MOHA

A Multi-Mode Hybrid Automaton Model for Learning Car-Following Behaviors

Lin, Qin; Zhang, Yihuan; Verwer, Sicco; Wang, Jun

DOI
10.1109/TITS.2018.2823418

Publication date
2019

Document Version
Accepted author manuscript

Published in
IEEE Transactions on Intelligent Transportation Systems

MOHA: A Multi-Mode Hybrid Automaton Model for Learning Car-Following Behaviors

Qin Lin, Yihuan Zhang, Sicco Verwer, and Jun Wang, *Senior Member, IEEE*

***Abstract*— This paper proposes a novel hybrid model for learning discrete and continuous dynamics of car-following behaviors. Multiple modes representing driving patterns are identified by partitioning the model into groups of states. The model is visualizable and interpretable for car-following behavior recognition, traffic simulation, and human-like cruise control. The experimental results using the next generation simulation datasets demonstrate its superior fitting accuracy over conventional models.**

***Index Terms*— Hybrid automaton, car-following behavior, simulation and control.**

I. Introduction

LEARNING car-following from real driving data is helpful for behavior recognition, traffic simulation, and human-like cruise control. A car-following model bridges *environmental variables*, like subject vehicle speeds, relative distances and relative speeds to a leading vehicle, and *control variables*, like acceleration or deceleration. Conventional approaches for representing car-following behaviors mainly include linear [3], [4] and non-linear models [5], [6]. A *gross fitting* strategy is usually used for identification, i.e., fitting a car-following model on all the collected data. It does not fully capture driver behavior in different driving scenarios due to heterogeneities of inter-driver difference [7] and intra-driver difference [8]. The inter-driver difference, discovering that different car-following models may apply to different drivers, is useful for driving behavior modeling and skills evaluation of individual drivers. The intra-driver modeling basically deals with the problem that individual drivers change their behaviors over the data collection period. The research goal in this paper is learning a general intra-driver model from large-scale human driving data.

This paper proposes a novel framework to learn a **m**ulti-m**o**de **h**ybrid **a**utomaton model (MOHA) averaging driving behaviors from thousands of human drivers' real driving data. The idea is discretizing environmental variables on a coarse-grained level and obtaining a stateful model. Distinguished driving patterns represented by multiple modes are identified by partitioning such a model into groups of states. Corresponding groups of car-following models are identified on a fine-grained level from real-value time series data. The underlying discrete and continuous dynamics of driving behaviors are discovered using such a hybrid model learning.

Higgs and Abbas [9] deal with pattern mining and divide-and-rule learning for car-following behaviors. They use time series segmentation and k-means clustering. The noticeable disadvantage is that it loses sight of time information. In addition, the patterns are not interpretable, and the switching among patterns is not discovered. These problems will be solved by MOHA in this paper. Another related work in [10] only uses conventional discrete timed automata, while this paper models driving behavior using a novel hybrid model with multiple modes. Verwer *et al.* [10] only consider a simple behavior classification problem, while our model is used for recognition, simulation, and control. This journal version extends [1] in three important ways. First, the model in [1] can be only used in an off-line fashion. In this version, the model is improved to be used on-line by a new state pattern mining approach to avoid clustering ambiguity problem in [1] (see Section III A). Such an improvement boosts the rationality of the model. The model has been validated in more datasets in this paper. Second, a potential application of cruise controller simulation is studied, which is not in [1] but initialized in [2]. This part actually covers an interesting, novel, and promising topic about learning for control. The experiments in [2] have also been updated by using the on-line model in this paper. Third, the model is compared with the state-of-the-art work [9], which is a future work in [1].

Fig. 1 shows a flowchart of the proposed approach. A symbolic representation of time series data provides a high-level overview of behavioral dynamics. Since time information is crucial for driving behavior modeling, time difference between two consecutive distinct events is computed to obtain timed strings. The learning process benefits from such a timed representation because it *explicitly* encodes the underlying varying-duration behaviors. A state-of-the-art automata learning algorithm named RTI+ (real-time identification from positive data) is applied to learn a direct and cyclic graphical model that describes discrete dynamics. On basis of this discrete event model, frequent common *state sequences* as patterns are extracted and clustered to identify modes. The continuous

Manuscript received March 27, 2017; revised September 24, 2017, January 31, 2018, and March 19, 2018; accepted March 26, 2018. This work was supported in part by the Technologiestichting STW VENI Project under Grant 13136 (MANTA), in part by NWO Project under Grant 62001628 (LEMMA), and in part by the National Natural Science Foundation of China under Grants 61473209 and 61773291. The Associate Editor for this paper was J. Zhang. *(Corresponding author: Jun Wang.)*

Q. Lin and S. Verwer are with the Intelligent Systems Department, Technische Universiteit Delft, 2600 AA Delft, The Netherlands.

