RIELAC, Vol. 44(3):e2303 (2023) ISSN: 1815-5928

RIELAC
Revista de Ingeniería Electrónica, Automática y Comunicaciones

Distributed architecture for fault detection in industrial equipment with improved Precision Score and Robustness Index

Yandy Pérez Ramos, Carlos Fernández-Aballí Altamirano, Julian L. Cárdenas Barrera, Francisco Herrera Fernández

ABSTRACT

Creating algorithms and systems that can process and store large amounts of data represents a great scientific, economic, and practical challenge. The application of machine learning (ML) to these problems is not trivial, and even less so if the processing of these algorithms needs to be distributed to handle the large computational load of data analysis and decision making. This paper presents a distributed and robust architecture to train, deploy, and execute distributed failure detection algorithm pipelines improving their Robustness and Precision. The solution is based on Smart Operational Realtime Bigdata Analytics (SORBA), a patented distributed architecture. The architecture combines the metrics of Robustness and Precision to automatically optimize the selection of industrial failure detection machine learning algorithm pipelines and their hyperparameters. A system of modules is developed for the acquisition, normalization, data conditioning, training, deployment, and online execution of machine learning algorithm pipelines. The solution was validated by comparing the Machine Learning (ML) results of two use cases: an industrial motor and a locomotive battery, with those achieved with Spark. The experiments showed an average improvement on the Precision Score of 28.76% and Robustness Index of 10.9%. The solution streamlines the implementation of successful applications and improves the performance of these indicators with respect to the solutions currently available in the Spark MLlib.

Keywords: Industrial failure detection, distributed architecture, Machine learning, Industrial data processing, Edge Computing.

RESUMEN

La creación de algoritmos y sistemas capaces de procesar y almacenar grandes cantidades de datos representa un gran reto científico, económico y práctico. La aplicación del aprendizaje automático (ML) a estos problemas no es trivial, y menos aún si el procesamiento de estos algoritmos necesita ser distribuido para manejar la gran carga computacional del análisis de datos y la toma de decisiones. Este trabajo presenta una arquitectura distribuida y robusta para entrenar, desplegar y ejecutar pipelines distribuidos de algoritmos de detección de fallos mejorando su Robustez y Precisión. La solución se basa en Smart Operational Realtime Bigdata Analytics (SORBA), una arquitectura distribuida patentada. La arquitectura combina las métricas de robustez y precisión para optimizar automáticamente la selección de algoritmos de aprendizaje automático de detección de fallos industriales y sus hiperparámetros. Se desarrolla un sistema de módulos para la adquisición, normalización, acondicionamiento de datos, entrenamiento, despliegue y ejecución en línea de pipelines de algoritmos de aprendizaje automático. La solución se validó comparando los resultados de Machine Learning (ML) de dos casos de uso: un motor industrial y una batería de locomotora, con los obtenidos con Spark. Los experimentos mostraron una mejora media de la

Recibido: 09/2023 Aceptado: 12/2023

puntuación de precisión del 28,76% y del índice de robustez del 10,9%. La solución agiliza la implementación de aplicaciones de éxito y mejora el rendimiento de estos indicadores con respecto a las soluciones disponibles actualmente en la MLlib de Spark.

Palabras clave: Detección de fallos industriales, Arquitectura distribuida, Machine learning, Procesamiento de datos industriales, Computación en la Nube

Título: Arquitectura distribuida para la detección de fallos en equipos industriales con mejor puntuación de precisión e índice de robustez

1. -INTRODUCCIÓN

In recent decades, strict quality standards and competitive pressure have forced industrial companies to transform their maintenance strategies and plans. These changes include the emergence of maintenance departments whose function is to ensure the employment of advanced techniques and practices to maintain the productivity of the company. The implementation of predictive maintenance strategies to increase component life, improving availability and reliability of their equipment, has an impact on the productivity of the plant [1,2], and has become an increasingly important issue for large and medium-sized companies. The Internet of Things (IoT) helps the strategies; however, it also introduces technological challenges, one of them being the necessary processing of large volumes of data. This high processing loads has led to the increasing employment of computer clusters [3,4]. Failure detection in industrial equipment is one of the interesting applications in this field [1,2].

Creating algorithms and systems that can process and store large amounts of data represents a great scientific, economic, and practical challenge. The application of machine learning (ML) to these problems is not trivial, and even less so if the processing of these algorithms needs to be distributed to handle the large computational load of data analysis and decision making [3,4].

This is added to the challenges involved in obtaining accurate and robust results in the detection of failures [5-7]. Most of the architectures developed so far are only used in the training process, and not in the online execution of algorithms [8-12]. Another important aspect is that these architectures do not contain integrated modules that measure the robustness and performance of the algorithms out of box. This negatively impacts the ability to train, deploy and predict failures quickly and effectively. There is a current trend towards the democratization of machine learning, which means making it more accessible to a wider range of organizations and enterprise customers. Right now, there is a limited number of machine learning specialists. To generalize the use of ML solutions, tools must be generated to facilitate the applications in a simple, effective, and fast way [13].

The era of distributed processing has seen the emergence of several architectures for distributed processing of large amounts of data (Bigdata) [8-12]. These distributed architectures possess a collection of independent entities that cooperate to solve a problem that cannot be solved individually. Selecting the right architecture requires quantifying applicability and performance. There are several parameters used to characterize the performance of algorithms, such as processing time, accuracy and robustness, the latter is defined as the ability to fulfill its function in the presence of certain irregularities in the data. Different measures are used to quantify this evaluation of the algorithms, usually expressed by a score that meets the indicated parameters.

Among these architectures is Spark, which has multiple benefits, such as usability, flexibility, scalability, and high fault tolerance. Spark is a distributed system that possesses properties including generality, fault tolerance, high in-memory data processing performance, and scalability. It adopts a flexible programming model using Resident Distributed Data (RDD), with a set of action and transformation operators whose operational functions can be customized by users according to their applications. It is positioned as a fast and general data processing system [9]. However, Spark has several disadvantages, among them are the limited ability to run algorithms online, limited lightweight runtime mode to process algorithms at the edge, and does not have out of the box capabilities for auto selection of algorithm pipelines based on Robustness Index and Precision Score. Spark was designed for generic use, without considering the particularities of distributed processing of machine learning algorithms [9].

