

Using Meta-Features to Boost the Performance of Classifier Fusion Schemes for Time Series Data

Neil H. W. Eklund, *Member, IEEE*, and Kai F. Goebel

Abstract—Accurate detection of faults and prediction of equipment remaining useful life result in considerable economic benefit to industry due to avoidance of unscheduled downtime and costly secondary damage. This paper described an approach to improving the performance of fault detection schemes that operate on time series data by fusing the results of an ensemble of classifiers that operate on the raw data. A meta-feature is designed to provide an indication of the historical state of other classifiers in the system, and to aid in fusion by allowing the individual classification results to be discounted appropriately. This meta-feature is shown to provide a substantial performance increase for fusion schemes. Data from actual aircraft engine/airframe systems are used to assess the performance of the approach.

I. INTRODUCTION

ABNORMAL condition detection is a critical first step in system prognosis. Abnormal conditions, also known as faults, are the first sign of a potential equipment failure at some future time. The direct cost of equipment failures is unavoidable: ultimately, the component must be replaced. Moreover, there are indirect costs to equipment failure that are in many cases far greater than the cost of the repair. One source of indirect costs is secondary damage – e.g., component failure in the compressor stage of a gas turbine often causes damage to the rear stages. Another indirect cost is unscheduled maintenance. It is often less expensive to replace a faulty component during scheduled maintenance before it has failed completely than to have a component fail in the field and have to shut the whole system down (e.g., power generation). Additionally, guaranteed uptime is sometimes a part of service contracts. Thus, for many systems there is considerable economic motivation to detect faults early and accurately.

Numerous algorithms have been used to classify data into faulty/non faulty categories. Typical steps in fault detection include sensor validation, pre-processing, feature extraction, classification, and perhaps post-processing. To boost detection performance, it is often advantageous to fuse the output from an ensemble of classifiers. Several studies (both theoretical and empirical) suggest that classifier ensembles are more accurate than the individual classifiers in the ensemble [14, 18]. As a result, classifier ensembles have been used for various application domains ranging from optical character recognition [20] to seismic signal

classification [28].

Classifier ensembles have been considered as one of the most active research directions in machine learning [9]. An ensemble of classifiers is a set of classifiers whose decisions or outputs are combined to arrive at a final classification decision. The individual classifiers in the ensemble can be any type of classifiers, although ideally they are “diverse”, i.e., they misclassify different cases [14]. Obviously, little gain in classification accuracy can be expected by fusing the output of classifiers that all make the same mistakes.

Once an ensemble of classifiers has been trained on the sensor data, the researcher is left with the task of fusing the output. There are numerous methods such as Dempster-Shafer [29], model-based approaches [22], fuzzy fusion [19], and statistics-based approaches [26] that address classifier fusion for diagnostic or prognostic systems. Some systems rely heavily on domain expertise to perform the fusion [1, 13]. Where domain expertise is insufficient or impossible to obtain, one has to rely on data driven approaches. Such a scenario is considered here.

The classifier fusion approach used in this paper is to treat the classifier outputs as an intermediate feature space [37], i.e., a transformation of the raw feature space. Because the intermediate feature space is composed entirely of classifier output, all of the features can be transformed such that they are near -1 for normal data and near +1 for abnormal data. Faults precede equipment failures, and so by definition come at the end of a time series of data for a particular piece of equipment. We assume the data to be of the “normal” class at the beginning of the time series, and that the data may or may not be of the “faulty” class later in the time series. Operating in such an intermediate feature space under this minor assumption allows the construction of the disturbance indicator meta-feature (DIMF).

A problem with treating snapshots of time series data as a stand-alone case for classification is that the classifier has no sense of history. As a result, “borderline” cases due to, for example, sensor noise or uncorrected environmental effects, may be misclassified. The disturbance indicator meta-feature is intended to provide an indication of time: while no prior cases have been classified as abnormal, the classification threshold can be made more liberal; after some cases have been classified as abnormal, the classification threshold might be made more conservative.

II. METHOD

A. Application

The results presented in this paper are based on data typically collected from large commercial aircraft. The data includes both continuous parametric sensor measurements (e.g., exhaust gas temperature, fuel flow, engine oil pressure, and engine core speed) and nonparametric information, e.g., Full Authority Digital Engine Control (FADEC) fault messages. The use of the expanded set of data, and in particular the fusion of parametric and non-parametric data is atypical, and an important step forward in the field of aircraft gas turbine engine and airframe systems diagnostics and prognostics.

The primary goal of diagnostics is to identify abnormal conditions of the system as early, accurately and reliably as possible. In this paper, we fuse the output from several diagnostic models that employ features extracted from parametric and non-parametric data [17, 36]. In a forward mode, given the data accessible at each flight, the diagnostic models make an assessment (normal or abnormal) about the condition of the engine.

Real data from 32 engine/airframe systems were used for this study. Details of the raw input values used are provided in [33]. Domain experts were asked to arrive at a consensus decision about time of fault onset. Data after this time was labeled faulty, and data prior to it was labeled normal.

