Learning to Predict Component Failures in Trains

Sebastian Kauschke[1], Immanuel Schweizer[2], Michael Fiebrig[4], and Frederik Janssen[3]

[1] skauschke@gmail.com
Technische Universität Darmstadt, Germany
[2] schweizer@cs.tu-darmstadt.de
Telecooperation Group
Technische Universität Darmstadt, Germany
[3] janssen@ke.tu-darmstadt.de
Knowledge Engineering Group
Technische Universität Darmstadt, Germany
[4] michael.fiebrig@dbschenker.eu
DB Schenker
Head of Technical Management
Components Locomotives & Wagons (L.RBA 2 (A))

Abstract. Trains of *DB Schenker Rail AG* create a continuous logfile of diagnostics data. Within the company, methods to use this data in order to increase train availability and reduce costs are researched. An interesting and promising application is the prediction of train component failure.
In this paper, we developed and evaluated a method that utilizes the diagnostic data to predict future component failures. To do so, failure codes were aggregated and a flexible labeling scheme is introduced. In an extensive experiment section, three different failure types are examined, a combination of them is evaluated, and different parametrizations are inspected in more detail.
The results indicate that a prediction for all of the different types indeed is possible starting from days up to weeks ahead of the failure. However, the level of data-quality and its quantity still have to be increased considerably to yield high quality models.

1 Introduction

Each year, up to 400 million tons of goods are transported by *DB Schenker Rail AG*[5]. To achieve this, a fleet of over 3500 trains is available. Reliability is a crucial issue, since delivery dates have to be met. Complications may lead to delays and increase the cost of a transport. Mechanical failures of the train itself is one of the main reasons for not reaching destinations in time. Failures of a mechanical component are costly, since not only the component itself has to be replaced or repaired, but also the train is standing

 In: T. Seidl, M. Hassani, C. Beecks (Eds.): Proceedings of the LWA 2014 Workshops: KDML, IR, FGWM, Aachen, Germany, 8-10 September 2014, published at http://ceur-ws.org

[5] `www.rail.dbschenker.de`

still and consequently has to be transported to a repair station, which causes difficulties with the global train schedule. Among others, these effects create additional costs.

To improve reliability, mechanical failure needs to be avoided as far as possible. Trains are regularly maintained according to a maintenance schedule, but since a train is a complex machine, not every component is checked each time. There are different kinds of inspections, some are smaller and occur more often and are used for small checkups. Others are large inspections that occur in longer intervals but with much more in-depth examination. Scheduled maintenance alone is therefore not able to provide constant monitoring of component status.

In other industries there have been successful implementations of constant monitoring and failure prediction such as logfile evaluation and engine temperature monitoring [1, 2]. It was shown that imminent failures can be detected early enough to avoid breakdowns. Fixed maintenance intervals can be replaced by on-demand maintenance, hence reducing cost and increasing reliability. The availability of the fleet will improve, allowing more goods to be transported. In this paper, we follow that path and present an approach to achieve failure prediction suited to the data provided by *DB Schenker Rail AG*.

The idea is to utilize the events occurring in the train's computer systems. The various soft- and hardware systems of a modern train provide a continuous stream of such events, including status changes and errors. This is referred to as diagnostics data. Given such data, a method to predict failure of various kinds of components is developed. The approach uses the frequency of certain types of log entries in a variable time window before the actual failure to label a dataset and then train a classifier using the labeled data. The resulting model then is able to to predict such failures on new data.

The method incorporates in-depth knowledge from engineers of *DB Schenker Rail AG*. The experts helped significantly to accelerate the data mining process involved by reducing the amount of data to a feasible subset. The topic has been discussed in more detail in the diploma thesis of Sebastian Kauschke [3]. In this paper, only a subset of the work presented there will be shown.

In the next section, the approach is detailed with a focus on the employed pre- and postprocessing of the data. Then, the experimental setup is skechted (Section 3 and the results are shown in Section 4. The following section provides related work and in Section 6 the paper is concluded and future research is given.

