Data Driven Detection of Railway Point Machines Failures

Iwo Doboszewski, Simon Fossier, Christophe Marsala

Data Driven Detection of Railway Point Machines Failures

Iwo Doboszewski
AGH University of Science and Technology,
al. Mickiewicza 30, 30-059 Krakow, Poland,
Sorbonne Université, CNRS, LIP6,
4 place Jussieu, 75005 Paris, France,
idobosz@agh.edu.pl

Simon Fossier
Thales Research & Technology,
1 av. Augustin Fresnel,
91767 Palaiseau Cedex, France
simon.fossier@thalesgroup.com

Christophe Marsala
Sorbonne Université,
CNRS, LIP6,
4 place Jussieu, 75005 Paris, France,
Christophe.Marsala@lip6.fr

***Abstract*—In this paper, a novel approach to early detection of railway point machines failures is presented. Easily accessible data from Centralized Traffic Control (CTC) systems, along with meteorological data, are utilized to build a classification system recognizing risk factors for railway point machine failure. We present and discuss a framework that aims at extracting information from the raw railway logs, and discuss the issues that need to be solved to make the framework properly operational. We show that ensemble methods utilizing decision trees are able to provide meaningful classification accuracy for this problem.**

***Keywords*—Point machines; Failures prevention; Classification; Machine learning; Ensemble methods.**

I. Introduction

Point machines are responsible for correct setting of rails on turnouts. Correctness of their operations is a necessity for proper operations and safety of any railway network. A malfunctioning point machine can result in either technical breaks or switch to manual operation introducing delays and increases risks or directly cause train crash.

In this paper, the term *commonly recorded data* refers to data recorded by the Centralized Traffic Control (CTC) system (communication, operations and failures of the point machines) and publicly available meteorological data. Recording of such information in Poland and most of the countries is mandatory, according to the safety regulations, so obtaining this data does not require the introduction of specialized measurement systems into the operating point machines and is basically investment-free.

Our research is a first step for failure prediction and aims at establishing whether this commonly recorded data can be used to highlight what are the conditions under which failures happen. The goal is to discover whether there are patterns of failures that can be explained by this data, or if failures are occurring either randomly or due to other external conditions, based on records coming from the railway supervision system and current weather conditions. This allows for recognition of risk factors for failures, which is a valuable result from the practical point of view, as it allows to create a more informed maintenance policy.

Our approach can be decomposed as follows. First, events are represented in terms of numerical features that can be used for the classification. Then the classification system for the operations of the point machines is trained. The system attempts to classify the events as correct operations or failures based on the features representation. It is trained and validated in a supervised setting, as the recorded events are already labeled as correct or failures by the CTC system. Several classification methods are applied and the results are compared.

The novelty of this approach is that the analysis is based on a huge volume of real-world data recorded over long period of time on multiple stations. This gives comprehensive overview of most common failures and operations characteristics that point machines are experiencing. To our best knowledge, this approach to the analysis of the point machines failures has not been tried before.

The paper is organized as follows. In Section II, we set the general problem of point machine reliability, along with a state of the art of existing solutions. In Section III, the data that is used for the calculations is described. In Section IV, our planned approach for the data analysis is presented. Then, in Section V, the setting for calculations is presented and the results are summarized. Finally, in Section VI, conclusions are presented and future work is discussed.

II. Point machines – principles, state of the art

Point machines are devices responsible for turnout setting and are managed remotely by CTC system. They are composed of an engine and a movement transmission system with stretcher bar setting the connecting rails. Connecting rails can be set to the two positions allowing the train to either turn or proceed forward.

Once a setting to a specific train position is ordered by a CTC system, a point machine is either locked in the position it is already in, or the engine moves the stretcher bar to another position. The movement needs to be executed within a predefined time, otherwise the operation is considered to be a failure. This time is usually set around 10 seconds, depending on the specific network requirements. Such a movement is refer further as an *operation*. In the case of the data discussed further, this time is 11 seconds.

