Performance Analysis of Different Machine Learning Algorithms for Identifying and Classifying the Failures of Traction Motors

Xiaoyu Xian, Haichuan Tang, Yin Tian*, Qi Liu and Yuming Fan
Dept. of Technology Research, CRRC Academy Co., Ltd., Beijing, China

*Corresponding author email: tianyin@crccgc.cc

Abstract. This paper addresses electric motor fault diagnosis using supervised machine learning classification. A total of 15 distinct fault types are classified and multilabel strategies are used to classify concurrent faults. We explored, developed, and compared the performance of different types of binary (fault/non-fault), multi-class (fault type) and multi-label (single fault versus combination fault) classifiers. To evaluate the effectiveness of fault identification and classification, we used different supervised machine learning methods, including Random forest classification, support vector machine and neural network classification. Through experiment, we compared these methods over 4 classification regimes and finally summarize the most suitable machine learning algorithms for different aspects of health diagnosis in traction motors area.

Keywords: Machine learning; Fault diagnosis; Random forest; Neural network.

1. Introduction

1.1. Motivation and Background
Rail freight is responsible for moving around 10 trillion tonne-kilometres each year [1]. Because of this heavy reliance on freight, a locomotive in-service failure that takes 5 hours to repair on a medium traffic stretch of rail (50 trains/day) costs an estimated $45,000 in resource consumption, labour, and opportunity cost [2]. In the US alone, an average of 300 in-service failures occurs each year, a small percentage of worldwide ISFs [3]. One locomotive component particularly prone to failure is the traction motor. Electric traction motors that provide the primary propulsion for locomotives, powered either directly from overhead transmission lines or, in the case of diesel-electric locomotives, from an on-board diesel generator. If these motors fail, a locomotive will be unable to move its load and will require maintenance before continuing operation. The most common traction motor failures are bearing faults, stator and rotor shorts or open circuits, or rotor eccentricities [4]. Traction motor failures comprise around 10% of all in-service failures, so quick resolution of these issues represents a large opportunity for cost and time savings.

Due to the large cost of in-service failures, significant efforts are made to accurately predict and diagnose component faults so repairs can be made quickly. Commercial diagnostics and monitoring software has become standard in the rail industry, with a variety of companies such as GE Transportation, Electromotive Diesel Inc., and Motive Power, Inc. offering comprehensive locomotive health monitoring and fault detection. These systems combine data from a variety of on-board sensors and a fault detection algorithm to detect and classify faults. Fault detecting schemes for locomotives traction motors and for most mechanical machinery falls into two main categories: knowledge-based “expert systems” and data-
based “machine learning” systems. Fast and reliable fault detection and classification is valuable for the rail industry because of the high and often cascading cost of equipment downtime and delays. Complicating detection is that components are often difficult to observe directly, making inspection costly. Traditional fault detection schemes for complex machinery generally require a model or expert knowledge to be hard-coded into the detection system. Data-driven methods, conversely, require less domain knowledge. These techniques can also detect many different fault types based on data from a single sensor and may be better able to detect minute changes not apparent from traditional fault detection. This simplicity is important because while many fault detection and classification problems can be solved to a high degree of accuracy by using multiple sensors, large data-sets, extensive preprocessing for feature extraction, and complex classification algorithms, these approaches are costly and sometimes impractical.

1.2. Related Work

In an expert system, knowledgeable “experts” define a base of knowledge about the correct operation of the mechanical system. This knowledge base includes rules for assessing the state of the system based on observable parameters input into the system (generally physical sensors in the mechanical system). These rules may be hard-coded (e.g. if temperature of motor exceeds a set reasonable operating temperature, system is in a fault state) or may be derived from a computational model of the system. Motor fault detection has been accomplished with an expert system via stator current monitoring [5] as well as monitoring for a specific frequency of motor vibration [4]. Expert systems have a few notable drawbacks, though. They require a significant amount of experience and application-specific knowledge to define the initial knowledge base, and the fault detection system will only be as powerful as this knowledge base and the defined set of rules are discerning.