Training and performance assessment of both the base classifiers and the classifier fusion methods is done using the leave-one-out method. In leave-one-out, each set of data is examined in turn, training on the other 31 sets of data and evaluating system performance on the left out set.

B. Performance Evaluation

The data used in this study are strongly unbalanced – 79% “normal”, 21% “faulty”. Fortunately for the flying public, the class distribution of data fleet-wide is even more skewed than in the data studied here. Thus, special care is used to evaluate fusion system performance in light of the unbalanced data.

Overall classification accuracy (or alternatively overall classification error) is by far the most popular measure of classifier performance and is used almost exclusively in the design and evaluation of classifier ensembles. However, classification accuracy is a poor choice for comparing classification systems, for two reasons. First, it assumes that classification errors for all classes have equal cost consequences – which rarely holds for real-world applications. However, exact costs associated with different misclassification are rarely known and can be difficult to estimate *a priori*. Thus, a method of comparing classification systems over a range of misclassification costs is desirable.

The second reason accuracy alone is unsuitable as a metric for comparing classification systems is that it is sensitive to class distribution. The same classification system will have

		Classes assigned by Classifier	
		Negative	Positive
True Classes	Negative	N^{00}	N^{01}
	Positive	N^{10}	N^{11}

Fig. 1. - A confusion matrix for 2-class classification problems. For a particular classification threshold, the number of cases in a particular cell (N^{**}) is tabulated.

different accuracy if the class distribution of the test set is changed. Moreover, it is simple to invert the accuracy relationship between two classifiers simply by changing the class distribution. Consider the case of two trivial classifiers, one that always classifies “class 1” and another that always classifies “class 0” – as the class distribution goes from majority 1 to majority 0, the relative desirability from an accuracy perspective shifts. The class distribution problem is exacerbated in real world data. Often one class is incredibly infrequent (e.g., in-flight shutdowns of aircraft engines) and researchers intentionally bias the training and test data to increase the relative frequency of the minority class.

The Receiver Operating Characteristic (ROC) curve is a method of comparing classifiers that shows performance over the full range of misclassification cost and is totally insensitive to class distribution. The ROC curve, generally defined only for two class problems, is a plot of “true positive” rate against “false alarm” rate. The output of a classifier, for a particular classification threshold, can be summarized in a confusion matrix (Fig. 1). The true positive rate is the ratio of accurately classified positive cases to the total number of positive cases. Using the notation of Fig. 1,

$$TPR = \frac{N^{11}}{N^{11} + N^{10}}. \quad (1)$$

The false positive rate – or false alarm ratio (FAR) – is the ratio of inaccurately classified negative cases to the total number of negative cases,

$$FAR = \frac{N^{01}}{N^{00} + N^{01}}. \quad (2)$$

An ROC is produced by varying the decision threshold of the classifier. Note that, for a given classifier, the accuracy varies with decision threshold and class distribution. In fact, by varying the decision threshold, one can reduce false alarms to any arbitrarily small number through sacrificing true positives, and vice versa. Depending on the primary goal of the diagnostic or prognostic system, one can weight the performance of the model differently and choose the most appropriate fusion scheme based on the ROC curve.

ROC curves represent the behavior of a classifier without regard to class distribution. Hence ROC curves are a more accurate representation of classifier performance. Because one wants to maximize true positives and minimize false alarms, curves that are Pareto dominant in this respect are superior. Note that some classifiers may be dominant in some regions and dominated in others. Although it necessarily loses some of the subtlety of the full ROC, a good summary measure is the area under the curve (AUC). An ideal classifier has an AUC of 1.0; a random classifier has an AUC of 0.5.

C. Base Classifiers

The raw data for the baseline classifiers consisted of a mixture of raw engine sensor values, transformed FADEC messages, and several features constructed from these values.

Twelve classifiers were used to transform the raw data to the intermediate feature space. Linear and nonlinear neural networks (NN), support vector machine (SVM), random forest (RF), and linear regression were all used to build classifiers; details are presented in [33]. Table 1 compares the performance of the classifiers in terms AUC. Seven random forests are shown (RF01... RF07), with “m” indicating the number of variables examined at each split, and “balanced” indicating that the training data were randomly chosen such that the classes were balanced.

D. Disturbance Indicator Meta-Feature

The base features for the intermediate feature space is composed of the raw output from the twelve classifiers. In addition, the disturbance indicator meta-feature was calculated for each classifier and used as twelve additional features. Finally, the mean of the twelve individual DIMFs was used as an additional feature.

