An Approach Based on Fog Computing for Providing Reliability in IoT Data Collection: A Case Study in a Colombian Coffee Smart Farm

Ana Isabel Montoya-Munoz * and Oscar Mauricio Caicedo Rendon *

Departamento de Telemática, Universidad del Cauca, Popayán, Cauca 190002, Colombia
* Correspondence: aimontoya@unicauca.edu.co (A.I.M.-M); omcaicedo@unicauca.edu.co (O.M.C.R.)

Received: 16 September 2020; Accepted: 3 November 2020; Published: 14 December 2020

Abstract: The reliability in data collection is essential in Smart Farming supported by the Internet of Things (IoT). Several IoT and Fog-based works consider the reliability concept, but they fall short in providing a network’s edge mechanisms for detecting and replacing outliers. Making decisions based on inaccurate data can diminish the quality of crops and, consequently, lose money. This paper proposes an approach for providing reliable data collection, which focuses on outlier detection and treatment in IoT-based Smart Farming. Our proposal includes an architecture based on the continuum IoT-Fog-Cloud, which incorporates a mechanism based on Machine Learning to detect outliers and another based on interpolation for inferring data intended to replace outliers. We located the data cleaning at the Fog to Smart Farming applications functioning in the farm operate with reliable data. We evaluate our approach by carrying out a case study in a network based on the proposed architecture and deployed at a Colombian Coffee Smart Farm. Results show our mechanisms achieve high Accuracy, Precision, and Recall as well as low False Alarm Rate and Root Mean Squared Error when detecting and replacing outliers with inferred data. Considering the obtained results, we conclude that our approach provides reliable data collection in Smart Farming.

Keywords: internet of things; reliability; outliers; fog computing; Smart Farming

1. Introduction

The Internet of Things (IoT) has emerged as a suitable technology to collect and transmit data in different domains [1] ranging from agriculture [2], cities [3], and transportation [4] to smart-homes [5]. Reliability is necessary for data collection to guarantee the effectiveness of IoT-based services. Making decisions based on inaccurate data can negatively impact the quality of crops and, consequently, lead to losing money. Several failures affect IoT data collection: (i) random errors by lack of sensor reading repeatedly (e.g., scatter in the measured data), (ii) spurious reading (i.e., non-systematic reading errors) by a fake measure when some sporadic physical events happen (e.g., if a camera flash triggers when measuring light intensity); and (iii) systematic errors such as calibration, loading errors, and environmental errors [6]. These failures cause anomalies or outliers, avoiding reliability in data collection and the good decision-making of end-users (e.g., farmers).

In Smart Farming (SF), not all data go to the Cloud, nor do all applications operate in a Cloud style. Note that in developing countries the Internet connectivity is still constrained. Some data and applications must operate in the Fog Tier. For instance, SF applications can process data or deliver meaningful information to trigger rain and temperature alarms for crops that are too sensitive to climate variations. Thus, SF applications need to work with reliable data in the Fog Tier. These applications would not be useful to the farmer if they operate with unreliable data since it would cause the farming stakeholders to make wrong decisions that, in turn, would affect the yield and control of crops.
In the literature, Refs. [7–14] considered the reliability in IoT and IoT-based SF applications, but they did not exploit the capabilities provided by Fog Computing (FC) for accomplishing data reliability. FC provides computation, storage, and networking resources closer to end-devices, which allows facing the limitations of IoT applications supported in the Cloud Computing [15,16]. In developing countries, the SF applications that operate directly at Cloud suffer Internet constraints, outages, or connectivity issues. Such issues will not be present if a Fog Tier is used. Refs. [17–25] managed reliability in FC-based IoT by using optimization, self-adaptability, and redundancy techniques; these investigations did not include mechanisms for outliers detection and data inference for reliability purposes. Refs. [26–28] introduced methods to detect anomalies in datasets without considering an SF environment based on FC and IoT. Ref. [29] proposed mechanisms for detecting and handling outliers by interpolation; this work did not consider FC-and-IoT-based SF.

This paper proposes an approach intended to provide data collection reliability in Things-based SF, which focuses on outlier detection and treatment. Our proposal includes a conceptual Things-Fog-Cloud based architecture that incorporates mechanisms for detecting and treating outliers. The Failure Detection Mechanism (FDM) finds outliers in datasets by Machine Learning (ML) algorithms. The Failure Recovery Mechanism (FRM) replaces the identified outliers with data inferred by using interpolation techniques. Our approach is novel because there is no approach based on ML and interpolation techniques, to the best of our knowledge, aimed to provide data reliability to Things-Fog-Cloud-based SF applications that support decision-making to farming stakeholders. We evaluate our approach by deploying the three Tiers-based architecture in a Colombian coffee smart farm. We run the FDM and FRM mechanisms at the Fog Tier over this real implementation to perform the data reliability testing. The datasets used in the case study contain real information about the coffee crop temperature in different time scales (hour, day, and month) and, further, information about the data collection sensor technologies (Intel, Omicron, and Libelium). Results show our mechanisms achieve high Accuracy, Precision, and Recall as well as low False Alarm Rate (FAR) and Root Mean Squared Error (RMSE) when detecting and replacing outliers with inferred data. Considering the obtained results, we conclude that our approach provides reliable data collection in Smart Farming to support correct decision-making.

