Sensor Selection for Maneuver Classification

Kari Torkkola, Srihari Venkatesan, and Huan Liu

Abstract— **To determine when to present information from various devices or services to the driver of an automobile, it is necessary to determine whether a driver is engaged in a difficult driving situation that requires extensive attention. We present simulator experiments in determining what sensors make classification of driving states into such maneuvers possible, using various machine learning techniques. Our findings indicate that a small number of derived sensor signals can accomplish the task.**

I. Introduction

A. Maneuver Classification Problem

With increasing proliferation of portable, and in some cases, fixed devices in cars, there are also increasing demands to manage the distractions they present to the driver. One particular subtask is the management of timing of information presented to the driver from these devices. A simple example is having the system handle incoming cell phone calls which could be redirected to voicemail, slightly delayed, or passed to the driver to answer. An ongoing call could be interrupted. A critical component of such a system is for the vehicle to determine when the driver is in a state that she or he could take the call or could continue an existing call without being an increased risk to the traffic. Other situations are called "difficult driving maneuvers" or "critical driving events". We will use the word maneuver for these events throughout the rest of the paper. During these maneuvers non-critical information will not be presented to the driver.

Such a maneuver classifier would be one component of a more general system that monitors the driver and his environment and provides feedback—a driver assistance system. This feedback can be in the form of warning alerts, verbal suggestions, visual cues on a heads-up display, etc. A first generation system, Motorola Driver's Advocate, has been documented in [1,2].

Manuscript received April 30, 2004.

Kari Torkkola is with Motorola Labs, 2900 South Diablo Way, MD DW286, Tempe, AZ 85282, USA (phone: 602-659-6620; e-mail: Kari.Torkkola@ motorola.com).

Srihari Venkatesan and Huan Liu are with Arizona State University, Department of Computer Science, Tempe, AZ 85284. (e-mail: huanliu@asu.edu). This work was supported in part by IMES/Ford and CEINT 2004 at Arizona State University.

Since the goal is to determine whether the driver is capable of receiving new information from a device or a service, this implies estimation of the cognitive load of the driver [3,4,5,6]. Most often this is done by observing the driver directly or indirectly. In our case the aim is slightly different – detecting a driving maneuver from the sensor inputs rather than attempting to directly assess the state of the driver. This implies a higher cognitive load directly related to the actual driving situation. Most of the previous work concentrates on driver behavior recognition and prediction [7,8], rather than on driving state recognition. Recognition of typical driving events and their long-term prediction using acceleration and location sensors with neural networks is studied in [9].

In this paper we study which sensors are needed for driving state classification. Our approach is purely data-driven. Sensor data is collected in a driving simulator; it is manually labeled with maneuvers, after which the problem reduces to that of feature selection [11]: "What sensors contribute most to the correct classification of the driving state into maneuvers/non-maneuver?" Since we are working with a simulator, we have sensors available that would be hard or expensive to arrange to a real vehicle. The simulator-based approach makes it possible to study the problem without implementing the actual hardware in a real car.

We proceed by giving background to the general feature selection problem. We then describe our environment for driving simulation and data collection, including the sensor set we are studying. Feature selection experiments for the maneuver classification problem are described next, followed by actual classification experiments, results, and conclusions.

II. Feature Selection

A. Background

Feature selection is a data preprocessing technique that is extensively used in the machine learning community and has been an active area of research since 1970's. It is a process of finding a minimal subset of features that are necessary and sufficient to learn the target concept [11]. Feature selection has been shown very effective in removing redundant and irrelevant features, increasing the efficiency of the learning task, improving learning

performance like predictive accuracy, and enhancing the comprehensibility of the learned results. Numerous algorithms have been proposed in literature for feature selection [12,13,14].

As mentioned in [14,16], feature selection algorithms can be broadly classified in to two models: the filter model and the wrapper model. Filter model based algorithms propose an evaluation criterion based on the characteristics of the data that evaluates the effectiveness of the selected features. In the wrapper model a predetermined learning algorithm is used to evaluate and determine which features are selected. For each new feature subset generated, the wrapper model learns the classifier and measures its goodness (by using accuracy, recall, precision values). The features selected are those that have higher goodness measure. Although wrapper approaches give better results they are more computationally intensive than filter approaches. Hybrid models have been proposed [16,17,18] that used both filter model and wrapper model in their algorithms. In these methods, a set of features for a given cardinality is chosen based on a goodness measure that is dependent on the inherent characteristics of the data. Then cross-validation is employed to decide the final best subset across different cardinalities.