2 Prediction of Failures based on Diagnostic-Code Frequency

The proposed approach uses the frequency of diagnostic-code occurrence as a measure to detect anomalies in the train's behavior and predict failures. The existing diagnostics data is used to create the features. We start by giving some details on the diagnostic data. Then a hypothesis is proposed, followed by a detailed description of the preprocessing of the data.

2.1 Diagnostics Data

Since the 1990s trains are equipped with on-board computers that connect and control the various systems. A train of the so-called "'Baureihe 185"' (*BR185*) for example

has 37 main systems including a total of 70 subsystems. Each of the activities in those systems and subsystems is recorded in the diagnostics data file. The diagnostic messages include, e.g., protocol messages, events, warnings, and errors as well as analog measurements. A total of over 6900 types of diagnostic messages exist for this train type.

Each diagnostic message consists of the diagnostic-code, the system and subsystem this code belongs to, timestamps when it occurred first and when it vanished (diagnostic messages may span a certain amount of time), as well as the *environment data*. The *environment data* can include status variables, analog measurements, or simply plain text.

Please note that the data provided to conduct this research suffers from low coverage in regard to the recorded timespan which made high-quality predictions a true challenge.

A feature consists of an aggregation of past occurrences of a certain diagnostic-code in a specific time-frame. This way, for each day and train a feature vector is produced that incorporates information about how often diagnostic-codes appeared in that time-frame. Those vectors are then labeled in two categories: *normal* and *warning* where *normal* is representing a point in time where no failure was imminent, and *warning* is denoting an imminent failure.

A classifier is then trained on the labeled data, and 5-fold cross-validation is used to evaluate the classifier.

2.2 Hypothesis

In the following we propose a hypothesis concerning the dependency between diagnostic-codes and failure events.

Hypothesis 1 *Failure of components of the train are preceded by an increased appearance of specific diagnostic-codes in a certain time-frame before the failure occurs. The failure can be predicted by detecting which diagnostic-codes show this kind of increase and how much they increase opposed to a situation with no imminent failure.*

Based on this hypothesis a method to predict failures is created by the use of machine learning. To proof the hypothesis it is necessary to answer the questions to what extent a time-frame before the breakdown must be examined, and how far in advance a failure can be predicted. Those values will be discovered experimentally.

2.3 Preparations

The available data spans 18 months in time and includes over 3.7 million diagnostic-codes in a total of 187 trains. In that timeframe the majority of failures happened, most of them being of little interest for this research. To decide which failures are of interest, the following criteria have to be met:

- the component has failed because of deterioration
- the component needs to be attached to one of the systems sending diagnostic messages

Table 1. Exemplatory occurences of diagnostic-codes

Day:	1	2	3	4	5	6	7	8
CodeA:	1	0	2	1	1	3	2	0
CodeB:	1	1	4	3	2	4	2	2
CodeC:	2	4	2	1	1	2	4	3

- the failing component causes a certain amount of cost or affects the trains ability to move
- the date of the failure needs to be known

At *DB Schenker Rail AG* a database including all repair station activities exists. It is mainly used for accounting purposes, but in this case it was used to search for activities that indicate failures which fit the required criteria. The database contains information about the dates the failures happened and on which trains they happened.

From the failures meeting the criteria, the three most frequent ones in the given timeframe were chosen:

- *ASG instand setzen* (repair motor control, found 142 times)
- *LZB instand setzen* (repair guiding system, found 126 times)
- *LZB Empfangsantenne einstellen* (repair antenna of guiding system, found 93 times)

2.4 Preprocessing

At first a decision has to be made which diagnostics-codes are relevant for the failure to be predicted. In an ideal environment, all existing diagnostics-codes (over 6900) would be used as features at first. This feature set can later be reduced by methods that single out the necessary features for the specific failure prediction.

In this case an expert was questioned and stated only diagnostic-codes of the system the failing component belongs to should be examined. With this information, the 30 most frequent diagnostics-codes of the relevant systems in the timeframe before the failure are chosen to be the features. The frequency of each code occurrence per train per day is calculated and will be used for the further steps.