To calculate the DIMF, the data are first normalized to the range [-1 1], and then smoothed using the Epanechnikov-Bartlett kernel [34, 35]:

$$\hat{y} = \frac{\sum_{i=1}^N K_{\lambda}(x_0, x_i) y_i}{\sum_{i=1}^N K_{\lambda}(x_0, x_i)} \quad (1)$$

where:

$$K_{\lambda} = \begin{cases} \frac{3}{4} \left(1 - \frac{(x_0 - x_i)^2}{\lambda} \right) & \left| \frac{(x_0 - x_i)}{\lambda} \right| \leq 1 \\ 0 & \left| \frac{(x_0 - x_i)}{\lambda} \right| > 1 \end{cases} \quad (2)$$

with $\lambda = 11$ for these results. A line is made connecting the first and last points of the smoothed data, and the residuals with respect to this line are calculated (the process is illustrated in Fig. 2). The point with the maximum absolute residual (subject to a minimum threshold magnitude) is used

TABLE I
BASE CLASSIFIER PERFORMANCE

classifier	AUC
linear NN	0.875
feedforward NN	0.881
SVM	0.789
RF01 (m=1)	0.853
RF02 (m=1, repeat)	0.860
RF03 (m=2)	0.861
RF04 (m=3)	0.851
RF05 (m=1, balanced)	0.864
RF06 (m=2, balanced)	0.877
RF07 (m=3, balanced)	0.878
linear regression	0.881
bootstrap linear regression	0.878

at the transition point for the DIMF: samples equal to or prior to this point are labeled -1, samples after this point are labeled +1.

Fig. 2 is an example of calculating the DIMF for the output of two different classifiers on the same underlying data. The calculation for one classifier is on the left hand side, and another classifier is on the right. The raw classifier data is plotted in the top row of axes, with class labels (unknown to the classifier). The kernel smooth (thick solid line) and line connecting the beginning and end of the time series (dashed line) is also shown. The second row of axes is the absolute difference (residual) between the smoothed data and the line connecting the beginning and end of the series. The maximum absolute residual is used as the transition point for the DIMF, as shown in the third row of axes. The bottom left set of axes is the mean DIMF across all twelve classification schemes for the same case. Fig. 3 shows the flow of data for calculating the mean DIMF.

Fig. 2 illustrates several interesting features of the DIMF. Some classification schemes will trigger the DIMF transition quite close to the point of fault onset (right hand column), while other classification schemes will trigger the transition early (left hand column), late (not shown), or not at all (not shown). However, in cases where there is a fault, the transitions from low to high values of mean DIMF tend to be well correlated with fault onset (or first fault onset in the case of multiple faults in a time series). When there is no fault, the mean DIMF value rarely gets very high (i.e., few individual classifier systems trigger).

E. Random Forest Meta-Classifier

Random Forest (RF) [5] is a classification method that applies bagging [4] to a variation of classification trees [6]. A standard classification tree is constructed by splitting the data on the best of all possible features at each node. For RF, only a randomly selected subset (chosen always from the full set) of features is eligible to split each node. Moreover, each individual tree is constructed on a bootstrap sample of the data. Finally, in contrast to standard classification trees, the individual RF trees are typically not pruned; rather they are grown to 100% node purity. Although hundreds of trees may be developed, RF's can be trained very quickly (e.g., much faster than neural networks for a given data set and

