A rule-based neural network approach to model driver naturalistic behavior in traffic

Linsen Chong a,⇑, Montasir M. Abbas b,1, Alejandra Medina Flintsch c,3, Bryan Higgs d,2

a Department of Civil and Environmental Engineering, Room 1-245, Massachusetts Institute of Technology, Cambridge, MA 02139, United States
b Charles Via Department of Civil and Environmental Engineering, 301-A, Patton Hall, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, United States
c Virginia Tech Transportation Institute, Virginia Polytechnic Institute and State University, 3500 Transportation Research, Plaza (0536), Blacksburg, VA 24061, United States
d Charles Via Department of Civil and Environmental Engineering, 301-D, Patton Hall, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, United States

A R T I C L E   I N F O
Article history:
Received 15 December 2011
Received in revised form 3 August 2012
Accepted 6 September 2012

Keywords:
Driving behavior
Fuzzy logic
Reinforcement learning
Artificial neural network
Car-following models
Safety critical events
Naturalistic data

A B S T R A C T

This paper proposes a rule-based neural network model to simulate driver behavior in terms of longitudinal and lateral actions in two driving situations, namely car-following situation and safety critical events. A fuzzy rule based neural network is constructed to obtain driver individual driving rules from their vehicle trajectory data. A machine learning method reinforcement learning is used to train the neural network such that the neural network can mimic driving behavior of individual drivers. Vehicle actions by neural network are compared to actions from naturalistic data. Furthermore, this paper applies the proposed method to analyze the heterogeneities of driving behavior from different drivers' data.

Driving data in the two driving situations are extracted from Naturalistic Truck Driving Study and Naturalistic Car Driving Study databases provided by the Virginia Tech Transportation Institute according to pre-defined criteria. Driving actions were recorded in instrumented vehicles that have been equipped with specialized sensing, processing, and recording equipment.

© 2012 Elsevier Ltd. All rights reserved.

1. Introduction

Driver behavior determines vehicle actions in traffic. The task of driving could be different when the surrounding traffic condition is different. Consequently, driver behavior and the resulting vehicle actions should be different. For instance, in congested situations when a driver is not able to drive freely, the driving task should be to follow a leading vehicle. When a driver observes a sudden break from the leading vehicle, the driving task should be to avoid the incoming conflict.

In this research, we focus on modeling two types of driving behavior: car-following behavior and evasive behavior. Car-following behavior usually appears in conditions when a driver is interacting with a leading vehicle before their relative distance becomes too close. Longitudinal action acceleration is considered to be the only action. Evasive behavior appears in safety critical events and vehicles should take evasive actions to avoid upcoming conflicts, especially rear-end collisions. During a safety critical event, both longitudinal and lateral actions are considered.
1.1. Car-following models

In the last 50 years, a large number of car-following models have been proposed to model the process of drivers’ “follow” behavior with leading vehicles (Brackstone and McDonald, 1999). According to literature, relative speed and relative distance between two vehicles are the most common underlying factors that would eventually lead to the construction of car-following models.

Most car-following models fall into two categories: safety distance models and psycho-physical models. Safety distance models assume that the following vehicle is able to stop before becoming too close to its leader. The minimum distance of two vehicles is guaranteed to be greater than a safety distance threshold (Gazis et al., 1961; Wiedemann, 1974; Gipps, 1981; Wiedemann and Reiter, 1992; Fritzsche, 1994; Kesting and Treiber, 2008). Psycho-physical models divide traffic situations into several regimes based on human’s recognition on different traffic patterns where drivers’ tasks and behavior are different (Wiedemann, 1974; Wiedemann and Reiter, 1992; Fritzsche, 1994).

1.2. Car-following model calibration efforts

Car-following models could be calibrated using macroscopic or microscopic data. Macroscopic data are usually obtained from aggregated traffic stream while microscopic data include more information on individual vehicle trajectories. Rakha et al. (2007) purposed a macroscopic calibration method that used loop detector data. Menneni et al. (2008) used microscopic and macroscopic data both in the process of model calibration. Kesting and Treiber (2008) used publicly available microscopic trajectory data to study car-following behavior on individual drivers. Their data were collected from an instrumented car with a radar sensor. Two models: Intelligent Driver Model (IDM) and Velocity Difference Model (VDM) were calibrated using genetic algorithm. Ossen and Hoogendoorn (2004) pointed out that the development of accurate and robust models reply on appropriate microscopic data, especially when analyzing heterogeneous behavior in different individual drivers.