2.5 Aggregation of features

A feature consists of the frequency of a diagnostic-code $CodeX$.

In Table 1 an exemplary amount of occurrences per day is shown for *CodeA*, *CodeB*, and *CodeC*.

For each day and each diagnostic-code a vector A_x is calculated. In the following example $x = 3$ is chosen:

$$A_x = \begin{pmatrix} CodeA \\ CodeB \\ CodeC \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \\ 2 \end{pmatrix}$$

This Vector sums up the occurrences of all diagnostics-codes on day x. To achieve a time-span evaluation, a vector V is built, accumulating the vectors $A_{x-v}, \ldots, A_x$ by averaging the values. For $x = 5$ and $v = 3$ this is:

$$V_{x,v} = \begin{pmatrix} (0+2+1+1)/4 \\ (1+4+3+2)/4 \\ (4+2+1+1)/4 \end{pmatrix} = \begin{pmatrix} 1 \\ 2.5 \\ 2 \end{pmatrix}$$

2.6 Labeling

Each of the three failure-types will be handled as a separate classification problem. For every day x and every train a vector $V_{x,v}$ is calculated as described in Section 2.5, covering the v days preceding x. For each case of a failure a warning-timeframe is defined. The warning-timeframe ends with the day of the failure (S) and is of length w.

The vectors are labeled the following way:

1. For all failure dates: vectors in the range of $S - w, \ldots, S$ that belong to the defunct train are labeled as *warning*.
2. All other vectors are labeled as *normal*

The vectors consist of a total of 30 features.

2.7 Postprocessing

Since the diagnostics data used for this experiment is incomplete and not all the days in the 18 month time interval are covered for every train, there is a necessity to filter out days that have no recorded entries at all. This means, if there is no entry on a specific day for a train, the data on this day will be marked as *invalid*. It will be excluded from vector generation and will not be used for training and evaluation.

This is especially problematic for the labeling process, since 50 % - 75 % of the time there are no or few data recordings available for the *warning* vectors. When the coverage of data for a specific failure date is lower than 50 %, all the vectors in the *warning*-timespan are marked as *invalid* and not used for training and evaluation.

3 Experimental Setup

In the following the algorithms used in the experiments are summarized. Then, a total of five different experiments are defined.

3.1 Training and Evaluation

The WEKA Suite [7] is used for training and evaluation. The classifiers *JRip*, *J48*, *RandomForest*, and *SMO* are chosen for classification. This selection ensures that most symbolic algorithms are present and that also a statistical classifier is employed.

The WEKA parameterization of the classifiers were set as follows:

Table 2. v and w values for experiment 1 and 2

(a) Exp. 1: Motor Control Failure

Configuration	v	w	number of failures
a	5	3	49
b	10	5	46
c	30	10	40

(b) Exp. 2: Guiding System Failure

Configuration	v	w	number of failures
a	5	3	85
b	10	5	87
c	30	10	89

JRip: *Folds*: 3, *minNo*: 2, *Optimizations*: 2.

J48: *confidenceFactor*: 0.25, *minNumObj*: 2, *subtreeRaising*: true.

RandomForest: *maxDepth*: unlimited, *numFeatures*: unlimited, *numTrees*: 10.

SMO: *C*: 300, *Tolerance*: 0.001; *Epsilon*: 1E-12, *Kernel*: Polykernel (Cache Size: 250007; E=5).

To verify the results, 5-fold cross-validation was used. The main performance indicators are the *precision* and *recall* values of the *warning* class. Usually, *accuracy* is a good measurement for the performance of a classifier, but in this case *accuracy* is over 97 % in most cases. The reason for this are the unbalanced classes. The *warning* class is a minority class by a factor of up to 100. Another measure that reflects the performance of the classifier is the area under curve (AUC) value. We decided to include this evaluation measure instead of regular accuracy as it is capable to incorporate unbalanced classes.

3.2 Five Experiments

A total of five experiments were conducted during our study: One experiment for each of the chosen failure types, and one experiment for a combination of all three failure types. The fourth experiment is used to show that a discrimination between the three types of failure is possible. In the last experiment a wider range of the v and w values is examined, to show where the peak potential can be achieved.

