Intelligent Industrial IoT system for detection of short-circuit failure in windings of wind turbines

Marcos A. Araujo Ferreira Junior. *‡, Luís Fabrício de F. Souza*†¶, Francisco Hércules dos S. Silva *‡, Elene Firmeza Ohata . *¶ Jefferson Silva Almeida*¶ and Pedro P. Rebouças Filho*¶

* Laboratório de Processamento de Imagens e Simulação Computacional (LAPISCO)
† Programa de Pós-Graduação em Engenharia de Teleinformática (PPGETI),
‡ Federal Institute of Education, Science and Technology Ceará (IFCE), Ceará, Brazil,
¶ Universidade Federal do Ceará (UFC), Fortaleza, Ceará, Brazil,
Email: {marcos.aurelio, fabricio.freitas, herculessilva, elene.ohata}@lapisco.ifce.br,
pedrosarf@ifce.edu.br.

***Abstract*—With the parameters set of the industry 4.0 and the growth of new intelligent and interconnected systems, those concepts have enabled innovative advances in several areas, among them, solutions for different renewable sources of energy efficiency. This study has as its objective the detection of short-circuit faults in wind turbines utilizing an analysis of vibration images. Using the Internet of things (IoT) context, we created a methodology to check the operating condition of a machine. The proposed method obtained excellent results, presenting a new interconnected approach to the industry 4.0 for short-circuit detection of induction generator squirrel-cage model, a widely used and growing model in the renewable energy market. Using the Random Forest-LBP combination, we achieved 87.9% accuracy with no false positives between the normal class and the failure classes.**

***Index Terms*—Intelligent IoT, Wind Turine, Fault Detection**

I. Introduction

Currently, several technologies are seeking less impact on the environment through renewable sources and energy efficiency, among them, solar energy and wind energy gain strength amid industrial effects, becoming options for on-demand solutions from various sectors around the world. Wind energy is one of the most promising and efficient clean energy generation, generating 3.86 % of power relative to world demand, this being 318.1 GW, operating in 103 countries, and [1]–[3] estimate that the growth could reach 19% by 2030.

The world is leaning toward new sustainable technologies, since, by reducing investment in fossil fuels as a result of climate problems and limited resources, it creates a conjuncture that drives new ways of generating renewable energies [4]. The generation of renewable energies had a significant increase in 22% between 2014 and 2015 [2]. Within this global perspective based on data from 2007, it was assumed that the worldwide installation would reach 240 GW by 2012 and 540 GW by 2017, with a view of 600 GW in 2019, anticipating the growth of 40 GW per year, having an estimate of 1000 GW by 2030 [5].

Even with a perspective of the expected growth of such technologies, there are several challenges in solving current problems, which, if solved, will allow guarantees for future expansion. With technological advancement within industries, coupled with new technologies such as the Internet of Things (IoT) [6], are key points for future advancement within a new perception that directly introduces the concepts fostered by the Industry 4.0 [7], with futuristic vision connected to solutions with intelligent technologies, complementing the need to change the new generation of industries. Within the current context of the wind power generation process and challenges, points that affect operational problems directly impact energy costs and maintenance costs up to 30% of the cost of energy [8], [9]. Maintenance and technical assistance in wind farms receive attention to preventive measures for optimization and effectiveness of energy generating systems [10] [9]. The study of [11] found faults through records in wind turbines, and also the conclusion that the electric generator is the most costly and most defective component, in private the Squirrel Cage Induction Generator (SCIG), robust technology with future propensities in the market in wind turbines in the coming years. [11] In view of this scenario that involves means for technological apparatus searched by the Industry 4.0 [7], we based this study on the detection of short-circuit failures, with a new approach for classification of the vibration signals generated by generators of induction type (SCIG), with an Internet view of Things (IoT) [6].

All studies works has in commom the approach of the use of signal in a dimension (1D) in the treatment for data analysis, common practice between data analysis in signals in short-circuit faults. Our study aims to identify incipient motor failures (SCIG) using machine learning with a new approach in its classification reported in the next session, arriving very significant results in the field of computer vision.

Our study brings several contributions addressing new methodologies; an innovative model applied in digital image processing (two-dimensional signal), a typical problem of signal processing (signal in one dimension). The system integrates a shared cloud tool, aiming at the popularization of new technologies applied to machine learning, allowing to intensify the use of database improving the precision of fault detection algorithms. The method was thought to use IoT concepts by aggregating different local systems to a cloud structure, this integration is an essential step in the current cultural context of industry 4.0 management, allowing greater exchange of information and experiences among professionals.

II. Methods of Literature

In this session is presented a short explanation of machine learning techniques and extraction of characteristics used to solve the proposed problem. In the characteristic extraction step, we use classic extractors and CNN models.