The main contributions of this paper are:

- An approach that introduces a Things-Fog-Cloud architecture that combines ML and Interpolation techniques to intelligently and automatically provide data reliability on SF applications.
- An ML-based mechanism for outlier detection in IoT-based Smart Farming data collection.
- An interpolation-based mechanism for replacing the outliers detected with inferred data.
- An extensive evaluation of the proposed approach by a case study involving real data collected in a network based on a Things-Fog-Cloud and deployed in a Colombian Smart Coffee Farm.

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 introduces the proposed approach. Section 4 presents the evaluation of the proposed approach. In Section 5, we expose conclusions and future work.

2. Related Work

Table 1 summarizes research works on reliability, which considers the involvement of IoT, FC, SF, mechanisms for outliers detection, and mechanisms for data recovery when outliers happen.
Table 1. Related work.

<table>
<thead>
<tr>
<th>Ref</th>
<th>Description</th>
<th>Reliability Definition</th>
<th>IoT</th>
<th>FC</th>
<th>SF</th>
<th>Outlier Detection</th>
<th>Data Recovery</th>
</tr>
</thead>
<tbody>
<tr>
<td>[9]</td>
<td>A system architecture for monitoring the reliability of IoT</td>
<td>The ability of the system to prevent itself from failing by continuously introspecting its state and take decisions without human intervention</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[12]</td>
<td>A micro-climate monitoring and control system for greenhouses</td>
<td>Mean-battery life and network mean packet reliability</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[13]</td>
<td>A methodology for data cleaning to eliminate yield data errors in Winter cereals yield tracking system</td>
<td>The correct data collection in yield maps</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[14]</td>
<td>A new framework for monitoring IoT systems intelligently</td>
<td>The high ratio of theoretical value with the sensor information received correctly</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[15]</td>
<td>An adaptive and reliable undervolting scheme for WSN nodes</td>
<td>Availability over operating areas below the minimum specification of voltage</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[16]</td>
<td>A detailed framework to cater full fledged agricultural-solutions using IoT</td>
<td>No formal definition</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[17]</td>
<td>A proactive computing and task distribution scheme for ultra-reliable and low-latency fog computing networks</td>
<td>A probabilistic constraint on the maximum offloaded computing delay</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[18]</td>
<td>A reliable and privacy-preserving selective data aggregation scheme for Fog-based IoT</td>
<td>Integrity of data</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[19]</td>
<td>A framework for reliable data transmission in a healthcare IoT system supported in Fog computing</td>
<td>Fault-tolerant data transmission</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[20]</td>
<td>A framework for developing and deploying on animal welfare applications</td>
<td>Replicas of the database in the Cloud, in case of failures in the local database</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[21]</td>
<td>An IoT platform to face soilless culture needs in full recirculation greenhouses using moderately saline water</td>
<td>The ability to avoid network access failures</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[22]</td>
<td>An IoT system with fog assistance and cloud support that analyzes data generated from wearables on cows</td>
<td>Availability</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[23]</td>
<td>SmartHerd, an IoT platform solution that addresses the connectivity and animal welfare in a smart dairy farming scenario</td>
<td>Availability</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[24]</td>
<td>An IoT application that monitors the cattle in real-time and identifies lame cattle at an early stage</td>
<td>Accuracy</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[25]</td>
<td>An IoT platform for agricultural monitoring automation, and pest management image analysis</td>
<td>No formal definition</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[26]</td>
<td>A mechanism for discovering anomalies in a temperature dataset using the DBSCAN algorithm</td>
<td>The ability to detect outliers</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[27]</td>
<td>An anomaly detection framework for streaming data</td>
<td>The ability to detect outliers</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[28]</td>
<td>An in-network knowledge discovery approach for detecting outliers in sensor networks</td>
<td>Degree of truthfulness of the readings obtained from each sensor</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>[29]</td>
<td>A mechanism that handles outliers via an interpolation method based on neural networks</td>
<td>Data Quality</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

This approach: An approach for providing reliability in Fog-based IoT data collection in Smart Farming. The ability of a system to provide the correct data over a period of time.
Refs. [7–14] introduced IoT applications, mostly in SF domains. These works focus on deploying specific applications such as monitoring for greenhouses, a control system for watering crops, and a system to minimize the use of fertilizer and pesticides. Although they highlight reliability as an essential feature for IoT-based SF, they did not design their solutions from a reliability perspective. Refs. [17–19] managed reliability in FC-based IoT by using optimization, self-adaptability, and re-transmission techniques. Nevertheless, none of these works included failure data detection and data recovery mechanisms to provide reliability in SF scenarios. Refs. [20–25] introduced novel IoT-based SF applications using FC features, such as local processing, deploying control modules near the access network, minimizing the amount of data transferred to the cloud, and reducing delay. These works did not exploit the FC capabilities for data reliability improvement. Refs. [26–28] introduced mechanisms to detect anomalies in datasets. However, none of these works compared two or more methods for anomaly detection nor considered an SF environment based on FC and IoT. Ref. [29] proposed mechanisms for detecting and handling outliers by interpolation, but this work did not consider FC-and-IoT based-SF.

