Recognizing Blind Spot Check Activity with Car Drivers Based on Decision Tree Classifier Approach

Colombiano Kedowide, Charles Gouin-Vallerand, Évelyne Vallières
LICEF Research Center, Télé-Université du Québec, Montréal, Canada
{colombiano.kedowide,charles.gouin-vallerand,evelyne.vallieres}@teluq.ca

Abstract

Blind spot check is important driving activity that is a good indicator of drivers' proficiency and vigilance. By recognizing the blind spot check activity with drivers, it is possible to quantify and qualify the proficiency of the drivers, but also to cross validate this information with other data such the fatigue level. Thus, in this paper, we present a blind spot check activity recognition system where decision tree classifiers are modeled for each drivers and are used to automatically recognize the blind spot checks.

Introduction

On one hand, driving fatigue is an important risk factor for road safety. For instance, The US National Highway Traffic Safety Administration estimates that in the US alone approximately 100,000 crashes each year are caused primarily by driver drowsiness or fatigue (United States Department of Transportation, 2005). In Canada (Quebec province), the driving fatigue is identified as one of the causes leading to an average of 104 deceased per year on the Quebec's road (SAAQ, 2010), which correspond to 20% of all death on Quebec's road.

On the other hand, since a decade, there is a growing interest in intelligent vehicle. A notable initiative on intelligent vehicles was created by the U.S. Department of Transportation with the mission of prevention of highway crashes (Emery et al., 2005). A range of new technologies allows monitoring and assisting of drivers, such as automatic speed controls or blind spot monitoring, preventing car crashes.

In this paper, we present our recent research on car driving activity recognition using machine learning techniques. This research project focuses on recognizing the blind spot check activity with car drivers, this activity being a good indicator of drivers' proficiency and vigilance. Moreover, the proposed activity recognition method will be integrated in a larger project that aims to compare driving behaviors versus fatigue level of senior car drivers. This research project, supported by the Canadian Automobile Association (CAA) Foundation, will use the drivers' behavior monitoring for (i) recognize automatically the driving activities; (ii) create specific models of drivers based on recognized activities; (iii) use the model to teach to senior drivers more road safety and fatigue self-awareness. More specifically, this paper presents in Section 2 an overview of the related work in the domain of driving activity recognition. Section 3 introduces the methodology used in our work. In Section 4, we then introduce and discuss the different ML algorithms we use for recognizing driving activities. In Section 5, we present the experimental setup we used to do our proof of concept and the related results. Finally, we conclude this paper with a discussion and a conclusion.

Related Work

Most of the work on driver activity monitoring is focused on the detection of driver alertness through monitoring eyes (Jia et al., 2002), (Jiangwei et al., 2004), (Wahlstrom et al., 2004), face, head, or facial expressions (Baker, et al., 2004), (Smith et al., 2003), (Zhu et al., 2004). In order to deal with the varying illumination, methods such as (Zhu et al., 2002) use infrared imaging in addition to normal cameras. Learning-based methods such as (Baluja et al., 1994), (Liu et al., 2002) exist for detecting driver alertness and gaze directions.

Harini Veeraraghavan et al (Veeraraghavan et al., 2005) present two different learning methods applied to the task of driver activity monitoring. The goal of their methods is to detect periods of driver activity that are not safe, such as talking on a cellular telephone, eating, or adjusting the dashboard radio system. The system presented here uses a side-mounted camera looking at a driver's profile and utilizes the silhouette appearance obtained from skin-color segmentation for detecting the activities. The unsupervised method uses agglomerative clustering to succinctly represent driver activities throughout a sequence, while the supervised learning method uses a Bayesian eigen-image classifier to distinguish between activities.

Paul Viola et al (Viola et al., 2005) have described a face detection Framework that is capable of processing images extremely rapidly while achieving high detection rates as a process for training an extremely simple and efficient classifier which can be used supervised focus of attention operator and they present a set of experiments in the domain of face detection.

Christopher J.C. Burges (Burges et al., 1998) give numerous examples and proofs of most of the key theorems and they how Support Vector machines can have very large (even infinite) VC dimension by computing the VC dimension for homogeneous polynomial and Gaussian radial basis function kernels and describes linear Support Vector Machines (SVMs) for separable and non-separable data, working through a non-trivial example in detail.

