Fault Detection and Diagnosis for Solar-Powered Wireless Mesh Networks Using Machine Learning

Vinicius C. Ferreira*, Ricardo C. Carrano, Joacir O. Silva, Célio V. N. Albuquerque,
Débora C. Muchaluat-Saade and Diego Passos
Midiacom Laboratory
Universidade Federal Fluminense - UFF
Niterói, Brazil
Email*: viniciusferreira@id.uff.br

***Abstract*—The inherent complexity of Wireless Mesh Networks (WMNs) makes management and configuration tasks difficult, specially for fault detection and diagnosis. In addition, manual inspections are extremely costly and require a highly skilled workforce, thus becoming impractical as the problem scales. To address this issue, this paper proposes a solution that makes use of machine learning techniques for automated fault detection and diagnosis (FDD) on solar-powered Wireless Mesh Networks (WMNs). We have used the Knowledge Discovery in Databases (KDD) methodology and a pre-defined dictionary of failures based on our previous experience with the deployment of WMNs. Thereafter, the problem was solved as a pattern classification problem. Several classification algorithms were evaluated, such as Naive Bayes, Support Vector Machine (SVM), Decision Table, k-Nearest Neighbors (k-NN) and C4.5. The SVM presented the best results, achieving a 90.59% overall accuracy during training and over 85% in validation tests.**

***Index Terms*—Machine learning, fault detection and diagnosis, wireless mesh networks.**

I. Introduction

Wireless Mesh Networks (WMNs) inherit the challenges and dynamic nature of wireless links, while typically presenting complex topologies. A high degree of automated configuration and maintenance is a very desirable feature in these networks, since their operation will frequently require skilled labor. The high complexity of these networks, coupled with the difficulty of access they may have, resulted in the search for automated management methods in order to improve reliability and reduce costs and downtime. In this scenario, maintenance is fundamental and an automated fault detection and diagnosis method must be pursued.

The problem of fault detection and diagnosis (FDD) is a widely studied field in engineering [1][2]. This problem is typically solved using one of three methods: (1) analytical solution and (2) application of statistical models, which comprises quantitative methods [3], or (3) the use of Artificial Intelligence (AI), as the history-based method [4]. Given the difficulties to model a WMN, its behavior, usage, and possible changes to the system, the use of quantitative methods becomes impractical, therefore, the AI approach is used in our work.

Our proposition's goal is to define a methodology that provides automated FDD for WMNs. For this purpose, a highly complex WMN deployed in the REMOTE project [5] was used as our solution target and its testbed network as our development environment. The REMOTE network consists of 41 mesh nodes installed along a power line for communication and supervision purposes. During seven years since deployment, REMOTE's network nodes presented different failure modes that could only be diagnosed through costly and sporadic physical inspections, which often required the shutdown of the transmission line along which the nodes were installed. This failure modes' knowledge was used to define a dictionary of failures, which is our diagnostic space.

We have already developed a WMN management platform performing network monitoring functions, named MeshAdmin [6]. Therefore, network nodes' monitored data were used as our history database. It was necessary to define a method and a specific AI technique to perform the FDD task. The Knowledge Discovery in Databases (KDD) methodology [7] and the machine learning techniques [8] were chosen for the proposition.

To produce a history of labeled faults, a set of real-problem emulations were performed in the network. Based on a labeled history database, the supervised learning approach was used as the machine learning technique. Several classification algorithms were considered and tested, namely: *Naive Bayes*, *Support Vector Machine* (SVM), *Decision Table*, *k-Nearest Neighbors* (k-NN) and *C4.5*. Another database, containing only naturally occurred faults' data, not emulated ones, was used for results validation.

This paper goals are: (1) propose the KDD methodology and the supervised learning approach to solve the fault detection and diagnosis problem in WMNs; (2) describe each solution step, the difficulties faced during the development of the proposed solution and how they were overcome; (3) provide an autonomous FDD module to be part of the WMN management integrated platform.

