Indirect Tire Monitoring System - Machine Learning Approach

O Svensson[1], S Thelin[2], S Byttner and Y Fan

Department of Intelligent Systems and Digital Design, Halmstad University, Box 823, S301 18 Halmstad, Sweden

E-mail: [1]oskarsvensson.94@gmail.com, [2]simon.thelin90@gmail.com
{stefan.byttner,yuantao.fan}@hh.se

Abstract. The heavy vehicle industry has today no requirement to provide a tire pressure monitoring system by law. This has created issues surrounding unknown tire pressure and thread depth during active service. There is also no standardization for these kind of systems which means that different manufacturers and third party solutions work after their own principles and it can be hard to know what works for a given vehicle type. The objective is to create an indirect tire monitoring system that can generalize a method that detect both incorrect tire pressure and thread depth for different type of vehicles within a fleet without the need for additional physical sensors or vehicle specific parameters. The existing sensors that are connected communicate through CAN and are interpreted by the Drivec® Bridge hardware that exist in the fleet. By using supervised machine learning a classifier was created for each axle where the main focus was the front axle which had the most issues. The classifier will classify the vehicles tires condition and will be implemented in Drivecs cloud service where it will receive its data. The resulting classifier is a random forest implemented in Python. The result from the front axle with a data set consisting of 9767 samples of buses with correct tire condition and 1909 samples of buses with incorrect tire condition it has an accuracy of 90.54% ($\pm$ 0.96%). The data sets are created from 34 unique measurements from buses between January and May 2017. This classifier has been exported and is used inside a Node.js module created for Drivecs cloud service which is the result of the whole implementation. The developed solution is called Indirect Tire Monitoring System (ITMS) and is seen as a process. This process will predict bad classes in the cloud which will lead to warnings. The warnings are defined as incidents. They contain only the information needed and the bandwidth of the incidents are also controlled so incidents are created within an acceptable range over a period of time. These incidents will be notified through the cloud for the operator to analyze for upcoming maintenance decisions.

1. Introduction

Knowledge in component failures before they occur is important for the heavy duty vehicle industry. The industry demand products that work dynamically towards the customer to reduce costs and remain efficiency in the fleet. European-Union has decided upon a law[1] that is based on environmental grounds that all new passenger cars from November 1st 2014 are equipped with a Tire Pressure Monitoring System (TPMS). The system is based upon physical sensors installed inside the tires. Explicitly heavy duty vehicles are not included in the law.

NHTSA(National Highway Traffic Safety Administration) is an agency of the executive branch of the U.S. government. With a mission to save lives and prevent injuries combined with reducing vehicle related crashes. They have published a report[2] that conclude that 12.4%

of all passenger vehicles in the U.S. of model years 2004-2011 have at least one tire that is severely underinflated as defined by FMVSS No.138[3](25% or more below what the vehicle manufacturer recommend). The report also found that 23.1 percent vehicles in the study without TPMS had at least one severely underinflated tire.

Tire monitoring is time consuming and operators have problems with scheduling proper maintenance. The global market for TPMS is projected to reach 4 billion US dollars by 2020[4]. The primary goal of TPMS is to reduce underinflation in order to make vehicles safer to operate together with improved fuel economy. The resulting classifier and cloud module shown in this article is researched and developed for Drivec AB which is a company located in Helsingborg, Sweden. It show that it is possible to create a generalized indirect Tire Pressure Monitoring System by using supervised machine learning. The system created is targeting large fleet operators with different vehicle models. It enables them to view the vehicles tire condition on each axle from a cloud service, unlike the usual tire pressure monitoring systems.

2. Related Work

Remote diagnosis, maintenance and prognosis for advanced driver assistance systems is currently a hot topic. Mostafa Anwar Taie among others have created a framework[5] which predict the remaining useful life for the prognosis of the advanced driver assistance system's safety critical components by using machine learning algorithms. This framework can be used during development phase but mainly after production.

Niclas Persson, Fredrik Gustafsson and Markus Drevö wrote a thesis about Indirect Tire Pressure Monitoring Using Sensor Fusion[6]. Markus Drevö is working at the Swedish company Nira Dynamics which is one of the leading companies in indirect tire pressure monitoring systems. This thesis explains a system based on wheel radius and vibration analysis. With both approaches combined it can detect a pressure loss of 15% in one, two, three or four tires and independently detect a pressure loss of 25% within one minute. This solution does not use any supervised machine learning methods but instead tests values against a predetermined threshold.