Conversely, a machine learning fault detection system does not attempt to define the knowledge base and rules used to identify faults initially. Rather, examples of sensor readings for normal mechanical system operation, as well as examples for various possible fault conditions, are used to “train” the machine learning system. The fault detection algorithm learns the combinations of inputs that indicate various mechanical system states by fitting a mathematical function of the inputs so as to minimize error in the function output (fault state prediction). There are a few levels of detail possible for machine learning systems. In the simplest case, the algorithm makes a single binary prediction of whether the mechanical system operate in a normal or fault condition. This problem regime is known as binary classification. Motor fault detection has been achieved using a variety of binary classification algorithms, including artificial neural networks [6–13] and support vector machines [14]. If more detail about the specific failed component is required, a multiclass classification regime is used. Each class corresponds to a specific type of fault or to the normal condition, and the fault detection system predicts exactly one class based on the input information. The classification of motor data into one of a number of fault categories has also been accomplished repeatedly and to a high degree of success using artificial neural networks [15–18], support vector machines [19–22], decision trees [23], and multi-classifier ensembles [24]. A third regime, multi-label classification, extends this capability even further. In this scheme, each fault type is represented by a “label”, and the fault detection system assigns any number of labels to a data example based on the input information. This allows the system to predict when multiple faults occur simultaneously. Multi-label classification techniques based on artificial neural networks [25], relevance vector machines [26] and extreme learning machines [27] have recently been successfully used to classify motor or engine data with as many as 15 single and 5 double possible fault cases. Machine learning methods offer a few notable advantages over knowledge-based systems. They do not require specifying any application-specific expert knowledge in the fault detection system and are often able to learn very nuanced predictive rules. The main drawback of machine learning methods is that they require training data of normal and faulty mechanical system operation, which can be costly and in some cases unfeasible to produce. In general, more complex machine learning algorithms require more training data to be able to predict faults reliably. Despite this challenge, recent advances in fault detection have focused on developing highly complex algorithmic tools to detect faults with increased accuracy. For example, Delgado et al. achieved a high degree of accuracy in multiclass classification of bearing faults by creating a custom, application-specific neural network-based classifier [16]. Likewise, Wong
et al. employed a detailed method for engine fault diagnosis by using three sensors for data collection, two specialized feature extraction techniques, three separate Bayesian extreme learning machine classifiers, and a probabilistic committee machine that combines predictions from these three classifiers to make a final multi-label classification [27].

2. Data Generation

Data is gathered from a single accelerometer mounted on a motor test apparatus that allows faults to be deliberately introduced. It was created using a Machinery Fault Lite Simulator manufactured by SpectraQuest, Inc. in Richmond, Virginia. This machine contains an electric motor driving a shaft with two bearings, between which were mounted two rotor plates. Bearing faults were simulated by installing either a bearing with an inner race fault, outer race fault, or a ball fault. Rotor imbalance faults were simulated by adding threaded nuts to one or both of the two rotors in a total of nine configurations to create a load eccentricity. An accelerometer and wireless data transmitter were placed on the far bearing mounting.

Tests were conducted by running a motor under each of the fault conditions listed in Table 1 at 12, 14, 16, 18 and 20 revolutions per minute. Each test was divided into 0.8-second data-blocks during which the accelerometer voltage was recorded at 4096 evenly spaced times. A total of 384 such data-blocks of time-series data were recorded for each of the listed fault conditions. Each data block corresponds to one example for subsequent machine learning problem formulations. Additionally, some fault conditions were tested in combination with other faults, and these combinations are also listed for each fault condition in Table 1. Thus, a total of 46 total test conditions were run: one test with no faults, 15 with a single fault, and 30 with a double fault.

| Number | Fault Conditions | fault Label | Component Label | Multiclass Label |
|--------|------------------|-------------|-----------------|-----------------|
| 0      | None             | 0           | 0               | 0               |
| 1      | Ball, Level 1    | 1           | 1               | 1               |
| 2      | Ball, Level 2    | 1           | 1               | 2               |
| 3      | Ball, Level 3    | 1           | 1               | 3               |
| 4      | Ball, Level 4    | 1           | 1               | 4               |
| 5      | Inner Bearing Race | 1       | 1               | 5               |
| 6      | Outer Bearing Race | 1       | 1               | 6               |
| 7      | Rotor A Imbalance, 1 nut, inner position | 1 | 2 | 7 |
| 8      | Rotor A Imbalance, 1 nut, outer position | 1 | 2 | 8 |
| 9      | Rotor B Imbalance, 1 nut, inner position | 1 | 2 | 9 |
| 10     | Rotor B Imbalance, 1 nut, outer position | 1 | 2 | 10 |
| 11     | Rotor A Imbalance, 2 nuts, same side | 1 | 2 | 11-17 |
| 12     | Rotor A Imbalance, 2 nuts, opposite sides | 1 | 2 | 18-24 |
| 13     | Rotor A Imbalance, 3 nuts, same side | 1 | 2 | 25-31 |
| 14     | Rotor A and Rotor B Imbalance, same side | 1 | 2 | 32-38 |
| 15     | Rotor A and Rotor B Imbalance, opposite sides | 1 | 2 | 41-45 |
| 16     | Rotor and Bearing Combinations | 1 | 3 | 46 |