B. Random Forests for Feature Selection

Random forest is a special kind of ensemble learning technique [15]. A Random Forest grows an ensemble of classification (or regression) trees, each trained on a subset of the training data using only a subset of available features. To classify a new data sample, the data is classified through each of the trees of the forest. The final prediction is done based on majority voting among the trees.

Let us assume that the dataset has N instances and M features. Each tree in the random forest is grown on a bootstrap sample of N data points. 'm' features ($m < M$) are randomly chosen from all M features and the best split attribute among these is chosen to split the node. Random forests constructed in this manner have several nice properties. The construction process is extremely fast compared to other contemporary ensemble classification methods. It is scalable to huge datasets and to high dimensional datasets. It is also possible to inherently determine the importance of each input feature as a by-product of the forest construction process.

The importance of a feature follows from the frequency of appearance of the feature in the nodes of the trees in the forest [15]. Each decision node in the trees is chosen as the best value of the best feature that splits the current training data in that node into two parts. The goodness of the split is evaluated using information gain, which reflects how "pure" the two parts are in terms of the classes of the classification task. The measure of importance of a feature is the sum of the information gains of those nodes in the forest that split using this feature. Thus all features can be ranked based on this importance measure. The whole point in this measure is that each feature is not ranked alone but in the classification task, together with all other features.

This method of ranking the features was used in our experiments for selecting the most useful sensors for maneuver classification.

C. CFS Feature Selection Algorithm

Correlation-based Feature Selection (CFS) [24] is a filter based feature selection algorithm that uses correlation among features to select the best features for the given dataset. It evaluates the effectiveness of a feature subset by considering the individual predictive ability of each feature along with the degree of redundancy between them. It selects subset of features that have high correlation with the class but low intercorrelation between themselves.

CFS handles classification and regression problem separately. In case of a classification problem, it discretizes the continuous valued features and uses Symmetric Uncertainty to estimate the degree of association between discrete features. In the case of a regression problem the correlation between different type of features is calculated as below: Pearson's Correlation Coefficient is used to determine the correlation between two numeric features. For computing the correlation between a discrete feature and numeric feature it computes the Pearson's coefficient between the numeric feature and the individual indicator vectors for each value the discrete feature takes, and computes a weighed correlation measure using these individual correlations. When both the features are discrete, CFS computes the Pearson's coefficient for all combinations of the indicator vectors for each feature and computes a weighted average of these coefficients.

We use CFS as a benchmark against Random Forests.

III. EXPERIMENTS

A. Driving Simulator

The experiments were conducted in the Motorola driving simulator lab, which is an instrumented car in a surround video virtual world with full visual and audio simulation (although no motion or G-force simulation) of various roads, traffic and pedestrian activity. The Motorola driving simulator consists of a fixed based car surrounded by three front video (approx 150 degree forward) and 1 rear screen (approx. 50 degree rear view for center mirror and drivers side mirror). All driver controls such as steering wheel, brake, accelerator are monitored and affect the motion through the virtual world in real-time. Various hydraulics and motors provide realistic force feed back to driver controls to mimic actual driving.

The basic driving simulator software is a commercial product called DriveSafety from GlobalSim. Motorola has extended this standard product with a set of simulated sensors that, at the behavioral level, simulate a rich set of current and future onboard sensors in the near future. This

set consists of a radar for locating other traffic, a GPS system for position information, a camera system for lane positioning and lane marking, a mapping data base for road names, directions, locations of points of interest etc. There is also a complete car status system for determining the state of engine parameters (coolant temp, oil pressure etc), driving controls (transmission gear selection, steering angle, window and seat belt status etc.). The simulator setup also has several video cameras, microphones and eye tracking infrared sensors to record all driver actions during the drive that is synchronized with all the sensor output and simulator tracking variables.

The DriveSafety simulation package also has an authoring tool component called HyperDrive that can be used to easily create a virtual world consisting of any mix of road types, intersections, freeways, and traffic signals. There is also a scripting mechanism that can create other vehicles and under experimenter control have them exhibit any pre-programmed or standard behavior. It is possible to have well behaved ambient traffic on the road or we can make a car pull out in front of the subject on command. In this experiment we created a rich set of freeway and suburban roads, populated with a mix of normal and erratic vehicles, and monitored the subject driver's use of the driving controls during free form 'driving' through the world to determine when a driver was in a maneuver and when they were in normal 'cruising' mode.

