Experimental comparison of the diagnostic capabilities of classification and clustering algorithms for the QoS management in an autonomic IoT platform

Luis Morales, Clovis Anicet Ouedraogo, José Aguilar, Christophe Chassot, Samir Medjiah, Khalil Drira

Experimental Comparison of the Diagnostic Capabilities of Classification and Clustering Algorithms for the QoS Management in an Autonomic IoT Platform

Luis Morales[1], Clovis Anicet Ouedraogo[3], Jose Aguilar[5], Christophe Chassot[3,4], Samir Medjiah[2,4], Khalil Drira[3]

[1] Departamento de Automatización y Control Industrial, Escuela Politécnica Nacional, Quito, Ecuador
[2] Université de Toulouse, UPS, Toulouse, France
[3] CNRS-LAAS, Toulouse, France
[4] INSA, F-31400 Toulouse, France
[5] CEMISID, Universidad de Los Andes, Mérida, Venezuela
luis.moralesec@epn.edu.ec, {ouedraogo, medjiah, chassot, khalil}@ laas.fr, aguilar@ula.ve

Abstract

The IoT platforms must allow the communication between the Applications and Devices according to their non-functional requirements. One of the main non-functional requirements is the Quality of Service (QoS). In a previous work has been defined an Autonomic Internet of Things (IoT) platform for the QoS Management, based on the concept of autonomic cycle of data analysis tasks. In this platform have been defined two autonomic cycles, one based on a classification task that determines the current operational state to define the set of tasks to execute in the communication system to guarantee a given QoS. The other one is based on a clustering task that discovers the current operational state, and based on it, determines the set of tasks to be executed in the communication system. This paper analyzes the diagnostic capabilities of the system based on both approaches, using different metrics. For that, a real scenario has been considered, with simulations that have generated data to test both tasks. Each technique has different aspects to be considered for a correct QoS management in the context of IoT platforms. The classification technique can determine very well the learned operational states, but the clustering approach can carry out a more detailed description of the operational states. Additionally, due to the classification and clustering technique used, called LAMDA (Learning Algorithm for Multivariate Data Analysis), the paper analyzes the operational state profile determined by them, which is very useful in a diagnostic process.

1 Introduction

The classical components of an IoT ecosystem, according to the standards are [1][2]: Devices, Network, IoT platform, and Application. Devices are the responsible of the collection of the data, and in some case, the preprocessing of these data and the execution of specific tasks; the Applications exploit the advantage of a set of interconnected Devices in order to carry out actions in the environment; the Network allows the communication between the different components in the IoT platform, particularly, between the devices and applications; and finally, the IoT platform manages smart capabilities like the autonomy,

security, among other things, and can support heterogeneous non-functional requirements of Applications and Devices. On the one hand, the IoT platforms are based on European Telecommunications Standards Institute (ETSI) SmartM2M and OneM2M [1][2] specifications, which define that an IoT platform consists mainly of two types of entities, IoT server(s) and gateway(s). These entities are implemented as Chain of Network Functions (NFs) that achieve a connection between applications and devices. Namely, NFs are processing functions, with defined functional behaviors and interfaces (e.g. a load balancing function applied on Internet protocol (IP) packets). On the other hand, these platforms are not able to sustain Quality of Service (QoS) to IoT missions critical Applications like remote surgery [34]. As per [34], QoS in IoT is one of the critical factors, which needs research for its management and optimization. Given the limited availability of resources (i.e. storage, bandwidth, etc.), and the processing capabilities of the IoT gateways and servers, the QoS must be adjusted to fulfill the requirements of IoT applications.

In previous works, an Autonomic IoT platform [3] based on the concept of autonomic cycle of data analysis tasks [4][5], has been proposed, to improve the QoS in IoT platforms, thanks to an adequate dynamic provisioning of the IoT platform resources (i.e. gateways and servers). The Analysis of Data (AD) is a science that allows converting the considered data in knowledge, with the final goal of improving some given properties (here QoS) [4]. AD has been used in multiple areas, due to their abilities discovering useful knowledge, which is used in decision-making processes. The concept of Autonomic Cycle of AD Tasks has been proposed in [4][5], which is a type of autonomous intelligent supervision that allows reaching strategic objectives around a given problem. The autonomic cycles in [3] integrate a set of AD tasks, which autonomously and collectively work to achieve the strategic objectives pursued by these cycles. Each task interacts with the others, and has a specific role in the cycle: observing the process, analyzing and interpreting what happens in it, and making decisions about the process that allow reaching the objective for which the cycle was designed.

More generally, regarding QoS management in IoT, several studies were performed at different levels, as presented by White et al [35]. In this study, out of 162 articles considered: 129 deal with the QoS management issue in the IoT at the "physical layer", 128 at the "link layer", and 117 at the "network" level, compared to only 21 at the platform level. [35] concludes that the platform level is a main research objective. The two autonomous cycle presented in this paper are part of this level. The first autonomous cycle is based on the idea of detection of the current operational state in the IoT platform that describes its operability, and according to the detected operational state, it determines the set of tasks to guarantee a given QoS requirement. In this case, a classification model is required to learn the classes that represent the operational states; and additionally, a classification system that determines for each operational state the set of tasks to be executed to guarantee a given QoS. The second autonomous cycle is based on the discovery of the current operational state, and according to it, determine the set of tasks to be executed. In this case, it is required a clustering algorithm that in real time detects the current operational state, which can correspond to one known (an old cluster) or can be new. Then, this autonomous cycle uses the centroid of each cluster, to define in running time the set of tasks to reach the desired QoS requirements. These autonomous cycles are described in detail in [3].

In this paper, the goal is to build an efficient pattern recognition model for the diagnosis of the current state, in a context of QoS management in IoT platforms. This paper considers the two main types of pattern recognition algorithms: classification for supervised learning

and clustering for unsupervised learning. To determine the diagnostic capabilities of these two models, classification and clustering, for the QoS management in IoT context, in the paper are experimentally computed and analyzed the classical metrics of quality of both models. To evaluate the sensitivity and specificity of both models, in the paper, additionally is computed the ROC (Receiver Operating Characteristic) curve. Generally, a diagnostic method with high sensitivity is required, since each state (pattern) of the system must give a positive result during the diagnostic test when there is a degradation of the QoS performance. A diagnostic method with high specificity is also required because the interest is to detect negative results when an operational state is not included in the learned classes/labels.

This work focuses on the study of the performance of these pattern recognition models in an experimental real scenario, to determine the interest of each autonomic cycle in the context of the QoS management in an IoT platform. An evaluation of both models is done, identifying which of them represents a diagnostic method with more chances of discerning the operational states to detect the degradation of the QoS. The best model must be capable of supporting dynamically the correct diagnosis of the current situation in the IoT platform, to determine the set of tasks to deploy to guarantee a given QoS. Also, the paper analyses the profile generated by the classification and clustering models, to carry out a diagnosis of the problem. The paper uses mainly a recent extension of the LAMDA (Learning Algorithm Multivariable and Data Analysis), which allows building classification and clustering models [27], to complete this analysis.

This paper is organized as follows: the next section introduces the literature linked to this work; Section III introduces the LAMDA. Section IV presents the architecture of the IoT platform; Section V describes the experimental scenario; Section VI presents the experiments with the classification and clustering models, the analysis of the results, and particularly, how can be used the profiles given by the extension of the LAMDA technique, in diagnosis tasks.

2. Literature review

In IoT, pattern recognition algorithms have been used in several works. This paper is interested in the application of the two main types of pattern recognition algorithms: classification [6-15], and clustering [16-20] algorithms. The survey in this section makes possible, on the one hand, to clear the scope of the current use of these algorithms and, on the other hand, to motivate the application of these algorithms to sustain of non-functional requirements, such as QoS in IoT platforms.

The first group of works is about how the classification techniques have been used in IoT. A first work is [6], where the authors propose a framework to generate context information from streams of raw IoT sensor data, using artificial neural network (ANN). In this way, they propose a context recognition model to be used by an intelligent IoT platform for data acquisition, preprocessing, classification, modeling, reasoning and inference. Before building the model, the raw sensor data were pre-processed using weighted average low-pass filtering and a sliding window algorithm. Mahalle et al. [7] develop a decision theory based on object classification to provide contextual information, and present the logical framework for object classification in a context aware IoT. The proposed object classification model is useful to improve network lifetime in terms of energy consumption. This paper also presents

proof of the concept, for a context-aware access control solution for makes identity management of ubiquitous objects.

In [8] are defined the challenges of security in IoT. They present the state-of-the-art in IoT security and an approach to security detection using streaming data analytics to classify and detect security threats in their early stages. Additionally, they propose methodologies and show results about this new IoT cyber-security technique for threat detection. Hong et al. [9] propose a mechanism for the assignment of virtual networks based on the identification of the service contexts. They use statistical properties of the network traffic, such as mean packet length, mean inter-packet arrival time and standard deviation inter-packet arrival time, to identify service contexts (e.g. Video Streaming, Video Conference, File Transfer Service). They apply an incrementally add dimensions scheme to separate services until all services are identified. For example, Video Streaming and FTP shows identical statistical properties in two dimensions (MPL: Mean Packet Length, MIAT: Mean Inter-Arrival Time), however, with one more dimension (SDIAT: Standard Deviation of Inter-Arrival Time), the two services can be clearly separated. They used this approach to find out the QoS needs.

Currently, there is no a uniform way to represent, share, and understand IoT data, leading to information silos where the devices cannot work and learn together with minimal human intervention. [10] discusses the limitations of current storage and analytical solutions, points the advantages of semantic approaches for context organization, and proposes an unsupervised model to learn word categories automatically. The model was evaluated in the Miller-Charles dataset and an IoT semantic dataset extracted from an IoT platform. Siby et al. [11] proposes the idea of IoTScanner. The IoTScanner integrates a range of radios to allow local reconnaissance of existing wireless infrastructure and participating nodes. It enumerates the devices, identifies connection patterns, and provides information for technical support and home users alike. Using IoTScanner, they classify streaming IP cameras, and show that their classification approach achieves a high accuracy in an IoT setting consisting of many IoT devices. Also, they use their approach for the centralized management and control of the ubiquitous wireless communication in an IoT platform. In [12] is explored the categorization of IoT devices in terms of their functionality and capability. They show how the model can describe the context information about both IoT devices and humans, in a way that may help to the Human-IoT Ecosystem design using situation theory. That is very important, for the utilization of the augmenting reality in the context of IoT technology, to capture the real-world context and to support decision making and actions in the real world via powerful smart objects in a human-IoT ecosystem.