Stefan Byttner, Thorsteinn Rögnvaldsson and Magnus Svensson created *Consensus self-organized models for fault detection* (COSMO)[7]. COSMO is an unsupervised approach which builds up knowledge over time. It uses an onboard self-organized search for models to find relations between the vehicles data values together with an offline server application which compares the parameters of the models. A significant benefit is that the vehicles normal behavior is found under real operating conditions and not observed in a number of tests in laboratory environment. This has also formed the basis for several licentiate theses.

2.1. Industrial relation

Each vehicle in this fleet is equipped with Drivec® Bridge that fetch CAN-bus data in real time, it is a part of the existing fleet management system. When the whole implementation is done for the cloud service, the application will analyze the data in real time from the vehicles and signal when faults occur. Projects mentioned above does not have a connection to an existing fleet management system. COSMO use the Volvo Analysis and Communication Tool (VACT), that is capable of recording data using telematics to communicate remotely. The existing solutions on direct TPMS and indirect TPMS does not deliver the data to a cloud service. The information is locked within the vehicle. The application of this project will analyze the result and make it viewable for the operators regardless of the vehicles location and without having to include the driver in the decision process.

3. Methodology

Drivec AB provide expert knowledge in CAN-bus interpretation thanks to their hardware Drivec Bridge. It has enabled the research and development of the result shown in this article.

3.1. Data

Data management is handled with data obtained from the vehicle itself and from offboard measurements. The data from the vehicles contains vehicle specific information where every sample output includes an average sum and standard deviation for each parameter during a period of 10 seconds. The reason for this is that an interval of 10 seconds is the minimum amount of time required for receiving trustworthy GPS data. Offboard measurements include tire depth in millimeters and tire pressure in bar of all wheels of the vehicle. Together with a time stamp for the current date and time. From this data sets has been created to be used within the supervised machine learning process. The data sets need to contain data that is valid. To obtain validness in the data sets filtering has been done. The filtering makes sure that the received data only represent when the vehicle is driving with a constant speed without any heavy turns or heavy accelerations. It is done in both Python, for the training, and in Node.js with JavaScript for the cloud service. The parameters that the filtering is based upon are the vehicle speed, vehicle speed standard deviation, GPS speed, GPS course change standard deviation, individual wheel speed per wheel and the brake signal.

The targets are set per day and vehicle according to the offboard measurements of the vehicles tire pressure and thread depth. From the beginning the targets were decided by looking if the difference in pressure of left and right tire was greater than 0.5 bar. This would set the target as bad for the vehicle. It was discovered that 0.5 bar was too low and resulted in too many vehicles with rather small differences being classed as bad. Together with Drivec it was decided that a vehicle need to have a major difference in either tire pressure or thread depth to be classed as bad. This to avoid causing false alarms or alarms for minor issues which do not yet need to be resolved. The resulting value for deciding whether the vehicle shall be classed as bad is if it has a tire pressure difference greater than 1.5 bar or if the difference in thread depth is greater than 6mm. When the targets have been determined the training will be executed within Python where the Scikit-learn module provides the classifier. The input from the data set and targets will be used to call the fit method to generate the classifier that predicts the class to which the unseen samples belong. Random forest is used as classifier.

When solving a problem with a supervised machine learning method features are created. These features main purpose is to build a concise model from a distribution of class labels. The resulting classifier is then used to assign class labels to the testing instances where the values of the predictor features are known, but the value of the class label is unknown. The features are chosen from available parameters from the data set and need to have a physical buoyancy. The features used are related to different speed sources.

3.2. Random Forest Classifier

The random forest is an ensemble approach[8]. It is a divide-and-conquer approach used to improve performance. But with the regular tree in mind the random forest enhance this by combining trees. The tree may be referred to as weak learners but when assembling all these trees together they can possibly create a strong learner. If there exist a sample of N cases at random with a given replacement it will create a subset of the given data input. How large this subset should be may vary. At each node there should be a number X where X is the selected predictor variables at random from all the predictor variables. The predictor variable that provides the most accurate split is then later on used to perform a binary split on that specific node. When running random forest, the given input will go trough all the trees and the result may be different. It is crucial to know that if there is a large number of predictors the

predictor will be more different between the nodes. The big benefits with random forest is that it is rapid and good at handling imbalanced data.

3.3. Optimization and Evaluation

To optimize the classifier it is crucial to use different tools ensure that the outcome is as good as possible and that it behaves in the way that is best for the given situation. Confusion matrix, seen in figure 1, is used to evaluate the quality of the output of a classifier on the data set. The diagonal of the matrix represent the number of points for which the predicted label is equal to the true label. While the off-diagonal elements are those that are mislabeled by the classifier. This will result in a false negative or a false positive. A system like this requires more false positive mislabels than false negative since a false negative would trigger a false alarm.

