Fuzzy Condition Monitoring of Recirculation Fans and Filters

M. Gerdes · D. Scholz

Abstract A reliable condition monitoring is needed to be able to predict faults. Pattern recognition technologies are often used for finding patterns in complex systems. Condition monitoring can also benefit from pattern recognition. Many pattern recognition technologies however only output the classification of the data sample but do not output any information about classes that are also very similar to the input vector. This paper presents a concept for pattern recognition that output similarity values for decision trees. The concept can be used on top of any normal decision tree algorithms and is independent of the learning algorithm. Performed experiments showed that the concept is reliable and it also works with decision tree forests to increase the classification accuracy.

1 Introduction

Aircraft systems can be complex and difficult to monitor. A good condition monitoring does not only depend on good sensors and a good model but also interpreting of the numbers. Interpreting sensor data however is not a trivial task. Classification and condition monitoring as shown in this paper also reduces the amount of information that needs to be monitored. Often an expert is needed to interpret the data and make a meaningful "classification".

M. Gerdes
Hamburg University of Applied Sciences
Aero - Aircraft Design and Systems Group
Berliner Tor 9, 20099 Hamburg, Germany
E-mail: mike.gerdes@haw-hamburg.de

D. Scholz
Hamburg University of Applied Sciences
Aero - Aircraft Design and Systems Group
Berliner Tor 9, 20099 Hamburg, Germany
E-mail: info@ProfScholz.de

CEAS Aeronautical Journal
An Official Journal of the Council of European Aerospace Societies
ISSN 1869-5582

CEAS Aeronaut J
DOI 10.1007/s13272-011-0021-9

Received: 4. April 2011
Revised: 26 July 2011
Accepted: 27 July 2011

This paper is based on a presentation at the German Aerospace Congress, September 27-29, 2011, Bremen, Germany

GERDES, Mike; SCHOLZ, Dieter: "Fuzzy Condition Monitoring of Recirculation Fans and Filters". In: Springer, CEAS Aeronaut. J., DOI 10.1007/s13272-011-0021-9, 2011. - Online First, URL: http://www.springerlink.com/content/8151u1028n6x127u

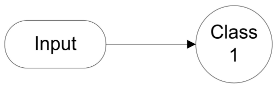

Fig. 1 Common classification mapping of one input vector to one class

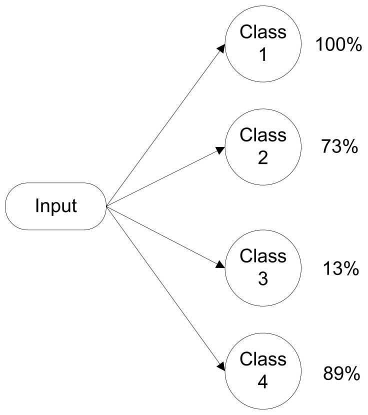

Fig. 2 Classification mapping of one input vector to one class and output of similarity

With pattern recognition and classification replacing the human expert by a computer algorithm is possible. Pattern recognition and classification are a typical application of artificial intelligence technologies. With pattern recognition it is possible to find complex patterns that are hidden in the sensor data of a system. Classification maps an input vector to a class based on a learned or given pattern. In case of system monitoring the class can be a failure or condition of a system.

Most classifiers map an input vector to one output class (Figure 1; the input vector is mapped to "class 1"). However for monitoring systems and reducing NFF (No Failure Found) failures knowing the probability of an input vector belonging to all possible classes, instead of only the most likely class (Figure 2) is useful. The input vector is still mapped to "class 1", but the input vector also matches the pattern of "class 4" in 89 % of the criteria for "class 4". Artificial Neural Networks (ANN) can output the similarity of an input vector to other classes, if one output node is available for every class. But ANNs have their own disadvantages compared to other methods [8][4] . There are several different methods for calculating the similarity of signals. Many of these methods are used in speech recognition [1].