Y. Zhang and J. Wang are with the Control Science and Engineering Department, Tongji University, Shanghai 201804, China (e-mail: junwang@tongji.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2018.2823418

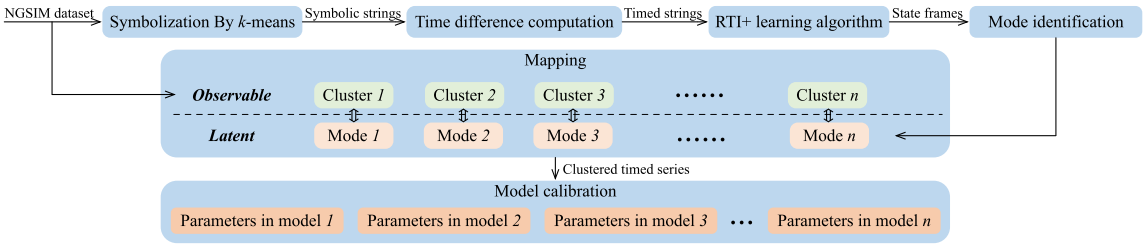

Fig. 1. Flowchart of MOHA. The clustering is first deployed on the states to identify modes (the latent variables under the dotted line). The original numerical time series data are also clustered correspondingly with the help of the mapping. Different car-following models are trained from the clustered time series data in these modes. The number of modes determines the number of car-following models needed and the number of clusters in the original time series data.

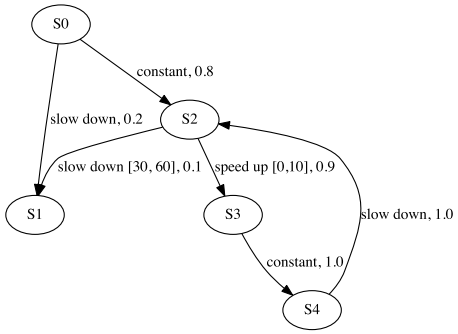

Fig. 2. A simple example of PDRTA. Note that it is only used for illustrating MOHA learning. It is not the model learned in the experiment.

linear and non-linear car-following models: Helly [4] and IDM [6] are trained from time series data in individual modes using a differential evolution algorithm.

This paper for the first time uses a multi-mode hybrid model to learn car-following behaviors. The results show that the fitting accuracy is significantly improved over conventional models. Moreover, the potential application of MOHA is promising, e.g. a valid car-following model can be derived for traffic simulations and a human-like controller can be achieved for car following.

The remainder of the paper is organized as follows. Sections II and III discuss discrete timed automata learning and mode identification of MOHA. Experiments are conducted in Section IV. Another potential application is discussed in Section V. The conclusion is in Section VI.

II. Learning Discrete Timed Automata for MOHA

Learning regular languages or (minimum) deterministic finite automata (DFA) is still the main task of grammatical inference [11]. The learning algorithms require discrete event strings as input. However, lifetime of events is important for characterizing behaviors. An algorithm for efficient learning timed automata is proposed in [12]. This algorithm uses the timed strings $(a_1, t_1)(a_2, t_2)\cdots(a_n, t_n)$ to *explicitly* represent discrete events, where a_i is the ith event occurring with a time delay t_i from the $(i-1)$th event. A probabilistic deterministic real timed automaton (PDRTA) model defines a probability distribution over such timed strings. A PDRTA is defined in Definition 1. An example of PDRTA is illustrated in Fig. 2.

Definition 1: A PDRTA is a 4-tuple $\langle \mathcal{A}, \mathcal{E}, \mathcal{T}, \mathcal{H} \rangle$, where $\mathcal{A} = \langle Q, \Sigma, \Delta, q_0 \rangle$ is a 4-tuple defining the machine structure: Q is a finite set of states, Σ is a finite alphabet, Δ is a finite set of transitions, and $q_0 \in Q$ is the initial state. $\mathcal{E}$ and $\mathcal{T}$ are the event and time probability distributions, respectively. $\mathcal{E}: Q \times \Sigma \to [0, 1]$ returns the probability of generating/observing a given event in a given state. $\mathcal{T}: Q \times \mathcal{H} \to [0, 1]$ returns the same but for a given time range $[m, m'] \in \mathcal{H}$, where $\mathcal{H}$ is a finite set of non-overlapping intervals in $\mathbb{R}_+$. A transition $\delta \in \Delta$ in a PDRTA is a tuple $\langle q, q', a, [m, m'] \rangle$, where $q, q' \in Q$ are the source and target states, $a \in \Sigma$ is a symbol and $[m, m']$ is a temporal guard.

This paper uses the individual vehicle trajectories from a public dataset named the Next Generation SIMulation (NGSIM) [13]. Both the I80 and the US101 datasets in NGSIM provide precise location of each vehicle trajectory at a sampling frequency 10 Hz. Each dataset contains three 15-minute periods (noted as I80-1, I80-2, I80-3, US101-1, US101-2, US101-3). Based on the trajectory data, each pair of consecutive vehicles in a lane is extracted for learning car-following behaviors. Note that vehicle speed, relative distance, and relative speed are explanatory variables as inputs. Longitudinal acceleration is a response variable as an output.

A. Timed Strings Representation

k-means clustering method is used to symbolize numeric data. The clustering centroids in the I80-1 dataset are in Table I. A break point finding method is used to determine the "optimal" number of clusters [14]. The idea is to find the number of clusters that avoids sharp dropping of the criterion called "sum of squares within the cluster". Symbolic strings are then converted to timed strings. Fig. 3 shows a simplified example of speed feature to illustrate how the conversion works. In the experiment, all 3 input features are clustered at once.