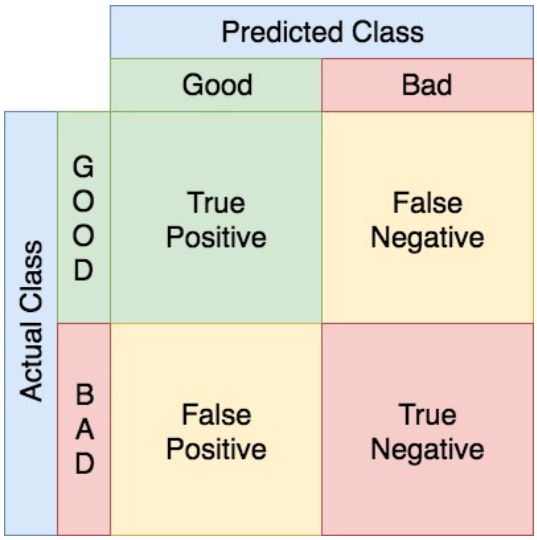

Figure 1: *Confusion Matrix*

Furthermore there is a need to evaluate how well the classifier perform on unknown data sets. This is solved by using stratified K-fold cross validation. It is a model variation technique for assessing how the result of a statistical analysis will generalize to an independent data set. K-fold cross validation is one way used to avoid over-fitting, where 90% from the unknown data set have good targets and 10% have bad targets in this case. The out-of-bag concept is used to estimate the generalization error which is the error rate of the out-of-bag classifier on the training set. An estimation of the error rate can be obtained based on the training data. This ensures that the classifier is built on about 63% of the available data and therefore pulling away 37% for testing.

3.4. Cloud

Drivec's products are all represented on a cloud service. The cloud is 100% scalable thus ensuring almost unlimited capacity in quantity of connected vehicles. The cloud provides better automatic software updates combined with a better disaster recovery. The resulting classifier from the supervised machine learning method which in this case has been the random forest. This classifier has been extracted in a format that been interpreted in the developed back-end module which is based on Node.js which becomes a process within the cloud service.

4. Results

It was concluded that the tires with the highest difference in both tire pressure and thread depth was the front axle. Therefore the focus was put to classify the tire condition of the vehicles front axle. The value of the vehicle speed that is used for the filtering was decided by analysis of the accuracy of the speed. Speeds under 30 km/h give a less reliable value hence the filtering is set to only accept data when the speed is above 30 km/h. Linearity between individual wheel speed and vehicle speed was found and resulted in a change of features in the classifier.

With a data set consisting of 9767 samples of buses with correct tire condition and 1909 samples of buses with incorrect tire condition from a total of 34 buses, the random forest classifier has an accuracy of 90.54% ($\pm$ 0.96%) after its 10 folded cross validation. After creating and analyzing a given confusion matrix. It provided shows that 10.93% of the mislabels are false negatives and 89.89% are false positives. This ensures that more bad tires will be classed as good than the other way around which will avoid false alarms on the cloud service. Incidents in the module contain the information that is displayed in the reports in the cloud service. Incidents can be created and updated depending on their state. Incidents and updates of an incident occur asynchronously on different vehicles and axles at the same time. The whole cloud process is described in figure 4. The module also has a threshold value that has been set to 50%. In figure 2 the score predictions the classifier makes is shown over time and that the classifier is proven to catch the event of going from a bad condition to a good condition. When the scoring has been equal or over 50% more than 7 times during a set period of time, an incident will be created. In figure 3 it is shown how many incidents that are created with updates to the the drastic change where no more incidents are created. To verify this the workshop was contacted and it was proven that they received a report late in the evening of March 16th from a bus driver that stated that the tires of the front axle were in critically bad condition. The tires were changed the following morning of March 17th 07:45. The project group had no information about this when it happened but it has been researched together with the data analysis from the given vehicle.