3. Data Preprocessing

3.1. Low-pass filtering
The motors in this experiment rotated at relatively low frequencies (12-20 revolutions per minute), so very high frequency accelerometer voltage changes (above 500 Hertz) were mostly due to noise in the collection process rather than a result of the motor test conditions. To remove this high frequency
noise, low pass filtering was performed by convolving a Hamming window of width eleven with the accelerometer voltage data.

3.2. Fourier transform
Because the data is based on a rotating source, individual accelerometer voltages were less useful predictors of motor condition than cyclical patterns in the data over the course of the test. That is, rather than trying to identify faults based on certain accelerometer voltages, the certain frequencies at which accelerometer voltages are most noticeably repeated were analysed. The features were first converted from the time domain to the frequency domain with a Fourier transform. Figure 1 shows one example data-block and the corresponding frequency-domain data obtained using an FFT. The data block originally had 4096 samples spaced at 0.8/4096 seconds. In the transformed feature-space, each feature corresponds to the magnitude of frequencies in a bin of size \( f_s/N \), where \( f_s \) is the sample rate. However, half of these bins correspond to negative frequencies which are mirror images of the positive frequencies, so these are ignored, yielding 2048 features or bins.

![Example Time Domain and Corresponding Frequency Domain Data.](image)

3.3. Principal Component Analysis
Principal component analysis (PCA) was used in this work to reduce the size of the feature-space to reduce over-fitting and increase processing speed. If the original set of \( n \) features for a dataset is viewed as a set of \( n \) dimensions which places each data example in \( n \)-space, PCA selects a new set of dimensions which also can project an example into \( n \)-space. However, these dimensions are special in that the first dimension is selected to preserve as much variance between data examples as possible; the second dimension is selected to be orthogonal to the first and to preserve as much of the remaining variance not preserved by the first, and so on. Thus, most of the information needed to define the data points is condensed into the relatively few dimensions. By comparison, discriminatory information may have been more or less equally contained in all of the features in the original feature-space. The data in this experiment initially contained 2048 features per example. After principal component analysis, all principal components containing less than 0.05% of total variance were removed, leaving a total of 284 principal-component features per example remaining.

3.4. Data Labeling
The descriptive labels (e.g. “Ball Fault Level 3”) were converted into a form recognizable by the classifier for each classification problem.
(1) For fault detection, all faulty data was labelled with a 1, and all normal data with a 0. Table 1 column “fault Label” shows assigned labels for each possible fault condition.

(2) For faulty component classification, normal, ball fault, rotor fault and combination fault data were labelled with a 0, 1, 2 or 3, respectively. Table 1 column “component Label” shows assigned labels for each possible fault condition.

(3) For multiclass fault classification, each unique combination of faults was assigned a unique integer. Table 1 column “Multiclass Label” shows assigned labels for each possible fault condition. In the meanwhile, for multi-label fault classification, each sample was assigned a binary label vector using K-hot encoding.

4. Numerical Experiments

4.1. Evaluation Metrics and Procedures

A generic accuracy equation scores a predicted set of labels $P_i$ correctly only if it exactly matches the true set of labels $T_i$.

$$\text{Accuracy} = \frac{\sum |P_i \cap T_i|}{\sum |T_i|}$$

This metric tends to underscore the predictive capability of a multilabel algorithm, as a prediction over $l$ labels could correctly predict $l-1$ labels for an example and still be considered incorrect. Thus, a few special multilabel metrics were used to evaluate the prediction accuracy for the multilabel results [28]. Hierarchical recall or sensitivity measures the proportion of predicted labels $P_{ij}$ that are correct to the total number of correct labels $T_{ij}$, considering $i$ examples and $j$ possible labels for each example.

$$hR = \frac{\sum |P_{ij} \cap T_{ij}|}{\sum |T_{ij}|}$$

Hierarchical precision or positive predictive value is the proportion of correct predicted labels to the total number of predicted labels.