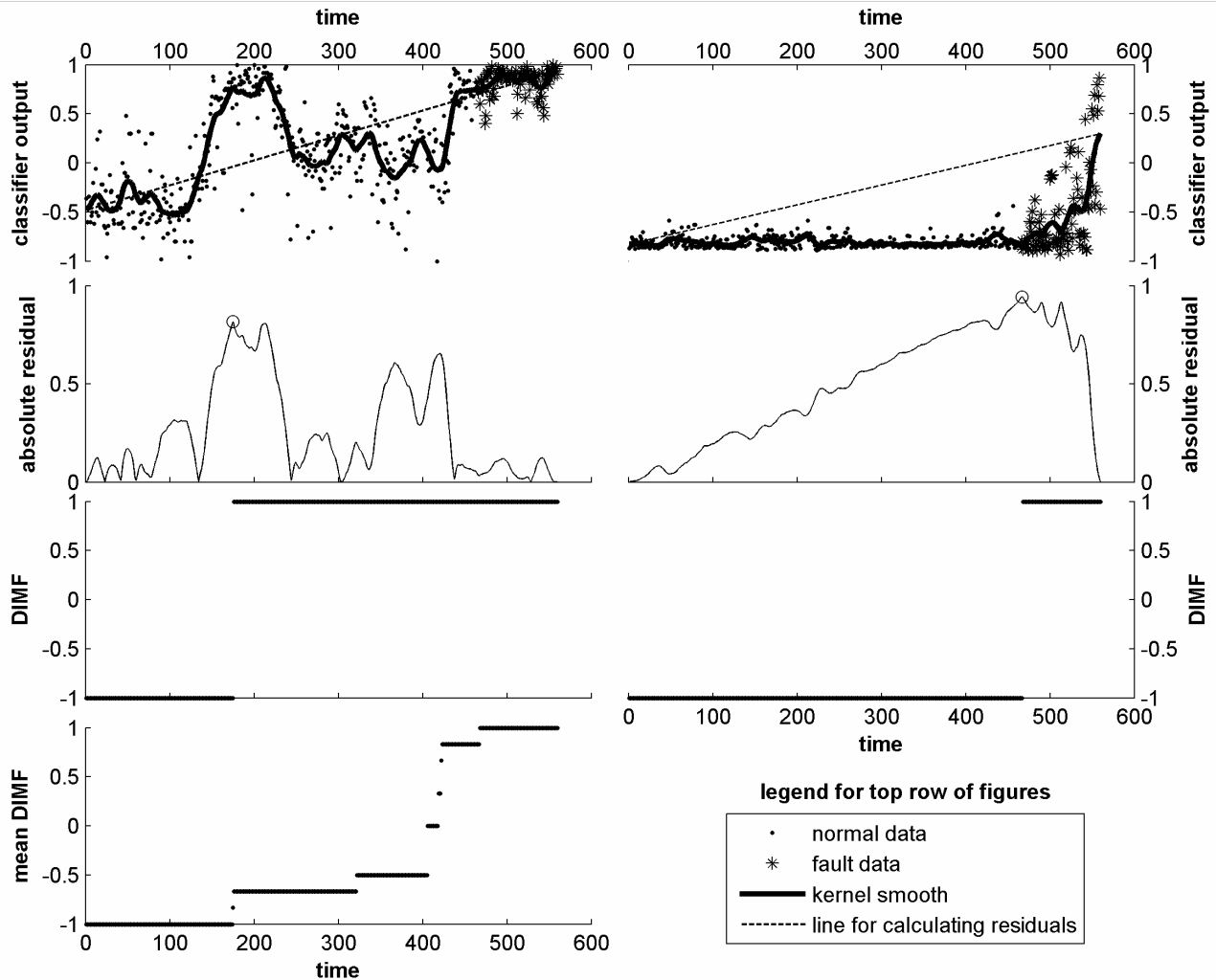


Fig 2. An example of calculating the DIMF for the output of two different classifiers (one in each column) on the same underlying data. Top row: raw classifier data with class labels (unknown to the classifier); the kernel smooth (thick solid line) and line connecting the beginning and end of the time series (dashed line) is also shown. Second row: absolute difference (residual) between the smoothed data and the line connecting the beginning and end of the series. Third row: maximum residual is used as the point for the DIMF. The bottom left set of axes is the mean DIMF across all twelve classification schemes for the same case.

processor). Each tree in the ensemble has an equally weighted vote (e.g., in a 1000 tree RF, 300 trees might vote +1, and 700 might vote -1). The sensitivity of the RF classifier can be varied (e.g., to create a ROC curve) by changing the number of positive (or negative) votes necessary to make a positive (or negative) classification.

Random forests generally exhibit a substantial performance improvement over the single tree classifier such as CART and C4.5. The RF generalization error rate compares favorably to Adaboost, yet RF is more robust to noise. The RF implementation used here was by Breiman et al. [7].

The RF meta-classifier was the best fusion method examined in [33] in terms of AUC, and was the dominant fusion method over a large portion (but not entirely) of the ROC curve. On this basis, the RF meta-classifier was chosen to examine the efficacy of the DIMF.

Four cases of input for RF meta-classifiers were examined. Case R was meant to serve as a baseline, and uses

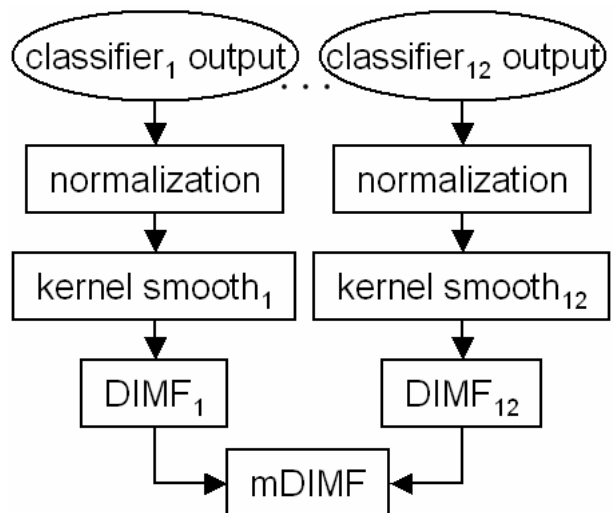


Fig 3. Data flow for calculation of the mean DIMF. Classifier output is first normalized (near -1 for normal data and near +1 for faulty data) and then smoothed for each classifier individually. The DIMF is calculated for each classifier, along with the overall mean, mDIMF.

