Journal of Nondestructive Evaluation (2019) 38:37
https://doi.org/10.1007/s10921-019-0574-9

Performance Based Modifications of Random Forest to Perform Automated Defect Detection for Fluorescent Penetrant Inspection

N. J. Shipway[1] · P. Huthwaite[1] · M. J. S. Lowe[1] · T. J. Barden[2]

Received: 19 November 2018 / Accepted: 4 March 2019 / Published online: 16 March 2019

Abstract
The established Machine Learning algorithm Random Forest (RF) has previously been shown to be effective at performing automated defect detection for test pieces which have been processed using fluorescent penetrant inspection (FPI). The work presented here investigates three methods (two previously proposed in other fields, one novel method) of modifying the FPI RF based on the individual performance of decision trees within the RF. Evaluating based on the F_2 Score, which is the harmonic mean of precision and recall which places a larger weighting on recall, it is possible to reduce the RF in size by up to 50%, improving speed and memory requirements, whilst still gain equivalent results to a full RF. Introducing a performance based weighting or retraining decision trees which fall below a certain performance level however, offers no improvement on results for the increased computation time required to implement.

Keywords Random Forest · Machine Learning · Dye penetrant · Fluorescent penetrant inspection · Automation

1 Introduction

Non-destructive testing (NDT) performs a vital role in inspecting the quality and integrity of materials and components for many industries, but especially for the safety critical aerospace industry. Developing reliable and repeatable testing methods for different applications is a challenge regularly faced by NDT engineers in many industries. There are a number of NDT methods which are used to detect both surface and subsurface cracks and defects, including ultrasonic, eddy current and radiography. The most widely used NDT method for detecting surface breaking cracks or defects in aerospace components is fluorescent penetrant inspection (FPI) which is used to inspect over 90% of metallic components at least once during manufacture [1,2].

Visual inspection of FPI processed parts can suffer the effects of human factors which may lead to variable results. Previous work has investigated the use of the established Machine Learning method Random Forest (RF) to perform automated defect detection for FPI. Results of this work found that RF was able to correctly identify 76% of defects with a false call rate of 0.42, demonstrating capability comparable to that of a human operator [3]. Whilst this work showed promising results, in order for this method to be successfully implemented in industry it will be necessary to improve the results to provide a higher true positive rate and lower false call rate.

This paper presents three modifications which were developed in an attempt to improve the effectiveness of RF at performing automated defect detection. Each of the three methods evaluates the performance of individual trees within the RF and based on this performance the RF is modified.

(1) **Reduced RF**: the first method removes poor performing trees from the RF, creating a smaller RF.
(2) **Weighted RF**: the second method introduces a performance based weighting to each of the trees in the RF such that those which have performed well on a validation set contribute more strongly to the final classification than those which have performed poorly.
(3) **Retrained RF**: the final method removes poor performing trees as in Reduced RF however a new tree is trained to replace it. The performance of this newly trained tree is tested. Should the newly trained tree perform above

✉ N. J. Shipway
n.shipway15@imperial.ac.uk

[1] Department of Mechanical Engineering, Imperial College London, Exhibition Road, London SW7 2AZ, UK

[2] NDE Laboratory, Rolls-Royce plc, Bristol BS34 7QE, UK

the performance threshold it is retained and if not, the retraining process is repeated until there is a full RF of well performing trees.

The remainder of this paper is laid out as follows; Sect. 2 gives the reader an introduction to FPI (Sect. 2.1), Machine Learning specifically RF (Sect. 2.2) and an overview of the proceeding work, using RF for FPI [3] (Sect. 2.3). A review of literature related to modifications of the RF method is then presented in Sect. 2.4. Section 3 outlines the three methods used before presenting results in Sect. 4 and drawing conclusions in Sect. 5.

2 Background

2.1 Fluorescent Penetrant Inspection

FPI is one of the oldest and simplest non-destructive testing (NDT) methods and yet still one of the most widely used in industry. The method works by applying a fluorescent dye to the surface of a clean component. During an allocated dwell time, this dye penetrates any surface breaking defects via capillary action. After the dwell time has elapsed, excess penetrant is washed off. The component is then dried, initially using compressed air and then in an oven. Developer powder applied to the surface encourages penetrant contained within defects to seep out, making them visible when the part is inspected under UV light.

FPI has the ability to easily and cheaply inspect large complex geometries and this has led to its prominence within industry. There are three main circumstances under which FPI takes place in the aerospace industry; new manufacture, repair and overhaul, and in-situ.

During new manufacture, FPI is predominantly used as a process control check. Because of this, components can be inspected numerous times at subsequent stages of manufacture, with a very low defect occurrence rate. Automated processing of parts has been common in industry for a number of years. Parts are loaded onto a carousel which then automatically cleans the components and then applies and subsequently removes the dye and developer, meaning parts exit the carousel ready for manual inspection.

The nature of new manufacture inspection of FPI processed parts means identical, newly manufactured components are inspected in low light conditions, with a very low defect occurrence rate. Work by Wall explains that human factors in NDT can refer to environmental, organisational and job factors, and human and individual characteristics, which influence the effectiveness, performance and reliability achieved in NDT inspections [4]. Considering all this, it is perhaps unsurprising that human factors contribute significantly in the visual inspection of FPI for new manufacture. However, these attributes cause FPI of new manufacture to be a good first application for automated inspection.

The two other inspection areas for FPI are in-situ and repair and overhaul. In-situ inspections are reactive rather than part of planned maintenance and therefore are bespoke to a specific situation. Repair and overhaul on the other hand, involves stripping an engine and inspecting each individual component. There is therefore a large component variety and a higher probability of components containing defects. Engine run components also typically cause much higher background noise levels as the components are dirtier and may contain surface scratches. This means distinguishing between penetrant associated with defects and excess penetrant from other sources can be more difficult. For these reasons automated FPI is unlikely to be initially introduced to in-situ and repair and overhaul inspections.

2.2 Machine Learning

Machine Learning, a branch of artificial intelligence, is the process of using a computer to design a system which is capable of learning from data in a manner of being trained. Generally systems learn and improve with experience, and with time, the model is refined so that is can be used to predict outcomes based on previous learning [5,6]. Machine Learning algorithms fall into one of two categories: supervised or unsupervised. Supervised learning consists of labelled training data and unsupervised consists of unlabelled training data [7].

