Car Ride Classification for Drive Context Recognition

Stefan Haas
Institute for Informatics
LMU Munich
Munich, Germany
Email: haasst@cip.ifi.lmu.de

Kevin Wiesner
Institute for Informatics
LMU Munich
Munich, Germany
Email: kevin.wiesner@ifi.lmu.de

Thomas Christian Stone
BMW Group
Munich, Germany
Email: thomas.stone@bmw.de

Abstract—**The automotive domain, with its more and more increasing number of comfort and infotainment functions, offers a field of opportunities for learning and context-sensitive functions. In this respect, personal and frequent trips of drivers provide very promising and interesting contexts. To identify frequent driving contexts in a set of recorded GPS tracks, this paper presents two different clustering algorithms: First, a hierarchical *Drive-Clustering*, which combines drives based on their number of common GPS points. Second, a *Start-Stop-Clustering*, which combines trips with the same start- and stop-cluster utilizing density based clustering. Especially the *Start-Stop-Clustering* showed particularly good results, as it does not depend on the concrete routes taken to a stop position and it is able to detect more trip clusters. To predict these trip contexts, a Bayesian network is presented and evaluated, with logged trip data of 21 drivers. The Bayes classifier uses context information such as the time, weekday and the number of persons in the car, to predict the most likely trip-context and thus achieves a good accuracy in the prediction of the different trip contexts.**

Keywords*–*Context-aware Vehicle; Spatial Clustering; Drive Context Prediction

I. Introduction

Context-awareness is an important building block in the development of intelligent systems as it can significantly improve the interaction between a user and a system. Any information that enables a system to provide the user with useful, context-related information or intelligent behavior, can be considered a context. Knowledge about a specific context is normally gathered by sensor readings and their interpretation [1][2].

With its steadily increasing number of comfort and infotainment functions, the automotive domain offers a unique field of opportunities for learning and context-sensitive functions. In recent years, many different context-aware advanced driver assistance systems (ADAS) have already been introduced. They are based on information which is provided by dedicated sensor systems, especially in the areas of safety and comfort, like the lane departure warning system (LDW), adaptive cruise control (ACC) or intelligent speed adaption (ISA).

Another interesting and promising context to advance vehicle personalization is the drive itself. Above all, the repeated drives of a person offer a lot of potential for finding consistent usage patterns and subsequently the possibility of automating recorded user behavior after a certain learning period. For example, if a driver usually checks his mail on the way to work or likes to listen to the news, the vehicle could adapt to his preferences by recognizing the drive context as a regularly drive to work and by automating the desired functions. This automation of functions could improve safety as well as comfort because the driver is no longer forced to adjust his personal settings by himself.

In the following, we will describe and evaluate different methods for the detection and prediction of repeated drives of individual drivers. To develop and evaluate our proposed methods, we had the possibility of utilizing recorded vehicle sensor data of 21 drivers collected over several months by a data logger. The collected data included many different sensor signals exchanged between the different in-car electronic control units (ECU) over the Controller Area Network (CAN) bus, ranging from Global Positioning System (GPS) position to seat belt status.

The contributions of our paper are two novel clustering methods for detecting repeated trips of individual drivers, a novel distance measure based on the Jaccard distance for comparing GPS tracks and a hybrid Bayesian network for predicting frequent drive contexts right away from the start of the trip based on contextual information like the time of the day or the number of passengers in the car.

The paper is structured as follows. Section II gives an overview on existing work in the fields of route prediction, route recognition, destination prediction and place mining. Section III outlines two new spatial clustering methods for detecting the frequent drive contexts of a particular driver. In Section IV, we present a hybrid Bayesian network to predict the frequent drive contexts of an individual driver right away from the start of the trip. The results we obtained running the before presented algorithms individually on the collected drive data of every single driver are described in Section V. We close our work in Section VI with a summary and an outlook on possible future work.