B. Learning PDRTAs

A state-of-the-art machine learning algorithm named RTI+ is used to learn car-following behaviors from unlabeled data. For more details about this algorithm, readers are referred to [12]. Traditional state machine learning algorithm starts by building a large tree-shaped automaton called augmented prefix tree acceptor from a sample of input strings. Every state of this tree can be reached by exactly one untimed string and therefore encodes exactly the input sample. State merges and transition splits are two main operations of structure and temporal guards learning in RTI+. A split of a

TABLE I

CODE BOOK OF k-MEANS CENTROIDS FOR NUMERIC DATA IN THE I80-1 DATASET

Symbols	a	b	c	d	e	f	g	h	i	j
Relative speed centroid (m/s)	0.79	3.02	−2.88	4.82	−3.12	−0.98	−9.67	2.52	−7.02	0.12
Relative distance centroid (m)	57.87	36.13	15.63	15.55	204.18	96.09	39.74	24.00	24.47	10.13
Speed centroid (m/s)	13.69	10.54	7.74	5.94	19.41	17.25	12.99	8.38	10.10	4.12

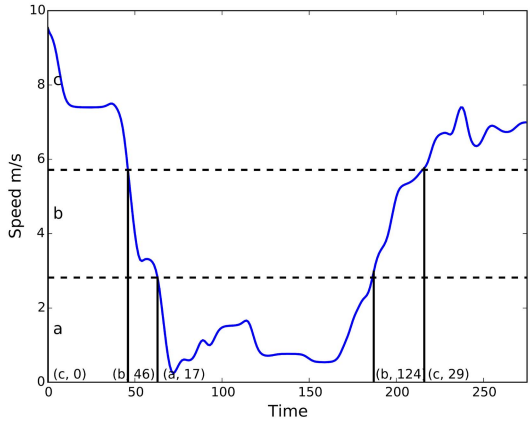

Fig. 3. Discretization of time series data in I80-1. Instead of using complete symbolic strings with total length 275, the timed string has only 5 tuples as input: $(c, 0)(b, 46)(a, 17)(b, 124)(c, 29)$. The number next to the symbol in each tuple denotes the time difference since the last event.

TABLE II

MAPPING BETWEEN TIMED STRINGS AND STATE SEQUENCES

Frames	Timed strings	State sequences
1	(slow down, 0)	S0, S1
2	(slow down, 0)	S0, S1
3	(const. speed, 0), (slow down, 50)	S0, S2, S1
4	(const. speed, 0), (speed up, 10), (const. speed, 5)	S0, S2, S3, S4
5	(const. speed, 0), (speed up, 6), (const. speed, 15), (slow down, 5), (speed up, 4), (const. speed, 15)	S0, S2, S3, S4, S2, S3, S4
⋮	...	...

transition $\delta = \langle q, q', a, [m, m'] \rangle$ at time t creates two new transitions $\langle q, q_1, a, [m, t] \rangle$ and $\langle q, q_2, a, [t+1, m'] \rangle$ when two tails show significantly different behaviors. The algorithm also greedily merges pairs of states (q, q') forming a smaller and smaller machine that generalizes over samples.

III. IDENTIFYING MODES IN MOHA

Once the timed automaton is learned, a mapping is built between the *observable variables* (time series data/symbolic data) and the *latent variables* (state sequences) by inputting the training strings to the model again.

Table II shows the frames mapping between the input timed strings and the output state sequences for the model in Fig. 2. The original time series data map to the states by looking at their timed strings and the corresponding states arrived. The parameters in numeric car-following models are obtained in each individual cluster of time series data, see Fig. 1.

Algorithm 1 Mode Identification With MISSI

Input: Dataset containing N state sequences DS, minimum substring length L_{min}, support ϵ, cut-off threshold τ
Output: state clusters (modes) SC
Extract substrings with length greater than L_{min} from DS;
Extract frequent substrings with occurrence greater than ϵ;
Subsrtings clustering on their similarity;
For *all unique states except initial state in SC* **do**
| majority voting to determine each state's belonging cluster;
End

A. State Subsequence Clustering

This paper proposes an algorithm for **m**ode **i**dentification on **s**tate **s**ubsequence cluster**i**ng (MISSI). An overview of MISSI is shown in Algorithm 1. Jaro-score is used to measure similarity between two strings [15]. A hierarchical clustering method is applied on frequent common strings using Jaro-score as distance. At the beginning, every string represents a unique cluster, then a hierarchical clustering computes the pairwise distance between two clusters. For clusters containing multiple strings, the average distance is computed. In each iteration, only one pair of clusters is merged. The *cut-off* threshold τ is a user-defined parameter for determining the number of clusters. The sequences in Table II are used to explain how MISSI works step by step.

1) Extracting frequent common substrings:
S0, S1; S0, S2; S2, S3; S3, S4; S2, S3, S4; S0, S2, S3. ϵ is set to 2 in this case, and thus the substring S2, S1 will not be extracted as a frequent common substring because it only occurs in one state sequence.
2) Clustering substrings: for instance, 2 clusters are obtained after a hierarchical clustering:
Substring cluster 1: S0, S1; Substring cluster 2: S0, S2; S0, S2, S3; S2, S3; S3, S4; S2, S3, S4.
3) Clustering states: State cluster 1 by a majority voting: S1; State cluster 2: S2, S3, S4.