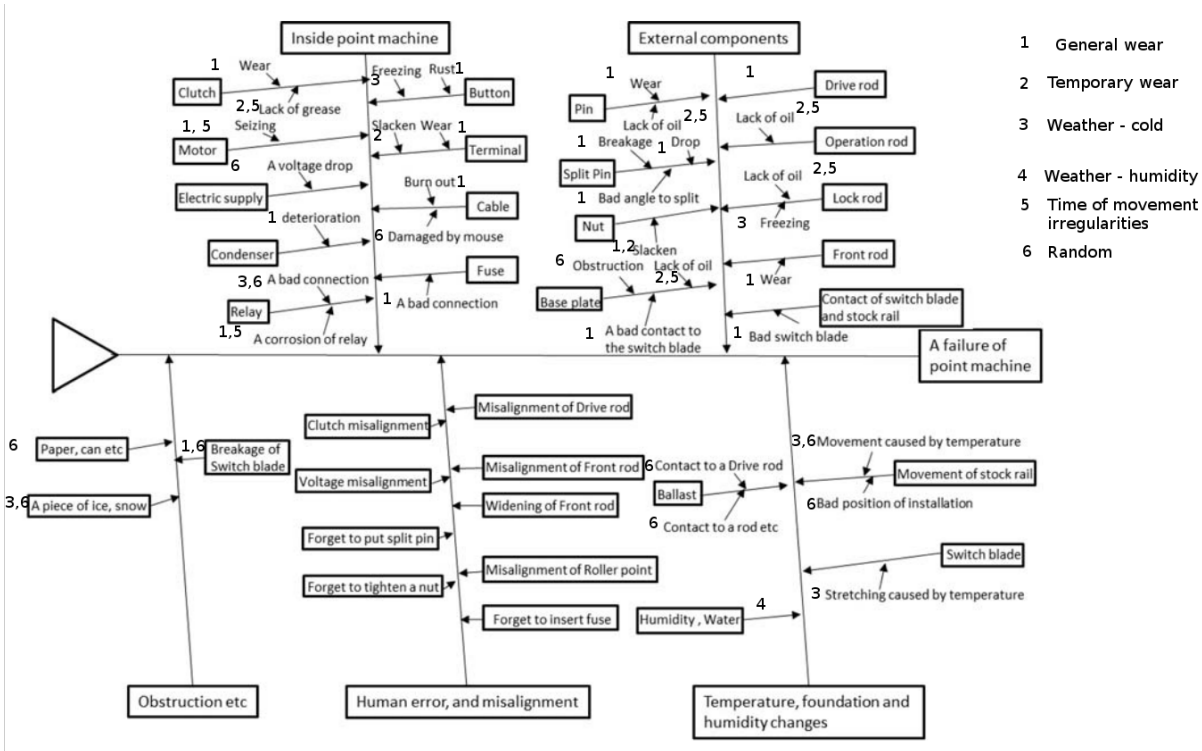

Fig. 1. Fishbone diagram of faults for point machines from [1] with marked connection to specific conditions.

A. Failure modes

While the point machine is not a very complex device in terms of number of components, it has multiple failure modes. Failure modes and mechanisms standing behind them are presented in Figure 1 from [1].

The failures can be divided among general failure causes. The ability to recognize risk factors is a valuable result from the practical point of view, as it allows to create more informed maintenance policy. In our case, the system that produces the data does not possess diagnostic features and does not distinguish the failures resulting from different failure modes and only acknowledges failures once the confirmation of successful movement does not arrive in specified time.

The data limitations do not allow us to distinguish precisely between different failure modes but it enables us to detect some general conditions that can be connected with groups of specific failure modes. The reason for the specific conditions choice is their presence in the data. These conditions are:

- General wear out due to high number of performed operations.
- Temporary usage (possible lack of grease).
- Weather conditions – cold and frost change mechanical properties of moving systems and can slow down or even block point machine movement.
- Weather conditions – high humidity can negatively impact electronics in point machine and even lead to a failure in extreme case.
- Irregularities in the time of the movement – varying time of point machine movement between consecutive operations can be a signal of mechanical issues and wear out that leads to a failure.

Some failure modes are connected to long-term humidity, but we do not a have a way to assess it, as the measured conditions for most train stations are similar. There are some statistical yearly differences in humidity, but they seem to be secondary, as the climate is the same on all analyzed stations (there is no big difference such as between swamp and desert).

There are failures that do not correspond to the identified conditions. For the sake of simplicity, we refer to them as *random* as they are indistinguishable from truly random failures given the knowledge provided by the data we use.

B. State of the art

Point machine condition monitoring has recently been a prominent field of study. Main approaches for providing early faults detection of a point machine seems to be based on the analysis of their power consumption [1] [2] [3]. In [4], authors estimate the remaining useful life based on electric current measurements, in [5] vision-based measurements of a gap between point machine blade and the rail is used, and in [6] an analysis of the sounds from the point machine is done.