Experiment 1: Motor Control Failure This experiment is based on the failure of the *Antriebssteuergerät* (*ASG*), which is the motor control unit of the train. It manages power distribution to the four electric engines. If it fails, it is often the case that one of the two power converters shuts down, which results in a 50 % power loss. This may stop the train from moving, depending on how heavy it is loaded.

For this type of failure, 142 instances have been recorded. After postprocessing the labeled data, only a certain number of those is still considered *valid* according to the criteria described in Section 3. This depends on how the timeframes are chosen for the w and v to built the vectors. Three combinations of v and w were chosen (see Table 2 (a)).

Table 3. v and w values for experiment 3 and 4

(a) Exp. 3: Guiding System Antenna Failure

Configuration	v	w	number of failures
a	5	3	17
b	10	5	17
c	30	10	20

(b) Exp. 4: Combining all three Failure Types

v	w	number of failures
30	10	97 (total)

Experiment 2: Guiding System Failure This experiment is based on the failure of the *Linienförmige Zugbeeinflußung* (*LZB*), which is one of the guiding systems used by *Deutsche Bahn*. It allows to coordinate the positions of all trains and achieve faster driving speeds. If it fails, the train driver is forced to rely on other guiding systems, which can reduce overall speed or cause other problems.

For the guiding system failure, 126 instances were recorded. Only a certain amount of those is still considered *valid* after postprocessing, according to the criteria described above that are dependent on w and v. For this experiment, also three combinations of v and w were chosen (cf. Table 2 (b)).

Experiment 3: Guiding System Antenna Failure The *LZB* has an antenna, which provides it with the information it needs to work. If the antenna fails, the *LZB* will cease to function properly, and the same restrictions as in experiment 2 will apply.

93 instances of antenna failure have been recorded. After postprocessing a major amount is considered *invalid*. The following combinations of v and w were chosen (see Table 3 (a)).

Experiment 4: Combining all three Failure Types In this experiment a multi-class approach is used to show that discriminating between the three types of failures also is possible. To achieve this, the feature generation and labeling process are altered.

In the feature generation step, the number of features is increased. The required features for the three failure types partly overlap, so a mixture of features is used to allow every failure to be predicted adequately.

The labeling process is adapted in the following way:

1. For all motor control failure dates: vectors in the range of the *warning* timeframe are labeled *warning1*.
2. For all guiding system failure dates: vectors in the range of the *warning* timeframe are labeled *warning2*.
3. For all guiding system antenna failure dates: vectors in the range of the *warning* timeframe are labeled *warning3*.
4. All other vectors are labeled as *normal*

As the $v = 30$ and $w = 10$ combination of the previous experiments showed the best results, for this experiment only this combination is used (Table 3 (b)).

Table 4. Results for Motor Control Failure

(a) $v = 5$, $w = 3$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.333	0.076	0.12	0.534
J48	0.167	0.010	0.02	0.513
RandomF.	**0.610**	**0.171**	**0.27**	**0.820**
SMO	0.517	0.148	0.23	0.573

(b) $v = 10$, $w = 5$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.592	0.229	0.33	0.623
J48	0.678	0.150	0.25	0.609
RandomF.	**0.835**	**0.380**	**0.52**	**0.904**
SMO	0.612	0.226	0.33	0.612

(c) $v = 30$, $w = 10$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.728	**0.675**	0.70	0.838
J48	0.752	0.577	0.65	0.905
RandomF.	**0.895**	0.667	**0.76**	**0.968**
SMO	0.757	0.487	0.59	0.742

Experiment 5: Extended Timeframes The last experiment is conducted to show which parameter combination of v and w achieves the highest *AUC*. *RandomForest* is used as the only classifier as it showed the best overall performance in all previous experiments. The failure type of experiment 1 is used for the evaluation. This choice is somewhat arbitrary and we believe that the parameters are indeed subject to change among different experimental setting, but, however, for demonstration purposes and due to space restrictions, we had to choose one of the above experiments.