II. Related work

Route recognition and prediction systems have been proposed in many different works [3][4][5][6][7]. In the majority of these publications, the general way to predict respectively recognize the current route is based on the comparison of the current driving trajectory against previously recorded trajectories using a distance measure. As comparing GPS tracks can not be done with classic L_p metrics due to their length related inequality, dimension and noise, novel more elastic distance measures are needed. Already proposed distance measures, were for example, based on the longest common sub-sequence (LCSS) algorithm [3][8][9], the Hausdorff distance [4] or the Jaccard distance [10]. In [8], this simple instance based learning approach of comparing the current route to already recorded routes is further enhanced by the inclusion of contextual information (e.g. time of the day) to better differentiate overlapping routes.

ISBN: 978-1-61208-366-7

Probabilistic approaches for route and destination prediction have been presented amongst others in [10][11][12] and [13]. The investigated prediction methods hereby often underlie a Bayesian approach and include additional contextual information like the time of the day, the particular weekday or even background information about locations to infer the most likely route or destination [13]. In [12], a Markov model is used instead of a Bayesian approach to predict the next location of a user.

Identifying personally important places of users in recorded GPS data has for example been investigated in [14][15][16][12] and [7]. Density based clustering hereby proved more efficient than classic partitioning algorithms like k-means [17][18][14][15], as the final clusters only consist of dense regions in the data space. Regions of low object density are not included in the final clusters and are considered as noise.

Our work differs from existing publications, as we focus on the personal repeated drives of individual drivers and their prediction. We thereby consider a set of similar drives included in a repeated drive cluster as a certain drive context and as a basis for learning and automating user settings to advance comfort and safety.

III. Detecting frequent drives

To detect frequent drive clusters of an individual driver, we present and evaluate two different spatial clustering methods explained in the following two Subsections. *Drive-Clustering* is based on the Jaccard distance and compares whole trajectories using hierarchical clustering, whereas *Start-Stop-Clustering* focuses on semantically similar routes based on the before determination of frequent start and stop positions of the particular driver. The goal of both algorithms is to identify repeated patterns in the set of recorded GPS tracks in order to detect repeatedly occurring drive contexts, e.g., drives from home to work. In Section V, we compare the obtained results of both algorithms applied to our test data set.

A. *Drive-Clustering*

An important factor in cluster analysis is a distance measure to determine the distances between elements contained in the data, for the purpose of grouping similar elements together in clusters. In trajectory data the standard way for identifying patterns is to compare whole trajectories. In our case, the trajectory data of each drive is stored as a sequence of GPS points $S_i = \{p_{i,1}, p_{i,2}, ..., p_{i,n}\}$, with $p_{i,1}$ being the start point of the drive and $p_{i,n}$ being the end or stop point.

To compare two point sequences we use a dissimilarity measure based on the well known Jaccard distance, which measures dissimilarity between sample sets [19] (see equation 1):

$$d(X,Y) = 1 - \frac{|X \cap Y|}{|X \cup Y|}. \tag{1}$$

Our dissimilarity measure thereby calculates the intersection of the two GPS sequences S_i and S_j by counting the number of common points $NOCP(S_i, S_j)$ contained in both sequences starting from the shorter sequence (see equation 2). This number of common points value is then divided by the number of points contained in the shorter sequence $min(S_i, S_j)$. In order to obtain a dissimilarity measure the whole term is subtracted from 1, so that a result of 0 signifies maximum similarity and a value of 1 maximum dissimilarity.

$$d(S_i, S_j) = 1 - \frac{NOCP(S_i, S_j)}{min(S_i, S_j)}. \tag{2}$$

GPS points of two geometrically similar trajectories are very unlikely to have the exact same coordinates, due to different driving speeds and other noise. Hence it is necessary to define a threshold Θ from which two points are considered as equal or contained in both sequences (common points), e.g., 50 meters. The threshold needs to be defined dependent on the logging frequency. In our case the logging frequency is $f = 1Hz$. So when we for example consider 135 km/h as the maximum vehicle speed, the maximum distance between two succeeding points will be $(135*1000)m/3600s = 37.5m$. In the evaluation we set the threshold to 50 meters, which is sufficient for driving speeds up to 180 km/h with a logging frequency of $f = 1Hz$.