An established supervised learning mechanism is decision trees, or Classification and Regression Trees (CART). They were among the first statistical algorithms to be implemented in electronic form in the latter decade of the 20th century [8]. The building of decision trees begins with a parent node containing all data points. This data is then split to produce two child nodes with greater homogeneity than the parent node [9]. This process of recursive partitioning continues until subsequent child nodes consist of only data points of one class. These child nodes are then known as leaf nodes. When all branches terminate in leaf nodes the decision tree is fully grown.

Despite their simplicity, decision trees represent a powerful technique which allows for complex non-linearity without having to specify the structure in advance [10,11]. They also offer exceptionally interpretable results which enhances understanding and dissemination of results [8]. However, their simplicity comes at a cost as large trees suffer from over-fitting whilst smaller trees may not be able to capture the main structure of the data [12]. These limitations of decision tree methods can be overcome by the use of ensemble techniques.

One such ensemble technique is Random Forest (RF). RF was developed by Leo Breiman in 2001 [13]. RF combines

predictions made by decision trees using a majority vote decision rule. Bootstrap sampling, a method of random sampling with replacement, is used to create a subset on which each tree is grown. A number of features are then randomly selected at each node which are considered for splitting [11,14,15]. The injection of randomness makes them accurate classifiers. The ability of an RF to predict has been found to be determined by the strength of the individual predictors and also the correlation between them (with low correlation producing the strongest performance) [13]. Random Forest has been found to be effective as it has accuracy as good, if not better than the Adaboost method [13], a similar decision tree technique without random elements. It is robust to outliers and noise and it is faster than other decision tree methods such as bagging or boosting [13]; the Law of Large Numbers also means over-fitting does not tend to occur for Random Forest [13]. Other advantages of the Random Forest method are that is it easy to follow the logic taken [16], easy to train [16,17], fast to train [18] and it has been demonstrated to be effective even with a small number of training examples (around 500 training examples were used by Chan et al. to train around 1–700 trees with 1–10 input features [17]). These attributes mean the method is well suited to the problem of automated defect detection.

2.3 Random Forest for Fluorescent Penetrant Inspection

Previous work considered the use of Random Forest (RF) to perform automated inspection of a number of test pieces which had been processed using FPI [3]. A brightness threshold was first performed on the images to obtain regions of interest (ROIs) which formed the training data set. 21 features were extracted from the ROIs which were used to train 100 decision trees. At each node, one of four possible data split methods were selected: decision stump, 2D linear decision learner, conic section or radial basis functions.

The paper showed good capability of the RF method to be able to perform automated defect detection, with the optimum RF being able to detect 76% of the defects within the test set with a false call rate of 0.42. This paper aims to further this work to investigate modifications to the RF to see if performance can be improved.

2.4 Random Forest Modifications

Here, we review what work has been done previously with regard to modifications to the RF method.

2.4.1 Weighted Random Forest

A number of papers have investigated the introduction of weightings to the RF method. These mainly seem to fall into one of two categories: weighted training data or weighted voting mechanisms.

Weighted Training Data

Schulter et al. [19] propose the Alternating Decision Forest (ADF). During training, globally tracked weight distributions are assigned to each training sample. These weights are iteratively updated, i.e. becoming higher for hard-to-classify samples and lower for easy ones. They also differ from traditional RFs as they are trained breadth-first rather than depth-first. Of the five data sets tried, results showed the ADF to perform better compared to the standard RF.

Robnik–Sikonja noted that not all trees are equally successful in labelling instances. Internal estimates were therefore used to identify instances most similar to the ones which are being classified. The votes of the trees were then weighted with the strength they demonstrate of those near instances. (More details given in Sect. 4 of [20]). Results demonstrated a minor improvement when using weighted compared to traditional RF.

Chen considered the problem of class imbalance [21]. Using a largely imbalanced data-set can cause the RF classifier to become more biased towards the majority class. In order to overcome this, a weight is applied to each class with the minority class given a higher misclassification cost. Results demonstrated superior performance compared to other class-imbalance techniques. However it can be vulnerable to noise (e.g. misclassified training data).

Weighted Voting Mechanisms

A number of papers have also investigated the use of weighted voting mechanisms to improve performance. Gunter et al. considered the problem of handwritten word recognition. Rather than applying a performance based weighting, a more general approach was used where the weights were considered as parameters to be selected such that the overall performance of the combined system is optimised. A genetic algorithm is used to actually determine an optimal combination of weight values. Comparing performance based weighting and genetic algorithms, it was found that the highest recognition rate was obtained with genetic weight optimisation [22].

Another modification into the voting mechanism performed by RF was into the use of dynamic integration to combine the predictions from each of the individual classifiers. Traditional RF sums the votes to come up with an overall prediction. Using dynamic integration based on local performance estimates was shown to improve the accuracy in 12 out of 27 datasets [23].

2.4.2 Reduced Random Forest

Bernard et al. posed the question of whether there are any decision trees in an RF which deteriorate the performance of an ensemble and if so, is it possible to form a more

accurate committee via removal of decision trees with poor performance. Two methods were used: Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS). For SFS, at each iteration, each remaining classifier is added to the current subset and the one which optimises the performance is retained. Similarly, for SBS each classifier of the current subset is removed and the one for which the remaining ensemble exhibits the best accuracy is discarded. Both these methods were tested on 10 datasets and it was found there always exists a subset of well selected trees able to outperform an ensemble grown with classical RF. Moreover, the best sub-forests were found to contain significantly fewer trees than the initial RF [24].

2.4.3 Other Random Forest Modifications

A number of other modification methods have been investigated including modifications to feature selection [14,22,25], sampling methods [19,21,24], node splitting [15,20], performance evaluation [20], considering similarity between training examples [20,23,26] and others, details of which can be found in [15].

3 Experimental Method

As outlined in Sect. 1, this work builds on that done by Shipway et al. in using the RF method to perform automated defect detection for FPI. This paper investigates the use of performance based modifications to improve the current method, investigating whether there are individual trees within the RF which consistently performed highly and whether there were some which consistently classified incorrectly. Using this information, the three theories were posed. The first investigation was whether the poor performing trees could be removed from the RF altogether without effecting the overall performance, hence creating a leaner RF which is computationally more efficient and yet is able to make classifications to the same standard as the full RF. The second theory was whether performance based weightings could be introduced to the classifications. Trees which consistently perform well in classification would contribute more strongly to the final decision, and on the contrary, trees which consistently perform poorly would contribute less. The final theory which is investigated in this paper is to retrain trees until they perform above a certain level.