In the project PAHMIR (Preventive Aircraft Health Monitoring for Integrated Reconfiguration) decision tree classifiers are used to interpret system sensor data and classify the system conditions. Decision trees are a simple and fast classifiers with feature extraction, learning and a high robustness. Decision trees are a method from the area of artificial intelligence and are used for decision making and classification [2][8]. They are often binary trees, where each node has an if-then-else function on an attribute of the sample

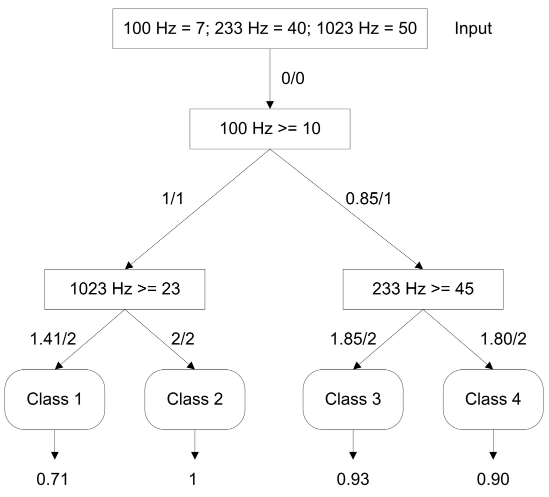

Fig. 3 Fuzzy Decision Tree Evaluation

data. Advantage of decision trees is the simple structure, the fast calculation and the inherent feature extraction. The ID3 algorithm (Iterative Dichotomiser 3, published by J. Ross Quinlan in 1986, used to generate decision trees [9]) was the first algorithm to construct decision trees. ID3 had some problems and was improved. The improved version of ID3 is C4.5 [10]. It enhances the ID3 algorithm with the ability to handle both discrete and continues attributes, it can handle samples with missing attributes and supports pruning of the tree at the end of the algorithm (removing branches from the tree). The algorithm to build a decision tree uses the concept of information gain to choose attributes from the data and build the tree. Output of a decision tree is only the most likely class for one data sample.

To get more information out of the classification the decision tree evaluation algorithm was modified to output the probabilities of all trained classes. This modification was done without changing the learning algorithm for decision trees and can be used with any binary decision tree. An example is shown in Figure 3. The figure shows a decision tree for deciding which class a sample belongs to. The example input vector is one frequency power (energy per unit time; based on the power spectrum) observation (100 Hz, 233 Hz and 1023 Hz). Other features as the maximum amplitude or average power in the frequency domain are also possible. The feature vector depends on the preprocessing. Each node in the tree is related to an attribute. The path from one node to another is labeled with the taken decisions. Weights are based on the distance of the attribute value from the sample value in the observation.

The highest calculated value (between 0 and 1) for every possible decision is returned at the end of the evaluation. Thus all possible paths are evaluated and we gain a measure of similarity of an input vector to all classes. In case of the example the evaluation of the decision tree classifies the sample as an example of Class 2 (weight = 1). The sample is also in similar to Class 3 (weight = 0.5) and 4 (weight = 0.45). Advantage of this approach is that the similarity of a data sample to different conditions can be calculated and that the decision tree generation algorithm does not need to be changed. It should be noted that the concept is designed in such a way that the characteristics of one attribute (mean, minimum, maximum ...) does not need to be known. In addition the input vector for training and classification does not have to be modified in any way to fit the new algorithm. Carrying these ideas further, this paper shows how a fuzzy evaluation of decision trees using numerical attributes can be used to gain more information about a system than just the current condition.

2 Concept

This section shows the concept that was developed for a fuzzy evaluation of decision trees. A decision tree is generated traditionally, but uses a fuzzy evaluation for the evaluation of an input vector:

1. Generation of feature vectors
2. A decision tree is generated with C4.5 based on labeled data samples.
3. The decision tree is evaluated using the fuzzy evaluation concept.

The fuzzy evaluation calculates an output for every leaf. Every output value of a leaf is between 0 and 1 and represents the similarity of the sample to all classes (see Figure 3 for an example). Similarity is a function of the distance of an input vector to a class. The similarity is calculated based on a weighting function of the decisions taken (node evaluations) to classify the sample. For condition monitoring and maintenance the similarity can indicate possible other faults and conditions of a system. The proposed fuzzy evaluation works as follows:

1. Every path between two nodes or a node and a leaf has two labels: *PathDepth* and *PathWeight*.
2. Start at the root node
3. *PathDepth* and *PathWeight* are 0 for the root node
4. Evaluate the node condition
5. Calculate path labels:
   (a) If the evaluation of the condition is true then label the *True* path with *PathDepth* +1 and *PathWeight* + 1.
   (b) If the evaluation of the condition is false then label the *False* path with *PathDepth* +1 and *PathWeight* + 1.
   (c) Label the other path to sub-nodes with *PathDepth* +1 and *PathWeight* + $nodeweight(AV, SV)$. See Equation 1 and 2
6. Choose a new node, with a labeled path to its parent.
7. Use the path labels for *PathDepth* and *PathWeight*.
8. If the node is no leaf then continue at 4.

