2021 IEEE Intelligent Transportation Systems Conference (ITSC)
Indianapolis, USA. September 19-21, 2021

Real-Time Prediction of Multi-Class Lane-Changing Intentions based on Highway Vehicle Trajectories

Anuj Abraham*, Yi Zhang*, Shitala Prasad

Abstract— **For fully automated driving in the real-world, safety, comfort, and reliability are the key essentials during operation. However, in the automotive industry, it is critical to identify the lane change intention of vehicles from the adjacent lane for safe driving. Thus, the intelligent assistance system should be smart enough to track such lane change desires and assist the driver to avoid any accidents that is caused by human errors on the road. Here, in this paper we propose to predict the driving behaviors of lane-changing vehicles on the highways by studying their chronology. Machine learning performs well in retaining the human-vehicle behavioral patterns and therefore, we involved random forest to do this prediction. We proposed new feature space for lane change prediction, where a really small window size of 3-seconds can lead to high-performance. The extensive experiments are carried out on real-traffic dataset from Next Generation SIMulation (NGSIM) where we set up a new state-of-the-art accuracy of 98.6%.**

Index Terms – Lane change; machine learning; random forest; intention prediction; autonomous vehicles; NGSIM.

I. Introduction

The continuing evolution of the technologies is promoting the development of the driving system to more higher automation [11]. The autonomous vehicle (AV) requires a complex system integration, which involves the hardware configuration design, software architecture design and the final critical safety assurance framework [8][6]. Among all these modules, the quality of the car perception, a module to make sense of environments and car-self, is the pre-requisite to determine the performance of all other modules to make an accurate, safe and correct driving decision, which is also the focus of this paper to predict the vehicles' lane-changing and keeping maneuvers on highway analysis. In a lane change crash environment, two vehicles move on parallel paths, one intentionally changes the lane and collides with the other vehicle. This is the most critical application as multiple vehicles may collide during lane change and the safety aspect is involved. Prediction in real-time and taking safe decisions in quick time is one of the most challenging problems within the transport system. In the past few years, prediction of the vehicle's lane-changing intention using learning algorithms has been of interest to researchers.

The research on two-lane change prediction approaches was carried out and tested using the NGSIM dataset [1]. The model uses a 20-second window to predict the lane-changing maneuvers with each 10 second before and after the crossing moment of the lane mark, however, the need for the latter 10-second information after the crossing moment prevents the implementation of the lane-changing intention prediction in real-time. Studies were also carried on timely detection of lane-changing maneuvers, where classification results show that lane changes can efficiently be predicted in real-time [13]. This paper proposes the density-based clustering to avoid efforts on labeling data before the learning of two maneuvers: lane-changing and lane-keeping. In the earlier research, lane-changing intention prediction is mainly formulated as a classification problem, and solved by support vector machine (SVM).This is because of its powerful kernel converting from a low dimensional space to a high dimensional space transform, which is consistent with the drivers' highly non-linear behaviors. Both [14] and [10] adopt the concept of sliding window. The simulation results in [14], indicate that 87% of all true positives within the first 0.3 seconds from the start of the maneuver. While in [10], a combination of SVM and Bayesian filters were applied to train the model meanwhile improve the prediction reliability. Only four major characteristics, steer angle, lateral positions and their respective first derivatives, are used for input features. The performance of the system shows that the model can accurately predict the lane changes before 1.3 seconds.

In this paper, we proposed a new feature space $\mathcal{X}^d$, where $d \in \mathbb{R}$ denotes the space dimension. The space $\mathcal{X}$ is defined *via* vehicle ID, global time, local positioning (x and y coordinates), velocity, acceleration and finally the lane ID. We experimented with these dimensions on NGSIM dataset with various different size windows to outcome the best possible solution for lane-changing prediction. A random forest (RF) is used as an smart assistance to build up this feature space for computing accurate and efficient prediction with the minimal time frame.