The number of common points (NOCP) algorithm iterates over all points $p_{i,k} \in S_i$ included in the shorter sequence and tries to find at least one point in the other sequence $p_{j,l} \in S_j$ whose distance is less or equal than the defined threshold distance Θ. If the set of found points in range is not empty, the number of common points counter is increased. Consequently, the presented distance measure is more elastic than distance measures based on dynamic programming, like the longest common sub-sequence (LCSS) or dynamic time warping (DTW), as it is able to match several elements of one sequence to just one element of the other sequence. This behavior is important in our case to handle traffic jams and different driving speeds. The implementation of the number of common points (NOCP) function can be significantly sped up by storing the queried sequences' points in a *k-d tree* [20].

To calculate the distance between two-dimensional GPS points we use a simplification of the *haversine* formula [21] based on the euclidean distance, which in contrast to the standard euclidean distance allows metric parametrization of our algorithms (ϕ latitude, λ longitude) (see equation 3).

$$\begin{gathered} dist(\phi_1, \lambda_1, \phi_2, \lambda_2) = (((111.3 * \cos{(\frac{\phi_1 + \phi_2}{2})} * \\ (\lambda_1 - \lambda_2)^2) + (111.3 * (\phi_1 - \phi_2)^2))^{\frac{1}{2}} * 1000. \end{gathered} \tag{3}$$

In order to avoid the problem of a very much shorter sequence being contained in a longer sequence and to speed up the comparison, the number of common points in the two sequences is only calculated, when the start and stop points of the two sequences are sufficiently similar, e.g., their respective distances do not exceed 250 meters ($p_{i,1} \sim p_{j,1}$ and $p_{i,n} \sim p_{j,m}$). Otherwise the maximum dissimilarity value 1 is returned without any further calculation (see equation 4).

$$d_{opt}(S_i, S_j) = \begin{cases} 1 - \frac{NOCP(S_i,S_j)}{min(S_i,S_j)}, & \text{if } p_{i,1} \sim p_{j,1} \\ & \wedge\ p_{i,n} \sim p_{j,m} \\ 1, & \text{otherwise} \end{cases}. \tag{4}$$

To group similar drive contexts in clusters, we use agglomerative hierarchical clustering, starting from single GPS sequences. To stop the calculation when no sequence anymore undercuts a distance ε to another sequence we need to define a similarity threshold, e.g., $\varepsilon = 0.05$. The smaller the value ε the more similar are the trips contained in a cluster. This threshold will cut the *dendrogram* at a certain level and lead to the final drive clusters. To predefine the minimum cluster size we use another parameter $MinDrives$, referring to the $MinPoints$ parameter in density based clustering [18].

B. Start-Stop-Clustering

Another way of determining frequent drive contexts of a certain driver is based on his frequent start and stop positions. In contrast to the above presented trajectory clustering method this method rather focuses on semantically similar drives with the same start and stop positions than on geometrically similar drives or routes.

As the vehicle is typically not parked at the exact same coordinates, it is necessary to merge similar parking positions to *start-stop-clusters*. To obtain these frequent start and stop position clusters of a particular driver, we use density based clustering, to be exact the DJ-Cluster algorithm presented in [14], which is a simplification of DBSCAN [18] [22]. Density based clustering has the advantage of explicitly eliminating outlier points compared with partitioning clustering, e.g., k-means [17] [22]. As we are only interested in dense regions included in the set of start and stop positions of an individual driver in order to identify frequent drive contexts, density based clustering is suitable for our task.

Consequently, the first step in Start-Stop-Clustering is to calculate dense regions of start and stop positions in the set of GPS sequences and to store the cluster IDs of every GPS sequences' start and stop points. Therefore, it is necessary to specify the two parameters $MinPoints$ and ε, representing the minimum cluster size and search radius respectively. Figure 1 shows an example of a dense point cluster found in the drive data of a particular driver with $\varepsilon = 100\text{m}$.