9. If the node is a leaf then return $\frac{PathWeight}{PathDepth}$ and the leaf label and continue with another node.

It is assumed that the tree is a binary tree with numerical attribute values. Returned values of leafs are between 0 and 1. The weight for each node (*nodeweight*) is calculated as shown in Equation 1. Equation 2 and Equation 3 limit the function values. *Nodeweight* can be between 0 and 1. AV (attribute value) is the value of the sample for the condition of the current node. SV (split value) is the value of the node, which marks the border of the condition e.g. the split value of "$x \leq 17$" is 17.

$$nodeweight(AV, SV) = 1 - \frac{|AV - SV|}{2SV} \tag{1}$$

$$if\ (nodeweight(AV, SV) < 0) \Rightarrow nodeweight(AV, SV) = 0 \tag{2}$$

$$if\ (nodeweight(AV, SV) > 1) \Rightarrow nodeweight(AV, SV) = 0 \tag{3}$$

It is possible to use a different weighting function. The requirement is that the weight for each node needs to be between 0 and 1 for the algorithm. If the weight is higher than 1 the result at a leaf can be higher than 1. The given weighting function was chosen for different reasons. One reason was that a black box approach for used for the monitored system and it is unknown what input values of vectors mean. It cannot be assumed that the training vectors for the algorithm contain the max or min values or even represent an average. These circumstances make it difficult to use absolute values. The only values that are available without adding additional information or calculation to the algorithm is the split value of a node. Also the correct classified class needs to have a value of "1" at the corresponding leaf. Choosing $2 \cdot SV$ as the limit where the weight will be 0 was an arbitrary choice which is based on some tests with input vectors. It is possible to use a higher or lower value. If the maximum and minimum values for features are available then it is possible to use those as limits and use Equation 4 and Equation 5.

$$if\ (AV \leq SV) \Rightarrow nodeweight(AV, SV) = 1 - \frac{SV - AV}{SV - min} \tag{4}$$

$$if\ (AV > SV) \Rightarrow nodeweight(AV, SV) = 1 - \frac{AV - SV}{max - SV} \tag{5}$$

A disadvantage Equation 4 and Equation 5 is that every decision returns a *PathWeight* of more than 0, which can result in quite high similarity values. The concept was designed with only numeric values in mind. However it is possible to use Boolean and discrete values as well with a small modification of the process.

Nodes with Boolean attributes If Boolean attributes are used instead of numerical values then the weight is assumed to be 0.

Nodes with discrete values If a node does have more than two children (one for every possible value of the attribute) then only the path linked to the evaluation of the node condition is weighted with 1. For every other unlabeled path to a child the weight needs to be calculated separately.

2.1 Fuzzy Decision Tree Evaluation Example

Below is an example. Each path from one node to another is labeled with *PathWeight* and *PathDepth* in the form of *PathWeight/PathDepth* e.g. 2/5. In Figure 3 an input vector set is given. The input vector (values of the power spectrum of the transformed input signal) contains the power (energy per unit time) of the frequencies at 100 Hz, 233 Hz and 1023 Hz. At the first node the 100 Hz value is checked if it is larger or equal than 10. The power is not larger than 10 (it is 7) so right path which is the *False* path get +1/+1. For the other path a 1 is added to the *PathDepth* and a 0.85 (the similarity) to the *PathWeight*. In the next step the 1023 Hz node is evaluated. The input vector does have a power at 1023 Hz which is higher than 23, so the *True* path gets +1/+1 for a total of 2/2 (+1/+1 to the 1/1 from the parent path). The other path gets a +0.41/+1 for a total of 1.41/2. The same process is done for the right hand node (233 Hz). Evaluating the node gives a +1/+1 to the *False* path and a +0.95/+1 to the other path. In the last step the weight of the leafs and the classes are calculated. The input vector is classified as class 2. The similarity to class 1 is 0.71, 0.93 to class 3 and 0.9 to class 4. A similarity of 0.93 means that the values do not have to move much to switch the classification result to class 3.

Only the attributes, which define a certain class are evaluated during the evaluation. Attributes, which the algorithm did not include in a decision path are not checked. E.g. after the root node either 1023 Hz or 233 Hz is checked for classification of a class. So a class is either defined by 100 Hz and 1023 Hz or by 100 Hz and 233 Hz, but not by all three attributes. Which attributes are in a decision path and are checked is decided by the decision tree generation algorithm (in this case C4.5). For more complex examples an attribute can be checked multiple times (with different decision values) in a decision path and attributes, which were neglected earlier, are checked again. However even in the simple example all attributes are at least evaluated once, because the algorithm checks all paths. And the evaluation results are used to calculate the similarity values.