Another alternative is Smart Operational Realtime Big Data Analytics (SORBA). This is a patented architecture [14] that meets the operational elements related to flexibility, scalability, fault tolerance, general deployment, and execution

mechanisms. SORBA's distributed architecture is prepared for training, deployment, and execution of machine learning algorithm pipelines (MLAP). Still, SORBA was designed without considering the specificities on the Robustness Index of the algorithms for incomplete observations and lacks an out of the box MLAP auto selection based on robustness.

The proposed architecture is based on SORBA and it builds upon it by using Precision Score and Robustness Index to optimize the self-selection, execution, and deployment of machine learning algorithms pipelines for failure detection in industrial equipment. Table 1 shows the benefits and shortcomings of Spark, SORBA and the proposed architecture for training, deployment, and execution of algorithms [9-21]. When compared to Spark, the added capabilities of the proposed architecture improve the performance of ML failure detection algorithms for industrial equipment, as it can be concluded from the three case studies presented in the paper.

2.- CHARACTERISTS OF THE PROPOSED ARCHITECTURE

The proposed architecture is based on the idea of training, executing, and deploying algorithm pipelines in a distributed way, using the resources of multiple devices. The different components of the architecture are illustrated in Figure 1 and 2. The components can be defined as follows:

- Executor: These are the processes that execute the calculation operations.
- Worker nodes: They have several executors working in parallel and can be physical servers, virtual machines or dockers.
- Cluster Handler: This component is extremely important since it is the one that controls all the exchange of messages between the worker nodes.
- Resource Handler: Regulates the use of worker nodes based on existing hardware resources.
- Manager Handler: Like the cluster handler, but in this case the program handler is responsible for distributing and properly controlling the jobs sent to the cluster.
- RESTful API: It is the interface that allows sending jobs to the cluster and the reception of results from the cluster.
- Applications: They are the ones that send the jobs to the cluster, and they are the ones that use the generated results to solve problems of various kinds, such as failure detection.

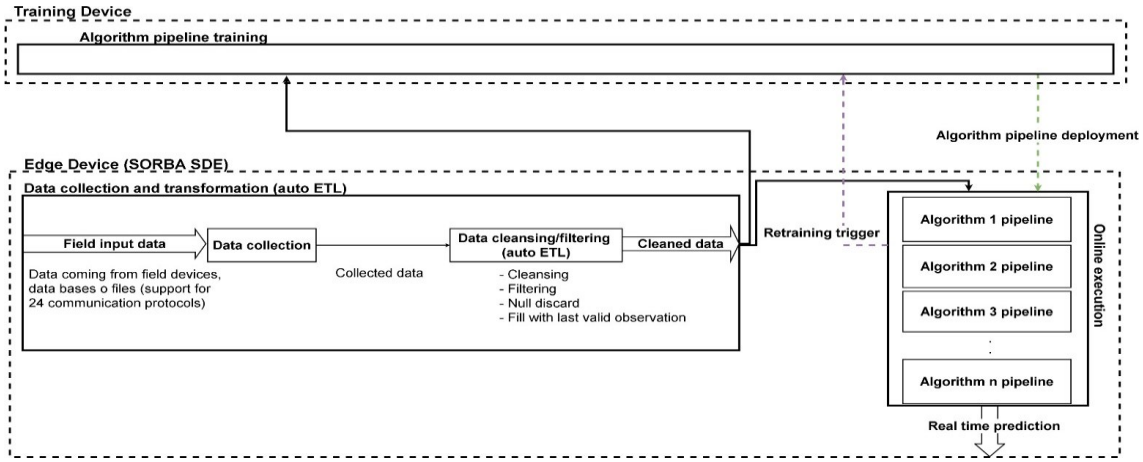

Figure 1
Online execution of distributed algorithms using edge devices.

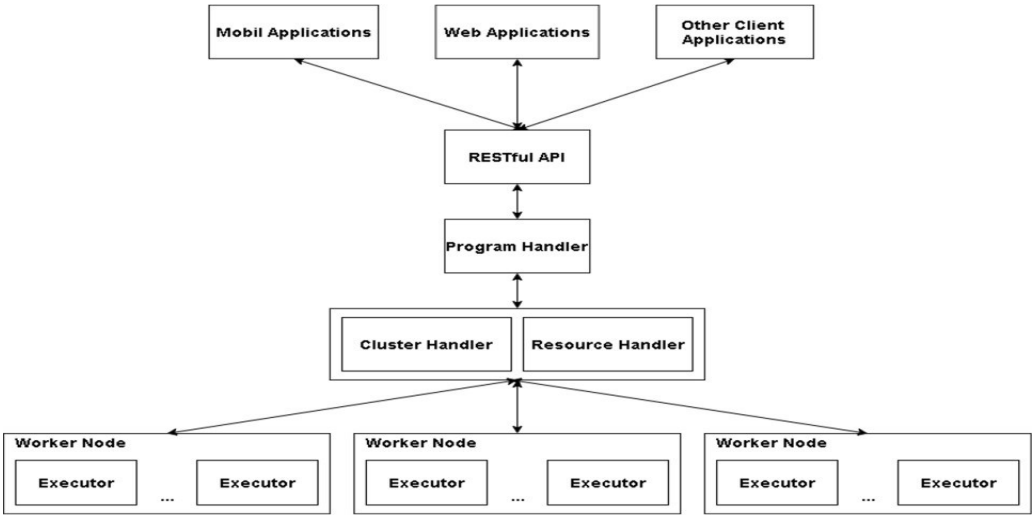

Figure 2
Proposed distributed architecture to train machine learning algorithm pipelines.

Table 1 compares the proposed architecture with Spark and SORBA. It also highlights the benefits of the proposed architecture addressed in this paper.