The data collection system consists of 4 separate machines capturing 4 separate time stamped databases; the simulator 'host' machine, the Driver Advocate machine, the cab monitor machine and the video capture machine. After a drive is completed, the software providing the operator's console also retrieves the databases from the individual machines and places them into a collection on another dedicated database repository machine. Three of these databases represent in excess of 425 separate fields. An additional screen of video is digitally captured in MPEG2 format, consisting of a quad combiner providing 4 different views of the driver and environment. The combined databases and video produce somewhere around 400Mb of data for each 10 minutes of drive time.

The primary database is captured by our Driver Advocate machine. This database currently contains 181 categorical fields, of which 84 also have associated numerical entries. There are additional 30 special fields associated with surrounding traffic and a potentially unlimited number of events that can be generated by various other portions of the simulator system, either programmatically or manually. The data collected by this machine contains an extensive scope of driving data – information about the auto, the driver, the environment, and associated conditions. This database also contains the most important information passed forward and condensed from the simulator host and cab monitor machines.

Fig 1. The driving simulator

B. Experimental Setup

The GlobalSim HyperDrive authoring tool was used to create the driving world scenario. The virtual drive simulated a six kilometer square multi-lane beltway with on and off-ramps, overpasses, and two and three-lane traffic in each direction (separated by a grass median). Interior to the squared beltway, connecting to each mid-side overpass, were four varied two-lane roads - urban, suburban, industrial, and rural environments. These crossed in the middle at a light controlled intersection. All drivers used daytime dry-pavement driving conditions with good visibility.

For a high-density driving environment, 59 "distracter" vehicles were added to the highway scenario along with high-density random "ambient" traffic. These distracter vehicles were randomly programmed to drive between ±10 percent faster/slower than the posted speed limit, providing the impression of a steady flow of normal traffic around the subject car. All distracter vehicles simulated alert, "good" driver behavior and reacted reasonably to any particular maneuver from the subject driver. This arrangement allowed a variety of traffic conditions and road types within a confined, but continuous driving space. Opportunities for passing and being passed, traffic congestion, and different levels of driving difficulty were thereby encountered during the drive.

Data was collected from four drivers each driving about 15 minutes in the simulated world. Drivers were instructed to engage in all possible maneuvers they could think of. The data is labeled for maneuvers by manually hand annotating the data using a special purpose tool that combines video playback with graphical visualization of the selected variables from the data.

The data was labeled with the following 12 maneuvers: ChangingLeft, ChangingRight, CrossingShoulder, Not-OnRoad, Passing, Reverse, SlowMoving, Starting, Stopping, Tailgating, TurningLeft, TurningRight, and U-

Turn. The labels could be overlapping, say every Passing Maneuver is a sequence of ChangingLeft followed by ChangingRight.

C. Chosen and Derived Sensors

From many of the available features, a subset of 15 features was selected as base features that could be foreseeable to be built into a car. They are described in Table I. An economically very viable alternative to adding more sensors is to calculate new derived variables from existing sensors. Of course, one cannot add new information to the sensor signal just by processing it, but it is possible to make important information more explicit. In order to enhance the feature set, the following features were added to the base feature set:

1) Quadratic terms, i.e., all the cross products and squares, of the numeric variables
2) First order time derivatives of the numeric features.
3) Second order time derivatives of the numeric features.
4) Short-time running entropies for steering, brake, and accelerator prediction errors. Entropy is calculated as described by Boer in [25], but within a sliding window.
5) Multivariable stationarity as described in [21] with delta=2 and delta=3 samples.
6) The output of a quadratic classifier trained using standard least-squares approach [22,23] with the 13 continuous valued features.

Adding theses features makes the total size of the feature set to 138. Feature selection is applied to this dataset for selecting the most relevant sensors for maneuver classification. In addition to selecting the best *base sensors*, we also evaluate what *derived features* are relevant to this problem.