The main differences between existing approaches and ours lays in the fact that, in our case, no additional, specialized measurement systems need to be provided and, moreover, all the approach works with only data that is already being recorded.

III. Data description and feature extraction

As explained earlier, our approach is based on two sources of data: the railway data and the weather data taken from [7].

A. Railway data

The data used for this research are raw logs from CTC system obtained from Thales Polska, recorded during about one year at 6 train stations in Poland. The records are composed of the communication (movement orders, operation confirmations etc.) between CTC system and traffic control devices (point machines, axle counters, signaling lights). The data comes from 373 point machines with various frequencies of operations, from a few to dozens per day. The devices in vast majority are produced by Thales Group. While the exact effects of wearing-out and useful life can be different between different types of point machines, we expect, that the most general causes of their failures are failing under similar scheme and can be extracted from this data set.

The events of interest are movement orders, movement confirmations and failures reports of point machines. All events are time-stamped. The duration of a point movement can be estimated as the difference of time stamps between reported start and end of the movement.

B. Categories of features

The two combined datasets are used to derive the features that represent conditions during operation. We distinguish three categories for such conditions: operations record features, time of operation features, and weather features.

1) Operations record features: These features correspond to general usage of the point machines, such as operations frequency and presence of failures. They aim to assess basic level of degradation from wearing out.

2) Time of movement features: These features are meant to represent changes in time of the movement of a point machine, as we expect that slowing down and irregular variations can be a signal of approaching failure. A number of statistics is calculated over specific, constant number of preceding operations. These preceding operations are referred to as preceding *time window*.

For the first aspect, general statistics on recent operations are used. For the second, we utilize a statistical normality test and counting of outliers as a way of measuring the movement irregularity. By outlier number, we mean the number of movement duration that are outside of interval $[\mathrm{mean}-3\cdot\mathrm{std}; \mathrm{mean}+3\cdot\mathrm{std}]$ with mean and standard deviation calculated for the whole window, excluding operation with longest and slowest time of movement.

3) Weather conditions features: There are two recorded weather parameters that can have an impact on failures:

temperature (due to its impact on material properties, it is expected that it has a direct consequence on point machine operation characteristics, such as a higher density of grease slowing down the movement of engine)

and *humidity* (a higher level is a risk factor for failure if the electronics is not properly sealed).

Window length	Correct operations	Failures	Number of devices
10	3 892 007	1842	432
20	3 887 229	1814	413
30	3 881 849	1791	396
45	3 873 425	1764	384
60	3 864 713	1737	373
100	3 839 490	1672	354

TABLE I
Number of valid operations for different window lengths

C. Features extraction

To summarize, the following features have been extracted from the data:

- Operations record features: 1) Total number of operations, 2) Number of operations performed recently (within last 72 hours), 3) Time since the last failure, 4) Numbers of preceding failures;
- Weather conditions features: 5) Humidity, 6) Temperature;
- Time of movement features (calculated for specified number of preceding operations): 7) Mean, 8) Standard deviation, 9) Maximum, 10) Minimum, 11) Anomaly score (statistic of Shapiro-Wilk normality test), 12) Number of outliers.

Several time window sizes have been experimented with to assess the impact of its size on the performance of the classifier. Window size varies from 10 to 100 operations. These correspond, on average, to a few days of operations.

Let us note that for every device and set window length w, the first $2 \cdot (w-1)$ recorded correct operations are not preceded by w operations and cannot have the time-based features calculated in the same way as other events. Because of that, they are discarded from further analysis. The same goes for failures that happen at the beginning of the record.

This results in different number of events for different sizes of the window; they are presented in the table I.

One issue in our approach is that operations recorded right before and after a failure belong, at the same time, to time window related to this failure and subsequent/preceding correct operations. This is problematic, as we are expecting that the failure window contains information distinguishing it from time windows that do not end in failure. We do not address this issue in this study.

IV. Classification of point machine events

A. Challenges to build the classification model

There are a few issues that need to be solved during the building of a classification system from recorded operations: events representation, choice of the building method, dataset balance, results validation and interpretation.

The first issue is to find a representation of events in terms of numerical features that have practical meaning. The features are derived in a straightforward manner based on heuristics for the factors that we expect to play a role in failures. Some of the features are based on the preceding operations. For that

a choice of the length of preceding period needs to be made. This is described in part III-C. The features themselves are presented in part III-C.

