Draft of paper accepted at
Proceedings of the 16th International IEEE Annual Conference on Intelligent Transportation Systems (ITSC 2013), pp. 276-281, The Hague, The Netherlands, October 6-9, 2013

Learning Driving Situations and Behavior Models from Data

Matthias Platho, Horst-Michael Groß and Julian Eggert

***Abstract*— For an Advanced Driver Assistance System recognizing the driving situation of other vehicles is a crucial prerequisite to anticipate their behavior and plan own maneuvers accordingly. Current methods for situation recognition usually rely on an expert for defining the considered driving situations manually while solely the parameters of the corresponding behavior models are learned from observations. Unfortunately, the performance of this approach is highly dependent on the skills of the expert. Furthermore, the data for training needs to be manually labeled to define when a certain type of situation is present, which can be very time-consuming and may introduce unwanted bias. In order to circumvent these problems, we propose to learn types of situations and behavior models from data simultaneously. The goal is to identify the set of driving situations for which the corresponding behavior models achieve the best fit to given observations. As both the assignment of observations to driving situations and the model parameters are unknown, an alternating, iterative algorithm minimizing the model error is employed. We show that the algorithm accomplishes to identify reasonable driving situations and that it can be successfully applied for behavior prediction when situation labels are missing.**

I. INTRODUCTION

Whenever an assistance system overrides the commands of the driver it has to ensure that the conducted maneuver causes no conflicts with other vehicles. In order to respect other vehicles the system has to predict the behavior of each nearby vehicle which is commonly realized by determining its driving situation and employing a corresponding behavior model. Works being concerned with the development of such systems usually define the considered driving situations explicitly using a human expert [1], [2], [3], [4].

In [1] the behavior of vehicles while crossing intersections is predicted. A particle filter matches each vehicle's current trajectory with a set of learned trajectories to predict which route for crossing the intersection the vehicle will choose. The method is tailored to only one specific type of situation: Two vehicles approaching a 4-way intersection from opposite sides.

Also specialized to a single driving situation is the work presented in [2], where a vehicle's deceleration behavior during car following is predicted. The prediction is based on a behavior model designed as a Dirichlet process mixture model which is trained on naturalistic driving data.

Multiple types of situations are considered in [3], which is concerned with learning a general longitudinal behavior model. This is accomplished by specifying a set of driving situations each of which employs an individual behavior model. The selection of the currently active behavior model is performed by a classifier according to the estimated driving situation.

In [4] multiple types of driving situations are considered as well, which are designed based on a set of selected behaviors. The goal is to predict the future trajectory of vehicles in highway scenarios, which is realized using a particle filter.

All of the approaches above have in common that the considered driving situations are handcrafted by human experts. While using human intuition provides a good way to incorporate relevant domain knowledge it also carries the risk of introducing unwanted bias. The recognition of the driving situation is in most approaches the crucial first step that determines the choice of the behavior model used and thereby also determines the following behavior prediction. This dependency leads to the problem that if a single designed driving situation actually subsumes two very distinct behaviors the corresponding behavior model will be highly inaccurate. Based on this insight we propose to learn *both* the considered driving situations and their corresponding behavior models *simultaneously*. The simultaneity is an important aspect as it accounts for the above mentioned dependency. Learning simultaneously enables the algorithm to identify these driving situations for which a common behavior model is adequate. An additional advantage of learning both is that the training data does not require any supervision in terms of labels regarding the 'actual' driving situation, which reduces the effort for learning and prevents labeling errors.

For the proposed learning method we use a set of observations, where each observation describes a vehicle's current state and its future behavior. The state is given by a set of features describing the dynamics of the vehicle and its current surrounding, e.g. nearby road users and traffic lights. The vehicle's behavior is represented by a discretized velocity profile, comprising its velocity for the next three seconds measured at 10 Hz. Representing behavior by velocity profiles has the advantage that while these profiles are good predictors for anticipating intentions [5] their simplicity makes them well manageable.