The initial state S0 is not necessary to be classified. Note that some states will be in multiple substring clusters (e.g., how to assign S2's state cluster ID if S0, S2 was in substring cluster 1 instead of substring cluster 2). To avoid the ambiguity of states interpretation, ambiguous states are classified by an additional majority voting. For the aforementioned case of S0, S2 in substring cluster 1, S2 will be classified into state cluster 2 because the majority of S2 exists in the substring cluster 2. A new example arriving string S0, S2,

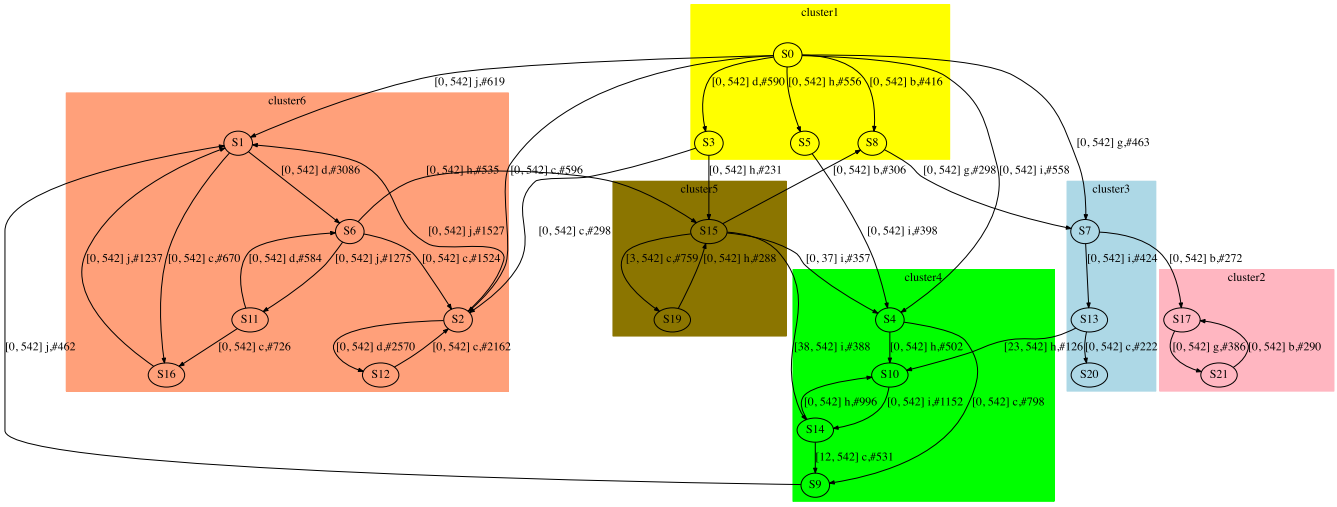

Fig. 4. Real-timed automaton learned from the whole I80-1 dataset. Note that the original solution from RTI+ has 34 states in total. The states with very low frequencies are removed to simplify the model interpretation. For instance, states with event "e" occurring very rarely are not shown in this figure. The arcs represent transitions between states. The information of timed guards, events, and number of occurrences is also printed next to the arcs, e.g., *[0, 542] j, #619* from S0 to S1.

S3, S4, S2, S1 is assigned mode IDs 2, 2, 2, 2, 1 based on the aforementioned clusters obtained (again, the initial state is skipped).

B. On-Line Inference

The states estimation is achieved online over arriving stream data. Starting with the initial state, the observed numeric data are converted to symbols according to the k-means codebook, e.g., assigning the observation as "constant" by looking at the closest centroid. The state is then transited from S0 to S2. The following transition is triggered until a new observation, e.g. "speed up", occurs. The time difference is also computed between two consecutive distinct events. The time guards in MOHA serves as additional conditions for checking the transition where the timing is inside a valid time guard, e.g., an event with different timings is possible to be triggered by different transition path. The mode ID and its corresponding car-following model is obtained because the modes have already been obtained in the step of state clustering. The output control variable is computed from the input data. The generation of car-following traces includes one-step and multi-step approaches [16]. The one-step approach evaluates the difference between the computed output with the ground truth at each time point. The real status of the subject vehicle is updated from the dataset in the next time point, thus the error will not be accumulated. The results of one-step testing are analyzed in Section IV. The multi-step generation only sets the initial state of the subject vehicle. During the generation procedure, real values of the subject vehicle in the dataset are not used to update its real-time information. The movement of the subject vehicle is updated completely using the computation model. The multi-step testing is discussed in Section V. Note that in both settings, the trajectories of leading vehicles are directly from the dataset.

IV. Experimental Results

The datasets are split to two sections, i.e. 80% for training and 20% for testing. k-means discretization and the state sequence clustering are both conducted only in the training dataset. To avoid over-fitting and obtain a less biased evaluation, the testing data are not used during clustering. To make a complete overview of driving behaviors, the whole dataset is used for model interpretation. As a consequence, some settings in the training dataset are not necessarily the same as those in the whole dataset. All the experiments are reproducible with our shared data and code in our repository.[1]

A. Model Interpretation

One of the main advantages of MOHA is its interpretability. The learned model from the whole I80-1 dataset is illustrated in Fig. 4. All modes are distinguished with different colors. There are loops with significantly large occurrences in Cluster 6, e.g., state sequence: "S1-S6-S11-S16-S1" with a corresponding symbolic transition loop: "d-j-c-j". The relative distances of "c" and "d" are very close, cf. Table I, but having negative and positive relative speed, respectively. They are associated with "j", which has a very small speed difference. This sequence can be interpreted as **steady car-following behavior at short distances**, i.e., adapting the relative speed to the leading vehicle.