$$hP = \frac{\sum |P_{ij} \cap T_{ij}|}{\sum |P_{ij}|}$$

Hierarchical f-measure is the harmonic mean of hierarchical recall and precision, used as a single metric by which to evaluate multilabel classification algorithms.

$$hF = \frac{2 \cdot hP \cdot hR}{hP + hR}$$

4.2. Levels of Testing

Three levels of granularity were considered in this experiment and each were tested separately. For fault detection, a classifier must determine whether a motor has any fault or has no fault. The goal of this trial was to assess the discriminative power of each classifier on a binary classification task. For faulty component classification, a classifier must determine the faulty component or components, if any. This trial was designed to assess each classifier’s suitability for a simple multiclass classification problem. For full fault classification, a classifier must determine the exact fault type or types and severity. Additionally, this last test was accomplished by two separate means for a total of 4 classification trials in the experiment; the first, multi-class classification using the label powerset method, transforms a multilabel classification problem into a multiclass problem by creating a separate class for each unique combination of labels. This trial was designed to assess each classifier’s suitability for a more complex multiclass classification problem. The number of classes in the label powerset increases exponentially with the number of possible labels in the multilabel problem, so this method becomes computationally infeasible with large sets of possible labels. The second method for full fault classification was multilabel classification, and this method does not have the drawback of exponential scaling with the number of possible labels. This trial assessed each classification algorithm’s discriminative power for a multilabel classification problem. Support vector classifiers are unable to solve problems of this
formulation, so only the random forest and neural network classifiers were used for this test. Thus, three trials were run for all three algorithms, and a fourth was run only for two of the three classifiers.

4.3. Machine Learning Trial
All classification algorithms were implemented in Python using scikit-learn [29]. For each trial, first 25% of data was removed to be used for final accuracy metrics. Then, on the remaining 75% of the data, grid search with shuffled and stratified 4-fold cross-validation was used to determine the optimal classifier hyper-parameters for each test. The following table shows the hyper-parameter options considered, grid search tests all possible unique combinations. The optimal hyper-parameters found by grid are bolded.

| Table 2. Random Forest Hyper-parameters. |
|----------------------------------------|
| Hyper-parameter Tested Values          |
| Number of classifiers 20, **200**      |
| Maximum tree depth **None**, 50, 100    |
| Maximum features per split 1, 3, 10    |
| Minimum samples per split 3, 10        |
| Minimum samples per leaf 3, 10         |
| Minimum weight fraction per leaf 0, 0.2, 0.5 |

| Table 3. Support Vector Machine Hyper-parameters. |
|-----------------------------------------------|
| Hyper-parameter Tested Values                 |
| Kernel function **polynomial**, radial basis, sigmoid |
| Degree of polynomial fit 1, 3, 5              |
| Shrinking **True**, False                      |

| Table 4. Multi-Layer Perceptron Hyper-parameters. |
|-----------------------------------------------|
| Hyper-parameter Tested Values                 |
| Activation Function logistic, identity, **rlu** |
| L2 regularization parameter **0.0001, 0.0003, 0.001** |
| Learning rate adaptive                         |

The models were trained and tested on a computer running a CPU with 32 logical cores at 3.4GHz, 64GB RAM, and NVMe solid state storage. The random forest implementation trains weak learners in parallel across all CPU cores. Lastly, the best-fitting hyper-parameters were used to train a classifier on all 75% of the training data, and this classifier was evaluated on the 25% data holdout.

5. Result

5.1. Accuracy
The following table shows the accuracy for each of the tested classification algorithms in the fault detection, faulty component classification, multiclass classification and multilabel classification problems.

| Table 5. Classification Accuracy. |
|-----------------------------------|
| Classifier Detection Component Multiclass Multilabel |
| Random Forest Classifier 0.998 0.968 0.965 0.831 |
| Support Vector Machine 0.979 0.880 0.872 — |
| Deep Neural Network 0.976 0.913 0.727 0.896 |
5.2. Multilabel Metrics
The multilabel performance metrics of overall accuracy, hierarchical recall (HR), hierarchical precision (HP) and hierarchical F1 measure (HF) are reported below for the random forest and deep neural network classifiers. Support vector machines cannot perform multilabel classification, so this classifier was omitted from the multilabel fault classification problem.