In [13], the authors define a context-aware service framework for IoT based on ontologies, to regulate smooth traffic flow in smart cities by analyzing real-time traffic environment. The framework uses the sensors and IoT devices in the Smart Traffic System to capture the user's preferences and context information, which can be travel time, weather conditions or the driving patterns. Additionally, they use ontology to derive higher level descriptions of traffic conditions and vehicles, from the perceptive observation of traffic information. The ontology supports Dynamic Bayesian networks to deal with time-series data and uncertainties linked to context observations that fit the definition of an intelligent IoT system. The framework uses contextual information, to assess real-time congestion situation on roads. Once the congestion situation is predicted, alternate congestion free routes are suggested. Also, in [14] is applied machine learning algorithms on network traffic data for the identification of IoT devices connected to a network. To train and evaluate the classifier, they have collected and

labeled network traffic data from nine distinct IoT devices. They used a supervised learning and trained a multi-stage meta classifier; the first stage classifies traffic as traffic generated by IoT and non-IoT devices, and in the second stage, each IoT device is associated with a specific IoT device class. In [15], they introduce a new classification model of attacks in compliance with the Open Systems Interconnection (OSI) layers. The model is used for ensuring security of data exchange in the IoT devices, with techniques to fight against these attacks.

The second group of works is about clustering techniques applied in the context of IoT. Kumar et al. [16] introduce an application-based two-layer architecture for IoT, which consists of a sensing layer and an IoT layer. Both layers are required for accomplishing IoT-based applications. The success of any IoT-based application relies on efficient communication and utilization of the devices and data acquired by the devices at both layers. The grouping of the devices helps to achieve the same, which leads to formation of cluster of devices at various levels. They propose two clustering algorithms based on heuristic and graph, respectively, which are evaluated on an IoT platform using standard parameters and compared with different approaches reported in the literature. Also, [17] proposes a Hierarchical Clustering algorithm for Dynamic and Heterogeneous IoT, where the parameters like network coverage, communication cost and power consumption are considered. It is used to analyze the energy efficient connectivity and communication in the context of IoT.

The adoption of service-oriented paradigms in the IoT open them capabilities through service interfaces, which enable other functional entities to interact with them. For that, clustering services offline, according to their similarity and matchmaking, or discovering service online in limited clusters, are necessary. [18] proposes a multidimensional model-based approach to measure the similarity between IoT services. Then, density-peaks-based clustering is employed to gather similar services together according to the result of similarity measurement. Based on the service clustering, the algorithms of dynamic service matchmaking, discovery, and replacement, are performed efficiently. Due to that the devices are going to perform tasks together for a certain scenario, collaborating between them, the semantic exchange in an energy efficient way, is a critical aspect. To address this issue, [19] proposes a clustering algorithm for collaborative processing in IoT network. The algorithm is evaluated on several IoT platforms. The parameters, like network coverage, communication cost and power consumption analysis, are evaluated.

In IoT, the objects usually generate large amounts of data, transferring the bottleneck of data processing from sensors to communication systems. Azevedo et al. [20] propose a grid-based data summarization approach, to reduce the data transference commonly required in IoT networks, by the data clustering task. They use a single uniform grid to partition the space into cells and to summarize data. That ensures the reduction of the amounts of data transferred. [21] presents a Context-aware Clustering with Hierarchical Addressing (CCHA) scheme of the things. This algorithm can be used in the scalability issue in the IoT context, and makes Identity Management (IdM) of ubiquitous things. Forming an ad-hoc network, this algorithm can be used for the interaction between these nomadic devices, to provide seamless service extend without the need of new identities to the things, addressing and IdM in the IoT. Table 1 presents a comparison of our proposed approach with the previous ones.

To further the analysis of the works in the literature, in this paper are identified the following three evaluation criteria:

- C1: Used for autonomous management
- C2: Used for QoS management
- C3: Used for managing an IoT platform

In the table presented below:

- In the Evaluation column, P is prototype, S is simulation, and x means no evaluation is conducted.
- In the criteria columns, ✓ means the criterion is met and x is the criterion is not met.

Table 1. Comparison with previous works

Work	ML Technique	Scope	Data Sources	Evaluation	Criteria		
					C1	C2	C3
[6]	Classification	Context-aware IoT service	streams of IoT sensor data	P	✓	x	x
[7]				S	✓	x	x
[13]				S	x	x	✓
[8]		Security in IoT	streaming of data analytics	S	x	x	✓
[15]				x	x	x	✓
[10]		Represent, share, and understand IoT data	IoT semantic data from IoT platforms	P	x	x	x
[11]		Model of IoT devices	Streaming of IoT data	P	x	x	x
[12]				x	x	x	✓
[14]				P	x	x	x
[16]	Clustering	Services for IoT-based applications	IoT platform data	S	x	x	✓
[17]				S	x	x	✓
[18]				P, S	x	(✓)*	✓
[19]		Manage the IoT network	IoT network data	S	x	x	✓
[20]				S	x	x	✓
Our approach	Classification and Clustering	Manage the QoS in IoT platforms	IoT platform data and IoT application data	P	✓	✓	✓

*User requirement at large including QoS

Previous works are focused on the utilization of the classification and clustering models, to solve specific problems (categorize or cluster devices, detect attacks, among other things). Additionally, none of them is in the context of QoS, to determine the degradation problems that could be occurring in an IoT platform. The main contribution of this paper is the determination of the diagnostic capability of these models (classification or clustering), to detect the current operational scenario in an IoT platform, but additionally, if they can provide useful information to diagnose the possible problems of degradation of QoS. This

work is important because it defines what of the autonomic cycles defined in [3] has the best diagnostic capability.

Additionally, according to table 1, in the majority of IoT works (C3), explicitly the QoS for IoT platform has not been studied (C2), and the autonomous behavior is considered in some works (C1). At the level of IoT data, there are some works that use prototype to generate real data (like this work). Finally, the only work that defines like its main scope the management of QoS is our approach, which additionally, is the only that mix clustering and classification approaches in order to study this problem.

3. Learning Algorithm Multivariable and Data Analysis (LAMDA)

This machine learning method has been selected for several reasons: i) it can work as a clustering and classification approach, which allows evaluating the autonomous cycles with only one technique, ii) it gives information about the descriptors of the classes or cluster that it uses, which is very important in the context of the generation of corrective actions (in this case, to improve the QoS provided by the platform), iii) we have developed improvement to this model recently, with very good performances in classification and clustering benchmarks. In this section is presented LAMDA, and its extensions, in the context of classification and clustering, that we have carried out.

LAMDA is based on fuzzy logic using the concept of the adequacy degree performing a similarity analysis among descriptors of the objects, to establish a relationship between each one and its respective class [32].

The adequacy degree of an object X to a class/cluster $C = \{C_1; C_2; \ldots; C_k; \ldots; C_m\}$ is computed non-iteratively. The vector of descriptors of an object X is:

$$X = [x_1; x_2; \ldots; x_j ; \ldots; x_n] \quad (1)$$

Where: x_j: descriptor j of the object X

These descriptors must be normalized in a range between [0,1] using (2):

$$\bar{x}_j = \frac{x_j - x_{jmin}}{x_{jmax} - x_{jmin}} \quad (2)$$

Where: x_{jmin}: minimum value of x_j, x_{jmax}: maximum value x_j and $\bar{x}_j$: normalized descriptor.

The main novelty of LAMDA with respect to other algorithms is that it can work for classification and clustering, added to the ability to create new classes/clusters if a new data X does not fit to any characteristic of a preexisting class/cluster C_k. The object X is sent to the Non-Informative Class (NIC), which is a class with the same type of distribution function than the other classes. Based on the above, LAMDA does not require having data of all the classes during the training stage, neither it requires the number of classes when working with unsupervised classification.

In the rest of this section, we present the main concepts of LAMDA (Marginal Adequacy Degree and Global Adequacy Degree), and the two LAMDA extensions used in this paper that improve its performance.

3.1. Marginal Adequacy Degree (MAD)

The MADs establish how similar a descriptor is with respect to the same descriptor in a given class [27]. For computing MAD, probability density functions are used, like fuzzy binomial function Eq (3).

$$MAD_{k,j}\left(\bar{x}_j \middle| \rho_{k,j}\right) = {\rho_{k,j}}^{\bar{x}_j}\left(1-\rho_{k,j}\right)^{\left(1-\bar{x}_j\right)} \tag{3}$$

Where: $\rho_{k,j}$ is the average value of the descriptor j that in class k, and it is calculated for the case of supervised learning using Eq. (4).

$$\rho_{k,j} = \frac{1}{n_{k,j}} \sum_{t=1}^{t=n_k,j} \bar{x}_j(t) \tag{4}$$

Where: $n_{k,j}$ is the number of data of the descriptor j belonging to class k.

For unsupervised learning, $\rho_{k,j}$ is calculated by Eq. (5).

$$\rho_{k,j}(t) = \rho_{k,j}(t-1) + \frac{\bar{x}_j(t) - \rho_{k,j}(t-1)}{n_k(t-1)+1} \tag{5}$$

$\rho_{k,j}(t)$ is the mean value of the descriptor j in the previously created cluster k and it is updated progressively each time a new element is added. $n_k(t-1)$ is the number of objects previously assigned to the cluster k.

It is considered $\rho_{NIC} = 0.5$, because with this value in the probabilistic function (Eq. (3)), the $MAD_{NIC} = 0.5$ for any value of the descriptor $\bar{x}_j$.

3.2 Global Adequacy Degree (GAD)

GADs establish the adequacy of the object to each class or cluster, which is computed mixing MADs using T-norms and S-Norms and the exigency parameter $\alpha \in [0\,,1]$, for a strict or permissible classification. GAD calculation is performed by Eq (6).

$$GAD_{k,\bar{X}}\left(MAD_{k,1}, \ldots MAD_{k,n}\right) = \alpha T\left(MAD_{k,1}, \ldots, MAD_{k,n}\right) + (1-\alpha) S\left(MAD_{k,1}, \ldots, MAD_{k,n}\right) \tag{6}$$

Where: $\bar{X}$: normalized object X, T: t-norm (intersection), and S: t-conorm (union).