Similarly, the loops in Clusters 2 and 4 represent the behaviors of **steady long distance** and **steady medium distance car-following** respectively. An intermediate state S15 in Cluster 5 indicates how to transfer out of Cluster 6. For example, "S6-S15-S4" with transitions "h" and "i", i.e., slowing down and speeding up to catch up, from the short distance following in Cluster 6 to the medium distance following in Cluster 4. The time split can also be seen in two branches of $[0, 37]$, i and

[1]The source code, shared data and animated video can be found in our repository: https://bitbucket.org/anjutalq/carfollowingrti

TABLE III
INTERPRETATION OF CLUSTERS IN THE I80-1 DATASET

Cluster ID	Dominant states	Dominant symbolic loops	Description
1	Remaining states	-	Intermediate process and infrequent states
2	17, 21	b-g	Steady long distance car-following
3	7,13,20	-	Intermediate process
4	4, 9, 10, 14	h-i	Steady medium distance car-following
5	15, 19	-	Intermediate process
6	1, 2, 6, 11, 12, 16	c-d-j	Steady short distance car-following

[38, 542], i from S15. They share the same symbolic transition condition but have distinct time guards. This means the speed-up action "i" is followed by a short or long duration of "h", i.e., after how much time the subject vehicle notices that its relative distance has been expanded and begins to catch up. A complete car-following example in the I80-1 dataset is also shown via an animated video in our repository.

B. Competing Methods

The following classical or state-of-the-art methods are also implemented in this paper for a fair and convincing comparison.

- **Gross fitting:** It uses a single car-following model. By comparing with the traditional method of gross fitting, we can investigate how much improvement can get using multiple models.
- **Symbolic clustering:** The clustering is deployed with the same setting as the state sequence clustering directly on the symbolic data. The comparison with symbolic clustering shows the value of clustering latent state sequences instead of clustering observable symbolic sequences.
- **Higgs:** It is a state-of-the-art method proposed by Higgs and Abbas [9]. It first segments multi-variate time series data by minimizing their variance by a bottom-up strategy. The segments are initialized with equal length. Then a pair of the adjacent segments with the lowest merge cost is merged in each iteration. The mean values representing the segmented piece-wise data are clustered by k-means method.
- **State model:** It is also based on the learned timed automaton. Without partitioning the model into modes, individual models are trained in each state. It introduces a large amount of parameters but helps us to investigate the benefit of clustering states.

The root-mean-square error (RMSE) is a widely used indicator for evaluating the acceleration error of car-following models. In addition, to overcome overestimation in high and low values, some papers [17], [18] use speed's *relative error* (RE), *absolute error* (AE), and *mix error* (ME) as additional indicators, which are defined as follows:

$$\mathrm{RE} = \sqrt{\left\langle \left(\frac{s^{sim} - s^{real}}{s^{real}} \right)^2 \right\rangle} \tag{1}$$

$$\mathrm{AE} = \sqrt{\frac{\left\langle \left(s^{sim} - s^{real} \right)^2 \right\rangle}{\left\langle s^{real} \right\rangle^2}} \tag{2}$$

$$\mathrm{ME} = \sqrt{\frac{1}{\left\langle \left| s^{real} \right| \right\rangle} \left\langle \frac{\left(s^{sim} - s^{real} \right)^2}{\left| s^{real} \right|} \right\rangle} \tag{3}$$

where s^{sim} and s^{real} are the computed and the real values respectively. $\langle s \rangle$ is the average value defined as $\langle s \rangle = \frac{1}{N} \sum_{i=1}^{N} s\,(i)$.

Table IV and Table V show the comparison using aforementioned indicators and their standard deviations. The average improvement of MOHA over gross fitting is summarized in Table VI and Table VII. Note that here a single-step approach is deployed for both training and testing. A multi-step approach will be also tested for a trajectory simulation in Section V. Readers are referred to [16] for a more detailed explanation about the difference. The runtime on the Intel 3.1-GHz i7 processors is also compared in Table VIII.

- MOHA and state model outperform gross fitting and other two clustering models. Both of them are based on the learned timed automata. State model uses much more parameters, e.g., 34 models in a 34-state automaton. The training of the state model does not take too long since the data are split over more models, and fewer data lead to a fast convergence. Such an overfitting problem is due to too few data during the training. To balance the bias (fitting error) and the variance (model complexity), it is suggested to use MOHA with high accuracy and low complexity.
- Symbolic method is the third best model with competing performance. However, such a model-free pattern mining method can only serve as a clustering tool rather than a control model generating the following vehicle's trajectories.
- RMSE of acceleration is a more sensitive indicator of larger magnitude. Due to an integral relation from acceleration to speed, the speed's error has been smoothed and thus has a smaller magnitude. In addition, the testing is essentially a one-step prediction evaluation, i.e., the error will not be accumulated. Therefore, the improvement is less obvious than the multiple-step prediction.
- The symbolic labeling, the timed automata learning, and the sequence clustering are quite efficient in computation cost. They are promising in car-following model calibration on large scale data. Among all the clustering methods, the Higgs model takes the longest time on clustering due to the time-consuming segmentation.