Table 1
Comparison between Spark, SORBA and proposed distributed architecture.

Metrics	Spark	SORBA	Proposed architecture
Performance	High efficiency	High efficiency	High efficiency
Flexibility	Yes	Yes	Yes
Scalability	Yes	Yes	Yes
Fault tolerance	Yes	Yes	Yes
Memory consumption	High	Medium	Medium
Security	Poor	Strong	Strong
Programing experience required	Yes	No	No
Machine Learning experience required	Yes	No	No
Popularity	Yes	No	No
Deploying algorithms at the edge	No	Yes	Yes
Running algorithms at the edge	No	Yes	Yes
Auto selection of algorithm pipeline based on score	No	Yes	Yes
Auto selection of algorithm pipelines based on robustness	No	No	Yes

3. METHODS

To quantify the effects of the auto MLAP selection layer added to the new architecture three industrial failure detection cases were performed applying Spark and the Proposed architecture. The results were compared in terms of Precision Score and Robustness Index [22]. Then Shapiro-Wilk's test [23] was performed to check data normality and Friedman's test [24] was used to validate the statistical significance of the results.

3.1 FAILURE DETECTION EVALUATION CASE STUDIES

Two experiments were conducted. The first experiment consisted of failure detection in an industrial motor. Industrial motors provide great capacity to intervene in different industrial processes and it is precisely what makes them vulnerable to different operating problems [5]. For this reason, their maintenance is a key and a recurring theme among professionals who intervene in the industrial field [5]. In the experiment, the industrial motor was connected to a Siemens PLC 1200 and the SDE was used to collect, clean/filter the data, and execute in real time the algorithm pipelines. The experiment used a period of 6 months of data for training, which contains 17 million observations. The industrial motor had multiple variables available including current (1), frequency (2), power (3), speed (4), torque (5) and AC voltage (6). In the experiment, several supervised algorithm pipelines were trained to classify a failure in the motor bearings.

The second experiment consisted of failure detection in a locomotive battery. Batteries are widely used in multiple industrial applications [25], especially in locomotives, where the starting system is highly dependent on them. The cost of a locomotive is approximately $3 million USD, which implies that predictive maintenance on the locomotive would help save top dollars in the long term [25]. The industrial locomotive battery used in the experiment is connected to a SDE that collected, clean/filtered the data, and executed the algorithm pipelines. A total of 38 million observations in a period of 5 months were used to train the algorithm pipelines. The variables monitored in the battery are speed (1), voltage (2) and current (3). In the experiment, several supervised algorithm pipelines are trained to classify a failure in the battery.

The following identifiers are used for each algorithm:

- SMOTE: Synthetic minority oversampling technique
- TOMEKLINKS: Tomek links for under sampling
- SMOTETOMEK: Synthetic minority oversampling technique and tomek links
- ZS: Z-Score
- SS: Standard scaling
- RS: Robust scaling
- PCA: Principal components analysis
- KPCA: Principal components analysis with kernels
- IPCA: Incremental principal components analysis
- SPCA: Sparse principal components analysis
- RPCA: Randomized principal components analysis
- LRC: Logistic regression classifier
- SVMC: Support vector machine classifier
- RFC: Random forests classifier
- DTC: Decision tree classifier
- GBC: Gradient boosting classifier
- KRC: Kernel ridge classifier
- SPP: SORBA Post-processing

The above algorithms were classified according to their functionality in the following way:

- Data balancing stage: None, SMOTE, TOMEKLINKS and SMOTETOMEK
- Pre-processing stage: None, ZS, SS and RS
- Dimensionality reduction stage: None, PCA, KPCA, IPCA, SPCA and RPCA
- Machine learning estimator stage: LRC, SVMC, RFC, DTC, GBC and KRC
- Post-processing stage: SPP

As a first step, the auto MLAP functionality of the proposed architecture was used to select the best four algorithm pipelines in terms of Precision Score and Robustness Index. Then they were tuned using the auto hyperparameter tunning available on the proposed architecture. As a result of this step the four best algorithm pipelines are:

- SMOTE-PCA-RFC-SPP
- SMOTE-PCA-DTC-SPP
- SMOTE-KPCA-RFC-SPP
- SMOTE-ZS-PCA-RFC-SPP

As a second step, the same four algorithm pipelines that the proposed architecture found were trained using Spark and Grid Search for the hyperparameter optimization.

A comparison of the four algorithm pipelines is made for the training process using Spark and the proposed architecture, using the Precision Score and Robustness Index. The training using Spark and the proposed architecture was performed with three virtual machines that form a cluster, each node has 4 CPUs, 8GB of RAM and 256GB of storage.

To simplify the naming of the algorithm pipelines a short naming convention is used on the following sections:

- SMOTE-PCA-RFC-SPP trained using the proposed architecture (Alg 1)
- SMOTE-PCA-DTC-SPP trained using the proposed architecture (Alg 2)
- SMOTE-KPCA-RFC-SPP trained using the proposed architecture (Alg 3)
- SMOTE-ZS-PCA-RFC-SPP trained using the proposed architecture (Alg 4)
- SMOTE-PCA-RFC-SPP trained using Spark (Alg 5)
- SMOTE-PCA-DTC-SPP trained using Spark (Alg 6)
- SMOTE-KPCA-RFC-SPP trained using Spark (Alg 7)
- SMOTE-ZS-PCA-RFC-SPP trained using Spark (Alg 8)

3.2 PERFORMANCE INDICES

There are many performance metrics that are used on ML such as precision, accuracy, and f1 score [26,27]. In this paper we use Precision Score and Robustness Index based on incomplete observations, which are typically used for supervised distributed algorithms pipelines. The Precision Score most of the time is used on supervised training where the features and target observations are available [27]. The Precision and Accuracy Scores are calculated using equations 1 and 2 respectively:

$$Precision = \frac{TP}{(TP + FP)} \tag{1}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

Where TP is defined as true positive, FP false positive, TN true negative, and FN false negative. The Robustness Index based on incomplete observations is introduced to quantify the effect on accuracy of the lack of observations in machine learning algorithms [22]. It is defined by equation 3:

$$Robustness = \frac{Accuracy^{40\%}}{Accuracy^{0\%}} \tag{3}$$

Where $Accuracy^{0\%}$ and $Accuracy^{40\%}$ are the accuracies obtained during the validation process with the test data group, when all the data is available and when 40% the data is missing.

