2019 IEEE Intelligent Transportation Systems Conference (ITSC)
Auckland, NZ, October 27-30, 2019

Context Discovery for Personalised Automotive Functions

Christoph Segler[1], Stefan Kugele[2], and Alois Knoll[2]

***Abstract*— *Context*: With the growing prevalence of machine learning applications within the automotive domain, we observe an increase of built-in sensors and generated data. This data includes complex data such as point clouds for the environmental model but also less sophisticated data like temperature and road type. When developing *data-driven functions* there is often no need for additional sensors since a huge amount of data is already sensed, but often hidden. *Aim*: We aim to discover vehicle signals that describe a system's or user's behaviour when interacting with vehicle functions. These signals could directly serve as *input* for data-driven vehicle functions. *Method*: Based on supervised feature selection algorithms we propose an approach to discover these vehicle signals. This approach can either be deployed in the backend on test fleet data or onboard the vehicle. *Results*: Based on seven test cases, we evaluated the approach with 17 feature selection algorithms on eight customer vehicle data sets. To evaluate the resulting signal subsets, we trained machine learning models that in turn were able to predict the behaviour of the user. *Conclusion*: These trained models achieved high accuracy in the prediction, which shows that current vehicles already collect enough data to predict the user's behaviour and the proposed approach was able to discover the appropriate vehicle signals. Considering the huge amount of data and vehicles as well as the highly diverse behaviour of every user, a scalable discovery approach considering every user is inevitable.**

I. Introduction

Machine learning is a highly emerging field, with a variety of applications. In the automotive domain, this trend drives further development of automated driving, new vehicle functions, and integration of new sensors. This results in more generated, processed, and collected vehicle signals. This sensory data does include not only complex data such as point clouds for the environmental model but also less sophisticated data like temperature, torque, and road type. This enables the development of new, mostly data-driven functions. Often no additional sensors are required since a considerable rich source of data is already present and just needs to be used. These *data-driven functions* range from (i) data-driven optimisation of vehicle functions, (ii) predictive maintenance, (iii) anomaly detection, and (iv) context-aware functions that adapt to the user's behaviour. (In this work we will use the term *user* instead of *driver*, since the approach is extendable to any passenger in the vehicle. Moreover, in a fully automated scenario, users will only occasionally be drivers.)

All these applications rely on vehicle data and are executed within the automotive electric/electronic (E/E) architecture. Today's E/E architectures are highly cost optimised, federated, highly integrated, and not optimised for data analytics or machine learning applications. We identified the following *challenges* when developing *data-driven functions*:

(C1) Current vehicles generate up to 12.000 signals (not including any radar or camera data) and it is not reasonable to use all signals as input for a single data-driven function. The high number (i. e. high dimensionality) of signals accessible in the vehicle leads to the *curse of dimensionality* [1]. This *curse* relates the problem of having a too large amount of data that may contain irrelevant, redundant, or not suitable data yielding a reduced accuracy and training efficiency. This does not only affect algorithms, but also the *function developer* who has to select the appropriate input signals for the data-driven function, and therefore, might miss relevant signals which are describing the system's or user's behaviour.

(C2) Another challenge derives from the high volume of data. Currently, each vehicle produces up to 25 GiB/h of data [2]. Limited resources for storing and transmitting data shows the need for an efficient data selection strategy.

(C3) Another challenge arises when accessing data in the vehicle. Due to the highly federated structure of the E/E architecture, there is no central point for data retrieval and analytics. However, for most algorithms, data needs to be aggregated from a large number of electronic control units (ECUs) to a central point.

(C4) Additionally, vehicle signals arrive asynchronously with different sampling rates. Most machine learning algorithms, however, require synchronous input data, which becomes more complex the more signals are being used as input for a data-driven function.

In this paper, we present an approach to minimise these effects and help function developers to discover all required signals relevant for a data-driven function under development. This approach can either be executed in the vehicle itself or in the backend. To evaluate the approach and to select the most suitable algorithm, we evaluate 17 state-of-the-art feature selection algorithms (cf. Table IV) on real customer vehicle data.

For this work we pose the following research questions:

RQ1 How to identify vehicle signals which are describing a user's or system's behaviour?

RQ2 How to assist the function developer to discover these vehicle signals which are already measured within high dimensional vehicle data?

RQ3 How to scale this over large vehicle fleets considering the characteristics of automotive E/E architectures?

This paper provides the following scientific contributions, but also contributes to the state of practice:

[1]BMW Group Research, New Technologies, Innovations, Garching bei München, Germany christoph.segler@bmwgroup.com

[2]Technical University of Munich, Department of Informatics, Garching bei München, Germany stefan.kugele@tum.de, knoll@in.tum.de

978-1-5386-7024-8/19/$31.00 ©2019 IEEE

(i) Approach for the discovery of vehicle signals which are describing a user's or system's behaviour within high dimensional automotive data.
(ii) Deployment strategy based on a categorisation of signal subsets in automotive vehicle fleets.
(iii) Evaluation of state-of-the-art feature selection algorithms on high dimensional automotive signal data.