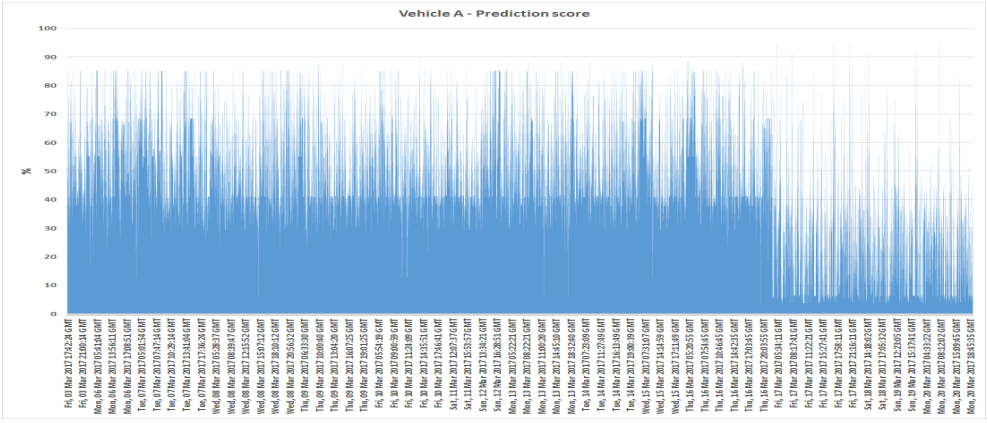

Figure 2: *The output created by the classifier from March 3rd to March 17th. The collection of outputs represent scores in percentage on the given dates. The scoring is fetched from bad to good with a distinct change on March 17th 2017. At this date the workshop changed tires without the classifier being aware of this fact.*

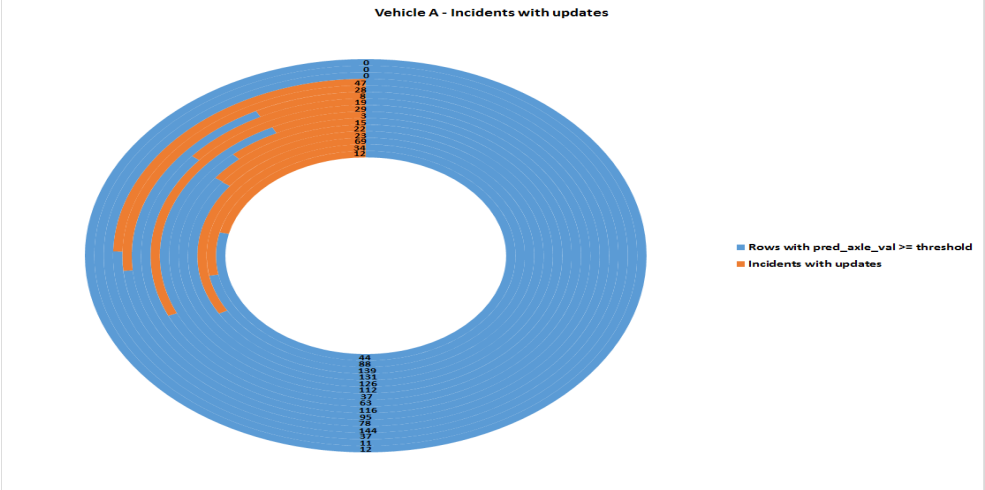

Figure 3: *Each date has a given collection of output scores above the threshold value of 50% which are represented as the blue area. During the reading of outputs, the ITMS module will perform a delay and check if the output score is above the threshold value. If the prediction meets the criteria, it will be added to the queue. If the number of bad predictions in the queue meet the given criteria an incident will be created seen as the orange area. If the number then get lower and then meet the requirement again, an update will occur for that specific incident. For example on March 3rd 44 output values were above the threshold. During the reading of the data 1 incident was created with 11 updates during that day. On March 17th no more incidents are created even though there may be values that meet the threshold value, however they do not occur close enough to each other to create an incident, which can be related to figure 2 and 4.*