3.3 VALIDATING THE PERFORMANCE EVALUATION

As part of the experiments and performance evaluation, the Shapiro-Wilk normality test was performed to determine if the Precision Score and Robustness Index are normally distributed [23]. The hypothesis is:

- H_0: The Precision Score and Robustness Index are not normally distributed.
- H_a: The Precision Score and Robustness Index are normally distributed.

When the p-value is greater than 0.05 the null hypothesis cannot be rejected, and it is concluded that the Precision Score is normally distributed. Then the experimental results regarding the Precision Score and Robustness Index were validated using Friedman test (FT) [24]. The validation assumes a general hypothesis that the Precision Score and Robustness Index for all supervised algorithm pipelines are the same:

- H_0: The Precision Score and Robustness Index for all supervised algorithm pipelines are the same.
- H_a: The Precision Score and Robustness Index for all supervised algorithm pipelines are not the same.

Then considering α = 0.05, FT can be calculated using the Equation 4:

$$F_T = \left[\frac{12}{b(k)(k+1)}\sum_{j=1}^{k} T_j^2\right] - 3b(k+1) \quad (4)$$

Where k is the number of algorithms, b is number of experiments and T^2 the sum squared of the ranks of the algorithm. If the critical value is within a given range the null hypothesis can be rejected, and it can be concluded that the Precision Score and Robustness Index of the supervised algorithm pipelines are not statistically equal.

4. RESULTS

The following section presents the results for the Industrial motor and Locomotive battery experiments. Four different MLAPs optimized and trained using Spark and the proposed architecture are applied to each case. The results indicate that the MLAP auto- selection and hyperparameter optimization based on Precision Score and Robustness Index improves the failure detection performance.

4.1 INDUSTRIAL MOTOR FAILURE DETECTION

Table 2 shows the comparison of the supervised algorithm pipelines trained using Spark and the proposed architecture for the bearing failure detection on the industrial motor. The Precision Score and Robustness Index based on incomplete observations were used for the comparison.

Table 2
Comparison of the supervised algorithm pipelines trained using Spark and the proposed architecture for failure detection in an industrial motor.

Architecture	Algorithm pipelines	Precision Score	Robustness Index
Proposed Architecture	Alg 1	0.848	0.728
	Alg 2	0.825	0.722
	Alg 3	0.983	0.723
	Alg 4	0.931	0.725
Spark	Alg 5	0.561	0.580
	Alg 6	0.572	0.565
	Alg 7	0.647	0.593
	Alg 8	0.673	0.528

Figures 3 and 4 show that industrial motor failure worsened over time. Alg 3 provided the best failure detection results, predicting the failure 6 days in advance as can be seen in Figure 4. The nature of the failure is multiplicative, it was

sudden in various parameters of the equipment and is related to a structural failure. It can be classified as a multiple failure situation because various elements of the equipment were affected. It can also be classified as an abrupt failure because it evolved rapidly. According to expert reports, the reason for the failure was due to the motor bearing and it was related to the lack of lubrication in the bearing. This condition is very common, critical and affects many mechanical equipment, causing rapid deterioration of the mechanical parts of the system.

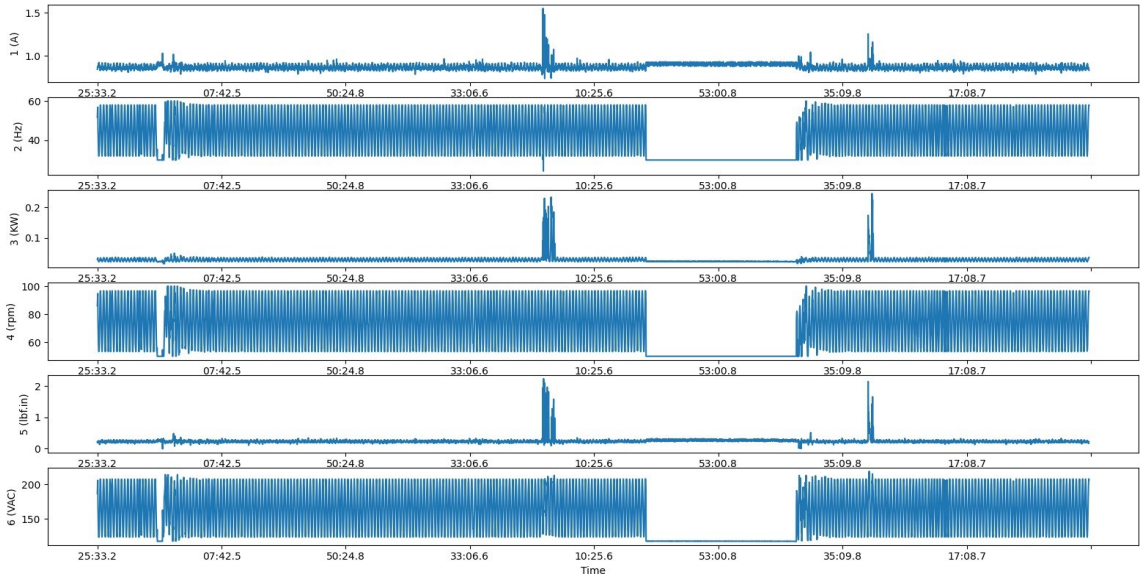

Figure 3
Variables collected in the industrial motor.