The remainder of this paper is structured as follows: Section II summarises related work followed by the primary approach of this work in Section III. Section IV presents the evaluation and gained results followed by their discussion in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

With the advance of intelligent systems, customer demand adaptive and data-driven functions which learn according to their behaviour. These data-driven functions capture hidden knowledge for continuously improving their capabilities and provide a highly personalised user experience and are known as *Context Aware Systems* [3]. Their fields of application in the automotive domain range from intelligent window levers [4], intelligent in-car infotainment systems [5], intelligent suspension controls [6] to intelligent cruise controls [7]–[10]. The mentioned works solely focus on the implementation of a specific function and only use a small number of hand-picked vehicle signals as input. To the best of our knowledge, we are not aware of any related work that focuses on discovering relevant vehicle signals for a variety of vehicle functions and that considers the characteristics of automotive E/E architectures.

Feature selection algorithms are widely used to automatically select appropriate input data for data-driven functions in the field of machine learning/data-mining and to resolve the curse of dimensionality (cf. **C1**). Hall gives one of the most intuitive definitions for feature selection: "Feature subset selection is the process of identifying and removing as much irrelevant and redundant information as possible. This reduces the dimensionality of the data and may allow learning algorithms to operate faster and more effectively" [11]. A wide range of applications where large amounts of irrelevant or redundant information are present use this method. In early applications, only a few domains used more than 40 features [12]. There was a rapid change, and nowadays, data mining applications deal with up to hundreds of thousands of features [13]. Another typical application of feature selection besides machine learning is gene subset selection in medical applications. Here, algorithms find a subset of marker genes for the identification of diseases (e. g. [14]). Other applications use feature selection in a variety of domains: Distributed P2P networks [15], data mining for vehicle maintenance [16], or context reasoning [17].

In the last years, a vast variety of algorithms has been developed. These algorithms can be categorised into *filter*, *wrapper*, or *embedded methods* [18], [19]. *Filter methods* act as a filter before the actual machine learning task. They process all features, rank them and only forward the selected features. *Wrapper methods* work similar to the filter methods with the exception that the output performance of the machine learning algorithm is used for ranking. *Embedded methods* are directly embedded into the machine learning algorithm.

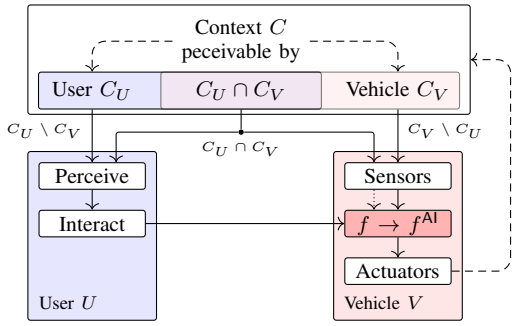

Fig. 1: General approach

In our case, we do not have a machine learning system in place yet, so only *filter methods* are suitable for the proposed approach. These methods can run without a machine learning system generating output and will provide a subset of relevant signals as a result. This subset can then be directly read by a *function developer* and allows integration of these specific signals into the *data-driven function*.

III. Approach

The general idea of the presented approach is depicted in Fig. 1. The *perceivable context* is at the core of the approach: the both *user* U and *vehicle* V can perceive information from the context. In this work, we define context according to the definition of Dey et al.: "Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves" [20].

We distinguish between different *types* of perceivable context. Table I gives a comprehensive overview including examples for each type: On the one hand, the user can perceive a specific set of information from the context upon which actions are taken (C_U). On the other hand, the vehicle can perceive a set of information from the context with built-in sensors (C_V). This context information does not only include environmental information but also information about the vehicle's interior, including vehicle function states. The context perceived by the user is not necessarily similar to the set of context information perceivable by the user ($C_U \setminus C_V$)

TABLE I: Types of perceivable context

Context set	Description and example
C_U	Information the user is able to perceive. *Example:* The user is able to perceive the current type of road the vehicle is travelling on.
C_V	Information the vehicle is able to perceive. *Example:* The vehicle measures the current speed of the vehicle with built-in sensors.
$C_U \setminus C_V$	Information which only the user is able to perceive and cannot be perceived by any vehicle sensor. *Example:* The emotions of the co-driver can only be perceived by other passengers and not by the vehicle.
$C_V \setminus C_U$	Information that only vehicle sensors are able to perceive and that cannot be perceived by the user. *Example:* Night vision systems in vehicles can detect persons or other objects at night, which the user cannot "see".
$C_U \cap C_V$	Information that can be perceived by the user and the vehicle. *Example:* The user "feels" the temperature within the vehicle and the vehicle itself can measure it with a built-in sensor.

and vice versa ($C_V \setminus C_U$).

In this work, we focus on interactions of the user with a specific vehicle function f based on information the user is perceiving from the context (C_U). The vehicle function f then controls specific actuators based on user interactions and vehicle sensors, which in turn, changes the vehicle's context and the environment.