Chih-Wei Hsu et al (Hsu et al., 2003) propose a simple procedure, which usually gives reasonable results and also they do not intend to solve challenging or difficult problems and they briefly introduce SVM basics which are necessary for explaining their procedure.

Mandalapu Saradadevi and Preeti Bajak (Mandalapu Saradadevi et al., 2008) present driver fatigue detection based on tracking the mouth and to study on monitoring and recognizing yawning. The authors proposed a method to locate and track driver's mouth using cascade of classifiers proposed by Viola-Jones for faces. SVM is used to train the mouth and yawning images. During the fatigue detection mouth is detected from face images using cascade of classifiers. Then, SVM is used to classify the mouth and to detect yawning then alert Fatigue.

Driving activity recognition

Learning method for building model

Machine Learning (ML) is a well-established field of Artificial Intelligence (AI) with many accomplishments. Several approaches have been developed over the last years. Most of these approaches have focused on supervised machine learning algorithms.

During a ML process, the first step is to collect the dataset, the availability of such data is very important. The second step is to prepare and preprocess the data; it includes instance or features (attributes) selection or even the construction of new features substituting the basic ones, to reduce the dimensionality of the data and then increase the efficiency of some ML algorithms. Another objective is to handle noise and assure the feasibility of the learning despite the possible very large size of data-set.

Finally, the evaluation of learned models is most often based on prediction accuracy. Two techniques are well known: the first one consists to split the training set (2/3 for training and 1/3 for testing) and the second one is cross-validation. It is helpful when the amount of data for training and testing is limited. At the end, every instance has been used exactly once for testing. Leave-one-out is a special case of a single instance. It is of course more expensive computationally, but useful when the most accurate estimate of a learned model's error rate is required.

Decision tree classifier, which we use in this paper, is one of the most widely used supervised learning methods for data exploration, approximating a function by piecewise constant regions, and does not require previous information on the data distribution (Baker et al., 2004). Decision trees models are commonly used in ML to examine the data and induce the tree and its rules that will be used to make predictions (Smith et al., 2004). The true purpose of the decision trees is to classify the data into distinct groups or branches that generate the strongest separation in the values of the dependent variable (Zhu et al., 2004), being superior at identifying segments with a desired behavior such as response or activation, thus providing an easily interpretable solution. The concept of decision trees was developed and refined over many years by J. Ross Quinlan starting with ID3 (Interactive Dichotomizer 3) (Zhu et al., 2004) (Zhu et al., 2002). Method based on this approach use an information theoretic measure, like entropy, for assessing the discriminatory power of each attribute (Baluja et al., 1994). The most popular decision tree algorithms are grouped (Baluja et al., 1994) as (a) classifiers from the machine learning community: IDS, C4.5, CART; and (b) classifiers for large databases: SLIQ, SPRINT, and SONAR.

Weka workbench used in this research implements two of the most common decision tree construction algorithms: ID3 and C4.5 (called version J48). ID3 is one the most famous Inductive Logic Programming methods, developed by Quinlan, an attribute based machine-learning algorithm that creates a decision tree on a training set of data and an entropy measure to build the leaves of the tree.

Methodology

In our work, we opted to use J48 because it handles both nominal and numeric values while as ID3 can only handle nominal values. The classification trees were constructed using the training set data, made of several configurations, each with the same set of options, but with probably different options settings together with known class information. The procedure started by partitioning the training set for every option based on the option settings, and the resulting partition was evaluated based on how well it separates the configurations of one class from those of another.

The knowledge flow interface provides an alternative to the Explorer for the users who like thinking in terms of how data flows throughout the system, allowing the design and execution of configurations for streamed data processing (Liu et al., 2002)

Using this module, we were able to specify the data stream model (Figure 1) by opting for Weka components (data sources, preprocessing tools, learning algorithms, evaluation methods, and visualization modules) and bond them into a directed graph that processed and investigated the data. It did not read in the dataset before learning started; instead, the data source module read the input instance by instance and passed it throughout the Knowledge Flow procession.