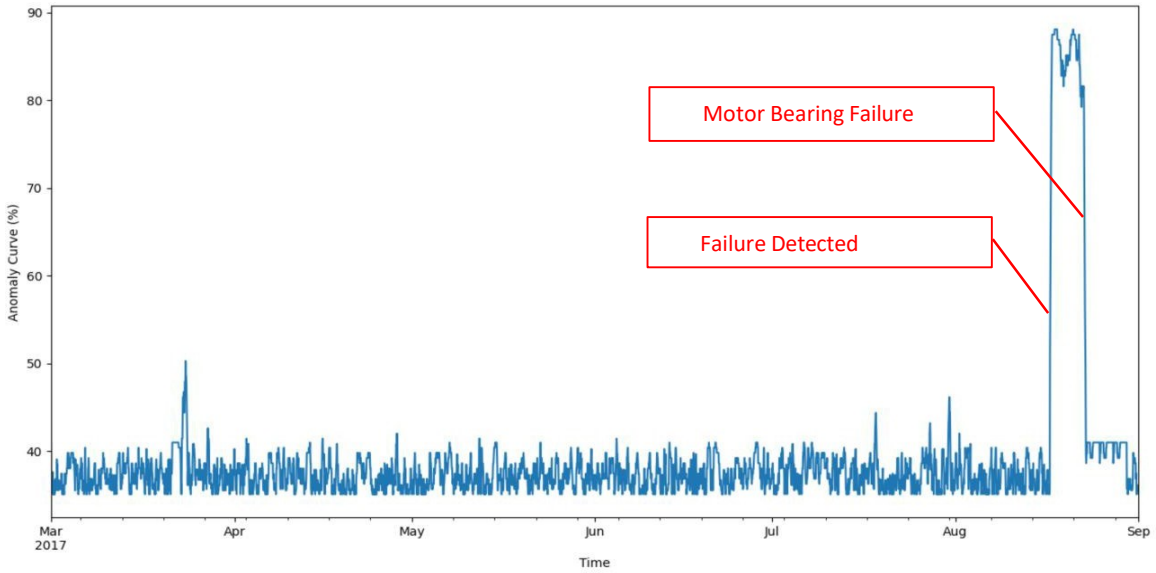

Figure 4
Anomaly curve of the motor bearing failure for the best supervised algorithm pipeline (Alg3).

4.2 LOCOMOTIVE BATTERY FAILURE DETECTION

Table 3 shows a comparison of the supervised algorithm pipelines trained using Spark and the proposed architecture for the locomotive battery failure detection. The Precision Score and Robustness Index based on incomplete observations were used for the comparison.

Table 3

Comparison of the supervised algorithm pipelines trained using Spark and the proposed architecture for failure detection in a locomotive battery.

Architecture	Algorithm pipelines	Precision Score	Robustness Index
Proposed Architecture	Alg 1	0.894	0.726
	Alg 2	0.916	0.717
	Alg 3	0.871	0.699
	Alg 4	0.824	0.696
Spark	Alg 5	0.656	0.663
	Alg 6	0.673	0.655
	Alg 7	0.531	0.625
	Alg 8	0.519	0.576

Figures 5 and 6 show that the failure worsens over time. Alg 2 provided the best failure detection results, predicting the failure 3 days in advance as can be seen in Figure 6. The nature of this failure is of a multiplicative type, it was a gradual failure in various parameters of the equipment and is related to a structural failure. It was a multiple failure type because various elements of the system were affected. The reason for the failure is related to the disuse of the battery according to the criteria of the engineers who participated in the discovery and troubleshooting of this event. As in the other cases, this failure is very common and affects many systems that use batteries as secondary power or starting systems.

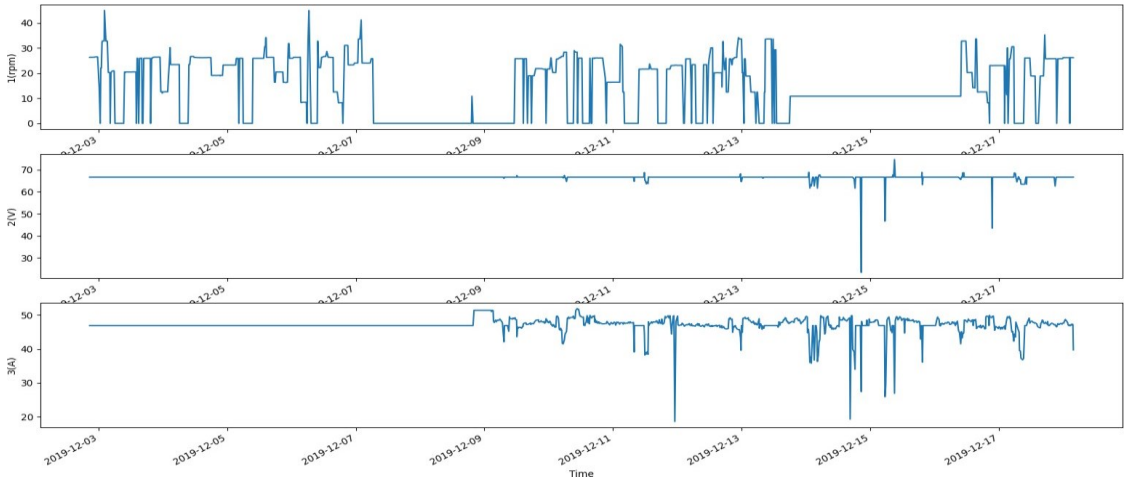

Figure 5
Variables collected on the locomotive battery.