Given the number of driving situations to identify, the algorithm aims at finding the corresponding behavior models that describe the observations best, i.e. minimize the deviation between the velocity profiles estimated by the models and the actual velocity profiles. The learned behavior models can then be used to predict the future velocity profile of a vehicle, given its current driving situation. It is important to note that the problem at hand can not be solved by clustering, as the assignment of observations to driving situations should

M. Platho and H.-M. Groß are with Department of Neuroinformatics, Technical University of Ilmenau, D-98684 Ilmenau, Germany matthias.platho@tu-ilmenau.de

J. Eggert is with Honda Research Institute Europe GmbH, D-63073 Offenbach am Main, Germany jeggert@honda-ri.de

not be based on the similarity of the observations' velocity profiles but based on the underlying behavior models. For example an observation where a car slows down while approaching a red traffic light can cause a similar velocity profile like an observation where a car approaches a slow driving vehicle. However, while in one case the behavior and thus the velocity is probably a function of properties concerning the traffic light, in the other case the determinants of behavior will be based on the leading vehicle. Identifying these differences is a key element of the proposed method.

The remainder of this paper is structured as follows. In Section II the typical working principle of behavior prediction systems is discussed and the definitions and notations used in the following are introduced. In Section III the learning algorithm will be explained including the employed minimization function and the methods used for optimization. Section IV describes the evaluation procedure used to investigate the properties of the proposed method and in Section V we will show that our algorithm extracts competitive behavior models on simulated and real-world data. In Section VI an outlook on future work is given.

II. SYSTEM OVERVIEW

Generally speaking, a driving situation is a certain situational context which determines the behavior of a vehicle. The driving situation of a vehicle is defined by the vehicle's own state and the relations to its surrounding, which can be both quantified by a set of measureable features. Vehicles in the same driving situation are assumed to follow a common behavior pattern. Typical driving situations are tagged *Car-Following* or *Free-ride*, to name a few.

The novel aspect in our work is that we identify the set of considered driving situations D via learning while in most works D is handcrafted or artificially limited. The advantage of learning is twofold. First, it allows for learning behavior models without the need to label the training data with information about the assumed driving situation. Second, the driving situations that were identified via learning are able to capture even properties that are inaccessible to human intuition and might thereby fit real-world behavior better.

To direct the learning process the quality of a learned set D needs to be quantifiable. According to the definition of a driving situation it appears reasonable to obtain the measure from behavior information, as vehicles in the same driving situation behave according to the same model. One possibility is to let the algorithm learn both D and the corresponding behavior models simultaneously and measure the fit between learned and actual behavior. Given a set of training data containing state, situational context and behavior information of vehicles, the goal of the learning algorithm is to identify these driving situations for which the corresponding behavior models match the behavior of the vehicles best. Understandably, due to the fact that a behavior model takes state and situational context of a vehicle as input and provides an estimated behavior as output it can also be used for behavior *prediction*. Hence, the contribution of this work is to propose a learning algorithm that identifies both driving situations and behavior models from data which can then be utilized for behavior prediction.

In the following the typical working principle of behavior prediction systems is presented for highlighting the challenges for a learning algorithm and to introduce the notation used in this paper.

A prediction system performs four steps to arrive at an estimated behavior for an individual vehicle, which are depicted on the right side of Figure 1. The four steps are as follows.

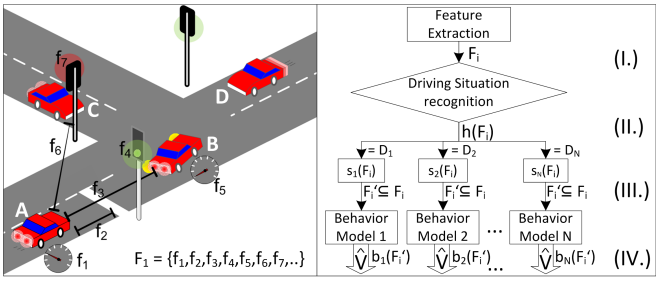

Fig. 1. Left: Features extracted for an individual vehicle that describe its current driving situation, e.g. velocity, state of nearby traffic lights or distances to other cars. Right: Standard architecture of behavior prediction systems depicting the utilization of the extracted features.

I. **Feature Extraction:** By taking sensory measurements, a set of features F_i is obtained, which is expected to capture the driving situation and the behavior of the vehicle adequately. (The index i simply denotes this measurement as the i-th observation.)

