing calculating speed and reducing data dependency. Compared with models with $T = 20$ and $T = 10$, they both lead to a similar validation result (Fig. 4). We, therefore, conducted simulations to evaluate the performance of NNa and NNv with $T = 10$ under traffic oscillation and the results are illustrated in Fig. 7.

In this simulation, the NN models predict the first (velocity) and second (acceleration) order finite differences of positional data. We then obtain the predicted trajectories by the time integration of velocity and acceleration. Additionally, there are generally two approaches to simulate trajectories which can be defined as the types of leader the follower depending on. The first approach is to assume leaders' trajectories are always given and we need to generate their followers' trajectories by the model. The other approach is to fix only the first leader and all followers' initial/boundary conditions by the data, and calculate trajectories of a group of subsequent vehicles by the model (Treiber and Kesting, 2013b). In this section, we choose the first approach to run the simulation. Eight followers' trajectories are simulated with respect to their initial trajectory points and their data-driven leader. The result in Fig. 7 indicates that both models fail to predict a continuous trajectory.

Knowing this, the comparisons between the predicted acceleration rate and the real acceleration rate, and between the predicted velocity and the real velocity, become less significant. This result also indicates the rest of tests and evaluations in this paper should be done in the trajectory level to precisely reflect real car-following scenarios.

To investigate deeper, all current outputs from neural networks have an effect on the following inputs when testing at the trajectory level. Therefore, the error might accumulate. Hence, a potential solution is to break down the correlation between inputs and outputs, in order to break down the error-accumulating chain. This approach requires a reformation of the neural network architecture, which we will discuss in the next subsection.

4.3. A gap-based neural network model

The gap-based car-following models is another option for predicting following behaviours but at a trajectory level. The previous analyses from (Helly, 1961; Kometani and Sasaki, 1961; Newell, 2002) provide a theoretical foundation for a gap-based model. Based on this idea, we developed a gap-based neural network model and evaluated its performance.

4.3.1. Architecture

As we discussed above, in order to make a better prediction at the trajectory level, it is necessary to break down the link between inputs and outputs. The outputs are utilised by followers to change the following condition. Therefore, the inputs have to depend purely on the leader vehicle's condition, because the decisions made by the leader will not relate (or only a little) to the decision made by the follower. Thus, from the perspective of the trajectory data and the leader's behaviours, the neural network input we select is the leader's displacements (LD_t shown in Fig. 8). From the perspective of the follower's

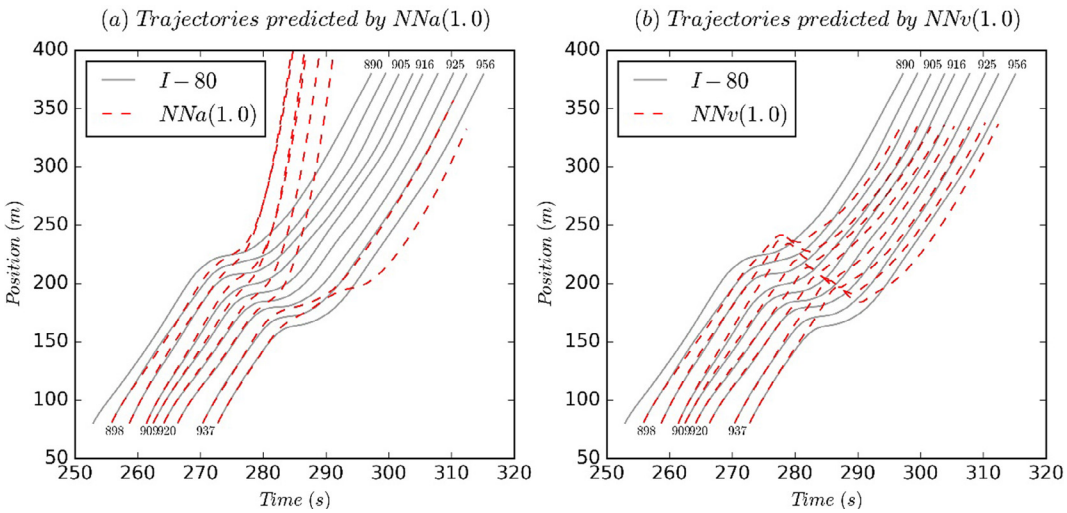

Fig. 7. Comparison of the trajectories generated by neural network models and real data.

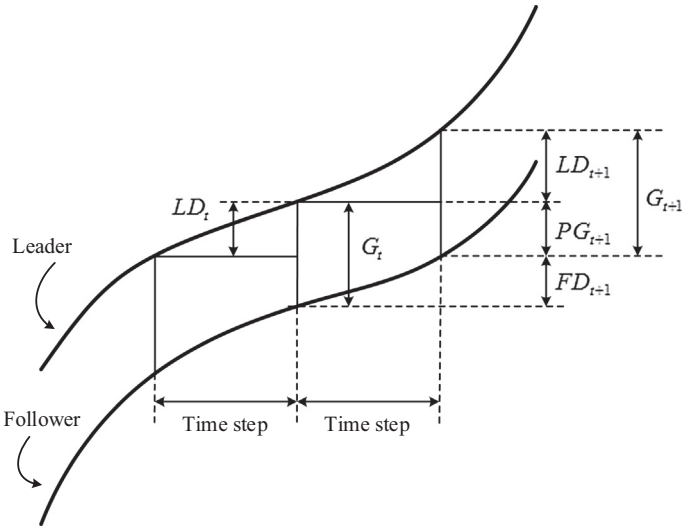

Fig. 8. A typical car-following trajectory.

Table 2
NNpg settings.

Model	T (0.1 s)	Prediction	Number of hidden units
NNpg(0.1)	1	Partial gap	$3 \times 1 = 3$
NNpg(1.0)	10	Partial gap	$3 \times 1 \times 10 = 30$
NNpg(2.0)	20	Partial gap	$3 \times 1 \times 20 = 60$

behaviours, the neural network output we choose is the follower's desired partial gap (PG_{t+1} shown in Fig. 8). Therefore, the input and output are not associated with each other. Moreover, the follower's velocity and acceleration rate can be calculated from the follower's displacement (FD_{t+1}) which is related to PG_{t+1} and G_t.

We name this new type of neural network NNpg, referring to a neural network used to predict the desired partial gap distance (PG_{t+1}). G_t and LD_t denote the gap distance and leader displacement at time t; LD_{t+1} and FD_{t+1} represent the leader displacement and follower displacement at time $t + 1$.

Similar to all NNa and NNv models, we define the network settings for NNpg in Table 2. The number of hidden units are shrunk to 1/3 of the number in NNa and NNv due to the reduction of the input dimension.