The remainder of the paper is structured as follows. In Section II we provide a brief description of related work. Section III presents our proposal for fault detection and diagnosis in solar-powered WMNs. Each selected classification algorithm and its performance are presented in Section IV. Finally, Section V brings conclusions and future works.

978-3-901882-89-0 @2017 IFIP

II. Related Work

WMNs' complexity begins with its unstable wireless medium, the requirements for an enhanced scalability in a multi-hop mesh environment and energy management. These obstacles reflect in the communication protocol and network architecture [9], demanding expertise and skilled workforce in operation.

In WMNs, there are two types of nodes: mesh routers and mesh clients. Some WMN architectures can be more flexible, with mesh routers built on general-purpose computer systems or even allowing mesh clients to perform essential mesh networking functions [10]. This work focuses on Infrastructured WMNs, which consists of mesh routers forming a backhaul for clients to connect. The WMN infrastructure can be built using various types of radio technologies and protocols [11].

Infrastructured WMNs present some critical issues to network management, such as: dynamic environment, fluctuating link quality, unfriendly placement and resource constraints [12]. Management functions as monitoring, fault detection and diagnosis are essential to surpass those challenges and must be carried out.

Fault detection and diagnose techniques can be broadly characterized into distributed and centralized approaches. The distributed approach relies on local-view and the absence of centralized monitoring to reduce network data traffic [13]. The centralized approach relies on data aggregation and data processing in a node with higher computational power, without resource constrains, reducing network nodes' resource use. Since our proposal already takes advantage of a centralized monitoring tool and some networks might not offer a proper local-view by neighbors information exchanges, as the REMOTE's network by its linear aspect, the centralized approach was the appropriate choice.

Other studies use a centralized approach to perform the diagnosis [14][15][16]. Among them, [14] uses information from the node's routing table and the RIPPER algorithm [17], a classifier based on rules for intrusion detection. An automated FDD technique using simulation models is proposed in [15]. The work uses monitored network metrics to feed a simulator and compares the performance of the simulated model to the network's gathered data. If there is a significant difference, the proposal systematically injects faults in the simulated environment. When the performance of the simulator after a specific fault gets close to the network's performance, the system assumes a particular fault as the diagnosis. Sympathy [16] deals with FDD in wireless sensor networks. The proposed solution relies on connectivity metrics, flows and node operation information. The collected data feeds a decision tree, based on empirical knowledge of an expert, to diagnose the problem cause.

Different from previous proposals, the solution presented in this paper employs a centralized and completely autonomous process for fault detection and diagnosis on an individual node level, based on machine learning techniques. This approach has the advantage of not requiring an analytical model or the supervision of an expert in the operation phase. It also seeks an increased diagnosis accuracy, reducing costs and workforce required for network maintenance and a possible lack of information can be treated more effectively, since it does not use simulation.

III. Proposed Solution

The proposed methodology is based on the KDD, which follows some traditional steps — the so-called five steps for extracting knowledge from a database [7]. These steps are: attribute selection, preprocessing (data cleaning and enrichment), data transformation (if needed), data mining and result evaluation.

Our proposition starts defining the scope of action of the system. Using our previous WMN operation experience [5][12], a set of specific faults were pointed-out as relevant and critical to network maintenance. These faults are: high processor usage, high RAM consumption, battery failure, low efficiency of the powering system, in our case a solar panel, antennas misalignment and defects on RF cable connectors. The regular operational state of a mesh node and the defined dictionary of failures are our diagnosis space. A history database with labeled data had to be produced to train a classifier. To this purpose, fault occurrences were emulated in a testbed network. After that, the KDD methodology steps were used as guideline.

A. REMOTE Network

The REMOTE project deployed a communication infrastructure based on WMN. While the ultimate goal is to deploy our solution at the production network of the REMOTE project, in the development phase, prototypes were evaluated in a WMN testbed located on one university campus at UFF.

Both the production network and the testbed are infrastructured WMNs and have similar characteristics as energy constraints, the use of a solar power system and multiple radios with the same technologies. Therefore, the methodology and the scenarios used in the development phase have immediate applicability for the production network.