2.2 Fuzzy Decision Tree Forest Evaluation

A decision tree forest can also be evaluated using the proposed concept [3][5]. Each decision tree in the forest is evaluated separately using fuzzy decision tree evaluation. If all trees are evaluated then the weights (similarities) for a class from all trees are added together and are divided by the number of trees (taking the average of a class over all trees in the forest). This is done for all available classes.

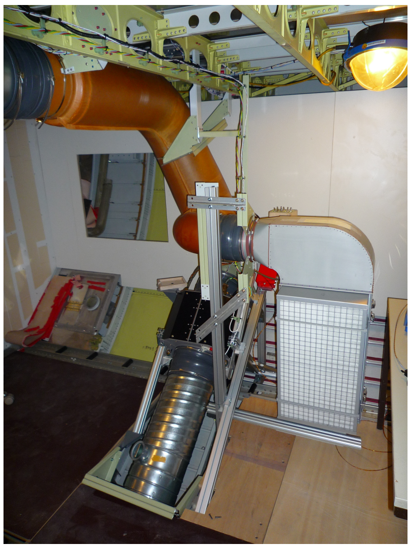

Fig. 4 Test Rig at Airbus Operations GmbH

Table 1 Experiment Conditions

Value 1	0	15	30	45	60	75	90
Valve 2	0	15	30	45			

3 Experiments

Experiments were performed to ensure the performance of the concept. Goal of the experiments was to evaluate the fuzzy decision tree evaluation for a single decision tree and for a decision tree forest.

3.1 Database

Data for the experiments was recorded on a test rig. The test rig resembles a part of the air conditioning system of the A340-600. Focus of the test rig is the HP recirculation fan and filter. 29 different conditions were recorded. The test rig is equipped with two valves, one valve at each end of the two tubes. Table1 shows the possible settings for both valves. 20 samples of one second duration were recorded for every condition of the test setup.

Valve1 is the inlet valve and *Valve2* is the outlet valve. 28 conditions were recorded (Table 1). In addition to these 28 conditions a *not running* condition was recorded. In total 580 samples were recorded. The samples are labeled *valve1/valve2* e.g. 15/0 which represents the test case valve 1 is closed by 15 degree and valve 2 is fully open.

3.2 Setup

The Java software Weka was used to perform the experiments. J48 (C.45) was used to build the decision tree. A signal analysis was done before those data samples were fed into Weka. The signal analysis includes noise reduction, Fast Fourier Transformation, mean/max calculation ... (see [6] for details). An optimized parameter set for the signal analysis was generated [7]. 20 randomly ordered samples of every class were selected for the decision tree generation.

3.3 Single Fuzzy Decision Tree Evaluation

The experiments themselves were simple. A decision tree was calculated using the data samples and the signal analysis parameters. The decision tree was then evaluated with fuzzy decision tree evaluation. Test case "15/0" was used to compare the results. Five different node weighting functions are tested to be able to evaluate the influence of the node weighting function. The average of all 20 fuzzy evaluations of test case "15/0" was taken to reduce the influence of noise.

1. Equation 1 as the default node weighting function
2. Equation 6 as an example of a function where the weights are always close to 1 and 0
3. Equation 7 as a "flat" function with many values close to 1
4. Equation 4 and Equation 5 as a function that always does have a value $0 < x \leq 1$
5. Equation 8 and Equation 9 as a function with more values that are 0 and which scales with the value range

$$nodeweight(AV, SV) = 1 - \frac{|AV - SV|}{0.01 \cdot SV} \tag{6}$$

$$nodeweight(AV, SV) = 1 - \frac{|AV - SV|}{10SV} \tag{7}$$

$$if\ (AV \leq SV) \Rightarrow nodeweight(AV, SV) = 1 - 5 \cdot \frac{SV - AV}{SV - min} \tag{8}$$

$$if\ (AV > SV) \Rightarrow nodeweight(AV, SV) = 1 - 5 \cdot \frac{AV - SV}{max - SV} \tag{9}$$

Each equation(set) replaces Equation 1. Equation 2 and Equation 3 are left unchanged. For each equation the same parameter set and decision tree was used.