The second issue is to choose a method for the classification of the events themselves. In our study, we focus on the use of a machine learning algorithms. There exists many algorithms that can be used for such a task and a selection should be made. This issue is discussed in V-C.

The third issue is the lack of balance in the data set that can results in poor classification performance for the classifier. In extreme cases trained model can classify all events as the more prevalent class (i.e. correct operations). There are two basic ways of dealing with this problem [8]: to re-sample the dataset in order to make it more balanced, or to adapt the machine learning algorithm so that it is more sensitive to less represented class. This process is studied in V-A2.

The fourth issue is concerned with the validation of the results to figure out which model is the best and whether its performance is satisfying. For that, we use of the F-metric as a performance metric. This issue is discussed in V-B2.

B. Overview of the global workflow

The conceptual steps of the milestones of our approach, from raw data to the final results, go as follows:

1) Extraction of events (points movements and failures) from raw logs
2) Preprocessing of events: removal of outages, repeating alarms and other recorded events that are artifacts of the system;
3) Definition of features (the presented analysis starts here);
4) Calculation of features (specific technical details - length of time windows etc.);
5) Division into training and test set;
6) Choice of machine learning methods, parametrization;
7) Choice of assessment metrics and results evaluation.

In this milestone, steps of practical computations can be highlighted:

- Parsing of messages
- Events extraction
- Events validation
- Calculations of features
- Cross validation: Training and test dataset division; Model learning and validation; Metrics calculation; Summarizing results

The first three steps are conducted according to the methodology described in [9].

V. Experiment

In this section subsequent steps of calculations are described. After detailing data preparation, basic validation and chosen machine learning methods are presented. Then results for decision trees, random forests and bagging are summarized.

A. Data preparation

1) Train and test set construction: The events from all 6 train stations are merged. A 10-fold cross validation is performed with randomized events to evaluate the accuracy of the model.

The first problem that needs to be solved is the lack of balance of the classes in the data set, as there are around 1800 successful movements per single failure on average (there are 1805 failures and around 3,200,000 correct operations). Lack of balance in the data is a well researched issue and several solutions exist [8]:

- Data-level approaches - These include randomized or informative (rule-based) resampling of the data set and synthesis of new data.
- Algorithm adaptation - Introduction sensitivity toward less frequent class to increase its impact on the learning process.
- Cost-sensitive learning - Augmentation of the costs to decrease the impact of the unbalance of the data set.
- Boosting - Combining multiple learning instances in order to improve their ability to generalize.

2) Dataset resampling: In our approach, we propose to balance the data set by resampling it: successful operations are undersampled. Additionally, failures are oversampled for ensemble methods. Oversamplling is done exclusively for ensemble methods in order to avoid overfitting.

Let us stress that there are no general rules on what the ratio of classes should be; this is strongly specific to the data and the method utilized. Therefore, the right balance between classes needs to be solved for each of learning method. Basically, our aim is to include as much correct operations as possible without decreasing the ability to generalize, especially regarding detection of failures. As a consequence, for each algorithm, several values have been checked in the following way.

Number of failures per operations is increased (5, 10, 50, 100, etc.) until the accuracy on the test set starts to decrease significantly. For comparison, whole dataset without resampling is used as well.

B. Validation

1) Baseline: Our baseline to compare with is random classification, that is, classification according to binomial distribution with probabilities of classifying the event as failure or correct operation being, respectively, $p_f = \frac{n_f}{N}$ and $p_{corr} = \frac{n_{corr}}{N}$, where n_f (resp. n_{corr}) is the number of recorded failures (resp. correct operations) and $N = n_f + n_{corr}$ is the total number of recorded events. Indeed, classification accuracy for a learning model should be higher than the certain percentile (denoted further by α) of the mean of N random variables defined above. It means, that the accuracy a should fulfill $a > q_\alpha(X_N)$, where q_α is a quantile of order α with X_N being a sum of N random variables equal to 1 or 0 for, respectively, properly or improperly classified operation. Using the central limit theorem [10], this is equivalent to

$$a > p_f + q_\alpha \sqrt{\frac{p_f(1-p_f)}{N}}, \tag{1}$$

with q_α being the quantile of order α of standardized normal distribution.

2) Metrics: For the selection of the most efficient model a metric for model performance assessment is introduced. This allows for quantification of their performance and standardization of comparison of different models. There are various metrics that can be used for that, but not all of them fit the specificity of the studied situation.