Each mesh router is composed of three modules:
Communication module: consists of a router with two wireless interfaces, a client access interface consisting of an IEEE 802.11g radio, and an interface for communication between nodes (backbone), consisting of an IEEE 802.11a radio. The backbone radio is connected through an RF splitter to two directional antennas pointed towards specific nodes. The network protocol used is the Optimized Link State Routing (OLSR) [11], a link state based protocol designed for ad hoc networks. In the REMOTE's network an OLSR variation is used, the OLSR-ML. This variation uses as cost function the Minimum Loss (ML) metric [18], which results in routes with minimum error probability in end-to-end communication.

Power module: is formed by a solar power system that comprises a 40 W solar panel, a charge controller and a bank of three 12 V/7Ah lead-acid sealed batteries connected in

parallel, resulting in a voltage of 12 V and total rated capacity of 21 Ah.

Sensing module: used for site supervision, contains two LM35 temperature sensors, one LDR 5mm light sensor, voltage and current sensors for the solar panel, batteries and primary load (the communication module). It allows monitoring the following physical data of the mesh router: Solar Panel Voltage, Solar Panel Current, Battery Voltage, Communication's Module Voltage, Communication's Module Current, External (ambiance) Temperature, Internal (sealed box) Temperature, Incident Light Intensity, Bytes in/out for each Network Interface, Available Flash Memory, Available RAM memory, CPU Load Average and Link Quality. Instantaneous samples of these data are collected and stored by MeshAdmin every ten minutes.

B. Populating the Training Database

After the fault dictionary was defined, it was necessary to obtain samples of each of these events occurrences to generate a proper training database. At this stage, fault emulations were performed by causing artificial problems in the network for five months.

Six emulations were performed, inflicting faults and monitoring their effects in the collected data. Each one had an observable influence over a mesh node monitored parameters. In the following paragraphs, we define these emulations and explain how they have influenced the monitored parameters when compared to a node at regular operational state.

High processor usage - The objective was to create high demand for CPU usage in the node. To emulate the effect of a large and constant flow of processing requests, a random byte stream was compressed while the average CPU Load was monitored by MeshAdmin (among all other collected node information).

High RAM consumption - Network nodes used in this work have a reduced amount of available flash memory (4 MB), used mostly for the operating system installation. When the node is on, most file activities occur in RAM, including writing into temporary files. If a faulty process creates a large temporary file, the node may run out of memory, affecting core functionalities. On extreme cases, the lack of available RAM may cause node unresponsiveness, which triggers a watchdog to reboot it. In this experiment, we employed a process that creates large temporary files in order to verify how this behavior would be manifested in the data collected by MeshAdmin.

Battery Failure - The battery pack is comprised of three batteries in parallel. Initially, tests consisted of replacing one or more good batteries for defective batteries. Subsequently, additional tests were performed with an incomplete battery bank, with batteries removed gradually. A reduction of the node autonomy could be observed, resulting in a shutdown during certain periods of the day. As expected, node autonomy varied according to the number of defective or missing batteries, but also according to the intensity of sunlight during the period, which affects battery charging during the day.

Low Efficiency of the Solar Panel - Several different tests on the solar panel were performed, all aiming at reducing its efficiency. This was accomplished by casting full and partial shadows over the panel and also by varying the shadow incidence angle. Most tests, indeed, resulted in a lower efficiency of the solar panel, also reducing the node autonomy.

Antennas Misalignment - Directional antennas were misaligned to cause a drop on the received signal strength. To register this link quality drop, the routing metric was monitored. During misalignment tests, the quality level of the tested link varied, as expected.

Defects on RF cable connectors - Antennas are connected to the router via an N-type male connector. The connector was brought to a poor contact condition, which caused a significant increase in the variance of the signal quality level, as expected.

C. Attribute Selection

The impact of each of the nodes' states in the monitored parameters were evaluated. A comparison of the node behavior and its measured parameters before and during tests was performed. This way, it was possible to infer the most relevant parameters using MeshAdmin graphical analysis tools.