In Sect. 3.1 an overview of the work done in [3] shall be presented. All three modification methods are based on the performance of individual trees within the RF so Sect. 3.2 presents the performance evaluation methods used before going on to present the three modification methods in Sects. 3.3, 3.4 and 3.5.

3.1 Data and Original Training Method

The dataset for this work and the preceding work was produced from a set of rectangular titanium alloy plates (100 mm × 50 mm × 4 mm) each with a number of cracks of known size artificially induced at random locations using thermal fatigue loading. 36 plates were used with 124 cracks ranging from 0.1 mm to 3.4 mm in length.

The parts were processed using ultra high sensitivity penetrant (1D4 [27]) and photographed using a Canon EOS 6D DSLR camera with an EF 100mm f/2.8 USM macro lens. Each image obtained was annotated manually using a graphics editor to indicate which indications are defects. This creates a mask image which provides ground truth information to the defect detection software. Each pixel is a 16-bit integer and therefore has brightness up to 65,535. The first step in the method performs a brightness threshold on the images, extracting ROIs around any pixel with a brightness level greater than 1000. This focuses the software to only regions containing penetrant. 10,733 ROIs were extracted from the 36 training images and 4617 were extracted from the 18 test images.

Next, a number of features were extracted from each of the ROIs and these features were subsequently used to train the RF. A total of 21 features were extracted including area, average and standard deviation of pixels within an ROI of brightness greater than 100, peak brightness and Haar like features. A number of these features were extracted from greyscale, red, green and blue levels.

These features were then used to train the RF. An RF is grown by selecting a random subset of the total dataset and training a decision tree. Another random subset is then selected with replacement and another decision tree is grown. This process is continued until all trees within the forest have been grown.

To use the RF to perform automated defect detection, the brightness threshold is performed to extract ROIs and the same features are extracted from these as was done for the training data. These features are then passed down each tree within the forest so each tree votes for whether it believes a particular ROI to be defective or not. For an RF of 100 trees, if 25 trees voted that an ROI was defective and 75 trees voted it to be defect free the system would output that it was 75% confident the region is defect free.

This section has provided an overview of the methodology more thoroughly laid out in Sect. 3 of [3].

3.2 Performance Evaluation Methods

This paper aims to investigate the use of performance evaluation methods to improve the overall performance of the RF method in performing automated defect detection for FPI.

Six performance metrics were used and these shall be presented below.

3.2.1 Precision

Precision, or confidence as it is commonly known [28], can be seen as a way to analyse the cost performance of an algorithm. Precision, in the case of automated defect recognition, is considering the total number of regions identified by the software as being defective and determining the proportion of those which correspond to an actual defect. Should the precision be low, the algorithm will be incorrectly identifying false indications as being defective which could lead to a high scrap rate or a large number of components which need to go forward for human inspection.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{1}$$

3.2.2 Recall

Recall, or sensitivity, on the other hand, is a way to analyse the safety performance of an algorithm. Considering the total number of defects within the dataset, recall denotes how many were correctly identified by the software. Should the recall be low, a large proportion of the defects are being missed. This could lead to defective components continuing on to the next stage of manufacture or in the worst case, onto engine build. From this, it can be seen that for applications such as the aerospace or medical industry, recall is of the utmost importance. However, for other industries such as in the case of food packaging or other less safety critical environments, precision may be prioritised.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{2}$$

3.2.3 F_1 Score

Often an effective solution is one which balances both precision and recall. The F_1 score aims to perform this by calculating the harmonic mean between precision and recall.

$$F_1\ Score = \frac{2(Precision)(Recall)}{Precision + Recall} = \frac{2(True\ Positives)}{2(True\ Positives) + False\ Positives + False\ Negatives} \tag{3}$$

3.2.4 F_2 Score

The F_2 score is a development on the aforementioned F_1 score. It calculates the harmonic mean of precision and recall however places a larger weighting on recall, i.e. safety.

$$F_2\ Score = \frac{5(Precision)(Recall)}{5(Precision) + Recall} = \frac{5(True\ Positives)}{5(True\ Positives) + False\ Positives + 4(False\ Negatives)} \tag{4}$$

3.2.5 Overall Accuracy

The overall accuracy is perhaps the simplest of all metrics considered. It calculated the proportion of instances which were correctly classified.

$$Overall\ Accuracy = \frac{True\ Positives + True\ Negatives}{No.\ of\ image\ patches} \tag{5}$$

3.2.6 Overall Performance

The final metric considered is overall performance. The five metrics described above all have advantages as they each draw out particular aspects of a well performing algorithm. The author wanted to introduce a metric which combines these five metrics in order to attempt to produce a more versatile performance metric. This overall performance metric was therefore one developed by the author which simply sums the five metrics presented above.

$$Overall\ Performance = Precision + Recall + F_1\ Score + F_2\ Score + Overall\ Accuracy \tag{6}$$

3.3 Reduced Random Forest

The first of the three RF modifications is the reduced RF. This aims to investigate whether there are some trees which can be removed without affecting, or perhaps increasing, the performance of the RF. The complete dataset is split into four groups, A, B, C and D. Of these four subsets, two are used to train the original Random Forest, one is used to analyse the performance of the RF and based on the results of this some trees are removed. The final subset is used to analyse the overall performance of the reduced RF.