1.3. Modeling safety critical events in traffic

We consider crash and near crash events as safety critical events in this study. To model driver behavior during safety critical events, most researchers modify car-following model and enable crashes to happen. Hamdar and Mahmassani (2008) adjusted several existing car-following models such that congestion dynamics and model accident-prone behaviors could be captured. The revised car-following and lane-changing models were developed under different degrees of relaxation on the safety constraints and implemented in a microscopic simulation framework. Xin et al. (2008) extended the capability of a car-following model (Gipps model) to simulate unsafe driving conditions. When a driver is in a subconscious driving state, an unsafe driving behavior is triggered.

1.4. Drawbacks of model calibration methods

Several drawbacks from previous model calibration methodologies were pointed out. Firstly, errors from data measurement can deteriorate model performance significantly. Ossen and Hoogendoorn (2008) discussed about three findings of a calibrated car-following model (GHR model) performance: (1) measurement errors can yield a considerable bias; (2) parameters that minimizing the objective function do not necessarily capture car-following dynamics best and (3) measurement errors can substantially reduce model sensitivity and reduce reliability. Similarly, Brockfeld et al. (2004) used the same data to calibrate 10 different car-following models. As a result, all models shared the same problem with particular sets of data. They pointed out that no model appears to be significantly better than any other model and models with more parameters did not necessarily provide better results.

Secondly, since different car-following models were developed from different data resources, no model is expected to match all sample data resources well. So if no prior knowledge about data is provided, it is difficult to choose the “best” model.

In the study of modeling driver evasive behavior, no systematic methodologies have been fully developed, probably due to the lack of events data of individual drivers.

1.5. Traffic states and actions

We think driver behavior in traffic is a state-action mapping problem that driver’s actions depend on the traffic situation. Traffic states are defined by a set of variables that can represent vehicle kinematic conditions and its surrounding environment. Relative distance from the leading vehicle, relative distance and the acceleration of leading vehicle have been used as state variables in existing car-following models.

During safety critical events, driver actions are more complicated and not limited to longitudinal actions only. For instance, a driver may take a maneuver and execute a lane change simultaneously. Developing lateral action models are necessary.
1.6. State and action partitioning

We believe that as a type of human decision process, driver behavior is a realization of driving rules that closely related to traffic states. Driving rules provide the mapping policy from states to actions. In order to construct our state-action mapping rules, fuzzy logic is used to partition the traffic state variables. In our purposed methodology, fuzzy logic partition sets and fuzzy driving rules are embedded in a neural network structure. Traffic state variables (that are continuous) are clustered into several discrete fuzzy sets and fuzzy rules associate these fuzzy sets with actions.

1.7. Brief introduction on reinforcement learning

Reinforcement learning is an area in machine learning research. One of the remarkable research papers by Jouffe (1998) illustrates the basic mechanism of reinforcement learning. In terms of training artificial agents to perform certain actions and reach long term goals, reinforcement learning gives rewards to agent actions that are close to target actions and penalize actions that are far away. The only information available for learning is the system feedback, which describes in terms of rewards and punishment on the task the agent has to realize. At each step, the agent receives a reinforcement feedback signal based on the last action it performed. The problem involves optimizing not only the direct reinforcement, but also the expected amount of reinforcements the agent can receive in the long run. In this paper, the artificial intelligent agent is actually the fuzzy rule based neural network. The objective of reinforcement learning is to extract driving rules from naturalistic dataset such that the neural network can perform similar actions as the “guiding” drivers.

In transportation research, reinforcement learning has been applied mostly in network route choice analysis and real time traffic signal control. Avineri and Prashker (2005) proposed a feedback reinforcement mechanism in modeling route-choice decision-making under uncertainty. Bogers et al. (2007) used reinforcement learning to simulate traveler route choice behavior with the assumption that travelers can learn about route travel time from their experiences and find the best route choice in the long run. In traffic signal optimization research, Abdulhai proposed a Q-learning algorithm in an isolated intersection and then a corridor in developing signal plans in a dynamic traffic environment (Abdulhai et al., 2003; Jacob and Abdulhai, 2006). Bingham (2001) used neural network fuzzy logic to partition traffic state of intersection for traffic controller agent to learn. Zhang et al. (2007) used a Neuro-Fuzzy Actor-Critic control method isolated intersection control and arterial control in real time.

1.8. Learning algorithm for continuous state variables

The existing reinforcement learning algorithms are mainly dealing with discrete state-action mapping problem. In our case, traffic state variables (vehicle speed for example) and action variables are continuous. Therefore, we discretize continuous variables using fuzzy-logic such that continuous state and can be fit into regular reinforcement learning algorithm framework. We will describe this in the latter section. To generate continuous actions, we revised the output of the learning algorithm. We will describe this later in the following section.