In order to evaluate if other non-considered parameters could affect the diagnosis, the following methodology was employed: (1) All measured parameters were considered as a primary candidate group and the overall accuracy of a classification algorithm was measured; (2) Each parameter was then progressively removed to form a new candidate group, this candidate group's overall accuracy was also measured; (3) If the parameter removal resulted in an overall accuracy gain, the parameter was definitely discarded. The process was repeated until all available parameters were either discarded or preserved and a final group of relevant parameters could be obtained.

D. Preprocessing and Transformation Steps

During the steps of preprocessing and data transformation, long periods of node inactivity, resulting in unusable information, spurious data and other known issues, such as infrastructure network problems affecting the server, were removed from the database.

Also, notice that the gathered data is a time series composed of instantaneous samples of the monitored parameters, while nodes' failures progress over time. Hence, it was necessary to define an observation interval comprising several samples, so a single instance in the training database was representative of a failure.

The first step of transformation was scaling the data. The main advantages of scaling are to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges and numerical difficulties during the calculation [19], therefore the data was normalized.

The data enrichment and transformation process was based on the description of a time continuous phenomena, like battery charge and discharge. To this end, the simple moving average and moving standard deviation of each parameters'

time series were evaluated using the following past intervals' samples: 1 hour, 2 hours, 4 hours, 8 hours, 12 hours and 24 hours. To evaluate which observation interval was more adequate, we used the same methodology employed in the selection step. The overall accuracy gain of each observation interval was measured and the observation interval with higher overall accuracy was chosen.

E. Data Mining and Evaluation

The sensing module generated data containing several numerical attributes, and we have added one nominal attribute to the problem, the class. Different classification models can be used to solve problems with such characteristic. These models are: statistical, linear, rule-based, instance-based and divide-and-conquer [20].

Statistical models are used to study the probability of an instance belonging to a certain class based on the value of its attributes and distribution of each attribute per class. The most common statistical models are based on Bayes Theorem of conditional probability. In this work, the Naive Bayes [21] algorithm was chosen to represent this group of algorithms.

Linear models are divided into two subgroups: those using linear regression and linear transformation to create a class membership function, and those that work with the hypothesis of linear separability of classes, and thus try to find hyperplanes that divide the classes in a vector space. For this work, the Support Vector Machine (SVM) [22] was used.

Rule-based models focus on trying to find rules which best separate the given class from the others. These models start from the observation of a specific class to create a rule that separates it from the others as accurately as possible. At the end, a set of rules is created and evaluated in succession to determine the class of a given instance. The rule-based algorithm tested in this work was the Decision Table Majority [23].

Instance-based models use distance functions to determine a group of instances at the training database closest to the evaluated instance. The majority class of this group is chosen to label the targeted instance. Under this class of algorithms, we used the k-Nearest Neighbors (k-NN) [24].

Finally, the divide-and-conquer models, usually represented as decision trees, recursively analyze each attribute and possible cutoff points that result in a subgroup of instances that better separate classes, with higher level of purity of the majority class in each subgroup. The algorithm C 4.5 [25] was used.

The performance measures taken in order to evaluate the classification algorithms are: the overall accuracy, the ratio of correct classifications of all classes; F-Measure [26], the harmonic mean of Precision and Recall, as a measure for the exactness and completeness of an algorithm classification for each class.

Given these classification algorithms and the performance measurements, the KDD steps could be started. All the classification algorithms were tested, the overall accuracy and F-Measure used to evaluate their performance and the confusion matrix was the representation used to show the results of the best algorithm for the task. For the construction of the classifiers and performance tests the Weka tool [27] was used.

IV. Results and Discussion

The results presented in this section are divided in three parts:problem description, algorithm evaluation and results evaluation.

A. Problem Description

The first investigation was about the observation interval, and the Decision Table was used for that purpose. This choice was based in the algorithm simplicity — it does not have any parameter adjustments —, and also because it has an attribute selection embedded. Therefore, the lack of a previous attribute selection does not affect its results as it would affect other algorithms. Later, when the attribute selection step takes place, each attribute will already be in its best representation.