In a nutshell, the major contributions of this paper are three-fold:

- We explored different window sizes and look-back times to generate a new feature space for computing real-time lane-changing predictions using machine learning algorithms with high accuracy.
- Unlike [1], our feature space is defined using the past data only, which makes the real-time prediction implementable. Meanwhile, we still maintain a high accuracy even if the acquired information only before the changing moment. Noticeably, our vector size is roughly 7 times smaller than that of Benterki *et al.* [1].
- Further, we investigated the proposed approach on NGSIM dataset and compared to existing methods, our

Anuj Abraham, Yi Zhang and Shitala Prasad are affiliated with the Institute for Infocomm Research (I2R), Agency for Science, Technology and Research (A*STAR), Singapore 138632. Emails: {anuj_abraham,zhang_yi, shitalap}@i2r.a-star.edu.sg. *Contributions are equal.

2021 IEEE International Intelligent Transportation Systems Conference (ITSC) | 978-1-7281-9142-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ITSC48978.2021.9564738

method outperforms the others by adopting RF. We, additionally, tested our method with different learning algorithms such as Bayes network, SVM, multilayer perceptron, radial basis function (RBF), k-NN and rule-based decision table. Although SVM is highly recommended and widely used by other researchers for lane-changing prediction, our extensive comparison with other different ML algorithms, RF proves to be the best outperforming solution in terms of accuracy and computation cost.

The paper is organized as follows. The specific descriptions of the proposed approach to predict multi-class lane changing and keeping maneuvers are introduced in Section II. This section includes information on the feature extraction and learning predictor, followed by simulation results in Section III. Lastly, the paper is concluded with future scope in Section IV.

II. Proposed Approach to Predict Multi-Class Lane Changing and Keeping Maneuvers

In this section, we elaborately discussed the feature extraction method on the NGSIM dataset and illustrated the way to select the time window before any lane-changing moments. Next, we explained the learning predictor using random forest.

A. Feature Extraction

In this paper, we adopt Next Generation Simulation (NGSIM) US Route 101 dataset [7] to train and test our algorithms. The studied road segment is on the southbound of the US 101 in Los Angeles, CA, and its length is roughly 640 meters with 5 major lanes. 8 cameras, mounted at the top of the a 36-story building adjacent to the freeway, are responsible for recording the video of 8 different sub-segments of the whole studied area, respectively. The total recording duration is 45 minutes covering from the uncongested period, congestion developing period towards the congested period. There are 18 attributes collected in the original dataset, and we reserve 7 attributes to prepare for our following features' calculation. The detailed information of these 7 attributes are listed as follows:

- Vehicle_ID: Vehicle identification number.
- Global_Time: Elapsed time.
- Local_X: Lateral (X) coordinate of the front center of the vehicle with respect to the left-most edge of the section in the direction of travel.
- Local_Y: Longitudinal (Y) coordinate of the front center of the vehicle with respect to the entry edge of the section in the direction of travel.
- v_Vel: Instantaneous velocity of vehicle.
- v_Acc: Instantaneous acceleration of vehicle.
- Lane_ID: Current lane position of vehicle. Lane 1 is left-most lane, and lane 5 is right-most lane. Lane 6 is the auxiliary lane between Ventura Boulevard on-ramp and the Cahuenga Boulevard off-ramp. Lane 7 is the on-ramp at Ventura Boulevard, and Lane 8 is the off-ramp at Cahuenga Boulevard.

Features embedded into the following algorithms are calculated based on the above 7 attributes, and a time window of obtained features are combined together for maneuver predictions. Fig. 1 are drawn to clearly illustrate the time window we selected for the prediction at k. We have three types of maneuvers to predict: Lane-changing right (LCR), Lane-changing left (LCL) and Lane-keeping (LK), and our goal is to predict which maneuver will happen at future time k: If the lane-changing instant happens at time k, all feature information from previous time slot, such as $k-m$ and time $k-n$, will be collected to prepare for training, where m and n are integers. On the other hand, if the vehicle still keep at current lane at time k, then features in the same selected window will be used for predicting LK at k. Both window size, $m-n+1$, and the look-back time n can be adjusted accordingly.