There is no approach based on ML and interpolation techniques, to the best of our knowledge, aimed to provide data reliability to IoT-Fog-Cloud-based SF applications that support decision-making to farming stakeholders. Our approach seeks to address the shortcomings mentioned above by introducing a Fog-based and reliability-oriented architecture that incorporates a mechanism for detecting outliers in data and another mechanism for handling them to offer reliable data. In addition, it is to highlight that our approach operated with data collected in a real SF scenario.

3. Approach for Reliable Data Collection in IoT-Based Smart Farming

This section introduces a motivation scenario, the proposed reliability-oriented and Fog-based architecture, the Failure Detection Mechanism, and the Failure Recovery Mechanism.

3.1. Motivation

Let us consider a Smart Coffee Farm as a scenario involving various IoT devices, such as network-connected weather stations and wireless sensors. These IoT devices collect significant and heterogeneous data about the environment and the coffee crop. These data are the primary input for IoT-based applications like data analysis to predict coffee production, traceability to track the coffee source, alarm to inform about variables out of normal ranges, and irrigation systems [30,31]. Since a coffee farmer makes decisions based on the information that applications provide, data supporting them must be reliable. For instance, if the weather variables’ monitoring system fails due to information loss during the data collection, the coffee production estimation may be inaccurate. This problem may affect farmers’ annual schedules, avoiding a suitable organization of resources, storage space, and recruitment. In addition, if a coffee quality variable gets outside the range of the standard parameters, such as pH levels at the fermentation phase, the coffee may not achieve exportation quality. This issue may cause the farmer to lose much money (approximately USD 1000 per hectare). Therefore, the data collection must be reliable, aiming to support the smart coffee farm proper operation.

Considering the above scenario, we argue that the Things-Fog-Cloud continuum requires new ways to ensure reliable delivery of data retrieved by IoT-based devices for guaranteeing the proper operation of applications located at the Fog and Cloud tiers. In this sense, the approach proposed in this paper is more full-fledged than the ones presented in Section 2 because it includes a reliability-oriented and Fog-based architecture that incorporates mechanisms for detecting outliers and inferring data to recovering from them in the data collection process.

3.2. Reliable Fog Computing-Based Architecture

A Fog hierarchical architecture includes three tiers called Cloud, Fog, and Things [32]. The Cloud Tier comprises one or more Data Centers for providing services that consume many computational resources (e.g., big data analysis). The Fog Tier includes Fog Nodes (FNs). An FN is any physical or
virtual entity (e.g., routers, switches, wireless access points, video surveillance cameras, controllers, and servers) that improves the interaction between the Things Tier and Cloud Tier to enhance IoT-based services, such as data collection based on wireless sensors, tracking systems, and, in general, delay-sensitive applications. The Things Tier consists of IoT-enabled devices including sensor nodes. This Tier is responsible for interacting with the environment and sending data to the Fog Tier [33].

Figure 1 presents our Fog-based and reliability-oriented architecture. Unlike the traditional Fog-based architectures [34,35], our architecture separates the Fog Tier into two layers: categorizing resources in layers depending on domain requirements leads to improve IoT reliability by properly locating management services [36]. As making decisions with reliable data are pivotal for IoT-based applications, we locate FDM and FRM mechanisms in Layer 2 FN because it is closest to the IoT devices. In this way, we avoid applications running on upper Tiers operate with inaccurate data. Note that outliers in collected data can negatively impact statistical analysis and the accuracy in estimation and forecasting models of harvest, leading to making wrong decisions and, consequently, lost money. Then, Layer 2 FN comprises several edge computing nodes that collect data from IoT devices and host our reliability-targeted mechanisms. FDM is to detect outliers (see Section 3.3), while FRM is to infer correct values for replacing the detected outliers (see Section 3.4). Layer 1 includes Fog Controllers (FCs) to coordinate the network tiers and perform preliminary data analytic because they have more processing, storage, and networking resources than Layer 2 FN.

![Figure 1. Fog hierarchical architecture.](image-url)

### 3.3. Failure Detection Mechanism

FDM aims at identifying and isolating outliers from correct data. These outliers can happen by failures, such as a sensor with a damaged battery and incorrect reading. The main benefits of detecting outliers in the Fog Tier are: characterizing normal and abnormal data for early treatment of outliers and ensuring data quality before using it in data analytics and forecasting applications. FDM receives data from sensors and identifies the data’s failures as outliers (i.e., an abnormal and extreme observation of data) using ML algorithms. FDM tags the outliers at the dataset. The dataset with these tagged outliers is the input to FRM.
FDM considers three well-known outlier detection techniques [37]: Clustering-based, Isolation-based, and Classification-based. Clustering-based approaches group similar data instances into clusters with the same behavior [38]. Data instances are identified as outliers if they do not belong to clusters or generate significantly smaller clusters than other ones. The dissimilarity measure between two data instances is the Euclidean distance. Isolation-based approaches focus on separating outliers from the rest of the data points instead of profiling normal ones [39]. Classification-based approaches learn a classification model using the set of data instances (training) and classify an unseen instance into one of the learned (normal/outlier) class (testing) [40].