The observation intervals were: 1h, 2h, 4h, 8h, 12h and 24h. The Decision Table overall accuracy results are depicted in Table I. As the observation interval increased, the algorithms overall accuracy also increased, but they did not result in a significant change of the overall accuracy. Thefore the tests were stopped with 24 hour observation period as the highest accuracy result.

TABLE I
Overall accuracy per Observation Interval

Observation Interval (h)	1	2	4	8	12	24
Overall Accuracy (%)	71.08	73.99	76.25	78.13	78.13	78.88

This result is consistent with the nature of the problem, since it accounts for a full day (e.g., a full cycle of battery recharge due to sunlight is comprised, as well as a full cycle of discharge at night). Some of the faults only manifest themselves in certain periods of the day. For example, a battery failure is probably not detectable during day, when the solar panel is capable of supplying energy to the system. However, during the night, the failure becomes apparent with a faster discharge of the batteries.

In summary, each instance of the training database is comprised of a moving average and standard deviation of the samples during the past 24-hour window. Each instance also contains the timestamp of the last sample from the observed period.

The next step is the attribute selection. Several candidate groups were tested and the C4.5 algorithm was used to evaluate this test. The C4.5 is a tree algorithm that evaluates the potential class entropy reduction for each attribute. Therefore, it can be seen as a classification algorithm with attribute selection embedded method, since it already assesses all attributes and only uses the relevant ones. The C4.5 was used with the confidence factor (CF) set to its default value, 0.5.

Initially, the group of all attributes were: date, uptime, the moving average and standard deviation over the past 24h of the physical data monitored by the sensing module. All these

attributes are listed in Table II and the selected ones are marked with an X.

After the attribute selection, each instance used for the classification was defined as: date, uptime, solar panel current average, battery voltage's standard deviation and average, internal (sealed-box) temperature average, incident light intensity average, available memory average, cpu load average and link quality's standard deviation and average, class (representing a fault or regular operation).

TABLE II
SELECTED ATTRIBUTES FOR ALGORITHM EVALUATION

Monitored parameter	Standard deviation	Average
Solar Panel Voltage		
Solar Panel Current		X
Battery Voltage	X	X
Communication Module Voltage		
Communication Module Current		
External Temperature		
Internal Temperature		X
Incident Light Intensity		X
Available Memory		X
CPU Load		X
Link Quality	X	X
Network Interfaces Traffic		

B. Algorithm's evaluation

The selected algorithms were applied to the training database and evaluated. The k-NN, C4.5 and SVM algorithms have parameters to be set while the Decision Table and Naive Bayes do not have any parameters. For k-NN, the k parameter corresponds to the size of the group which will be used to define the class of the target instance. For the C4.5 algorithm, the parameters are the minimum number of instances per leaf of the created decision tree and the confidence factor (CF), a parameter used to prune the tree. For the SVM, the C parameter, indicating the complexity and limits for the solution, and also a definition of the kernel function were used.

1) Naive Bayes and k-NN: Despite all efforts, the Naive Bayes and k-NN results were much lower than expected when compared to other algorithms. To use the Naive Bayes algorithm the data was discretized using an algorithm embedded in Weka [28], which uses an information entropy minimization heuristic as a tree algorithm. The Naive Bayes obtained an overall accuracy of 49.50%. Meanwhile the k parameter of k-NN was investigated, the best overall accuracy result for this algorithm was 46.02%, found when using k=1. Hence, there were no further investigations using those classifiers and their results will be used just for comparison purposes.

2) Decision Table: As already depicted in the problem description, the Decision Table obtained an overall accuracy of 78.88%. The F-Measure is presented in the Table V, and its lowest point is the High Memory Consumption. This was an unexpected result since the detection of a Battery Fault is considered the most complex problem due to all variables involved, as the battery natural charge/discharge cycle, the climate influence in the charging cycle and node's energy consumption in the discharge cycle, while High Memory Consumption is one of the simplest, possibly diagnosed with just one attribute, the available memory average.