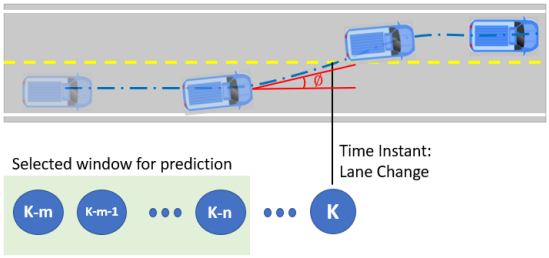


Fig. 1: Illustration of selected time window before lane-changing moment

We use vector $\mathbf{X}_i(k)$ to denote the features of vehicle i at time k, and each vector $\mathbf{X}_i(k)$ is composed by 7 features we calculated based on the above mentioned 7 attributes, shown as follows:

$$\mathbf{X}_i(k) = [\phi_i(k), V_i^x(k), V_i^y(k), A_i^x(k), A_i^y(k), D_i^x(k), D_i^r(k)] \quad (1)$$

For the definition of each feature, details can be found below:

- $\phi_i(k)$: the angle at k between the y-axis perpendicular to the entry edge of the section in the direction of travel and the direction of the car is pointed, shown in Fig. 1.

$$\phi_i(k) = \arctan\left(\frac{|Local\_X_i(k) - Local\_X_i(k-1)|}{|Local\_Y_i(k) - Local\_Y_i(k-1)|}\right) \quad (2)$$

- $V_i^x(k)$ and $V_i^y(k)$: lateral and longitudinal velocities, namely, the vehicle velocity on x-axis and y-axis at k, respectively.

$$\begin{aligned} V_i^x(k) &= v\_Vel_i(k)\sin(\phi_i(k)) \\ V_i^y(k) &= v\_Vel_i(k)\cos(\phi_i(k)) \end{aligned} \quad (3)$$

- $A_i^x(k)$ and $A_i^y(k)$: lateral and longitudinal accelerations, namely, the vehicle acceleration on x-axis and y-axis at k, respectively.

$$\begin{aligned} A_i^x(k) &= v\_Acc_i(k)\sin(\phi_i(k)) \\ A_i^y(k) &= v\_Acc_i(k)\cos(\phi_i(k)) \end{aligned} \quad (4)$$

- $D_i^l(k)$: the distance between the ith car position and its closet left lane marking at k.

- $D_i^r(k)$: the distance between the ith car position and its closet left lane marking at k.

Accordingly, the time series for vehicle i can be illustrated as follows:

$$\mathcal{X}(i) = [\mathbf{X}_i(k-m), \mathbf{X}_i(k-m+1), \ldots, \mathbf{X}_i(k-n)] \quad (5)$$

By selecting two features (lateral velocity and lateral acceleration), Fig. 2 is drawn to illustrate the different characteristics of 3 classes for the selected features. Fig. 2a depicts the data points under the axes of the features acceleration and velocity for 3 classes, and the selected data points all in a 5-second time range before the lane-changing moment. Obviously, data points under class LK are relatively concentrated to the original point, on the other hand, data points under classes LCL and LCR are distributed dispersedly. Fig. 2b and Fig. 2c show the data points capturing lateral acceleration and velocity at different time instants, in other words, the change of the lateral acceleration and velocity with the increase of the time, respectively. The 5-second time segment is selected just before the lane-changing moment for classes LCL and LCR in two figures, also, the same duration of the time segment is selected if the next time step is lane-keeping to illustrate class LK. Clearly, when closing to the lane-changing moment, the range of both acceleration and velocity under classes LCL and LCR is larger than that of LK.

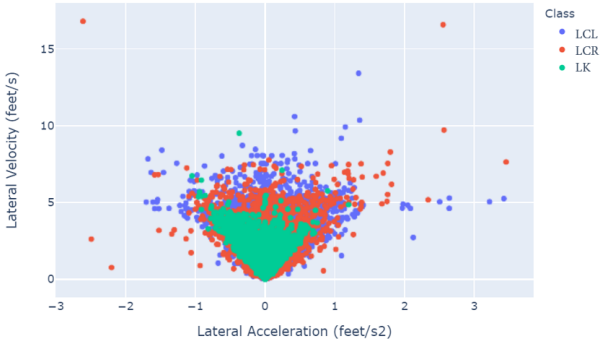
Compared with the lane-keeping cases, the lane-changing cases are quite limited. To solve this issue, we apply the moving window strategy on each trajectory of the vehicle to generate more examples. As the time granularity of Global_Time after changing to date time in NGSIM dataset is 0.1 second, we can obtain 10 examples for each case under unit second. Accordingly, the definition of the time series for vehicle i with look-back time u and window size w is illustrated as follows:

$$\begin{aligned}&\mathcal{X}_u^w(i) = \\ &[\mathbf{X}_i(k-(wf+t-1)-uf), \mathbf{X}_i(k-(wf+t-2)-uf), \\ &\ldots, \mathbf{X}_i(k-t-uf)], \quad \text{for}\,\{\forall t: 1 \le t \le f\}\end{aligned} \quad (6)$$

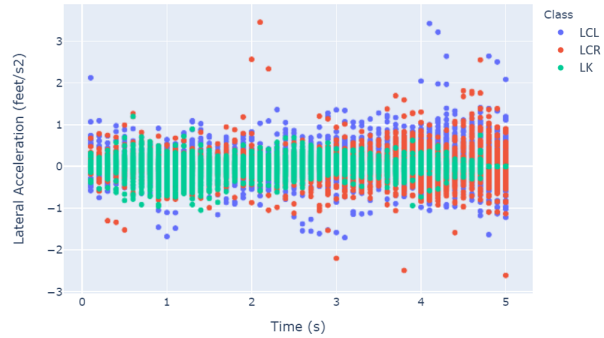
where u and w are the look-back time and window size under the unit of the second, respectively. f is the frame frequency of the dataset, for our case, if $\triangle$ is the time series step, then $\triangle = \frac{1}{f} = 0.1$ second. t is the integer between 1 and f. By doing so, we obtain 2800 examples for each maneuver under each case listed in Table I. We consider training on 9 cases to investigate the impacts of the window size w and the look-back time u, and the corresponding feature sizes or vectors (fv) of each example are 70, 140 and 210 respectively for cases 1-4-7, 2-5-8 and 3-6-9. Also, 80% and 20% of examples are used for training and testing, respectively.

B. Learning Predictor

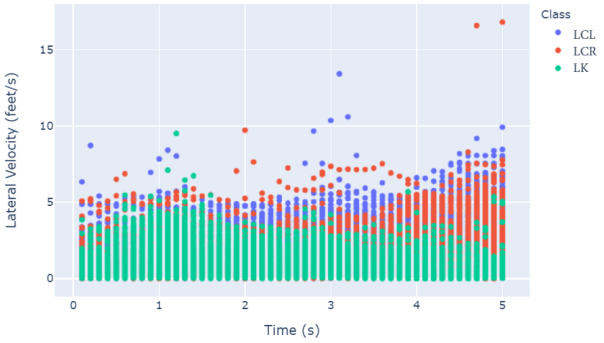
Random forest has been successfully involved in many practical problems such as air quality prediction [9], chemoinformatics [15], ecology [5], object recognition [18], bioinformatics [16], whether prediction [4] and traffic control



(a) Scatter plot for lateral acceleration and lateral velocity



(b) Vehicle lateral acceleration according to the elapsed time



(c) Vehicle lateral velocity according to the elapsed time

Fig. 2: Data analysis for lateral acceleration and lateral velocity under 3 classes

[12]. Thus, in this paper we incorporate random forest [17] for lane prediction as the performance in such fields is outstanding. In general terms, RF is an ensemble method using tree-based classifiers $\{\chi(x, h*k)|k = \{1, 2, \ldots\}\}$, where the $h*k$ are independent identically distributed random vectors and x is an input pattern to RF. In training, RF algorithm creates multiple CART-like trees [3], [2], each trained on a bootstrapped sample of the original training data $\mathcal{D}$ and searches only across the randomly selected subset of the input variables to determine a split.

