2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

Comparison of rule-based and machine learning methods for lane change detection

Nolwenn Monot, Xavier Moreau, André Benine-Neto, Audrey Rizzo, and François Aioun

Abstract— **This paper provides two methods to detect lane changes of the vehicles around the ego vehicle on highway scenarios. The first method is rule-based with changing lane probabilities over the vehicle distance to the lane and the transverse speed. The second uses neural network with a sliding window of the past trajectory of the vehicle input. Moreover, these methods are compared with a dataset extracted from open road data records with an autonomous vehicle prototype. The results show that the transverse speed as an input of the neural networks has a significant impact on the training and the results and that the rule-based method is slower to detect the lane change than the neural networks.**

I. Introduction

The lane change is the only lateral maneuver a vehicle can make on a motorway, not only it presents a risk of collision [1], but it can also degrade the fluidity of the traffic. In a motorway context, a fully autonomous vehicle needs to analyze the surrounding environment and should be able to detect if another vehicle is changing lane. The detection of a lane change must intervene before the vehicle is actually in the new lane in order to anticipate braking and limit the risk of collision. This way, the autonomous vehicle should be more comfortable and less stressful for the users.

Since this is a critical question, researches have already been conducted on this subject not only to detect a lane change, but also more generally, to interpret the behavior of other vehicles. The majority of the studies can be separated in two categories: probabilistic and artificial intelligence.

Various studies explored the problem with probabilistic approaches. In [2], authors used Bayesian networks to detect the lane change. In [3] Dempster-Shafer theory has been used to predict maneuvers on highway, in which a distinction is made between lane change, cut in and overtaking. [4] and [5] estimate the maneuver of a vehicle with Hidden Markov Models (HMM). [6] also uses HMM but to detect lane change of the ego vehicle in order to model the driver behaviour for active system safety. The distinction among these three probabilistic methods lies in the fact that Dempster-Shefer theory is based on a belief measures whereas the Bayesian network uses probability and the Hidden Markov Model employs probability distribution over sequences of observations. The advantage of the probabilistic methods is that it is easy to understand the cause-effect relationship and the results. However, it can be difficult to express the knowledge in probability distribution.

Another solution to the problem of the lane change detection of the vehicles around the ego vehicle with artificial intelligence is explored in [7] with support vector machine and in [8] with neural networks. Neural networks are interesting because they do not need a model and can represent any linear or non linear function. Yet, the training of a networks needs a representative and considerable database and the analysis of the results can be complex because it is hard to interpret the model.

Other methods include fuzzy logic [9] or case-based reasoning [10], which can be expanded with new cases as the vehicle is driving. In [11], a trajectory is predicted with potential field in order to include the adjacent vehicles that can be the source of lane change abortion and false alarm of lane change detection.

In this paper, two methods (each in one of the main categories) are compared in order to identify lane change behavior of the vehicles around the ego vehicle. One is a rule-based method that uses trajectory prediction in a first step and lane change probabilities over the trajectory prediction in a second step. The other is a neural network classification method.

The contribution of this paper, in the case of the rule-based method is that rules are developed using two criteria: one on the predicted lateral distance to the lane and the other on the lateral speed estimated with a Kalman filter. Using not only the trajectory prediction but also the transverse speed should enable the method to accurately detect lane change in turns and straight roads. Moreover, with the Kalman filter, the transverse speed is more representative and less noisy.

The second method uses a classic neural network. Instead of a more complex Recurrent Neural Network (RNN) to classify the intention of the vehicle, it takes as inputs a sliding window of the previous positions and velocities of the surrounding vehicle and polynomial coefficients, representing the adjacent lanes. Moreover, these two methods are compared with dataset recorded on motorway in real conditions.

The paper is organized as follows. Section II makes the statement of the lane change problem and describes the data used for the detection, sections III and IV present respectively the rule-based and neural network methods and section V compares the two methods with the dataset. Finally

N. Monot is with the IMS Laboratory, Univ. Bordeaux, Bordeaux INP, CNRS, UMR 5218, 33405 Talence, France and also with Groupe PSA, 78943 Velizy-Villacoublay, France (e-mail: name.surname@mpsa.com).