In [11], various metrics for model assessment are discussed. Some of these metrics are presented hereafter, each defined from the confusion matrix as in Table II where failures are denoted 'positive cases' and correct operations are denoted 'negative cases'.

	Classified as failures	Classified as correct
Failures	True Positive (TP)	False Negative (FN)
Correct operations	False Positive (FP)	True Negative (TN)

TABLE II
CONFUSION MATRIX.

The *Accuracy* metric is valued as:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}. \tag{2}$$

This metric is not suitable for imbalanced datasets, as the prevailing correct operations might overshadow the very low fraction of correctly classified failures.

The two following metrics, and their aggregation, are often more suitable in presence of unbalanced datasets.

The *Precision* corresponds to the correctness of the prediction of the classifier when it predicts a failure. It is valued as:

$$Pre = \frac{TP}{TP + FP} \tag{3}$$

The *Recall* corresponds to the ratio of failures that are correctly predicted among all existing failures in the dataset. It is valued as:

$$Rec = \frac{TP}{TP + FN} \tag{4}$$

These two measures can be aggregated to build the *F-measure*. This metric is defined as a harmonic mean of precision and recall:

$$F = \frac{2}{\frac{1}{Pre} + \frac{1}{Rec}} = \frac{2 \cdot Pre \cdot Rec}{Pre + Rec} \tag{5}$$

Let us note that, as a harmonic mean, F-measure is more sensitive to the lower value than the arithmetic mean (means inequality). In [11] it is encouraged to use the F-measure in case of highly imbalanced data with emphasis on correct classification of less frequent classes.

For actual model selection, the metrics are applied to the averaged confusion matrix coming from the 10-fold cross-validation.

Metric	Depth 12	Depth 18	No max depth
Precision	0.48	0.0041	0.082
Recall	0.64	0.0065	0.80
F-measure	0.0082	0.013	0.081

TABLE III
BEST RESULTS OBTAINED WITH SINGLE DECISION TREE FOR DIFFERENT TREE DEPTH

C. Machine learning models

The machine learning models used in our experiment are decision-tree based: decision trees, random forests, and bagging of decision trees [12].

The main advantages of these two models lies in the fact that they are not distance-based (which is desired, as it reduces the need to introduce arbitrary distance metric in the process). Furthermore, the size of the dataset has a modest number of dimensions, making some tools such as neural networks more problematic to use due to the high risk of overfitting.

All calculations has been done using Python libraries [13], [14] and [15].

1) Decision trees: There are three important aspects for the configuration of a tree. The first is the way in which the splits of the nodes on the tree are calculated. At this point, we do not interfere with this process. The second is linked with the complexity of a tree - the number of features and samples that are taken into account when performing a split. The third is the size of the tree - it can be controlled by setting maximum number of nodes or its depth.

At this point, we only work with the third aspect. For the depth of the tree, we go from small to big to check the impact of size of the tree on the accuracy of the classifier.

Additional feature of the decision trees is modification of the cost function by inverse of class frequencies which reduces impact of lack of dataset balance. This is used further in V-D0a.

2) Ensemble methods: The next step is to use ensemble methods, i.e. forests of decision trees, that collectively can provide better classification quality and insight into the classification process.

Two ensemble methods are tried here: random forests and bagging. At this point, for both methods, multiple algorithm parameter configurations are used and fitted to the data with with different parameters each time. The validation procedure for those ensemble methods is the same as for random forests - cross validation and metrics described in V-B2.

From the previous analysis of the single decision tree performance we know that the depth of the tree providing the best performance is around 6. Therefore, for the forest we use lower values.

The decision trees used as basis for the forest are configured in the way described in V-C1.

D. Classification with decision trees

To summarize, we use the following settings for the algorithm:

1) Dataset balance: None, algorithm weights modification, undersampling, both algorithm modification and undersampling.
2) Window size modification; we take specific sizes, from short to long: 10, 30, 60 and 100.
3) Finally, there are algorithms specific parameters. For tree, those are depths that are set as 1, 6, 12, 24.

In general, it gives $4^4 = 256$ parameters sets independent calculations runs.

At first, the dataset balance issue is solved. We take a single decision tree, train it on the data merged from different sources, with three different window sizes and test all balance configurations. Different max depths are checked and the comparison is made.

To decrease the number of calculations, we decide to cut them short - that is, start from the lower value of the parameters and stop them once there is no visible change to the quality of results (or the quality decreases). Additionally, dataset balance and algorithm parameters are chosen only for all devices merged together.