The former experiments showed the highest results in the (c) configuration, which was the largest settings of v and w. To allow a better overview, if the performance can be further increased, a grid of *AUC* values is generated, with v values ranging from 20 to 100, and w values from 10 to 50, each in incremental steps of 10.

4 Results

In this section the results of the experiments will be shown and discussed. Note that for both *precision* and *recall*, the values are shown for the *warning* class. As the *AUC* is insensitive to imbalanced class distributions, we preferred this type of measure over regular *accuracy*.

4.1 Results Experiment 1: Motor Control Failure

In Table 4 the *RandomForest* classifier shows the overall highest performance. For all classifiers performance increases come along with larger timespan v and warning time w. RandomForest reaches *AUC* values of up to 0.904 (0.76 F1-score) in configuration (c). *JRip*, *J48* and *SMO* deliver similar *precision* values, but lack *recall* and their *AUC* values are lower than *RandomForest*.

All classifiers show a significant increase of *recall* with each step up in timespans, while *precision* interestingly does also increase.

Table 5. Results for Guiding System Failure

(a) $v = 5$, $w = 3$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.351	0.077	0.13	0.536
J48	**1.000**	0.071	0.13	0.537
RandomF.	0.684	0.154	**0.25**	**0.818**
SMO	0.403	**0.172**	0.24	0.585

(b) $v = 10$, $w = 5$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.500	0.224	0.31	0.624
J48	0.586	0.148	0.24	0.633
RandomF.	**0.832**	0.345	**0.49**	**0.926**
SMO	0.620	**0.384**	0.47	0.691

(c) $v = 30$, $w = 10$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.750	0.690	0.72	0.843
J48	0.789	0.693	0.74	0.895
RandomF.	**0.942**	0.701	**0.80**	**0.983**
SMO	0.773	**0.810**	0.79	0.903

Table 6. Results for System Antenna Failure

(a) $v = 5$, $w = 3$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.568	**0.292**	**0.39**	0.668
J48	**1.000**	0.069	0.13	0.546
RandomF.	0.706	0.167	0.27	**0.858**
SMO	0.556	0.278	0.37	0.639

(b) $v = 10$, $w = 5$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.581	0.443	0.5	0.741
J48	0.652	0.155	0.25	0.591
RandomF.	**0.800**	**0.412**	**0.54**	**0.957**
SMO	0.556	0.361	0.44	0.680

(c) $v = 30$, $w = 10$

Classifier	Precision	Recall	F1-Score	AUC
JRip	0.766	**0.695**	0.73	0.867
J48	0.822	0.550	0.66	0.845
RandomF.	**0.916**	0.649	**0.76**	**0.959**
SMO	0.732	0.669	0.70	0.834

4.2 Results Experiment 2: Guiding System Failure

The results shown in Table 5 are similar to those of experiment 1. With a *AUC* value of up to 0.983 and a F1-score of 0.8 in the widest timeframe (configuration (c)), the results of *RandomForest* are the highest of all classifiers. *SMO* achieves the highest *recall* value in all configurations, but the *precision* of 77.7 % is lower than *RandomForests* 94.2 %. *J48* and *JRip* show similar results with *AUC* values of 69 %.

The significant increase of the *recall* value with each step up is also evident for all the classifiers. However, *J48* has its best *precision* at the (a) configuration.

4.3 Results Experiment 3: System Antenna Failure

RandomForest shows high values in *recall* and *precision* in Table 6 which is also shown by an *AUC* value of 0.959 for configuration (c). *JRip* achieves the highest *recall* value of