For the Clustering-based approach, FDM uses the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) due to it being a well-known and established grouping algorithm that does not need the desired number of clusters as K-means does [41]. DBSCAN groups the observations of a dataset into high and low-density clusters using a simplified minimum density level estimation based on distance radius named epsilon (esp), minimum points, and thresholds for the number of neighbors [42]. DBSCAN mainly needs two parameters: min_samples, and esp. With preliminary tests, we verify that varying the parameter min_samples does not affect the outliers detection performed by FDM due to the fact that we are considering the higher density cluster, such as the one with the correct values, and the other ones, such as clusters with outliers. The eps parameter controls the maximum distance between two samples. If eps is too small, FDM cannot cluster most data. If eps is too large, FDM can merge all clusters into a single one [43].

For the Isolation-based approach, we use the Isolation Forest (IF) algorithm that poses the ability to identify anomalies (outliers in our approach) from a dataset. This algorithm performs recursive random splits on attribute values' generating trees [27] that can isolate any data point from the rest of the data. Random partitioning produces noticeably shorter paths for outliers. Hence, when a forest of random trees collectively produces shorter path lengths for particular samples, they have a high probability of being outliers [44]. For IF, the parameter of contamination defines the proportion of outliers in the dataset [44]. If the contamination is set too low than the real rate of outliers, the model will not detect them. If the parameter is set too high, the number of False Positives will increase.

For the Classification-based technique, we use the Support Vector Machine (SVM) to determine if an instance falling outside a boundary is an outlier [45]. SVM separates the data from different classes by fitting a hyperplane between them, which maximizes the separation [46]. In SVM, the parameter “nu” represents the fraction of outliers in the dataset. This parameter is analogous to the contamination parameter in IF. If nu is set up to a too low value than the actual number of outliers, the model will not detect all outliers. If it is set higher than the actual number of outliers, the model will detect false normal values as outliers.

3.4. Failure Recovery Mechanism

FRM infers data to replace outliers without losing significance in data, aiming to achieve reliability and, consequently, accurate models in IoT-based applications. Figure 2 depicts inputs and outputs in FRM. From a high-abstraction level, FRM operates as follows. It receives the dataset with the outliers tagged. Then, it removes and replaces them with data inferred by interpolation techniques. Finally, it delivers the corrected data to applications in the Things-Fog-Cloud continuum for further processing. These applications will be reliable regarding data due to the FDM and FRM operation.

FRM infers data to replace outliers by considering three algorithms: Cubic Spline, Linear, and Nearest Neighbor. Cubic Spline is an interpolation method that returns the straight line connecting data points with a polynomial function to obtain a continuous and smooth curve [47]. Linear interpolation is a curve fitting method using linear polynomials to construct new data points within the range of a discrete set of known data points [48]. Nearest Neighbor interpolation is a proximal interpolation method of multivariate interpolation in one or more dimensions used in image processing [49]. The Nearest Neighbor algorithm takes a rounded value of the expected position and finds the closest data value at the integer position [50].
4. Evaluation

We evaluate our approach as follows. Initially, we implemented and deployed the architecture introduced in Section 3.2 in a Colombian Coffee Smart Farm scenario. Then, we collected temperature datasets at the Colombian Coffee Smart Farm. After, we run the FDM and FRM at the Fog Tier of the mentioned architecture. Finally, we tested the proposed mechanisms’ effectiveness to deliver reliable data regarding Accuracy, Recall, Precision, FAR, F-Score, and RMSE.

4.1. Scenario

Figure 3 depicts the implemented and deployed architecture in a Colombian Coffee Smart Farm, introduced in Section 3.2. This network aims at providing reliable data for coffee farm services such as watering, fertilizing, harvesting, and forecasting production. The farmers make important decisions to increase their earnings based on the data collected and the information obtained by such services; the quality of historical data, such as temperature, is crucial to their Accuracy. Farmers need to trust in the information offered by services; therefore, it is pivotal to achieve high data reliability.

The Things Tier includes devices such as sensors of temperature, humidity, light, and moisture. The sensors cover three hectares of a coffee crop, the coffee processing center, and the coffee storage center. In particular, this tier comprises: (i) a weather station Smart Agriculture PRO from Libelium-Waspmote Plug and Sense! [51] with external solar panels, which supports several radio technologies (ZigBee, Wi-Fi, 4G, SigFox, LoRaWAN) and sensors for monitoring soil moisture, solar radiation, atmospheric pressure, pluviometer, air temperature, and air humidity. This station covers all the three coffee farm hectares. (ii) a Datalog X-PRO from OMICRON [52] with an Internal rechargeable battery, which supports Wi-Fi, SigFox and 3G, and sensors for soil moisture, air temperature, and air humidity. This Omicron device operates in just one of the hectares of the coffee farm. (iii) sets of new low-cost sensor-tags from INTEL that support BlueTooth Low Energy, ZigBee, and Wi-Fi sensors for temperature and humidity. A set operates in each hectare of the coffee crop. The primary communication protocol used at the Things Tier was IEEE 802.11n (i.e., IEEE 802.11ax, IEEE 802.11 b, and IEEE 802.15.4) because it is the protocol supported by all the IoT devices deployed in the case study farm. Using IEEE 802.11 is not necessary to pay for a subscription, such as with SigFox or 3G/4G.