1.9. Paper layout

Firstly, the framework of our proposed fuzzy logic based neural network is presented. Subsequently, the naturalistic driving database and safety critical events extraction process are described. Then, the trained neural network models are evaluated by using samples from car-following situations and safety critical events. Finally, cross-validations between different drivers are performed, driver heterogeneities are illustrated and the idea of “mega” driver is presented.

2. The proposed fuzzy rule based neural network

Neural network acts as a driver simulator in this study. As Fig. 1 shows, the proposed neural network structure has four layers. The first layer is the input layer. Each node represents a continuous state variable. The second layer is the fuzzy membership layer. States are fuzzified into linguistic terms such as: “Speed is High” and “Speed is Low.” Each node is a discrete fuzzy set and has a membership function. The membership function can be triangular, trapezoidal or Gaussian (Jang et al., 1997). The third layer is the fuzzy rule layer. Each node represents a fuzzy rule and is connected to a number of discrete fuzzy sets of the second layer. For each true, a firing strength function is defined to indicate its strength. The fourth layer consists of a number of action nodes. Each fuzzy rule chooses one action. The output action is the weighted average of the selected actions (where fuzzy rule strengths are the associated weights). Several following paragraphs elaborate more on the layer design.

\[ S_i = \text{the } i\text{th input state variable.} \]
\[ n = \text{number of input variables.} \]
\[ NM_i = \text{number of fuzzy sets or membership functions for the } S_i. \]
\[ M_{i}^{(l)} = \text{a}(i)\text{th fuzzy set or membership function for the } i\text{th input variable.} \]
$R_j$ = the $j$th fuzzy rule.  
$N$ = number of fuzzy rules.  
$w_q^j$ = weight between $j$th fuzzy rule and action $q$.  
$\lambda_j$ = weight between $j$th fuzzy rule and critic.  
$A_q$ = output of $q$th discrete action.

where $i = 1, \ldots, n$, $a(i) = 1, \ldots, NM_i$, $j = 1, \ldots, m$ and $q = 1, \ldots, P$.

2.1. State layer

Same as most car-following models, we keep relative distance, relative speed and vehicle speed as state variables. Also, we believe that as a sequential task, current action should be related to previous action, we add actions of the previous time step as state variables. To model driver lateral behavior, we use yaw angle (angle of vehicle longitudinal relative axis to lane markings) as the lateral action and previous yaw angle is considered to be a state variable. To represent vehicle lateral position, we add variable lane offset (lateral position relative to the center of the lane) into state variables in safety critical event situations.

State variables are defined in the following equation:

$$
S_1 = v \\
S_2 = \Delta x \\
S_3 = \Delta v \\
S_4 = a' \\
S_5 = y' \\
S_6 = o
$$

(1)

where $S_i$ is the $i$th state variable, $v$ is the vehicle speed, $\Delta x$ is the relative distance to the leading vehicle, $\Delta v$ is the relative speed (speed of the leading vehicle minus the following vehicle), $a'$ is the previous acceleration, $y'$ is the previous yaw angle and $o$ is offset. $S_5$ and $S_6$ are used only when modeling safety critical events.
2.2. Fuzzy sets layer

In our designed network, each state variable is linked to two nodes (fuzzy sets) in the second layer. For state variable $S_i$, these two fuzzy sets are defined by linguistic terms: “$S_i$ is high” and “$S_i$ is low”. Each fuzzy set has a membership function. A triangular fuzzy membership function is used to transfer the state variable information to the two fuzzy sets,

$$
\mu_{low}(S_i) = \begin{cases} 
\frac{S_i - S_{lb,i}}{S_{ub,i} - S_{lb,i}} & S_i < S_{lb,i} \\
0 & S_i = S_{lb,i} \\
1 & S_i > S_{ub,i}
\end{cases} 
$$

$$
\mu_{high}(S_i) = \begin{cases} 
0 & S_i < S_{lb,i} \\
\frac{S_i - S_{lb,i}}{S_{ub,i} - S_{lb,i}} & S_{lb,i} < S_i < S_{ub,i} \\
1 & S_i > S_{ub,i}
\end{cases} 
$$

where $\mu_{low}(S_i)$ is the membership function of fuzzy set “$S_i$ is Low”, $\mu_{high}(S_i)$ is the membership function of fuzzy set “$S_i$ is High”, $S_{lb,i}$ is the lower bound of state variable $S_i$ and $S_{ub,i}$ is the upper bound of state variable $S_i$. The state bounds come from the minimum and maximum of naturalistic state data extracted from an individual driver. Upper and lower bounds are considered to be constant.