A. Machine learning

Multilayer Perceptron (MLP) is composed of an input layer, which can contain multiple hidden layers and an output layer [12]. It consists of a combination of perceptrons, neurons designed to solve non-linearly separable problems, using a backpropagation error technique to aid learning.

Equation 1 shows the output signal of a neuron y_j in iteration n from a weight w_{ji} that weights the output of neuron y_i The activation function φ_j, if y_j belongs to the input layer, y_i is a sample of the set of inputs.

$$y_j(n) = \varphi_j \left(\sum_{i=0}^{m} w_{ji}(n) y_i(n) \right) \tag{1}$$

The instantaneous value of the total error in the iteration n, $\varepsilon(n)$, is generated from the summation of errors of the set of neurons of the output layer, computing the difference between y_j and the value desired d_j, Equation 2.

$$\varepsilon(n) = \sum_{j \subset C} \frac{1}{2}(d_j(n) - y_j(n))^2 \tag{2}$$

The negative derivative of the total error represents the local gradient δ_j, in a neuron j, Equation 3.

$$\delta_j = -\frac{\partial \varepsilon(n)}{\partial v_j(n)} \tag{3}$$

In Backpropagation, Equation 4, the gradient δ_j in neuron j, depends on the local gradient of a neuron k on its right, the weighting between both and its sign of output yj.

$$\delta_j(n) = \varphi_j'(v_j(n)) \sum_{k}^{a} \delta_k(n) w_{kj}(n) \tag{4}$$

Naive Bayes is based on Bayes Decision Theory [13], the classifier of the supervised method is premised on using probabilistic parameters to classify the samples according to the percentage referring to the stipulated class. It is by calculating the posterior probability $P(w_i|x)$, from the a priori probability $P(w_i)$, and the probability distribution functions (PDF) $p(x|w_i)$ and e $p(x)$, as shown by Equation 5 [14].

$$P(w_i \mid x) = \frac{p(x \mid w_i) \cdot P(w_i)}{p(x)} \tag{5}$$

Equation 6 demonstrates the decision rule from the results of Equation 5 for samples of two classes a binary classification problem, for example.

$$x \rightarrow \begin{cases} P(w_1 \mid x) > P(w_2 \mid x) & belongs \quad to \quad w_1 \\ P(w_1 \mid x) > P(w_2 \mid x) & belongs \quad to \quad w_2 \end{cases} \tag{6}$$

Random Forest is an unsupervised network based on the data set method chosen based on the initial set. In its training, the algorithm improves the classification by calculating the margin of the correct classifications $h(X) = Y$, in relation to the wrong classifications $h(X) \neq Y$, as shown by Equation 7 [15].

$$mg(X,Y) = av_k I(h_k(X) = Y) -_{j \neq Y}^{max} av_k I(h_k(X) = j) \tag{7}$$

The generalization error seen in Equation 8, is given by the probability that the margin calculated in Equation 7 is less than zero.

$$PE = P_{X,Y}(mg(X,Y) < 0) \tag{8}$$

Support Vector Machines (SVM) are based on the Statistical Theory of Learning. The SVM was proposed by Vapnick [16] to define the class regions through separating hyperplane w, which best define the output y_j between 1 and -1, then the result of Equation 9 will always be equal to 1.

$$y_i[(w(x_i) + b)] = 1, \quad i = 1, ..., l. \tag{9}$$

To find the optimal separation hyperplane and consequently obtain the best classification, we must minimize the normal vector to the separation hyperplane w, as seen in Equation 10.

$$\varphi(w) = ||w||^2 \tag{10}$$

Other modified models (versions) are used to handle multiclass problems: linear SVM, SVM Polynomial, SVM RBF [17].

B. Classic Feature Extraction Methods

Local Binary Patterns (LBP) is a powerful texture descriptor [18]. Computes the texture locally by comparing pixel by neighborhood pixel. Equation 11 demonstrates the operation, in which g_p is one of the pixels P within a radius R from a central pixel, g_c.

$$LBP_{P,R} = \sum_{p=0}^{P-1} f(g_p - g_c)^{2P} \tag{11}$$

For each operation between pairs, a binary value is assigned to $f(x)$, as shown by Equation 12.

$$f(x) = \begin{cases} 1, & if \quad x \geq 0 \\ 0 & otherwise \end{cases} \tag{12}$$

The Gray Level Co-Occurrence Matrix (GLCM) [19] characterizes the texture of an image by calculating the frequency at which pairs of pixels with specific gray level values occur in an image. The Equations 13, 14 and 15 are three of fourteen attributes extracted by GLCM and, respectively, compute the second angular momentum, variance and entropy in an image.