II. **Driving Situation Recognition:** Based on the measured features the current driving situation of the vehicle is determined. Formally, the second step can be interpreted as a classifier that takes the features as inputs and returns a situation label, i.e. $h(F_i) \rightarrow D = \{D_1, ..., D_N\}$ with N being the number of considered driving situations. Note that this step can be skipped when only a single driving situation is regarded, that is $N = 1$.

III. **Feature Selection:** According to the estimated driving situation a subset F_i' of the measured features F_i is selected, which is found to be relevant for the subsequent prediction step. This makes sense if some features do not carry useful information for a certain driving situation. For example in the left image of Figure 1 it can be assumed that features regarding vehicle C do not affect the current behavior of vehicle A. As this step is not mandatory it can be omitted. Alternatively, depending on the prediction model used, feature selection can also be implicitly carried out by the learning algorithm itself, like in LASSO [6] regression or CART [7].

IV. **Prediction:** The selected behavior model takes the set of features F_i' as input and predicts the considered vehicle's behavior for a specified time span.

The challenge in developing a suitable learning algorithm is that three aspects have to be learned at once, which are furthermore highly interdependent. Besides D also the fea-

ture selection $s_n(F)$ and behavior prediction models $b_n(F)$ need to be learned. Note that learning D is equivalent to learning $h(F)$ as D is fully determined by $h(F)$ and its possible outputs.

For the given learning problem there exists no closed form solution, that is why our algorithm works in an iterative and alternating manner, which is detailed in the next section.

III. BEHAVIOR MODEL LEARNING BASED ON VELOCITY PROFILES (BMLVP)

A. Target function

At first the target function for learning will be developed. The overall goal of the BMLVP algorithm is to minimize the total error e between estimated behavior and actual behavior for observations i. A behavior is represented by a velocity profile that is denoted as V and $\hat{V}$ for actual and estimated velocity profile, respectively:

$$e = \sum_i ||V_i - \hat{V}_i||_2 \tag{1}$$

A profile consists of thirty entries that describe the velocity of the considered vehicle for the next three seconds, sampled at 10 Hz. Time is denoted as $t \in \{1, ..., 30\}$. So Equation 1 can be rewritten as

$$e = \sum_i \sum_t ||V_{it} - \hat{V}_i(t)||_2 \tag{2}$$

The predicted velocity profile $\hat{V}_i$ is obtained from consecutively applying $s_n(F_i) \rightarrow F_i'$ and $b_n(F_i')$, where n is chosen according to the output of the driving situation assignment function h. One possibility to define h is to adopt the specification given above where

$$h(F_i) : F \rightarrow D_n \in \{D_1, ..., D_N\} \tag{3}$$

In this case h resembles a classifier, which leads to a problem: This classifier can not be trained because ground truth information is unavailable as D is unknown itself. Therefore a different approach is taken where observations are directly assigned to driving situations using an assignment matrix H. The matrix H is of order $I \times N$ with

$$H_{in} = p(D_n|i) \text{ with } \sum_n H_i = 1 \tag{4}$$

denoting the probability of the i-th observation to belong to driving situation D_n. The advantages of this representation will become clear later. Equation 2 then becomes

$$\min_{H,b,s} \quad \sum_i \sum_t ||V_{it} - \sum_n H_{in} b_n(s_n(F_i), t)||_2 \tag{5}$$

The predicted velocity profile $\hat{V}$ in Equation 2 is replaced by a sum term. The prediction for the i-th observation assuming the n-th driving situation is obtained by applying the prediction model b_n to the subset of features obtained by the selection function $s_n(F_i)$. This prediction is weighted by H_{in}. The predicted velocity profile $\hat{V}_i$ is thus the weighted average of the predictions provided by the individual behavior models. Note that we are interested in explaining each observation by a *single* behavior model that is why we will use an update rule that converges to a single one and $n-1$ zeros in each row of H.

The definitions of b and s will be given in the following subsection.