4.3.2. Results, applicability and critiques

The cross-validation of NNpg in Fig. 9 shows a similar pattern with other network types. Moreover, taking only one previous data as an input is not sufficient to predict accurate partial gaps. While NNpg(1.0) and NNpg(2.0) both outperform NNpg(0.1) and converge to a similar cost (around 70 calculated by Eq. (4)) but with different converging speeds.

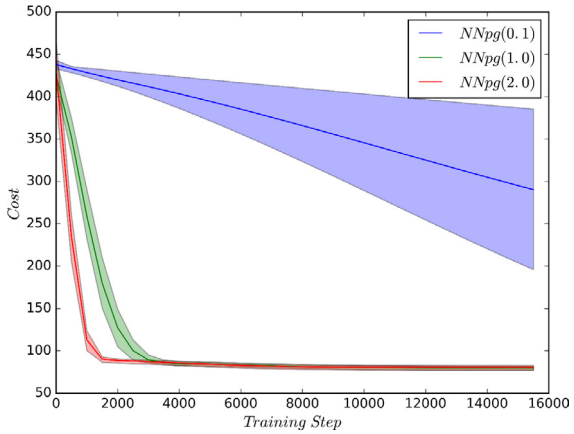

Fig. 9. Cross-validation of NNpg model.

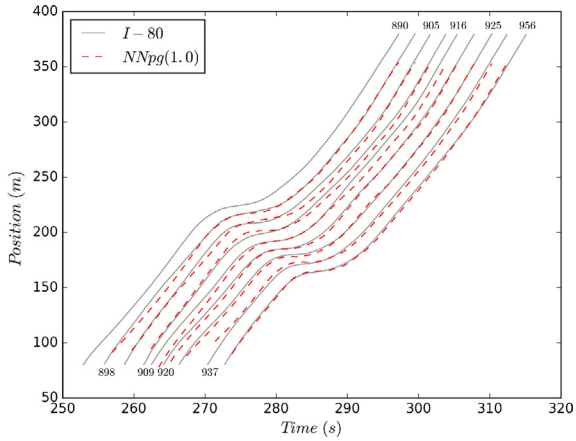

Fig. 10. Trajectory comparison between the real trajectories and the trajectories predicted by NNpg(1.0).

Similar to the reason of selecting NNa(1.0) and NNv(1.0) for the test at the trajectory level, we choose the NNpg(1.0) model for this test and the following summarise its settings:

- Input type: Leader's displacement (*LD*);
- Input time steps (*T*): Last one second (10 steps);
- Output: Desired partial gap to maintain (*PG*); and
- Number of hidden units: 30.

The trajectory results generated by NNpg(1.0) are shown in Fig. 10. The simulation follows the same approach as it in Fig. 7 and the predicted oscillation phase outperforms previous models. However, due to the less informative inputs (only consist of the leader's positional data) compared to NNa and NNv which have three types of inputs, the NNpg(1.0) can only predict a recommendation of PG_{t+1} without a follower's driving condition. In reality, the predicted driving behaviour is not only conditional to leader's behaviour, but also conditional to follower's. Additionally, in a control problem, this model is not fully functional and the partial gap is not capable of providing with a driving motion such as acceleration. Alternatively, the partial gap is a raw information about controlling vehicles, but an appropriate and functional control mechanism would be acceleration-based. Hence, a valid car-following model should take these into consideration.

To solve the problem of uninformative inputs and focus on an appropriate control mechanism, we attempt to apply a variation of neural network type, as described in the next section.

5. Recurrent neural network based model

Through above simulations and tests, we find:

- A sequence of historical driving behaviours can improve NN models' performance;
- Predicting and comparing at the trajectory level improves NN models' performance; and
- A valid car-following model has to consider both the leader's and the follower's driving condition.

Through the above, we propose a recurrent neural network (RNN) based car-following model to satisfy those requirements. Namely, the RNN's built-in mechanism for a sequential historical data as inputs is to eliminate the effort for tracking the previous data points and these inputs can contain both the leader's and the follower's information. Further, we set a trajectory- level objective function to capture a driver's behaviour.

5.1. Recurrent neural networks (RNNs)

The RNN has been applied in many fields, such as handwriting recognition (Graves et al., 2009) and speech recognition (Sak et al., 2014). The RNN has an internal state that represents the current situation. RNNs can build their internal memory as the foundation for the next prediction. A typical RNN's architecture can be shown as in Fig. 11. It is clear that RNNs take a sequence of inputs to generate another sequence of outputs. All inputs and outputs are arranged in order. Therefore, RNNs learn the hidden sequence order as well as the corresponding output value.

To achieve this RNN architecture, the equations for computing the output are also upgraded as follows:

$$h_t = \mathrm{ReLU}(W_h h_{t-1} + W_i I_t + b_i) \tag{5}$$

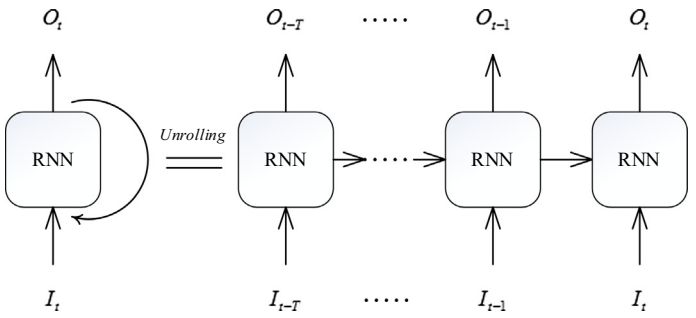

Fig. 11. A typical RNN architecture, folded RNN (left) and unfolded RNN (right).

$$O_t = W_o h_t + b_o \tag{6}$$

where W_i and b_i denote the input weights and biases, h_t is the hidden state that represents the internal memory at time t, I_t represents the input to the network at time t, W_h are the weights of the hidden state, W_o and b_o stand for the output weights and biases, and O_t represents the output at time t. The ReLU in Eq. (5) is a nonlinear activation used to calculate h_t.

We expect a model for controlling vehicle, so it is necessary to predict acceleration rate rather than velocity or partial gap. Therefore, a RNNa model is proposed under this concern. Note that the RNN model predicts on a sequence of data and the sequence length can vary, so the constant sequence length T used in NNa, NNv and NNpg is not needed.