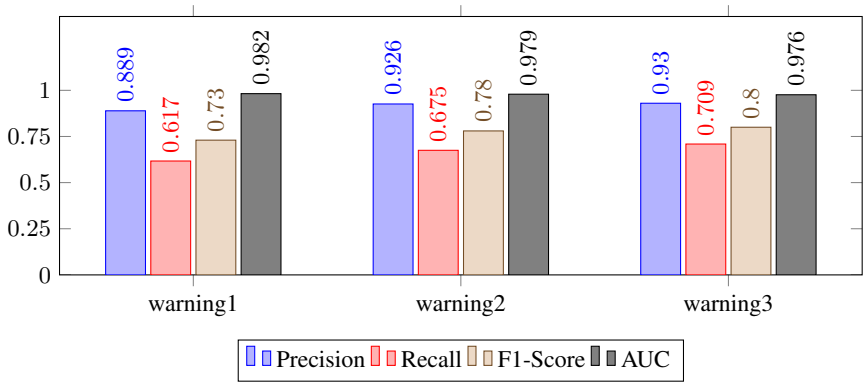

Fig. 1. Results for Experiment 4: RandomForest Classifier; $v = 30, w = 10$

Table 7. Experiment 4: Confusion Matrix for RandomForest

	classified as			
	normal	warning1	warning2	warning3
normal	**19753**	25	16	7
warning1	128	**209**	2	0
warning2	107	1	**226**	1
warning3	44	0	0	**107**

69.5 % and the second highest *AUC* value of 0.867 while outperforming *RandomForest* in F1-score at least in configuration (a). *J48* has a similar *AUC* of 0.845 and *SMO* has 0.834.

The significant increase of *recall* with bigger timeframes is also present here. All classifiers except *J48*, which shows a similar trend as before, achieve their best results in configuration (c).

4.4 Results Experiment 4: Combining all three Failure Types

Figure 1 shows that all of the three different warning types could be predicted with a high *precision* of 88.9 %, 92.6 % and 97.6 %, respectively while also the *recall* was above 60 %. The F1-score never falls below 0.7. Therefore, we can conclude that our hypothesis posed in Section 2.2 seems to be valid. In the confusion matrix depicted in Table 7 it can be seen that only a total of 1.3 % of all *normal* cases are classified incorrectly and that the actual warnings are predicted quite accurately. However, about a third of them is falsely classified as *normal*, which explains the *recall* values of 60 % to 70 %.

4.5 Results Experiment 5: Extended Timeframes

In the last experiment, different configurations for the days taken into account (v) and the number of days before the failure that are labeled as failure (w) were examined.

Table 8. Results for Extended Timeframes

(a) AUC

	w=50	w=40	w=30	w=20	w=10
v=100	0.991	0.995	0.994	0.984	0.982
v=90	0.992	0.995	0.99	0.992	0.989
v=80	**0.999**	0.996	0.993	0.995	0.992
v=70	0.997	0.997	0.996	0.995	0.992
v=60	0.992	0.989	0.995	0.99	0.99
v=50	0.99	0.989	0.988	0.99	0.972
v=40	0.982	0.984	0.991	0.988	0.981
v=30	0.978	0.979	0.978	0.976	0.968
v=20	0.966	0.97	0.973	0.972	0.967

(b) F1-score

	w=50	w=40	w=30	w=20	w=10
v=100	**0.999**	0.998	0.998	**0.999**	0.998
v=90	0.998	0.998	0.998	0.998	0.998
v=80	**0.999**	**0.999**	0.998	**0.999**	**0.999**
v=70	0.998	0.998	0.998	0.998	**0.999**
v=60	0.997	0.997	0.998	0.998	0.998
v=50	0.996	0.996	0.997	0.998	0.998
v=40	0.995	0.995	0.996	0.997	0.997
v=30	0.994	0.994	0.994	0.996	0.996
v=20	0.991	0.992	0.993	0.994	0.995

Table 8 (a) shows that an increase of the v and w values can further improve the *AUC*. The highest result is given at $v = 80$ and $w = 50$ with an *AUC* of 0.999 compared to the original value achieved in experiment 1 (cf. Section 4.1) which was 0.968. Interestingly a different picture manifests when inspecting the F1-score. Here, there is no single best configuration but a total of seven ones achieved the highest value (cf. Table 8 (b)). Among them, also the best one for *AUC* is present, but, however, it seems that F1-score is not so sensitive to these two parameters.