B. Feature Selection and Behavior model

A velocity profile consists of thirty individual values, one for each time step in the prediction horizon. It is possible to take a non-parametric approach for their representation and learn regression models that predict the velocity for each time step individually. The drawback is that this method would hardly be able to exploit the smoothness of velocity profiles, which comes from the fact that vehicles are physical systems and their movement is subject to inertia. It is therefore reasonable to model a profile by a smooth function. An example of such a function is

$$v_i(t) = v_{i0} + a_{i1}t + a_{i2}t^2 \tag{6}$$

which can also cope with non-linear velocity profiles. In order to obtain invariance towards the initial velocity v_0 it can be reduced to

$$v_i(t) - v_{i0} = a_{i1}t + a_{i2}t^2 \tag{7}$$

Despite its simplicity this representation works well as it will be shown in Section V.

The two parameters a_1 and a_2 are independently learned via multiple linear regression

$$a_{ij} = \beta_{j0} + \beta_{j1}f_1 + ... + \beta_{j|F_i|}f_{|F_i|} \tag{8}$$

for $j \in \{1, 2\}$ with β_f denoting the regression coefficients. The behavior model for the n-th driving situation is obtained by inserting Equation 8 into Equation 7

$$b_n(F_i, t) = (\sum_{k=0}^{|F_i|} \beta_{1nk} f_{ik})t + (\sum_{k=0}^{|F_i|} \beta_{2nk} f_{ik})t^2 \tag{9}$$

where f_{i0} is defined as 1 for providing the intercept of the regression. Note that a behavior model is uniquely defined by the vectors $\vec{\beta_1}$ and $\vec{\beta_2}$ of its regression coefficients.

Linear regression was chosen over other regression methods mainly due to its computational efficiency but also due to its ability to handle weights for observations. The need for handling weighted observations arises from the property that the assignment matrix H acts as a weight matrix, and is treated as such in the update rules presented in subsection *C*.

The functions s for selecting subsets of features F_i' are realized by a matrix S of size $N \times |F_i|$ with

$$s_{nk} = \begin{cases} 1 & \text{if } f_{ik} \in F_i' \text{ for } D_n, \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

The predicted velocity profile $\hat{V}_i(t)$ is therefore obtained by

$$\hat{V}_i(t) = \sum_n H_{in} \left((\sum_{k=0}^{|F_i|} s_{nk}\beta_{1nk}f_{ik})t + (\sum_{k=0}^{|F_i|} s_{nk}\beta_{2nk}f_{ik})t^2 \right) \tag{11}$$

C. Minimization algorithm

Minimizing the total error e between estimated and actual behavior requires the simultaneous adaptation of assignment matrix H, feature selection matrix S and regression coefficients β. As for this optimization problem no analytic solution exists an iterative, alternating algorithm is employed, which is inspired by the EM-algorithm [8]. The overall working principle is randomly initializing H and then sequentially updating β, S and H.

The update rule for β is a straightforward weighted linear regression

$$\beta_{jn} = (F^T \vec{H}_n F)^{-1} F^T \vec{H}_n \vec{a}_j \tag{12}$$

where $\vec{H}_n$ denotes the n-th column of matrix H. Note that H is used as the weight matrix which allows to neglect observations that are improbable to belong to the regarded driving situation and vice versa.

Due to the property that no gradient for S can be computed the update is accomplished based on an exhaustive search on all pairwise swaps of entries. A swap is performed by exchanging a 1 and a 0 for the same feature. For each possible swap the resulting model error e' is computed and compared to e when using the original S matrix. If e' of the best possible swap is lower than e then S is updated accordingly, otherwise S remains unchanged.

The entries of the assignment matrix H are updated according to an observation's model error for the individual behavior models.

$$p_{in} = \frac{H_{in}}{e_{in}} \tag{13}$$

$$w_{in} = \frac{p_{in}}{\sum_n H'_{in}} \tag{14}$$

$$H'_{in} = l \times w_{in} + (1 - l) \times H_{in} \tag{15}$$

The current probability of the i-th observation in the assignment matrix is divided by its error using the n-th behavior model (Equation 13) and normalized (Equation 14). The parameter l acts as the learning rate and controls the impact of an update (Equation 15). Note that due to the multiplication in Equation 13 the rows of H converge to a vector with all zeros and a single 1, which is a desired property.

As the update rules for β and H ensure that the total error e is decreased and the update of S at least maintains it, each iteration is guaranteed to improve e. As the error converges asymptotically to a certain value, the criterion used for stopping the minimization procedure is based on the relative improvement compared to the previous iteration.