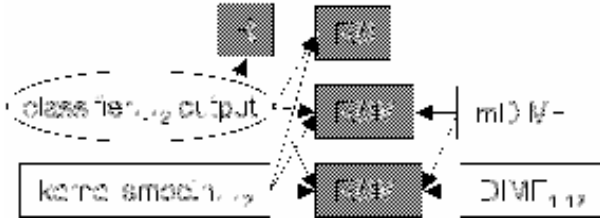


Fig 4. Meta-classifier inputs (white) and corresponding meta-classifiers (hatched) for the four cases of meta-classifier input used.

only the raw output of the 12 classifiers to build a RF (i.e., it is the same meta-classifier as used in [33]). Case RS uses the raw output of the 12 classifiers and the kernel-smoothed output of the 12 classifiers (from construction of the DIMF) to build a RF. Case RS was included to assess the effect of mere smoothing (in time) of the feature space on performance. Case RSM uses the data in Case RS, and the mean DIMF to build a RF. Case RSMI uses the data in Case RSM, and the 12 individual DIMF features to build a RF. The data sources for each classifier are illustrated in Fig. 4.

The number of variables examined at each split was fixed at five for all three meta-classifiers. RFs are not very sensitive to this value [5], and performance did not vary appreciably when it was increased or decreased over a small

TABLE II
META-CLASSIFIER PERFORMANCE

case	AUC
R	0.902
RS	0.899
RSM	0.914
RSMI	0.916

range (beyond this range it in either direction performance decreases). The RF accuracy stabilized above about 300 trees for these data; however, to ensure stability and because the rapid training of the RF permitted) 1000 tree forests were generated in each case.

III. RESULTS

Permutation tests [40] were used to assess the statistical significance of differences in AUC. The AUC for the four meta-classifiers is listed in Table II; Fig. 5 shows the ROC curves of the four meta-classifiers. Also shown in Fig. 5 is the best performing of the single classifiers (from [33]) used as input for the meta-classifiers, a feedforward neural network (NN, Table I). Fig. 5 is divided in to three sections to better visualize certain aspects of classifier performance. The main set of axes (Fig. 5, left) shows the overall ROC;

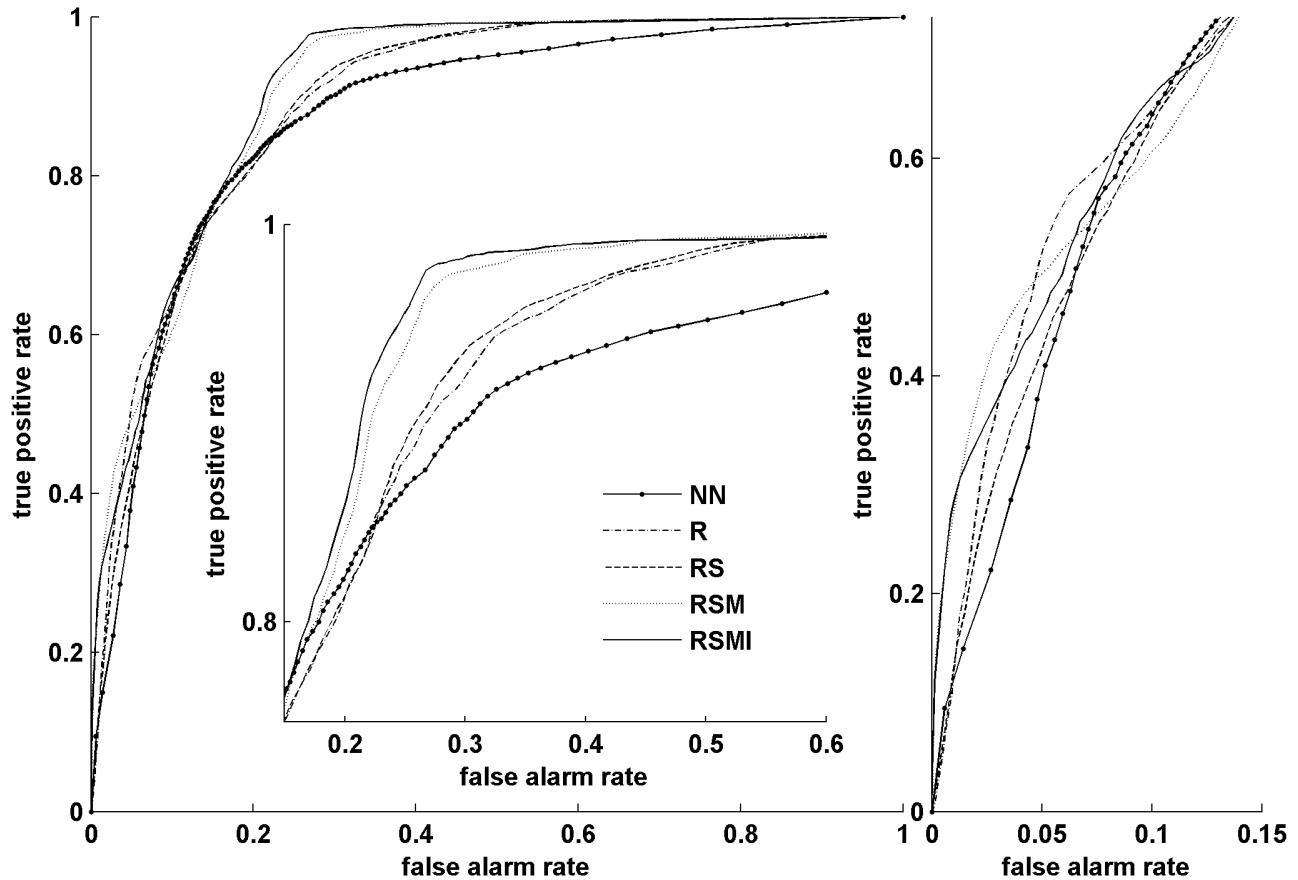


Fig 5. ROC curves for four meta-classifiers plus neural network (NN). Left main axes is full ROC; inset and right axes are details of regions of ROC presented for clarity.