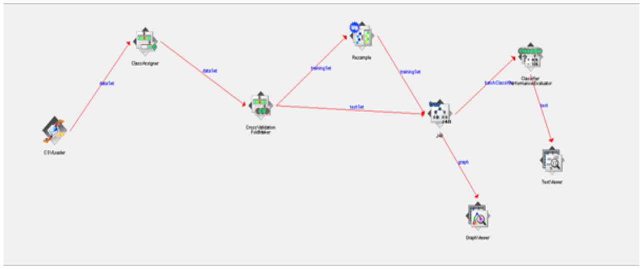


Figure 1 The graph model built

In the Knowledge Flow interface, we first created a data source by choosing the CSVLoader instrument, and then connected it to the CSV file (containing our data). In order to specify the attribute used by the class, we opted for the ClassAssigner tool connected to the DataSources through a dataset component. Following this, we chose a yes/no class for crossvalidation by J48 classifier.

The CrossValidationFoldMaker tool was attached in the data flow model to create the folds for implementing the classifier, and twice linked to the J48 component through the testSet and the trainingSet options before linking to resample filter, which adds instances to a class. The next step consisted in deciding on a ClassifierPerformance Evaluation tool and its attachment to the data flow model via a batchClassifier and set the number of folds as 3. We also needed two visualization components: TextViewer connected to ClassifierPerformance Evaluator to view the data or the models in textual form, and the GraphViewer connected to J48 classifier, in order to get a graphical representation of the decision trees resulted from the cross validation for each fold.

Experimental setup and results

The data used in the research was collected from software developed by our team, which is based on the Microsoft Kinect camera and API. This system monitors the driver's head within the car and quantify the head activity by a position into the space (x,y,z), a head yaw and a head pitch. We also quantify the level of head activity in the time, by taking into consideration the degree of modification of the three head qualifier for a period of time. The more the driver is moving his head; the more the head activity quantifier is high. Table 1 presents some sample of data collected by the head tracking software.

To acquire the required data, we installed the Kinect and our software in the LiSA laboratory car (a Nissan Versa car, Figure 2) of the LICEF research center and we ran a driving scenario in the city of Montreal for about an hour. During the driving scenario, the experimenter, seated at the back of the car, was monitoring the car driver and logging the driving activities such as looking at the blind spot, turning, braking, etc. We used this labeled information to train our model, create the decision tree classifier and validate it. All the information collected by the Kinect and the experimenter logging was merge in a Single CSV file, which was used by the Weka framework to train the classifier.

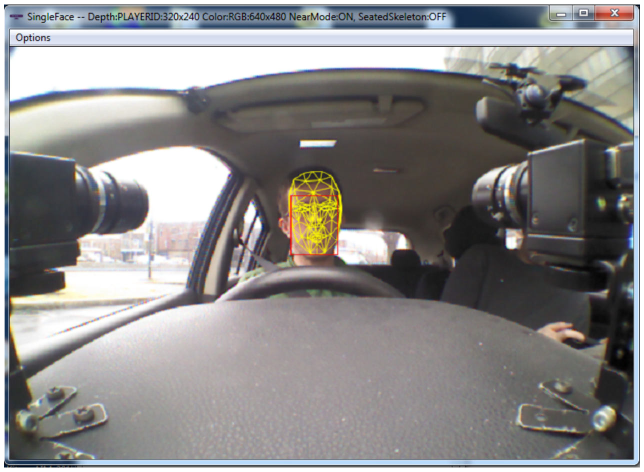

SingleFace -- Depth:PLAYERID:320x240 Color:RGB:640x480 NearMode:ON, SeatedSkeleton:OFF
Options

Figure 2 Kinect camera and head recognition system installed in the LiSA lab[1].

Table 1 Sample of data provided by the Head tracking software

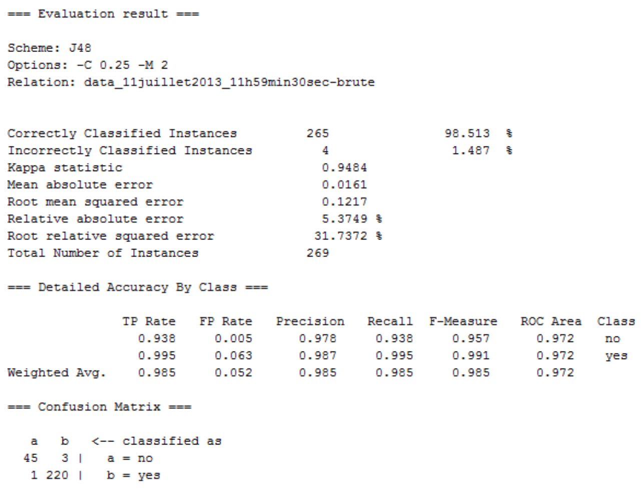
	Head Angle			Head Position			
Time	angle yaw	angle pitch	angle roll	X	Y	Z	Activity Quant.
775,127	-0,947	-0,349	0,716	-0,009	0,168	1,546	37,057
775,174	-1,196	1,140	0,880	-0,008	0,165	1,542	36,461
775,236	-1,124	1,827	0,860	-0,005	0,157	1,547	36,427
775,283	-1,329	2,655	0,733	-0,005	0,161	1,543	27,485
775,330	-2,114	0,339	0,525	-0,007	0,173	1,542	19,434
775,377	-3,463	-0,394	1,051	-0,009	0,173	1,546	19,505