2.3. Fuzzy rule layer

Nodes in the third layer represent fuzzy rules. Fuzzy rules provide a state action mapping policy to determine the “optimal” actions from the fourth layer. Each fuzzy rule is applied to one combination of fuzzy sets from the second layer. For example, a fuzzy rule can be represented as:

WHEN “$S_1$ is low” and “$S_2$ is low” and “$S_3$ is high” and “$S_4$ is high” and “$S_5$ is high” and “$S_6$ is high”, THEN “Deceleration $a = -0.05$ g” and “Yaw angle $y = 0.1$ rad”.

In our design, to determine actions in driver car-following behavior, each fuzzy rule is associated with four fuzzy sets originated from four continuous state variables (speed, range, range rate, previous acceleration). Since each state variable has two fuzzy sets “Low” and “High”, the number of fuzzy sets combination should be $2^4 = 16$. Thus, 16 nodes are used in this layer. To determine actions during safety critical events, $2^6 = 64$ rules are used.

Firing strength for the $j$th fuzzy rule is calculated by

$$
FS_{lj} = 4 \prod_{i=1}^{4} \mu_{a(i)}(S_i) 
$$

where $a(i)$ is the linguistic term of fuzzy set (either “Low” or “High”) of the $i$th input state variable and $j$ represents the $j$th fuzzy rule. $\mu_{a(i)}$ is the membership function of $a(i)$.

2.4. Actor and critic layer

Actions in this paper are continuous. To achieve this goal, discrete action sets that include only a limited number of representative actions are defined and then continuous actions are generated through a weighted average of the discrete actions from the sets. Two discrete fuzzy sets are defined:

$$
A = \{a_{d1}, a_{d2}, a_{d3}, a_{d4}, a_{d5}\}
$$

$$
Y = \{y_{d1}, y_{d2}, y_{d3}, y_{d4}, y_{d5}\}
$$

where $a_{di}$ represent discrete accelerations and $y_{di}$ represent discrete yaw angles. $a_{d}$ and $y_{d}$ form actor nodes in this layer.

We use lower quartile (25th percentile, cuts off lowest 25% of data), median (50th percentile, cuts off 50% of data), upper quartile (75th percentile, cuts off highest 25% of data) and the maximum as the values of five discrete variables in each set.

To model car-following behavior, only set $A$ is needed. For one fuzzy rule, one discrete action $a_{di}$ is selected as "optimal". In the case of safety critical events, set $A$ and set $Y$ are required, one fuzzy rule chooses one $a_{di}$ and one $y_{di}$.

Critic node $V$ represents the value of the following state as a deterministic outcome of the actions determined by fuzzy rules. Critic node acts as intermediate nodes used in neural network training but not used in creating the final outputs when training is finished.
2.5. Neuron weights

Weights are located between the fuzzy rule layer and the actor critic layer. There are two types of weights: action weight and critic weight. Critic weight $k_j$ links the $j$th fuzzy rule to the critic node $V$ to determine the value of the following state. Value means how close the following state estimated by neural network is to the following state from naturalistic data and given in the following equation:

$$V_s = \sum_{j=1}^{64} FSR_j \cdot \lambda_j$$

(6)

where $FSR_j$ is the firing strength of the $j$th fuzzy rule $R_j$. $V_s$ is the value of state $s$. Weights $\lambda_j$ are only served as auxiliary parameters to update $w_{jk}$ and will described in detail.

Action weight $w_{jk}$ links the $j$th fuzzy rule to the $k$th action node. Weights $w_{jk}$ show competitions between discrete actions in the same action set. Fuzzy rule $R_j$ selects discrete action $a_{dk}$ from set $A$ when weight $w_{jk}$ is the largest and selects discrete action $y_{dk}$ with the largest weight $w_{jk}$. The reward function is an estimation of how good the selected actions are with respect to naturalistic state $S_t$, at time step $t$.

Temporal difference (TD) error (Sutton and Barto, 1998) is calculated as

$$\delta_a = r_{a,t+1} + \gamma V_{s_{t+1}} - V_{s_t}$$

(9)

$$\delta_y = r_{y,t+1} + \gamma V_{s_{t+1}} - V_{s_t}$$

(10)

where $r_{a,t+1}$ is the reward function when action $a$ is taken at state $S_t$, $r_{y,t+1}$ is the reward function when action $y$ is taken, $\gamma$ is the discounting factor. $V_{s_t}$ is the value of current state and $V_{s_{t+1}}$ is the value of the following state.