$$\sum_i \sum_j p(i,j)^2 \tag{13}$$

$$\sum_i \sum_j (i-\mu)^2 p(i,j) \tag{14}$$

$$-\sum_i \sum_j p(i,j) log(p(i,j)) \tag{15}$$

The Hu [20] moments are calculated from the central moments, Equation 16, invariant to translation, and from invariant moments to scale, Equation 17. All seven moments display invariant characteristics of rotation, translation, and scale

$$\mu_{pq} = \sum_m^p \sum_n^q \binom{p}{m} \binom{q}{n} (-\bar{x})^{(p-m)} (-\bar{y})^{(q-n)} M_{mn} \tag{16}$$

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\left(\frac{p+q}{2}+1\right)}} \tag{17}$$

C. Convolutional Neural Networks

Oxford Visual Geometry (VGG) [21] was runner-up in the 2014 ILSVRC challenge [22]. Its architecture consists of 16 uniform convolutional layers. Convolutions Factorized was the strategy used responsible for increasing depth, without causing overfitting of the model.

Inception, also called GoogleNet [21], was the winner of the ILSVRC 2014 challenge [22]. It was developed by the Google Brain team inspired by the architecture created by LeNet5 (convolutional layers, pooling layers and fully connected layers), the network implemented a new element called the creation module. This module applies several small convolutions in order to drastically reduce the number of parameters.

ResNet was presented at ILSVRC 2015. Although the VGG talks about the correlation between the depth of the network and the accuracy, experiments performed in the article Residual Neural Network (ResNet) [23], disagree with this view, considering classic CNNs. With increasing network depth, accuracy becomes saturated and then rapidly degrades mainly due to the increasing gradient problem [24]. To address this problem ResNet authors have introduced a new building block, Residual Block.

Extreme Inception (Xception) is a CNN model that was proposed by François Chollet [25], in which is presented the Depthwise Separable Convolution layer. The model obtained good results in the JFT dataset [26], which contains 350 million images. Although Xception has the same number of parameters as Inception V3, this CNN uses the model parameters more efficiently.

MobileNet as Xception is based on a simplified architecture that uses separable convolutions in depth, introduced by Howard et al. [27]. The new architecture added two hyper-parameters (Width Multiplier, Resolution Multiplier), allowing the model to work efficiently with hardware restrictions on mobile and embedded systems.

Nasnet is a CNN model developed by Google Brain research team [28]. The authors propose a building block in a small set of data and then transfer to a large dataset, looking for the best convolutional layer. The model explores the hypothesis that it is possible to create an efficient architecture directly from the dataset.

III. Methodology

In this section, we will present how to apply the proposed methodology. First, the sensor used is introduced, then the experimental model of a wind turbine, the cloud training and classification system are described. Those procedures demonstrates that it is possible to detect and acquire data anywhere, thus being viable for industry 4.0.

The method consists of monitoring short-circuit faults by extracting characteristics collected with a three-axis sensor, specifically a MEMs accelerometer [29]. The collection of the vibration signal is made periodically in the radial direction of the electric generator. The signal is sent to an embedded system, where the signal originally from 1D becomes 2D per image plot then the image is sent to a database in the cloud. By transforming the vibration data into images, it enabled the use of extractors with specific characteristics of the images, allowing to explore characteristics that only 2D signal extractors can highlight. After the extraction step, the classification of the standard occurs and sends to the device the condition of the generator: normal, low impedance fault or high impedance fault.

In the Lapisco Image Interface Platform for Application Development (LINDA) are implemented all extraction and classification techniques used.As a cloud tool, it can be used by any device with internet access, applying industry concepts 4.0.

Figure 2 shows the steps of the methodology. Step 1, the user has the option to predict the condition of the generator or train a new network.Step 2, when choosing the option to train a new network, the LINDA tool will be used, where the steps of project definition, image upload, attribute extraction, training, and forecasting will be performed. Step 3, analysis of the results to choose the best classifier-extractor combination that satisfies the proposed problem. Step 4, the trained network is applied in an API to perform predictions with higher speed.

A. Experimental model and data acquisition.

A wind turbine simulator [11], described in Figure 1, was used. It consists of two squirrel-cage motors, full-scale and full-variable-speed. The two motors are 1CV three-phase electrically connected in delta with a 220V supply, 60Hz and rated current of 3A. A WEG-branded CFW-08 frequency converter powers the stator winding. Each engine has a predetermined function, motor 1 simulates wind speed, and motor 2 simulates a full-scale generator.

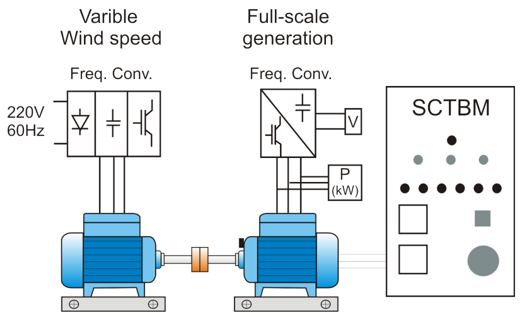


Fig. 1: Test-bench layout used to simulate short circuit faults.