The fuzzy aggregators with which the best results have been obtained in the experimentation carried out [27] is the Hammacher operator.

$$T(a,b) = \frac{ab}{p + (1-p)(a+b-ab)} \tag{7}$$

$$S(a,b) = \frac{a+b-ab-(1-p)ab}{1-(1-p)ab} \tag{8}$$

Finally, the normalized object $\bar{X}$ is assigned to the class/cluster with the maximum GAD (Eq. (9)):

$$index = \max\left(GAD_{1,\bar{X}}, GAD_{k,\bar{X}}, \dots, GAD_{m,\bar{X}}, GAD_{NIC,\bar{X}}\right) \tag{9}$$

3.3. LAMDA-HAD algorithm

LAMDA-HAD improves the performance of the original algorithm in classification tasks, avoiding in certain applications, to send incorrectly classified objects to the *NIC*, and improving the classification process when an object has high similarity to more than one class [27]. This approach proposes the calculation of the *GAD* of the *NIC* adaptable to each class, and the calculation of the Higher Adequacy Degree *(HAD).*

The average value of the *GAD* of the class *p* for each individual in a class *k*, is calculated according to Eq (10).

$$MGAD_{k,p} = \frac{1}{n_k}\sum_{t=1}^{t=n_k} GAD_{p,t} \tag{10}$$

Where: n_k is the number of objects belonging to class k and $p = \{1,..,m\}$, $GAD_{p,t}$ is the *GAD* of the individual t for the class p, in the class $k, MGAD_{k,p}$ are used to compute the following parameters:

Adaptable GAD_{NIC}: The *GAD* of the *NIC* of each class is computed by Eq. (11), and it is the mean value of the all *GAD* belonging to a class k.

$$GAD_{NIC_k} = \frac{1}{m}\sum_{p=1}^{p=m} MGAD_{k,p} \tag{11}$$

Adequacy Degree of the GAD (AD_{GAD}): it corresponds to calculate the adequacy degrees of the *GAD* of the object with respect to the $MGAD_{k,p}$ in all classes (Eq. (12))

$$AD_{GAD_{k,p,\bar{X}}} = MGAD_{k,p}{}^{GAD_{p,\bar{X}}}(1 - MGAD_{k,p})^{(1-GAD_{p,\bar{X}})} \tag{12}$$

For $\bar{X}$ evaluated in each class.

Higher Adequacy Degree (HAD): it is calculated adding the AD_{GAD} in each class, resulting *HAD*:

$$HAD_{k,\bar{X}} = \sum_{p=1}^{p=m} AD_{GAD_{k,p,\bar{X}}} \tag{13}$$

Selection of the Class: The maximum value of the *HAD*, (Eq. (14)) allows determining the index E_I of the class to which the object has a greater probability of belonging.

$$E_I = \max(HAD_{1,\bar{X}}, \dots, HAD_{k,\bar{X}}, \dots, HAD_{m,\bar{X}}) \tag{14}$$

Finally, it is verified if the maximum *GAD* of the object in the estimated class E_I is greater than the corresponding GAD_{NIC} (Eq. (15)). If this condition is met, then the object is assigned to the class E_I, otherwise is assigned to the *NIC* class.

$$index = \max(GAD_{E_I,\bar{X}}, GAD_{NIC_{E_I}}) \tag{15}$$

3.4. LAMDA-RD algorithm

LAMDA-RD improves the performance of the original algorithm in clustering tasks, avoiding generating clusters that do not correspond with the number of desired groups [33]. This approach establishes a similarity measure to perform an automatic merging process between similar clusters through hybridization of the fuzzy logic and distance measurements. The split task is considered as an intrinsic characteristic of the capacities of LAMDA, since it can generate new groups based on the global adaptation of the individual to each pre-existing cluster and the NIC.

This approach consists of simple mathematical operations that do not consume excessive machine time and is based on the following concepts:

Cauchy Marginal Adequation Degree (CMAD). This parameter corresponds to the MAD calculated by the fuzzy Cauchy function presented in Eq. (16).

$$CMAD_{k,j} = \frac{1}{1 + dist(\bar{x}_j, \rho_{k,j})} \tag{16}$$

For the NIC, is set $CMAD_{NIC,j}(\bar{x}_j|\rho_{k,j}) = 0.5$, to keep the MAD criterion of the original LAMDA method.

Robust Marginal Adequation Degree (RMAD). This parameter corresponds to the product of the CMAD with a penalty parameter in each cluster $K_{k,\bar{X}}$ (Eq. (17)).

$$RMAD_{k,j}(\bar{x}_j|\rho_{k,j}) = K_{k,\bar{X}} \times CMAD_{k,j} \tag{17}$$

The penalty parameter $K_{k,\bar{X}}$ (Eq. (18)), requires to set by the user the d_{nb} (average distance between neighbours) and then to compute the average distance of the descriptors of the sample X to the center of each cluster ($d_{k,\bar{X}}$) using Eq. (19).

$$K_{k,\bar{X}} = \frac{d_{nb}}{d_{nb} + dist(d_{k,\bar{X}}, d_{nb})} \tag{18}$$

$$d_{k,\bar{X}} = dist(\bar{x}_j, \rho_{k,j}) = \frac{1}{n}\sum_{j=1}^{n}\left|\bar{x}_j - \rho_{k,j}\right| \quad (19)$$

Once calculated RMAD, the process for the calculation of GAD is like the original LAMDA algorithm, see Eq. (6).

For more details of this algorithm, and particularly the merging process performed for LAMDA-RD, see [33].

4. Architecture

In [3] has been proposed an autonomic QoS-oriented IoT platform with the architecture shown in the Figure 1. This architecture is composed of three components: the general controller, the managed entity (typically IoT gateways and server) and the applications/objects.

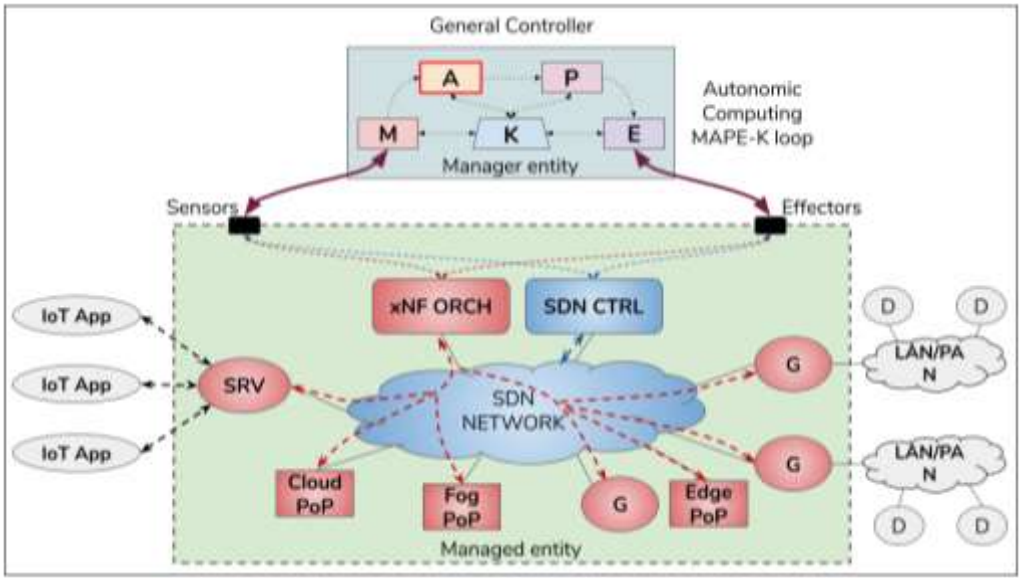


Fig. 1: High-level architecture

The autonomic IoT platform is based on the general controller, which is responsible for choosing the adaptation actions in the managed entity, to avoid the degradation of the QoS and to meet the "end-to-end" QoS required by the applications.

The connected objects and applications provide/exchange data with the IoT platform, the connected objects are sensors that supply data to applications, and actuators that can be controlled by these applications; the applications are the software underlying business activities benefiting from IoT. The managed entities are of two types (see Fig. 1): 1) those on which will be applied the adaptation actions, i.e. the entities involved in the implementation of the IoT platform, and those involved in the implementation of the network when this one is SDN-enabled (software defined networking), and 2) the entities in charge of the implementation of these adaptations, typically the ANFs (Applicative NFs) [22] or VNFs (Virtual NFs) orchestration entities and SDN networks control entities.

The distributed IoT platform considered in this work respects the OneM2M specifications [2]. It is composed of two types of entities: (i) IoT gateways (or MN-CSE for Middleware Node Common Services Entity): their primary role is to allow the connection of objects to the Internet through LAN (local area network) or PAN (personal area network) communication links, thus allowing the dynamic deployment of ANFs; (ii) IoT server (or IN-CSE for Infrastructure Node CSE): its role is to establish the link between applications and IoT gateways. The IT infrastructure in which the middleware entities are located is assumed to have different types of virtualization environments: Cloud, Fog or Edge [31]. This IT infrastructure is also supposed to rely on one or more IP networks interconnecting the IoT platform and the Cloud/Fog/Edge entities. Finally, the IT infrastructure includes the entities in charge of: (i) xNF ORCH: deployment / parameterization / activation / deactivation/ migration of xNFs (VNF/ANF [22]); (ii) SDN Controller: implementation of SDN actions on routers via a protocol such as Openflow [23] (e.g. OpenDaylight[24], Floodlight[25]).

For example, let consider an IoT application like automatic wildfire surveillance and monitoring system that enables early fire spotting by: (i) monitoring temperature over the forest using temperature sensors and (ii) detecting suspicious smoke with surveillance cameras. In case of fire, alerts with live streaming over the fire zone are delivered to wildfire management authorities. The proposed prototype enables maintaining operative QoS (including latency), thanks to the use of NFV (Network Function Virtualization), SDN and edge resources.

On the other hand, the autonomic IoT platform is based on the Autonomic Computing model initially defined by IBM [28] [29] [30], which is based on 4 major functional elements defined in the following way (see Fig. 2):

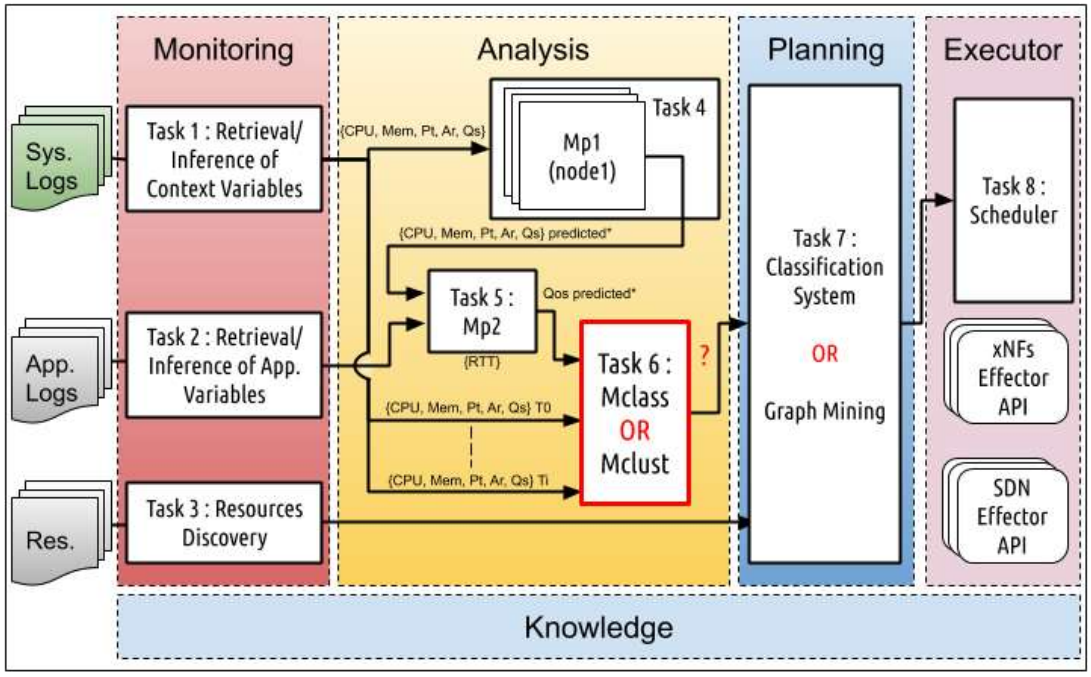


Fig. 2: General Controller - Autonomic Cycle for QoS provisioning.