Table 2 Averaged evaluation results

Class	Eq. 1	Eq. 6	Eq. 7	Eq. 4 & Eq. 5	Eq. 8 & Eq. 9
0/0	0.9585	0.5714	0.9917	0.8294	0.6462
15/0	1.000	1.000	1.000	1.000	1.000
15/15	0.9872	0.8357	0.9974	0.9209	0.8505
30/0	0.9996	0.9213	0.9999	0.8000	0.8000

3.4 Decision Tree Forest

A decision tree forest with three trees was created for the experiments with a decision tree forest. Each tree does have a classification accuracy of at least 90 %. Every tree uses a different signal processing parameter set, which generates different training vectors and test vectors for the evaluation. During the experiments the classification accuracy compared to a single decision tree with a classification accuracy of 95 % was evaluated. The fuzzy evaluated results for the same 20 data samples and neighboring classes as in the experiments for the single decision tree were also evaluated.

4 Results

This section shows the results of the experiments. First the results for a single fuzzy decision tree evaluation are shown and second the results for a fuzzy decision tree forest evaluation are shown.

4.1 Single Fuzzy Decision Tree Evaluation

Table 2 shows the averaged results for five fuzzy decision tree evaluations with data samples of class "15/0". The numbers show that the correct class is classified. For comparison the similarity of the three neighboring classes (0/0, 15/15 and 30/0) is shown. The classes are very similar to class "15/0", but the variance in the results depends on the node weighting function. If a node weighting function is "flat" (meaning that the results are often higher than zero) then the similarities are all close to 1 and have little variance (often between 0.9 and 1). On the other hand if the function is "narrow" (meaning that often results are produced, which are zero) then the variance is higher and similarities can range from 0 to 1.

This effect occurs because the $PathWeight$ is often much higher than zero for "flat" functions. The maximum "narrow" function is, if only 1 or 0 would be valid results. Then the similarity is based on the number of the "True" decision. This may be desirable for some applications, but it hides some information that might be useful for condition monitoring. Small variations of an attribute are not represented in a very "narrow" function, they are filtered out. But often the goal of condition monitoring is not only the similarity but also the ability

Table 3 Similarity matrix

Valve2/Valve1	0	15	30	45	60	75	90
0	0.5714	1.0000	0.9213	0.5000	0.6500	0.7583	0.6550
15	0.6720	0.8357	0.6929	0.6667	0.6512	0.7278	0.4944
30	0.7500	0.5000	0.4000	0.6000	0.4444	0.5714	0.4286
45	0.6500	0.7500	0.5712	0.5667	0.2500	0.5000	0.2500

to detect slight movements in the similarity to predict into which "direction" a similarity is moving if one or more values are modified. Finding a fitting node weighting function depends on the problem and the goals of the application.

Table 3 shows a complete similarity matrix. The matrix contains the results of all leafs of the decision tree for Equation 6. In the matrix the full similarity variance of the results is visible. The table shows that the most similar classes do not always have to be neighbors, but instead they can be classes farther away. Two of the neighbors are very similar, which is the desired result. The similarity of the other classes is mixed. The tendency is that the similarity is lesser if a class does have a higher distance from "15/0".

4.2 Fuzzy Decision Tree Forest Evaluation

This section shows how the results improve, when a decision tree forest is used. The similarity values of the decision tree forest was calculated by taking the average similarity values of the single decision trees.

4.2.1 Classification Accuracy

The accuracy of a classification is defined as the number of correct classifications (CC) in relation to the number of all classifications (TS), where TS is the number of correct classification plus the number of wrong classifications (see Equation 10).

$$Classification\ Accuracy = \frac{CC}{TS} \tag{10}$$

Classification accuracy is a number between 0 and 1, which can be transformed into a percentage value. The accuracy of the decision tree evaluation is not influenced by the fuzzy decision tree evaluation. This is because of the fact that the *True* paths are weighted with "1" while all other paths are weighted with a positive number lower than 1 ($0 \leq x < 1$). The *True* path will always have the highest weight. However it is possible that the classification accuracy of fuzzy decision tree evaluation is slightly lower than for standard decision tree evaluation because of the limits of numerical computations. If the weight of a node is very close to 1 then it is possible that due to rounding errors it is counted as a 1 instead of a value that is lower than 1. This case happened in some experiments and was more frequent, if a 32 bit Java floating point data