The Fog Tier includes an FN per hectare and a Fog Controller. In this case study, a single Fog Controller and three FNs were used due to the farm extension and the few IoT devices generating data. FN collects data from the Things Tier and runs FDM and FRM for providing reliable data to the Fog Controller; each FN functions on an ESP32 module and a Raspberry Pi 3 Model B with integrated Wi-Fi and Bluetooth connectivity. FNs are separated between 80 and 100 m (approximately) from each other. The Fog Controller Node runs on a Raspberry Pi 4 Model B that manages and processes the farm information. The Fog Tier communicates with the others using a REST-based services style by
the HTTP protocol; it was necessary to use a standardized north interface that would allow simple interaction with the Cloud. This node sends aggregated information to the Private Cloud Tier that performs an in-depth analysis of the information provided by lower tiers via a Linux VPS server over Hyper-V. This server offers the following farm services: environmental variables monitoring, historic coffee production, production forecast, and IoT infrastructure management.

![Figure 3. Coffee smart farming.](image)

4.2. Coffee Smart Farming Datasets

We conduct evaluation experiments with three temperature datasets (Table 2). The first one, named “Per_Hour”, contains data collected with the Omicron device every minute and a half from 2:00 p.m. to 2:57 p.m. on 5 November 2019. The second dataset named “Per_Day” was generated from an Intel device collecting data every 5 min on 6 January 2020. The third dataset called “Per_Month” was formed by the Libelium weather station collecting data every two minutes (approximately) from 10th November to 10th December of 2019. For testing purposes, we included randomly 1%, 5%, and 10% of outliers in each of the three datasets above mentioned using the InterQuartile Range (IQR) [53], generating in total nine datasets. Three datasets “Per_Hour” with 24 instances collected by Omicron device with 1%, 5%, and 10% of outliers, respectively. Three datasets “Per_Day” with 288 instances collected by Intel device with 1%, 5%, and 10% of outliers, respectively. Three datasets “Per_Month” with 22,532 instances collected by Libelium device with 1%, 5%, and 10% of outliers, respectively. The datasets and codes used in this paper are located in [54].

<table>
<thead>
<tr>
<th>Name</th>
<th>Scale</th>
<th>Number of Instances</th>
<th>Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Per_Hour</td>
<td>Hour</td>
<td>24</td>
<td>Omicron</td>
</tr>
<tr>
<td>Per_Day</td>
<td>Day</td>
<td>288</td>
<td>Intel</td>
</tr>
<tr>
<td>Per_Month</td>
<td>Month</td>
<td>22,532</td>
<td>Libelium</td>
</tr>
</tbody>
</table>

![Table 2. Datasets for coffee smart farming.](image)
Table 3 presents the structure of the datasets mentioned above. The columns represent the features and the row their format. The first two features express the date and time in the human-readable and in Unix timestamp format, respectively. The third feature (value_temp) is the original temperature data in degrees Celsius collected by the Omicron, Intel, or Libelium devices. The fourth feature (Test_1) contains the temperature data, including the outliers randomly created.

<table>
<thead>
<tr>
<th>Date</th>
<th>Timestamp</th>
<th>Value_temp</th>
<th>Test_1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(dd/mm/yyyy hh:mm:ss x.x.)</td>
<td>(int)</td>
<td>(float)</td>
<td>(float)</td>
</tr>
</tbody>
</table>

4.3. Test Environment

We evaluate FDM and FRM in a virtual machine running a Ubuntu 64-bit operative system and using Python version 2.7.17 and R version 3.6.3 for the ML and Interpolation algorithms, respectively. In Python, we deploy a ML library by using scikit-learn [55]; in particular, DBSCAN from sklearn.cluster, IsolationForest from sklearn.ensemble, and OneClassSVM from sklearn.svm. In R [56], we deploy the following functions to interpolate: spline for Cubic Spline interpolation, approx for Linear interpolation, and loess.smooth for Nearest Neighbor interpolation. It is worth mentioning that we use the CRISP-DM methodology for the construction of the mechanisms [57]. This methodology consists of the following steps:

1. Business understanding focuses on understanding the objectives and requirements from a business perspective (i.e., provide reliable data in SF).
2. Data understanding takes care of the initial data collection and allows becoming familiar with the data (i.e., extract the datasets from the architecture deployed in our case study farm).
3. Data preparation covers all the activities necessary to build the final dataset.
4. During modeling, apply data mining techniques to our data, including the tuning of their parameters to achieve the best results (i.e., FDM and FRM construction).
5. The model’s evaluation to determine if they are useful to the business needs.
6. Deployment involves exploiting the models within a production environment (i.e., deployment of FDM and FRM in the Colombian coffee farm).