Let say, a classifier χ_{h_k} built on k-th training dataset $\mathcal{D}_k$ with total n instances in whole dataset $\mathcal{D}$. We define RF accuracy for $d_i \in \mathcal{D}$ as:

$$\mathcal{A}(\chi_{h_k}) = \frac{\sum_{i=1}^{n} I(h_k(d_i) = y_i; d_i \notin \mathcal{D}_k)}{\sum_{i=1} I(d_i \notin \mathcal{D}_k)} \quad (7)$$

TABLE I: Parameters settings for different cases

	Case 1	Case 2	Case 3
Window Size w (s)	1	2	3
Look-back Time u (s)	0	0	0
	Case 4	Case 5	Case 6
Window Size w (s)	1	2	3
Look-back Time u (s)	1	1	1
	Case 7	Case 8	Case 9
Window Size w (s)	1	2	3
Look-back Time u (s)	2	2	2

TABLE II: Generated dataset description

Dataset	#Classes	#Train	#Test	Feature vector #fv
NGSIM	3 (LCL, LCR, and LK)	6720	1680	70, 140 and 210 for cases 1-4-7, 2-5-8 and 3-6-9

where $I(\cdot)$ is the indicator function and $Y = \{y_1, y_2, \dots y_i\}$ is the class feature space. The larger $\mathcal{A}$, the better the tree.

For classification, each tree in RF casts a unit vote for the most popular class at input x and the output of classifier χ is determined by the majority vote of the trees t_i. By limiting the number of trees, the computational complexity of the algorithm is controlled and also the correlation between trees are decreased. As a result, RF can handle really high dimensional data and use a large number of trees. According to [2], RF computational time is computed as:

$$\chi_{cost} = c \times n \sqrt[2]{MN} \log(N) \tag{8}$$

where c is the controlling factor, n is the number of trees in χ and (M, N) are the number of variables and number of samples in $\mathcal{D}$. Note, that even though RF is not computationally expensive yet the memory consumption is huge as they need $N \times n$. However, the advantages of RF are: (1) it runs efficiently on large datasets such as traffic data, (2) it handles large amount of variables and generates unbiased estimation of generalization error, (3) it is relatively robust to noise and locate outliers and (4) it has less computational complexity. The discussions on the detailed usage are given in next Section.

III. Experiments and Results

In this section, we empirically demonstrate the proposed RF based real-time prediction of multi-class lane-changing performance on benchmark NGSIM US Route 101 dataset [7] and compare with the state-of-the-art methods. The generated NGSIM dataset description used in this paper is listed in Table II. In this study, out of 8400 examples (2800 examples for each maneuver class) extracted, of which 80% are used for training and remaining for testing. The corresponding feature vector (fv) of each example are 70, 140 and 210 respectively for cases 1-4-7, 2-5-8 and 3-6-9.

A. Results

RF classifier was applied on NGSIM dataset. The most conventional statistical classification methodologies, such as Bayesian and machine learning (ML) algorithms like artificial neural networks (ANN) and support vector machines (SVM), create complex rules for decision making, which is based on many complex variables in feature space. However, RF classifiers using trees label pixels consider one or several variables. In contrast with other ML methods, RF further adds prediction variables M which are used in each nodes for tree growth. In general, RF needs only the number of classification trees n and M. In each node, a subset of M ranging from 1 to *max* has to be specified. The experimental results for NGSIM dataset using RF classification are given in Table III. The RF outperforms in terms of overall accuracy compared to other state-of-the-art methods (see Table III) and found that the proposed features actually boost the achievement. The production grows to 98.6% using RF which is respectively 0.3% and 2.9% higher than ANN and SVM [1], even though the feature vector is reduced by 86.8% ($(1600 - 210)/1600 \approx 86.8\%$). Table III underlines the feature vector size used for classification and it is clear that the proposed vector indicates not only the high accuracy performance under RF but also less computational efforts due to small feature vector, when compared to the SOTA methods [1].