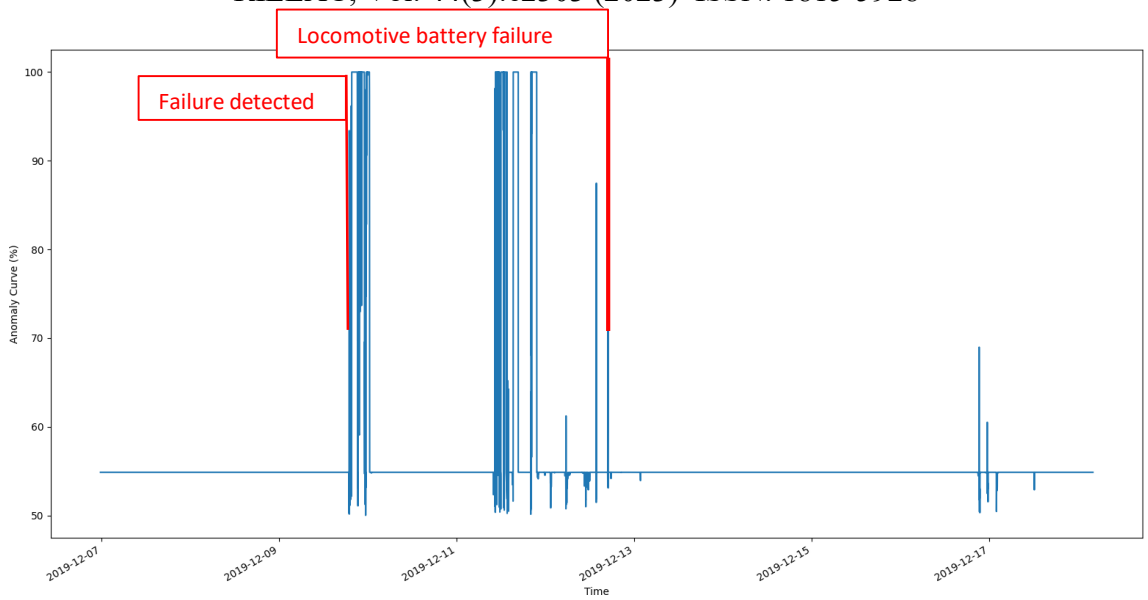

Figure 6
Anomaly curve of the locomotive battery failure for the best supervised algorithm pipeline (Alg2).

5. ANALYSIS

The objective of the study was to determine if the MLAP auto selection and hyperparameter optimization capabilities based on Precision Score and Robustness Index improved the results achievable with Spark and Grid Search. The analysis allows concluding that the proposed architecture has superior performance.

5.1 PERFORMANCE EVALUATION

Firstly Shapiro-Wilk normality test was performed to determine if the Precision Score and the Robustness Index obtained are normally distributed. The Precision Scores p-value were 0.1575 and 0.1575, and for the Robustness Index they were 0.6889 and 0.6889. In both cases they were greater than 0.05, therefore the null hypothesis could not be rejected, allowing us to conclude that they were normally distributed.

Then the results obtained for the Precision Scores and Robustness Index were validated using the Friedman test. The analysis is presented in Table 4 and Table 5 respectively. In Friedman test as a general hypothesis is that results for all supervised algorithm pipelines are the same:

- H_0: The Precision Score for all supervised algorithm pipelines is the same.
- H_a: The Precision Score for all supervised algorithm pipelines is not the same.

Table 4
Precision Score data used in the Friedman test for supervised algorithm pipelines.

Precision Score	Alg 1	Alg 2	Alg 3	Alg 4	Alg 5	Alg 6	Alg 7	Alg 8
Industrial motor	0.848	0.825	0.983	0.931	0.561	0.572	0.647	0.673
Wastewater blower	0.891	0.843	0.852	0.649	0.512	0.526	0.493	0.513
Locomotive battery	0.894	0.916	0.871	0.824	0.656	0.673	0.531	0.519

Rank	Alg 1	Alg 2	Alg 3	Alg 4	Alg 5	Alg 6	Alg 7	Alg 8
Industrial motor	6	5	8	7	1	2	3	4
Wastewater blower	8	6	7	5	2	4	1	3
Locomotive battery	7	8	6	5	3	4	2	1
$\sum$ Rank	21	19	21	17	6	10	6	8
$\sum$ Rank^2	441	361	441	289	36	100	36	64

Considering α = 0.05, *FT* for the scores can be calculated using the following equation:

$$F_T = \left[\frac{12}{b(k)(k+1)}\sum_{j=1}^{k} T_j^2\right] - 3b(k+1) = 17.222 \quad (5)$$

Where k is 8, b is 3 and $T2$ the sum squared of the rank column. The critical value is 14.067 and 17.222 > = 14.067, therefore, the null hypothesis can be rejected and conclude that the Precision Score of the supervised algorithm pipelines is not statistically equal. We can then conclude that the supervised distributed algorithm pipelines trained using the proposed architecture present an improvement in the Precision Score.

In the case of the Robustness Index a test was performed to determine the impact of incomplete observations on the accuracy obtained for the algorithm pipelines on the three experiments [22]. Several trainings were performed using a portion of the data, where 0%, 10%, 20%, 30%, and 40% of the total available observations were removed. The algorithm pipelines that were trained using the proposed distributed architecture show less reduction in Accuracy when there are incomplete observations, see Figure 7. Table 5 shows the results for the Robustness Index Friedman test. Considering α = 0.05, *FT* was calculated using the following equation:

$$F_T = \left[\frac{12}{b(k)(k+1)}\sum_{j=1}^{k} T_j^2\right] - 3b(k+1) = 15.111 \quad (6)$$

Where k is 8, b is 3 and $T2$ the sum squared of the rank column.