In the initial state of the discovery phase, the function f only uses sensor input predefined by the function developer. Hence, only this particular part of the context is being used as input. However, it is not necessarily overlapping with the context the user is perceiving. The idea of the approach is to identify the *subset* of context information $C_U \cap C_V$, on which users are making their decisions and coincidentally which is already measured by built-in vehicle sensors from other functions and are not yet used by the corresponding function f. This *subset* $C_U \cap C_V$ is then used as an additional input for the function f and allows this function to learn the user's behaviour and proactively act based on the learned behaviour. We then refer to this function as *intelligent function* f^{AI}. Vice versa when discovering signals describing a system's behaviour we observe the interaction of the system with a vehicle function instead of the user's interaction.

A. Signal Discovery

In order to discover the afore-mentioned signal subset $C_U \cap C_V$, we use supervised filter feature selection algorithms. These algorithms rank all available vehicle signals according to their correlation to a predefined *label*. A *label* is the state of the function f for which the developer wants to identify all relevant signals to describe the user's or system's behaviour (e. g. seat heating *on* or *off*). Based on this ranking, the signal subset $C_U \cap C_V$ is identified. Only the best-ranked signals will then be used as input for f^{AI}.

B. Signal Subset Processing

After the identification of the signal subset $C_U \cap C_V$ it can be used in two different ways: (i) As insight for the function developer for manual development of an intelligent function f^{AI} or a *non-data-driven* function f, or (ii) as automatic input for an intelligent function f^{AI}.

In the first case, the signal subsets are manually integrated into f or f^{AI}. Additional expert knowledge is used during this manual process to optimise the selected input further; however, this will not scale for signal subsets varying for each user. In the automatic case, no manual assessment of the selected signal subset is required and the signal subset is automatically used as input for f^{AI}. This allows very user-specific personalisation, but lacking the manual verification by the developer the input of the resulting function is not fully transparent at design-time and therefore requires additional verification steps within the function f^{AI}.

C. Signal Subset Types & Algorithm Deployment

For this approach, we distinguish between three different types of *signal subsets* $C_U \cap C_V$. These subset types range from intuitive correlations of a system to very user-specific correlations. This classification is crucial for the *deployment* of the approach.

1) Signal Subset Types: We categorised the signal subset types into the following three categories:

(system) These correlations hold true for *all* users/systems. For example, the steering torque of the steering wheel is only related to the physics of the driving dynamics and not to any user behaviour. These correlations are similar among all users.

(group) Correlations of this type are only true for a *group of users*, and not for all users. In the data of a field study (cf. Section IV), we could for example identify for a group of users a correlation between the *driving dynamics control* (i. e., *sport*, *comfort*, or *eco*) and whether a co-driver is present or not.

(user) These correlations are very user-specific and at the core of *personalisation*. An example is parallel usage of several functions (indicated by the respective signals) at the start-up of the vehicle, i. e., which functions are triggered when the user is commuting from home to work.

2) Algorithm Deployment: For every intelligent function f^{AI}, the signal subset type classification has to be done by the developer and is relevant for the deployment of the approach. The simplest deployment is an execution on recorded test fleet traces in the backend. These traces are already present in the OEMs (Original Equipment Manufacturer) backend and are collected from test fleet vehicles which are equipped with data loggers. By using this data, no additional cost for the vehicle and the data collection are added. Any algorithm can be deployed in this case, as computing resources in the backend can be considered as nearly unlimited. Performing first analyses on this data allow early signal discovery and fast development, but will not cover any user-specific behaviour. In this case, only signal subsets of the type system can be identified, due to the partly unrealistic interactions of test drivers with vehicle functions. Only collecting data from test vehicles and not from real users can be considered is highly privacy preserving for the user as no actual user data is stored or processed.

To capture signal subsets of the type group or user algorithms need to operate on real user data. The transmission of all data of all vehicles would be highly inefficient and very costly, even if technically possible (cf. **C2**), but will only add slightly more complexity to the vehicle itself. The transferred amount of data can be minimised by only collecting data of a subset of users, but will also lead to less accurate results by not capturing the behaviour of all users. This will only allow discovering signal subset of the type group and the results will not fully generalise the behaviour of all users.

By deploying the algorithms directly onboard the vehicle, no actual user data needs to be transmitted to the backend and the data collection is much more efficient. However, due to the deployment in the vehicle, additional computational resources have to be added to the vehicle and restrict the applicable algorithms. With the deployment within the E/E architecture, we also need to consider the characteristics of this architecture

TABLE II: Deployment comparison

Deployment / Data	Backend Test Fleet	Backend Set of users	Onboard Set of users	Onboard All users
Complexity vehicle	● ● ●	○ ● ●	○ ○ ●	○ ○ ●
Complexity data collection	● ● ●	○ ○ ●	○ ● ●	○ ● ●
Privacy preserving	● ● ●	○ ○ ●	○ ● ●	○ ● ●
Algorithm complexity	● ● ●	● ● ●	○ ○ ●	○ ○ ●
Early validation	● ● ●	○ ○ ○	○ ○ ○	○ ○ ○
Type system	✔	✔	✔	✔
Type group	✖	✔	✔	✔
Type user	✖	✖	✖	✔

Legend: Good: ● ● ●, notable: ○ ● ●, moderate: ○ ○ ●, poor: ○ ○ ○; Signal subset type covered: ✔, not covered: ✖.