X. Moreau and A. Benine-Neto are with the IMS Laboratory, Univ. Bordeaux, Bordeaux INP, CNRS, UMR 5218, 33405 Talence, France (e-mail: name.surname@ims-bordeaux.fr).

A. Rizzo and F. Aioun are with Groupe PSA, 78943 Vélizy-Villacoublay, France (e-mail: name.surname@mpsa.com).

This work took place in the framework of the OpenLab 'Electronics and System for Automotive' combining IMS Laboratory and Groupe PSA company.

978-1-7281-0322-8/18/$31.00 ©2018 IEEE

the paper is wrapped up with conclusions and perspectives.

II. Problem Statement

A. Scenarios

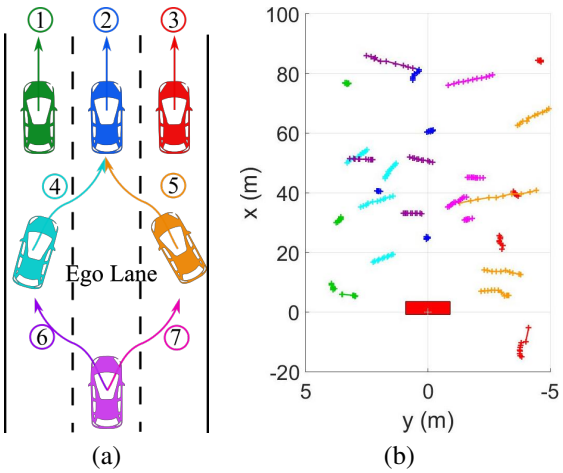
The ego vehicle drives on a divided highway so that all the vehicles in the environment drive in the same direction as the ego. As stated in the introduction, the objective is to detect if a vehicle will change lane or remain in its current lane, so that the longitudinal speed of the ego can be adapted accordingly. Therefore, only the vehicles nearby are considered *i.e* vehicles on the same lane, vehicles at the direct right lane and vehicles at the direct left lane of the ego vehicle. Seven scenarios of lane change for the surrounding vehicles can be defined as described in Table I and illustrated in Fig. 1 (a).

TABLE I
EXO VEHICLES LANE CHANGE SCENARIOS

Number	Initial lane position	Action
1	left	remains in same lane
2	middle	remains in same lane
3	right	remains in same lane
4	left	changes to the middle lane
5	right	changes to the middle lane
6	middle	changes to the left lane
7	middle	changes to the right lane

B. Data used for the detection

The database is created using records of an autonomous vehicle of the Groupe PSA AVA project (Autonomous Vehicle for All) driving on open roads. During the record, the ego vehicle is driving on a divided highway with 2 or 3 lanes and all vehicles driving in the same direction. The ego vehicle can be either on the right, middle or left lane. However, in the samples of records used to create the database for this study, the ego vehicle does not change lane so that it does not impact the intention detection. All the samples of the database are labelled manually to meet the scenarios described in Table I.



(a) (b)

Fig. 1. Lane change scenarios (a) and examples of records (b).

The position of the objects in the environment are recorded using a data fusion of the different sensors equipping the ego vehicle. Other vehicles are positioned in the ego vehicle frame of reference. The camera on board of the ego vehicle represents the lanes as third order polynomials, such that:

$$\begin{aligned} y_R(x) &= C_{0R} + C_{1R}x + C_{2R}x^2 + C_{3R}x^3 \\ y_L(x) &= C_{0L} + C_{1L}x + C_{2L}x^2 + C_{3L}x^3, \end{aligned} \tag{1}$$

with $y_R(x)$ and $y_L(x)$ the lateral position of the right and left lane at a distance x in front of the ego vehicle. C_{iR} and C_{iL} are the i^{th} order coefficients of the two polynomials.

