Sequence Based Classification for Predictive Maintenance

Julio Martinez
juliomz

Christianne Dennison
cdenn

Zhengyi Lian
zylian

Abstract

PACCAR, a commercial vehicles company, has provided a dataset of sensor snapshots that were taken using communication networks during the operation of several vehicles, as well as repairs that were performed for specific failures on these vehicles. Our objective is to predict future repairs for predictive maintenance. Most approaches in the literature involve predicting repairs with only one time window. We show that in addition to the repair, time window prediction is significantly more difficult due to various inherent challenges associated with the sensor snapshot and imbalanced data. Models employed for prediction are Random Forest (industry standard), LSTM RNN, and GMM (Gaussian Mixture Models). Our findings show that while time window is intractable, repair code prediction is best at 43% recall with Random Forest.

1. Introduction

PACCAR, a commercial vehicles company, provided a dataset of field snapshots taken using communication networks during the operation of several vehicles and repair data on some vehicles (see Figure 1). The snapshots were taken asynchronously during one of four events: trip start (vehicle turning on), trip end (vehicle off), periodic (hourly on a trip), and fault (issued by vehicle's computer). These include temporal data readings of various pressures and temperatures, acceleration, lifetime idle/engine hours, bus utilization and other observations on the vehicle's condition and environment. Repair data include vehicle ID, repair date, repair cost and failure codes ATA3, ATA6 and ATA9 identifying the failure location.

PACCAR is interested in early prediction of faults that require a costly repair from relevant sensor readings for preemptive maintenance to mitigate them. We aim to apply Random Forests[5], LSTM[6] RNN[6], GMM, and compare with our previous work[8] with HMMs[2][7] on this problem, to determine the best model for predicting repairs.[1] This can help to inform the type of preemptive maintenance required, when it should be done, and future sensor investments for field data collection.

2. Background and Related Work

Predictive maintenance is a sparse field[9] that is primarily studied by those in industry. Most industry work is not published, such as the predictive maintenance system for Verizon's cell towers[12].

[11] has a similar setup to ours, using vehicle sensor readings to predict a repair type. They used windows of size 1, 15, 30, and 60 days before the ensuing repair, and found that the DTC (diagnostic trouble code) field was the best earliest predictor of the vehicle needing maintenance. They also used linear regression and random forests for classification. Above a certain threshold of likelihood, they would predict a failure. To simplify our model, we are choosing only the maximal prediction, and unlike their model, we are trying to predict time to failure along with type of failure, not only type of failure within a fixed window.

[9] pointed out three main approaches to predictive vehicle maintenance: Remaining Useful Life (RUL) prediction, Deviation Detection, and supervised classification. Deviation detection can easily be solved but cannot predict the specific fault that will occur without prior knowledge. RUL is typically done component-wise. A combination of supervised classification and RUL is best for our application given that we wish to predict the RUL for each system of the vehicles as well as classify by the failure codes.

[10] uses KNN, C5.0, and Random Forest learning algorithms to predict a failure within some fixed set of window sizes as [11] did, which differs from our approach to predict the time to failure. Their evaluation approach was based on cost savings of executing on the early maintenance within simulation while ours is based on prediction recall of failure types. [4] is yet another example of using random forests to predict the time to failure on lead-acid batteries.

Many of these papers do not report accuracy, but instead report on cost savings during simulation. In the case of [10], the numbers important to calculate cost savings, such as the exact costs of early repair, late repair and time out of service, were held back by the company providing the data. Since we are not able to acquire these numbers, we do not report on cost savings but instead recall.

Given the ubiquity of Random Forests and the dearth of GMM in the literature, we included these two models in our

Vehicle Snapshot Data:
- 1015071 snapshots of 63 raw features collected from 797 vehicles during their trips.