V. A HUMAN-LIKE CRUISE CONTROLLER

The drawbacks of an automatic cruise control (ACC) system lie on an inconsistency between systems and human drivers,

TABLE IV
TESTING DATA ERROR IN NGSIM DATASETS: HELLY MODEL

Mean ± Std.		Helly					
		I80-1	I80-2	I80-3	US101-1	US101-2	US101-3
RMSE (m/s^2)	Gross	0.9981±0.3343	1.4641±0.3971	1.6424±0.3754	1.6429±0.2859	1.5413±0.3051	1.3454±0.3402
	Symbolic	0.9319±0.3218	1.3774±0.3623	1.5648±0.3836	1.6005±0.2916	1.3656±0.2170	1.3012±0.2812
	Higgs	1.0999±0.5240	1.4207±0.3743	1.6216±0.3693	1.6273±0.2999	1.4753±0.3402	1.3220±0.2971
	MOHA	0.9225±0.3156	1.3659±0.3653	1.5552±0.3714	1.5962±0.2875	1.3452±0.2054	1.2984±0.2767
	State Model	**0.9122±0.3231**	**1.3648±0.3629**	**1.5541±0.3778**	**1.5899±0.2781**	**1.3405±0.2064**	**1.2962±0.2775**
RE (m/s)	Gross	0.0943±0.2848	0.0278±0.0103	0.0339±0.0194	0.0450±0.0949	0.0315±0.0682	0.0451±0.0780
	Symbolic	0.0145±0.0080	0.0267±0.0103	**0.0269±0.0100**	0.0186±0.0081	0.0178±0.0068	0.0245±0.0107
	Higgs	0.0507±0.0304	0.0266±0.0506	0.0274±0.0555	0.0355±0.0689	0.0250±0.0399	0.0443±0.0511
	MOHA	0.0146±0.0082	**0.0265±0.0102**	**0.0269±0.0100**	0.0188±0.0085	0.0178±0.0068	**0.0244±0.0106**
	State Model	**0.0144±0.0081**	0.0266±0.0104	0.0269±0.0101	**0.0185±0.0081**	**0.0177±0.0068**	0.0244±0.0107
AE (m/s)	Gross	0.0148±0.0095	0.0257±0.0092	0.0412±0.0244	0.0278±0.0097	0.0199±0.0049	0.0329±0.0115
	Symbolic	0.0127±0.0059	0.0245±0.0092	0.0256±0.0097	0.0165±0.0060	0.0164±0.0048	0.0207±0.0079
	Higgs	0.0159±0.0092	0.0346±0.0179	0.0360±0.0258	0.0206±0.0106	0.0180±0.0055	0.0290±0.0143
	MOHA	0.0128±0.0059	**0.0243±0.0091**	**0.0256±0.0095**	0.0166±0.0061	**0.0156±0.0047**	0.0201±0.0075
	State Model	**0.0126±0.0060**	0.0243±0.0092	0.0256±0.0096	**0.0164±0.0059**	0.0177±0.0068	**0.0200±0.0075**
ME (m/s)	Gross	0.0184±0.0194	0.0261±0.0092	0.0422±0.0168	0.0291±0.0109	0.0219±0.0064	0.0348±0.0128
	Symbolic	0.0132±0.0063	0.0250±0.0092	0.0258±0.0093	0.0170±0.0062	0.0172±0.0052	0.0219±0.0081
	Higgs	0.0166±0.0099	0.0367±0.0211	0.0372±0.0221	0.0216±0.0114	0.0188±0.0063	0.0314±0.0161
	MOHA	0.0133±0.0063	**0.0248±0.0091**	0.0258±0.0092	0.0171±0.0064	0.0162±0.0051	**0.0211±0.0078**
	State Model	**0.0131±0.0064**	0.0249±0.0092	**0.0257±0.0093**	**0.0169±0.0061**	**0.0161±0.0051**	**0.0211±0.0078**

TABLE V
TESTING DATA ERROR IN NGSIM DATASETS: IDM MODEL

Mean ± Std.		IDM					
		I80-1	I80-2	I80-3	US101-1	US101-2	US101-3
RMSE (m/s^2)	Gross	1.0917±0.8706	1.4327±0.3938	1.6060±0.4151	1.6334±0.4064	1.4801±0.2717	1.3180±0.2793
	Symbolic	0.9857±0.4282	1.3610±0.4298	1.5341±0.3654	1.5563±0.2550	1.3875±0.1992	1.2964±0.2524
	Higgs	1.0679±0.7976	1.3871±0.3972	1.5860±0.4262	1.5862±0.2578	1.4594±0.2041	1.3025±0.3390
	MOHA	**0.9798±0.4340**	1.3280±0.3908	**1.5289±0.3659**	**1.5555±0.2567**	**1.3634±0.1992**	**1.2944±0.2497**
	State Model	1.0174±0.4718	**1.3254±0.3810**	1.5332±0.3842	1.5583±0.2510	1.3966±0.2693	1.2971±0.2690
RE (m/s)	Gross	0.0799±0.2204	0.0360±0.0108	0.0338±0.0191	0.0419±0.0936	0.0514±0.0916	0.0572±0.1030
	Symbolic	0.0159±0.0083	0.0262±0.0107	0.0265±0.0102	0.0180±0.0080	0.0185±0.0071	0.0293±0.0104
	Higgs	0.0468±0.0335	0.0302±0.0734	0.0265±0.0507	0.0355±0.0689	0.0210±0.0117	0.0481±0.0713
	MOHA	**0.0152±0.0086**	**0.0261±0.0109**	0.0264±0.0102	**0.0180±0.0080**	**0.0179±0.0070**	**0.0239±0.0104**
	State Model	0.0153±0.0082	**0.0261±0.0109**	**0.0264±0.0101**	0.0181±0.0081	0.0182±0.0071	0.0241±0.0106
AE (m/s)	Gross	0.0263±0.0119	0.0297±0.0093	0.0410±0.0238	0.0174±0.0094	0.0171±0.0072	0.0285±0.0122
	Symbolic	0.0137±0.0065	0.0240±0.0095	0.0253±0.0098	0.0160±0.0057	0.0162±0.0049	0.0210±0.0077
	Higgs	0.0234±0.0137	0.0308±0.0184	0.0385±0.0295	0.0175±0.0089	0.0170±0.0059	0.0267±0.0135
	MOHA	**0.0135±0.0067**	**0.0237±0.0094**	**0.0251±0.0098**	**0.0159±0.0057**	**0.0157±0.0049**	**0.0200±0.0077**
	State Model	0.0139±0.0068	**0.0237±0.0094**	**0.0251±0.0098**	0.0160±0.0057	0.0160±0.0052	0.0201±0.0077
ME (m/s)	Gross	0.0289±0.0173	0.0343±0.0094	0.0431±0.0166	0.0187±0.0106	0.0192±0.0112	0.0346±0.0130
	Symbolic	0.0142±0.0067	0.0245±0.0094	0.0254±0.0093	0.0165±0.0060	0.0173±0.0053	0.0216±0.0079
	Higgs	0.0233±0.0131	0.0328±0.0201	0.0385±0.0210	0.0190±0.0102	0.0172±0.0061	0.0360±0.0150
	MOHA	**0.0139±0.0069**	**0.0243±0.0095**	**0.0252±0.0093**	**0.0165±0.0060**	**0.0163±0.0053**	**0.0210±0.0079**
	State Model	0.0141±0.0067	**0.0243±0.0095**	**0.0252±0.0093**	**0.0165±0.0060**	0.0166±0.0055	**0.0210±0.0079**