The system has a board that controls the Short-Circuit Test Board in Machines (SCTBM), whose function is to simulate electrical connections of short-circuit between coils of the coiled stator. Low impedance and high impedance faults were applied, which comprise 9.26% of the total windings.

The engine that has the function of simulating the wind starts its operation with the frequency of 45Hz and evolves up to 60Hz. The acquisition of vibration data comprises the entire operating range of the generator, from 45Hz to 60Hz.

B. IoT framework

Lapisco Image Interface for Application Development (LINDA) is a web application divided into two systems. The first in Java that runs the web service and manages the sending of images between devices and workstations with the processing of the platform in a computational cloud. The second system is a database in PostgreSQL that is a free technology that has the function of storing the processing algorithms (extractors and classifiers) described in the section, as well as the images used in the training and the alternatives between the classifier-extractor pair. LINDA is a generic tool, can be employed in various problems. For this article, we chose the theme of wind turbine failure detection using vibration data.

To start new training, the user must follow the instructions. Name the project, choose the number of classes, choose the type of action (extraction or classification). Extraction, the user has nine resource descriptors available; Figure 3 shows the layout of the project settings page. Subsequently, the images are loaded and separated by classes. In Figure 4, the user can check the thumbnails of the images sent to the system. Next, the configuration is made to perform classification, allowing the possibility to choose seven classifiers described in the section II. When performing the classification, the user is allowed to verify through bar graphs, the results of the accuracy of all classifiers for a given characteristic extractor as observed in Figure 5. The platform also generates confusion matrices for each result shown in Figure and Figure 7.

Hardware used in LINDA hosting was an Intel Xeon server with 16 threads and 32Gb of memory, running the Linux Ubuntu 16.04 64-bit operating system, Java version 1.8.0 and Python version 3. Classifiers and descriptors are implemented in Python and use the TensorFlow and Keras libraries.

The platform displays a list with the API settings; consequently, the address and hash code to be used. This option requires programming knowledge if there is a desire to integrate LINDA with other platforms by creating an IoT model.

C. Feature Extraction Steps

The acquisition of vibration data is performed periodically for 10 seconds at a sampling rate of 5 kHz. Data processing occurs in an integrated system and sent to the cloud. The data sent is images with a window of 5208x833 pixels to prevent loss of information.

Dataset is organized into three classes: normal, low impedance fault and high impedance fault. In Table I we can observe the number of images present in the dataset for each class. Each class contains images of the three axes of the accelerometer. Class 0 corresponds to 248 of the X-axis, 248 of the Y-axis and 248 of the Z-axis. Class 1 corresponds to 153 images of the X-axis, 153 images of the Y-axis and 153 images of the Z-axis. Finally, Class 2 corresponds to 183 images of each axis X, Y and Z.

TABLE I: Organization of the dataset and number of images per class.

Class	Generator condition	Axis	Number of images
0	Normal	X	248
		Y	248
		Z	248
		Total	744
1	Low impedance fault	X	153
		Y	153
		Z	153
		Total	459
2	High impedance fault	X	183
		Y	183
		Z	183
		Total	549