Veh Ref ID	Event DateTime	Event Type Description	Acc Pedal Position	Ambient Air Temp	Barometric Press	Brake Switch	Bus Utilization	Cat Intake Gas Temp	Cat Outlet Gas Temp	...	Altitude	Engine Start Ambient	Engine Start Coolant	Latitude	Longitude
254.0	2016-04-03 18:00:44	trip_start	0.0	38.59	101.5	0.0	49.0	211.78	214.28	...	48.0	37.77	86.66	37.5	-121.01

Vehicle Repair Claims Data:
- 713 repairs from 558 vehicles

Chassis Reference Number	Model Vehicle	Build_Dt	Divry_Dt	In Service Date	Miles	Rpr_Dt	ATA3	ATA3Desc	ATA6	ATA6Desc	ATA9	ATA9Desc	Fail Type	Repair Cost
194.0	T880	1/22/2016	2/12/2016	2/12/2016	10	2016-02-03	43	EXHAUST SYSTEM (043)	43007	DIESEL EXHAUST FLUID (007)	43007021	FLUID - DEF, DEF SYSTEM (007)	MISCELLANEOUS	very low

Figure 1. Sample snapshot and repair data.

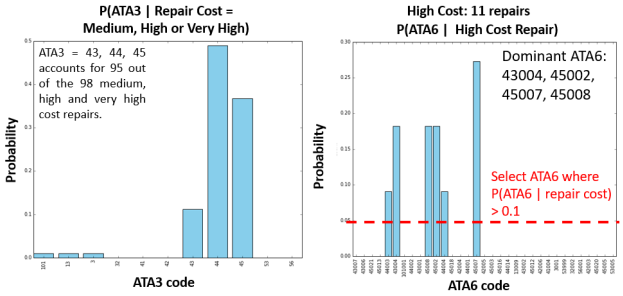

Figure 2. Left: Identifying ATA3 codes strongly associated with medium, high and very high cost repairs. Right: Identifying ATA6 codes that are more strongly associated with high cost repairs. The same was done for very high and medium cost repairs.

approach. Since we are handling time series data, LSTM RNN were also included.

3. Dataset and Pre-processing

3.1. Data Cleaning

Inconsistent formating of dates, duplicate rows, and missing data were identified and dealt with in both the sensor snapshot and repair data. Snapshot features with many missing values or just one value were removed, while rows with missing values were imputed with relevant mean values. The data were then shifted and whitened to have mean 0 and variance 1. In the repair data, 24% of rows were missing the vehicle ID to identify relevant snapshots and another 13% of rows had no snapshots before the repair date, which had to be discarded. An anomalous high-cost repair that resulted from an undecking accident was also removed because the snapshots will be irrelevant to predicting that accident. This reduced the number of valid repairs in our data from 1128 to 713.

3.2. Data Discretization

Once the data were cleaned, the data needed to be put into a format that was useful for processing. We created a map of failure types, which had arrays of snapshots that corresponded to the hyperparameter k, the maximum number of windows of snapshots before the failure. The window $w = 0$ corresponded to the window immediately before the failure. The window size hyperparameter, $|p|$, determined the number of sequential snapshots contained in each window. A data map was created for each of the four sets of fixed (k, p) pairs: $(1, 500), (2, 250), (5, 100), (10, 50)$. These sizes were specifically chosen to keep the total number of snapshots constant per pair for accurate comparison.

3.3. Selection of Failure Codes to Predict

We found that 95 out of the 98 repairs in the 'medium', 'high' or 'very high' cost categories only occur in three locations specified by their ATA3 codes (43:exhaust system, 44:fuel system, 45:power plant). In addition, 77 out of those 95 repairs can be traced further to just six ATA6 codes (43004, 44004, 45002, 45007, 45008, 45021) which dominate the repairs in the 'medium', 'high' or 'very high' cost categories (see Figure 2). We used P(ATA6|cost) > 0.1 to identify dominant ATA6 codes in each repair cost category. As the ATA3 is a broad failure code encompassing many 'low' and 'very low' cost repairs, we decided to predict the ATA6 instead as a more meaningful indicator of potential repair cost. Among the 713 valid repairs in our data, 278 of them ($\sim$39%) had the six ATA6 codes that are more strongly associated with 'medium', 'high' or 'very high' cost repairs, which are our target cost categories.