An RF of 100 trees is trained using the 21 features laid out in [3] and the images from data subsets A and B. Subset C is then passed down the RF. Each of the six performance evaluation methods in Sect. 3.2 is then calculated for each of the 100 trees within the RF. These have been sorted according

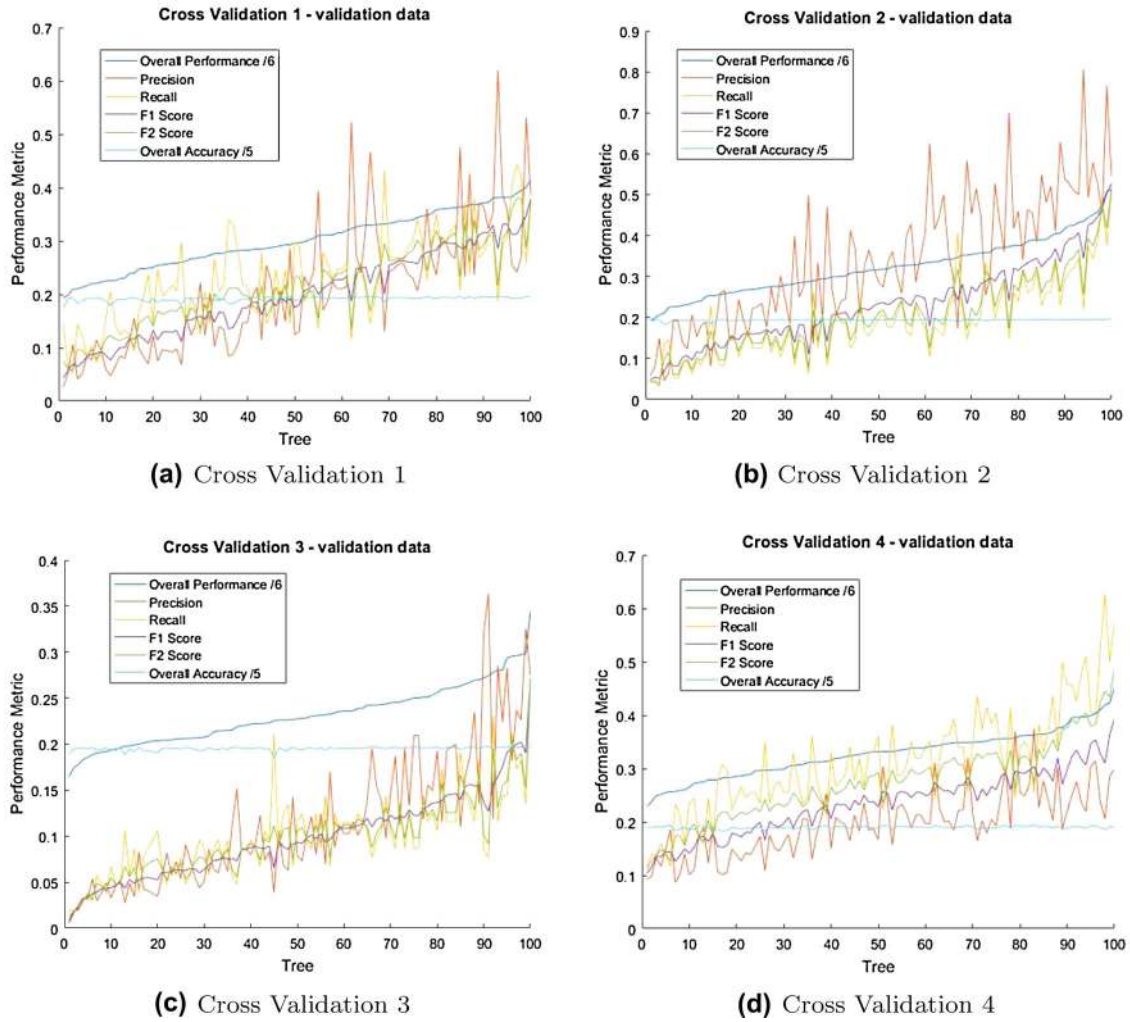


Fig. 1 Graphs to show how each of the six performance evaluation varies for four different RFs

to overall performance and plotted in Fig. 1. Cross Validation has been used to demonstrate the results are not dependent on the input data, allowing the evaluation of the algorithm performance independent to input data. Cross Validation 1 trains on groups A and B, reduces based on group C and tests on group D. Cross Validation 2 trains on groups B and C, reduces based on group D and tests on group A. Cross Validation 3 trains on groups C and D, reduces based on group A and tests on group D. Cross Validation 4 trains on groups D and A, reduces based on group B and tests on group C.

Trials were undertaken varying the threshold value at which trees are removed. Six different values were used and these can be seen in Fig. 2. Ordering the metrics into ascending order, thresholds are selected at 25%, 50% and 75% of each of the X and Y axis. The X axis represents the number of trees, so the threshold value would be the value of e.g. the overall performance of the 25th best performing tree within the forest. The Y axis is the performance metric. So for example we could calculate the maximum overall performance value out of all the trees and then set the threshold value at 25% of this value.

3.4 Weighted Random Forest

The second of the three RF modifications is the weighted RF. As explained in Sect. 3.1, when an RF is being used to perform defect classification, each tree within the forest produces a probability that the region of interest contains a defect. The average of all the probabilities produced by the RF are then averaged in order to find the overall probability of a region being defective. This modification introduces a weighting to each of the individual tree probabilities, with those trees which have performed well contributing more highly than those which have not. The weighting value, W, is calculated as shown in equation 7 where P is the performance metric, n is a variable, and 2.7 is an arbitrary value selected

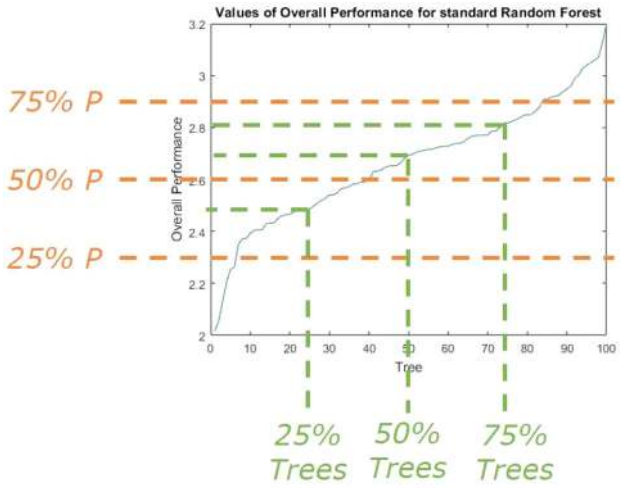


Fig. 2 Threshold Values used to reduce and retrain RFs

as it is approximately 50% of the overall range of Overall Performance values, see Fig. 2.

$$W = \left(\frac{P}{2.7}\right)^n \tag{7}$$

The overall prediction, Y, by an RF is given in equation 8 where t is the number of trees and y_i is the prediction by an individual tree.

$$Y = \frac{\sum_{i=1}^{t}(W\, y_i)}{t} \tag{8}$$

3.5 Retrained Random Forest

The final RF modification was retraining. As in Sect. 3.3, an RF of 100 trees was trained using the images from data subsets A and B. The performance of these 100 trees was then evaluated using the six performance metrics laid out in Sect. 3.2. Any trees which perform under a specified threshold (again, the six threshold values used in Sect. 3.3 are used) are removed, and instead another tree is grown. The performance of this tree is evaluated and should it perform above the threshold it is added to the RF. Should the performance be less than the threshold, the tree is disregarded and another tree is trained. This process is repeated until the RF consists of 100 trees which all perform above the threshold value when tested on the data from subset C. The algorithm is illustrated in Fig. 3.