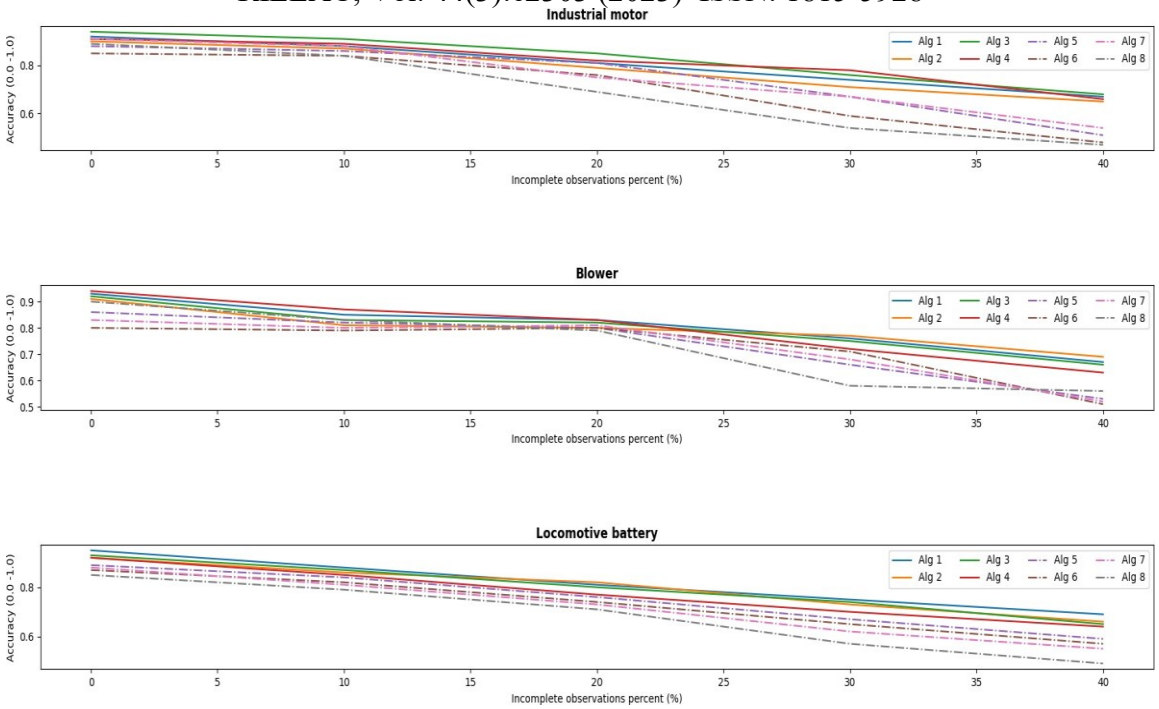

Figure 7: Comparison between algorithm pipelines reduction in accuracy when there are incomplete observations.

Table 5
Robustness data used in the Friedman test for supervised algorithm pipelines.

Robustness Index	Alg 1	Alg 2	Alg 3	Alg 4	Alg 5	Alg 6	Alg 7	Alg 8
Industrial motor	0.728	0.722	0.723	0.725	0.580	0.565	0.593	0.528
Wastewater blower	0.720	0.758	0.717	0.670	0.616	0.638	0.627	0.622
Locomotive battery	0.726	0.717	0.699	0.696	0.663	0.655	0.625	0.576

Rank	Alg 1	Alg 2	Alg 3	Alg 4	Alg 5	Alg 6	Alg 7	Alg 8
Industrial motor	8	5	6	7	3	2	4	1
Wastewater blower	6	7	5	4	2	8	3	1
Locomotive battery	8	7	6	5	4	3	2	1
$\sum$ Rank	22	19	17	16	9	13	9	3
$\sum$ Rank^2	484	361	289	256	81	169	81	9

From Equation 6 the critical values are 14.067 and 15.111 >= 14.067, therefore, the null hypothesis could be rejected, making it possible to conclude that the Robustness Indexes of the supervised algorithm pipelines are not statistically equal. Then we can conclude that the supervised distributed algorithm pipelines trained using the proposed architecture present an improvement in the Robustness Index based on incomplete observations.

6. CONCLUSIONS

In this paper, a novel distributed and robust architecture for failure detection in industrial equipment has been presented. The incorporation of the Robustness Index, together with Precision Score, in the optimization process for automatic selection of industrial failure detection machine learning algorithm pipelines and their hyperparameters, not only streamlines the implementation of solutions, but improves the performance of this indicators with respect to the solutions currently available in the MLlib Spark. Finally, the results obtained allow concluding that:

- The supervised algorithm pipelines that were trained with the proposed architecture perform better in detecting industrial equipment failures. This can be determined by the experiments performed in previous sections where an average Precision Score improvement of 28.76% and 10.94% on the Robustness Index was accomplished.
- The automatic algorithm selection, combined with the hyperparameter auto tunning mechanisms simplifies the implementation algorithms with optimal results, therefore helping democratize access to machine learning solutions when compared to MLlib Spark.
- The integrated mechanism of deployment and online execution of the algorithm pipelines helps to simplify the process of implementing applications related to the detection of failures in industrial equipment.

REFERENCES

1. Dalzochio J. et al. Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges. Computer in Industry. 2020;123*()*:1-15
2. Poor P., Ženíšek D., Basl J. Historical Overview of Maintenance Management Strategies: Development from Breakdown Maintenance to Predictive Maintenance in Accordance with Four Industrial Revolutions. International Conference on Industrial Engineering and Operations Management. Pilsen; Rep. Checa; 2019. p. 495-504
3. Thanigaivelan N.K., Nigussie E., Kanth R.K., Virtanen S., Isoaho J. Distributed internal anomaly detection system for Internet-of-Things. 13th IEEE Annual Consumer Communications Networking Conference (CCNC). Las Vegas; USA; 2016. p. 319–320.
4. Almasoud A., Al-Khalifa H., Al-salman A., Lytras M. A Framework for Enhancing Big Data Integration in Biological Domain Using Distributed Processing. Applied Sciences. 2020; 10(20)020:1-16.
5. Neupane D., Seok J., Bearing Fault Detection and Diagnosis Using Case Western Reserve University Dataset With Deep Learning Approaches: A Review. IEEE Access, 2020; 8():93155–93178.
6. Gangsar P., Tiwari R. Signal based condition monitoring techniques for fault detection and diagnosis of induction motors: A state-of-the-art review. Mechanical System and Signal Processing. 2020; 144():1-37.
7. Venkatasubramanian V., Rengaswamy R., Yin K., Kavuri S.N. A review of process fault detection and diagnosis: Part I: Quantitative model-based methods. Computers & . Chemical Engineering. 2002; 27(3):293–311.
8. Xu F., Qin Y., Chen L., Zhou Z., Liu F. λDNN: Achieving Predictable Distributed DNN Training with Serverless Architectures. IEEE Transactions on Computers. 2022;71(2):450-463.
9. Tang S., He B., Yu C., Li Y., Li K. A Survey on Spark Ecosystem: Big Data Processing Infrastructure, Machine Learning, and Applications. IEEE Transactions on Knowledge and Data Engineering. 2020;4():9469-9478.
10. Torabzadehkashi M., Rezaei S., Alves V., Bagherzadeh N. CompStor: An In- storage Computation Platform for Scalable Distributed Processing. IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). Vancouver; Canadá; 2018. p. 1260–1267.
11. Sparks E.R. et al. MLI: An API for Distributed Machine Learning. IEEE 13th International Conference on Data Mining. Texas; USA; 2013. p. 1187–1192.
12. Samiei S., Joodaki M., Ghadiri N. A Scalable Pattern Mining Method Using Apache Spark Platform. 7th International Conference on Web Research (ICWR). Teheran; Irán; 2021. p. 114–118.
13. Finley T.K. Democratization of Artificial Intelligence: One Library's Approach. Information Technologies and Libraries. 2019;38(1):8-13
14. Perez-Ramos Y., Ferrante A. Method and system for developing an anomaly detector for detecting an anomaly parameter on network terminals in a distributed network. 2017. USA Patent Number: 10218722