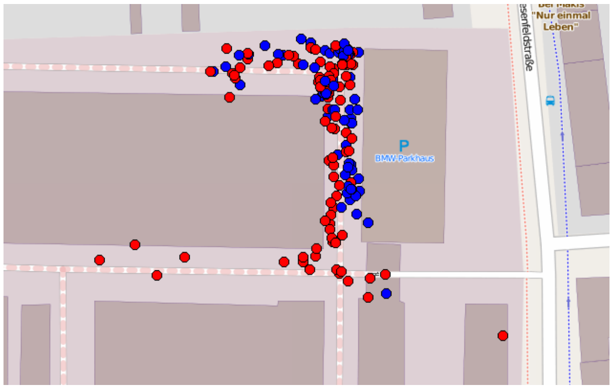

Figure 1. Visualization of the start (red) and stop points (blue) of a driver. All shown points are included in the same point cluster.

The binary dissimilarity measure for Start-Stop-Clustering then looks as follows (see equation 5):

$$d(S_i, S_j) = \begin{cases} 0, & \text{if } C_s(p_{i,1}) = C_s(p_{j,1}) \\ & \wedge\, C_e(p_{i,n}) = C_e(p_{j,m}) \\ 1, & \text{otherwise} \end{cases} . \qquad (5)$$

Two GPS sequences S_i and S_j are considered as equal, when their corresponding start $(p_{i,1}, p_{j,1})$ and stop points $(p_{i,n}, p_{j,m})$ lie in the same start C_s respectively end cluster C_e.

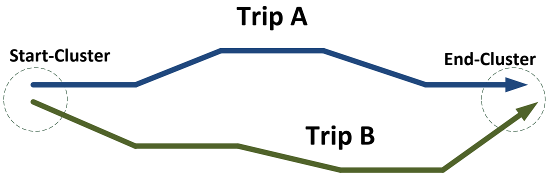

Figure 2. Illustration of a route-independent *Start-Stop-Cluster*.

Hence, the final drive clusters are comprised of GPS sequences whose start and stop points lie in the same dense region or point cluster and therefore have the same cluster IDs. The found frequent drive contexts are direction-dependent just like those obtained with the above presented Drive-Clustering approach. However, the drives included in a *Start-Stop-Clustering* drive context cluster do not necessarily follow the same routes. In contrast to *Drive-Clustering* they are route-independent (see Figure 2). To predefine the minimum cluster size we also use the $MinDrives$ parameter.

IV. Predicting Frequent Drive Contexts

To predict frequent drive contexts that have been identified with one of the above presented methods, we propose a hybrid Bayesian network. The structure of the network is shown in Figure 3.

The goal is to predict a present frequent driving context, e.g., a drive to work, as early as possible during the drive. Therefore we make use of contextual information associated with a certain drive context cluster. The contextual information used to infer the current drive context includes the start point of the drive, the number of passengers in the car, the weekday, the start time and the fuel level.

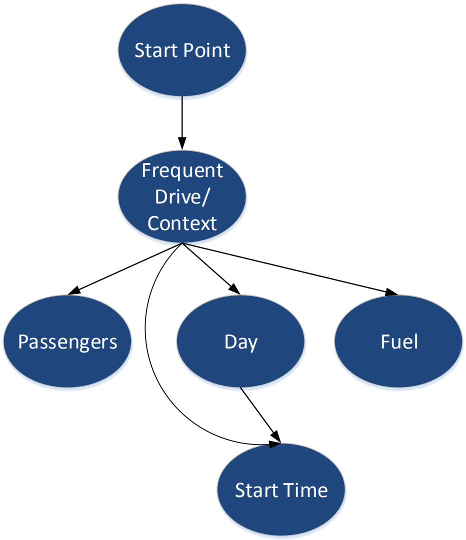

Figure 3. Topology of the hybrid Bayesian network fo predicting the most likely frequent drive context.