For the scenarios 4, 5, 6 and 7 that correspond to lane changes, the records have been interrupted before the vehicle is on the ego lane in order to not provide too much information and because the objective of these algorithms is to detect a lane change as early as possible. There are not only records on straight roads but also on roads bends to verify whether the algorithms can distinguish a lane change to a turn. Some examples of positions used for the algorithms can be seen on Fig. 1 (b), the color code meets the colors on Fig. 1 (a).

In order to have a fair comparison between the two methods, the data used for the lane change detection are the same. For each object i surrounding the ego vehicle, at an instant k, these data include: the polynomial coefficients of the lanes at the right and left of the ego vehicle; the last 25 samples of longitudinal and lateral positions and speeds, $i.e.$ respectively, $X_i(k)$, ..., $X_i(k-25)$, $Y_i(k)$, ..., $Y_i(k-25)$, $V_{xi}(k)$, ..., $V_{xi}(k-25)$ and $V_{yi}(k)$, ..., $V_{yi}(k-25)$. The sampling period,T_s, is 40 ms, therefore is consists of 1 s sliding window.

III. The Rule-Based Method: Kalman Filtering and Probabilities

The first method described for the lane change detection is the rule-based method. Rules are defined in order to identify if a vehicle is changing lane. These rules are based on the transverse speed of the object V_{yi} and its distance to the lane markings. Probabilities depending on these two parameters are used to estimate the state of the object. The higher the transverse speed and the lower the distance to the lane, the greater the chance of the vehicle changing lane.

However, when V_y is computed using the formula:

$$V_{yi}(k) = \frac{Y_i(k) - Y_i(k-1)}{T_s}, \tag{2}$$

with T_s the sample time, the signal is noisy and not representative of the reality. Then, V_{yi} is estimated using a Kalman filter.

A. Kalman filtering and prediction

The Kalman filter [12] is able to estimate the states of a system using a series of measurement and a dynamic model of the system. Because the only signals available concerning the other vehicles are their relative position to the ego and their velocity, a simple model of evolution is used. These vehicles are considered to have constant longitudinal and

lateral speeds such that the equations describing the evolution of their relative trajectories to the ego are:

$$\begin{cases} X_i(k+1) = X_i(k) + T_s V_{xi}(k) - T_s V_{xego}(k) \\ V_{xi}(k+1) = V_{xi}(k) \\ Y_i(k+1) = Y_i(k) + T_s V_{yi}(k) \\ V_{yi}(k+1) = V_{yi}(k), \end{cases} \tag{3}$$

with $X_i(k)$ and $Y_i(k)$ the longitudinal and lateral position of the object i relative to the ego vehicle position at the instant k, and $V_{xi}(k)$ and $V_{yi}(k)$ the actual longitudinal and lateral speed of the object i at the instant k.

The state space representation of the model can be written:

$$x(k+1) = Ax(k) + Bu(k), \tag{4}$$

with the states vector $x(k) = [X_i(k)\ V_{xi}(k)\ Y_i(k)\ V_{yi}(k)]^T$, the input $u(k) = V_{xego}(k)$ the velocity of the ego vehicle and the matrices:

$$A = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } B = \begin{bmatrix} -T_s \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{5}$$

Kalman filtering is made in two steps, a prediction step and an update. By naming $\hat{x}_{k|k}$ the estimated state and $P_{k|k}$ the error covariance matrix at the time k, the prediction step computes the estimation of the current state using the relations:

$$\begin{aligned} \hat{x}_{k|k-1} &= A\hat{x}_{k-1|k-1} + Bu(k) \\ P_{k|k-1} &= AP_{k-1|k-1}A^T + Q, \end{aligned} \tag{6}$$

with $\hat{x}_{k|k-1}$ the predicted state estimate, Q the covariance of the process noise and $P_{k|k-1}$ the a priori predicted estimate error covariance.