15. Misra P., Yadav A.S. Improving the Classification Accuracy using Recursive Feature Elimination with Cross-Validation. International Journal of Emerging Tecnologies. 2020;11(3):659–665.
16. Abas M.A.H. Ismail N., Ali N.A., Tajuddin S., Tahir N. Agarwood Oil Quality Classification using Support Vector Classifier and Grid Search Cross Validation Hyperparameter Tuning. International Journal of Emerging Trends in Engineering Research. 2020;8(6):2551–2556.
17. Siriwardhana Y., Porambage P., Liyanage M., Ylianttila M. A Survey on Mobile Augmented Reality With 5G Mobile Edge Computing: Architectures, Applications, and Technical Aspects. IEEE Communications Surveys and Tutorials. 2021;23(2):1160–1192.
18. Lv Z., Chen D., Lou R., Wang Q. Intelligent edge computing based on machine learning for smart city. Future Generation Computer Systems. 2020;115():90–99.
19. Grasso M., Colosimo B.M., Semeraro Q., Pacella M. A Comparison Study of Distribution-Free Multivariate SPC Methods for Multimode Data," Quality Reliability Engineering International. 2015;31(1):75–96.
20. Bersimis S., Psarakis S., Panaretos J. Multivariate statistical process control charts: an overview. Quality Reliability Engineering International. 2007;23(5):517-543.
21. Choi S.W., Martin E.B., Morris A.J., Lee I.B. Adaptive Multivariate Statistical Process Control for Monitoring Time-Varying Processes. Industrial & Engineering Chemistry Research. 2006;45(9):3108–3118.
22. Askarian M., Escudero G., Graells M., Zarghami R., Jalali-Farahani F., Mostoufi N. Fault diagnosis of chemical processes with incomplete observations: A comparative study. Computers and Chemical Engineering. 2016;84():104–116.
23. Royston P. Approximating the Shapiro-Wilk W-test for non-normality. Statistic and Computing. 1992;2(3):117–119.
24. Irigaray D., Dufrechou E., Pedemonte M., Ezzatti P., López-Vázquez C. Accelerating the Calculation of Friedman Test Tables on Many-Core Processors. In: High Performance Computing. Alemania: Springer; 2020. p. 122–135.
25. Lee S. et al. Diagnosing various failures of lithium-ion batteries using artificial neural network enhanced by likelihood mapping. Journal of Energy Storage. 2021;40():102768.
26. Syed D., Refaat S.S., Abu-Rub H. Performance Evaluation of Distributed Machine Learning for Load Forecasting in Smart Grids. 2020;():1–6.
27. Souza P.S.S., Santos Marques W., Rossi F.D., Cunha Rodrigues G., Calheiros R.N. Performance and accuracy trade-off analysis of techniques for anomaly detection in IoT sensors. International Conference on Information Networking (ICOIN). Da Nang; VietNam; 2017. p. 486–491.

CONFLICT OF INTEREST

There is no conflict of interest among the authors, or with any institution with which each is affiliated, or with any other institution.

The opinions expressed herein are solely the responsibility of the authors and do not represent the position of the Institution or institutions with which they are affiliated.

AUTHORS' CONTRIBUTIONS

Yandy Pérez Ramos: Conceptualization**,** Data curation**,** Formal analysis**,** Investigation**,** Methodology**,** Project administration**,** Resources**,** Software**,** Validation**,** Visualization**,** Writing - original draft

Carlos Fernández Aballí Altamirano : Data curation**,** Writing - original draft

Julian L. Cárdenas Barrera : Project administration**,** Supervision

Francisco Herrera Fernández : Project administration**,** Supervision**,** Visualization**,** Writing - review & editing

AUTORS

Yandy Pérez Ramos. Automation Control Engineer, Doctor of Technical Science. CEO SORBOTICS, Jacksonville, Florida, USA, yramos@sorba.ai. Interests: Artificial Intelligence, Systems Optimization, Predictive Maintenance, Advanced Process Control. ORCID: 0000-0002-0769-3773.

Carlos Fernandez-Aballi Altamirano. Mechanical Engineer, Doctor of Technical Science. Technical Consultant SORBOTICS, Jacksonville, Florida, USA, cfernandez@sorba.ai. Interests: Thermodynamic Systems, Solar Energy, Renewable Energy, Industrial Design ORCID: 0000-0002-5191-2937.

Julian L. Cárdenas Barrera. Telecommunications Engineer. Doctor of Technical Science. Associate Professor, University of New Brunswick. Canadá. jcardena@unb.ca, Electrical and Computer Engineering, NB Power Industrial Research Chair on Smart Grid Technologies. ORCID: 0000-0002-3674-3995.

Francisco Herrera Fernández. Electrical Engineer, Doctor of Technical Science. Consulting Professor, Senior Researcher, Department of Automatic Control, Universidad Central "Marta Abreu " de Las Villas. herrera@uclv.edu.cu, Interests: Application of Artificial Intelligence in Automatics. Predictive systems. Modeling and identification of systems. ORCID: 0000-0002-0774-0752.