[1] The camera system in the periphery of the image is not the Kinect System, which is the point of view, but the FaceLab system.

After training the classifier and applying the model, we achieved an accuracy of 98.513 %, meaning that 265 instances out of 269 were correctly classified in our model. We also obtained the values of several performance measures for numeric prediction, presented in Figure 3 and Figure 4.

The decision tree resulted from the first data set of the (Figure 1) has as a central root joint the head tracking attribute. The second level of ramification is based on angle pitch and the third and the last levels of ramification are based on angle yaw. If the angle yaw is less than or equal to 26.6512, we got a positive head tracking and that was confirmed what the confusion matrix showed.

As seen in the Knowledge Flow interface, by creating partitions through cross validation we covered a way of anchoring in the results of each partition, which the Explorer Weka's module did not present yet.



```
=== Evaluation result ===

Scheme: J48
Options: -C 0.25 -M 2
Relation: data_11juillet2013_11h59min30sec-brute


Correctly Classified Instances         265               98.513  %
Incorrectly Classified Instances         4                1.487  %
Kappa statistic                          0.9484
Mean absolute error                      0.0161
Root mean squared error                  0.1217
Relative absolute error                  5.3749 %
Root relative squared error             31.7372 %
Total Number of Instances              269

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.938     0.005      0.978     0.938     0.957      0.972    no
                 0.995     0.063      0.987     0.995     0.991      0.972    yes
Weighted Avg.    0.985     0.052      0.985     0.985     0.985      0.972

=== Confusion Matrix ===

   a   b   <-- classified as
  45   3 |   a = no
   1 220 |   b = yes
```

Figure 3 Evaluation results

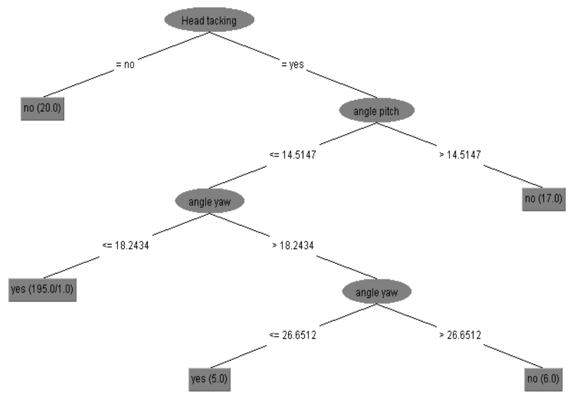


Figure 4 Decision tree

The goal of this first experimentation was to validate our approach with one driver. Of course, in the presented experimental setup, the trained classifier is dedicated to a specific driver (the participant) and more work is required to validate if differences exist between drivers' classifiers. In a larger experimentation (June 2014), which will involve 30 participants, we will have enough data to cross-validate our approach with several drivers and we will be able to test if a generalized classifier for all drivers is viable. Moreover, we plan to integrate more driving activity recognition (e.g. turning, braking, looking at the radio, etc.) to the project by using other contextual information that we are already collecting (e.g. 3-axis acceleration, GPS position or the car speed).

Conclusion

New technologies allow the monitoring of car drivers' activities and assistance while driving. By recognizing driver's activities, it is possible to actively assist the drivers in his driving tasks and use the driving information for teaching road safety and self-regulation. Looking at the blind spot is an important driving activity, which is often forgotten by drivers. Some electronic devices detect cars in the blind spot, but such devices are not common and don't replace a good driving behavior. The method proposed in this paper recognizes the blind spot check, with a cheap device (i.e. Microsoft Kinect) and in an effective way.