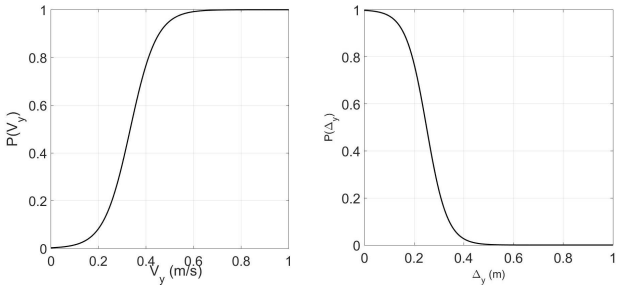
The update step is used to obtain a more accurate estimation using the observations to correct the predicted states:

$$\begin{aligned} \tilde{y}_k &= z_k - H\hat{x}_{k|k-1} \\ S_k &= HP_{k|k-1}H^T + R \\ K_k &= P_{k|k-1}H^T S_k^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k\tilde{y}_k \\ P_{k|k} &= (I - K_k H)P_{k|k-1}, \end{aligned} \tag{7}$$

with z_k the measure, S_k the covariance of the update, R_k the noise covariance, K_k the Kalman gain and $H = I_4$ the observation model.

In order to detect the lane change as early as possible, a step of prediction using the dynamic model is added. The objective is to determine the position of the vehicle at an horizon $t_h = 0.6$s. The definition of t_h is empiric, it is chosen so that the detection can be made early without false detection of lane change. Indeed, if the transverse speed of the vehicle is high, the prediction step may estimate wrongly that the vehicle will be in another lane at $t + t_h$. On the contrary, if $t + t_h$ is too small the detection of the lane change may be late. The estimation of the position at $t + t_h$ is computed such that:

$$\hat{x}_{k|k+t_h/T_s} = A^{t_h/T_s}\hat{x}_{k|k}. \tag{8}$$



(a) (b)

Fig. 2. Lane change probability in function of the transverse speed (a) and the distance to the closest line at t_s (b).

B. Lane change probabilities

Two probabilities are defined in this section and are used in the rules to detect a lane change. These two probabilities, named $P(V_y)$ and $P(\Delta_y)$, estimate respectively the probabilities of a lane change in function of the vehicle transverse speed and its distance to the closest line at $t + t_h$. They are computed using sigmoid functions such that:

$$P(\chi) = \frac{1}{1 + \mathrm{e}^{-\beta(\chi - \alpha)}}. \tag{9}$$

Table II summarizes the values β and α for the probabilities and Fig. 2 shows their plots. The parametrization of the probabilities were chosen after analysing the records of lane change. It can be noted that some vehicles around the ego can change lane with a transverse speed lower than $V_y = 0.4$ m/s and that some vehicles drive close to the ego lane marking. This choice is a compromise between the needs to detect the lane change early and not make false lane change detections.

TABLE II
TABLE OF PROBABILITIES PARAMETERS

$\boldsymbol{P}$	$\boldsymbol{\beta}$	$\boldsymbol{\alpha}$
$P(V_y)$	18	0.33
$P(\Delta_y)$	-24	0.25

C. Rules of the lane change

The rules used to the detect a lane change are the following:

if Between t and $t+t_h$ the vehicle has crossed a line **then**
  The vehicle will change lane
else
  Compute the probabilities $P(V_y)$ and $P(\Delta_y)$
  Compute $P(V_y \cap \Delta_y) = P(V_y)P(\Delta_y)$
  if $P(V_y \cap \Delta_y) > 0.5$ **then**
    The vehicle will change lane
  else
    The vehicle will remains in its lane
  end if
end if

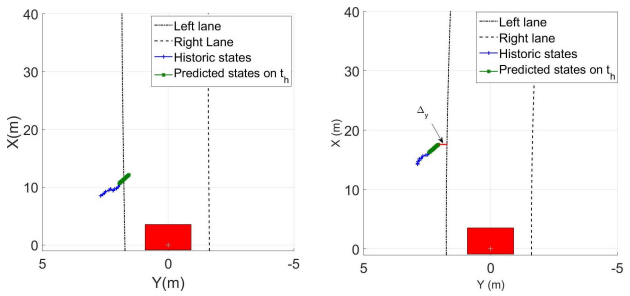


Fig. 3. Examples with (a) and without (b) prediction of a crossing line.