The objective function or the cost function performing at the trajectory level is defined as follows:

$$a_{t+1} = RNNa(g_t, \Delta v_t, v_t) \tag{7}$$

$$\begin{aligned} x_{t+1} &= v_t \Delta t + (1/2) a_{t+1} \Delta t^2 \\ v_{t+1} &= v_t + a_{t+1} \Delta t \end{aligned} \tag{8}$$

$$C(W, b) = \frac{\left((x^l_{t+1} - x_{t+1}) - g_{t+1}\right)^2}{\left(g_{t+1}\right)^2} \tag{9}$$

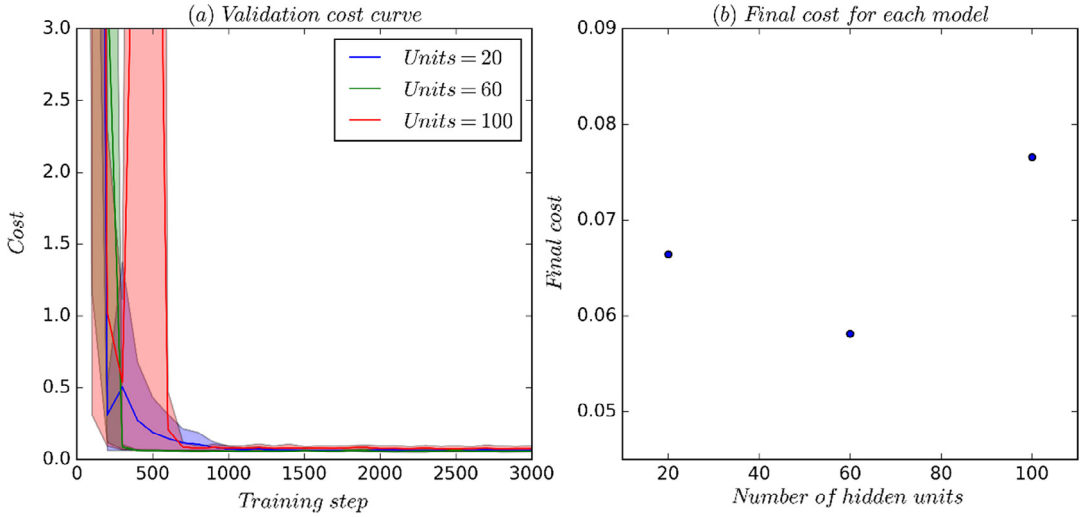

Fig. 12. Cross-validation for selecting the number of hidden units in RNNa.

Table 3
Calibrated parameters of the IDM and OVM.

IDM	a	v_0	s_0	T	b
Calibrated value	2.01	27.19	6.73	1.53	1.77
OVM	p_0	p_1	p_2	p_3	p_4
Calibrated value	0.57	7.42	8.26	0.129	2.30

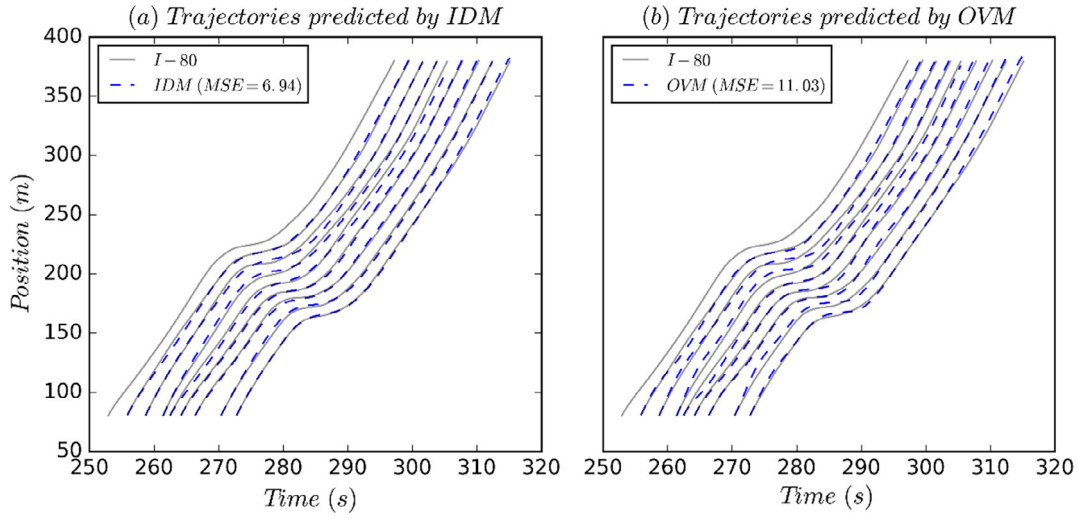

Fig. 13. Trajectories predicted by IDM and OVM.

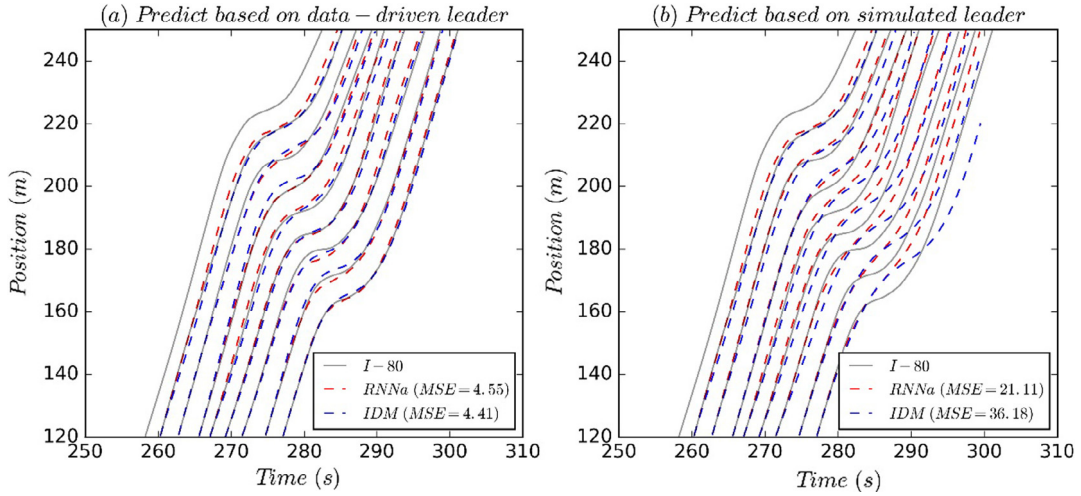

Fig. 14. Trajectories simulated with two different approaches.

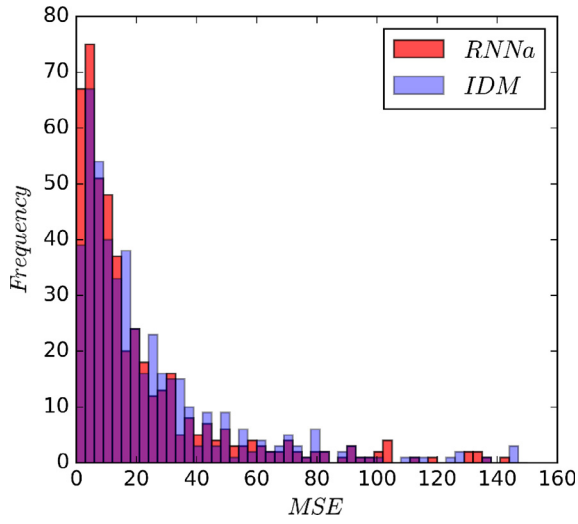

Fig. 15. Histogram of trajectories MSE for the IDM and RNNa models.