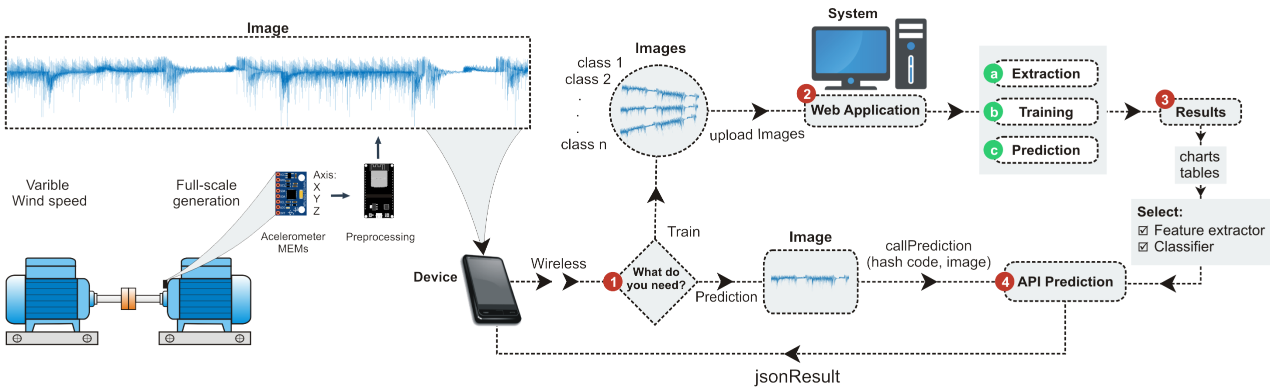
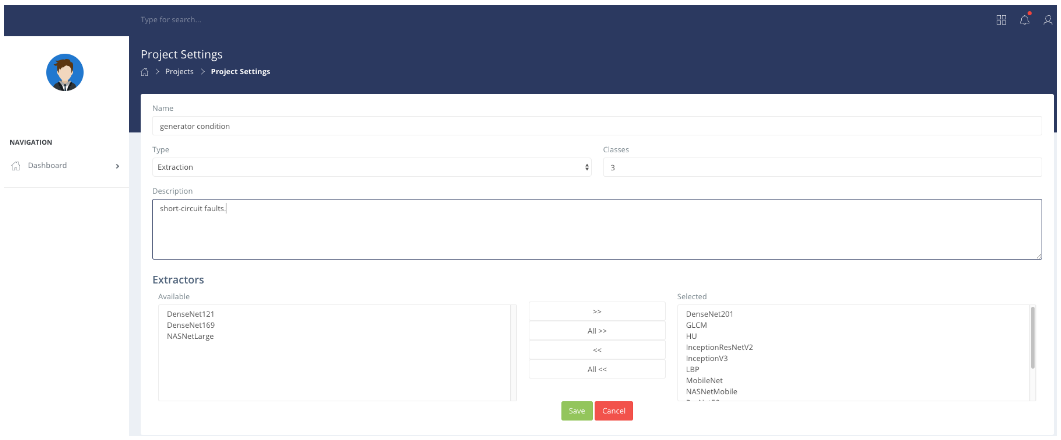


Fig. 2: Flowchart of the steps employed in our approach to classifying wind turbine conditions. A user with a device with internet access can choose to access LINDA to train a new network or use an already trained network (1). In LINDA the user uploads images separating into classes. Then, image feature extraction, network training, and prediction are performed (2). The results of various combinations of extractors and classifiers are shown (3). The best classifier-extractor combination is chosen to carry out the prediction of the problem to be solved.


Type for search...

Project Settings

Projects > Project Settings

NAVIGATION

Dashboard

Name: generator condition

Type: Extraction

Classes: 3

Description: short-circuit faults.

Extractors

Available: DenseNet121, DenseNet169, NASNetLarge

>> | All >> | << | All <<

Selected: DenseNet201, GLCM, HU, InceptionResNetV2, InceptionV3, LBP, MobileNet, NASNetMobile

Save | Cancel

Fig. 3: Creation of a project on the LINDA platform.

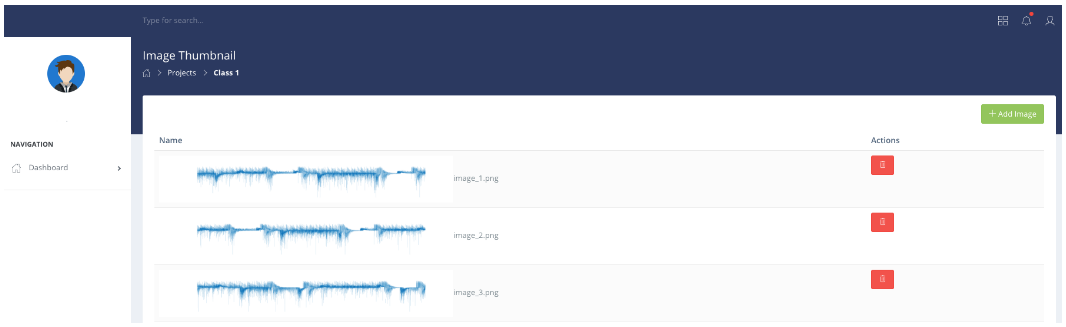

Type for search...

Image Thumbnail

Projects > Class 1

NAVIGATION

Dashboard

+ Add Image

Name		Actions
	image_1.png	
	image_2.png	
	image_3.png	

Fig. 4: Thumbnail of a class of images collected from the generator.

D. Machine Learning Methods.

In order to achieve the best combination of extractors with classifiers, we have selected different types of classifiers to detect patterns that indicate the condition of the generator.