4 Results and Discussion

The results below have been presented as Free-Response Receiver Operator Characteristic (FROC) curves. The FROC curve shows the true positive rate plotted against the number of false calls per image. It is a modification of the Receiver Operator Characteristic (ROC) curve which can be used when there can be more than one region of interest in an image. For each of the methods, the cross validation laid out in Sect. 3.3 has been performed. Four different results are therefore presented for each case, showing the methods dependency on the data. For each of the figures in this section, a solid line shows the original RF whilst a dashed line shows the modified RF.

4.1 Reduced Random Forest

A subset of the results from using a reduced RF are shown in Fig. 4. Three performance metrics are shown in each column: overall performance, F_2 score and overall accuracy. Each metric is then shown at the three Y-axis threshold values explained in Sect. 3.3, making up 3 columns of Fig. 4. Y-axis threshold values are shown as they offer a wider range of threshold values compared to the X-axis threshold values.

Considering overall performance (Fig. 4a–c), it can be seen that at threshold values of 25% and 50% of the overall range of performance values, a reduced RF produces results equivalent to the full RF for all cross validations. At a threshold value of 75%, performance drops off for CV2 and CV3. As CV1, on the whole, appears to perform worse than the other cross validations, the fact that performance improves for CV1 at a threshold value of 75% shall be treated with caution. The same trends were followed for Recall and F1 score. Precision followed similar results however performance degraded at a threshold value of 50% and above. These results have been omitted for brevity.

The second set of results shown in Fig. 4 are those of F_2 score (Fig. 4d–f). As for overall performance, at threshold values of 25% and 50% the performance is equivalent to that of the full RF. For a 75% reduction in trees, the degradation

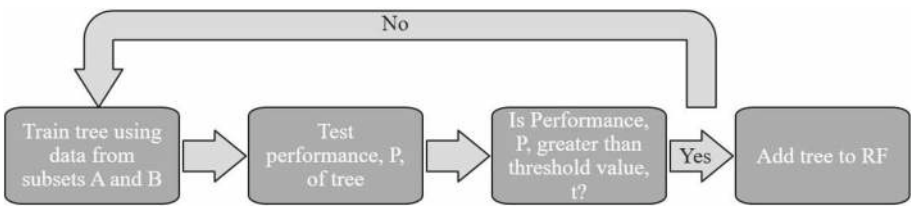
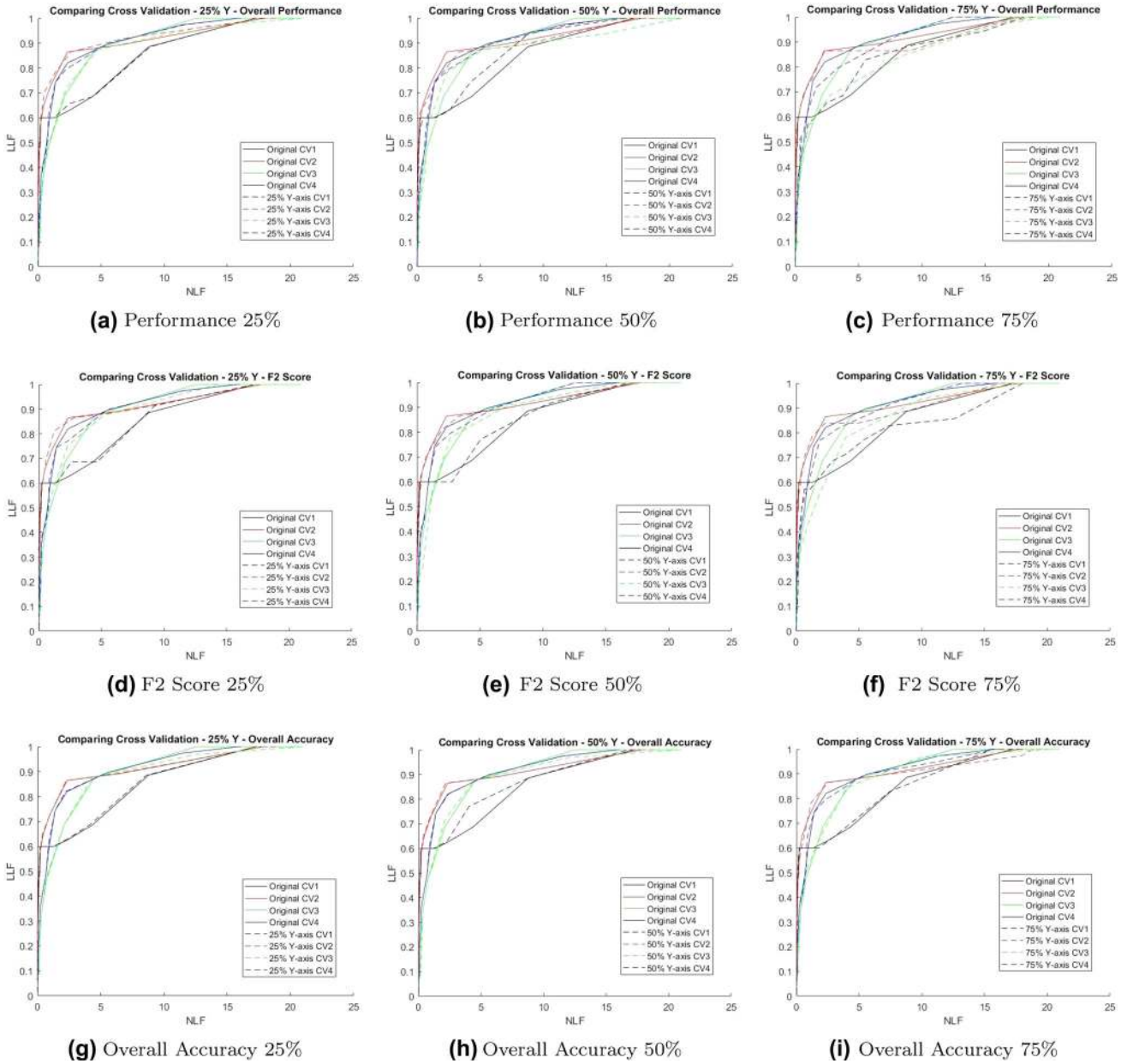