D. Application to unseen data

In section B it was stated that learning classifiers h, which map observations to driving situations based on the observation's features (see Equation 3), is not feasible. Instead the proposed BMLVP algorithm maps each observation individually to a driving situation. The question is now how new observations that were not part of the training data can be assigned to driving situations. The natural solution that we use is to learn the classifiers *after* the algorithm has converged, where the 'correct' driving situation d_i is simply taken from each observation's highest entry in the assignment matrix H.

$$d_i = n | \quad H_{in} = \max_m H_{im} \tag{16}$$

IV. EVALUATION

The BMLVP method proposed in this paper aims at identifying those driving situations for which holds that vehicles in the same situation behave according to the same model. We pursue two goals with our algorithm: First, to be able to identify driving situations comparably or better than a human expert, as the latter can potentially be subject to the bias of human introspection. The second goal is to allow for learning behavior models from unlabeled data.

Unfortunately the first goal is difficult to evaluate, as whenever the algorithm outperforms a human expert it is hard to rule out the possibility that the expert simply lacks the required skills for identifying driving situations correctly. We therefore take a slightly different approach and use for a comparison ground truth taken from *CarD*, a microscopic traffic simulator, presented in [9]. The intuition behind this is that in the simulator vehicles are modeled as agents which follow at each time step one out of four behaviors based on their current driving situation. By using the simulator the actual driving situations are thus known and can be compared to the ones identified by the proposed method. A good result would be either a high match between identified and actual driving situations or if the behavior prediction error of the learned models excels the prediction error of models learned based on the simulator's ground truth.

The degree to which the second goal is achieved - learning behavior models from unlabeled data - is in turn directly quantifiable: By assessing the algorithms performance on a real-world dataset where no labels for the actual driving situation are given. A dataset recorded from inner-city driving is used that was first presented in [10]. The total error in the behavior prediction using the BMLVP algorithm is compared to the error resulting from a baseline algorithm and a state of the art regression algorithm. The baseline algorithm computes the predicted velocity profile from a straightforward extrapolation of the current dynamics of the vehicle. The state of the art regressor used is a Random Forest Regressor [11] which takes all features as inputs and returns predicted velocities. The algorithms will be referred to as *BASELINE* and *RFR*, respectively.

For driving situation recognition and behavior prediction a set of 10 variables is used. Note that 'target car' refers to

the currently considered vehicle for which a recognition or prediction is performed.

- *Velocity (VEL)*: Velocity of target car in $^m/_s$
- *Acceleration (ACC)*: Acceleration of target car in $^m/_{s^2}$
- *Traffic light distance (TLD)*: Distance to the stopping line of the next, relevant traffic light in m
- *Traffic light state (TLS)* : State of next, relevant traffic light. 1 if green, 0 otherwise
- *Car ahead relative speed (CAS)*: Relative velocity between target car and its leading car in $^m/_s$
- *Car ahead distance (CAD)*: Distance between target car and its leading car in m
- *Car ahead TTC (TTC)*: Time to contact between target car and in ts leading car in s
- *Intersection distance (ID)*: Distance to the entry point of the next intersection in m
- *Major Road (MJ)*: Whether target car is driving on a major road (1) or minor road (0).
- *Time (TIME)*: (*RFR* only) Time instance for which the velocity is predicted in s. Values are $0.1, 0.2, 0.3...3.0$

The features TTC, ID and MJ are only available for the simulated data and are thus not used for the evaluation on the real-world dataset.

The simulated data was obtained by running the microscopic traffic simulator for about 40 minutes on an urban intersection scenario as shown in Figure 2.

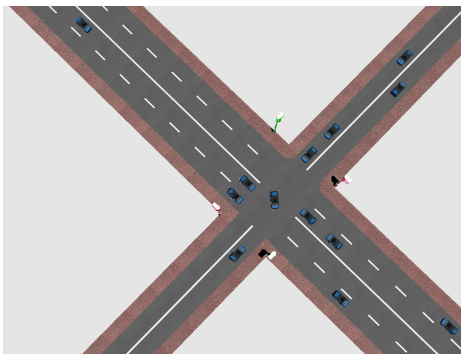

Fig. 2. Top view on the simulated intersection in the microscopic traffic simulator *CarD*.