The RNNa model takes inputs including gap (g_t), relevant speed (Δv_t) and vehicle speed (v_t) at time step t and output acceleration (a_{t+1}) for the next time step. Then, the follower's position (x_{t+1}) and velocity (v_{t+1}) for the next step are updated based on a_{t+1} and the time interval Δt which equals 0.1 in here. Based on the aforementioned discussion, we perform a comparison of the predicted gap with the gap in data. The predicted gap is calculated by subtracting the predicted follower's position (x_{t+1}) from the data-driven leader's position (x^l_{t+1}). The gap difference is squared then divided by the squared gap in data $(g_{t+1})^2$ to reduce the gap sensitivity (Kesting and Treiber, 2008). The $C(W, b)$ denotes the cost calculated based on current weights and biases in the RNNa model. The RNNa gradually minimises this cost by backpropagating a small update through time in the direction of optimising the weights and biases.

5.2. Results and performance comparison

We run a cross-validation about the average cost with different validating data for selecting an appropriate number of hidden units in RNNa's cell. From Fig. 12, all three models with different unit numbers tended to converge to a cost near 0.07. Noted that the cost jump occurred at the 500th step on the curve of units = 100 was caused by mini-batch training as data in batches were different. The cost result indicates that the number of hidden units in the range of (20, 100) does not greatly affect the performance of the RNNa model. We choose the model with 60 units for the rest of tests as it performs slightly better than other models shown in Fig. 12 (b).

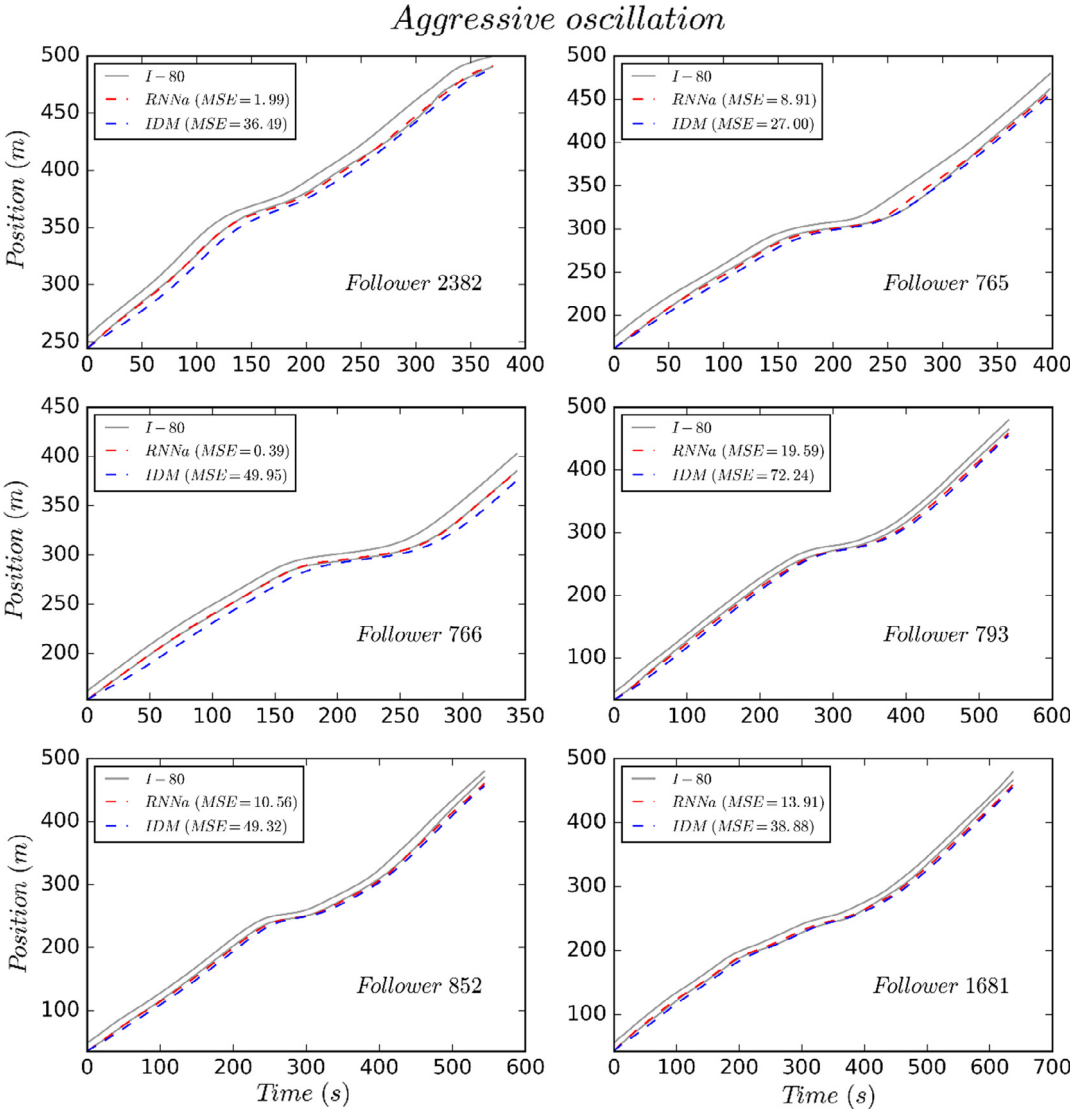

Fig. 16. Predict aggressive drivers in a traffic oscillation.

The following settings summarise the configuration for the selected RNNa:

- Input type: gap (g_t), relevant speed (Δv_t), follower's velocity (v_t);
- Number of units in cell: 60; and
- Output: Follower's acceleration rate (a_{t+1}).

Since the RNNa model has different property to classical car-following models, a comparison of RNNa with classical models is conducted. Firstly, we calibrate two of the most popular classical models, IDM and OVM, by Global Least-Squared Errors Calibration (Treiber and Kesting, 2013a) which is calibrated at a trajectory level to reduce the impact of the accumulative error. The model formulas are list as following and calibrated model parameters can be found in Table 3.

$$a_{IDM} = a\left[1 - \left(\frac{v}{v_0}\right)^4 - \left(\frac{s_0 + vT + \frac{v\Delta v}{2\sqrt{ab}}}{s}\right)\right] \tag{10}$$

$$a_{OVM} = p_0((p_1 + p_2 tanh(p_3 s - p_4)) - v) \tag{11}$$

In Eqs. (10) and (11), the v, Δv and s represent the follower's velocity, relevant speed and gap distance respectively. Others are the unknown parameters to be calibrated.