the right hand set of axes provides detail of the low false alarm region of the ROC; the inset axes provides a detailed view of the portion of the ROC where differences between the classification schemes are most pronounced.

Not surprisingly [14, 18, 33], the performance of the single classifier is inferior to any of the meta-classification schemes ($p < .001$). The NN is outperformed at very low false alarm rates (below 0.065) and at false alarm rates above 0.23 (Fig. 5). The performance of all five classification schemes is approximately equal for false alarm rates between 0.09 and 0.18 (and corresponding true positive rates of 0.6 to 0.8).

The R and RS meta-classifiers, which have access to the results from all 12 classifiers, perform substantially better than the NN classifier ($p < .001$). The performance difference is particularly pronounced at false alarm rates above 0.25 (Fig. 5). There is no statistically significant difference between R and RS, which suggests that smoothing the raw classifier output in time has little effect on classification accuracy.

The mDIMF has a substantial effect on classifier fusion performance. In fact, the mDIMF can be used as a stand-alone fusion scheme, which yields a classification system with an AUC of 0.899, comparable to the R and RS schemes. Both of the meta-classification schemes that employ mDIMF perform substantially better than the ones that do not ($p < .001$). This suggests that mDIMF has a much larger effect overall than just classifier fusion or smoothing the classifier output.

Interestingly, there is a statistical difference between the performance of the RSM and RSMI schemes ($p < .005$), although, on a practical scale, this difference is very small. This suggests that, while the individual DIMF features add complexity to the classification system, they provide a net benefit to classification accuracy.

How does the mDIMF meta-feature aid classifier fusion to achieve substantially superior performance? We hypothesize that the mDIMF is acting not just as an additional classification mechanism, but rather *interacts with the other features to modify the effect of the raw classifier feature on the meta-classifier*. An additional experiment was performed to test this hypothesis.

To test the interaction effect of mDIMF on fusion scheme performance, an RF meta-classifier trained on the RSM data was exposed to a synthetic data set. One of the classifier inputs was varied systematically over 21 evenly spaced steps from its minimum to maximum value. The mean DIMF (mDIMF) input was held at one of three levels, -1, 0, or 1. This results in 63 unique combinations of input value. For each of the 63 combination of input value, the RF evaluated 50,000 cases, using randomly generated values for the other 23 input features (3,150,000 cases total). The mean fusion RSM output was recorded for each random realization. By holding one classifier input and mDIMF input at fixed levels, and varying the other inputs randomly over many combinations of values, the average effect of the classifier

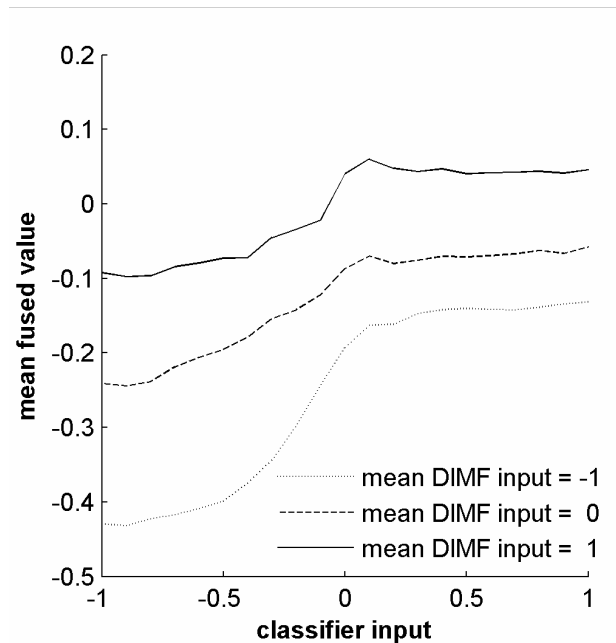


Fig 6. Mean RSM fusion scheme output as a function of classifier input for three levels of mDIMF input. Note the clear interaction between classifier input and mDIMF.

and mDIMF interaction is isolated.

Fig 6. is a plot of the mean RSM fusion scheme output as a function of classifier input for each level of mDIMF input. This plot shows a clear interaction between classifier input and mDIMF input. At low mDIMF, the fusion scheme discounts low classifier input values substantially compared to high values of classifier input. Meanwhile, at high values of mDIMF, the fusion scheme discounts low classifier input values much less, about half as much as it does at low mDIMF values. This supports the hypothesis that the mDIMF does not only act as a stand-alone signal, but rather that mDIMF interacts with the other classifiers to modify their impact on the fused output.