In our next work, we will extend the capability of our system to recognize driving activities, by integrating other driving measures such as the driving speed, the GPS location and the 3-axis acceleration of the car. With such measures, we will be able to create complete models of driving activities and use these models to assist actively drivers and use the information for teaching purpose. Ultimately, recognizing more driving activities will allow us to create Bayesian models based that are be able to predict next driving activities and will show driving tendencies for specific drivers (e.g. looking less the blind spot while turning on right corner than left corner).

Acknowledgments

We want to thank the foundation of the Canadian Automobile Foundation, Section Quebec, for funding this work through the study on the impact of the driving fatigue with senior drivers. We also want to thank Hughes Perreault for his work on the Kinect API.

References

Emery, L., Srinivasan, G., Bezzina, D., & al. 2005. *Status report on USDOT project – an intelligent vehicle initiative road departure crash warning field operational test*. Proc. 19th Int. Technical Conf. Enhanced Safety of Vehicles, Washington, DC, June

United States Department of Transportation. 2005. *Saving lives through advanced vehicle safety technology*.

Société de l'assurance automobile du Québec. 2010. Bilan routier 2010. http://www.saaq.gouv.qc.ca/rdsr/sites/files/12011001.pdf (in French)

Kim Hong, Chung. 2005. *Electroencephalographic study of drowsiness in simulated driving with sleep deprivation*. International Journal of Industrial Ergonomics. Volume 35, Issue 4, April 2005, Pages 307-320.

Jia, Q. & Yang, X. *2002. Real-time eye, gaze, and face pose tracking for monitoring driver vigilance. Real Time Imaging, 8(5):357-377, Oct 2002.*

Jiangwei, C. Linsheng, J. Lie, G. Keyou, G. & Rongben, W. 2004. *Driver's eye state detecting method design based on eye geometry feature*. In Intelligent Vehicles Symposium, pages 357-362, June 2004.

Wahlstrom, E. Masoud, O. & Papanikolopoulos, N. 2003. *Vision-based methods for driver monitoring*. In IEEE Intelligent Transportation Systems Conf., volume 2, pages 903-908, Oct 2003.

Baker, S. Matthews, I. Xiao, J. Gross,R. Kanade, T. & Ishikawa, T. 2004. *Real-time non-rigid driver head tracking for driver mental state estimation*. In 11th World Congress on Intelligent Transportation Systems, October 2004.

Smith, P. Shah, M. & da Vitoria Lobo, N. 2003. *Eye head tracking based methods-determining driver visual attention with one camera.* IEEE Transactions on Intelligent Transportation Systems, 4(4):205- 218, Dec 2003.

Zhu Y. & Fujimura, K. 2004. *Head pose estimation for driver monitoring*. In Intelligent Vehicles Symposium, pages 501-506, June 2004.

Zhu, Z. Fujimura, K. & Ji, Q. 2002. *Real-time eye detection and tracking under various light conditions*. In ETRA '02: Proceedings of the Symposium on Eye Tracking Research & Applications, pages 139-144. ACM Press, 2002.

Baluja, S. & Pomerleau, D. " Non-instrusive gaze tracking using artificial neural networks." Technical Report CMU-CS-94-102, Carnegie Mellon University, 1994.

Liu, X. Xu, F. & Fujimura, K. "Real-time eye detection and tracking for driver observation under various light conditions*." In IEEE Intelligent Vehicles Symposium, June 2002.*

Veeraraghavan, H. Atev, Bird, N. Schrater, P. & Papanikolopoulos, N. 2005. *Driver Activity Monitoring through Supervised and Unsupervised Learning. Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems Vienna, Austria, September 13-16.*

Viola. P and Jones, M.J. 2005. *Robust real-time face detection." International Journal of Computer Vision. V57i2.137-154.*

Wei Hsu, C. Chang, C.C. & Lin, C.H.. 2003. *A practical guide to support vector classification.* Technical report, Department of Computer Science and Information Engineering National Taiwan University Taipei 106, Taiwan.

Evgeniou, T. Pontil, M. & Poggio, T. 2000 *Statistical learning theory: A primer. International Journal of Computer Vision, 38(1):9-13.*

Saradadevi M. & Bajak, P. 2008. *Driver Fatigue Detection Using Mouth and Yawning Analysis*. *IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.6, June.*

Christopher J.C. Burges. 1998. *A Tutorial on Support Machines for Pattern Recognition. Data Mining and Knowledge Discovery 2, 127-167, 1998.*