D. Examples

In this section two examples are presented in order to illustrate the method.

Fig. 3 (a) shows the first example. In this situation, the vehicle is on the left lane of the ego vehicle and its past positions in blue suggest that the vehicle is moving to the ego lane. The future predicted states are plotted in green. At $t + t_h$ the vehicle is supposed to cross the line and be in the same lane as the ego. Then, a lane change intention is detected.

The second example, shown on Fig. 3 (b), presents a scenario where the prediction estimates that the vehicle will still be in the same lane at $t + t_h$. The two probabilities are calculated, giving $P(V_y) = 0.9973$ and $P(\Delta_y) = 0.02676$, the intersection gives $P(V_y \cap \Delta_y) = 0.0275$ meaning that a lane change is not detected. In this scenario, the vehicle will indeed change lane but the vehicle is still too far from the line meaning the probability $P(\Delta_y)$ is too low. More samples are needed so that the vehicle lane change is detected. Indeed, at $t+0.2s$, 5 samples later, $P(\Delta_y) = 0.51$ and $P(V_y) = 0.998$ meaning the lane change is detected.

IV. The Machine Learning Method: Neural Network

The second method uses neural networks to determine a lane change as a classification problem.

A. Neural network structure

A neural network is composed of three layers: an input layer, a hidden layer and an output layer. The parameters of the input layer are the 1s past trajectory and speeds and the two lines polynomials coefficients. The hidden layer in the neural network used in this paper contains only one layer of thirty neurons and because there are 7 classes, the output layer has seven outputs. A diagram of the neural network is presented on Fig. 4 (a).

The matrices parameters are written Θ_1 for transition from input layer to hidden layer and Θ_2 from hidden layer to output layer. Each neuron has a sigmoid activation function g written:

$$g(z) = \frac{1}{1 + \mathrm{e}^{-z}}. \tag{10}$$

From an input vector $d = [X(k) \ldots C_{3L}]^T$, the activation function $g(z)$ and the weights Θ_i, the equations used to

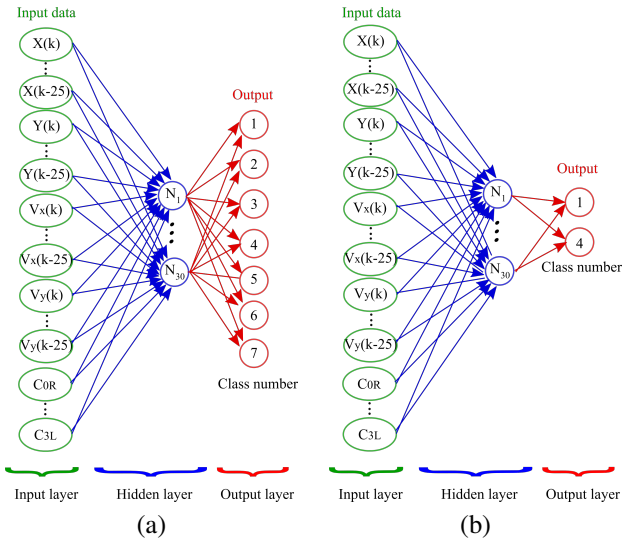


Fig. 4. Structure of the neural networks.

compute the output c can be found in [13]. c is 7x1 vector, each row corresponding to one class, with values between 0 and 1. The class corresponding to the input d is the one with the value the closest to 1.

The problem is that this structure does not take into account the initial lane position of the vehicle as an input. Therefore, inconsistencies may appear in the output. For example, the neural network may estimate that a vehicle is in class 4 (moving from the left lane to the middle lane) instead of 2 (remaining in the middle lane) which is impossible. Because the information of the initial lane is known, these inconsistencies can be limited by adding this information in the input vector d or totally avoided by creating three different networks, one for each lane.