- The Monitor (M): it catches the data from the context. Eventually, based on the information acquired from the managed entities, it generates, in case of difficulty, alarms that it sends to the analysis component. For the for QoS provisioning, it carries out the next tasks:
    - Retrieval/Inference of Context Variables;
    - Retrieval/Inference of Application Variables;
    - Resources Discovery.
- The Analyzer (A): it prepares a diagnosis based on the information recovered from the Monitor, to determine if there is a degradation of QoS in the system and its causes. For the QoS provisioning, it carries out the next tasks:
    - Prediction of the workload of the managed entities by the IoT platform;
    - Prediction of the QoS that can offer the IoT platform to the application;
    - Identification and Diagnosis of the platform operational state: this task can be carried out by a classification or a clustering model that will be tested in this paper.
- The Planning (P): it determines the adaptation actions that have to be performed with the aim to maintain the targeted end-to-end QoS, according to the diagnosis carried out in the previous step and transmits this choice to the execution component.
    - If the analyzer is based on the classification model, then the planning is based on a classification system that determines the set of actions to execute for each operational state.
    - If the analyzer is based on the clustering model, then the planning is based on a graph mining model that discover in real-time the set of actions to be executed in the current operational state.
- The Executor (E): based on the actions defined for the planning component, it schedules and executes the planned actions via the xNF orchestration and SDN control entities API.

Back to the example:

- The Monitor collects and processes the following data: Latency (the amount of time that takes a message to reach the source since a given destination), Resources consumption in every IoT platform node (CPU/MEM), etc.
- The Analyzer, based on the monitoring data, collected and calculated above:
    - Detects the abnormal situations from the data collected by the sensors deployed in the forest;
    - Diagnoses the future state of the environment (IoT platform).
- The Scheduler determines the appropriate plans to be implemented according to the diagnosed states, like the QoS mechanisms or the fault tolerance mechanisms to be used.
- The Executor implements the generated plans. In the example, this implementation results in the deployment of QoS mechanisms and redundancy gateways

Our approach presents the next novelties:

- It has an autonomous behavior in order to self-management
- It can consider any situation in the IoT context, and particularly, new functionalities like for example, the time slicing to consider the QoE.

- It is based on machine learning techniques, in order to use the real-time data in the context to build knowledge models.

5. Experimental Scenario

This section analyzes the main functionality of our approach, which is about its capability for detecting operational scenarios, and according to that, it adapts the IoT platform. The detection of the operational scenario can be carried out according to two approaches: i) If the operational scenario (state) are known, then the system can use classification approaches in order to determine the current state (class) in the IoT platform; ii) If the operational scenarios are not known, then the system must discover them. In the first case can be used classification models, and for the second, clustering models. In both cases, it is very important to know the quality of the models before to be used in a real context. This is the goal of this section, but additionally, carry out the comparison between both models, in order to determine which one is better in the context of diagnosis in one autonomic IoT platform.

To have data to evaluate the classification and clustering models, in this paper is considered the following IoT case study. This case study has been presented in [3] and consists in the monitoring of a forest fire based on a connected IoT-based system. It enables early fire spotting by: (i) monitoring temperature over the forest using temperature sensors and (ii) detecting suspicious smoke with surveillance cameras. In case of fire, alerts with live streaming over the fire zone are delivered to wildfire management authorities. The proposed prototype must maintain the QoS (including latency) thanks to the use of software modularity, NFV, SDN and edge resources. In general, it has been have collected data from this infrastructure to analyze the behavior of our autonomic cycles based on classification or clustering models.

The Figure 3 describes the experimental scenario. In the cloud is the server where are executed the applications. The variables/descriptors that characterize the behavior of the applications at this level are: (i) RTT (Round-Trip Time): the time between a service request and the response (in msec.); (ii) LOSS: indicates the success/failure of the requested service (expressed as true/false). The variables that characterize the server are: (i) CPU: the percentage of consumption of the computing resources of the server (expressed as a percentage of the available resource); (ii) MEM: the percentage of consumption of the memory of the server (ii) MEM-AVR: the simple moving average (SMA(x)) of the memory consumption percentage during an interval of time (in this case, x requests), due to a particular memory operation related to the typical garbage collector mechanism of IoT entities; (iii) Pt: the processing time of the request on the server (in msec.); iv) Qs: the queue size on the server; and v) Ar: the arrival rate of requests to the server (represented as integers).

In the fog level is the GI entity, which is the intermediate gateway; the variables that characterize its behavior are the same as those of the server: MEM, MEM-AVR, CPU, Pt, Qs and Ar. In the edge level is the GF1 entities, which are the final gateways directly connected to the objects, and the variables that characterize its behavior are the same as those of the server: MEM, MEM-AVR, CPU, Pt, Qs and Ar. The IoT platform entities (server and gateways) are real entities whose specifications are provided in Table 2.

Table 2. Description of the entities of the MW

Element	Description	Specifications		
		RAM	CPU	Disk
App	Wildfire Surveillance and Monitoring System [3]	-	-	-
Server	Entity that represents the set of instances of the infrastructure node specified in the oneM2M standard (i.e. IN-CSE)	4Gb	2×i7-7500U	10Gb
Gateway {I/F1}	Entity that represents an intermediate node as specified in the oneM2M standard (i.e. MN-CSE)	1Gb	4×ARMv7	8Gb

The applications are emulated through stochastic HTTP traffic injectors, where requests are sent to the server and to the gateways {I, F1}.

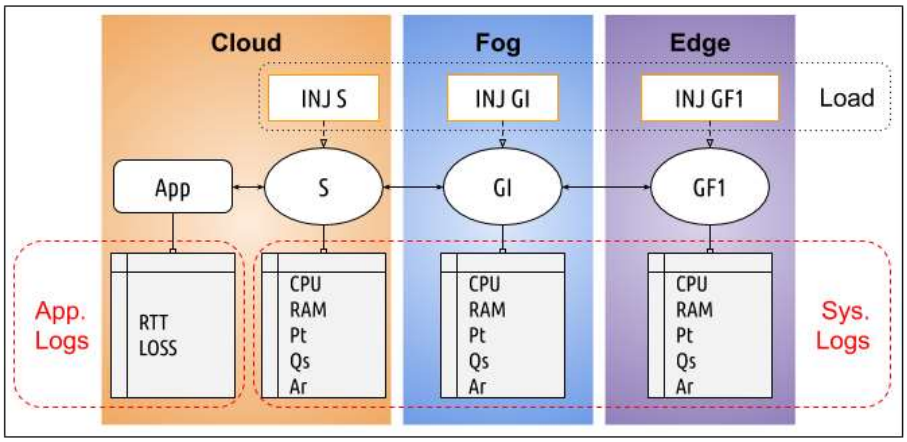


Fig. 3: Experimental Scenario

In this context, there are 8 operational scenarios for an application, which are defined in Table 3. The first one is where the three managed entities of the IoT platform are not loaded, and the last one is where they are loaded. The other ones are combinations of them and determine the operational scenarios to be detected by the machine learning models (classification or clustering). To generate artificially these scenarios has been injected load in each component of each computational level (e.g., on the server in the cloud level, at the gateway GI in the fog level, and so for the rest). For example, for the first scenario has not been injected load in the entities.

Table 3. Operational state of the IoT platform

N	Cloud	Fog	Edge	Injectors		
				S	GI	GF1
1	Not loaded	Not loaded	Not loaded	OFF	OFF	OFF
2			Loaded			ON
3		Loaded	Not loaded		ON	OFF
4			Loaded			ON
5	Loaded	Not loaded	Not loaded	ON	OFF	OFF
6			Loaded			ON
7		Loaded	Not loaded		ON	OFF
8			Loaded			ON

6. Experiments and Result Analysis

In general, this paper uses the classical metrics used in the literature to evaluate the performance of the knowledge models. Particularly, the goal of this section is to know if it is possible to build good models with the data from the IoT platform (If this data is useful), and also, evaluate the quality and usefulness of the models built with the LAMDA technique in the context of our approach.

Thus, in this work are computed the classical metrics of the classification and clustering domains to determine the quality of the proposed models, and additionally, the receiver operating characteristic (ROC) analysis. The ROC analysis provides a statistical method for the assessment of the diagnostic accuracy of a model, being used for three specific purposes: determine the cutoff value with the highest sensitivity and specificity, evaluate the discriminating capacity of a diagnostic model, and compare the discriminating capacity of different diagnostic models. In this way, the ROC curves are very useful in our context, in order to give us information about the quality of the classification and clustering models to diagnose the current operational state. Particularly, there are two metrics of the ROC analysis that are interesting (see Table 4): the proportion of true positives over the total of loaded operational states (that is the sensitivity), and the proportion of false positives on the total of normal operational state (it is known as the 1-specificity metric). With these values is built the ROC curve.