D. Results

We ran four sets of classification experiments to compare the results of two different feature selection methods using two different classifiers. In the first experiment we attempted to discover those sensors that discriminate maneuvers from non-maneuvers. The second experiment attempted to find sensors that discriminate all 12 different maneuver types from each other and from non-maneuvers. The sequence data was treated as instantaneous labeled data. Since the maneuvers can be overlapping, there could be more than one class assigned to the same instance. In order to overcome this multi-class problem, we developed a single pass algorithm that duplicates all the overlapping instances and creates one instance for each class label. The algorithm also tries to preserve the sequential nature of the input by not interleaving the class labels of overlapping instances. Third and fourth experiment concentrated in discriminating lane changes from the rest of the maneuvers and from non-maneuvers. Classification results are presented in Table II both as the accuracy and as the recall of the maneuvers. In this table, each of the four experiments was run with a Random Forest or a Naïve Bayes [26] as the classifier. Also each of the four experiments was run with six different feature sets. These feature sets are

1) All 138 original and derived features.
2) 8 top features selected by a Random Forest
3) 16 top features selected by a Random Forest
4) 32 top features selected by a Random Forest
5) As many top features selected by a Random Forest as the CFS-method selected
6) Features selected by CFS.

For the Random Forest algorithm 500 trees were constructed. As the results indicate, as a classifier, Random Forests are vastly better than Naïve Bayes. As a feature selection algorithm, RF also produces features that either perform similarly, or better than features produced by CFS. 8-16 top RF-features already perform acceptably in the general maneuver/non-maneuver classification problem. We list the top features in Table III. In the case of Random Forests, either 8, 16, or 32 top features of these were used in classification experiments. As expected, turn signal and speed are high almost on every list. It is also interesting to see some derived variables, such as stationarity of the sensors, and entropy of steering and braking ranked quite high. This indicates that to some extent new sensor hardware can be exchanged to software by computing new variables based on existing ones.

TABLE I
BASE FEATURE SET

Feature	Type*	Description
Accelerator	Cont	Normalized accelerator input value
Brake	Cont	Normalized brake input value
Speed	Cont	Speed of the subject (m/s)
Steer	Cont	Normalized steering angle
TurnSignal	Disc	Status of Indicator lights
AheadLaneBearing	Cont	Bearing of the current lane 100 meters ahead
CrossLaneAcceleration	Cont	Acceleration in the direction perpendicular to the lane (m/s^2)
CrossLaneVelocity	Cont	Velocity in the direction perpendicular to the lane (m/s)
distToRightLaneEdge	Cont	Distance to the right lane edge (m)
DistToLeftLaneEdge	Cont	Distance to the left lane edge (m)
laneOffset	Cont	Offset relative to the center of the lane
LateralAcceleration	Cont	Acceleration in the direction perpendicular to the motion of the vehicle (m/s^2)
HeadwayDist	Cont	Distance from the subject's front bumper to the rear bumper of any other vehicle ahead. (m)
HeadwayTime	Cont	Time in seconds to vehicle ahead.
VehAhead	Disc	Name of the closest vehicle in the front of the subject in the same lane as the subject.

*Type denotes the feature type indicating whether the feature is Discrete or Continuous.

TABLE II
CLASSIFICATION RESULTS

Classific-ation Method	Experiment	Feature Selection Method: All 138 Features		Random Forest 8 features		Random Forest 16 features		Random Forest 32 features		Random Forest *		CfsSubsetEval BestFirst	
		Acc	Recall	Acc	Recall	Acc	Recall	Acc	Recall	Acc	Recall	Acc	Recall
Random Forest	Man/NonMan	93.4	90.2	91.2	90.0	92.9	91.9	93.4	90.6	93.5	91.1	93.0	90.5
	All Man/ Non Man	72.5	64.2	70.5	62.9	72.6	64.9	72.5	64.3	72.5	64.4	72.6	64.1
	ChangingLeft	97.6	89.1	96.8	76.8	97.4	84.9	97.6	88.5	97.1	81.4	97.2	82.8
	ChangingRight	96.8	86.6	94.5	68.8	95.9	81.5	96.6	84.0	96.0	81.8	95.9	81.1
Naïve Bayes	Man/NonMan	79.6	55.0	79.8	70.3	79.3	52.3	80.3	54.6	80.5	55.4	80.6	56.1
	All Man/ Non Man	15.7	39.8	36.5	41.1	24.2	34.1	23.3	40.6	22.9	40.2	30.7	39.8
	ChangingLeft	23.4	95.1	91.3	43.6	89.9	43.6	86.4	67.2	90.7	45.1	88.7	53.4
	ChangingRight	20.9	97.4	90.0	41.4	89.0	43.4	89.1	44.4	88.8	43.4	88.6	45.9

* The number of features selected here is the same as the number of features given by CfsSubsetEval-BestFirst algorithm
RandomN – Top 'N' features are selected from Random Forest experiment

IV. DISCUSSION AND CONCLUSION

Previous work on sensor selection for driver assistance systems is practically nonexistent. Either real cars have been used in experiments with a very limited number of available base sensors, or simulator experimentation has concentrated on other aspects but sensor selection. Derived features have not been usually considered at all.