3.4. Smoothing

As an additional dataset for comparison, before discretization, we smoothed the sensor readings using a 1d convolutional filter of sizes 50 and 100. Given that some of the readings showed significant oscillation, mostly because of the difference between initial sensor readings for "trip start" and "periodic" (for instance, the engine oil temperature would be colder at trip start), we hypothesized that the average recall could benefit from smoothing out these readings.

3.5. Train, Validation, and Test Set

The repairs corresponding to each ATA6 code were split further based on their specific ATA9 codes, and then assigned into training, validation and test sets roughly in the proportion of 70%, 15%, 15% via systematic sampling to minimize distribution mismatch (see Figure 3). Next, vehicles which never appeared in the repair data are assumed to have 'healthy' snapshots and also split into test, validation and training sets in a similar proportion.

4. Method

4.1. HMM Classifier

Given our previous work[8] on this dataset with HMMs, we decided to use HMMs as another model to compare. Unlike our previous work, we are using a HMM for each class,

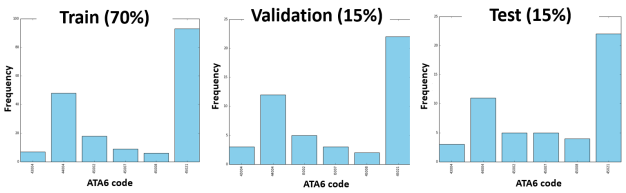

Figure 3. Comparison of train, validation and test distributions over ATA6 fault codes to ensure no distribution mismatch.

ignoring the window and only considering failure prediction. The approach taken here was to maximize the log likelihood that a sequence is observed by a model. We capture the failure code by fitting a different HMM for each ATA6 failure code and then for a new sample we select the model (and hence repair code) that gives the maximum likelihood of observing that sample.

4.2. RNN Classifier

Recurrent Neural Networks (RNN) have shown promising results with sequential problems and thus the team decided to experiment with a specific type of RNN, the Long-Short Term Memory (LSTM) network to compare the performance to that of Random Forests.

In our work, the LSTM includes a hidden layer with n units. The LSTM output is then fed into a vanilla linear forward propagation layer, which is then fed into a Softmax activation function. Finally, the loss function is the cross entropy function which measures the probability error. In addition to the cross entropy loss, regularization over all the weights matrices (not the biases) was performed. For regularization, L-2 Norm was chosen with regularization parameter λ. For the implementation, the Tensorflow library was used. For optimization of the parameters, mini-batch gradient descent was performed for a certain number of epochs with learning rate α.

In order to use the cross entropy loss function correctly, the labels were converted into their one-hot encoding representations where the size of the classes set (labels) was the number of repair codes. After various configurations, by holding all parameters constant and grid-like searches for others, it was found that the following parameters gave the best results: number of epochs set to 30, $\lambda = 0.01$, number of hidden units was set to 25, $\alpha = 0.001$.

4.3. Gaussian Mixture Model

Gaussian Mixture Models (GMM) can be used to estimate a joint probability density for data generated by Gaussians associated with different hidden states, but the hidden states do not necessarily correspond to the failure states we are trying to predict. To get around this, we partitioned our training data by failure codes c and number of time windows w before a repair, and fitted a GMM to each (c, w) subset. It is not apparent when a sample may transition from a normal to failure state before a repair is needed, so we postulate each $GMM_{c,w}$ to have at least 2 hidden states. The joint density from each $GMM_{c,w}$ can then be used to estimate the log-likelihood of a new sample s coming from subset (c, w), from which we can pick a most likely code c that generated the sample s.