Fig. 3 Algorithm in order to retrain trees for RF



(a) Performance 25% **(b)** Performance 50% **(c)** Performance 75%

(d) F2 Score 25% **(e)** F2 Score 50% **(f)** F2 Score 75%

(g) Overall Accuracy 25% **(h)** Overall Accuracy 50% **(i)** Overall Accuracy 75%

Fig. 4 Reduced RF: the results of reducing an RF based on three performance metrics for three different threshold values. Each graph shows the results of four different datasets, arranged using cross validation

in performance is smaller than that for overall performance, precision, recall and F_1 score.

The final set of results shown are those for overall accuracy (Fig. 4g–i). Of all six performance metrics, those produced using overall accuracy offer the best performance, with reduction at all three thresholds offering performance equivalent to that of the full RF. However, overall accuracy must be treated with caution when being used in a situation in which there is a large class imbalance. For example, if the training data consists of 95% non-defective ROIs and 5% defective ROIs, having a classifier which always classifies an ROI as being defective would still produce an accuracy of 95%.

The distinction between the performance metrics can be seen most clearly in the case of a 75% threshold, see Fig. 5. It can be seen that F_2 score and overall accuracy consistently perform well (excluding CV1) and therefore F_2 score shall be selected as the most appropriate performance metric.

Considering the points above, it can be said that using performance evaluation, an RF can be reduced by as much as 50% whilst still offering results equivalent to that of a full RF.

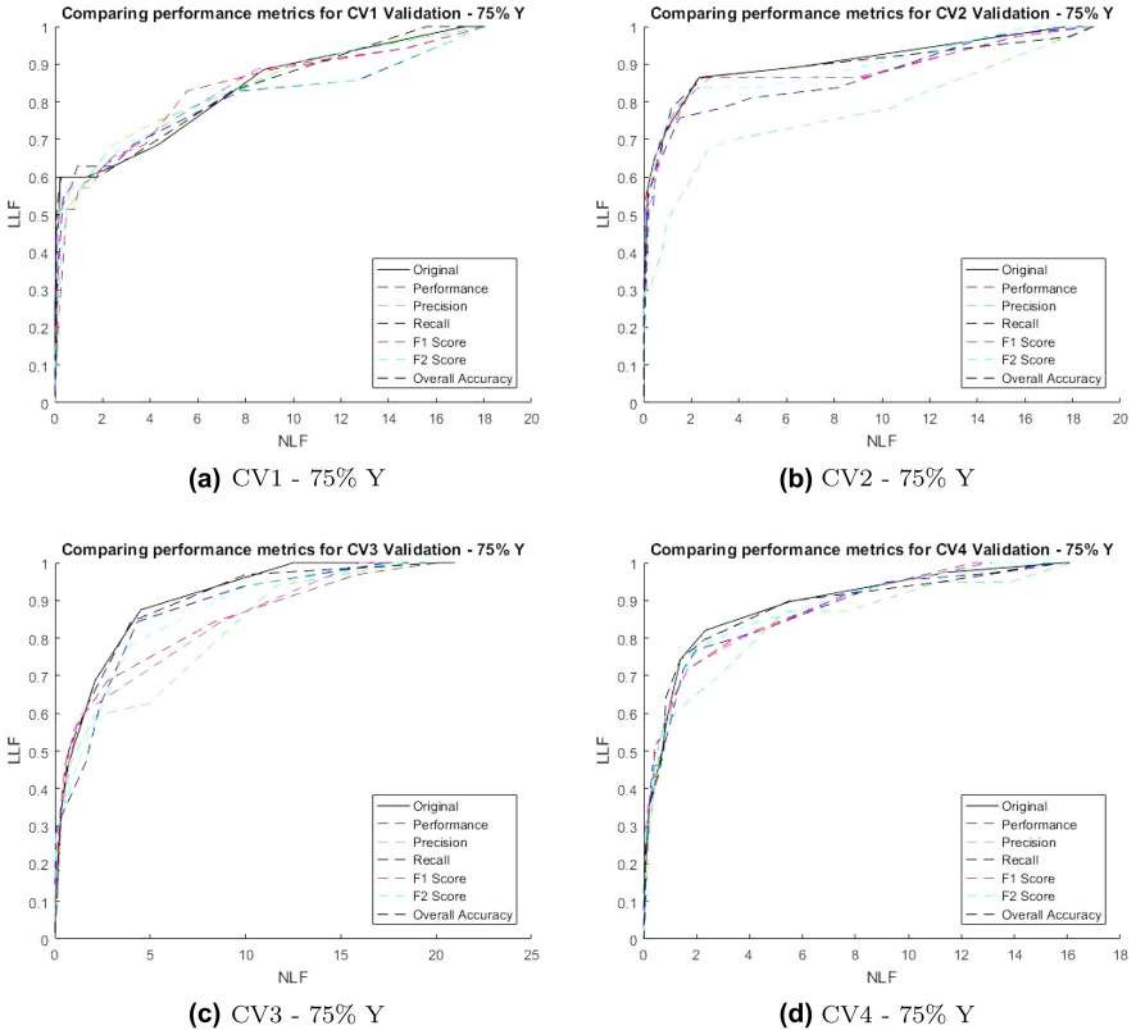


Fig. 5 Reduced RF: the results of reducing an RF based on one threshold value. This figure allows the comparison of the different performance metrics for each of the four cross-validation datasets

4.2 Weighted Random Forest

Results of a weighted RF are shown in Fig. 6. It can be seen that for all but CV1, using a weighted RF appears to perform worse than the original RF. As discussed in Sect. 4.1, CV1 appears to perform differently compared to the other cross validation datasets and therefore it shall be concluded that using a performance based weighted RF performs worse than an unmodified RF.