3) SVM: For the SVM, there are two settings to be made: the kernel function and the C parameter. Initially the C parameter was set to 1 and the kernel functions were evaluated with kernel parameters set to zero. The kernel variation results, in Table III, were in favor of a linear kernel function. Therefore this was the kernel used to find the C parameter with best performance.

TABLE III
SVM'S OVERALL ACCURACY PER KERNEL FUNCTION

Kernel Function	Linear	Polynomial	Radial Basis	Sigmoid
Overall Accuracy (%)	77.05	74.79	76.30	75.43

With the kernel function set with a linear function, the search for the parameter C took place. After C=100 the results did not improve significantly while the processing time suffered a large increase. The search was stopped in C=100, which represents the result, and the overall accuracy increased with the C parameter as in Table IV.

TABLE IV
SVM'S OVERALL ACCURACY PER C PARAMETER

C	1	10	25	50	100
Overall Accuracy (%)	77.05	87.79	88.43	89.3	90.59

The best overall accuracy of 90.59%, achieved with linear kernel function and C=100, will be used in the overall classifiers comparison. Observing the F-Measure in Table V, it is possible to verify the difficulty in diagnosing the Battery Fault. Therefore this might be a weakness of this classifier.

4) C4.5: The C4.5 algorithm minimum instances per leaf was set in 2, a small value, and the confidence factor was varied. The tests have shown that the confidence factor variation did not have a significant impact over the overall accuracy, with less than 1% of accuracy gains. The C4.5 algorithm presented a good result, the overall accuracy of 88.45% will be used as the C4.5 score, achieved with a set of CF=0.5 and 2 instances per leaf. The F-Measure result in Table V shows the same weak points as the majority of the classifiers, the Battery Failure and Low Solar Panel's Efficiency.

C. Results Validation and Interpretation

The final step is to compare all the classifiers and choose the best ones to validate. The best classifiers generated with each algorithm are now compared in Table V. The highest accuracies and F-measures were achieved by SVM and C4.5 algorithms.

TABLE V
F-MEASURE PER CLASS AND THE OVERALL ACCURACY

Class	Naive Bayes	1-NN	Decision Table	SVM	C 4.5
Regular Operational State	52.25	60.19	80.49	94.9	88.19
High Processor Usage	52.41	62.21	78.99	99.9	88.28
High Memory Consumption	50.87	23.09	71.29	93.6	85.98
Battery Failure	44.09	51.83	75.36	87.6	81.47
Low Solar Panel's Efficiency	43.83	45.27	74.92	93.1	81.91
Antennas Misalignment	51.03	46.78	80.94	90.9	87.92
RF Cable Connector defected	51.67	56.87	81.64	90.7	87.57
Overall Accuracy	49.05	46.02	78.88	90.59	88.45

Additional validation tests were performed with the data of a mesh node collected during one month. This validation database had two distinct phases, the first fortnight, in which the node appeared to be in perfect state, and the second fortnight, the moment when a battery fault occurred.

The C4.5 classifier was the first to be validated. The results were not as expected, the real accuracy was under 65%, in the best case, battery fault diagnosis. In the first fortnight the accuracy was of 5% of the regular operation state diagnosis, which indicates an overfitting to the training data. This model was then discarded.

The SVM algorithm responded as expected in the first fortnight, classifying 85% of the entries as regular operational state. For the second fortnight 40% of the classifications were indeed for battery fault and 37% were for low solar panel's efficiency. As previously presumed during SVM's F-Measure analysis, the distinction between battery fault and low solar panel's efficiency was proved a weak point.

Besides the F-Measure, another indicator of this assumption was the confusion matrix for the SVM classifier presented in Table VI. In this table, the columns are the real classes, while the rows are the classifier predictions.