| Classifier         | Accuracy | HR  | HP  | HF  |
|--------------------|----------|-----|-----|-----|
| Random Forest      | 0.831    | 0.885 | 0.997 | 0.937 |
| Deep Neural Network| 0.896    | 0.947 | 0.953 | 0.950 |

5.3. Effects of PCA and Low-pass Filtering
The following table shows the accuracy of the random forest classifier on the multilabel classification problem using datasets prepared with and without PCA and low-pass filtering. As can be seen, the combination of principal component analysis and filtering significantly increases classification accuracy, but neither technique alone has a positive effect on accuracy. This suggests that the classifier performs better with fewer, more representative features (the result of PCA), but if this technique is used without first smoothing the high-frequency noise from the data, many of the resulting principal components will then correspond to the high-frequency features which are of little use in the classification and result in lower classification accuracy.

| PCA | Hamming | Smoothing | Accuracy |
|-----|---------|-----------|----------|
| Yes | Yes     |            | 0.92     |
| Yes | No      |            | 0.44     |
| No  | Yes     |            | 0.64     |
| No  | No      |            | 0.62     |

5.4. Sensitivity to Noise
The sensitivity of this technique to synthetic, random noise was assessed and the results shown in Figure 5. The random forest classifier was used to classify data to which random noise had been added at varying relative intensities. In this work, signal-to-noise ratio refers to the average magnitude of the data values, divided by the average magnitude of the noise. At a signal-to-noise ratio above 0.1, classification performance begins to decrease significantly, and at a signal-to-noise ratio of 1.0, classification accuracy is little better than random prediction.

![Figure 2. Test Accuracy versus Relative Noise Intensity.](image-url)
6. Conclusion
Random forest classification performed comparably to the more complex classifiers (support vector classification and neural network classification) on the fault detection trial. It outperformed both other algorithms on faulty component classification and full fault classification by label powerset trials. Only on the multilabel fault classification trial did the neural network classifier outperform random forest. The results of random forest classification in this paper show comparable accuracy to the results of more complex multilabel classifiers on problems with an equal or smaller number of possible faults, so these results suggest the promise of random forest for use in real-world classification regimes.

Each trial represents a substantial increase in classification problem complexity. Binary classification is the simplest, followed by component classification, full fault classification, and finally multilabel fault classification. For each classifier, increasing the problem complexity resulted in decreased accuracy. The better performance of the neural network classifier on the multilabel problem is likely a result of the highly adjustable structure of neural networks, which made this classifier more suited to learning the problem structure than the random forest classifier.

References
[1] Railway statistics: 2014 synopsis, 2014.
[2] Bryan W Schlake, Christopher PL Barkan, and J Riley Edwards. Train delay and economic impact of in-service failures of railroad rolling stock. Transportation research record, 2261(1):124–133, 2011.
[3] Rail safety fact sheet, Dec 2016.
[4] Subhasis Nandi, Hamid A Toliyat, and Xiaodong Li. Condition monitoring and fault diagnosis of electrical motors—a review. IEEE transactions on energy conversion, 20(4):719–729, 2005.
[5] Andrew M Knight and Sergio P Bertani. Mechanical fault detection in a medium-sized induction motor using stator current monitoring. IEEE Transactions on Energy Conversion, 20(4):753–760, 2005.
[6] Fiorenzo Filippetti, Giovanni Franceschini, Carla Tassoni, and Peter Vas. Recent developments of induction motor drives fault diagnosis using ai techniques. IEEE transactions on industrial electronics, 47(5):994–1004, 2000.
[7] Bo Li, M-Y Chow, Yodyium Tipsuwan, and James C Hung. Neural-network-based motor rolling bearing fault diagnosis. IEEE transactions on industrial electronics, 47(5):1060–1069, 2000.
[8] Sinan Altug, Mo-Yuen Chen, and H Joel Trussell. Fuzzy inference systems implemented on neural architectures for motor fault detection and diagnosis. IEEE transactions on industrial electronics, 46(6):1069–1079, 1999.
[9] M-Y Chow, Robert N Sharpe, and James C Hung. On the application and design of artificial neural networks for motor fault detection. ii. IEEE Transactions on industrial electronics, 40(2):189–196, 1993.
[10] Rangarajan M Tallam, Thomas G Habetler, and Ronald G Harley. Stator winding turn-fault detection for closed-loop induction motor drives. IEEE Transactions on Industry Applications, 39(3):720–724, 2003.
[11] Xiao Zhi Gao and Seppo J Ovaska. Genetic algorithm training of elman neural network in motor fault detection. Neural computing & applications, 11(1):37–44, 2002.
[12] Turker Ince, Serkan Kiranyaz, Levent Eren, Murat Askar, and Moncef Gabbouj. Real-time motor fault detection by 1-d convolutional neural networks. IEEE Transactions on Industrial Electronics, 63(11):7067–7075, 2016.
[13] Bulent Ayhan, M-Y Chow, and M-H Song. Multiple discriminant analysis and neural-network-based monolith and partition fault-detection schemes for broken rotor bar in induction motors. IEEE Trans- actions on Industrial Electronics, 53(4):1298–1308, 2006.
[14] P Konar and P Chattopadhyay. Bearing fault detection of induction motor using wavelet and support vector machines (svms). Applied Soft Computing, 11(6):4203–4211, 2011.
[15] Manjeevan Seera, Chee Peng Lim, Dahaman Ishak, and Harapajan Singh. Fault detection and diagnosis of induction motors using motor current signature analysis and a hybrid fmm–cart model. IEEE transactions on neural networks and learning systems, 23(1):97–108, 2012.