MLP was trained using the Levenberg-Marquardt method, with numbers of neurons ranging from 2 to 1000 in its hidden layer. The Bayesian classifier used the Gaussian density function. The random search for Random Forest (RF) was used to find: the number of characteristics for better division ranging from 1 to 10; the maximum depth is 6 or none of the trees to be used; the minimum number of samples to divide an internal node ranges from 1 to 10; the minimum number of samples per sheet also varies from 1 to 10 [30]. The SVM classifier considered the kernels linear and radial basis function (RBF) with the hyperparameters C and γ varying according to the

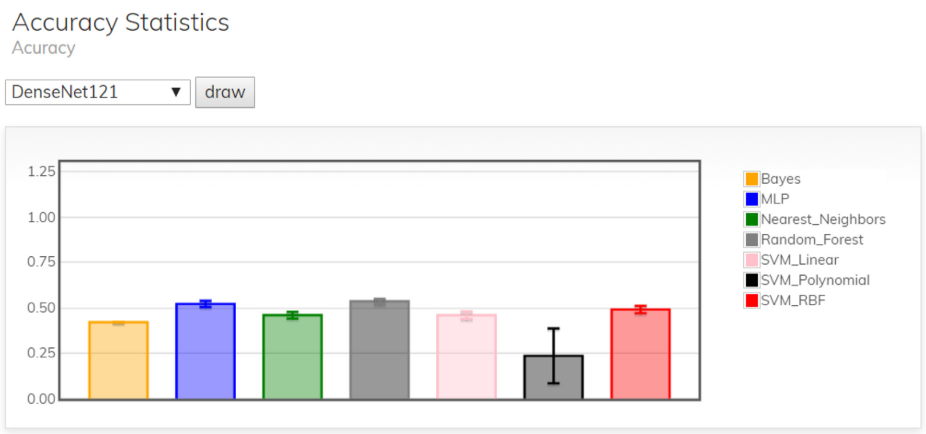


Fig. 5: LINDA Statistical Precision Bar Graph.

TABLE II: Number of attributes returned by each extractor.

Extractor	Number of attributes
LBP	48
GLCM	14
Hu Moments	7
Densenet 121	1024
Densenet 169	1664
Densenet 201	1920
InceptionResNetV2	1536
InceptionV3	2048
MobileNet	1024
NASNetLarge	4032
NASNetMobile	1056
ResNet50	2048
VGG16	512
VGG19	512
Xception	2048

values of the range 2^{-5}, 2^{-4}, 2^{-3},..., 2^{15} and 2^{-15}, 2^{-14} ,..., 2^{3}. The MLP and SVM hyperparameters were defined using 10-fold cross-validation.

E. Performance evaluation

The following metrics were used to evaluate the performance of the classifiers: Accuracy, Precision, F1-score and Recall. For a better understanding, see Figure 6, where the confusion matrix is used as a reference for calculating the metrics. The matrix is mounted with True Positive (TP) values that correspond to the accuracy number of the classifier. False Negative (FN) when erroneously classifies the class, the output was for class 2 indicates and is class 1. False Positive (FP) when classifying return failures (class 1 or 2) as normal. True Negative (TN) quantifies the number of times that the low impedance or high impedance classes are classified correctly.

Accuracy (Acc) calculates the number of true positives in relation to the amount of data. Equation 18 shows how to calculate accurately.

$$Acc = \frac{Actual\ Condition\ of\ the\ generator}{number\ of\ total\ data} \tag{18}$$

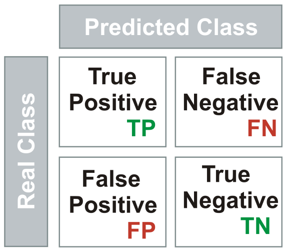


Fig. 6: Confusion Matrix.

Precision (P) is a rate that evaluates whether what was classified as normal (Class 0), was correct. Equation 19 shows how the calculation is done.

$$P = \frac{TP}{TP + FP} \tag{19}$$

Recall (R) measures how often the classifier finds the pattern of a class. The calculation can be observed in Equation 20.

$$R = \frac{TP}{TP + TN} \tag{20}$$

F1 score is the harmonic mean between Accuracy and Recall. It represents the overall efficiency of the classifier, shown in the Equation 21.

$$F1 = \frac{2PR}{P + R} \tag{21}$$

IV. Results and Discussions

This section presents the results related to the classification of the motor stage condition. The data were processed on a Core i7 at 3.6 GHz with 8 GB of RAM computer, running Ubuntu LTS 16.04.

Figure 8 presents the classification results, ranked in decreasing order of accuracy, and Table III presents the detailed data. Nonetheless, only the five top results of the combinations classifiers with feature extractors were exhibited for further analysis. Overall, all combinations obtained a good performance in accuracy, achieving at least 75%. However, GLCM is not

TABLE III: Top five results obtained by features extractors and classifiers, listed in decreasing order of accuracy.