TABLE VI
SVM CONFUSION MATRIX

	regular operational state	high processor usage	high RAM memory consumption	battery failure	low efficiency of the solar panel	antennas misalignment	defects on the RF cable connectors
regular operational state	8148	0	0	4	5	0	0
high processor usage	4	863	0	0	0	0	0
high RAM memory consumption	17	0	812	3	19	3	0
battery failure	15	0	2	3172	295	5	0
low efficiency of the solar panel	9	0	2	216	3436	1	0
antennas misalignment	0	0	0	6	1	4049	31
defects on the RF cable connectors	0	0	0	0	0	3	1724

Through the confusion matrix, it is possible to notice a correlation between battery failure and low efficiency of the solar panel. This correlation between these diagnoses represents a possibility of error, as already seen in the validation. However, the solution presented high reliability indicators, such as high accuracy and F-Measure.

A battery fault diagnosis usually takes advantage of a battery current sensor as a fundamental asset to verify the battery capacity, instrument not used in this work. Another noticed point was Battery Failure's different causes, such as lower capacity, higher inner resistance, among others [29]. In the database population tests only the high inner resistance was considered, while in the validation the cause was lower capacity, which might have led to this misinterpretation.

Analyzing the likelihood of the output of each class by the classifier, it was possible to see that in the Low Solar Panel's Efficiency classification cases the likelihood of this diagnosis was around 60% of certainty, while for these same cases, the probability for Battery Fault was 30%. To workaround this problem, the solution was using as output the two most probable causes of the diagnosis and their likelihood as a final result.

This way the autonomous diagnosis solution for solar-powered WMNs is feasible and satisfactory. Using the two most probable causes as the SVM classifier result, a 93.7% accuracy, measured as the ratio of correct classification in the primary or secondary diagnoses, was achieved in its worst scenario, battery fault.

V. CONCLUSIONS

This work presented a proposal for an autonomic solution for FDD in solar-powered WMNs, using AI techniques. To use an AI approach a reduced scope of the work was determined by the creation of a fault dictionary. As the fault dictionary was created, the problem could be seen as a pattern recognition problem, more specifically, a classification problem.

Several classification algorithms were considered and studied in the process. A performance evaluation method had to be defined and the search for the most suitable classifier has been carried on. For this purpose, the steps for knowledge discovery in databases were followed. Five months of real network tests were made to populate the database generating a set of examples to be worked upon in order to produce training, test and validation databases.

With the databases formed, we compared a number of well-known classification algorithms for the problem, namely Naive Bayes, Decision Table, k-NN, SVM and C4.5. The result of this evaluation showed that the C4.5 and the SVM algorithms had the best overall prediction performances, with accuracy over 85%. This accuracy level indicates that an autonomous solution is, indeed, feasible.

Their results were brought to a validation test. In this test, the C4.5 presented overfitting characteristics, with poor results when new data was used. While the SVM has shown a good overall performance. An already expected weak point, the Battery Failure detection was identified. This problem was solved using a multi-classification solution - the two classes with higher likelihood of success are presented to the user. With this adjustment, the classifier presented the correct diagnosis (between the two indicated) in all cases and the work was considered satisfactory.

Our future work is to investigate an outlier detection and the multi-class approach, since our final solution was in favor of the multi-class. After those tests, we intent to integrate the solution to MeshAdmin and perform tests for possible adjustments and deployment in the REMOTE testbed network. As a final goal, the solution will be implemented in the production network of the REMOTE project.

ACKNOWLEDGMENT

The authors would like to thank CAPES, CNPQ, FAPERJ, ANEEL and TBE, for making this work possible through research support.

REFERENCES

[1] N. Bayar, S. Darmoul, S. Hajri-Gabouj, and H. Pierreval, "Fault detection, diagnosis and recovery using Artificial Immune Systems: A review," *Engineering Applications of Artificial Intelligence*, vol. 46, pp. 43–57, 2015.

[2] Y. Zhang and J. Jiang, "Bibliographical review on reconfigurable fault-tolerant control systems," *Annual reviews in control*, vol. 32, no. 2, pp. 229–252, 2008.

[3] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis: Part I: Quantitative model-based methods," *Computers & chemical engineering*, vol. 27, no. 3, pp. 293–311, 2003.

[4] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin, "A review of process fault detection and diagnosis: Part III: Process history based methods," *Computers & chemical engineering*, vol. 27, no. 3, pp. 327–346, 2003.