Using the start point of the drive we are able to eliminate impossible contexts, e.g., a drive from work to home if the start point is home, which significantly reduces the possible

Copyright (c) IARIA, 2014. ISBN: 978-1-61208-366-7

contexts, prevents false positives and speeds up the implementation. The variable *Frequent Drive/Context* represents the *a priori* probability distribution over the set of identified drive contexts, already constrained by the current start point. The variables *Day*, *Passengers* and *Fuel* are conditionally independent of each other given the *class* variable *Frequent Drive/Context*. The variables described so far all underlie a discrete probability distribution.

In contrast to the other probability variables, we model the variable *Start Time* as continuous. By the edges between *Frequent Drive/Context*, *Day* and *Start Time* we receive a drive context dependent start time *probability density function* (PDF) for every single day. This enables a stronger differentiation between the drive contexts, as the start time probabilities for the different contexts are also day dependent.

To approximate the probability density function for the start times associated with a certain drive context we use *kernel density estimation* (KDE) (equation 6) with a Gaussian kernel (equation 7) and Scott's *rule of thumb* (equation 8) for bandwidth selection h [23]:

$$\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - x_i}{h}). \tag{6}$$

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp{(-\frac{x^2}{2})}. \tag{7}$$

$$h_{scott} = n^{-1/(d+4)}. \tag{8}$$

By using *kernel density estimation* we receive continuous day and context dependent probability density functions for the start times, with high probabilities during day times the drive context normally occurs (see figure 4).

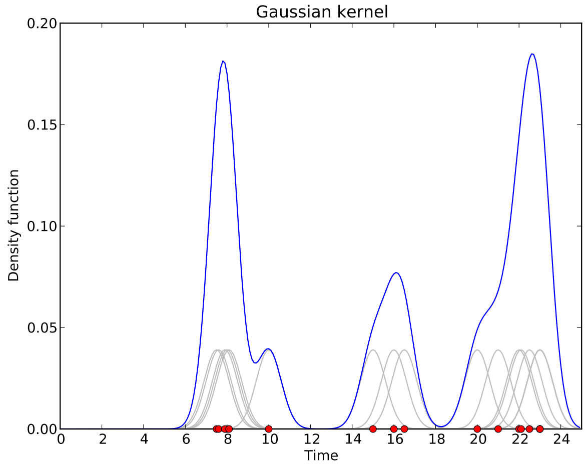

Figure 4. Example of a probability density function for the *Start Time* variable of a particular drive context.

We deliberately do not use *Laplacian correction* to deal with zero probabilities. When a drive context has not occurred before, at a certain day or time, the probability for the whole context will be zero. This helps in preventing false positives.

The probability for a certain context C, given the start point s, the weekday d, the time t, the number of persons in the car p and the fuel level f, can then be calculated with the following formula:

$$P(C|s, d, t, p, f) \propto \\ P(C|s)P(d|C)P(t|d, C)P(p|C)P(f|C). \tag{9}$$

The context C_i leading to the highest probability value $P(C_i|s, d, t, p, f)$ is then assumed to be the present context:

$$\arg\max_{C_i}\{P(C_i|s, d, t, p, f)\}. \tag{10}$$

V. Evaluation

To evaluate the described methods, we had access to a data set collected by 21 drivers over several months. The logger used for collecting the data records all kinds of data bus traffic, also when the car is not moved, e.g., when the electronic key is pressed. To filter out this unwanted noise, we only used recorded data for our evaluation where the vehicle was at least moved 1 kilometer (air-line distance). The minimum number of filtered drives of one driver was 216, the maximum number 986. The majority of the probands ranged between 400 to 600 recorded drives.

A. Drive clustering

Figures 5 and 6 show the results obtained applying Start-Stop-Clustering and Drive-Clustering to the data set. Figure 5 illustrates the average number of found clusters for different minimum cluster sizes (MinDrives={3,5,10}). Figure 6 presents the average share of repeated drives of the total quantity of drives.