Classifier	Extractor	Accuracy	Precision	F1-Score	Recall
RF	**LBP**	**87.67±1.99**	**83.67±2.63**	**83.55±2.62**	**83.54±2.59**
SVM RBF	DenseNet201	79.07±2.43	73.56±3.19	73.15±3.06	73.09±3.07
SVM RBF	DenseNet169	77.12±2.88	70.66±4.20	69.85±4.17	69.79±3.98
SVM Linear	DenseNet201	76.92±2.24	70.96±3.17	70.63±3.20	70.71±3.20
SVM RBF	VGG16	75.53±2.15	69.01±3.06	67.99±3.12	67.91±2.86

TABLE IV: Time from top five results showed in TABLE III.

Classifier	Extractor	Extraction time (ms)	Training time (s)	Score time (s)
RF	**LBP**	**410.000±0.000**	**23.226±0.545**	**6.779±0.352**
SVM RBF	DenseNet201	6.000±0.000	18.499±0.254	8.458±0.318
SVM RBF	DenseNet169	11.000±0.000	28.662±0.418	9.200±0.227
SVM Linear	DenseNet201	6.000±0.000	16.169±0.223	7.513±0.244
SVM RBF	VGG16	17.000±0.000	10.397±0.213	3.155±0.163

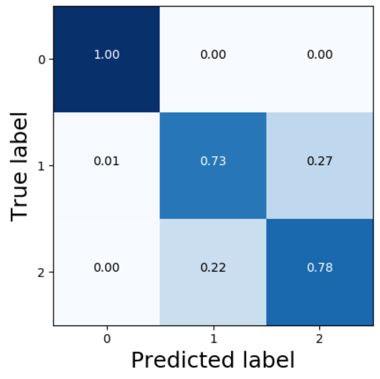
present in the top five results, as well as Naive Bayes and MLP classifiers. The combination of the Random Forest classifier and the LBP extractor achieved the best performance, reaching 87.67%, 83.67%, 83.54% and 83.55% in accuracy, precision, recall and F1-score, respectively.

The confusion matrices of the two best combinations are shown in Figure 7. In both combinations, we see that the metrics fall due to confusion between the failure classes (low and high impedance). On the other hand, the normal operating class is totally different from the fault classes. We can highlight the LBP-Forest Random combination since it reached 100 % accuracy in the classification of the normal class. The DenseNet201-SVM-RBF combination also obtained a satisfactory result, differentiating the normal class from the two fault classes with 97% accuracy. However, we detected an increase in errors between the fault classes.

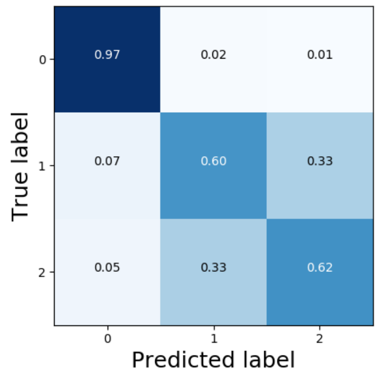
Even though there is a confusion between the fault classes, the proposed approach differentiates regular operation and fault operation, which is more significant than detecting the fault type. Once a fault is detected, the system will activate a protection mechanism, regardless of the fault type.

Because of the proposed approach is a real-time application, Table IV presents the extraction time, training time and score time of the top five results, which are essential parameters to analyze the viability of the approach.

The LBP extractor, which is a classic image feature extractor, achieved a better performance than deep extractors, specifically 8.6% higher in accuracy. However, from Table IV, LBP extractor has the highest extraction time. The deep extractors are faster because they reshape the image according to their architectures, while LBP processes the image in its full dimension, which is 5208x833 pixels.



(a) LBP - Random Forest



(b) DenseNet201 - SVM RBF

Fig. 7: Confusion matrices of the best two results after feature extraction and classification.

V. Conclusion and Future Work

We propose in this work a method of pre-classification of vibration signals with a two-dimensional signal visualization approach. Through the induction of high and low impedance short circuits, we trace the collected signals and use them for training the classifiers. Classification can be done remotely, at any place with internet access.

The validation of the classifiers demonstrated the ability of the proposed method, with the combination of RF and LBP, to distinguish, in a three-class problem, between normal vibration signals, low impedance failure, and high impedance failure, obtaining 87.67% accuracy. For a binary fault or non-failure problem, it obtained 100.00% accuracy, as shown in the confusion matrix presented in the Results section.