Table 3. ROC metrics

Sensitivity	$= TP/(TP + FN) = PVP$ (proportion of true positives)
Specificity	$= TN/(TN + FP) = PTN$ (proportion of true negatives) $= 1 - PFP$ (proportion of false positives)

Where: TP: true positives, FN: false negatives, TN: true negatives and FP: false positives

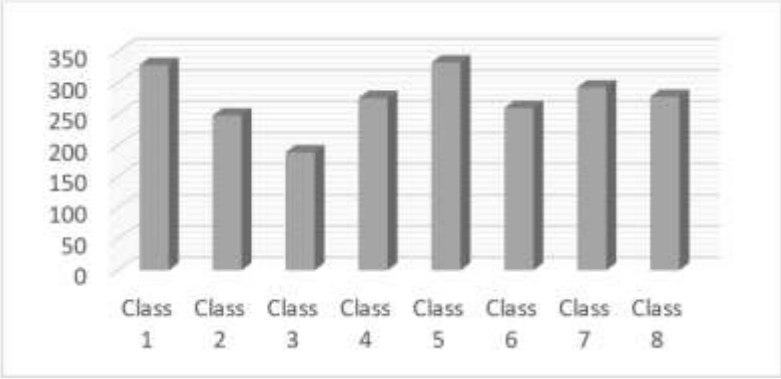
The metrics of classification computed in this work are: i) *Accuracy*: it is the ratio of the number of correct predictions with respect to the total number of input samples (a high accuracy is very important in the diagnosis context to determine a current bad operational state in the system, and consequently, to be able to act); ii) *Precision*: it is the ratio of correct predictions among the total of predictions, which must be high in the context of bad operational states; iii) *Recall*: it is the ratio of predictions that have been detected over the total amount of correct predictions, which must be maximal to detect all the instances that correspond to a bad state; and iv) *F-measure*: it is the ratio between precision and recall.

The metrics of clustering calculated are: i) *Cohesion*: it is a measure of how similar an object is to its own cluster, and it is important a high value in diagnosis to have robust groups of operational states; ii) *Separation*: it is a measure of how similar an object is to other clusters, and it is important a high value in diagnosis for the same reason of previous metrics; iii) *Silhouette*: is the ratio between cohesion and separation; iv) *Calinski-Harabasz coefficient*: it evaluates the cluster validity based on the average between clusters and within

cluster sum of squares, and it is important a high value in diagnosis to have robust groups of operational states.

For the construction of the models, LAMDA-HAD and LAMDA-RD techniques have been used, which are fuzzy classification/clustering algorithms based on the LAMDA paradigm that combines the concepts of neural network architectures and fuzzy clustering [26][27][33]. One of the main advantages of LAMDA over other classifiers, such as the well-known *K-means* and *GK-means* techniques, is that LAMDA is capable of generating new classes after the training stage in a non-iterative way, so it can be used online. In this paper, the two improvements described above are tested in the QoS Management in an Autonomic IoT Platform, and the results with these models are presented.

Before for starting, data sciences techniques have been applied to the dataset in order to prepare it before performing the testing stage. These techniques eliminate atypical data, values with null data, and data with noise, always trying to keep the classes as balanced as possible. The final class distribution is presented in Figure 4.



Class	# Objects
1	329
2	248
3	189
4	276
5	333
6	260
7	293
8	278

Fig. 4: Number of individuals by class

Also, the descriptors have been analyzed to determine their relationships and their discriminant powers. In this way, during this phase has been deleted the next descriptors: LOSS and Qs.

6.1. Test of the classification model

The results of the LAMDA-HAD classifier have been obtained through a training carried out with 80% and a test with 20% of the database (around 450 individuals), with a total of 2206 instances and 16 descriptors (see section 4, where are described these variables/descriptors). The scheme of the test (hold-out method) is shown in Figure 5.

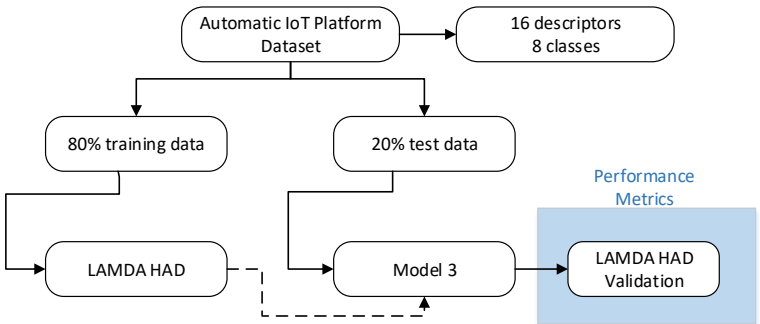


Fig. 5: Hold-out method used for validating LAMDA-HAD in classification

Figure 6 shows the typical LAMDA results graph (when the algorithm is tested in classification mode), in which is presented the distribution of GADs for each individual considered in the test stage. With these results it is possible to observe the behavior of the GADs calculated for each individual, and which of them is incorrectly assigned.

With the aim of improving the classification model, and based on the results presented in Figure 6, it has been considered to eliminate the descriptors that are observed to have an abnormal behavior and could decrease the performance of the algorithm. However, after several tests performed, It has been concluded that the best results, which are presented below, are obtained with 16 descriptors, concluding that each one has relevant information regarding the characterization of the classes.

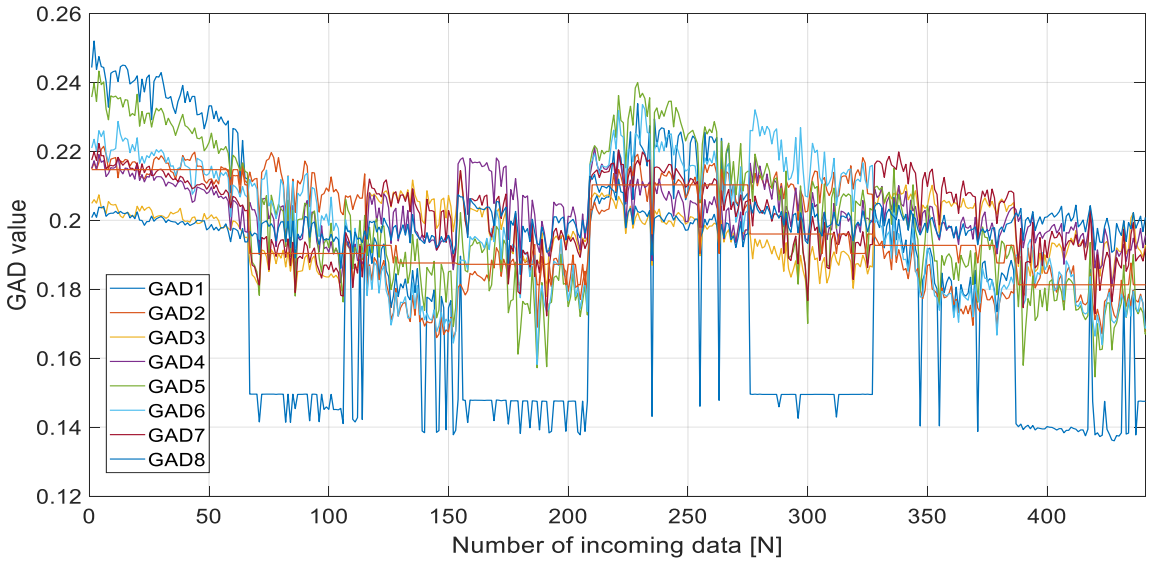
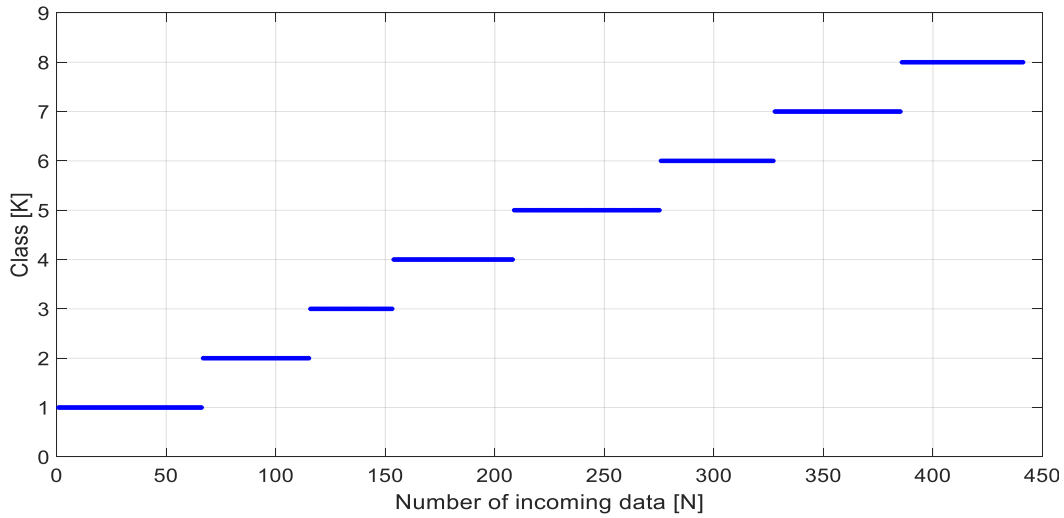


Fig. 6: Obtained GADs in each class

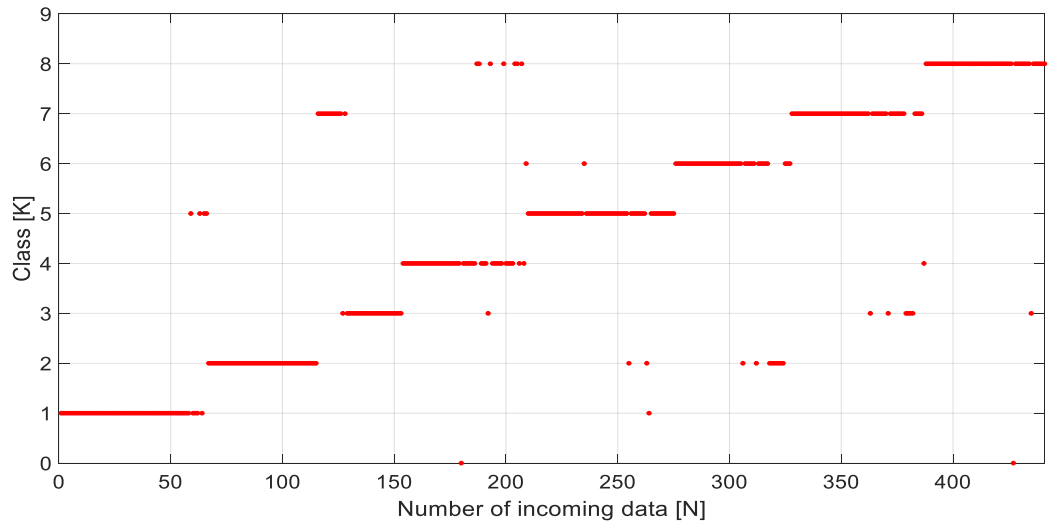
Figure 7a shows the actual distribution of the objects in each class, and Figure 7b shows the assignment of the objects made by the LAMDA-HAD algorithm. According to Figure 7, in each class there are errors in the classification. In other hand, although according to the graph it seems that an individual has been assigned to more than one class, this really is not true, each individual has been assigned to one and only one class at a time.