[5] B. Siqueira, D. Passos, D. C. Muchaluat-Saade, and C. V. N. Albuquerque, "LIBR: ID-based routing for linear Wireless Mesh Networks," in *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*. IEEE, 2015, pp. 461–466.

[6] R. De Tommaso do Valle and D. C. Muchaluat-Saade, "MeshAdmin: An integrated platform for wireless mesh network management," in *2012 IEEE Network Operations and Management Symposium (NOMS)*. IEEE, 2012, pp. 293–301.

[7] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI magazine*, vol. 17, no. 3, p. 37, 1996.

[8] T. M. Mitchell, *Machine Learning*, 1st ed. McGraw-Hill, 1997.

[9] R. C. Carrano, L. Magalhães, D. C. Muchaluat-Saade, and C. V. N. Albuquerque, "IEEE 802.11s multihop MAC: A tutorial," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 1, pp. 52–67, 2011.

[10] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Computer Networks*, vol. 47, no. 4, pp. 445–487, 2005.

[11] M. E. M. Campista, P. M. Esposito, I. M. Moraes, L. H. M. K. Costa, O. C. M. B. Duarte, D. Passos, C. V. N. Albuquerque, D. C. Muchaluat-Saade, and M. G. Rubinstein, "Routing metrics and protocols for wireless mesh networks," *IEEE Network*, vol. 22, no. 1, pp. 6–12, 2008.

[12] J. L. Duarte, D. Passos, R. L. Valle, E. Oliveira, D. C. Muchaluat-Saade, and C. V. N. Albuquerque, "Management issues on wireless mesh networks," in *2007 Latin American Network Operations and Management Symposium (LANOMS)*. IEEE, 2007, pp. 8–19.

[13] J. Chen, S. Kher, and A. Somani, "Distributed fault detection of wireless sensor networks," in *Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*. ACM, 2006, pp. 65–72.

[14] Y. Zhang and W. Lee, "Intrusion detection in wireless ad-hoc networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*. ACM, 2000, pp. 275–283.

[15] L. Qiu, P. Bahl, A. Rao, and L. Zhou, "Troubleshooting wireless mesh networks," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 5, pp. 17–28, 2006.

[16] N. Ramanathan, K. Chang, R. Kapur, L. Girod, E. Kohler, and D. Estrin, "Sympathy for the sensor network debugger," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*. ACM, 2005, pp. 255–267.

[17] W. W. Cohen, "Fast effective rule induction," in *Proceedings of the twelfth international conference on machine learning*, 1995, pp. 115–123.

[18] D. Passos, C. V. N. Albuquerque, M. E. M. Campista, L. H. M. K. Costa, and O. C. M. Duarte, "Minimum loss multiplicative routing metrics for wireless mesh networks," *Journal of Internet Services and Applications*, vol. 1, no. 3, pp. 201–214, 2011.

[19] C.-W. Hsu, C.-C. Chang, C.-J. Lin *et al.*, "A practical guide to support vector classification," *See: https://www.cs.sfu.ca/people/Faculty/teaching/726/spring11/svmguide.pdf*, 2003.

[20] I. H. Witten, F. Eibe, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd ed. Morgan Kaufmann, 2011.

[21] I. Rish, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22. IBM New York, 2001, pp. 41–46.

[22] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 1–27, 2011.

[23] R. Kohavi, "The power of decision tables," in *Machine Learning: ECML-95*. Springer, 1995, pp. 174–189.

[24] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine learning*, vol. 6, no. 1, pp. 37–66, 1991.

[25] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[26] M. V. Joshi, "On evaluating performance of classifiers for rare classes," in *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*. IEEE, 2002, pp. 641–644.

[27] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[28] K. B. Irani and U. M. Fayyad, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proceedings of the International Joint Conference on Uncertainty in AI*, vol. 334, no. 571, 1993, pp. 1022–1027.

[29] A. Fasih, "Modeling and Fault Diagnosis of Automotive Lead-Acid Batteries," Ph.D. dissertation, The Ohio State University, Ohio, Jun. 2006.