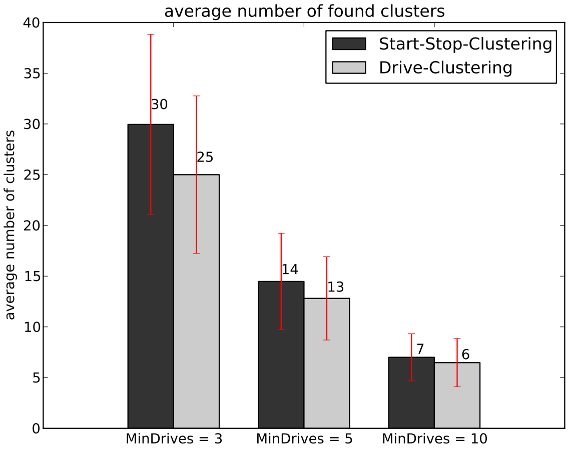

Figure 5. Average number of found clusters with Start-Stop- and Drive-Clustering dependent on the minimum number of drives contained in the clusters (MinDrives).

As one can see, Start-Stop-Clustering is on average able to identify more clusters than Drive-Clustering (see Figure 5). However, with increasing the minimum cluster size, the difference between the average number of found clusters by Start-Stop-Clustering and Drive-Clustering decreases. This leads to the assumption that for frequent drives (MinDrives=10), drivers usually have a preferred route that they normally take, whereas

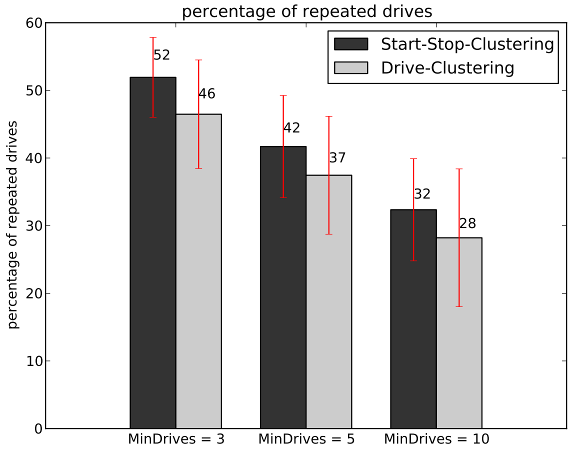

Figure 6. Percentage of repeated drives identified with Start-Stop- and Drive-Clustering dependent on the minimum number of drives contained in the clusters (MinDrives).

for less frequent drives (MinDrives=3) they also take different routes to the same destination. In addition to the number of found clusters, Start-Stop-Clustering is on average able to assign a larger fraction of the overall number of drives to a repeated drive cluster compared to Drive-Clustering, as it also includes all route alternatives (see Figure 6).

As we are rather interested in detecting frequent drive contexts than the frequent routes taken by a driver, Start-Stop-Clustering is more appropriate for our use case. Especially large clusters (MinDrives $\geq$ 10) may provide promising and interesting contexts, on the basis of which usage patterns may possibly be learned and automated. The average fraction of trips repeated at least 10 times by the participants during the survey amounts to approximately 30% of the overall trips (see Figure 6).

To keep the set of frequent driving contexts up-to-date one could use a shifting time frame and only consider drives for the cluster calculation that for example occurred during the last 6 months. This would lead to a slow exclusion of no longer appearing driving contexts over time and also limit the amount of data used for the context identification.

B. Prediction

To evaluate our proposed Bayesian inference system for predicting frequent drive contexts, we made use of cross-validation and focused on clusters identified by Start-Stop-Clustering with a cluster size larger than 10 drives.

Figure 7 shows the overall prediction result for all drives, including also non-frequent drives, as well as the prediction result for solely frequent drives belonging to a cluster. The prediction result improves significantly, to almost 100% ($\sim$97%), when a prediction result is considered correct when lying within the top 3 predictions.