Most of the objects have been correctly assigned to their respective class; however, there are certain mis-assigned individuals that decrease the performance of the algorithm. This is

evident for class 3, which is particularly the one that fewer individuals have when performing the data analysis, because there were a large number of atypical values that were identified and eliminated, as well as data with noise. Despite having performed this procedure in this class, it can be observed that almost 40% of its objects are poorly classified and have characteristics more similar to class 7, which is considerably affecting the metrics of the model obtained by LAMDA-HAD. According to the Figure 8, which corresponds to the ROC graphics for each class, the above is verified, since the classes that present a better classification are those that go through the vertices (0,0), (0,1), and finally (1,1), which shows that the class 3 is the one that has more problems, since its metric of true positives is low (due to the confusion it presents with class 7). Also, the class 4 in certain objects tends to be confused with class 8. The remaining classes are well modeled, and only one object was sent to the NIC (class 0), which corresponds to an atypical data due to noise in the descriptors.



(a)



(b)

Fig. 7: (a) Real distribution of the objects in each class, (b) objects assigned by the LAMDA-HAD algorithm

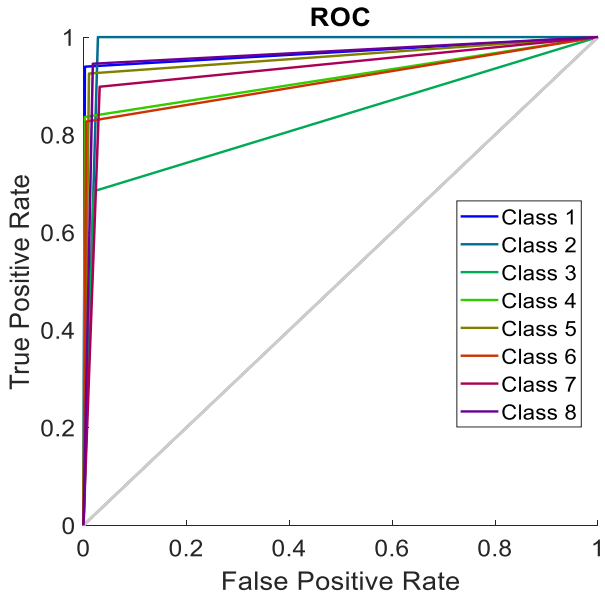


Fig. 8: ROC metric of the classification model

A comparative analysis of LAMDA-HAD results and other classification algorithms, such as: Linear Discriminant Analysis LDA, Random Forest, Support Vector Machines SVM and Feedforward Neural Networks FNN, are shown in Table 4.

Table 4. Performance metrics for the classifier

Algorithm	**Accuracy**	**Precision**	**Recall**	**F-meas.**	**Sens.**	**Spec.**	**AUC**
LAMDA-HAD	0,8740	0,8507	0,8678	0,8574	0,8678	0,9746	0,9212
LDA	0,9410	0,9426	0,9391	0,9390	0,9391	0,9915	0,9653
RF	0,9864	0,9864	0,9871	0,9863	0,9871	0,9980	0,9925
NN	0,7357	0,7339	0,7302	0,7281	0,7302	0,9615	0,8458
SVM	0,1817	0,1444	0,1508	0,0762	0,1508	0,8795	0,5151

In general, based on these metrics, it can be determined that:

- In all the metrics, the algorithm with the best performance is RF, which has a 10% higher yield than LAMDA-HAD, except in the metrics of specificity and sensitivity, where the results have a lower range of difference (2% and 7% respectively) which makes it possible to show that the performance of LAMDA-HAD is quite acceptable.
- It is also important to emphasize that RF and LDA are not clustering techniques.

- LAMDA-HAD has better performance than SVM and FNN with the advantage of being a non-iterative method.

In particular, the results obtained with LAMDA-HAD show that:

- Even though there is a small imbalance in class 3 with respect to the other classes, the accuracy and F-measure metrics are very similar (around 86%), which shows a good trained model.
- The AUC curve is a value that must vary between 0 and 1, and must never be less than 0.5, which would be the worst case and, corresponds to the gray line in Figure 7. In this case, the average value of the ROC in each class is 0.9212, very close to 1, which implies that the model is good at predicting true positives, and in turn, it has a low percentage of false positive predictions, that is, it has a good performance in this case study.
- Regarding the values of sensitivity and specificity of the model, they are good. The high sensitivity means that 86% of the objects assigned as positive in a class were correctly identified, while the high specificity indicates that 97% of the objects classified as belonging to a class, then did belong to that class, being able to conclude that the model is quite adequate in this application.

In Figure 6, an abnormal behavior of GAD1 can also be noticed, specifically in classes 2, 4, 6 and 8, in which it is seen that it is well below the other GADs, being almost continuous in certain sections. Through experimentation, it has been concluded that the descriptor Ar of GF1 is the one that causes this behavior, reason why it has been proceeded to eliminate it with the purpose of evaluating the obtained results and finding possible improvements. The new GADs, without this descriptor, are presented in Figure 9, and their corresponding metrics in Table 5.

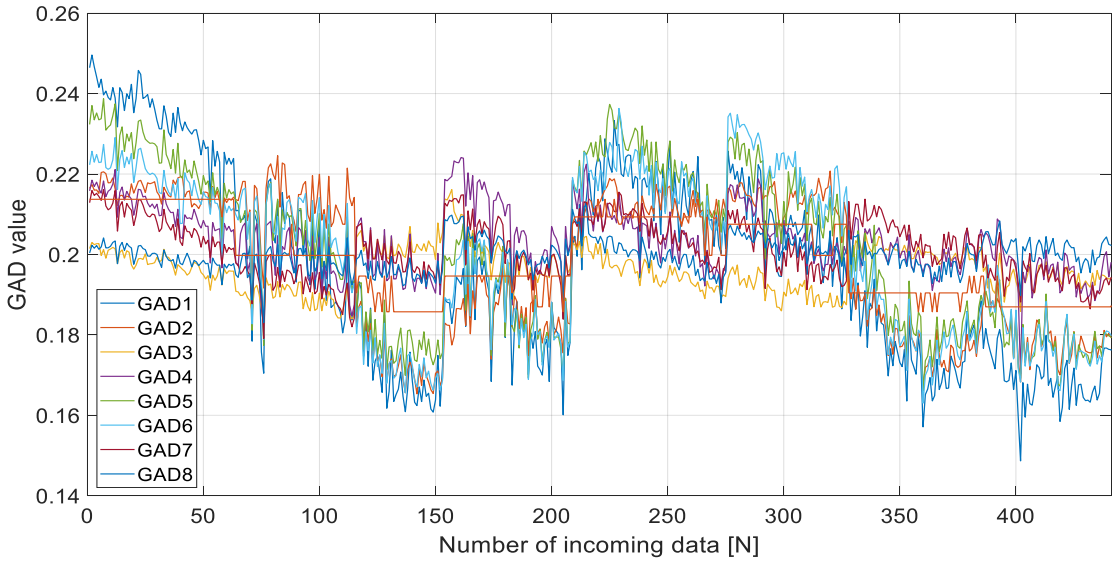


Fig. 9: GADs in each class, eliminating the Ar-GF1 descriptor

Table 5. Performance metrics for LAMDA-HAD, eliminating the Ar-GF1 descriptor

Accuracy	Precision	Recall	F-meas.	Sens.	Spec.	AUC
0,8436	0,7977	0,8429	0,8153	0,8429	0,9601	0,9015

As is shown in Figure 9, the GADs now have a similar behavior in all classes; however, analyzing the Table 5, it is evident that the performance of the LAMDA-HAD model has decreased in all metrics, especially F-measure is the most affected, since it went from 85% to 81%. It is due because for simple inspection, now all the GADs have a more homogenous behavior in each class (see Figure 9), with respect to those shown in Figure 6. However, these results show a lower performance in the classification model (see Table 4), specifically in Precision, and therefore in F-measure, which means that by eliminating the aforementioned descriptor, relevant information for the algorithm has been lost. This implies that the ratio of correct predictions among the total of predictions has been affected, and the characteristic of a good detection of the operational states is being lost. The other metrics decrease in an interval of 2-3%, which is not considerable. However, since none of their values improve, it is another sign that the best model is the one that considers all the information

Finally, with this experiment, it has been possible to conclude that although this descriptor generates an abnormal behavior in the GAD1, this is a characteristic of the model that must be considered with the goal of obtaining better results.

6.2. Test of the clustering model

The clustering tests, in this case study, have been carried out with all the individuals, i.e. 2206, through the LAMDA-RD algorithm that works by taking the individuals as data streams to perform the grouping operations. The algorithm has been calibrated based on different experiments, presenting in this paper the best results, comparing them with well-known clustering techniques. Basically, it consists in: first, evaluating how good is the model obtained with respect to other clustering algorithms; second, analyzing the quality of the clusters found, and finally, comparing it with the results obtained when it is considered labeled clusters (real classes).

Table 6 shows the metrics (silhouette "SC", cohesion "SSW", separation "SSW", SSW/SSB, and Calinsky-Harabarz Coefficient "CHC") calculated for the different clustering algorithms. A good clustering algorithm must generate clusters with high intra-cluster homogeneity, a good inter-cluster separation and high connectivity between neighboring individuals. The results show that with an adequate calibration, LAMDA-RD presents results as good as the DBSCAN algorithm. The other methods, such as Kmeans, Kmedoids and Agglomerative Hierarchical Tree (AHT), require as an input parameter the number of clusters to build, choosing the value of 8 (corresponding to the real groups of the data). In these three algorithms, it is shown that the metrics are not the best, and that the formed clusters by LAMDA-RD are denser (lower cohesion value) and more separated from each other (higher value of SSB).

LAMDA-RD finds 15 clusters, consisting of the elements shown in Figure 9, in which it can be seen that up to cluster 6 there is a considerable amount of elements, especially in cluster 1 and 3, and from cluster 7 to 15 are scattered elements; this corresponds to data that, as is seen when constructing the classification model, may correspond to atypical data (outliers) that were not eliminated, and for that reason, they are not assigned to a cluster; this decreases the performance of the clustering algorithm.

It is important to clarify as in the previous case, that each object has been assigned to a single cluster, however, due to the amount of data, Figure 10 does not show that.