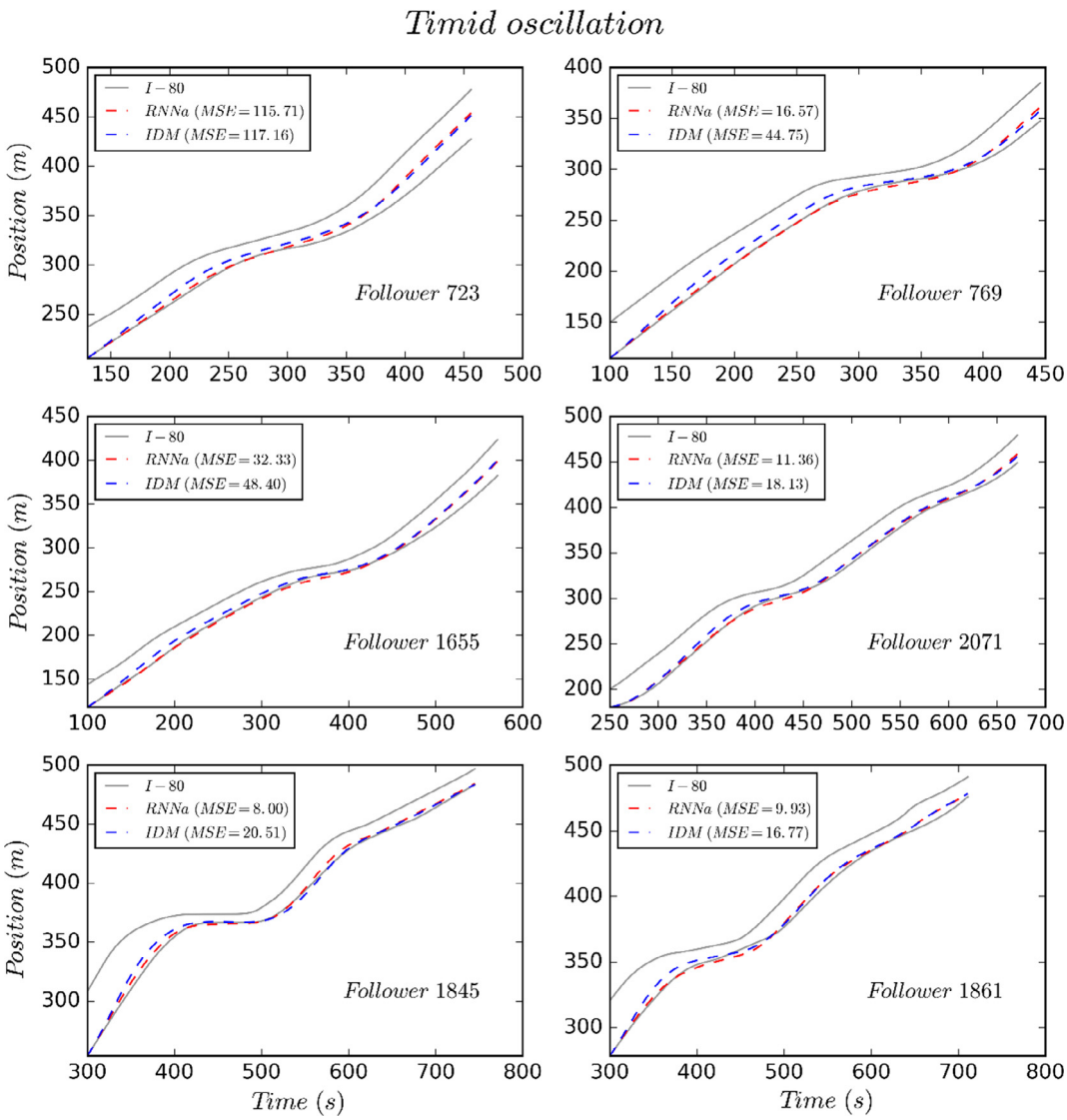

Fig. 17. Predict timid drivers in a traffic oscillation.

The calibrated IDM and OVM are also compared in above-mentioned oscillation scenarios and their accuracy is quantified by calculating the Mean Squared Error (MSE) between data and predicted trajectory, as used in many other studies (Qu et al., 2015, 2017; Xu et al., 2016; Liu et al., 2017). The MSE is defined as below

$$MSE = \frac{\sum_{i=1}^{n}(\bar{x}_i - x_i)^2}{n} \tag{12}$$

where $\bar{x}_i$ and x_i denote the ith predicted and field position respectively. Again, every follower's trajectory is simulated based on its data-driven leader and the result is shown in Fig. 13.

The IDM (MSE = 6.94) has more accurate predictions compared with OVM (MSE = 11.03) in the traffic oscillation. Therefore, we choose the IDM as a reference to compare with the RNNa model.

For the subsequent tests, we look into the oscillation segment and run simulations based on different simulating approaches. The first approach is the same as it in above simulations which predicts a follower's trajectory based on data-driven leader and follower's initial state from data. The second approach is to simulate the trajectories of a group of subsequent vehicles based on one fixed first vehicle's trajectory and followers' initial state. The results of these two approaches are shown in Fig. 14.

As illustrated in Fig. 14(a), both RNNa and IDM fit field data very well with the MSE less than five when predicting by the first approach. However, Fig. 14(b) indicates that the MSE of RNNa model is 58.3% of IDM's, which means RNNa has a better