We do as follows. At first, the choice of dataset balance method is done. Then window size is chosen. Finally, impact of tree size on the classification is looked at.

a) Training, dataset division - results: For the setting with no dataset balancing, the best obtained precision on test set was around 6%. For most of other parameters values, no failure has been correctly classified. Clearly, a form of dataset balance needs to be introduced.

For the balance by undersampling, there are a few possible ratios of operations-to-failure. We pick randomly n_{corr} correct operations so that the ratio $n_{corr}/n_{fail} = n$ for specific n equals, consecutively, 5, 10, 50, 100 and 500. n_{fail} is a number of failures in the dataset. In this case, the classification accuracy is much higher, allowing for the precision at level of around 60%.

For all settings taken into consideration (depth of the tree and window size), augmenting the weights gave the best results in terms of precision of detection of failures and correct operations. Precision for failures has been on a very close level (for the setting with both undersampling and weights augmenting), but the precision in classification of correct operation has been better for settings with only weights augmentation, presumably due to more sparse training set not covering misclassified cases.

Although, interestingly, while the precision has been lower for undersampling methods, the method wasn't overtraining as much for the case of greater depths (equal to 12 and 18), but in this case the precision was still below the level obtained by weights augmenting. On the other hand, it this effect could be simply due to the smaller training set.

b) Window size: There is a slight impact of window size in terms of precision of failures classification.

Better values are achieved for rather short windows, although as the parameters calculated for windows are not decisive for short trees, it could be due to the lower amount of data points (longer window means rejection of operations from the start).

Metric	Resampling, training set	Resampling, testing set	No resampling, training	No resampling, testing
Precision	0.96	0.0024	0.0026	0.0026
Recall	0.8	0.8	0.79	0.79
F-measure	0.88	0.0049	0.0052	0.0052

TABLE IV
EXAMPLE OF CLASSIFICATION RESULTS FOR BAGGING CLASSIFIER WITH MAXIMUM DEPTH 2, 10 ESTIMATORS AND 0.1 FRACTION OF SAMPLES.

c) Depth of the tree: From the experiments, we can infer that a tree depth around 6 gives the best results. In most of the settings, the precision of failures identification is around 75-80% but decreases once the tree gets deeper (to max depth of 12 or 18). It presumably results from overtraining. The exact value at this point is not very important, as it might change with changes in the method in the future.

d) Decisive features: We are able to look at the content of the trees, that is, the features that are present in the splits.

The features included in shortest trees (depth equal to 3) are the one derived from operations and failures record - time since last failure, temporal operations number, preceding failures number. Additionally, statistics of time movement—mean and minimum—also appear in them.

For trees of depth 4, additional statistics appear—standard deviation and normality of time of the movement. Also, temperature is included.

For trees of depth 6 and more, all features are used. Let us note that such a tree contains 31 splits, thus some features are reused multiple times.

E. Classification by Random Forest

The following parameters are set tested, with calculations performed on most of the combinations (some configurations have been skipped for undersampling once it was recognized that all events are classified as correct operations regardless):

- Dataset balance: None, undersampling, weights, weights and undersampling
- Window length: 10, 20, 30, 45, 60, 100
- Trees depth: 2,3,4,5
- Number of trees: 10, 25, 50, 100

For oversampling, there are three ratios (number of failures: number of correct operations): 1:10, 1:100 and 1:500.

a) Dataset balance:

1) No balancing – for this method the classification results are poor - the algorithm classifies all the events as correct operations, regardless of its detailed configuration.
2) Undersampling – retains much better precision, although it can be due to much smaller testing dataset with correct operations. With no additional weights balance, there is very little success in classification of failures, as everything tends to be classified solely as a correct operations.
3) Weights balancing – this specific balancing method provides the best results regardless of dataset resampling.