For future works, we can extend the application of the method to other types of signals, such as current and axial

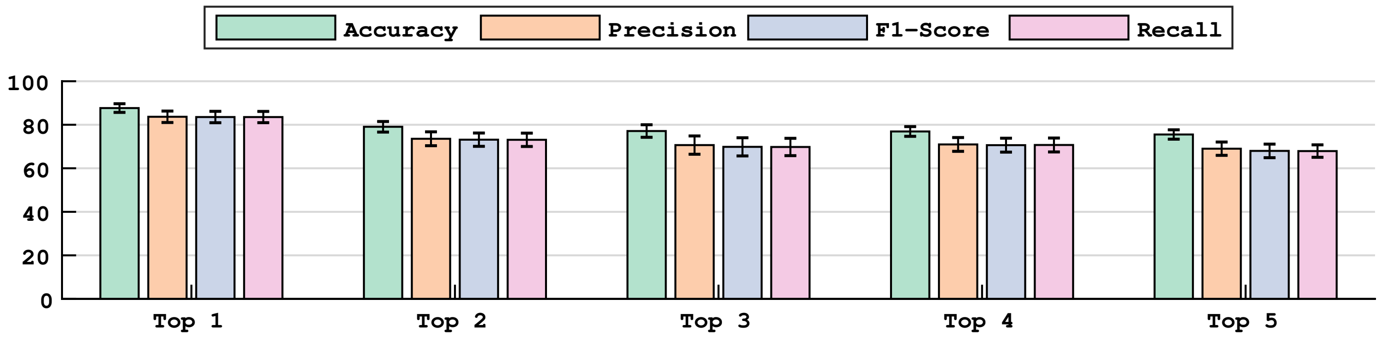


Fig. 8: A plot representation of TABLE III. Top 1 - RF + LBP, Top 2 - SVM RBF + DenseNet201, Top 3 - SVM RBF + DenseNet169, Top 4 - SVM Linear + DenseNet201, Top 5 - SVM RBF + VGG16.

flow.

Acknowledgment

The authors would like to thank The Ceará State Foundation for the Support of Scientific and Technological Development (FUNCAP) for the financial support (6945087/2019). As well as to the Federal Institute of Education, Science and Technology of Ceará (IFCE).

References

[1] P. P. Rebouças Filho, N. M. Nascimento, I. R. Sousa, C. M. Medeiros, and V. H. C. de Albuquerque, "A reliable approach for detection of incipient faults of short-circuits in induction generators using machine learning," *Computers & Electrical Engineering*, vol. 71, pp. 440–451, 2018.

[2] "Global wind energy council."

[3] Y. Lin, L. Tu, H. Liu, and W. Li, "Fault analysis of wind turbines in china," *Renewable and Sustainable Energy Reviews*, vol. 55, pp. 482–490, 2016.

[4] L. Y. Pao and K. E. Johnson, "Control of wind turbines," *IEEE Control systems magazine*, vol. 31, no. 2, pp. 44–62, 2011.

[5] V. Nelson and K. Starcher, *Wind energy: renewable energy and the environment*. CRC press, 2018.

[6] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[7] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business & information systems engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[8] Y. Kumar, J. Ringenberg, S. S. Depuru, V. K. Devabhaktuni, J. W. Lee, E. Nikolaidis, B. Andersen, and A. Afjeh, "Wind energy: Trends and enabling technologies," *Renewable and Sustainable Energy Reviews*, vol. 53, pp. 209–224, 2016.

[9] H. Polinder, J. A. Ferreira, B. B. Jensen, A. B. Abrahamsen, K. Atallah, and R. A. McMahon, "Trends in wind turbine generator systems," *IEEE Journal of emerging and selected topics in power electronics*, vol. 1, no. 3, pp. 174–185, 2013.

[10] B. Hahn, M. Durstewitz, and K. Rohrig, "Reliability of wind turbines," in *Wind energy*. Springer, 2007, pp. 329–332.

[11] V. Yaramasu, B. Wu, P. C. Sen, S. Kouro, and M. Narimani, "High-power wind energy conversion systems: State-of-the-art and emerging technologies," *Proceedings of the IEEE*, vol. 103, no. 5, pp. 740–788, 2015.

[12] S. K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets, and classification," *IEEE Transactions on neural networks*, vol. 3, no. 5, pp. 683–697, 1992.

[13] J. O. Berger, *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media, 2013.

[14] S. Theodoridis and K. Koutroumbas, "Pattern recognition–fourth edition, 2009."

[15] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[16] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[17] K.-B. Duan and S. S. Keerthi, "Which is the best multiclass svm method? an empirical study," in *International workshop on multiple classifier systems*. Springer, 2005, pp. 278–285.

[18] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[19] R. M. Haralick, K. Shanmugam *et al.*, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.

[20] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE transactions on information theory*, vol. 8, no. 2, pp. 179–187, 1962.

[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[23] S. Wu, S. Zhong, and Y. Liu, "Deep residual learning for image steganalysis," *Multimedia tools and applications*, vol. 77, no. 9, pp. 10 437–10 453, 2018.

[24] Y. Bengio, P. Simard, P. Frasconi *et al.*, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[25] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[26] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[27] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[28] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.

[29] R. S. Beebe and R. S. Beebe, *Predictive maintenance of pumps using condition monitoring*. Elsevier, 2004.

[30] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.