(cf. **C3**, **C4**). A significant advantage of this deployment is its privacy-preserving nature by not transmitting any user-related data and only transmitting the resulting signal subsets. When trying to identify highly personalised signal subsets of the type user a deployment in the vehicle is inevitable. If we additionally consider the case of an automatic selection of input for f^{AI}, a deployment in the backend would also add communication overhead by transmitting data back and forth between backend and vehicle.

Table II provides a rating of the pros and cons of the before mentioned deployment types. The decision where and to which extent the signal discovery should be performed is highly dependent on the intelligent function f^{AI} and cannot be answered in a general manner. This decision has to be done by the developer with expert knowledge for their function.

IV. Evaluation and Results

In order to evaluate our approach, we have chosen seven exemplary test cases which we evaluated on real vehicle data sets from eight customer vehicles by using 17 state-of-the-art feature selection algorithms. For the evaluation of the resulting signal subsets, we trained and evaluated a Support Vector Machine (SVM) classifier on each data set, test case and algorithm. The evaluation was performed on a workstation (Intel Core i7-5930K, 64 GiB RAM), but would not differ from an implementation in the backend or in the vehicle itself.

A. Evaluation Environment

The evaluation environment is described in the following:

1) Data Sets: The used vehicle data sets were collected during a field study which has been conducted at the BMW Group. These data sets originate from vehicle traces logged from current generation BMW 7 Series (G11) and contain all vehicle data frames which were sent over the internal vehicle communication networks. These vehicle signals were decoded, scaled in the range of $[0, 1]$, and sampled with a sampling rate of 30s, resulting in the data sets shown in Table III. The number of samples per data set varies between around 5.600 and 29.000. The variation originates from the different durations the cars were driven. The number of vehicle signals for all data sets is approximately 12.000 and depends on the car's options and used functions. All data sets were divided in a *training set* (first 66% of the data set) and a *test set* (last 33% of the data set). The *training set* is solely used to identify the signal subsets and training of the SVM classifier, whereas the *test set* is used to evaluate the classifier including the signal subsets. This allows a clear distinction between the signal discovery and the evaluation of the resulting signal subsets.

TABLE III: Data Sets

Data Set	Samples	Signals
Data Set 1	5652	12011
Data Set 2	28773	12491
Data Set 3	24018	11972
Data Set 4	28355	12226
Data Set 5	9572	11996
Data Set 6	6940	11855
Data Set 7	22372	12128
Data Set 8	7562	12391

2) Test Cases: As exemplary test cases for the evaluation, we selected one function that is directly controlled by the vehicle itself (system) and six functions that are controlled by the user (group or user):

(Day/Night) This first test case is based on the function that sets the display mode of the navigation system either to *day mode* or *night mode*. Only the vehicle's brightness sensor determines this mode without using any user input (system).

(ACC) This test case is based on the *Adaptive Cruise Control* (ACC), which automatically holds a predefined speed and distance to the preceding vehicle. In this test case we want to identify all correlating signals (i. e., which describe the driver's interaction with this function) if the function is either switched *on* or *off*.

(ACC Gap) Additionally, we have a dedicated test case for the predefined gap-distance (i. e., distance to the preceding vehicle) setting for the ACC. The driver can set this value from the smallest gap setting 1 to the largest gap setting 4 in steps of 1.

(DDC) This test case is based on the function *Driving Dynamics Control* (DDC). This function adjusts various driving dynamic properties (e. g. suspension, engine) and can be manually set by the driver. Possible states of this function are: *comfort*, *sport*, and *eco pro*.

(Seat Heating) For this test case, we observe the current state of the driver's seat heating; we only consider the states *on* and *off*.

(Window) This test case is based on the state of the driver's window, which can either be *opened* or *closed*.

(HVAC) For the last test case, we want to identify all signals correlating to the currently set temperature of the Heating Ventilation Air Conditioning (HVAC) system. The temperature can be set by the user from 16.0°C to 28.0°C in steps of 0.5°C.

The label for each test case is created based on the associated function signals. For each test case, all output signals of the function are removed (e. g. in the seat heating test case, the signals with the current state of the seat heating and the temperature of the seat are removed). This avoids identification of circular correlations—output signals that are directly correlated to input signals. The function developer already knows them and thus they are not considered for signal discovery. In the evaluation, these correlations would also directly affect the training of the SMV classifier and therefore distort the evaluation results of the signal subsets.

3) Feature Selection Algorithms: The presented approach heavily relies on the performance of the used feature selection algorithm. In order to exclude this impact on our evaluation, we evaluated the presented approach with 17 state-of-the-art

feature selection algorithms. The selection of the algorithms is based on the extensive survey paper by Li et al. [19]. The used algorithms are listed in Table IV. In case the algorithm requires discrete input data, we extend the preprocessing pipeline with a discretisation step for each vehicle signal (cf. Section IV-A.1). In this case, each signal is discretised into 20 equally sized bins, based on the defined range of the signals. By evaluating our approach with multiple algorithms, we can also identify the best-suited algorithms for automotive signals—at least for the presented test cases. The implementation of the feature selection algorithms is partly based on the Python package *scikit-feature* [19].