IV. DISCUSSION

Fault detection is a critical first step in system prognosis. Avoidance of unscheduled downtime and costly secondary damage make the accurate detection of faults and prediction of equipment remaining useful life of enormous economic benefit to industry. It is often advantageous to fuse the output from an ensemble of classifiers [14, 18]. This paper presents an approach that uses the raw classifier outputs as an intermediate feature space [37] that a meta-classifier operates on to classify system state as faulty or normal. Based on the previous literature [14, 18, 33], it is not surprising that an ensemble of classifiers (in this case the R and RS meta-classifiers) were able to out-perform all of individual classifiers (listed in Table I) [33].

The most significant result of this research is that the use of meta-features that provides information about the historical output of the classifiers provides significant benefit to a meta-classification scheme for time series data. Meta-

classifiers that incorporated the DIMF or mDIMF features performed substantially better than those that did not. Another important finding is that the mean DIMF alone is a could be used to achieve good performance with less computational overhead (e.g., in on-wing applications, where computational power is limited) than the RF approaches.

The mDIMF feature is monotonic. This is an important characteristic for prognostic applications. It also appears (based on this early research) to provide an early warning of oncoming faults. Exploring how these properties can be harnessed for applicability in prognostics (i.e., remaining life estimates or time to flight deck effect estimates) will be the subject of future work.

Once a classification system is constructed (based on individual classifiers or classifier ensembles), domain expertise must then be applied to implement the system as part of a prognostic reasoner [38, 39, 1, 13]. For example, one issue important to practical application of this technique but beyond the scope of this paper is resetting the DIMF in the prognostic reasoner after the fault has been addressed. Another successful heuristic at implementation time might be to reset the DIMF of a particular classifier after a certain number of cycles if no fault manifests (e.g., if one classifier false-alarms). In addition, the method needs to be tested for run-time applications where the DIMF is calculated in a forward mode. Logic that guarantees that no unnecessary DIMF transitions are generated without exceeding a minimum threshold should also be implemented. These practical issues are subjects of future work.