Driving simulator is a perfect tool for this kind of experimentation. It allows controlled collection of data, using a desired number of sensors of a desired kind.

REFERENCES

[1] D. Remboski, J. Gardner, D. Wheatley, J. Hurwitz, T. MacTavish, and R. M. Gardner, Driver performance improvement through the driver advocate: A research initiative toward automotive safety, *Proc. of the 2000 International Congress on Transportation Electronics*, SAE P-360, 2000, pp. 509–518.

[2] C. Wood, B.Leivian, N.Massey, J.Bieker, and J.Summers, Driver advocate tool, *Proc. Driver Assessment*, 2001, pp. 295–299.

[3] F. Wada, M.Iwata, S.Tano, Information presentation based on estimation of human multimodal cognitive load, *Joint 9th IFSA World Congress and 20th NAFIPS Int. Conference*, Vancouver, BC, Canada, vol.5, pp: 2924-2929, 25-28 July, 2001.

[4] Dario D. Salvucci, Predicting the effects of in-car interface use on driver performance: An integrated model approach, *Int. Journal of Human-Computer Studies*, vol. 55,pp. 85–107, 2002.

[5] Erwin R. Boer, Behavioral entropy as a measure of driving performance, *Proc. Driver Assessment*, 2001, pp. 225–229.

[6] J. Tanaka, S. Ishida, H. Kawagoe, and S. Kondo, Workload of using a driver assistance system, *Proc. of the IEEE Conf. on Intelligent Transportation Systems*, 2000, pp. 382–386.

[7] A. Liu and A.Pentland, "Towards real-time recognition of driver intentions," *Proc. of the IEEE Conf. on Intelligent Transportation Systems*, 1997, pp. 236-241.

[8] N. Oliver and A. Pentland, "Graphical models for driver behavior recognition in a smartcar," *Proc. of the IEEE Intelligent Vehicles Symposium*, 2000, pp. 7--12.

[9] D. Mitrovic, Learning driving patterns to support navigation decision making - Preliminary results, Road Safety Conference 1998, Wellington, New Zealand.

[10] D. Mitrovic, Driving event recognition by Hidden Markov Models, 4th International Conference on Telecommunications in Modern Satellite, Cable and Broad-casting Services, 1999, IEEE.

[11] K. Kira, and L. Rendell, "The feature selection problem: Traditional methods and a new algorithm", in *Proceedings of Ninth International Conference on Artificial Intelligence*, 129-134, 1992.

[12] A. L. Blum, and P. Langley, "Selection of relevant features and examples in machine learning", *Artificial Intelligence*, 1997.

[13] M. Dash, and H.Liu, "Feature Selection for Classification", *Intelligent Data Analysis: An International Journal*, 1997.

[14] R. Kohavi, and G. John, "Wrapper for Feature Subset Selection", *Artificial Intelligence*, 1997, pp. 273 – 324.

[15] L. Breiman, "Random Forests", Technical Report, University of California, Berkeley, 2001.

[16] S. Das, "Filters, wrappers and a boosting-based hybrid for feature selection", *Proceedings of the 18th ICML*, 2001.

[17] A. Y. Ng, "On feature selection: learning with exponentially many irrelevant features as training examples", *Proceedings of the Fifteenth International Conference on Machine Learning,* 1998.

[18] E. Xing, M. Jordan, and R. Carp, "Feature selection for high-dimensional genomic microarray data", *Proc. of the 18th ICML*, 2001.

[19] H. Liu, H. Motoda, and L. Yu, "Feature Selection with Selective Sampling", *Proceedings of the Nineteenth International Conference on Machine Learning*, 2002, pp. 395 – 402.

[20] L. Yu, and H. Liu, "Feature Selection for High Dimensional Data: A Fast Correlation-Based Filter Solution", *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.

[21] K. Torkkola, "Automatic Alignment of Speech with Phonetic Transcriptions in Real Time", *ICASSP*, 1988.