4) Evaluation Method: Based on the ranking (i. e., ordered list of scored signals) of each feature selection algorithm $\mathcal{A}$, we trained an SVM classifier ($\mathrm{SVM}_\mathcal{A}$) with the top $k = \{5, 10, 15, 20, 25, 30\}$ ranked signals to predict the function's state. We have selected this type of classifier, because of its robustness to the *curse of dimensionality* [21] and simple replication for similar evaluations.[1] The implementation of the SVM is based on the python package *scikit-learn* [22]. To evaluate the performance of the SVM classifier, we only evaluate the SVM classifier on the test set.

5) Metrics: To asses the results of the evaluation, we introduce four different metrics:

(Balanced Accuracy) The SVM's *balanced accuracy* using the signal subset of algorithm $\mathcal{A}$ is denoted by $\mathsf{bacc}(S_{|\mathcal{A}})$, where $S_{|\mathcal{A}} = \{s \in S \mid s \in \mathcal{A}(S)\}$, $\mathcal{A}(S)$ returns the top k rated signals of algorithm $\mathcal{A}$, and bacc is defined as $\mathsf{bacc}\colon 2^S \to \mathbb{R}$. We calculate the mean balanced accuracy $\mathsf{bacc}(S_{|\mathcal{A}})$ of the trained $\mathrm{SVM}_\mathcal{A}$ with the selected signal subset $S_{|\mathcal{A}}$ of the feature selection algorithm $\mathcal{A}$ on the test data over each test case. The algorithms only provide a signal ranking and no optimal number of signals to be selected. Therefore, we only consider the best performing SVM for the different sizes k of the signal subset $k = \{5, 10, 15, 20, 25, 30\}$.

(Difference) Next, we compared each trained $\mathrm{SVM}_\mathcal{A}$ with a trained SVM classifier using all signals as training input denoted by SVM. We compare the balanced accuracy $\mathsf{bacc}(S_{|\mathcal{A}})$ with the SVM classifier trained with all signals S, i. e., $\mathsf{bacc}(S)$, by computing the *difference*: $\mathsf{bacc}_\Delta \stackrel{\text{def}}{=} \mathsf{bacc}(S_{|\mathcal{A}}) - \mathsf{bacc}(S)$. We determine the mean of the differences for each algorithm and test case independently. This indicator shows whether the selected signal subset represents the given information adequately and whether $\mathrm{SVM}_\mathcal{A}$ and SVM performed similarly.

(Count) For this metric, we count the number of runs n where $\mathrm{SVM}_\mathcal{A}$ was at least performing as good as SVM trained with all signals S, i. e., $\mathsf{bacc}(S_{|\mathcal{A}}) \geq \mathsf{bacc}(S)$.

(Median Exe-Time) To illustrate the computational complexity of each algorithm, we measured the execution time t of each algorithm per run. This execution time heavily depends on the test case, the size of the data set, and the current workload on the hardware the algorithm is executed on. To exclude outliers and to asses a rough estimate of the computational complexity we only use the median execution time over all runs per algorithm, i. e., $\tilde{t}$. When assessing this execution time, we aim at getting a feeling of the applicability of the respective algorithms in an onboard execution scenario. These values are subject to correction, due to the high dependability of these values to the implementation of each algorithm.

[1] As a kernel of the SVM we used a *radial basis function kernel* with a *kernel coefficient* $\gamma = 1/n_{signals}$ and *shrinking heuristic*.

B. Results

Table IV gives a comprehensive overview of the results of the 952 runs (7 test cases, 8 data sets, and 17 algorithms) can be found in Table IV. The top three values per row are highlighted to indicate the best performing algorithms.

When comparing the balanced accuracy, we observe different values throughout all algorithms and test cases. The test cases *Day/Night* and *DDC* show the best values whereas the test cases *Seat heating*, *Window*, and *HVAC* show the worst values. When comparing the balanced accuracy $\mathsf{bacc}(S_{|\mathcal{A}})$ with the balanced accuracy $\mathsf{bacc}(S)$ (i. e. bacc_Δ), the algorithms DISR, FSCORE, FISHER SCORE, MRMR, TRACE RATIO FISHER, and CFS show the least difference bacc_Δ and the best performance over all test cases. In the test cases *ACC Gap*, *DDC*, and *HVAC* these six algorithms perform even better than the classifier based on all signals SVM, whereas in all others, they have a similar performance. Considering the number of runs n in which $\mathsf{bacc}(S_{|\mathcal{A}}) \geq \mathsf{bacc}(S)$, we see that DISR, FSCORE, FISHER SCORE, MRMR, TRACE RATIO FISHER, and CFS show the best performance throughout all test cases and data sets. Comparing the execution times only the algorithms CHI^2, FSCORE, FISHER SCORE, and TRACERATIO FISCHER had a median execution time lower than one minute. When varying k for the selected signal subset size and comparing the achieved balanced accuracy of the respective $\mathrm{SVM}_\mathcal{A}$, we observed that over all runs the best performance is achieved with a different number of selected signals k. But the values are similar to each other which results from the used machine learning algorithm (SVM) and its robustness against redundant or irrelevant inputs; here redundant or irrelevant signals in the selected signals subset.