TABLE III: Comparative study of improvements in proposed with SOTA methods

Methods	ML algorithms	Accuracy (%)	Window size (s)	#fv
Benterki *et at.* [1]	SVM	95.7	20	1600
	ANN	98.3	20	1600
Proposed	kNN (Case 3)	98.5	3	210
	RF-100 trees (Case 3)	98.6	3	210

In next set of experiments, we evaluated the proposed features with different ML methods. The results for Bayes network, multi-layer perceptron, SVM, *k*-NN, decision tree and RF are shown in Table IV. In this comparison, we perform rigorous training and testing on all different ML methods with different feature dimensions, *i.e*, different cases are of different dimensions, as defined in Table I. As we can see, the best performance is achieved by RF (**bold**) for all the cases 1-9, whereas *k*-NN is the second best performing ML. For Case 3, *k*-NN is very close to RF, with a difference of 0.1%. That is, the lazy algorithm is marginally close to RF for cases 3 and 6. It is because for cases 3 and 6, $\mathcal{X}^{d=210}$ is comparatively larger than cases 1-2, 4-5, and 7-8. As a key property of *k*-NN, it keeps all the training sample unforgottenly and can perform very well when they are in right proportion.

We further analyzed that the accuracy $\mathbb{A}$ is also affected by different types of ML algorithms. In our case, on this dataset, the best performing ML is RF followed by *k*-NN, multi-layer perceptron, Bayes, decision tree and lastly SVM. Unlike [1], in our case SVM did not outperform other methods and just managed to achieve 88.6% accuracy for Case 1, which is the best among other cases in SVM (see cases 2-9 in Table IV for SVM).

TABLE IV: Accuracy comparison of NGSIM datasets on different SOTA methods

Learning algorithms		Accuracy (%)								
		Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9
Bayes network		90.175	91.1171	90.7806	82.5034	83.9838	85.2624	77.7927	79.1386	80.4845
Linear SVM		88.6	84.9933	83.58	84.253	84.3876	85.7335	83.109	84.4549	83.3782
Multi-layer perceptron		95.0202	95.895	97.712	89.9058	79.8789	94.4145	82.0996	87.2813	90.9825
RBF		87.8197	86.9448	86.9448	73.8896	74.5626	75.5047	70.1884	71.5343	71.1306
kNN		93.6743	97.3082	**98.4782**	88.0888	95.0875	98.183	86.4065	95.0202	97.3755
Rule based-Decision Table		87.8869	86.2719	87.0794	82.1669	83.1763	82.4361	78.4657	79.8116	81.4266
Random Forest	100 trees	96.8371	98.0485	**98.5868**	95.6258	96.6353	97.6447	94.3472	96.7699	97.7793
	10 trees	95.9623	96.6353	96.7699	93.8089	95.8277	96.4334	92.7995	94.0781	96.7026
	5 trees	95.4913	96.1642	96.3661	92.7995	94.2799	94.9529	90.646	92.5976	94.2127

TABLE V: Computation cost comparison of NGSIM datasets on different SOTA methods

Learning algorithms		Time taken to build the model (sec)								
		Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9
Bayes network		0.8	1.47	2.2	0.9	1.9	2.89	0.7	1.36	2.71
Linear SVM		51.64	142.18	207.98	51.25	139.46	203.66	53.44	132.11	215.88
Multi-layer perceptron		198.56	729.15	1611.76	200.05	719.03	1595.52	199.96	729.18	1617.81
RBF		2.8	4.61	8.59	2.51	7.96	7.35	3.08	5.7	11.64
kNN		0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Rule based - Decision Table		4.8	7.94	12.94	5.62	11.16	17.14	7.61	17.99	30.83
Random Forest	100 trees	11.6	14.39	16.01	14.01	16.27	16.3	16.57	20.22	17.79
	10 trees	1.18	1.5	1.61	1.42	1.64	1.66	1.47	1.65	2.12
	5 trees	0.59	0.79	0.83	0.71	0.83	0.8	0.73	0.83	0.83