[22] C. M. Bishop, *Neural Networks for Pattern Recognition,* Oxford University Press, Oxford, New York, 1995.

[23] K. Torkkola, N. Massey, B. Leivian, C. Wood, J. Summers, S. Kundalkar, "Classification of Critical Driving Events", *Proc. of the ICMLA*, Los Angeles, June 23-24, 2003. pp 81-85.

[24] M. A. Hall, "Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning", *Proceedings of the Seventeenth International Conference on Machine Learning,* 2000.

[25] E. R. Boer, Behavioral Entropy as an Index of Workload, *Proceedings of the IEA 2000 / HFES 2000 Congress*. 2000.

[26] T. M. Mitchell, Machine Learning, WCB/McGraw Hill , ISBN: 0-07-04287-7, 1997.

TABLE II
FEATURES SELECTED

Random Forests				CfsSubsetEval-BestFirst	
Maneuver	**All Maneuvers (cont)**	**ChangingLeft (cont)**	**ChangingRight (cont)**	**Maneuver**	**All Maneuvers (cont)**
quadClassValue	stat2	distToRightLane_ra3_rd3_ra3_rd3	aheadLaneBearing*aheadLaneBearing	crossLaneVelocity*crossLaneVelocity	steer_ent15
speed	turnSignal	accelerator*speed	stat3	crossLaneVelocity*Headway Time	stat3
speed*speed	aheadLaneBearing*laneOffset	speed*aheadLaneBearing	accelerator*aheadLaneBearing	distToLeftLane*distToRightLane	**ChangingLeft**
stat3	speed*distToLeftLane	aheadLaneBearing	speed*distToLeftLane	distToRightLane*laneOffset	turnSignal
stat2	brake_ent15	crossLaneVelocity_ra3_rd3_ra3_rd3	speed	quadClassValue	distToRightLane_ra3_rd3
steer*aheadLaneBearing	accelerator*speed	crossLaneVelocity*distToLeftLane	distToRightLane_ra3_rd3_ra3_rd3	brake_ent15	distToLeftLane_ra3_rd3_ra3_rd3
turnSignal	crossLaneVelocity*crossLaneVelocity	aheadLaneBearing*aheadLaneBearing	aheadLaneBearing_ra3_rd3_ra3_rd3	steer_ent15	HeadwayDist_ra3_rd3_ra3_rd3
steer_ent15	steer*steer	speed*speed	speed*speed	stat3	accelerator*crossLaneVelocity
speed*distToLeftLane	aheadLaneBearing*aheadLaneBearing	stat3	turnSignal	**All Maneuvers**	accelerator*HeadwayDist
brake_ent15	aheadLaneBearing*lateral Acceleration	laneOffset_ra3_rd3_ra3_rd3	distToLeftLane	speed	speed*aheadLaneBearing
aheadLaneBearing*lateralAcceleration	aheadLaneBearing	crossLaneVelocity*HeadwayTime	lateralAcceleration_ra3_rd3	turnSignal	speed*crossLaneVelocity
aheadLaneBearing*laneOffset	steer*aheadLaneBearing	aheadLaneBearing*HeadwayDist	aheadLaneBearing*distToRightLane	aheadLaneBearing	speed*distToRightLane
steer*steer	speed_ra3_rd3	accelerator	crossLaneVelocity*crossLaneVelocity	steer_ra3_rd3	crossLaneVelocity*HeadwayDist
laneOffset*laneOffset	steer	accelerator*aheadLaneBearing	steer_ra3_rd3	distToLeftLane_ra3_rd3	crossLaneVelocity*HeadwayTime
crossLaneVelocity*crossLaneVelocity	aheadLaneBearing*distToRightLane	accelerator*HeadwayDist	aheadLaneBearing*crossLaneVelocity	HeadwayDist_ra3_rd3	stat3
distToRightLane*distToRightLane	laneOffset*laneOffset	aheadLaneBearing*HeadwayTime	stat2	brake_ra3_rd3_ra3_rd3	**ChangingRight**
distToRightLane	distToRightLane	speed*distToRightLane	accelerator*laneOffset	speed_ra3_rd3_ra3_rd3	turnSignal
accelerator*speed	speed*distToRightLane	steer*aheadLaneBearing	**CfsSubsetEval-BestFirst**	steer_ra3_rd3_ra3_rd3	aheadLaneBearing