To sum up, the algorithms FSCORE, FISHER SCORE, MRMR, TRACE RATIO FISHER, and CFS provided signal subsets that achieved good and robust results for the trained $\mathrm{SVM}_\mathcal{A}$ over all test cases. From those algorithms, only FSCORE, FISHER SCORE, and TRACE RATIO FISHER in are addition distinguished by their fast execution.

V. DISCUSSION

We conclude from the evaluation that the proposed approach is suitable for discovering vehicle signals which are capable of representing the behaviour of a system or user interacting with a vehicle function and are already measured by the vehicle ($C_U \cap C_V$). Based on the selected signal subsets, the trained models were able to predict the behaviour of the system or user and showed a similar performance compared to a model trained with the full set of signals, which would not

TABLE IV: Evaluation results – top three values per row are highlighted

Algorithm $\mathcal{A}$	**CIFE** [23]	**CMIM** [24]	**CHI²** [25]	**DISR** [26]	**FCBF** [27]	**FSCORE** [28]	**FISHER SC.** [29]	**GINI I.** [30]	**ICAP** [31]	**JMI** [32]	**MIFS** [33]	**MIM** [34]	**MRMR** [35]	**RELIEFF** [36]	**SPEC** [37]	**T RATIO** [38]	**CFS** [39]	**NONE**
Balanced Accuracy $\mathsf{bacc}(S_{\mid\mathcal{A}})$																		
Day/Night	0.93	0.96	0.97	0.97	0.89	0.97	0.97	0.97	0.96	0.96	0.94	0.97	0.97	0.96	0.50	0.97	0.97	0.96
ACC	0.59	0.65	0.72	0.76	0.68	0.76	0.76	0.74	0.69	0.60	0.72	0.59	0.78	0.74	0.50	0.76	0.78	0.77
ACC Gap	0.52	0.59	0.65	0.70	0.62	0.72	0.72	0.70	0.60	0.53	0.67	0.52	0.74	0.60	0.52	0.72	0.72	0.65
DDC	0.72	0.73	0.91	0.99	0.79	0.99	0.99	0.89	0.73	0.80	0.96	0.79	0.99	0.86	0.60	0.95	0.93	0.84
Seat heat.	0.50	0.50	0.51	0.54	0.51	0.54	0.54	0.53	0.50	0.50	0.52	0.50	0.53	0.54	0.50	0.53	0.55	0.51
Window	0.51	0.51	0.61	0.55	0.57	0.59	0.59	0.62	0.51	0.51	0.52	0.51	0.62	0.57	0.50	0.59	0.63	0.59
HVAC	0.32	0.37	0.25	0.42	0.29	0.45	0.45	0.39	0.37	0.38	0.24	0.37	0.37	0.34	0.26	0.41	0.39	0.27
Difference bacc_{Δ}																		
Day/Night	-0.03	0.00	0.01	0.01	-0.07	0.01	0.01	0.02	0.00	0.00	-0.01	0.01	0.02	0.00	-0.46	0.01	0.01	–
ACC	-0.18	-0.12	-0.05	0.00	-0.09	-0.01	-0.01	-0.03	-0.08	-0.17	-0.05	-0.18	0.01	-0.03	-0.27	-0.01	0.01	–
ACC Gap	-0.13	-0.06	0.01	0.05	-0.03	0.07	0.07	0.05	-0.05	-0.12	0.02	-0.13	0.09	-0.05	-0.13	0.07	0.08	–
DDC	-0.13	-0.12	0.06	0.14	-0.05	0.14	0.14	0.05	-0.12	-0.05	0.12	-0.05	0.14	0.02	-0.24	0.10	0.08	–
Seat heat.	-0.01	-0.01	0.00	0.03	0.00	0.03	0.04	0.03	-0.01	0.00	0.02	-0.01	0.03	0.03	-0.01	0.02	0.04	–
Window	-0.09	-0.09	0.02	-0.04	-0.03	0.00	0.00	0.02	-0.09	-0.09	-0.07	-0.09	0.02	-0.03	-0.09	0.00	0.03	–
HVAC	0.05	0.10	-0.02	0.15	0.03	0.18	0.19	0.12	0.10	0.12	-0.03	0.10	0.10	0.07	-0.04	0.15	0.12	–
Count n																		
Day/Night	3	3	3	3	2	3	3	3	3	3	3	3	3	2	0	3	3	–
ACC	0	0	4	6	2	6	6	5	0	0	4	1	5	3	0	6	5	–
ACC Gap	1	3	3	5	2	6	5	4	3	1	4	1	7	2	1	5	7	–
DDC	4	4	5	7	3	6	6	6	4	5	6	6	6	6	3	6	6	–
Seat heat.	0	0	1	5	1	3	4	2	0	0	2	0	3	2	0	4	5	–
Window	0	0	5	1	3	4	4	4	0	0	0	0	4	2	0	4	6	–
HVAC	6	8	6	8	8	8	8	8	8	8	6	7	6	8	5	8	7	–
Sum	14	18	27	35	21	36	36	32	18	17	25	18	34	25	9	36	39	–
Exe-Time $\bar{t}$	4.0h	4.0h	0.3s	6.9h	27m	0.4s	21s	5m	4.0h	4.2h	4.0h	4.1h	3.9h	6m	29m	26s	14.8h	–
Algorithm $\mathcal{A}$	**CIFE**	**CMIM**	**CHI²**	**DISR**	**FCBF**	**FSCORE**	**FISHER SC.**	**GINI I.**	**ICAP**	**JMI**	**MIFS**	**MIM**	**MRMR**	**RELIEFF**	**SPEC**	**T RATIO**	**CFS**	**NONE**