TABLE VI
SUMMARY OF IMPROVEMENT IN EACH DATASET: HELLY MODEL (IN PERCENTAGE)

	I80-1	2	3	US101-1	2	3
RMSE	7.57	6.71	5.31	2.84	12.7	3.50
RE	84.52	4.68	20.65	58.22	43.49	45.90
AE	13.51	5.45	37.86	40.29	21.61	38.91
ME	27.72	9.96	20.85	41.24	26.03	39.37

TABLE VII
SUMMARY OF IMPROVEMENT IN EACH DATASET: IDM MODEL (IN PERCENTAGE)

	I80-1	2	3	US101-1	2	3
RMSE	10.25	7.31	4.80	4.77	7.88	1.79
RE	80.98	27.50	21.89	57.04	65.18	58.22
AE	48.67	20.20	38.78	8.62	8.19	29.82
ME	51.90	29.15	41.53	11.76	15.10	39.31

because the control algorithm of an ACC focuses more on mathematical modeling rather than driving behaviors or habits [19]. A valid car-following itself can be used as a controller which mimics real drivers' behaviors.

A human-like ACC system is learned using MOHA from a real car-following training dataset. The speed error of simulated traces and the real ones is evaluated in the testing dataset. The generation steps are as follows: 1). The subject vehicle starts from the initial state. 2). The speed, relative speed, and relative distance are computed online. Note that we only control the following vehicle, i.e., the trajectory of the leading vehicle is from the dataset. 3). The current cluster of the subject vehicle is determined by its current state using the online inference discussed in Section III-B, and then the parameter of car-following model is selected to generate the desired acceleration. 4). The status of the subject

TABLE VIII
COMPARISON OF RUNTIME

Models	Symbolic labeling (s)	Automata learning (s)	Clustering (s)	Training (s)	Testing (s)	Total (s)
Gross	-	-	-	488.24	3.84	492.08
Symbolic	69.72	-	53.75	2653.35	53.98	2830.80
Higgs	-	-	832.89	1534.62	33.95	2401.46
Proposed	69.72	16.09	24.56	2054.52	14.41	2179.30
State model	69.72	16.09	-	1690.41	22.36	1798.58

TABLE IX
COMPARISON OF SIMULATED TRAJECTORY

Indicators	Proposed	Gross	PID controller
RE (m/s)	**0.1298±0.0861**	0.1558±0.1156	0.2466±0.2852
AE (m/s)	**0.0968±0.1053**	0.1320±0.1322	0.1105±0.0875
ME (m/s)	**0.0882±0.0869**	0.1197±0.1175	0.1360±0.0973

vehicle, including speed, relative speed, and relative distance is continuously updated online using the acceleration computed in the last time step as well as the information of the leading vehicle from the dataset. This approach is compared with a standard PID controller. The results of comparing speed error in Table IX show that the proposed model outperforms others.

VI. CONCLUSION

In this paper, a novel hybrid model called MOHA is learned from numeric car-following data. The model is easily visualizable and interpretable for the study of car-following behaviors. Experiments demonstrate that MOHA achieves high model fitting accuracy. Besides the general usage in traffic simulation, the proposed model can be used for subject drivers' decision-making by recognizing or predicting surrounding vehicles' car-following states and designing a more human-like car-following controller.

The imperfections of the proposed method include two aspects. First, compared with classic method, the proposed model has higher complexity, though all processing steps can be automated. Second, from safety's perspective, a data-driven design of ACC system lacks theoretical guaranty, because it might be learned from poorly skilled drivers, though the proposed model is indeed an averaging model learned from thousands of human drivers.

In the near future, the complexity problem will be addressed using model selection and model abstraction from formal methods domain. The safety problem of learning-based model can be overcome by using hybrid model checking. Car-following is a relatively simple driving scenario. We also have some undergoing work on applying automata learning lens for complex driving scenarios like lane change intention prediction using non-deterministic automata, social behavioral interaction using timed automata and game theory, etc.

ACKNOWLEDGMENT

The author would like to acknowledge that this manuscript uses some materials presented in IFAC 2017 [1] and IAVSD 2017 [2].