$V_{s_{t+1}}$ is calculated according to Eq. (6)

$$V_{s_{t+1}} = \sum_{j=1}^{N} FSR_{j,t+1} \lambda_{jt}$$

(11)

$\lambda$ and $w$ are updated by TD errors, where

$$\lambda_{jt+1} = \lambda_{jt} + \beta \delta_a FSR_{j,t}$$

(12)

$$\lambda_{jt+1} = \lambda_{jt} + \beta \delta_y FSR_{j,t}$$

(13)

$$w_{ak,t+1} = w_{ak,t} + \beta \delta_a FSR_{k,t}$$

(14)

$$w_{yk,t+1} = w_{yk,t} + \beta \delta_y FSR_{k,t}$$

(15)

where $\beta$ is the learning rate.

TD errors only update weights when the respective discrete actions are chosen. See from Eqs. (14) and (15), when discrete action $a_{dk}$ is chosen by rule $R_j$, only weight $w_{ak,t}$ is updated.
2.8. Reward function

Reward function provides guidance for neural network to follow. Reward function encourages neural network to take actions that are close to driver actions and penalizes actions that are far away. When the performance of an action outcome is good, reward function is positive and provides a greater probability to be chosen in the future. Vice versa, when the performance is bad, the reward function should be negative. In this paper, actions from naturalistic driving database are used as references to determine “good” or “bad” actions.

Relative error is calculated as

\[ e_a = \left| \frac{a - a_n}{a_n} \right| \]  \hspace{1cm} (16)

for acceleration and

\[ e_y = \left| \frac{y - y_n}{y_n} \right| \]  \hspace{1cm} (17)

for yaw angle

where \( e_a \) is the (absolute) relative error of acceleration, \( e_y \) is the relative error of yaw angle, \( a_n, y_n \) are the naturalistic actions from database.

Then, a non-negative parameter \( e_{th} \) is defined as an acceptance threshold. When \( e_a \) or \( e_y \) is less than \( e_{th} \), reward function is positive and the value of “good action” weights will be increased. Reward functions are defined as

\[ r_a = \alpha(e_{th} - e_a) \]  \hspace{1cm} (18)

\[ r_y = \alpha(e_{th} - e_y) \]  \hspace{1cm} (19)

where \( r_a \) is the reward function of acceleration, \( r_y \) is the reward function of yaw angle and \( \alpha \) is the scaling factor.

Initially, when the neural network does not know which action to take, it may choose an extreme bad action and result in large \( e_a \), \( e_y \). When such situation happens, the reward function value becomes negatively high and slow down the weights update process. Therefore, reward functions are bounded by the following equations:

\[ r_a = -\alpha \quad \text{when} \quad e_a \geq 1 \]  \hspace{1cm} (20)

and

\[ r_y = -\alpha \quad \text{when} \quad e_y \geq 1 \]  \hspace{1cm} (21)

3. Naturalistic driving data

We use data from the Naturalistic Truck Driving Study (NTDS) and the Naturalistic Car Driving Study (NCDS) collected by Virginia Tech Transportation Institute. As opposed to traditional epidemiological and experimental/empirical approaches, this \textit{in situ} process used drivers who operate vehicles that have been equipped with specialized sensor, processing, and recording equipment. In effect, the vehicle becomes the data collection device. The drivers operate and interact with these vehicles during their normal driving routines while the data collection equipment is continuously recording numerous items of interest during the entire driving. Naturalistic data collection methods require a sophisticated network of sensor, processing, and recording systems. This system provides a diverse collection of both on-road driving and driver (participant, non-driving) data, including measures such as driver input and performance (e.g., lane position, headway, etc.), four camera video views, and driver activity data. This information may be supplemented by subjective data, such as questionnaire data.

As part of the NTDS study (Olson et al., 2009), four companies and 100 drivers participated. Each participant in this on-road study was observed for approximately 4 consecutive work weeks. One hundred participants were recruited from four different trucking fleets across seven terminals and one to three trucks at each trucking fleet were instrumented (nine trucks total). After a participant finished 4 consecutive weeks of data collection, another participant started driving the instrumented truck. Three forms of data were collected by the Data Acquisition System (DAS): video, dynamic performance, and audio. Approximately 14,500 driving-data hours covering 735,000 miles traveled were collected. Nine trucks were instrumented with the DAS.

Naturalistic Car Driving Study (NCDS) used the same data collection instruments. Over 100 variables and five video views were recorded for 12–13 weeks of driving per participant. Dataset included 2 million vehicle miles and 43,000 h of driving. 15 police reported, 67 non-police-reported crashes and 761 near crashes were reported.