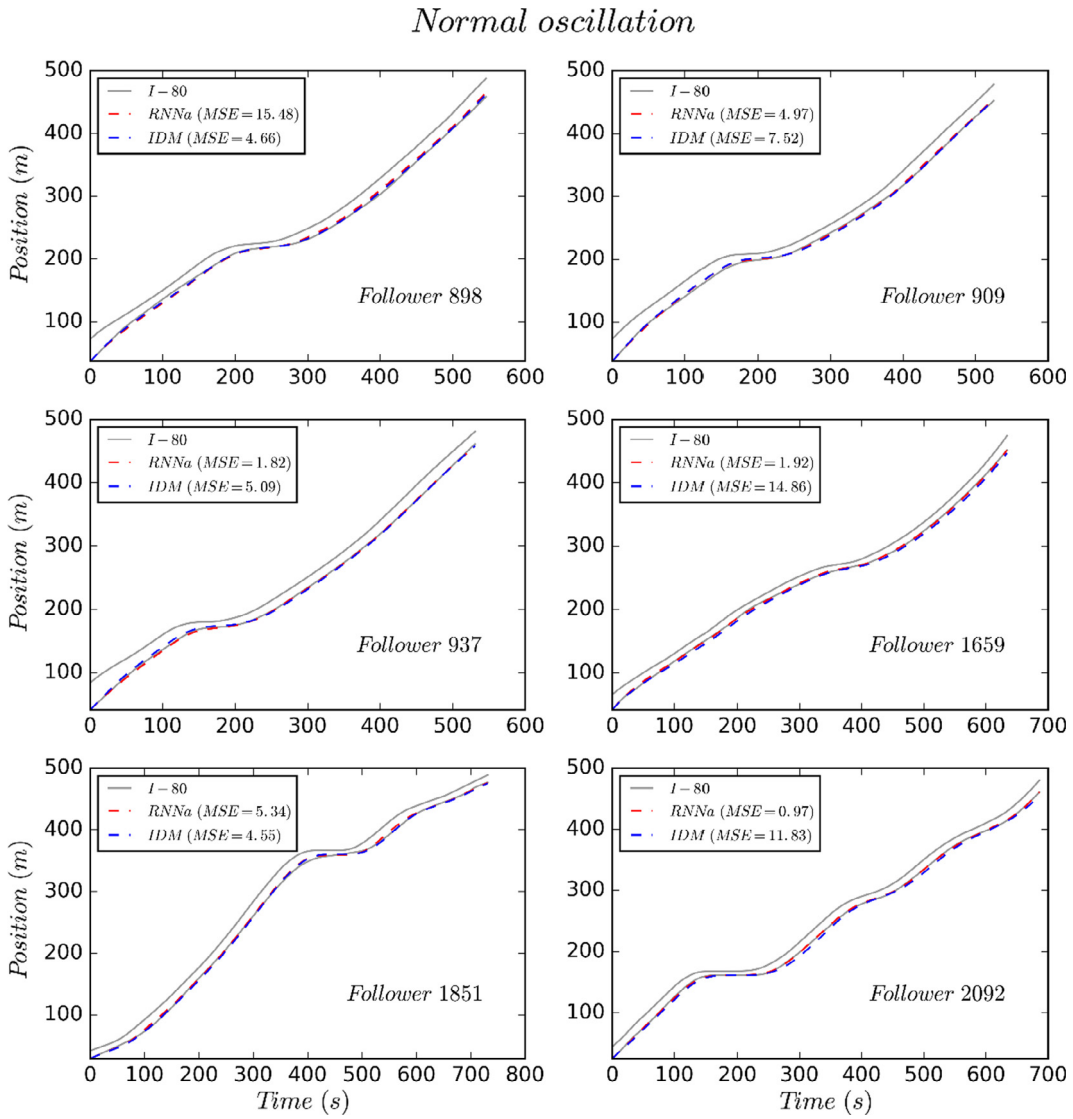

Fig. 18. Predict normal drivers in a traffic oscillation.

prediction effect to a group of vehicles. In order to dig the reason behind, we conduct more detailed simulations to find the difference.

Firstly, we use test data sets to test two models by calculating their MSEs on every follower's trajectory and show in a histogram (Fig. 15) to summarise models performance. As we can see, the RNNa model predicts trajectories with MSEs less than five more frequently. However, at a global level, these two models yield a similar MSE. The average MSE for all trajectories are very similar (29.08 for IDM vs 27.25 for RNNa). We then decide to examine the models' performances with respect to different driving behaviours.

As mentioned in the literature review, drivers' characteristic varies. Drivers, therefore, can be divided into three different groups including "aggressive", "timid" and "normal" (Chen et al., 2012). In the test data set, we distinguish trajectories into these three different characteristic tags and pick representative ones to plot in Figs. 16–18.

All these results indicate a special pattern of the RNNa model, which achieves a better fitness when a driver's initial motivation is given. To distinguish drivers' characteristics, the RNNa model can find the hidden information in a follower's initial data point, while most classical models (except Chen et al., 2012) cannot do so because they treat every driver as the same, regardless of initial data points. This can also be found in Fig. 19 which are tested under normal conditions. The RNNa model gathers the initial follower's information from data and analyses it in order to understand which type of driver he or she is. The follower 2974 in Fig. 19 can be defined as a timid driver because it attempts to remain a larger gap compared with the gap predicted by the IDM. Another example is the trajectory of follower 3233, with an "aggressive" characteristic being

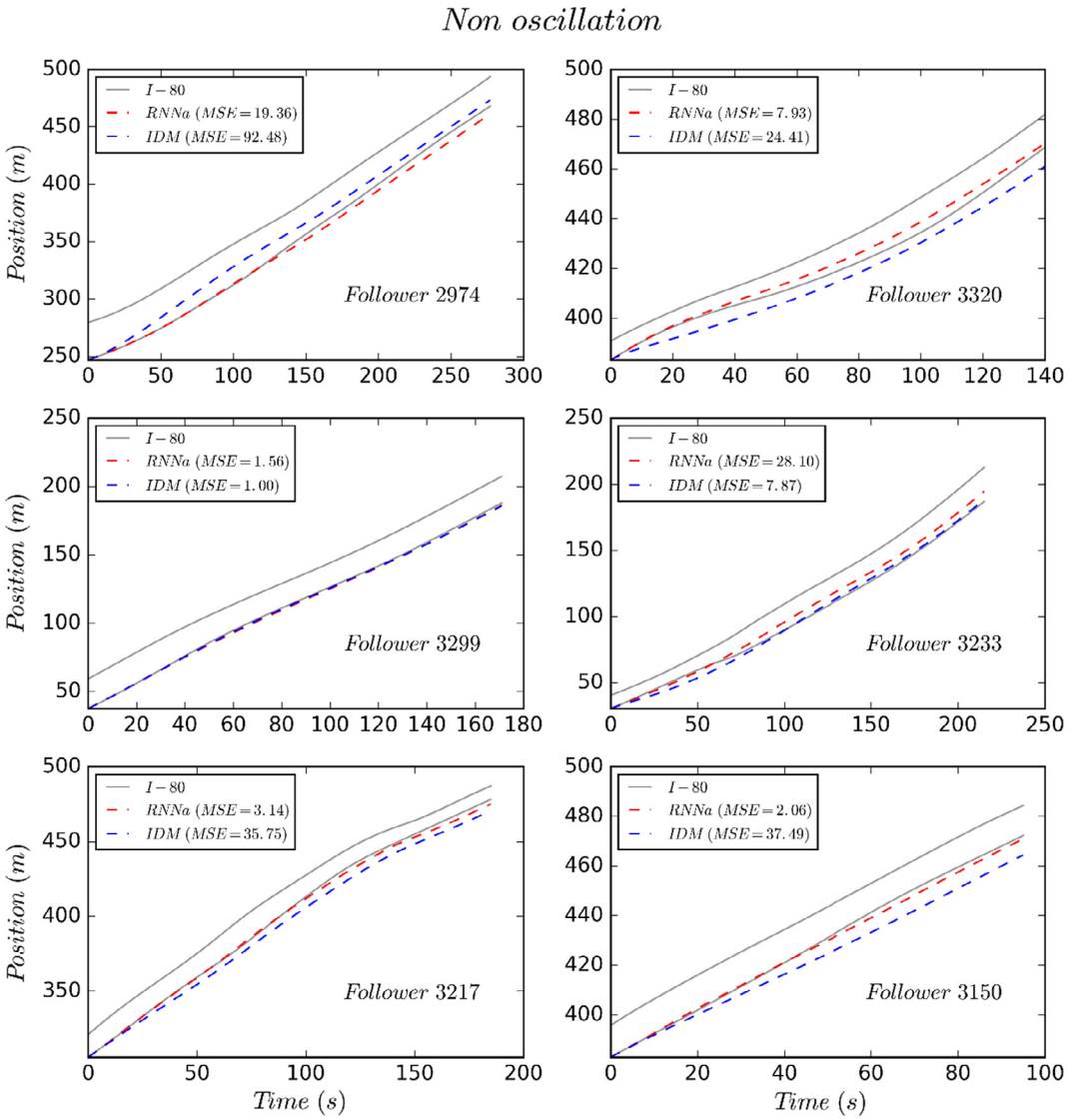

Fig. 19. Prediction in non-oscillation cases.