be possible in a vehicle due to restrictions of the vehicles E/E architecture (cf. Section I). Even if it would be possible, our approach gives the developer the possibility to gain insights into the used input of the intelligent function f^{AI} and helps to further optimise this function. Moreover, in all test cases, the trained model achieved a high balanced accuracy, which shows that current vehicles already collect enough data to predict the system's or user's behaviour—even with a basic machine learning algorithm (SVM).

The feature selection algorithms FSCORE, FISHER SCORE, and TRACE RATIO FISHER showed the best performance in the signal discovery process and the lowest computational complexity. This is required for onboard use and also desired for backend applications. These algorithms show due to a similar scoring calculation similar results. A drawback of these algorithms is that they only asses each signal by its own, and do not consider any interaction among signals. This can lead to redundant signals not being identified and cross-correlation between signals not being discovered. But in the case of an onboard execution, this can also be seen as the significant advantage of being able to distribute the scoring of each signal to the point where the signal is generated, without generating additional communication overhead.

Referring to the limited resources for storing histories of vehicle signals values, we showed in our previous work [40], [41] a modification of the FISHER SCORE algorithm which was able to rank vehicle signals without the need to store histories of signal values.

To sum up, we propose to deploy the discovery process directly in the vehicle, considering the best results shown by algorithms with low computational complexity and the advantages of an onboard deployment (cf. Section III-C.2). In the case of early development, we additionally propose to deploy the process on test fleet data, to gather first insights—even if no real user behaviour is covered in this case.

Threats to the Validity: A major threat to the validity of this evaluation results from the selection of the dataset and test cases. There is a vast amount of in-house test vehicles data; however, we purposely used real user data to demonstrate the approach's practical feasibility. As a premium car manufacturer, we highly consider the privacy of our customers. Hence, it is challenging—even as an OEM—to get a high number of full customer car traces. Due to limited computational resources and functions accessible in the used vehicles, we had to limit this evaluation to a selection of test cases. Therefore, the results of the evaluation are only valid for these test cases. We tried to select a wide range of user functions in order to generalise the results for other use cases, but the results may vary for other use cases and users. We are aware of these issues and try to gather more data and consider additional test cases in the future.

VI. CONCLUSION

In this paper, we presented an approach to discover vehicle signals, that represent a system's or user's behaviour when interacting with functionalities and therefore can be directly be used as input for *data-driven functions* to increase their performance and which are often hidden in the high amount of vehicle data (cf. **RQ1**, **RQ2**). Based on supervised feature selection algorithms, we proposed an approach discovering these signals within the vehicle and considering the restrictions of automotive E/E architectures. By using the resulting

signal subsets, we were able to train machine learning models which were able to predict the behaviour of the system/user. The trained model showed high accuracy in the prediction of the behaviour, which shows that current vehicles already collect enough data to predict a system's or user's behaviour, but this data has to be discovered first of all. Considering the vast amount of data and vehicles as well as the highly diverse behaviour of every user, we propose to deploy the discovery approach directly in the vehicle and in the case of early development additionally on test fleet data, to gather first insights (cf. **RQ3**). These results are promising, and we try to gather more data and consider additional test cases in future work.

REFERENCES

[1] R. Bellman, *Dynamic Programming*, 1st ed., ser. Dover Books on Computer Science. Princeton, NJ, USA: Princeton University Press and Dover Publications, 1957.

[2] McKinsey & Company, "Ready for inspection - the aftermarket in 2030," 2018.

[3] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *Handheld and Ubiquitous Computing*, ser. LNCS. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, vol. 1707, pp. 304–307.

[4] H. E. Byun and K. Chaverst, "Utilizing context history to provide dynamic adaptations," *Applied Artificial Intelligence*, vol. 18, no. 6, pp. 533–548, 2004.

[5] S. R. Garzon, "Intelligent in-car-infotainment systems: A contextual personalized approach," in *2012 8th International Conference on Intelligent Environments (IE)*, 2012, pp. 315–318.

[6] J.-S. Chiou and M.-T. Liu, "Using fuzzy logic controller and evolutionary genetic algorithm for automotive active suspension system," *International Journal of Automotive Technology*, vol. 10, no. 6, pp. 703–710, 2009.

[7] M. Canale, S. Malan, and V. Murdocco, "Personalization of acc stop and go task based on human driver behaviour analysis," *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 357–362, 2002.

[8] A. Rosenfeld, Z. Bareket, C. V. Goldman, S. Kraus, D. J. LeBlanc, and O. Tsimhoni, "Learning driver's behavior to improve the acceptance of adaptive cruise control," in *24th Conference on Innovative Applications of Artificial Intelligence*. AAAI, 2012.