In our test, the “following” vehicle is the instrumented vehicle. The measured data include speed, longitudinal and lateral accelerations, yaw angle, brake and acceleration, range, range-rate and azimuth. Range and range-rate data represents relative distance and relative speed in our study and were collected by the instrumented forward viewing radar from the following vehicle. Data were recorded at 10 Hz.
3.1. Car-following situations criteria

Car-following situations were automatically extracted from the enormous volume of driving data in the database in order to analyze car-following behavior. The filtering process is an iterative process where initial values and conditions are used. After the events are flagged, they are reviewed in the video data to obtain minimum noise. Visual inspection of the first subsets created revealed some non car-following events, so additional filtering was thus performed to remove these events from the database.

Specifically, car following situations were extracted according to these pre-defined conditions:

- Radar target ID > 0.
- Radar range ≤ 120 m.
- \(-1.9 \text{ m} < \text{Range} \cdot \sin (\text{Azimuth}) < 1.9 \text{ m}\).
- Speed ≥ 20 km/h.
- Rho-inverse ≤ 1/610 m\(^{-1}\).
- Length of car following period ≥ 30 s.

The automatic extraction process was verified from a sample of events through video analysis. For the random sample of 50 periods, all 50 were valid car-following situations.

3.2. Safety critical events identification

Crash and near crash events are used as safety critical events in this study. The characteristics of crash and near crash events were identified and analyzed in a previous work by VTTI (Olson et al., 2009).

Crash: Any contact with an object, either moving or fixed, at any speed. Object includes other vehicles, roadside barriers, and objects on or off of the roadway, pedestrians, cyclists, or animals. Tire strike is also considered to be a crash.

Near-Crash: Any circumstance that requires a rapid, evasive maneuver (e.g., hard braking, steering) by the subject vehicle or any other vehicle, pedestrian, cyclist, or animal, in order to avoid a crash.

For an event to be flagged, only one of the triggers has to be met. Those triggers are as follows:

- Longitudinal deceleration greater than or equal to \(-0.2 \text{ g}\).
- Forward time-to-collision of less than or equal to 2 s.
- Swerve greater than or equal to 2 rad/s\(^2\).
- Lane tracker status equals abort (lane deviation).
- Critical incident button.
- Analyst identified.

![Fig. 2. Accelerations of Driver Agent and naturalistic data during car-following situation 1.](image)
4. Experiment and preliminary results

Ten car-following situations of one individual driver from Naturalistic Car Driving study were selected. The ten situations last around 1500 s. We select three situations (around 500 s) for validation and use the remaining 7 in training. For safety critical events, we are interested in rear end conflicts whose radar data were more reliable. Because we have only four or five rear end conflicts for each individual driver, we use all of them for neural network training. After training, we choose several events in validation.

Before training, errors form data collection measurement should be excluded. We arbitrarily set up additional constraints to filter out outliers.

- Speed \geq 0 \text{ km/h}.
- Range \geq 0 \text{ feet} and range \leq 400 \text{ feet} (when there is no leading vehicle in front, we assume range = 400 feet).
- Range rate \geq -10 \text{ feet/s}.

![Fig. 3. Accelerations of Driver Agent and naturalistic data during car-following situation 2.](image)

![Fig. 4. Accelerations of Driver Agent versus naturalistic data during car-following situation 3.](image)
In our design, at one time step of one event, fuzzy rules scan their associated weights, select the "optimal" actions and update weights according to reinforcement learning algorithm. Neural network keeps updating the weights from the beginning of the events until the end. Thus, weights are trained and updated by 10 times the length of events (10 Hz resolution data) in each iteration. Theoretically, when the differences of critic/actor weights between two consecutive iterations become very small, the training process is considered to be finished. However, during the training process, the convergence of the weights may be premature and result in a bad local optimal. To avoid this premature convergence, we suggest use large number of training iterations. In our experiment, we implemented 400 iterations in training. Since each car-following situation in our database has over 1000 timing steps, neural network have been trained $1000 \times 400 = 400,000$ times and should be sufficiently trained.

During the learning process, the memory discount factor $\gamma$, the learning factor $\beta$ and the reward function scaling factor $\alpha$ affect the learning speed of the neural network. $\gamma$ controls the memory that the value of recently occurring states are weighted more. $\beta$ shows how fast neural network digests new information. $\alpha$ controls the magnitude of the reward function and eth controls the sign of the reward function. In our test, we set $\beta = 0.6$, $\gamma = 0.9$, $\alpha = 10$ and eth = 0.2.