identified at the initial point, the RNNa model attempts to predict an "aggressive" trajectory for this driver. However, after the time point 60, this driver tends to drive "normally" and RNNa can no longer fit to this "normal" trajectory. It indirectly proves that the RNNa model weights an initial given information more than what a classical model does and, in many scenarios, this initial information is a key for RNNa to understand the characteristic of a driver.

To summarise, we have the following findings:

- RNNa has a comparable performance with classical models in predicting the trajectory of the immediate subsequent vehicle;
- RNNa has a stronger performance than classical models in predicting the trajectories of subsequent eight vehicles;
- RNNa has a much stronger performance than classical models in predicting aggressive oscillations (oscillations caused by aggressive drivers);
- RNNa has a comparable performance with classical models in predicting timid and normal oscillations.

6. Conclusions

In this research, we adopt neural networks instead of classical car-following models to simulate traffic oscillations. We investigate the existing neural network based car-following models in congested traffic flow, and the results show that none of them is able to accurately predict drivers' behaviours under traffic oscillations. We further identify potential reasons which are caused by insufficient inputs and inappropriate objective function for these neural network models. In order to overcome this deficiency, we apply a new type of neural network and design its architecture at the trajectory level (RNNa). The RNNa model gathers a sequence of historical data to approximate an acceleration rate and integrates it to positional data. We further use trajectories to train RNNa.

We test the RNNa model under both oscillated and non-oscillated traffic flow. The results indicate that RNNa has a stronger performance in predicting the trajectories of a group of subsequent vehicles given the trajectory of the first vehicle and initial/boundary conditions for following vehicles, while it has comparable performance with IDM model in predicting trajectory of the immediate subsequent vehicle. Considering the fact that the same calibration parameters are used for all vehicles in a IDM model, NN-based models has advantages in identifying and differentiating different vehicles. We thus further compare the performances of RNNa and IDM in predicting the behaviours of different types of drivers. According to our results, RNNa significantly outperforms IDM in predicting oscillations caused by aggressive drivers. In this regard, the proposed RNNa model can be used to complement classical car following models in capturing traffic oscillations.

Acknowledgement

This research is jointly supported by Endeavour Cheung Kong Research Fellowship and the US National Science Foundation CMMI #1453949.

References

Aghabayk, K., Sarvi, M., Forouzideh, N., Young, W., 2014. Modelling heavy vehicle car-following behaviour in congested traffic conditions. J. Adv. Transport. 48, 1017–1029.
Bando, M., Hasebe, K., Nakanishi, K., Nakayama, A., 1998. Analysis of optimal velocity model with explicit delay. Phys. Rev. E 58, 5429–5435.
Bando, M., Hasebe, K., Nakayama, A., Shibata, A., Sugiyama, Y., 1995. Dynamical model of traffic congestion and numerical simulation. Phys. Rev. E 51, 1035–1042.
Bilbao-Ubillos, J., 2008. The costs of urban congestion: estimation of welfare losses arising from congestion on cross-town link roads. Transport. Res. Part A: Policy Pract. 42, 1098–1108.
Brackstone, M., McDonald, M., 1999. Car-following: a historical review. Transport. Res. Part F: Traff. Psychol. Behav. 2, 181–196.
Chandler, R.E., Herman, R., Montroll, E.W., 1958. Traffic dynamics: studies in car following. Operat. Res. 6, 165–184.
Chen, D., Ahn, S., Laval, J., Zheng, Z., 2014. On the periodicity of traffic oscillations and capacity drop: the role of driver characteristics. Transport. Res. Part B: Methodol. 59, 117–136.
Chen, D., Laval, J., Zheng, Z., Ahn, S., 2012. A behavioral car-following model that captures traffic oscillations. Transport. Res. Part B: Methodol. 46, 744–761.
Chong, L., Abbas, M.M., Medina, A., 2011. Simulation of driver behavior with agent-based back-propagation neural network. Transport. Res. Rec.: J. Transport. Res. Board 2249, 44–51.
Chong, L., Abbas, M.M., Medina Flintsch, A., Higgs, B., 2013. A rule-based neural network approach to model driver naturalistic behavior in traffic. Transport. Res. Part C: Emerg. Technol. 32, 207–223.
FHWA. 2008. *The Next Generation Simulation (NGSIM)* [Online]. Available: <http://www.ngsim.fhwa.dot.gov/> (Accessed).
Gipps, P.G., 1981. A behavioural car-following model for computer simulation. Transport. Res. Part B: Methodol. 15, 105–111.
Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J., 2009. A novel connectionist system for unconstrained handwriting recognition. IEEE Trans. Pattern Anal. Mach. Intell. 31, 855–868.
He, Z., Zheng, L., Guan, W., 2015. A simple nonparametric car-following model driven by field data. Transport. Res. Part B: Methodol. 80, 185–201.
Helly, W., 1961. Simulation of Bottlenecks in Single-Lane Traffic Flow. Elsevier Publishing Company.
Herman, R., Gazis, D.C., Potts, R.B., 1959. Car-following theory of steady-state traffic flow. Operat. Res. 7, 499–505.
Jia, H., Juan, Z., NI, A., 2003. Develop a car-following model using data collected by "five-wheel system". In: IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, pp. 346–351.
Jiang, R., Wu, Q., Zhu, Z., 2001. Full velocity difference model for a car-following theory. Phys. Rev. E 64, 017101.
Kerner, B., 1999. Congested traffic flow: observations and theory. Transport. Res. Rec.: J. Transport. Res. Board 1678, 160–167.
Kerner, B.S., 2012. The Physics of Traffic: Empirical Freeway Pattern Features, Engineering Applications, and Theory. Springer.
Kesting, A., Treiber, M., 2008. Calibrating car-following models by using trajectory data: Methodological study. Transport. Res. Rec.: J. Transport. Res. Board 2088, 148–156.