REFERENCES

[1] Y. Zhang, Q. Lin, J. Wang, and S. Verwer, "Car-following behavior model learning using timed automata," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 2353–2358, 2017.

[2] Y. Zhang, Q. Lin, J. Wang, S. Verwer, and J. Dolan, "A data-driven behavior generation algorithm in car-following scenarios," in *Proc. 25th Int. Symp. Dyn. Vehicles Roads Tracks (IAVSD)*, 2017, pp. 227–232.

[3] L. A. Pipes, "An operational analysis of traffic dynamics," *J. Appl. Phys.*, vol. 24, no. 3, pp. 274–281, 1953.

[4] W. Helly, "Simulation of bottlenecks in single-lane traffic flow," in *Proc. Symp. Theory Traffic Flow*. New York, NY, USA: Elsevier, 1959, pp. 207–238.

[5] D. C. Gazis, R. Herman, and R. W. Rothery, "Nonlinear follow-the-leader models of traffic flow," *Oper. Res.*, vol. 9, no. 4, p. 545–567, 1961.

[6] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 62, no. 2, p. 1805, 2000.

[7] S. Hoogendoorn, S. Ossen, and M. Schreuder, "Empirics of multianticipative car-following behavior," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1965, no. 1, pp. 112–120, 2006.

[8] C. P. I. J. van Hinsbergen, W. J. Schakel, V. L. Knoop, J. W. C. van Lint, and S. P. Hoogendoorn, "A general framework for calibrating and comparing car-following models," *Transportmetrica A, Transport Sci.*, vol. 11, no. 5, pp. 420–440, 2015.

[9] B. Higgs and M. Abbas, "Segmentation and clustering of car-following behavior: Recognition of driving patterns," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 81–90, Feb. 2015.

[10] S. Verwer, M. de Weerdt, and C. Witteveen, "Learning driving behavior by timed syntactic pattern recognition," in *Proc. 22nd Int. Joint Conf. Artif. Intell. (IJCAI/AAAI)*, 2011, pp. 1529–1534.

[11] Y. Sakakibara, "Recent advances of grammatical inference," *Theor. Comput. Sci.*, vol. 185, no. 1, pp. 15–45, 1997.

[12] S. E. Verwer, "Efficient identification of timed automata: Theory and practice," Ph.D. dissertation, Dept. Softw. Technol., Delft Univ. Technol., Delft, The Netherlands, 2010.

[13] U.S. Department of Transportation. (2007). *NGSIM—Next Generation Simulation*. [Online]. Available: http://www.ngsim.fhwa.dot.gov

[14] C. Goutte, P. Toft, E. Rostrup, F. Å. Nielsen, and L. K. Hansen, "On clustering fMRI time series," *NeuroImage*, vol. 9, no. 3, pp. 298–310, 1999.

[15] W. W. Cohen, P. Ravikumar, and S. E. Fienberg, "A comparison of string metrics for matching names and records," in *Proc. KDD Workshop Data Cleaning Object Consolidation*, vol. 3, 2003, pp. 73–78.

[16] R. Nippold and P. Wagner, "Calibration of car-following models with single- and multi-step approaches," in *Proc. Winter Simulation Conf.*, 2012, Art. no. 410.

[17] A. Kesting and M. Treiber, "Calibrating car-following models by using trajectory data: Methodological study," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2088, no. 1, pp. 148–156, 2008.

[18] C. Chen, L. Li, J. Hu, and C. Geng, "Calibration of MITSIM and IDM car-following model based on NGSIM trajectory datasets," in *Proc. IEEE Int. Conf. Veh. Electron. Safety (ICVES)*, Jul. 2010, pp. 48–53.

[19] T. Hiraoka, T. Kunimatsu, O. Nishihara, and H. Kumamoto, "Modeling of driver following behavior based on minimum-jerk theory," in *Proc. 12th World Congr. ITS*, 2005, p. 3416.

Qin Lin is currently pursuing the Ph.D. degree with the Department of Intelligent Systems, Delft University of Technology. His research interests include machine learning, time series data mining, and syntactic pattern recognition.

Yihuan Zhang is currently pursuing the Ph.D. degree with the Department of Control Science and Engineering, Tongji University. His research interests include environment perception and decision making in area of autonomous vehicles.

Sicco Verwer received the Ph.D. degree from the Delft University of Technology. He is an Assistant Professor of computer science with the Delft University of Technology. His research interests include the theory and practice of machine learning and state machine learning in particular. His interests within this focus area are diverse. He has published papers on learning state machines, discrimination-aware data mining, software testing, fraud detection, mechanism design, and combinatorial solvers in machine learning. In particular, he is interested in machine learning algorithms resulting in models that are useful for subsequent tasks, such as visualization, testing, verification, data integration, control, and optimization.

Jun Wang (S'98–M'03–SM'12) received the Ph.D. degree in control engineering from the University of Leeds, U.K., in 2003. He has been a Professor of control engineering with the Department of Control Science and Engineering, Tongji University, since 2010, the Head of the Department since 2014, and the Vice Dean of the College of Electronics and Information Engineering since 2016. His research interests include smart sensing and intelligent control in the areas of autonomous vehicles and renewable energy. He is also a member of the Operational Committee for Education in Chinese Association of Automation. He was a Guest Editor of the *International Journal of Vehicle Design*. He is on the Editorial Board of *The International Journal of Driving Science*.