	Depth	Precision	Recall	F-measure
Training	2	0.0030	0.80	0.0060
Testing	2	0.0030	0.79	0.0061
Training	3	0.0035	0.78	0.0070
Training	3	0.0035	0.78	0.0070
Training	4	0.0043	0.77	0.0086
Testing	4	0.0042	0.76	0.0085

TABLE V
BEST RESULTS FOR EACH VALUE OF A TREE DEPTH

Forest size	Precision	Recall	F-measure
10	0.0039	0.75	0.0077
25	0.0041	0.74	0.0082
50	0.0042	0.75	0.0083
80	0.0043	0.76	0.0085

TABLE VI
BEST RESULTS ON TESTING DATASET FOR EACH VALUE OF FOREST SIZE

b) Window length: The differences of results between different window lengths turns out to be very little.

c) Trees configuration: We decided to set the maximum depth of the tree to be between 2 and 5. Generally, deeper trees allow for better results in terms of precision (and, subsequently, F-measure), at the slight expense of recall; the values in most cases change from, for precision, from 0.0025 to 0.0035, for recall from 0.75 to 0.81, with F-measure ranging between 0.0049 and 0.0069.

d) Forest sizes: Forest size has visible impact on the classification accuracy. While the smallest forest has clearly inferior overall performance for shallow trees, the differences get much smaller once the trees get deeper, although, bigger forests tend to provide slightly better results (usually, around 1% better for similar configuration).

Results summary: Overall, there is not much difference in results for different configurations.

Interestingly, in most cases the metrics calculated for classification results on training and testing dataset are the same, and overall, they do not differ strongly, at most up to few percents. The standard deviation calculated for the confusion matrices coming from the different training and testing dataset decompositions are

Judging by the standard deviation of confusion matrices coming from different training and testing dataset decompositions for the same parametrization, in most cases there seems to be the same level of differences between slightly different parameterizations and different training and testing datasets.

F. Bagging

As it was shown before, the window length parameter does not have a major impact on the final results, so we focused only on three values that give the best results.

In bagging, a set of decision trees is used with each of the classifier learning on the randomly chosen subset of data. Sampling has been done with replacement. Additionally, few failures are oversampled. Due to the poor results for lack of balance, calculations without balancing weights has been skipped for the dataset not balanced by resampling.

The parameters are set as follows, with calculations performed on all combinations:

- Dataset balance: oversampling, weights, weights + oversampling
- Window length: 20, 30, 60
- Trees depth: 2, 3, 4
- Failures fraction: None, 0.1, 1.0, 5
- Trees number: 10, 25, 50, 80

For oversampling, there are three ratios of number of failures to number of correct operations: 1:10, 1:1 and 5:1. These had been tried for the window size of 30, as this is the size that gives the best results.

a) Window size: There is no big impact of the window size on the results. For different values of window size, the results are very close (that is, the difference of metrics is less than 2-4%) and there is no single value better than the others.

b) Dataset balance: Apart from weights modification, typical for decision trees, dataset balancing by resampling has been tested. Given the previous negative experience with undersampling of the correct operations set, this time oversampling of failures has been tried. That is, once the dataset without any changes has been divided into training and testing datasets, the failures in the training dataset had been oversampled to a specified failures fraction. Despite oversampling, it was decided to keep the balancing weight in the calculations as well.

Oversampling for training purpose did not provide much of an advantage, and the performance metrics for the testing dataset results are very close to those without the procedure of resampling.

Let us stress that the difference between values for training and testing is strongly impacted by the dataset characteristics, as multiple instances for training with oversampling are repeated, which is not reflected in the testing set.

Overall, oversampling for failures does not provide models with the same quality in terms of F-measure as the model without it. It could be due to the overlearning of specific events, that are repeated dozens or hundreds of time to make up for worse proportion in the original data.

In further work we focus only on the setting with weights.

c) Trees configuration: For the trees themselves, the most important parameter is their maximum depth. It has a strong impact on the quality of the classification.

On the other hand, in the setting where the failures-to-correct operations had been 5, for deeper forests the algorithm started to classify the events very poorly—classifying the majority of correct operations as failures, thus degrading the results. Apart from it, the results for testing dataset have been almost identical, at least in terms of the introduced metrics, despite different values of metrics in training dataset in some instances.

Results presented in Table V show that the depth of the tree has a profound impact on the accuracy of the classifier. As we can see, the metrics for both test and training are almost identical.

Maximum samples	Precision	Recall	F-measure
0.1	0.0043	0.76	0.0085
0.25	0.0036	0.79	0.0072
0.5	0.0034	0.80	0.0067

TABLE VII
BEST RESULTS FOR EACH VALUE OF SAMPLES FRACTIONS

Metric	Single decision tree	Random forest	Bagging
Precision	0.0065	0.0035	0.0043
Recall	0.48	0.78	0.76
F-measure	0.013	0.0071	0.0085

TABLE VIII
BEST PERFORMING MODELS AMONG ALL TESTED CONFIGURATIONS OF DECISION TREE, RANDOM FOREST AND BAGGING

d) Forest size: Multiple forest sizes had been tried. In Table VI, it can be shown that the deeper forests provide better classification performance, but within limits - we can see the slowdown of improvements with increase from 25 to 80 estimators in a forest.

e) Training samples fraction: In Table VII, it can be seen that the classification quality decreases as the samples size increases, which might be somehow surprising. We do not have an explanation for this.