The last option was chosen with the training of three networks. Each of them has the same input and hidden layers. The networks for the left and right lanes have 2 outputs and the network for the middle 3. The rest of the document is focused on the left line because the methodology and results used for the other lane are similar. The diagram of the network specifically used for the left lane is presented in Fig. 4 (b), only the classes 1 and 4, with initial positions in the left lane, remain.

As explained in the rule-based method section, the transverse speed of the object can be computed with two methods: using the relation (2) or using a Kalman filter. In order to see the influence of the transverse speed on the neural networks, two neural networks are trained and compared: one with V_y as (2) and the other with V_y estimated with a Kalman filter.

B. Training

The learning process of the parameters Θ_1 and Θ_2 uses the backpropagation also described in [13]. The objective is to find the parameters minimizing the cost function $J(\Theta_1, \Theta_2)$

defined as:

$$J(\Theta_1,\Theta_2)=\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{2}\Big(c_0^{(i)}[k]\log(c^{(i)}[k])+ (1-c_0^{(i)}[k])\log(1-c^{(i)}[k])\Big) \tag{11}$$

with m the number of examples in the dataset, $c_0^{(i)}[k]$ the k^{th} row of the real output vector corresponding to the i^{th} input vector d of the dataset and $c^{(i)}[k]$ the k^{th} row of the output computed with d, Θ_1 and Θ_2 using the network.

The backpropagation is a recursive method and thus, needs an initialization of the parameters Θ_1 and Θ_2. Here, these values are chosen randomly close to zero (inferior to 0.1) for symmetry breaking.

The feature scaling method is also applied on the input data in order to speed up the convergence of the gradient descent. The features are modified to be on a similar scale between -3 and 3. Each feature F is normalized using the equation:

$$F'=3\frac{F-\text{mean}(F)}{\max(F)-\min(F)}. \tag{12}$$

In order to prevent overfitting and to predict future observation reliably, a regularization term λ is added in the cost function:

$$J(\Theta_1,\Theta_2,\lambda)=\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{2}\Big(c_0^{(i)}[k]\log(c^{(i)}[k])+ (1-c_0^{(i)}[k])\log(1-c^{(i)}[k])\Big) + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{j=1}^{s_l[1]}\sum_{k=1}^{s_l[2]}\Theta_l[j][k] \tag{13}$$

with $L=3$ the number of layers, $s_l=[p_l\ q_l]$ a vector containing the size of the matrix parameters Θ_l with p_l the number of row and q_l the number of columns. The parameter λ can be interpreted as a penalty on the highest order of the network parameters that enables to improve the generalization of the network.

The dataset is divided in a training set used for the networks training and a test set used to check the veracity of the networks. The value of λ is chosen so that it minimizes the cost function (11) of the test set. The networks are trained with different regularization parameters λ using (13), then the cost functions (11) are computed with the trained parameters for the training and test sets. Fig. 5 shows the variation of the cost function of the trained networks parameters in function of λ.

On both plots of Fig. 5, when λ rises the cost of the training data rises meaning that the networks are less specific to the training set. At the same time, the cost function of the test data starts to decrease at low λ and then rises. It represents the fact that at first, the models of lane change describe by the neural networks are overfitted, meaning they are too close to the training set, then they become more generalized and at the end, when λ is too high, the networks do not represent the function of the lane change correctly,

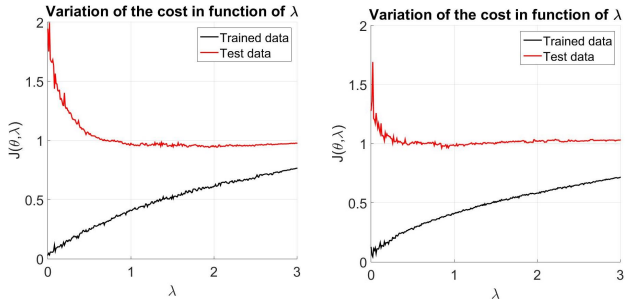


Fig. 5. Variation of the cost in function of the regularization parameter for the networks with classic V_y computation (a) and with Kalman V_y estimation (b).