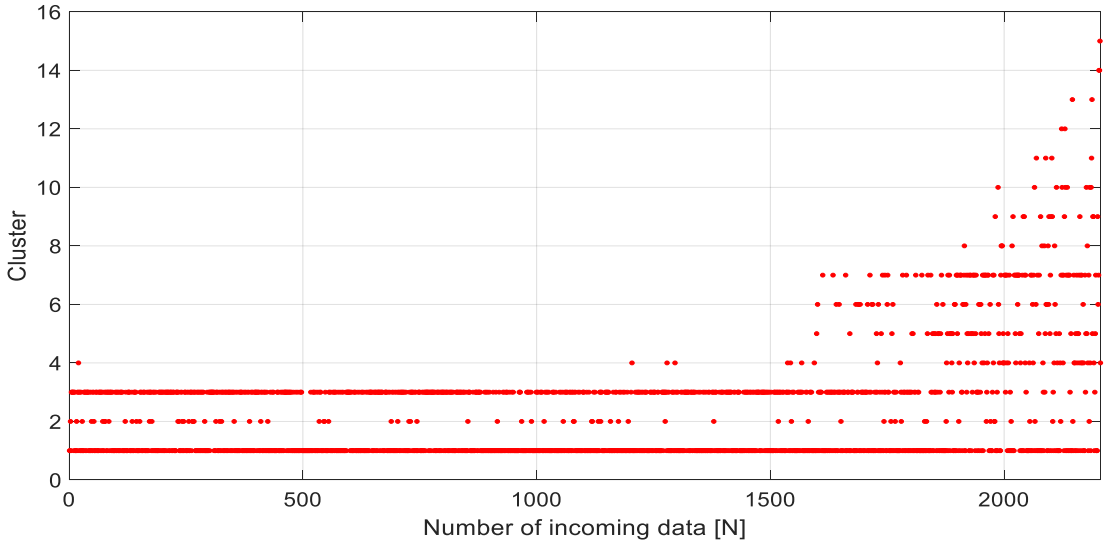


Fig. 10: Assigned objects to clusters using LAMDA-RD

Table 6. Performance metrics of clustering algorithms

Algorithm	SC	SSW	SSB	SSW/SSB	CHC	# CLUST.
LAMDA RD	-0,0261	0,6848	0,3672	1,8649	293,7209	15
KMEANS	0,0097	0,7308	0,2927	2,4971	786,5929	8
KMEDOIDS	0,0010	0,7301	0,2921	2,4990	787,1934	8
AHT	0,0104	0,7879	0,0268	29,4440	9274,8540	8
DBSCAN	0,0184	0,5301	0,4055	1,3072	262,0866	13

According to the Table 6, several aspects can be determined:

- Regarding the silhouette coefficient, all values are close to zero, which implies that there are objects near the edge of two clusters. In the case of LAMDA-RD, being close to zero and negative implies that some of these data may be incorrectly assigned to a cluster.
- Regarding the SSW coefficient, LAMDA-RD has the second lowest value, which shows good intra-cluster homogeneity, that is, a high similarity between the objects of the same cluster.

- Regarding the SSB coefficient, LAMDA-RD has the second highest value, which shows a good inter-cluster separation, close to that presented by the DBSCAN, with an important difference with respect to the rest.
- The good results presented by the Kmeans, Kmedoids and Agglomerative Hierarchical Tree algorithms are attributed to the fact that in this case study, it is initially known the number of clusters to build, and it is given as information to these algorithms. LAMDA-RD does not know this value, and builds clusters based on the similarity of objects in line, which is an additional advantage of this algorithm.
- Also, the metrics SSW/SSB and CHC, allow determining the quality of the clustering algorithm. Likewise, LAMDA-RD has the second lowest index, which allows us to determine that the algorithm has a quite acceptable performance.

In Table 7, the obtained results are presented when it is considered as clusters the classes formed by the 2206 labeled objects (real classes).

Table 7. Performance comparison between the Real Classes (Labeled Clusters) and the obtained by LAMDA-RD

	SC	**SSW**	**SSB**	**SSW/SSB**	**CHC**	**#CLUST.**
REAL CLUSTERS	-0.1497	0,6120	0,4979	1,5128	387.1738	8
LAMDA RD	-0,0261	0,6848	0,3672	1,8649	293,7209	15

The results show that:

- From the point of view of the Silhouette metric in the real classes, it is negative moving away from zero, which indicates that there is a small number of objects misassigned to those clusters. LAMDA-RD has a better value since it is closer to zero, that is, it has fewer objects misassigned to clusters; however, in both cases the ideal would be to obtain values greater than zero and close to 1, which is not fulfilled.
- Observing the SSW values, it is determined that the labels placed on the clusters (real classes) are better than those obtained by LAMDA-RD.
- The SSB metric shows that there is a greater separation distance between the clusters of the real classes, not so in the case of LAMDA-RD.
- The SSW/SSB and CHC indices also show that in the case of clustering by LAMDA-RD the results are not satisfactory with respect to the labeled clusters.

Based on everything analyzed above, it can be concluded that: although the LAMDA-RD algorithm presents quite good results in the clustering process compared to other classical techniques (results of Table 6), this grouping is not as adequate as the one which is obtained by the real classes, or by applying classifying models, such as the case of LAMDA-HAD (see Table 4 and 7). This is due to the great variability of the descriptors that, in turn, present complex behaviors as those observed in class 3, and additionally, sometimes with the presence of atypical data, which in this work have been eliminated almost in their entirety.

However, the behavior of the clustering algorithms is interesting, because they may be indicating sub-states that fail to be determined in the classification models, that is, that the experts who have labeled the data have not determined. Those extra subgroups (7 in the case

of LAMDA-RD and 5 in the case of DBSCAN) can be interesting to analyze, to determine if specific tasks of configuration are required in the context of the IoT platforms.

6.3 Profile of each cluster

One of the advantages of LAMDA is that can be built the profile of each cluster/class, which is interesting in order to define the set of tasks in the planning phase. Each descriptor has a specific behavior in each cluster, which is very useful in order to diagnose the current situation, and determine the tasks over the IoT platform to guarantee a given QoS. A profile determines the average value of each descriptor, which is very useful in order to determine the set of tasks to be executed to solve the degradation situation in the autonomic cycle.

Some of the profiles that can be determined are:

- General profile of the IoT Platform
- Profile by entity
- Profile with specific descriptors (e.g. CPU and/or MEM by entities)

The profile can be used to eliminate the descriptors not discriminating, to determine specific problems in the IoT platform, etc. A first example is the Figure 11, where is shown the profile of each state (class) of the IoT platform. For example, the class one represents a state without a high load in the different components of the IoT platform.

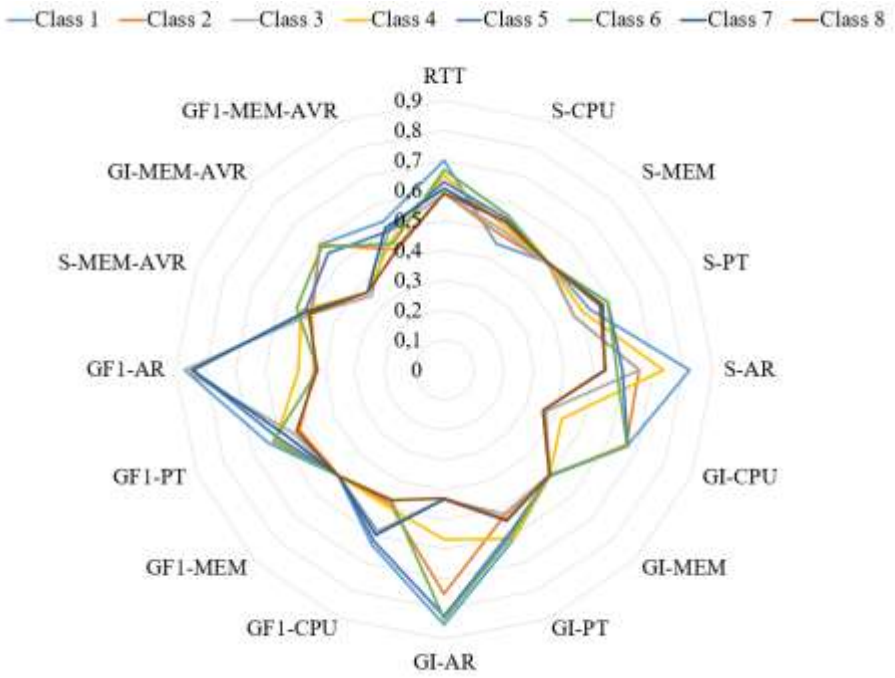


Fig. 11: General Profile of the IoT platform with 16 descriptors

In the Figure 11 are shown some descriptors that are not discriminating. These descriptors can be deleted to improve the profile (pattern) of each class (see Figure 12). In the Figure 12, the MEM descriptor of the all entities has been eliminated, as well as the RTT descriptor, because they have not an important difference in each class.

With the new set of descriptors, it is possible to build best patterns for each class (see Figure 12), in order to differentiate them well for defining the specific set of tasks for the QoS problem describes by in each class.

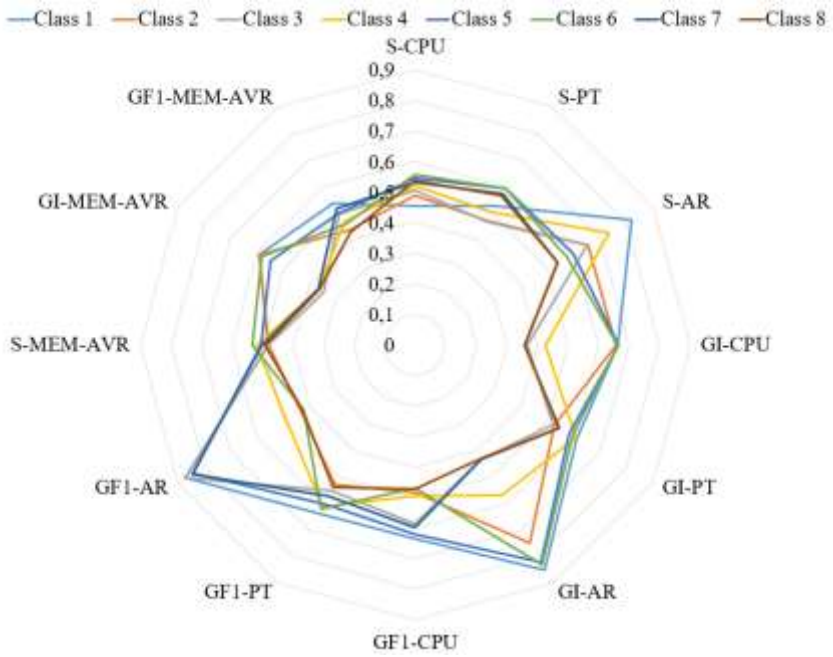


Fig. 12: General Profile of the IoT platform after the elimination of 4 descriptors

Additionally, it is possible profiles by entity (see Figure 13) or by descriptor (see Figure 14), in order to analyze specific aspects. For example, the Figure 13 shows the behavior of the server on the IoT platform, and the Figure 14 the workload of the different entities in the IoT platform.