4.3 Retrained Random Forest

Results of the retrained RF thresholding on the F_2 score are shown in Fig. 7. As for Sect. 4.1, the three Y-axis threshold values explained in Sect. 3.3 are presented.

The trends followed in the case of the F_2 score are equivalent to those shown for the other performance metrics and therefore only those produced using F_2 score have been shown for brevity. It can be seen in Fig. 7 that by retraining trees within the RF, no improvement in performance is observed. As the retraining process is computationally expensive, it therefore offers no advantage to use retraining.

5 Conclusions

This paper has investigated three methods of introducing performance based evaluation of individual decision trees within an RF in order to boost performance. Whilst two of these methods were found to offer no improvement, one method offered the ability to have a computationally leaner RF whilst still producing results of the same standard.

Evaluating the performance of trees using the F_2 score (a harmonic mean of precision and recall which places a larger weighting on safety) and disregarding the worst performing 50% is able to produce equivalent results than the entire RF.

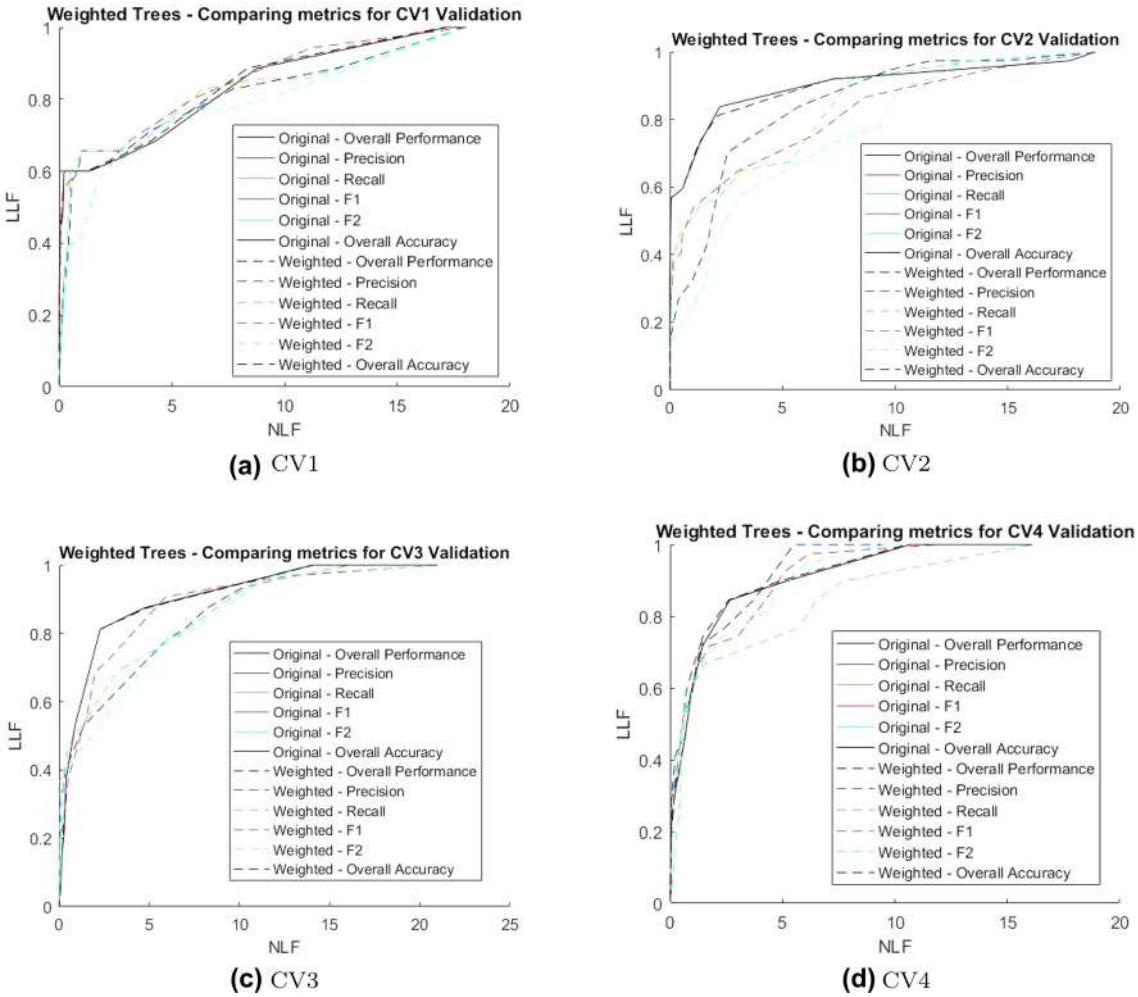


Fig. 6 Weighted RF: the results of weighting an RF based. Results shown for each of the four cross-validation datasets

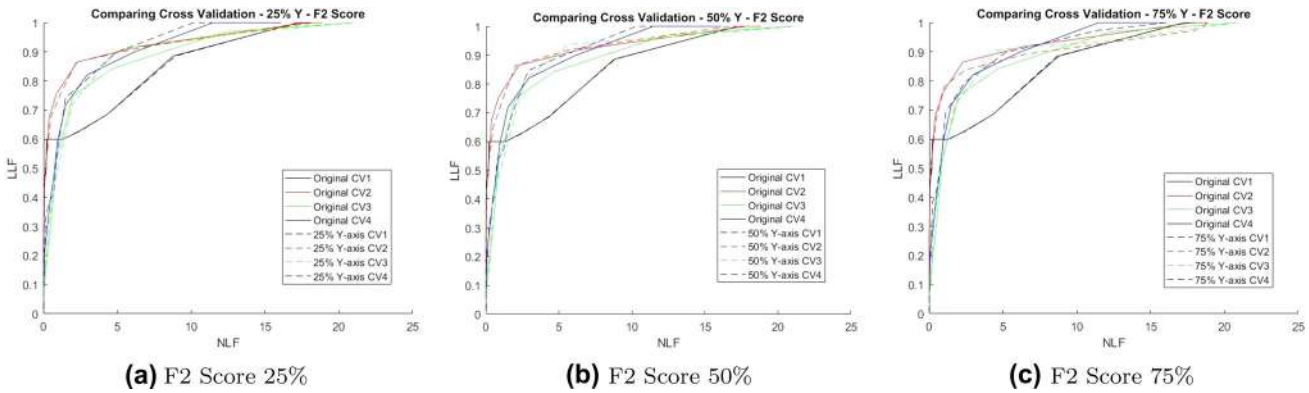


Fig. 7 Retrained RF: the results of retraining an RF based on one performance metric, F2 Score, for three different threshold values. Each graph shows the results of four different datasets, arranged using cross validation

This demonstrates the potential to have a faster classification algorithm for detecting defects in FPI processed parts.