Result summary

The standard deviation for all elements of confusion matrix coming from the 10-fold cross validation has been computed. They are lower at least one degree of magnitude than the mean value. It is rather reasonable and expected, and they are not included in the analysis here.

In Table VIII, it can be seen that while the single, unconstrained decision tree reaches the highest F-measure, it is in part due to its overtraining and overclassification of the events as correct. We prefer models that provide correct classification of not only one class.

Overall, we prefer to call bagging methods as most effective. It seems, that part of its advantage lies in the rather low fraction of failures taken into training of each tree, as seen in Table VII.

VI. CONCLUSION

In this paper, a new approach for detection of point machines failures using easily obtainable data (coming from CTC system and public weather stations) is presented.

A complete framework for information extraction and classification is shown. We presented and discussed a way to provide a compact numerical representation of the most important aspects of recorded events, along with a choice of machine learning algorithms.

We show that machine learning methods, specifically decision forests, have an ability, to a degree, to distinguish the conditions under which point machines operate properly or not.

The research is in an initial stage and there are still issues that need to be resolved before the method could be used for actual failures prediction and diagnosis. The main concern is the decrease of operations falsely classified as failures, which is too high. Possibly, this could be done by alternating the setting in which calculations are done. Also, some technical details (specifically features construction) could be altered as well. Another interesting result would be the extraction of the conditions that increase the risk of failure.

REFERENCES

[1] T. Asada, C. Roberts, and T. Koseki, "An algorithm for improved performance of railway condition monitoring equipment: Alternating-current point machine case study," *Transportation Research Part C: Emerging Technologies*, vol. 30, pp. 81 – 92, 2013.

[2] C. Letot, P. Dersin, M. Pugnaloni, P. Dehombreux, G. Fleurquin, C. Douziech, and P. La-Cascia, "A data driven degradation-based model for the maintenance of turnouts: a case study," *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 958–963, 2015.

[3] A. Pełka, "Diagnozowanie urzadzeń sterowania ruchem kolejowym na przykładzie napedu zwrotnicowego," Ph.D. dissertation, AGH University of Science and Technology, 2009.

[4] J. Sa, Y. Choi, Y. Chung, H.-Y. Kim, D. Park, and S. Yoon, "Replacement condition detection of railway point machines using an electric current sensor," *Sensors*, vol. 17, no. 2, p. 263, 2017.

[5] T. Xu, G. Wang, H. Wang, T. Yuan, and Z. Zhong, "Gap measurement of point machine using adaptive wavelet threshold and mathematical morphology," *Sensors*, vol. 16, no. 12, p. 2006, 2016.

[6] J. Lee, H. Choi, D. Park, Y. Chung, H.-Y. Kim, and S. Yoon, "Fault detection and diagnosis of railway point machines by sound analysis," *Sensors*, vol. 16, no. 4, 2016.

[7] I. of Meteorology and W. M. N. R. Institute, Warsaw, Poland. [Online]. Available: https://dane.imgw.pl/datastore

[8] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 04, pp. 687–719, 2009.

[9] I. Doboszewski, S. Fossier, and C. Marsala, "Extraction de connaissances sur les defaillances de compteurs d'essieux," *Revue des Nouvelles Technologies de l'Information*, vol. Extraction et Gestion des Connaissances, RNTI-E-34, pp. 311–316, 2018.

[10] J. Jakubowski and R. Sztencel, *Wstep do teorii prawdopodobieństwa*. Script, 2001.

[11] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, "Svms modeling for highly imbalanced classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 1, pp. 281–288, 2009.

[12] J. Gareth, D. Witten, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer, 2009.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[14] E. Jones *et al.*, "SciPy: Open source scientific tools for Python," 2001, [Online; accessed 2017-02-10]. [Online]. Available: http://www.scipy.org/

[15] D. Ascher, P. F. Dubois, K. Hinsen, J. Hugunin, and T. Oliphant, *Numerical Python*, ucrl-ma-128569 ed., Lawrence Livermore National Laboratory, Livermore, CA, 1999.