[16] M Delgado, Gianisalvo Cirrincione, A Garcia, JA Ortega, and Humberto Henao. A novel condition monitoring scheme for bearing faults based on curvilinear component analysis and hierarchical neural networks. In Electrical Machines (ICEM), 2012 XXth International Conference on, pages 2472–2478. IEEE, 2012.

[17] Jafar Zarei, Mohammad Amin Tajeddini, and Hamid Reza Karimi. Vibration analysis for bearing fault detection and classification using an intelligent filter. Mechatronics, 24(2):151–157, 2014.

[18] Vilas N Ghate and Sanjay V Dudul. Optimal mlp neural network classifier for fault detection of three phase induction motor. Expert Systems with Applications, 37(4):3468–3481, 2010.

[19] Saeed Abbasion, A Rafsanjani, Anoshirvan Farshidianfar, and Nishgoon Irani. Rolling element bearings multi-fault classification based on the wavelet denoising and support vector machine. Mechanical Systems and Signal Processing, 21(7):2933–2945, 2007.

[20] Xiaoyuan Zhang and Jianzhong Zhou. Multi-fault diagnosis for rolling element bearings based on ensemble empirical mode decomposition and optimized support vector machines. Mechanical Systems and Signal Processing, 41(1-2):127–140, 2013.

[21] Zhijian Liu, Hongrui Cao, Xuefeng Chen, Zhengjia He, and Zhongjie Shen. Multi-fault classification based on wavelet svm with pso algorithm to analyze vibration signals from rolling element bearings. Neurocomputing, 99:399–410, 2013.

[22] Tribeni Prasad Banerjee and Swagatam Das. Multi-sensor data fusion using support vector machine for motor fault detection. Information Sciences, 217:96–107, 2012.

[23] Bo-Suk Yang, Myung-Suck Oh, Andy Chit Chiow Tan, et al. Fault diagnosis of induction motor based on decision trees and adaptive neuro-fuzzy inference. Expert Systems with Applications, 36(2):1840–1849, 2009.

[24] Gang Niu, Tian Han, Bo-Suk Yang, and Andy Chit Chiow Tan. Multi-agent decision fusion for motor fault diagnosis. Mechanical Systems and Signal Processing, 21(3):1285–1299, 2007.

[25] George Georgoulas, Vicente Climente-Alarcon, Jose A Antonino-Daviu, Ioannis P Tsoumas, Chrysostomos D Stylios, Antero Arkkio, and George Nikolakopoulos. The use of a multilabel classification framework for the detection of broken bars and mixed eccentricity faults based on the start-up transient. IEEE Transactions on Industrial Informatics, 13(2):625–634, 2017.

[26] Wei Jiang and Liping Yang. A research on simultaneous fault diagnosis based on paired-rvm. In Computer Science and Education (ICCSE), 2017 12th International Conference on, pages 483–488. IEEE, 2017.

[27] Pak Kin Wong, Jianhua Zhong, Zhixin Yang, and Chi Man Vong. Sparse bayesian extreme learning committee machine for engine simultaneous fault diagnosis. Neurocomputing, 174:331–343, 2016.

[28] Carlos N Silla and Alex A Freitas. A survey of hierarchical classification across different application domains. Data Mining and Knowledge Discovery, 22(1-2):31–72, 2011.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.