Introducing a performance based weighting to each tree such that well performing trees contribute more significantly to the final classification than poor performing trees offers no improvement for the increase in computation. Similarly, retraining poor performing trees until they perform above a certain level also offers no improvement for the increased computational requirements.

Acknowledgements This work was supported by the Engineering & Physical Sciences Research Council Grant Nos. EP/L015587/1, EP/M020207/1 and by Rolls-Royce plc.

References

1. Brasche, L., Lopez, R., Larson, B.: A study of drying and cleaning methods used in preparation for fluorescent penetrant inspection. AIP Conf. Proc. **657**, 1323–1330 (2003)
2. Technik, A.G.L., Engine Overhauls: The Search for Material Damage. https://www.lufthansa-technik.com/engine-overhauls
3. Shipway, N., Barden, T., Huthwaite, P., Lowe, M.: Automated defect detection for fluorescent penetrant inspection using random forest. NDT & E International, p. NDT & E International. Elsevier, New York (2018)
4. Wall, M.: Human factors guidance to improve reliability of non-destructive testing in the offshore oil and gas industry. In: 7th European-American Workshop on Reliability of NDE, 2017. https://www.esrtechnology.com/index.php/news/latest-news/44-news/archived-news-articles/239-hois-project-presented-at-prestigious-reliability-workshop
5. Bell, J.: Machine learning. Wiley (2014). https://doi.org/10.1002/9781119183464
6. Mitchell, T.: Machine Learning, International Edition. McGraw-Hill series in Computer Science. McGraw-Hill, New York (1997)
7. Zhao, Z., Liu, H.: Spectral feature selection for supervised and unsupervised learning. In: Proceedings of the 24th International Conference on Machine Learning (2007)
8. De Ville, B.: Decision trees. Wiley Interdiscip. Rev. **5**(6), 448–455 (2013)
9. Speybroeck, N.: Classification and regression trees. Int. J. Public Health **57**, 243–246 (2012)
10. Ledolter, J.: Decision trees (Chap. 13). Data Mining and Business Analytics with R. Wiley, New York (2013)
11. Webb, A.R., Copsey, K.D.: Ensemble methods. Statistical Pattern Recognition, pp. 361–403. Wiley, Chichester (2011)
12. Hassan, M.A., Khalil, A., Kaseb, S., Kassem, M.A.: Exploring the potential of tree-based ensemble methods in solar radiation modeling. Applied Energy, vol. 203. Elsevier, New York (2017)
13. Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (2001)
14. Schroff, F., Criminisi, A., Zisserman, A.: Object class segmentation using random forests, In: Proceedings of the British Machine Vision Conference, BMVA Press, (2008), pp. 54.1–54.10, https://doi.org/10.5244/C.22.54.
15. Tripoliti, E.E., Fotiadis, D.I., Manis, G.: Modifications of the construction and voting mechanisms of the random forests algorithm. Data Knowl. Eng. **87**, 41–65 (2013). https://doi.org/10.1016/j.datak.2013.07.002
16. Biau, G., Scornet, E.: A random forest guided tour. TEST **25**(2), 197–227 (2016)
17. Chan, J.C.W., Beckers, P., Spanhove, T., Borre, J.V.: An evaluation of ensemble classifiers for mapping Natura 2000 Heathland in Belgium using spaceborne angular hyperspectral (CHRIS/Proba) imagery. Int. J. Appl. Earth Observ. Geoinform. **18**, 13–22 (2012)
18. Belgiu, M., Dragut, L.: Random forest in remote sensing: a review of applications and future directions. ISPRS J. Photogramm. Remote. Sens. **114**, 24–31 (2016)
19. Schulter, S., Wohlhart, P., Leistner, C., Saffari, A., Roth, P. M., Bischof, H.: Alternating decision forests. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition, (2013), pp. 508–515. https://doi.org/10.1109/CVPR.2013.72
20. Robnik-Å ikonja, M.: Improving random forests. In: Boulicaut, J., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) Machine Learning: ECML. Lecture Notes in Computer Science, vol. 3201, pp. 359–370. (2004). https://doi.org/10.1007/978-3-540-30115-8_34
21. Chen, C., Breiman, L.: Using random forest to learn imbalanced data. University of California, Berkeley
22. Guenter, S., Bunke, H.: Optimization of weights in a multiple classifier handwritten word recognition system using a genetic algorithm. Electron. Lett. Comput. Vis. Image Anal. **3**(1):25–44 (2004). https://elcvia.cvc.uab.es/article/view/v3-n1-guenter-bunke
23. Tsymbal, A., Pechenizkiy, M., Cunningham, P.: Dynamic integration with random forests. In: FÃŒrnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) Machine Learning: ECML. Lecture Notes in Computer Science, vol. 4212. Springer, Berlin (2006)
24. Bernard, S., Heutte, L., Adam, S.: On the selection of decision trees in random forests. In: 2009 International Joint Conference on Neural Networks (2009), pp. 302–307. https://doi.org/10.1109/IJCNN.2009.5178693
25. Rodriguez, J.J., Kuncheva, L.I., Alonso, C.J.: Rotation forest: a new classifier ensemble method. IEEE Trans Pattern Anal Mach Intell **28**(10), 1619–1630 (2006). https://doi.org/10.1109/TPAMI.2006.211
26. Cunningham, P.: A taxonomy of similarity mechanisms for case-based reasoning. IEEE Trans Knowl Data Eng **21**(11), 1532–1543 (2009). https://doi.org/10.1109/TKDE.2008.227
27. AMS K Non Destructive Methods and Processes Committee (2013) Inspection Material, Penetrant AMS2644F. SAE International. https://doi.org/10.4271/AMS2644F
28. Powers, D.: Evaluation: from precision, recall and f-measure to ROC, informedness, markedness & correlation. J Mach Learn Technol **2**, 37–63 (2011)