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. Part A Syst. Hum. 42, 1440–1449.
Kometani, E., Sasaki, T., 1961. Dynamic Behaviour of Traffic With a Non-Linear Spacing-Speed Relationship. Elsevier publishing co, Amsterdam.
Laval, J.A., Leclercq, L., 2010. A mechanism to describe the formation and propagation of stop-and-go waves in congested freeway traffic. Philos. Trans. Roy. Soc. Lond. A: Math., Phys. Eng. Sci. 368, 4519–4541.
Laval, J.A., Toth, C.S., Zhou, Y., 2014. A parsimonious model for the formation of oscillations in car-following models. Transport. Res. Part B: Methodol. 70, 228–238.
Li, X., Cui, J., An, S., Parsafard, M., 2014. Stop-and-go traffic analysis: theoretical properties, environmental impacts and oscillation mitigation. Transport. Res. Part B: Methodol. 70, 319–339.
Li, X., Ouyang, Y., 2011. Characterization of traffic oscillation propagation under nonlinear car-following laws. Transport. Res. Part B: Methodol. 45, 1346–1361.
Li, X., Peng, F., Ouyang, Y., 2010. Measurement and estimation of traffic oscillation properties. Transport. Res. Part B: Methodol. 44, 1–14.
Li, X., Wang, X., Ouyang, Y., 2012. Prediction and field validation of traffic oscillation propagation under nonlinear car-following laws. Transport. Res. Part B: Methodol. 46, 409–423.
Lipton, Z.C., Berkowitz, J., Elkan, C., 2015. A critical review of recurrent neural networks for sequence learning. ArXiv e-prints [Online], 1506. Available: <http://adsabs.harvard.edu/abs/2015arXiv150600019L> (Accessed May 1, 2015).
Liu, Z., Wang, S., Zhou, B., Cheng, Q., 2017. Robust optimization of distance-based tolls in a network considering stochastic day to day dynamics. Transport. Res. Part C 79, 58–72.
Manstetten, D., Krautter, W., Schwab, T., 1997. Traffic simulation supporting urban control system development. In: 4th World Congress on Intelligent Transport Systems, Berlin, Germany, pp. 1–8.
Mathew, T.V., Ravishankar, K.V.R., 2012. Neural network based vehicle-following model for mixed traffic conditions. Eur. Transp. – Trasp. Europei, 1–15.
Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., Khudanpur, S., 2010. Recurrent neural network based language model. Interspeech, p. 3.
Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814.
Newell, G.F., 2002. A simplified car-following theory: a lower order model. Transport. Res. Part B: Methodol. 36, 195–205.
Olson, R.L., Hanowski, R.J., Hickman, J.S., Bocanegra, J.L., 2009. Driver Distraction In Commercial Vehicle Operations.
Panwai, S., Dia, H., 2007. Neural agent car-following models. IEEE Trans. Intell. Transp. Syst. 8, 60–70.
Papathanasopoulou, V., Antoniou, C., 2015. Towards data-driven car-following models. Transport. Res. Part C: Emerg. Technol. 55, 496–509.
Qu, X., Wang, S., Zhang, J., 2015. On the fundamental diagram for freeway traffic: a novel calibration approach for single-regime models. Transport. Res. Part B 73, 91–102.
Qu, X., Zhang, J., Wang, S., 2017. On the stochastic fundamental diagram for freeway traffic: model development, analytical properties, validation, and extensive applications. Transport. Res. Part B 104, 256–271.
Rhoades, C., Wang, X., Ouyang, Y., 2016. Calibration of nonlinear car-following laws for traffic oscillation prediction. Transport. Res. Part C: Emerg. Technol. 69, 328–342.
Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1988. Learning representations by back-propagating errors. Cognit. Model. 5, 1.
Saifuzzaman, M., Zheng, Z., 2014. Incorporating human-factors in car-following models: a review of recent developments and research needs. Transport. Res. Part C: Emerg. Technol. 48, 379–403.
Sak, H., Senior, A.W., Beaufays, F., 2014. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. Interspeech, pp. 338–342.
Santana, E., Hotz, G., 2016. Learning a Driving Simulator. ArXiv e-prints [Online], 1608. Available: <http://adsabs.harvard.edu/abs/2016arXiv160801230S> (Accessed August 1, 2016).
Song, G., Yu, L., Xu, L., 2013. Comparative analysis of car-following models for emissions estimation. Transport. Res. Rec.: J. Transport. Res. Board 2341, 12–22.
Thiemann, C., Treiber, M., Kesting, A., 2008. Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data. Transport. Res. Rec.: J. Transport. Res. Board 2088, 90–101.
Tian, J., Jiang, R., Jia, B., Gao, Z., Ma, S., 2016. Empirical analysis and simulation of the concave growth pattern of traffic oscillations. Transport. Res. Part B: Methodol. 93, 338–354.
Treiber, M., Helbing, D., 2003. Memory effects in microscopic traffic models and wide scattering in flow-density data. Phys. Rev. E 68, 046119.
Treiber, M., Hennecke, A., Helbing, D., 2000. Congested traffic states in empirical observations and microscopic simulations. Phys. Rev. E – Stat. Phys., Plasmas, Fluids, Relat. Interdiscip. Top. 62, 1805–1824.
Treiber, M., Kesting, A., 2013a. Microscopic calibration and validation of car-following models – a systematic approach. Proc. – Soc. Behav. Sci. 80, 922–939.
Treiber, M., Kesting, A., eiber and Kesting, 2013 b. Traffic Flow Dynamics: Data, Models and Simulation. Springer-Verlag, Berlin Heidelberg.
Treiber, M., Kesting, A., Helbing, D., 2010. Three-phase traffic theory and two-phase models with a fundamental diagram in the light of empirical stylized facts. Transport. Res. Part B: Methodol. 44, 983–1000.
Treiber, M., Kesting, A., Schönhof, M., Helbing, D., 2007. Extending adaptive cruise control to adaptive driving strategies. Transport. Res. Rec.: J. Transport. Res. Board 2000, 16–24.
Xu, C., Yang, Y., Jin, S., Qu, Z., Hou, L., 2016. Potential risk and its influencing factors for separated bicycle paths. Accid. Anal. Prevent. 87, 59–67.
Zhao, T., Nie, Y., Zhang, Y., 2014. Extended spectral envelope method for detecting and analyzing traffic oscillations. Transport. Res. Part B: Methodol. 61, 1–16.
Zheng, Z., Ahn, S., Chen, D., Laval, J., 2011a. Applications of wavelet transform for analysis of freeway traffic: bottlenecks, transient traffic, and traffic oscillations. Transport. Res. Part B: Methodol. 45, 372–384.
Zheng, Z., Ahn, S., Chen, D., Laval, J., eng et al., 2011 b. Freeway traffic oscillations: microscopic analysis of formations and propagations using Wavelet Transform. Transport. Res. Part B: Methodol. 45, 1378–1388.
Zheng, Z., Ahn, S., Monsere, C.M., 2010. Impact of traffic oscillations on freeway crash occurrences. Accid. Anal. Prev. 42, 626–636.