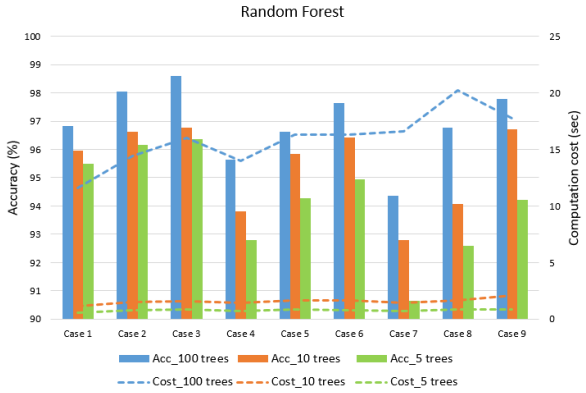


Fig. 3: Bar chart comparison of RF tree on NGSIM dataset

B. Ablation Studies

To further investigate RF algorithm, we evaluate it with different number of classifier trees and growth rates on the NGSIM dataset for traffic lane-changing prediction task and compare for different feature dimension d. Fig. 3 shows a bar chart comparison of 100, 10 and 5 trees. It is clear that the more the tree in RF, the better the performance. Also, we observed that as the feature space dimension d increases, the performance also increases. Thus, we conclude that for our proposed method the best real-time lane-changing prediction is achieved when $w = 3$ and $u = 0$.

Secondly, we observed change in performance boost with tree size. They are approximately directly proportional to each other. That is, when the tree size increases from 5 to 10 to 100, the improvements we achieved in RF are 0.5% and 0.8%, respectively.

Thirdly, we found that RF performs worst when $w = \{1\}$ and $u = \{2\}$ for all RFs in Table IV. The lowest achieved accuracy for RF is 90.7% for 5 trees which is still the best in Case 7 compared to other MLs. This confirms that the proposed feature space is well suitable for lane-changing prediction using RF.

Fourthly, we detail the parameters involved in RF to output the best result. The parameters are shown in Table VI. This table further gives a clear picture of various types error rates to further pinpoint the performance of RF with 100 trees. Next, we shows a detail of testing samples in terms of true positive (TP), false positive (FP), precision, recall, F-measure and ROC area for RF with 100 trees for Case 1 to predict multi-class lane change on NGSIM dataset (see Table VII).

In Table VIII, we also plotted the confusion matrix for all the three classes. We observed that with the proposed feature space, we achieved 98.6% accuracy, where right lane change is bit confused with left lane change in class *b* (LCR). This leave a future space for research.

C. Computation Cost

Lastly, we also conducted experiments over the computational cost involved with various ML algorithms. Table V shows a detailed analysis for the same. The highest computation cost to build the model is by multi-layer perceptron and then after its SVM followed by RF. The lightest is *k*-NN with only 0.01 second, as it does not need to build any complex models for prediction. In RF, the highest computation cost is directly proportional to the number of classification trees involved. RF with 100 trees is costly and RF with 5 trees is the lightest. Specifically, by decreasing tree size from 100 to 5, the accuracy in each case drops by around 2%, however, the computational time reduces by 80%, which is significantly high.

TABLE VII: Detailed accuracy by classes for Case 1

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC area	Class
	0.975	0.023	0.964	0.975	0.969	0.996	**LCL**
	0.911	0.023	0.926	0.911	0.918	0.987	**LCR**
	0.962	0.022	0.964	0.962	0.963	0.994	**LK**
Weighted Avg.	**0.955**	**0.022**	**0.955**	**0.955**	**0.955**	**0.993**	

TABLE VI: Parameters generated for Random Forest - 5 trees: Performance indices for Case 1

Parameters generated	Values
Out of bag error	0.1261
Time taken to build model	0.59 seconds
Correctly Classified Instances	1419 - **95.4913** %
Incorrectly Classified Instances	67 - 4.5087 %
Kappa statistic	0.931
Mean absolute error (MAE)	0.0522
Root mean squared error (RMSE)	0.1546
Relative absolute error (RAE)	11.9626 %
Root relative squared error	33.1146 %
Total number of instances	1486

TABLE VIII: Confusion matrix for Case 1

a	b	c	<– classified as
555	8	6	**a = LCL**
18	326	14	**b = LCR**
3	18	538	**c = LK**

D. Discussion

In this study, as explained in Section 2, we have varied and adjusted the look-back time u and window size w for predicting the multi-class lane-changing intentions. Further, we investigated their impacts on 9 different cases for various feature vector sizes, as explained in Table I.