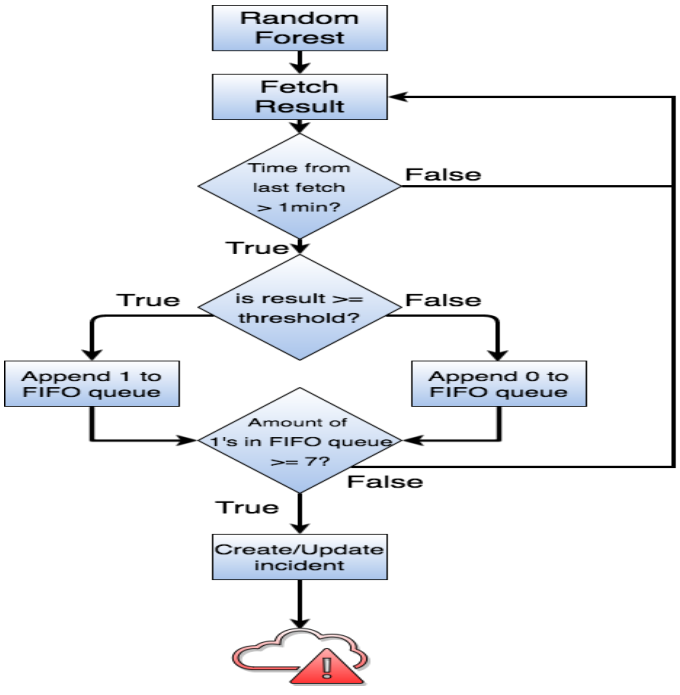

Figure 4: *Flowchart of ITMS process.*

5. Discussion

When doing the offboard measurements we measure tire pressure and thread depth. The tire pressure was measured by putting a tire pressure gauge over the valve of the tire. Some tires of the boogie and drive axle did not have valves to put the gauge on. This cause some measurements of the tire pressure to be unknown. However on the boogie and drive axle there are always two tires per side. The measuring of thread depth was focused on the center of the wheel, therefore there are no measurements for tires that are worn out at the sides, only the depth from the center is being measured.

From the offboard measurements we look at the difference in pressure between left and right wheel rather than the individual pressure of the wheel. This was considered as the better choice to start with and it also prevents users from disbelieving the first version of the system when warnings are created, rather receiving a more accurate warning for the whole axle than possibly incorrect warnings for the individual wheels. With the results from the classifier the warnings can be analyzed to create greater insight to solve problematic areas for further development when the time comes to try to pinpoint out specific wheel behaviour.

The system is currently an alpha version and is being tested internally at Drivec. For a more detailed description of the system see the Indirect Tire Monitoring System thesis [9]

6. Conclusions

This approach is using random forest as classifier and will receive real time data within a 10 second interval. The training of the classifier is done in Python and the classifier is exported to be used in a Node.js module in the cloud.

The practical study in this project is based on 34 MAN city buses with the fuel type of compressed natural gas. The buses have different lengths, 2 to 3 axles and are year models 2005 to 2011.

The European-Union law does not apply to heavy duty vehicles hence very few of them use a TPMS. It is difficult to say how well other solutions work for heavy duty vehicles but for passenger cars they work well since it has become a legal requirement. There is currently no product on the market that works in the same way as the developed tire monitoring system. The most similar product on the market is an indirect Tire Pressure Monitoring System (iTPMS) where the software use the vehicles ABS-system to receive data. Furthermore the data about the tires is kept in the vehicle and is only displayed for the driver. The most significant difference with this indirect tire monitoring system is that it will not display any information to the driver but will be a part of the fleet management system in the vehicle and show the data back-end for the operator. The solution is not oriented towards the driver of the vehicle but the vehicle itself. The driver is never exposed in the ITMS system and neither a part of the incident created. Therefore the integrity issue is put on the operator.

The most significant issue with random forest is overfitting. Random forest generally give good results which is why it is needed to be analyzed properly to avoid overfitting. Too large number of trees or too large depth of the trees is also an issue which may make the algorithm slow for real-time prediction.

Acknowledgments

We would like to thank Johan Göthe and the rest of the team at Drivec for the opportunity and the generous support throughout the whole project.

Oskar Svensson & Simon Thelin
Helsingborg 2017

References

[1] Jansen S, Schmeitz A, Maas S, Rodarius C, **2016** TNO 2014 R11423-v2 - *Final report, Study on some safety-related aspect of tyre, TNO innovation for life* 15

[2] National Highway Traffic Safety Administrationm, **2012** *Evaluation of the Effectiveness of TPMS in Proper Tire Pressure, DOT HS 811 681* 6

[3] National Highway Traffic Safety Administration, **2001** *Tire Pressure Monitoring System, FMVSS No. 138*

[4] Global Industry Analysts, Inc, **2015** *Tire Pressure Monitoring Systems - A Global Strategic Business Report*

[5] Taie M, Moawad E, Diab M, and ElHelw M, **2016** *Remote Diagnosis, Maintenance and Prognosis for Advanced Driver Assistance Systems Using Machine Learning Algorithms* 114–122

[6] Persson N, Gustafsson F and Drevö M **2002** *Indirect tire pressure monitoring using sensor fusion*

[7] Byttner S, Rögnvaldsson T and Svensson M **2011** *Consensus self-organized models for fault detection (COSMO)* 833–839

[8] Breiman L, **2001** random forests *Machine Learning* **45** 5–32

[9] Svensson O, Thelin S. *Indirect Tire Monitoring System - Machine Learning Approach* `http://urn.kb.se/resolve?urn=urn:nbn:se:hh:diva-34290`