4.4. Performance Metrics

We evaluate FDM’s ability to identify outliers by using the metrics involved in the confusion matrix: Accuracy, Recall, Precision, FAR, and F-Score [58]. We use the classical metrics used by other works in the literature for evaluating ML algorithms and Interpolation techniques [28,37,59]. Table 4 presents the confusion matrix, the terms “positive” and “negative” refer to the classifier’s prediction (i.e., normal or outlier). The terms “true” and “false” refer to whether the prediction corresponds to the proper observation. The True Negative indicates the outliers detected correctly. False Positive denotes values classified as normal that were outliers. False Negative exposes outliers incorrectly identified. True Positive states the well-classified normal values. Accuracy is the proportion of normal and outlier values correctly classified among the total number of classifications (see Equation (1)). Accuracy answers the question: how many classifications of the algorithm were identified correctly? Recall refers to the percentage of total normal values classified correctly by the algorithm (see Equation (2)). Recall answers the question: how many instances of the algorithm were identified correctly? Precision is the fraction of normal values that are properly-identified among the instances classified as normal (see Equation (3)). Precision answers the question: how many instances of the algorithm were predicted correctly? F-Score is the harmonic mean of Precision and Recall (see Equation (4)). F-Score is best if there is a balance between Precision and Recall. False Alarm Rate is
the percentage of falsely detected normal values of the instances classified as outliers (see Equation (5)). FAR answers the question: how many outliers of the algorithm were identified incorrectly?

\[
\text{Accuracy} = \frac{(TP + TN)}{(TP + TN + FP + FN)}
\]

\[
\text{Recall} = \frac{TP}{(TP + FN)}
\]

\[
\text{Precision} = \frac{TP}{(TP + FP)}
\]

\[
\text{F - Score} = \frac{(2 \times \text{Recall} \times \text{Precision})}{(\text{Recall} + \text{Precision})}
\]

\[
\text{FAR} = \frac{FP}{(FP + TN)}
\]

We evaluate FRM’s ability to infer data to replace outliers by using RMSE [60]. RMSE is a method of measuring the difference between values predicted by a model and their actual values. RMSE measures the amount of error between any two datasets. In this vein, we calculate the RMSE of the original temperature datasets versus the datasets interpolated to identify the most accurate technique. The less RMSE, the better the performance of the interpolation technique.

Table 4. Confusion matrix.

<table>
<thead>
<tr>
<th>Actual outlier</th>
<th>Predicted Outlier</th>
<th>Predicted Normal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual outlier</td>
<td>True Negative (TN)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td>Actual normal</td>
<td>False Negative (FN)</td>
<td>True Positive (TP)</td>
</tr>
</tbody>
</table>

4.5. Failure Detection Evaluation—Results and Analysis

We conducted experiments with different datasets to evaluate, in terms of Accuracy, Recall, Precision, FAR, and F-Score, several candidate algorithms for realizing FDM; this mechanism tags the detected outliers and generates the tagged outliers dataset that is the FRM’s input. This evaluation aims at selecting an algorithm for carrying out FDM. In this sense, we define the best performance condition for the outliers detection algorithm: high Accuracy, high Precision, and low FAR. In particular, we vary the parameters from each algorithm described in Section 3.3 as follows: for DBSCAN, the eps (\(\epsilon\)) parameter from 0.1 to 0.8 (with min_samples set in 3); for IF, the contamination (c) parameter from 0 to 0.5; and, for SVM, the nu parameter from 0.01 to 0.1. We evaluated the algorithms using the datasets described in Table 2 aggregating to each of them 1%, 5%, and 10% of outliers.

Table 5 compares in the Per_Hour dataset the candidate algorithms for performing FDM. DBSCAN and IF obtained the same high performance in any outliers’ percentage, meaning that these algorithms can operate correctly with a small dataset and few outliers. SVM obtained the worst performance; in this dataset with 10% of outliers, this algorithm got 33% and 80% of FAR and Recall, respectively, which indicates that SVM did not find all the outliers and present many false positives. These results are because SVM ignores the spatial correlation of neighboring nodes, which makes the results of local outliers inaccurate [37].

Table 6 compares in the Per_Day dataset the candidate algorithms for carrying out FDM. All evaluated algorithms’ performance decreased in this dataset (12 times bigger than the Per_Hour dataset). The three evaluated algorithms obtained their best performance in this dataset with 1% of outliers, confirming that they function well when operating with a small dataset and few outliers. Overall, DBSCAN obtained the best results for the evaluated metrics. However, note that this algorithm got 7% and 3.5% of FAR when operating with 5% and 10% of outliers in the dataset, respectively,
meaning in the datasets mentioned, the outliers are closer to standard data (DBSCAN has problems to identify these types of outliers). SVM obtained excellent results in almost all evaluated metrics but slightly lower than DBSCAN did. In addition, FAR in SVM increased up to 6.9 when the dataset contains 10% of outliers. The IF algorithm got excellent results regarding Precision, F-Score, and FAR. However, its Accuracy and Recall are around 97% and 90% when this dataset contains 5% and 10% of outliers, respectively. This algorithm had problems with false positives.