The recall for each code c was evaluated on the validation set, which showed that that 2 hidden states was optimal for getting non-zero recall over more codes (higher $\mu_{recall}/\sigma_{recall}$). More hidden states would bias the algorithm towards predicting only the dominant code with a high recall and zero recall over the rest.

4.4. Random Forest

Random Forests, the popular tool among those who work on predictive maintenance, was used to classify the codes and the windows with three approaches: two-stage, two-stage with smoothing, and one-vs-all. Each model was tuned for the number of trees; on average, 100 trees was best.

The two-stage approach involved first classifying a set of snapshots as one of the selected codes, and second as one of the windows. There was 1 code model for the first stage, and k window models per (c, k) pair (see 3.2). The two-stage with smoothing was the two-stage approach but used the smoothed dataset.

The one-vs-all approach involved creating a model for each code, where the classification was binary. In order to combine the results from all of the models, the highest output probability was used. This approach was added after we decided to eliminate the window prediction (see section 6) and so used 1 model per code.

Given our unbalanced dataset, the weighted Random Forest algorithm[3] was used.

5. Test Results

Due to having an imbalanced dataset the average recall was reported, which is an average over the recall rate of each of the repair codes.

Results for all models are shown in table 5. The model yielding the greatest average recall is Random Forests with number of windows at 1 and windows size at 500. By closer inspection, one can see all models also perform close to optimal, if not optimally, for these window values and then start to degrade for other window pairs. Note that these results are for predicting the repair code alone, since predicting the actual time window in which the repair would occur proved to be intractable given this data set.

Note that these results differ significantly from those reported earlier on our poster; please see section 6 for more information.

Test Set Average Recall		
Method (Win Size, No. Win)	Mean (%)	SD (%)
LSTM (1,500)	14.28	34.99
LSTM (2,500)	14.26	34.99
LSTM (5,100)	12.37	27.31
LSTM (10,50)	13.40	27.45
HMM (1,500)	12.69	31.10
HMM (2,500)	12.19	19.30
HMM (5,100)	17.57	22.47
HMM (10,50)	17.93	22.61
RF (1,500)	**42.68**	44.62
RF (2,500)	31.68	43.68
RF (5,100)	34.51	41.83
RF (10,50)	31.54	40.15
RF Smoothed (1,500)	28.57	45.18
RF Smoothed (2,500)	29.58	44.60
RF Smoothed (5,100)	26.92	40.92
RF Smoothed (10,50)	23.94	38.37
RF one-vs-all (1,500)	26.49	35.86
RF one-vs-all (2,500)	26.02	35.23
RF one-vs-all (5,100)	24.91	35.13
RF one-vs-all (10,50)	25.04	35.38
GMM (1,500)	20.26	28.84
GMM (2,500)	15.87	28.24
GMM (5,100)	11.24	17.72
GMM (10,50)	13.69	24.46

6. Discussion

Predicting repair codes as well as number of time steps until that repair occurs proved to be challenging for a number of reasons: the imbalanced dataset, vehicles having indistinguishable sequences for different repairs, unpredictable repairs due to sudden failures, and finally previous work on the subject is sparse. A discussion about these challenges is given here.

Our first attempt towards predictive maintenance included predicting the time window in which a repair would occur. It is not difficult to imagine why this would be useful. Knowing when a repair will be needed allows to optimize how much longer a vehicle can be operated before any maintenance is necessary. In order to accomplish this, our models had to either be broken down into a joint distribution, or a two step process of first predicting the repair, and then given the repair finding the distribution among windows given each repair. However, predictions for time windows gave poor results. Figure 4 shows the mean squared error for the number of features used vs the error for each number of windows and window size pair. Interestingly, the number of features play little to no role in improving the error rate. We can conclude confidently that most (if not all) the features play no role in indicating the time frame of when a specific repair will occur.