Table 4 Averaged fuzzy decision tree forest evaluation

Class	Eq. 1	Eq. 6	Eq. 7	Eq. 4 & Eq. 5	Eq. 8 & Eq. 9
0/0	0.9654	0.6905	0.9931	0.8725	0.7206
15/0	1.000	1.000	1.000	1.000	1.000
15/15	0.9815	0.8665	0.9963	0.9370	0.8736
30/0	0.9976	0.8528	0.9995	0.9286	0.9095

Table 5 Similarity matrix for the decision tree forest

Valve2/Valve1	0	15	30	45	60	75	90
0	0.6905	1.0000	0.8528	0.5972	0.4833	0.5950	0.4718
15	0.6843	0.8665	0.7402	0.5637	0.4929	0.5537	0.4127
30	0.6630	0.5650	0.4061	0.6071	0.4648	0.6071	0.6111
45	0.6667	0.6533	0.4904	0.6389	0.4087	0.4401	0.4556

type was used instead of 64 bit Java floating point data type. Using a forest with three decision trees increased the classification accuracy from 95 % per single tree to 99 % for the complete forest. This is a significant classification accuracy improvement, which comes at the cost of three times the calculation time. However in the example application of air filter monitoring the time is not a critical factor.

4.2.2 Fuzzy Decision Tree Evaluation

Comparing the results of the same four classes that were evaluated for a single decision tree we get Table 4. The results are similar to a single decision tree with fuzzy evaluation, but the average similarity and the overall classification accuracy is higher. These results show that the fuzzy evaluation also works with a decision tree forest. But if one value changes in the input changes then the influnce on the similarity is less. Decision tree forests with fuzzy evaluation are better at calculating the overall similarity of an input, because different signal anaylsis steps are used and different trees are evaluated. Thus different features are checked and used to calculate the similarity result of a single decision tree.

5 Conclusions

The concept for fuzzy evaluation of decision trees, which is shown in this paper can achieve the desired performance and delivers good results. It is possible to calculate a similarity measurement for classes in a decision tree without changing the algorithm to create the tree. However the design of the node weighting function is important. Paths should be weighted with a low *PathWeight* to get a meaningful similarity measurement. A "flat" node weight function is more sensitive the changes in the input values. On the downside a "flat" function does have a lower variance in the similarity values. "Narrow" node weighting functions have the opposite effect. Fuzzy classification plus decision trees are a powerful tool for condition monitoring especially in the aircraft environment. The fuzzy decision tree evaluation is easily understood, fast and small, which makes it good for use in environments characterized by high safety requirements. With additional optimization of the weighting equation it is possible to increase the variance of the fuzzyfication.

References

1. Rabiner, Lawrence; Juang, B. H. : Fundamentals of Speech Recognition. Prentice Hall, 1993
2. Breiman, Leo; Friedman, Jerome H.; Olshen, Richard A.; Stone, Charles J. : Classification and Regression Trees. Wadsworth & Brooks, 1984
3. Breiman, Leo : Random forests. Machine Learning 45, 2001
4. Orero, Joseph Onderi; Levillain, Florent; Damez-Fontaine, Marc; Rifqi, Maria; Bouchon-Meunier, Bernadette : Assessing Gameplay Emotions from Physiological Signals. International Conference on Kansei Engineering and Emotion Reasearch 2010
5. Tong W., Hong H., Fang H., Xie Q., Perkins R.: Decision forest: combining the predictions of multiple independent decision tree models. Journal of chemical information and computer sciences 43, 2003
6. Gerdes, Mike; Scholz, Dieter: Feature Extraction and Sensor Optimization for Condition Monitoring of Recirculation Fans and Filters. In: DGLR: *Deutscher Luft- und Raumfahrtkongress 2009 : Tagungsband - Ausgewhlte Manuskripte* (DLRK, Aachen, 01.-04. September 2009). - ISBN: 978-3-932182-63-4
7. Gerdes, Mike; Scholz, Dieter : Parameter Optimization of Automated Signal Analysis for Condition Monitoring of Aircraft Systems. In: ESTORF, Otto von; THIELECKE, Frank (Hrsg.): 3rd International Workshop on Aircraft System Technologies, AST 2011 (TUHH, Hamburg, 31. Mrz - 01. April 2011). Aachen : Shaker, 2011
8. Russel, Stuart; Norvig, Peter: *Artificial Intelligence : A Modern Approach.* New Jersey : Pearson Education, 2003
9. Quinlan, J. R.: *Induction of Decision Trees.* Springer Netherlands, 1986.
10. Quinlan, J. R.: *Programs for Machine Learning.* Morgan Kaufmann, 1993.