Table 5. Comparison Per_Hour dataset.

<table>
<thead>
<tr>
<th>Outliers</th>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
<th>FAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td>DBSCAN (e = 0.3–0.8)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>IF (c = 0.04)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SVM (nu = 0.01–0.1)</td>
<td>83.33</td>
<td>82.61</td>
<td>100</td>
<td>90.48</td>
<td>0</td>
</tr>
<tr>
<td>5%</td>
<td>DBSCAN (e = 0.3–0.8)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>IF (c = 0.05–0.08)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SVM (nu = 0.09–0.1)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>10%</td>
<td>DBSCAN (e = 0.2)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>IF (c = 0.09–0.1)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SVM (nu = 0.09–0.1)</td>
<td>79.16</td>
<td>80.95</td>
<td>94.44</td>
<td>87.18</td>
<td>33.33</td>
</tr>
</tbody>
</table>

Table 6. Comparison Per_Day dataset.

<table>
<thead>
<tr>
<th>Outliers</th>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
<th>FAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td>DBSCAN (e = 0.2–0.6)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>IF (c = 0.01)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SVM (nu = 0.01)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>5%</td>
<td>DBSCAN (e = 0.2–0.6)</td>
<td>99.65</td>
<td>100</td>
<td>99.64</td>
<td>99.82</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>IF (c = 0.07)</td>
<td>97.56</td>
<td>97.45</td>
<td>100</td>
<td>98.71</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SVM (nu = 0.09)</td>
<td>95.83</td>
<td>95.62</td>
<td>100</td>
<td>97.76</td>
<td>0</td>
</tr>
<tr>
<td>10%</td>
<td>DBSCAN (e = 0.1–0.2)</td>
<td>99.65</td>
<td>100</td>
<td>99.62</td>
<td>99.81</td>
<td>3.45</td>
</tr>
<tr>
<td></td>
<td>IF (c = 0.2)</td>
<td>89.93</td>
<td>88.80</td>
<td>100</td>
<td>94.07</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SVM (nu = 0.1)</td>
<td>98.96</td>
<td>99.61</td>
<td>99.23</td>
<td>99.24</td>
<td>6.90</td>
</tr>
</tbody>
</table>

Table 7 compares in the Per_Month dataset the candidate algorithms for performing FDM. All evaluated algorithms obtained excellent results regarding Accuracy, Recall, Precision, and F-Score. Accuracy and Precision in DBSCAN decreased slight to 99.7% and 99.4% when the percentage of outliers moved from 5% and 10%, respectively. In this dataset, DBSCAN also had problems with false positives (FAR = 5.6 with 10% of outliers), indicating limitations to identify outliers closer to normal data. Regarding SVM, it is to highlight that its false positives increase when the percentage of outliers is equal to 10% (FAR = 7.09), which suggests inefficiency in classifying many outliers. FAR obtained by IF is perfect because the contamination parameter used was very close to the real percentage of outliers existing in the dataset; this is difficult to know in practice for an online operation.

Based on the results above, we consider DBSCAN as the best option for carrying out FDM. DBSCAN obtained an excellent performance for marking outliers over all the tests with a FAR lower than 6%, a perfect Recall and an Accuracy, Precision, and F-Score greater than 99% for the most extensive dataset. Furthermore, DBSCAN does not require a re-setting of the parameter eps when the outliers increase. Note that IF got the highest Precision and lowest FAR. However, the IF’s contamination parameter needs to be re-set when the outliers raise to keep high performance. The IF’s false negatives will increase significantly if the real number of outliers is lower than the contamination. In turn, the false positives of IF will increase when the actual number of outliers is higher than the contamination. This re-setting is unpractical because it is hard to know the percentage of outliers previous to the algorithm’s training.
Table 7. Comparison Per_Month dataset.

<table>
<thead>
<tr>
<th>Outliers</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
<th>FAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBSCAN (c = 0.1–0.2)</td>
<td>99.99</td>
<td>100</td>
<td>99.99</td>
<td>99.99</td>
<td>1.33</td>
</tr>
<tr>
<td>IF (c = 0.02)</td>
<td>98.98</td>
<td>98.97</td>
<td>100</td>
<td>99.48</td>
<td>0</td>
</tr>
<tr>
<td>SVM (nu = 0.02)</td>
<td>99.02</td>
<td>99.01</td>
<td>100</td>
<td>99.50</td>
<td>0</td>
</tr>
<tr>
<td>5%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBSCAN (c = 0.1)</td>
<td>99.78</td>
<td>100</td>
<td>99.77</td>
<td>99.89</td>
<td>0.44</td>
</tr>
<tr>
<td>IF (c = 0.06)</td>
<td>98.90</td>
<td>98.94</td>
<td>100</td>
<td>99.42</td>
<td>0</td>
</tr>
<tr>
<td>SVM (nu = 0.07)</td>
<td>97.89</td>
<td>97.78</td>
<td>100</td>
<td>98.88</td>
<td>0</td>
</tr>
<tr>
<td>10%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBSCAN (c = 0.1)</td>
<td>99.46</td>
<td>100</td>
<td>99.41</td>
<td>99.70</td>
<td>5.6</td>
</tr>
<tr>
<td>IF (c = 0.1)</td>
<td>99.57</td>
<td>99.52</td>
<td>100</td>
<td>99.76</td>
<td>0</td>
</tr>
<tr>
<td>SVM (nu = 0.1)</td>
<td>98.23</td>
<td>98.80</td>
<td>99.25</td>
<td>99.02</td>
<td>7.09</td>
</tr>
</tbody>
</table>