The differentiation between the different drive contexts is relatively accurate ($\sim$ 89% respectively $\sim$97% for top 3 matches). Moreover, in Figure 8 one can see that, when considering all drives, the main share in false predictions not lying within the top 3 matches is produced by false positives. A large fraction of false positives could be detected correctly

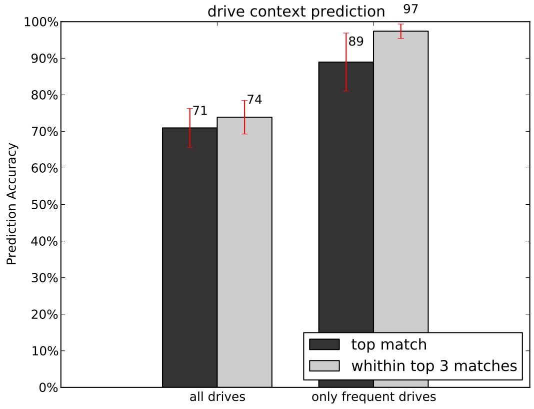

Figure 7. Prediction result for all drives and only frequent drive contexts (MinDrives=10).

($\sim$60%), but as there might be highly frequented start and stop positions like home, with overlapping context information, e.g., time and weekday, some infrequent drives were predicted as belonging to a frequent drive context.

In the evaluation we used a binary probability distribution for the day variable (workday, weekend) due to the relatively small minimal cluster size of 10 drives. It might be possible to achieve a better recognition of infrequent drives by assuming a discrete probability distribution for every day (Monday, Tuesday, Wednesday, etc.), which would also lead to time probabilities for every day for each drive context. However, this would only make sense with a higher minimal cluster size, in order to get representative probability distributions for every day.

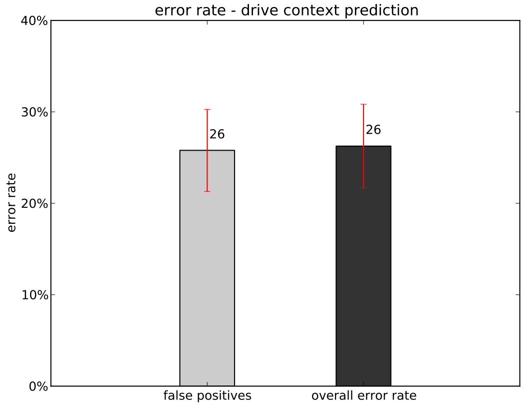

Figure 8. Overall prediction error rate and the share of false positives at the overall error rate.

Compared to the rate of false positives the rate of true negatives is extremely low and underlines the accuracy of our inference system related to the prediction of frequent drive contexts (see Figure 8). However, eliminating false positives is crucial in order to not annoy the driver with unwanted function automation and might only be solvable with little driver interaction. A solution could be providing the driver with the top 3 most likely contexts and letting the driver decide if one is appropriate for him in the current situation. If none is selected by the driver after a certain driving time the system assumes that in the current situation no function automation is wanted by the driver.

VI. Conclusion

In this paper, we investigated the detection and prediction of frequent drive contexts as an important building block for vehicle personalization. We proposed two different spatial clustering approaches for identifying frequent drive patterns in a GPS data set. Especially the route independent Start-Stop-Clustering is promising, as it is able to detect frequent drive patterns independently of the chosen route. The presented Bayesian inference systems accuracy in differentiating frequent drive contexts was about 89% respectively 97% for a top 3 match. Future work will consist of linking context information and adaptive function automation together, as well as in in-car field and acceptance tests.

Acknowledgment

The authors would like to thank the participants of the study and BMW Group for providing the drive data. All location and drive data was anonymized to ensure the participants' privacy.

References

[1] G. D. Abowd et al., “Towards a better understanding of context and context-awareness,” in Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing, ser. HUC ’99. London, UK, UK: Springer-Verlag, 1999, pp. 304–307.

[2] A. Schmidt, “Ubiquitous Computing - Computing in Context,” Ph.D. dissertation, Lancaster University, November 2002.

[3] O. Mazhelis, “Real-time recognition of personal routes using instance-based learning,” in IEEE Intelligent Vehicles Symposium (IV 2011), 2011, pp. 619–624.