Figure 4. Error vs Number of Features

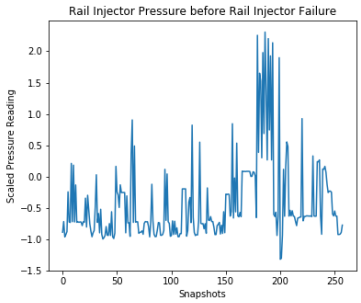

Figure 5. Snapshots vs Scaled Pressure Reading

One explanation for the lack of correlation between a window and a repair is that failures are more sudden. As we can see in figure 5, the pressure reading for a rail injector reading has a sudden anomaly or spike which is precisely the time at which failure for that rail injector occurred. The nature of this failure is that is it difficult to predict as there are no clear early signs that lead up to this failure. Unfortunately, given the features used in this data set, many of the features give little sign of imminent failure, and many of those are typically strongly correlated with high cost.

Sequences for different repairs are surprisingly hard to cluster. One approach we tried was using K-Nearest-Neighbors (KNN) as described in [13] which is commonly used for sequence classification tasks. In order to return the k nearest neighbors, a distance measure was necessary. Recall that a sequence of snapshots can be seen as a matrix, where each row is a single term in the sequence that is the snapshot at a particular time. Thus to measure the distance from different sequences, we had to in effect measure the "distance" between two matrices as follows. First, the L2-norm difference between any two snapshots from each sequence is computed and the resulting matrix of distances where the i, j entry denotes the distance from the ith snapshot of the sequence in question to the jth snapshot in the training sequence. Second, a sum across all elements is computed for the final distance measure. Using this distance measure, it was made clear that repairs were difficult to cluster or classify by neighbors with the

imbalanced dataset. The reasoning is as follows. With an imbalanced dataset that is highly noisy as ours, the dominating class (highest amount of samples corresponding to a specific class) is spread throughout the raw feature space. Therefore, the likelihood that the top k neighbors (or at least majority thereof) of any given new sequence correspond to the dominating class is very high. Thus running k nearest neighbors of any value of k results in mostly choosing the dominating class.

A challenge for training LSTM is in the shortage of data. Although at first the dataset may seem large, because this is sequence classification, the number of snapshots is a misleading number for how large the dataset is. Instead the number of sequences (which are a set of snapshots) for each repair is what is important. Unfortunately, after grouping the sequences by repair type, the resulting number of samples per class was low, on the order of 10^2, which is clearly not sufficient to have a well-trained LSTM model.

The low average recall for GMM was due to it predicting only 3 out of the 6 selected ATA6 codes with varying recall. Average recall degraded a lot from training to validation set, which suggested overfitting. This can be due to validation snapshots belonging to non-predicted failures being very similar to those of the predicted failures, which suggests that more feature engineering and selection is required.

The two-stage approach with Random Forest was most effective. It is likely that the smoothed dataset failed to give better results because of the information loss, as the tiny variations that were being wiped out from smoothing likely provided valuable insight into the failure.

Interestingly, the recall of each code's Random Forest one-vs-all model (when only tested against the positively-labeled data, not the entire dataset) was exceptionally high, around 90% for all but two codes which were around 60%-70%. This gives hope to the idea the data is separable for the different failures. The one-vs-all approach likely failed because the confidences from the different models weren't appropriately combined; a weighting feature similar to the one used in the Weighted Random Forest algorithm but applied to a collection of forests would likely help alleviate this situation.

Ultimately, repairs of interest are for high cost repairs. Thus we limited our models to only see data corresponding to high cost repairs. However, data for high cost repairs was limited, resulting in smaller data sets to learn from, which affected the recall rate for some codes more than others. In particular, there were 4 out of the 7 codes whose recall rate was nearly 0, meaning that our models were predicting very poorly for those repairs. A first attempt yielded recall of 100% for the "No Repair" class predictions and near 0% for the rest. After reducing the number of "No Repair" samples in order to balance the data for this class, the problem

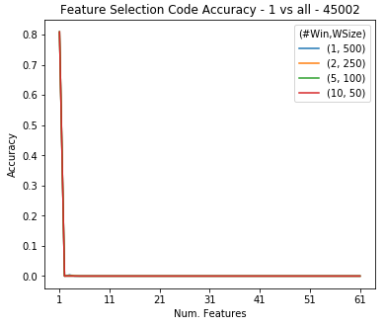

Figure 6. Feature Selection for Code 45002

shifted over to having the next most frequent repair code dominate instead. Thus, the imbalance of the data has drastic consequences, which led to poor performance with some codes and good performance with others.