[9] S. Lefèvre, A. Carvalho, Y. Gao, H. E. Tseng, and F. Borrelli, "Driver models for personalised driving assistance," *Vehicle System Dynamics*, vol. 53, no. 12, pp. 1705–1720, 2015.

[10] Y. Nishiwaki, C. Miyajima, N. Kitaoka, K. Itou, and K. Takeda, "Generation of pedal operation patterns of individual drivers in car-following for personalized cruise control," in *2007 IEEE Intelligent Vehicles Symposium*, 2007, pp. 823–827.

[11] M. A. Hall, "Correlation-based feature selection for machine learning," PhD Thesis, University of Waikato, Hamilton, New Zealand, 1999.

[12] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 245–271, 1997.

[13] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, no. 7-8, pp. 1157–1182, 2003.

[14] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1/3, pp. 389–422, 2002.

[15] K. Das, K. Bhaduri, and H. Kargupta, "A local asynchronous distributed privacy preserving feature selection algorithm for large peer-to-peer networks," *Knowledge and Information Systems*, vol. 24, no. 3, pp. 341–367, 2010.

[16] B. Schlegel and B. Sick, "Design and optimization of an autonomous feature selection pipeline for high dimensional, heterogeneous feature spaces," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016, pp. 1–9.

[17] V. Könönen, J. Mäntyjärvi, H. Similä, J. Pärkkä, and M. Ermes, "Automatic feature selection for context recognition in mobile devices," *Pervasive and Mobile Computing*, vol. 6, no. 2, pp. 181–197, 2010.

[18] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[19] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Computing Surveys*, vol. 50, no. 6, pp. 1–45, 2018.

[20] A. K. Dey, "Providing architectural support for building context-aware applications," Ph.D. dissertation, College of Computing, Georgia Institute of Technology, 2000.

[21] Y. Bengio, O. Delalleau, and N. Le Roux, "The curse of dimensionality for local kernel machines," *Techn. Rep*, vol. 1258, 2005.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[23] D. Lin and X. Tang, "Conditional infomax learning: An integrated framework for feature extraction and fusion," in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 68–82.

[24] F. Fleuret, "Fast binary feature selection with conditional mutual information," *J. Mach. Learn. Res.*, vol. 5, pp. 1531–1555, 2004.

[25] Huan Liu and Rudy Setiono, "Chi2: feature selection and discretization of numeric attributes," in *Proceedings of the International Conference on Tools with Artificial Intelligence*, Anon, Ed. IEEE, 1995.

[26] P. E. Meyer and G. Bontempi, "On the use of variable complementarity for feature selection in cancer classification," in *Applications of Evolutionary Computing*, ser. Lecture Notes in Computer Science, D. Hutchison, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, vol. 3907, pp. 91–102.

[27] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *Proceedings of the 20th international conference on machine learning (ICML-03)*, 2003.

[28] S. Wright, "The interpretation of population structure by f-statistics with special regard to systems of mating," *Evolution*, vol. 19, no. 3, p. 395, 1965.

[29] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*, 2nd ed., ser. Wiley-Interscience publication. New York NY u.a.: Wiley, 2001.

[30] C. W. Gini, "Variability and mutability, contribution to the study of statistical distribution and relaitons," *Studi Economico-Giuricici della R*, 1912.

[31] A. Jakulin, "Machine learning based on attribute interactions," Ph.D. dissertation, Univerza v Ljubljani, 2005.

[32] H. H. Yang and J. Moody, "Data visualization and feature selection: New algorithms for nongaussian data," in *Advances in Neural Information Processing Systems*, 2000, pp. 687–693.

[33] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE transactions on neural networks*, vol. 5, no. 4, pp. 537–550, 1994.

[34] D. D. Lewis, "Feature selection and feature extraction for text categorization," in *Proceedings of the workshop on Speech and Natural Language - HLT '91*. Morristown, NJ, USA: Association for Computational Linguistics, 1992, p. 212.

[35] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[36] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of relieff and rrelieff," *Machine Learning*, vol. 53, no. 1/2, pp. 23–69, 2003.

[37] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning," in *Proceedings of the 24th international conference on Machine learning - ICML '07*, Z. Ghahramani, Ed. New York, New York, USA: ACM Press, 2007, pp. 1151–1157.

[38] Feiping Nie, Shiming Xiang, Yangqing Jia, Changshui Zhang, and Shuicheng Yan, "Trace ratio criterion for feature selection," in *In AAAI*, 2008, pp. 671–676.

[39] Mark A. Hall and Lloyd A. Smith, "Practical feature subset selection for machine learning," in *Computer science '98*, 1998.

[40] C. Segler, S. Kugele, P. Obergfell, M. H. Osman, S. Shafaei, E. Sax, and A. Knoll, "Evaluation of feature selection for anomaly detection in automotive e/e architectures," in *41st International Conference on Software Engineering: Companion Proceedings, ICSE 2019, Montréal, Canada, May 25 - May 31, 2019*. IEEE, 2019.

[41] C. Segler, S. Kugele, P. Obergfell, M. H. Osman, S. Shafaei, E. Sax, and A. Knoll, "Anomaly detection for advanced driver assistance systems using online feature selection," in *2019 IEEE Intelligent Vehicles Symposium*. IEEE, 2019.