Fig. 5. Acceleration of Driver Agent A, Event A1.

Fig. 6. Yaw angle of Driver Agent A, Event A1.
4.1. Car following situations: longitudinal accelerations

Figs. 3 and 4 show the longitudinal acceleration derived by neural network during the three randomly selected car-following situations in validation dataset. We name the neural network simulator “Driver Agent” since it acts as a clone of the “guiding” driver. The blue scatter plots represent the naturalistic acceleration and the green curves show the Driver Agent acceleration. As a result, the agent can capture driver naturalistic behavior quite well even with a lot of fluctuations in naturalistic acceleration data. Although differences in driving behavior during different car-following situations from the same driver is expected (the discrepancy at around 100–200 tenth of seconds in Fig. 2 explain behavioral differences), the proposed neural network agent driver shows a good approximation.

4.2. Safety critical events

Figs. 5 and 6 show the longitudinal agent action acceleration and lateral action yaw angle during one event of a selected Truck Driver A. Unlike car-following situations, we use truck drivers instead of car drivers in this study because safety critical
events data (especially from radar) from truck driving study are much more accurate. The blue scatter plots represent the naturalistic driving actions and the green curves represent Driver Agent actions. Neural network captures driver naturalistic behavior quite well in this event with a high $R$ squared degree of accuracy (estimated actions versus naturalistic actions, $R > 0.95$) for acceleration and yaw angle respectively. We also test neural network performance on another event from the same driver, see Figs. 7 and 8.

Agent A captures Driver A’s naturalistic behavior during both events A2 and A1 quite well. It seems that even if states from two events could be significantly different, Agent A could recognize their characteristics and take appropriate actions. It may prove the concept that our proposed methodology will not result in an “average” driving behavior if using more events in training.

From the above four figures, during some periods of the events, such as $t = 28–30$ s in Fig. 5, the action of Agent A deviates a little bit from naturalistic data. We want to mention that data collection errors in traffic states and driver behavior inconsistency within an event may contribute to the discrepancies. Occasionally, when the leading vehicle ran out of radar detection zone, the radar assumed there was no vehicle in front by mistake. As a consequence, wrong traffic states leads to wrong actions. In reality, driver inconsistency issue can occur at any time, but our derived driving rules are fixed for individual drivers.

![Longitudinal Action Estimation](image1)

**Fig. 9.** Acceleration of Driver Agent B, Event B1.

![Lateral Action Estimation](image2)

**Fig. 10.** Yaw Angle of Driver Agent B, Event B1.
4.3. Cross validation

The idea of cross validation is to present the heterogeneities in driving behavior from different drivers in similar driving situations. To achieve this goal, driving rule of one driver (Driver A) could be applied an event (Event B1) experienced by another driver (driver B). Supposedly, Driver A will behave differently than Driver B in Event B1.

In our test, we estimate Agent A’s actions during Event B1 based on the driving rules obtained from A’s Event A1. Similarly, we estimate Agent B’s actions using Event A1.

First, Figs. 9 and 10 show an Agent B’s longitudinal and lateral actions during Event B1. Agent B has been trained using all its safety critical events.

Then, the longitudinal and lateral actions of Agent B in Event A1 are shown in Figs. 11 and 12. Figs. 13 and 14 show the longitudinal and lateral actions of Agent A in Event B1. It is very clear that Driver A and Driver B behave differently.

Table 1 lists the $R^2$ squared values as a statistical representation of the degree of accuracy of neural network estimated actions. The upper left corner and lower right corner of Table 1 represent degree of approximation when using events from the same driver in training and validation. The degree of accuracy is much higher than the cross validation counterparts (upper right and lower left). Again, heterogeneities between two drivers are clear.

![Fig. 11. Acceleration of Driver Agent B, Event A1.](image1)

![Fig. 12. Yaw Angle of Driver Agent B, Event A1.](image2)
4.4. The "Mega Agent" Idea

We want to design an imaginary neural network simulator agent (we name it Mega Agent) that can capture the behavior of both Driver A and Driver B but not an average behavior. The purposed fuzzy rule based neural network should handle this challenge. In our neural network design, different fuzzy rules dominate different locations of state space. According to our

Table 1

<table>
<thead>
<tr>
<th>Event</th>
<th>Agent A Acceleration</th>
<th>Yaw</th>
<th>Agent B Acceleration</th>
<th>Yaw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driver A</td>
<td>0.98</td>
<td>0.97</td>
<td>0.81</td>
<td>0.83</td>
</tr>
<tr>
<td>Driver B</td>
<td>0.82</td>
<td>0.60</td>
<td>0.97</td>
<td>0.92</td>
</tr>
</tbody>
</table>

Fig. 13. Acceleration of Driver Agent A, Event B1.