[4] J. Froehlich and J. Krumm, “Route prediction from trip observations,” in Proceedings of the Society of Automotive Engineers (SAE) 2008 World Congress, SAE Technical Paper 2008-01-0201, April 2008, pp. 1–13.

[5] D. Tiesyte and C. S. Jensen, “Similarity-based prediction of travel times for vehicles traveling on known routes,” in Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ser. GIS ’08. New York, NY, USA: ACM, 2008, pp. 14:1–14:10.

[6] A. Brilingaite and C. S. Jensen, “Online Route Prediction for Automotive Applications,” in Proceedings of The 13th World Congress and Exhibition on Intelligent Transport Systems and Services (ITS 2006), London, October 2006, pp. 1–8.

[7] K. Torkkola, K. Zhang, H. Li, H. Zhang, C. Schreiner, and M. Gardner, “Traffic Advisories Based on Route Prediction,” in Proceedings of Workshop on Mobile Interaction with the Real World, 2007, pp. 33–36.

[8] O. Mazhelis, I. Žliobaite, and M. Pechenizkiy, “Context-aware personal route recognition,” in Proceedings of the 14th international conference on Discovery science, ser. DS’11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 221–235.

[9] M. Vlachos, G. Kollios, and D. Gunopulos, “Discovering similar multidimensional trajectories,” in Data Engineering, 2002. Proceedings. 18th International Conference on, 2002, pp. 673–684.

[10] K. Laasonen, “Route Prediction from Cellular Data,” in Proceedings of the Workshop on Context-Awareness for Proactive Systems (CAPS). Helsinki, Finland: University Press, 2005, pp. 147–158.

[11] K. Tanaka, Y. Kishino, T. Terada, and S. Nishio, “A destination prediction method using driving contexts and trajectory for car navigation systems,” in Proceedings of the 2009 ACM Symposium on Applied Computing, ser. SAC ’09. New York, NY, USA: ACM, 2009, pp. 190–195.

[12] D. Ashbrook and T. Starner, “Using gps to learn significant locations and predict movement across multiple users,” Personal Ubiquitous Comput., vol. 7, no. 5, Oct. 2003, pp. 275–286.

[13] J. Krumm and E. Horvitz, “Predestination: Inferring destinations from partial trajectories,” in In Ubicomp, 2006, pp. 243–260.

[14] C. Zhou, N. Bhatnagar, S. Shekhar, and L. Terveen, “Mining personally important places from gps tracks,” in Data Engineering Workshop, 2007 IEEE 23rd International Conference on, 2007, pp. 517–526.

[15] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen, “Discovering Personally Meaningful Places: An Interactive Clustering Approach,” ACM Trans. Inf. Syst., vol. 25, no. 3, July 2007.

[16] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello, “Extracting places from traces of locations,” in Proceedings of the 2Nd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots, ser. WMASH ’04. New York, NY, USA: ACM, 2004, pp. 110–118.

[17] J. B. Macqueen, “Some methods of classification and analysis of multivariate observations,” in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.

[18] M. Ester, H.-P. Kriegel, J. S, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise.” AAAI Press, 1996, pp. 226–231.

[19] M. Lewandowsky and D. Winter, “Distance between sets,” in Letters to nature. nature publishing group, 1971, pp. 34–35.

[20] J. H. Friedman, J. L. Bentley, and R. A. Finkel, “An algorithm for finding best matches in logarithmic expected time,” ACM Transactions on Mathematics Software, vol. 3, no. 3, September 1977, pp. 209–226.

[21] R. W. Sinnott, “Virtues of the Haversine,” Sky and Telescope, vol. 68, no. 2, 1984, pp. 159+.

[22] J. Han, M. Kamber, and A. K. H. Tung, “Spatial clustering methods in data mining: A survey,” in Geographic Data Mining and Knowledge Discovery, Research Monographs in GIS, H. J. Miller and J. Han, Eds. Taylor and Francis, 2001, pp. 201–231.

[23] D. W. Scott and S. R. Sain, ”Multi-Dimensional Density Estimation”. Amsterdam: Elsevier, 2004, pp. 229–263.