5 Related Work

This section will provide an overview of work that was used directly or as an inspiration for the methods described in this paper. There are certain parallels, so some of the methods were adapted and adjusted to fit the specific problems. However, related work is rather rare as usually the algorithms are tied to a specific problem and it is hard to generalize to arbitrary scenarios.

Fulp, Fink and Haack [1] proposed a method based on the evaluation of system logfiles to predict hard disc failure. The logfile data was used to train a support-vector machine to recognize sequences or patterns in the messages which implied an impeding failure. With a *sliding window* method subsequences of the log were evaluated and classified as *fail* or *non-fail*. The training data consisted of the actual system log of a linux computing cluster. The results showed promising results with up to 73 % recognition rate up to two days ahead of the failure.

The work of by Létourneau, Famili and Matwin [4] originated from the aircraft domain. The authors examined the problematics of large amounts of data generating systems in an airplane. They addressed the issues of data gathering, labeling, and model integration and presented an approach to learn models from the data to predict issues with components of the aircraft.

In a Phd. thesis of Lipowsky [5], the author focused on condition monitoring of gas turbines. The differences between handling gradually occurring degradations and spontaneous failures were elaborated. An integrated system to deal with both cases

was developed, one based on a least-squares solution, the other based on nonlinear optimization.

6 Conclusions

In this paper we proposed a method to predict component failures based on system logs. The results show, that such a type of prediction indeed is possible. However, the effort to reach this goal is high. For every type of failure a separate classifier has to be trained, although the results of experiment 4 (cf. Section 4.4) show that a combined classification is possible. Nevertheless, such a combination still is prone to result in a reduced classification performance compared to treating each problem separately. Also, and perhaps most importantly, the requirement for methods such as proposed in this paper, namely data quality is not yet met. To assure a consistent data environment for the training, the data needs to be complete for the whole fleet. The data used for this research had huge gaps in the recorded timespan. This is a problem *DB Schenker Rail* has to solve, before methods like this can be used in a real-world environment. We also assume that the prediction quality will significantly increase given the data quality is improved.

Also, the problem of imbalanced classes is certainly present in domains such as failure prediction as usually cases of failure are quite rare compared to the regular cases where everything is fine. The results show that the *RandomForest* classifier seems to work well on imbalanced data, but, if tackled appropriately the performance will even increase.

For future work it is planned to carefully tune the parameters of the machine learning algorithms for each single problem. Also, given the data quality is enhanced, the approach has to be re-run on the new data. Another interesting topic is to inspect the effects of the values v and w also for the other experiments and figure out whether or not similar trends are present.

References

1. Fulp, E.W., Fink, G.A., & Haack, J.N., Predicting Computer System Failures Using Support Vector Machines. *WASL'08 Proceedings of the First USENIX conference on Analysis of system logs*, 2008.
2. Guo, P., & Bai, N., Wind Turbine Gearbox Condition Monitoring With AAKR And Moving Window Statistic Methods. *Energies 2011, 4, 2077-2093.*, 2011.
3. Kauschke, S., Nutzung Bahnbezogener Sensordaten Zur Vorhersage Von Wartungszyklen, Diploma Thesis, TU Darmstadt, Knowledge Engineering Group, http://www.ke.tu-darmstadt.de/lehre/arbeiten/diplom/2014/Kauschke_Sebastian.pdf, 5 2014.
4. Létourneau, S., Famili, F. & Matwin, S., Data Mining For Prediction of Aircraft Component Replacement. *IEEE Intelligent Systems Jr., Special Issue on Data Mining, p. 59-66*, 1999.
5. Lipowsky, H., *Entwicklung Und Demonstration Eines Integrierten Systems Zur Zustandsüberwachung Von Gasturbinen*, Phd. Thesis, Stuttgart, Institut für Luftfahrtantriebe, 2010.
6. Swets, J., *Signal Detection And Recognition By Human Observers*, New York, Wiley, 1964.
7. Witten, I.H., Frank, E., & Hall, M., *Data Mining - Practical Machine Learning Tools And Techniques.* Burlington: Morgan Kaufmann Publishers, 3rd edition, 2011.