It is observed that when u is kept constant and w is increased, the accuracy $\mathbb{A}$ for different cases also increases. Whereas, when w is kept constant with increasing u, its accuracy $\mathbb{A}$ decreases for cases 1-4-7, 2-5-8 and 3-6-9.

IV. Conclusion and Future scope

In this paper, we have proposed a real-time prediction strategy for multi-class lane-changing intentions based on highway vehicle trajectories. We proposed seven new features under various window frames and look-back times. We performed different tests on various ML algorithms to clearly understand the efficiency of our proposed features to achieve the highest accuracy. Although we use fewer training samples and low feature dimensions compared to SOTA methods, the performance achieved is a new state-of-the-art on the NGSIM dataset. Our proposed method uses only the past information for prediction with 98.6% accuracy and reduces the feature size significantly.

In future work, we will more focus on windowing function and make the system more accurate by reducing the dimension for a hard real-time system such as for a self-driving car. Furthermore, testing on a long short-term memory (LSTM) model to adopt the recurrent neural network (RNN) structure shall be considered in our future work.

References

[1] Abdelmoudjib Benterki, Moussa Boukhnifer, Vincent Judalet, and Maaoui Choubeila. Prediction of surrounding vehicles lane change intention using machine learning. In *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, volume 2, pages 839–843. IEEE, 2019.

[2] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[3] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[4] Sun Choi, Young Jin Kim, Simon Briceno, and Dimitri Mavris. Prediction of weather-induced airline delays based on machine learning algorithms. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–6. IEEE, 2016.

[5] D Richard Cutler, Thomas C Edwards Jr, Karen H Beard, Adele Cutler, Kyle T Hess, Jacob Gibson, and Joshua J Lawler. Random forests for classification in ecology. *Ecology*, 88(11):2783–2792, 2007.

[6] Xinxin Du, Marcelo H Ang, and Daniela Rus. Car detection for autonomous vehicle: Lidar and vision fusion approach through deep learning framework. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 749–754. IEEE, 2017.

[7] John Halkias and James Colyar. Next generation simulation fact sheet. *US Department of Transportation: Federal Highway Administration*, 2006.

[8] Philip Koopman and Michael Wagner. Challenges in autonomous vehicle testing and validation. *SAE International Journal of Transportation Safety*, 4(1):15–24, 2016.

[9] Dinesh Kumar et al. Evolving differential evolution method with random forest for prediction of air pollution. *Procedia computer science*, 132:824–833, 2018.

[10] Puneet Kumar, Mathias Perrollaz, Stéphanie Lefevre, and Christian Laugier. Learning-based approach for online lane change intention prediction. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 797–802. IEEE, 2013.

[11] Todd Litman. *Autonomous vehicle implementation predictions*. Victoria Transport Policy Institute Victoria, Canada, 2017.

[12] Yunxiang Liu and Hao Wu. Prediction of road traffic congestion based on random forest. In *2017 10th International Symposium on Computational Intelligence and Design (ISCID)*, volume 2, pages 361–364. IEEE, 2017.

[13] Vishal Mahajan, Christos Katrakazas, and Constantinos Antoniou. Prediction of lane-changing maneuvers with automatic labeling and deep learning. *Transportation research record*, 2674(7):336–347, 2020.

[14] Hiren M Mandalia and Mandalia Dario D Salvucci. Using support vector machines for lane-change detection. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 49, pages 1965–1969. SAGE Publications Sage CA: Los Angeles, CA, 2005.

[15] John BO Mitchell. Machine learning methods in chemoinformatics. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 4(5):468–481, 2014.

[16] Yanjun Qi. Random forest for bioinformatics. In *Ensemble machine learning*, pages 307–323. Springer, 2012.

[17] Vladimir Svetnik, Andy Liaw, Christopher Tong, J Christopher Culberson, Robert P Sheridan, and Bradley P Feuston. Random forest: a classification and regression tool for compound classification and qsar modeling. *Journal of chemical information and computer sciences*, 43(6):1947–1958, 2003.

[18] M Zeeshan Zia, Michael Stark, Bernt Schiele, and Konrad Schindler. Detailed 3d representations for object recognition and modeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2608–2623, 2013.