Given the high number of features (61) for our dataset, we did not perform true feature selection, since the time required would have been too long. Since our true goal was to determine how important each feature was to each code, we took our best model, Random Forest, and used the feature weights that had already been determined from each of the 1-vs-all models. We can see in figure 6 that the code 45002 performs best when only that code's most important features are used, and significantly: this feature has 0% recall with all of the features, but has over 80% recall when fewer features are used.

These results are reasonable since the engine oil temperature, for instance, likely has little effect on the suspension, while the components closer to the engine likely have an effect on other neighboring components to a larger degree. Including these other irrelevant features likely confuses the model and causes poor predictions.

7. Future Work and Conclusion

Collecting more data can play a significant role in balancing out the data and revealing whether or not data balance is sufficient to improve recall among each class.

Since one of the goals of PACCAR was to predict high-cost failures, most vitally more data for high-cost repairs must be collected. With enough data to support it, more specific failures could be predicted using the ATA9 codes.

As discussed in section 6, using an appropriately balanced 1-vs-all approach utilizing feature selection per model could produce an effective model.

We have shown that predictive maintenance is a challenging problem due to the noisy nature of the data set. However, from among HMM, LSTM, GMM and RF, we have shown RF performs the best yielding an average recall of approximately 43%.

8. Joint Project

In addition to CS229, all three of us took AA228, and we are combining the projects from both courses into one. We have created models specific to the concepts learned in AA228 for the project in that class. See [8] for the project in AA228.

References

[1] Paccar. https://github.com/christydennison/paccar. Accessed: 2017-12-09. 1

[2] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966. 1

[3] C. Chen, A. Liaw, and L. Breiman. Using random forest to learn imbalanced data. *University of California, Berkeley*, 110, 2004. 3

[4] M. K. Erik Frisk and E. Larsson. Data-driven lead-acid battery prognostics using random survival forests. Annual Conference of the Prognostics and Health Management Society 2014. 1

[5] T. K. Ho. Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278–282. IEEE, 1995. 1

[6] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1

[7] M. J. Kochenderfer, C. Amato, G. Chowdhary, J. P. How, H. J. D. Reynolds, J. R. Thornton, P. A. Torres-Carrasquillo, N. K. Üre, and J. Vian. *Decision Making Under Uncertainty: Theory and Application*. The MIT Press, 1st edition, 2015. 1

[8] J. Martinez, C. Dennison, and Z. Lian. Hmms for prediction of high cost-failures. *github.com/christydennison/paccar*, 2017. 1, 2, 6

[9] R. Prytz. Machine learning methods for vehicle predictive maintenance using off-board and on-board data. *Halmstad University Dissertations*, 9, 09 2014. 1

[10] R. Prytz, S. Nowaczyk, T. Rognvaldsson, and S. Byttner. Analysis of truck compressor failures based on logged vehicle data. Las Vegas, Nevada, USA. 9th International Conference on Data Mining. 1

[11] S. Ramanujam. The data science behind predictive maintenance for connected cars. 2016. 1

[12] K. Sennaar. Telecom machine learning applications comparing at&t, verizon, comcast and more. https://www.techemergence.com/telecom-machine-learning-applications/. Accessed: 2017-12-01. 1

[13] Z. Xing, J. Pei, and E. Keogh. A brief survey on sequence classification. *ACM Sigkdd Explorations Newsletter*, 12(1):40–48, 2010. 4