distToRightLane_ra3_rd3_ra3_rd3	crossLaneVelocity	accelerator*HeadwayTime	**Maneuver**	aheadLaneBearing_ra3_rd3_ra3_rd3	crossLaneVelocity
speed*distToRightLane	distToRightLane*laneOffset	distToLeftLane_ra3_rd3_ra3_rd3	speed	distToRightLane_ra3_rd3_ra3_rd3	distToRightLane_ra3_rd3
speed_ra3_rd3	speed*crossLaneVelocity	accelerator*accelerator	steer	HeadwayDist_ra3_rd3_ra3_rd3	distToRightLane_ra3_rd3_ra3_rd3
distToLeftLane*distToRightLane	distToRightLane*distToRightLane	aheadLaneBearing*lateral Acceleration	turnSignal	accelerator*speed	laneOffset_ra3_rd3_ra3_rd3
laneOffset_ra3_rd3_ra3_rd3	distToRightLane_ra3_rd3	**ChangingRight**	steer_ra3_rd3	accelerator*steer	accelerator*crossLaneAcceleration
aheadLaneBearing*aheadLaneBearing	distToLeftLane	speed*crossLaneVelocity	laneOffset_ra3_rd3	accelerator*crossLaneVelocity	brake*crossLaneVelocity
distToRightLane_ra3_rd3	distToLeftLane*distToRightLane	crossLaneVelocity	brake_ra3_rd3_ra3_rd3	brake*lateralAcceleration	speed*aheadLaneBearing
distToLeftLane_ra3_rd3	distToLeftLane_ra3_rd3_ra3_rd3	crossLaneVelocity*distToRightLane	speed_ra3_rd3_ra3_rd3	speed*speed	Speed*crossLaneAcceleration
distToLeftLane_ra3_rd3_ra3_rd3	speed*aheadLaneBearing	distToRightLane_ra3_rd3	steer_ra3_rd3_ra3_rd3	speed*crossLaneVelocity	speed*crossLaneVelocity
brake*speed	**ChangingLeft**	crossLaneVelocity*distToLeftLane	aheadLaneBearing_ra3_rd3_ra3_rd3	speed*distToLeftLane	speed*distToLeftLane
aheadLaneBearing*crossLaneVelocity	speed*crossLaneVelocity	accelerator*crossLaneVelocity	distToRightLane_ra3_rd3_ra3_rd3	speed*distToRightLane	speed*distToRightLane
aheadLaneBearing*distToRightLane	crossLaneVelocity	speed*aheadLaneBearing	HeadwayDist_ra3_rd3_ra3_rd3	steer*aheadLaneBearing	speed*laneOffset
distToLeftLane	turnSignal	distToLeftLane_ra3_rd3	HeadwayTime_ra3_rd3_ra3_rd3	steer*distToLeftLane	crossLaneAcceleration*laneOffset
distToRightLane*laneOffset	accelerator*crossLaneVelocity	laneOffset_ra3_rd3	brake*lateralAcceleration	steer*laneOffset	CrossLaneVelocity*HeadwayDist
All Maneuvers	crossLaneVelocity*distToRightLane	speed*laneOffset	speed*speed	aheadLaneBearing*laneOffset	distToLeftLane*distToRightLane
speed	distToLeftLane_ra3_rd3	aheadLaneBearing	speed*distToLeftLane	crossLaneAcceleration*crossLaneAcceleration	stat3
speed*speed	distToRightLane_ra3_rd3	distToLeftLane_ra3_rd3_ra3_rd3	steer*aheadLaneBearing	crossLaneVelocity*Headway Dist	stat2
quadClassValue	speed	laneOffset_ra3_rd3_ra3_rd3	steer*laneOffset	distToLeftLane*distToRightLane	
stat3	crossLaneVelocity*HeadwayDist	crossLaneVelocity_ra3_rd3_ra3_rd3	aheadLaneBearing*lateralAcceleration	quadClassValue	
steer_ent15	laneOffset_ra3_rd3	speed*distToRightLane	crossLaneAcceleration*crossLaneVelocity	brake_ent15	

ra3 / rd3 – Running Average / Difference with window size of 3 samples; a*b – Feature *a* multiplied with Feature *b* (Cross product); a_ent15 – Running entropy of the prediction error of feature *a*; statN – stationarity with delta = N; quadClassValue – The output of the quadratic classifier