At a rate of 20 Hz for each vehicle in the simulation all features and the retrospectively obtained velocity profile were recorded including the driving situation as perceived by the corresponding agent, resulting in 35506 observations. The first 8000 observations were used for training and the remaining ones for testing. The evaluation of the benefits from learning driving situations from data is performed by training two models using BMLVP. The first model, tagged *BMLVP-Sim*, omits learning H but takes the ground truth information of the actual driving situation to specify H. The second model, tagged *BMLVP-Orig* is learned as described above.

From the real-world dataset the first 15 minutes were used for learning and the remaining 14 minutes were used for testing. Note that though the features are taken from the ego-vehicle only they can also be obtained from other road users using nowadays sensor systems.

As evaluation metric for velocity profiles the Mean Sum of Squared Errors (MSSE) as given in Equation 1 is used. The match between learned and actual driving situations on the simulated data is determined by the purity measure, which was presented in [12].

For all experiments BMLVP's learning parameter l was set to 0.1 and the algorithm was stopped if the relative improvement r after an iteration was less than 1%.

V. RESULTS

In Table I the minimum, mean and maximum purity are given for 64 runs of BMLVP on the simulated dataset. The average purity is 0.63 which means that the assignment of observations to driving situations differs noticeable when the driving situations are learned versus when they are taken from the simulator's ground truth. The BMLVP algorithm identifies significantly different driving situations than which are actually used by the agents in the simulator.

TABLE I

MATCH BETWEEN LEARNED DRIVING SITUATIONS AND SITUATIONS USED BY THE SIMULATOR'S AGENTS.

Purity		
Min	Mean	Max
0.38	0.63	0.80

However, having a high match is neither necessary nor advantageous for the learned driving situations, as it is shown in Figure 3.

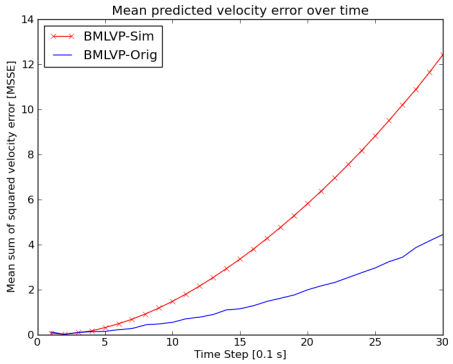

Fig. 3. Learned driving situations outperform the simulator's driving situations in prediction accuracy.

Here the results of the two models with the lowest training error are plotted. The average prediction error over time rises significantly faster for behavior models based on the simulator's actual driving situations (*BMLVP-Sim*) than the prediction error of behavior models based on learned driving situations (*BMLVP-Orig*). This result indicates that even when the 'correct' driving situations are known an approach that is based on learning can be beneficial. It can be assumed that the learned driving situations capture specific feature

constellations that are more relevant for the *future* behavior of a vehicle than the feature constellations that are used to determine the *current* behavior of an agent.

The evaluation on the real-world dataset also confirms the benefits of BMLVP. In Figure 4 the velocity prediction error over time is given; it shows that BMLVP is significantly more accurate than *BASELINE* and *RFR*. The unexpected bad performance of the random forests is due to the fact that training and test set were split by time stamp and thus have considerably different statistics. When training and test set are split randomly over the whole dataset then *RFR* performs better, but this is an unrealistic scenario for a method used in a driver assistance system.

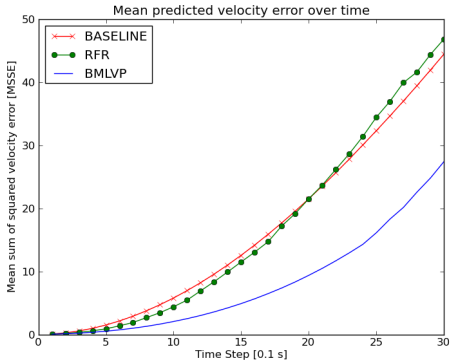

Fig. 4. Prediction error using BMLVP on the real-world dataset excels both *BASELINE* and Random Forests (*RFR).*

In the evaluation on the real-world dataset five driving situations were learned on the training set, which took about three hours on a single 3.2 GHz CPU. While the learning rate l and the stopping criterion r are parameters of BMLVP that simply trade computation time against accuracy, the single relevant parameter is the number N of driving situations to learn. Like in other methods the higher the number of N is chosen, the better the algorithm can adapt to the training set.