Fig. 14. Acceleration of Driver Agent A, Event B1.
algorithm, only the fuzzy rules that dominate state locations are updated, rules are actually updated “independently” if two states located far away.

The state space in this problem has four or six dimensions. Thus, differences in one dimension (lower speed and high speed for example) can cause two states separated from each other in the state space. This is the case of the safety critical events data: state variables from any two events vary significantly. Thus, states from two events located far away from each other in state space.

Preliminary, we use all the events of Driver A and Driver B to train the Mega Agent. Training parameters such as fuzzy sets thresholds and discrete action sets are adjusted before training. Performances of the Mega Agent (use Event A1 and B1) are presented in Figs. 15–18.

Compared to Figs. 11–14, the Mega Agent is capable to differentiate driving behavior between Driver A and Driver B. Mega Agent behaves like A in Event A1 and behaves like B in Event B1.

Table 2 shows the $R^2$ squared values of the Mega agent. Accordingly, $R^2$ squared values of the Mega Agent area are quite high (>0.9) although not as good as the values when the same events are used in both training and validation. Thus, Mega Agent is capable of mimicking the behavior of Driver A and B at the same time without losing much driver specificity.

Fig. 15. Acceleration of Mega Agent, Event A1.

Fig. 16. Yaw Angle of Mega Agent, Event A1.
In reality, a conservative driver might never experience safety critical events but an aggressive driver may have. In such case, the conservative driver may have no idea about the actions to do to get out of emergency situation if he/she gets into it in a sudden fashion. Through this Mega Agent training, the conservative driver will “learn” the crash avoidance actions from an aggressive driver and help him/herself to evade from upcoming crashes.

**Table 2**

<table>
<thead>
<tr>
<th>Event</th>
<th>Agent A Acceleration</th>
<th>Agent A Yaw</th>
<th>Agent B Acceleration</th>
<th>Agent B Yaw</th>
<th>Mega Agent Acceleration</th>
<th>Mega Agent Yaw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driver A</td>
<td>0.98</td>
<td>0.97</td>
<td>0.81</td>
<td>0.83</td>
<td>0.98</td>
<td>0.95</td>
</tr>
<tr>
<td>Driver B</td>
<td>0.82</td>
<td>0.6</td>
<td>0.97</td>
<td>0.92</td>
<td>0.97</td>
<td>0.91</td>
</tr>
</tbody>
</table>
5. Conclusions

In this paper, we first proposed a fuzzy rule based neural network approach to model driver decision process during car-following situations and safety critical events at the individual level. Fuzzy logic is used to partition traffic state variables and reinforcement learning method is used for neural network to learn driving behavior from naturalistic data. Estimated vehicle actions by neural network are compared to naturalistic data in our selected data sample. Our preliminary results show that neural network is able to capture driver behavior quite well.

It is worth mentioning that this research is an attempt to apply fuzzy rules based neural network machine learning technique in solving high dimensional state problems in microscopic traffic flow research. From the perspective of microscopic traffic behavior modeling, the proposed methodology is able to simulate lateral actions and will bring new insights in modeling driver maneuvering behavior. The proposed methodology also looks into events from different drivers to present driver behavior heterogeneities.

The next step of this research is to extend the capability of fuzzy based neural network to model other traffic behavior, such as lane-changing behavior, driver merging behavior at the upstream and downstream of ramps. It would be interesting to model individual driver behavior decision making process under these specific traffic conditions.

For the technical part of this paper, the training parameters speed, memory and scaling factor are fixed, but it would be interesting to see the performance of neural network using different combinations of these factor sets. Moreover, during our test, we found out that the performance of neural network was very sensitive to the driver dependent training parameters, such as discrete action sets and state bounds. Theoretically, it would be better if these parameters are pre optimized before training. In our approach, we set the parameters based on statistical quartiles. Nevertheless, training parameters optimization methodology is worth developing in future research.

Acknowledgements

This material is based upon work supported by the Federal Highway Administration under Agreement No. DTFH61-09-H-00007. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the view of the Federal Highway Administration.

The authors would like to express thanks to Dr. C.Y. David Yang, the FHWA Agreement Manager, for his continued support and guidance during this project. They would also like to thank individuals at Virginia Tech and the Virginia Tech Transportation Institute who contributed to the study in various ways: Greg Fitch, Shane McLaughlin, Brian Daily and Rebecca Olson.

References