REFERENCES

- [1] Ashby, M. and Scheuren, W. (2000), "Intelligent Maintenance Advisor for Turbine Engines (IMATE)", Proceedings of the IEEE Aerospace Conference, 11.0309, 2000.
- [2] Bishop, C.M. (1995), *Neural Networks for Pattern Recognition*, Oxford University, Press.
- [3] Breiman, L. (1996a), "Stacked regression", *Machine Learning*, Vol. 24, No. 1, pp 49-64.
- [4] Breiman, L. (1996b), "Bagging predictors", *Machine Learning*, Vol.24, No.2, pp 123-40.
- [5] Breiman, L. (2001), "Random forest", *Machine Learning*, 45(1), pp. 5-32.
- [6] Breiman, L.; Friedman, J.; Olshen, R. and Stone, C. (1984), *Classification and Regression Trees*, Wadsworth, Belmont, CA.
- [7] Breiman, L. and Cutler, A. (2004), *Random Forest Toolbox Version 5.1*, http://www.stat.berkeley.edu/users/breiman/RandomForestes/cc_software.htm, University of California at Berkeley.
- [8] Cawley, G.C. (2000), *Support Vector Machine Toolbox v0.50 beta*, <http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox>, University of East Anglia, School of Information Systems, Norwich, Norfolk, U.K. NR4 7TJ.
- [9] Dietterich, T.G. (1999), "Machine learning research: four current directions", *AI Magazine*, Vol. 18, No. 4, pp97-136.
- [10] Freund, Y. and Schapire, R. (1999), "A Short Introduction to Boosting", *J. Japanese Society Artificial Intelligence*, Vol. 14, No.5, pp. 771-780.
- [11] Freund, Y. and Schapire, R.E. (1996), "Experiments with a new boosting algorithm", In Saitta, L. (Ed.), *Proceedings of the 13th International Conference on Machine Learning*, pp148-156.
- [12] Gerra-Salcedo, C. and Whitley, D. (1999), "Genetic approach to feature selection for ensemble creation", *Proceedings of Genetic and Evolutionary Computation Conference*, pp236-243.
- [13] Goebel, K. (2001), "Architecture and Design of a Diagnostic Information Fusion Tool", *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 15 (4), Sept. 2001, pp. 335-348.
- [14] Hansen, L. and Salamon, P. (1990), "Neural network ensembles", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 10, pp 993-1001.
- [15] Haykin, S. (1999), *Neural Networks: a comprehensive foundation*, 2nd Ed., Prentice Hall, New Jersey, 1999.
- [16] Ho, T.K. (1998), "The random subspace method for constructing decision forests", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8), pp832-844.
- [17] Hu, X.; Eklund, N. and Goebel, K. (2006), "Using Rank Permutation for Aircraft Engine Fault Diagnostics", *Proceedings of 60th MFPT*, 2006, to appear.
- [18] Krogh, A. and Vedelsby, J. (1995), "Neural network ensembles, cross validation, and active learning", In Tesauro, G., Touretzky, D. & Leen, T. (Eds), *Advances in Neural Information Processing Systems*, Vol. 7, pp231-238, Cambridge, MA, MIT Press.
- [19] Loskiewicz-Buczak, A. and Uhrig, R. (1994), "Decision Fusion by Fuzzy Set Operations", *Proc. third IEEE Conf. Fuzzy Systems*, Vol. 2, pp.1412-1417.
- [20] Mao, J. (1998), "A case study on bagging, boosting, and basic ensembles of neural networks for OCR", *Proceedings of IEEE World Congress on Computational Intelligence*, Vol. 3, pp1828-33
- [21] Nadaraya, E. (1964), "On estimating regression". *Theory of Probability and its Applications*. 10, 186-190.
- [22] Nelson, M. and Mason, K. (1999), "A Model-Based Approach to Information Fusion". *Proc. Information, Decision, and Control*, pp. 395-400.
- [23] Opitz, D.W. (1999), "Feature selection for ensembles", *Proceedings of 16th International Conference on Artificial Intelligence*, pp379-384.
- [24] Petit-Renaud, S. and Denoeux, T. (2004), "Nonparametric Regression Analysis of Uncertain and Imprecise Data Using Belief Functions", *International Journal of Approximate Reasoning*, Vol. 35, No. 1, pp. 1-28.
- [25] Platt, J.C. (1999), "Fast training of support vector machines using sequential minimal optimization", *Advances in Kernel Methods - Support Vector Learning*, (Eds) B. Scholkopf, C. Burges, and A. J. Smola, MIT Press, Cambridge, Massachusetts, chapter 12, pp. 185-208.
- [26] Rao, N.S.V. (2000), "Finite Sample Performance Guarantees of Fusers for Function Estimators", *Information Fusion*, Vol. 1, no. 1, pp. 35-44.
- [27] Schapire, R.E. (1990), "The strength of weak learnability", *Machine Learning*, Vol.5, No.2, pp197-227.
- [28] Shimshoni, Y. and Intrator, N. (1998), "Classification of seismic signals by integrating ensembles of neural networks", *IEEE Transactions of Signal Processing*, Vol. 46, No. 5, pp1194-1201.
- [29] P. Smets (1994), "What is Dempster-Shafer's model?" *Advances in the Dempster-Shafer Theory of Evidence*, Yager, R., Fedrizzi, M., and Kacprzyk, J., (Eds.), John Wiley & Sons, New York, pp. 5-34.
- [30] Tumer, K. and Oza, N.C. (1999), "Decimated input ensembles for improved generalization", *Proceedings of the International Joint conference on Neural Networks*, Washington DC.
- [31] Vapnik, V.N. (1995), *The Nature of Statistical Learning Theory*, Springer-Verlag, New York.
- [32] W. Yan, K. Goebel, and J. Li (2002), "Classifier Performance Measures in Multi-Fault Diagnosis for Aircraft Engines", *Proceedings of SPIE, Component and Systems Diagnostics, Prognostics, and Health Management II*, vol. 4733, pp. 88-97.
- [33] Goebel, K., Hu, X., Eklund, N., and Yan, W. (2006), "Fusing diverse monitoring algorithms for robust change detection". *Proceedings of SPIE*, to appear.
- [34] Epanechnikov, V. (1969). "Nonparametric estimates of a multivariate probability density", *Theory of Probability and its Applications* 14: 153-8.
- [35] Bartlett, M. (1963). "Statistical estimation of density functions", *Sankhya, Series A* 25: 245-54.
- [36] Eklund, N. and Goebel, K. (2005). "Using neural networks and the rank permutation transformation to detect abnormal conditions in aircraft engines." *Soft Computing in Industrial Applications*, 2005.

SMCia/05. Proceedings of the 2005 IEEE Mid-Summer Workshop on; 28-30 June 2005 Page(s):1 – 5

- [37] Kuncheva, L., Bezdek, J., and Duin, R. (2001). "Decision templates for multiple classifier fusion: an experimental comparison", *Pattern Recognition*, Vol. 34, pp. 299-314.
- [38] Goebel, K., Bonanni, P., and Eklund, N., (2005). "Towards an Integrated Reasoner for Bearings Prognostics" IEEE Aerospace Conference, 05-12 March 2005, Page(s):1 – 11
- [39] Goebel, K., Eklund, N, and Bonanni, P. (2006). "Fusing Competing Prediction Algorithms for Prognostics." IEEE Aerospace Conference, 04-11 March 2006, to appear.
- [40] Venkatraman E. (2000). "A permutation test to compare receiver operating characteristic curves." *Biometrics*, 56, 1134-1138.