4.6. Failure Recovery Evaluation—Results and Analysis

We conducted experiments with the datasets outputted by FDM for evaluating, regarding RMSE, several interpolation techniques that allow replacing the outliers with ‘accurate’ data. Figure 4 shows that the Nearest Neighbor technique obtained the worst RMSE; Nearest Neighbor selects the nearest datum’s value without considering neighboring data values, which tends to increase noise, especially in the Per_Month dataset. From a conceptual perspective, the Cubic Spline should offer better results than the Linear [61]. Still, our experimental results indicate a lower RMSE with the Linear, which is due to the data points in our datasets are closely near (our data samples are measurements of temperature). Thus, the connection between each data point by a straight line allows for achieving high-accuracy interpolation.

5. Conclusions and Future Work

This paper introduced an FC-based architecture approach that incorporates a mechanism for detecting outliers and another for inferring data intended to replace them. The evaluation demonstrated our approach’s effectiveness in a real network deployed in a Colombian Smart Coffee Farm. For the failure detection mechanism, we selected the DBSCAN algorithm due to it presenting an excellent performance for marking outliers over all the tests with a FAR lower than 6%, a perfect Recall as well as an Accuracy, Precision, and F-Score greater than 99% for the most extensive dataset. We selected the linear interpolation for the failure recovery mechanism because it infers data with low RMSE allowing for replacing the detected outliers properly. Considering the obtained results, we concluded that the proposed approach is suitable for providing reliability in the IoT-based Smart Farming data collection process and supports the correct decision-making.

Apart from that, our proposal falls behind in terms of power consumption analysis, architectures validation, or comparison, and optimal location of nodes per layer in the Fog Tier. As future work, we intend to involve more features into the dataset, such as humidity, pressure, and light. Furthermore, we plan to improve failure detection with the ability to differentiate outliers from events of interest.
Author Contributions: Conceptualization, A.I.M.-M. and O.M.C.R.; Methodology, A.I.M.-M. and O.M.C.R.; Software, A.I.M.-M.; Validation, A.I.M.-M. and O.M.C.R.; Formal analysis, A.I.M.-M.; Investigation, A.I.M.-M.; Resources, A.I.M.-M. and O.M.C.R.; Data curation, A.I.M.-M.; Writing—original draft preparation, A.I.M.-M. and O.M.C.R.; Writing—review and editing, A.I.M.-M. and O.M.C.R.; Visualization, A.I.M.-M.; Supervision, O.M.C.R.; Project administration, O.M.C.R.; and Funding acquisition, O.M.C.R. All authors have read and agreed to the published version of the manuscript.

Acknowledgments: The authors thank Innovacción-Cauca (SGR-Colombia) under project “Alternativas Innovadoras de Agricultura Inteligente para sistemas productivos agrícolas del departamento del Cauca soportado en entornos de IoT ID 4633-Convocatoria 04C-2018 Banco de Proyectos Conjuntos UEES-Sostenibilidad”, the Telematics Engineering research Group (GIT) of the Universidad del Cauca, and TECNICAFÉ.

Funding: The research of Professor Ph.D. Oscar Mauricio Caicedo was funding by the Universidad del Cauca. The research of the Eng. Ana Isabel Montoya-Munoz was funding by the project “Alternativas Innovadoras de Agricultura Inteligente para sistemas productivos agrícolas del departamento del Cauca soportado en entornos de IoT ID 4633-Convocatoria 04C-2018 Banco de Proyectos Conjuntos UEES-Sostenibilidad” of InnovAcción Cauca.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations
The following abbreviations are used in this manuscript:

- IoT: Internet of Things
- SF: Smart Farming
- FC: Fog Computing
- FN: Fog Node
- FDM: Failure Detection Mechanism
- FRM: Failure Recovery Mechanism
- ML: Machine Learning
- FAR: False Alarm Rate
- RMSE: Root Mean Squared Error
- WSN: Wireless Sensor Network
- DBSCAN: Density-Based Spatial Clustering of Applications with Noise
- IF: Isolation Forest
- SVM: Support Vector Machine
- eps: epsilon

References


60. Chai, T.; Draxler, R.R. Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature. *Geosci. Model Dev.* **2014**, *7*, 1247–1250. [CrossRef]

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.