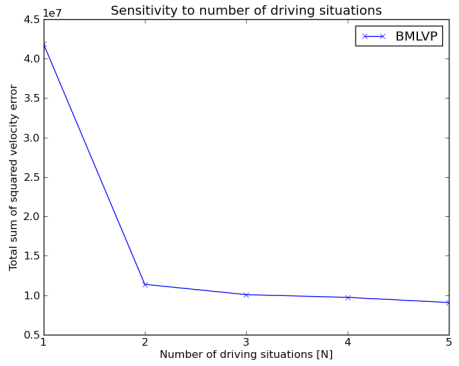

Fig. 5. The choice of the number of learned driving situations N affects the overall performance only slightly for $N > 3$.

Eventually increasing N decreases the learned models accuracy on unseen data, so a kind of regularization will be needed, which is subject to future research. Nevertheless, at least on the dataset used, N is no critical parameter as long as it is above 1. This is depicted in Figure 5 where the number of learned driving situations is plotted against the total model error on the test set. While learning only a single driving situation leads to a high model error the improvement by using more driving situations than two is rather moderate.

VI. CONCLUSION AND FUTURE WORK

In this paper a novel approach for learning driving situations and behavior models simultaneously from data has been proposed. It is based on the idea that the driving situations defined by human experts might be subject to unwanted bias or inadequate for a prediction task. Even if an expert would be always correct, supervised learning behavior models from data requires a time-consuming labeling of the data set.

The method we propose is inspired by the EM-algorithm and allows to identify these driving situations for which the corresponding behavior models anticipate the velocity profiles of other road users most accurately. On both simulated and real-world data we show that our approach learns driving situations with competitive behavior models.

The prediction model we use is based on a simple linear regression. We plan to replace this model by more powerful methods to improve the prediction accuracy further. At the same time extensions for achieving regularization will be investigated.

REFERENCES

[1] E. Käfer, C. Hermes, C. Wöhler, H. Ritter, and F. Kummert, "Recognition of situation classes at road intersections," in *Proc. IEEE Int Robotics and Automation (ICRA) Conference*, 2010, pp. 3960–3965.

[2] P. Angkititrakul, C. Miyajima, and K. Takeda, "Analysis and prediction of deceleration behavior during car following using stochastic driver-behavior model," in *Proc. IEEE Intelligent Transportation Systems (ITSC)*, 2012.

[3] M. Eilers and C. Möbus, "Learning the human longitudinal control behavior with a modular hierarchical bayesian mixture-of-behaviors model," in *Proc. IEEE Intelligent Vehicles Symp. (IV)*, 2011, pp. 673–678.

[4] T. Gindele, S. Brechtel, and R. Dillmann, "A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments," in *Proc. 13th Int Intelligent Transportation Systems (ITSC) IEEE Conf*, 2010, pp. 1625–1631.

[5] M. Liebner, M. Baumann, F. Klanner, and C. Stiller, "Driver intent inference at urban intersections using the intelligent driver model," in *Proc. IEEE Intelligent Vehicles Symp. (IV)*, 2012, pp. 1162 – 1167.

[6] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. o. t. Royal Statistical Society. Series B*, pp. 267–288, 1996.

[7] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. Chapman & Hall/CRC, 1984.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *J. o. t. Royal Statistical Society. Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[9] M. Platho, H.-M. Groß, and J. Eggert, "Traffic situation assessment by recognizing interrelated road users," in *Proc. IEEE Intelligent Transportation Systems (ITSC)*, 2012.

[10] M. Garcia Ortiz, J. Fritsch, F. Kummert, and A. Gepperth, "Behavior prediction at multiple time-scales in inner-city scenarios," in *Proc. IEEE Intelligent Vehicles Symp. (IV)*, 2011.

[11] L. Breiman, "Random forests," *Machine Learning*, vol. 45, 2001.

[12] Y. Zhao and G. Karypis, "Criterion functions for document clustering: Experiments and analysis," *Machine Learning*, 2001.