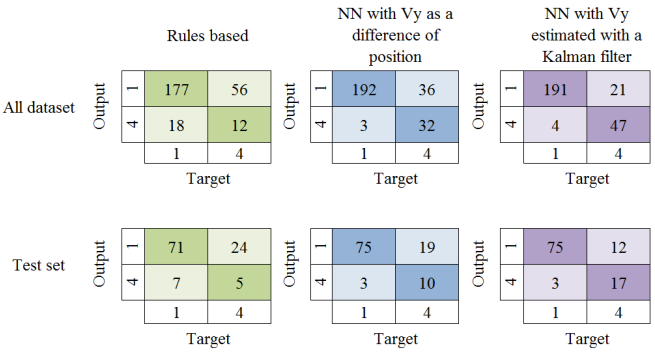


Fig. 6. Confusion matrices of all the dataset and the test set for the three lane change detection methods.

parameters are missing. For the neural network with V_y computed with (2) the value of λ that minimizes the cost function is $\lambda=1.94$ and for the other with V_y estimated with a Kalman filter $\lambda=0.85$.

V. Results

The rule-based method and the two neural networks are compared in this section.

Fig. 6 shows the confusion matrices of all the dataset and the test set for the three methods. It is important to note that for each method most of the errors occur when a lane change is not detected (output of the algorithm is class 1 whereas the actual situation belongs to class 4). Moreover, with a transverse speed less noisy, the neural network is able to reduce the number of mistakes.

However, for the rule-based method, as presented with the second example in section III.D, more samples may be needed in some cases where the algorithm classified the situation as lane following case. So on the 56 cases of lane changes where the rule-based method determined that the situation belong to class 1, in real time some of the lane changes could have been detected with 3 or 4 more samples. This also means that, potentially, the neural networks are quicker to detect a lane change than the rule-based method.

There exist several methods to describe the efficiency of a classifier [14]. In this paper, the accuracy, the precision and the recall are analyzed. The accuracy represents the overall effectiveness of a classifier, the precision the fraction

of relevant instances among the retrieved instances and the recall the effectiveness of a classifier to identify positive label.

TABLE III
TABLE OF TEST SET ERROR METRICS FOR THE THREE METHODS

	Accuracy	Class 1		Class 4	
		Precision	Recall	Precision	Recall
Rule-based	72%	75%	91%	42%	17 %
Neural Network 1	79%	78%	96%	77%	34%
Neural Network 2	86%	86%	96%	85%	59%

For a class i the precision of the class is computed with the equation:

$$\text{precision}_i = \frac{t_{pi}}{t_{pi} + f_{pi}}, \tag{14}$$

whereas the recall used the relation:

$$\text{recall}_i = \frac{t_{pi}}{t_{pi} + f_{ni}}, \tag{15}$$

with t_{pi} the number of true positive (classifier output and real output is in class i), f_{pi} the number of false positive (classifier output is in class i but real output is not) and f_{ni} the number of false negative (real output is in class i but classifier output is not).

Tables III presents the values of these measures for each methods with 'Neural Network 1' and 'Neural Network 2' referring respectively to the neural network with V_y computed as a difference of position and V_y estimated with the Kalman filter. Globally, the second neural network presents the best results in term of accuracy, precision and recall. All the classifiers are less effective to detect the lane change (class 4) than the line following (class 1). Indeed, by analyzing the recalls, when they are able to detect more than 90% of the class 1, 17%, 34% and 59% of class 4 situations are respectively detected for the rule-based method, the Neural Network 1 and the Neural Network 2. The precisions indicates that for the two neural networks, each class has the same fraction of relevant selected items. For the rule-based method, the precision is better with the class 1 than the class 4.

The traverse speed V_y has a significant impact on the neural network training and his results as a classifier. Having a more representative transverse speeds helps to classify the lane change scenarios.

VI. CONCLUSION

In this paper, two methods of lane change detection have been made. The first one uses a rule-based method with probabilities over the future position of the vehicle and its transverse speed. If in the next 0.6s the vehicle will be sufficiently close to the line and its transverse speed high enough then a lane change is detected. The second method uses neural networks with one-second sliding window of the vehicle past trajectory and speeds. Two neural networks are trained, with the transverse speed being computed either by the difference of position or estimated by a Kalman filter. All these methods are compared using a dataset of real scenarios recorded on open road with an autonomous vehicle prototype. The neural networks with the transverse speed computed using a Kalman filter presents better results in term of accuracy, precision and recall.

These methods are currently being compared on board of a vehicle in real time. The first remarks that can be made is that the rule-based method is less accurate than the neural networks in turn with more false lane change detection and that the second neural network is the fastest to detect lane change. More detailed analysis with a large database still need to be made in order to determine on average how much time before each method detect a lane change and which of them is the most relevant.

REFERENCES

[1] B. Morris, A. Doshi, and M. Trivedi, "Lane change intent prediction for driver assistance: On-road design and evaluation," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pp. 895–901, IEEE, 2011.

[2] D. Kasper, G. Weidl, T. Dang, G. Breuel, A. Tamke, A. Wedel, and W. Rosenstiel, "Object-oriented bayesian networks for detection of lane change maneuvers," *IEEE Intelligent Transportation Systems Magazine*, vol. 4, no. 3, pp. 19–31, 2012.

[3] M. Tsogas, X. Dai, G. Thomaidis, P. Lytrivis, and A. Amditis, "Detection of maneuvers using evidence theory," in *Intelligent Vehicles Symposium, 2008 IEEE*, pp. 126–131, IEEE, 2008.

[4] P. Boyraz, M. Acar, and D. Kerr, "Signal modelling and hidden markov models for driving manoeuvre recognition and driver fault diagnosis in an urban road scenario," in *Intelligent Vehicles Symposium, 2007 IEEE*, pp. 987–992, IEEE, 2007.

[5] P. Liu, A. Kurt, *et al.*, "Trajectory prediction of a lane changing vehicle based on driver behavior estimation and classification," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pp. 942–947, IEEE, 2014.

[6] G. Li, S. E. Li, Y. Liao, W. Wang, B. Cheng, and F. Chen, "Lane change maneuver recognition via vehicle state and driver operation signalsresults from naturalistic driving data," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*, pp. 865–870, IEEE, 2015.

[7] G. S. Aoude and J. P. How, "Using support vector machines and bayesian filtering for classifying agent intentions at road intersections," tech. rep., 2009.

[8] J. Zheng, K. Suzuki, and M. Fujita, "Predicting drivers lane-changing decisions using a neural network model," *Simulation Modelling Practice and Theory*, vol. 42, pp. 73–83, 2014.

[9] F. Bocklisch, S. F. Bocklisch, M. Beggiato, and J. F. Krems, "Adaptive fuzzy pattern classification for the online detection of driver lane change intention," *Neurocomputing*, vol. 262, pp. 148–158, 2017.

[10] R. Graf, H. Deusch, M. Fritzsche, and K. Dietmayer, "A learning concept for behavior prediction in traffic situations," in *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pp. 672–677, IEEE, 2013.

[11] H. Woo, Y. Ji, H. Kono, Y. Tamura, Y. Kuroda, T. Sugano, Y. Yamamoto, A. Yamashita, and H. Asama, "Lane-change detection based on vehicle-trajectory prediction," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1109–1116, 2017.

[12] R. E. Kalman *et al.*, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[13] M. T. Hagan, H. B. Demuth, M. H. Beale, *et al.*, *Neural network design*, vol. 20. Pws Pub. Boston, 1996.

[14] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.