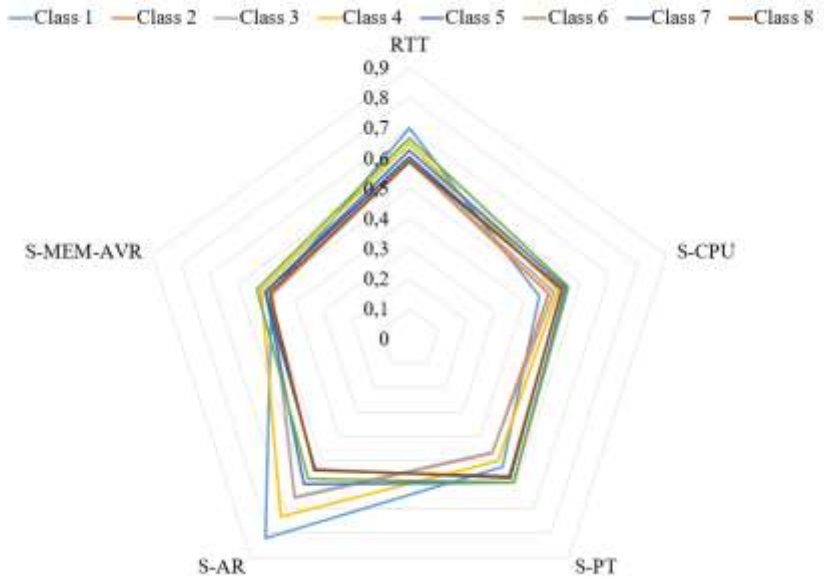


Fig. 13: Profile of the server on the IoT platform

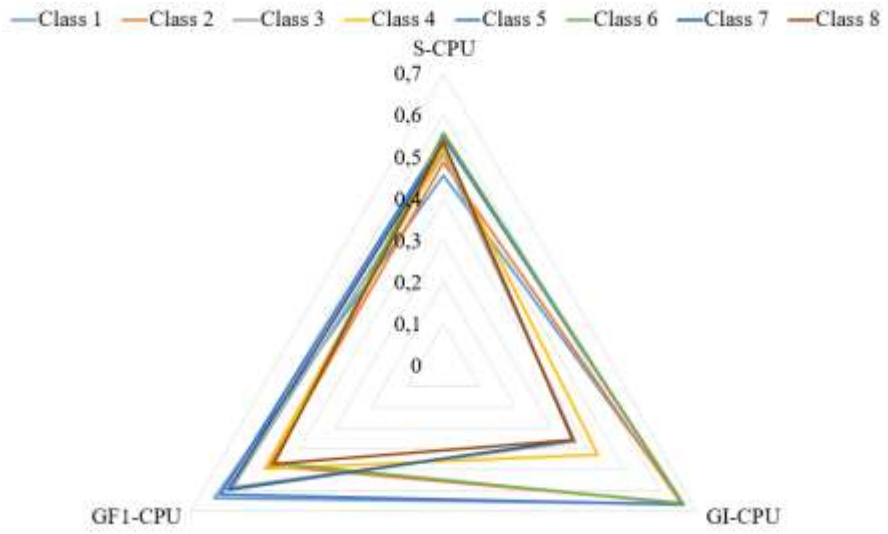


Fig. 14: Profile of the CPU descriptor on the IoT platform

That is very interesting in the context of the diagnostic tasks, mainly when it is not known the operational state of the platform (in the clustering context). This information can determine the specific problem (e.g. a CPU overloaded), the specific entity with the problem (e.g. server), which is very useful in order to determine the set of actions in the planning phase. For the autonomic cycle based on the clustering strategy, it is very important, because this set of studies means a deep analysis of the different aspects of the IoT platform for a more precise diagnosis, in order to determine the set of specific tasks to carry out.

As well, as it has been said before, these profiles give us as second very interesting information, the interest of the different features. For example, by entity some descriptors are more sensitive to variation and more discriminating than others, as the S-AR with respect to S-MEM-AVR that gives more information about the different operational states of the server (see Figure 13). However, to conclude about under what conditions a descriptor can be preferable to the other, in future works will be analyzed the deployment cost of the features, in order to determine the optimum number of descriptors (the most relevant and less expensive), which allow a good diagnosis of the state of the IoT platform with an acceptable margin of error.

6.4. Discussion about the results

According to the previous experimental results, it is possible to conclude that our approach gives useful information, in order to make decisions about the specific aspects to consider guiding the processes of provisioning of resources, in order to improve the QoS. For example, if it is known the current state in the platform, then can be established the specific transport functions to reach QoS goals; or if it is analyzed the cluster of the current state, then the profile can be used to determine the specific problems (e.g. the lack of gateway memory), and act over these specific components or with the specific transport functions that improve the platform performance.

Particularly, in the autonomous cycle based on the idea of detection of the current operational state in the IoT platform, if the classes that represent the operational states are very well known, then a classification system work very well to guarantee a given QoS. In the second autonomous cycle based on the discovery of the current operational state, it can determine very interesting things, due to the profile built with the LAMDA technique. Specifically, it defines the behavior of the components of the platform, which can be used to make better decisions. In addition, it can discover subclasses inside the classes, something that cannot be carried out with the other autonomic cycle. In the literature, there are not this type of study, where the behavior of machine learning techniques is studied, in order to determine their diagnosis capabilities in the context of the QoS management in an IoT platform.

In a management context, these results can be used to determine the new inversions in the IoT platform, in order to customize it to the applications used in its context, among other things; and in academic context, these types of analysis must be carried out to design the new transport functions, the suitable methods of the components of an IoT platform to the QoS requirements, etc.

Finally, about mathematical modelling of the QoS parameters in IoT platform, [36] presents the factors necessaries to consider while designing any IoT-enabled system. It also proposes the mathematical models for QoS parameters like reliability, communication complexities, latency, and aggregation of data for IoT. Our approach can use these mathematical models about QoS parameters in IoT platforms, in order to measure the degradation of performance, or any other aspect, which is one of the advantages of this proposal.

7. Conclusions

In this article is analyzed the behavior of the two autonomous cycles that allow avoiding the degradation of the QoS in IoT platforms. Each of them depends on models of machine learning, in one case of a classification algorithm and on the other case of a clustering algorithm. This article has been dedicated to analyze the behavior of both models in a real IoT context.

In this regard, the results indicate that the classification model is quite robust, because it is capable of determining the operations states in the system, but it is up to the expert to label the information to be used by the model. For the clustering algorithms, they are affected by the characteristics of the descriptors, giving good results, but not better than those obtained with the classification.

One aspect to highlight is that the clustering algorithms give more clusters than the number of operational states, which is important to analyze, because they can be sub-states within states, which the experts do not know. That could improve the QoS results, due to the capability of giving more precise answers to the needs of the adequacy of the IoT platforms before the degradation of QoS. That will imply further analysis in that context.

Also, in this work is evaluated the phase of analysis of both cycles, which allows them to diagnose what may be happening in the IoT platform. In general, in the case of the cycle

based on the classification model, the dependence of experts is greater, both to label the operational states (data) and to determine the tasks aimed at improving the platform. In the case of the cycle based on the clustering model, although the results are not so good, not depending on the expert gives more robustness to the process, but it requires to develop more complex models, which must permanently evaluate their robustness.

Finally, this paper has shown the interest of the utilization of a technique like LAMDA, in order to carry out an exhaustive diagnosis process, to determine the QoS problems in an IoT platform. LAMDA can be used to build both classification and clustering models, in order to compare their qualities in a diagnostic process. Additionally, LAMDA generates a set of profiles that provide useful information to diagnose the problem, in order to carry out a deep analysis about the current QoS degradation situation in an IoT platform, and its possible solution.

One of the main problems with this type of approach is the necessity of good data, which can imply a very good preprocessing data phase that in real-time applications can be a problem. The other limitation is that the data-driven models have a useful life, which require every certain time the retraining the models in order to follow the current context.

With respect to future works, the rest of data driven models of the autonomic cycles will be developed, in order to test the autonomic cycles in real contexts. Also, the design of new autonomous cycles will be carried out, in order to solve other types of problems in IoT platforms using our approach, like the providing of resources to guarantee the QoE of real-time applications, one of the main challengers in 5G networks.

References

[1] ETSI TS 102 690, "Machine-to-Machine communications (M2M); Functional architecture," v2.1.1, 2010.

[2] oneM2M TS v1.6.1, "oneM2M functional architecture," 2015.

[3] C. Ouedraogo, S. Medjiah, C. Chassot. J. Aguilar, "Autonomic Middleware based on Data Analysis Tasks for the QoS Management in the IoT", Technical Report, LAAS, 2018.

[4] J. Aguilar, J. Cordero, L. Barba, M Sanchez, P. Valdiviezo, L. Chamba, "Learning Analytics Tasks as Services in Smart Classroom', Universal Access in the Information Society Journal, Springer, Vol. 17, No. 4, pp. 693–709, 2018.

[5] J. Aguilar, O. Buendia J. Cordero "Specification of the Autonomic Cycles of Learning Analytic Tasks for a Smart Classroom", Journal of Educational Computing Research, vol 56 no. 6, pp. 866-891, 2018

[6] A. Otebolaku, G. Lee, "Towards context classification and reasoning in IoT," 14th International Conference on Telecommunications (ConTEL), pp. 147-154, Zagreb, Croatia, June 2017.

[7] P. Mahalle, N. Prasad, R. Prasad, "Object Classification based Context Management for Identity Management in Internet of Things" International Journal of Computer Applications, Vol. 63, No.12, 2013.

[8] R. Ferrando, P. Stacey, "Classification of Device Behaviour in Internet of Things Infrastructures: towards distinguishing the abnormal from security threats".

International Conference on Internet of Things and Machine Learning, Liverpool, United Kingdom, Oct. 2017.

[9] K. Hong, J. Moon. C. Won, J. Ju, "Identifying Service Contexts for QoS Support in IoT Service Oriented Software Defined Networks", In Mobile, Secure, and Programmable Networking (Ed. B. Samia, et al.), Springer, pp. 99-108, 2017.

[10] M. Antunes, D. Gomes, R. Aguia, "Towards IoT data classification through semantic features", Future Generation Comp. Syst., Vol. 86, 792-798. 2018.

[11] S. Siby, R. Ranjan, N. Tippenhauer. "IoTScanner: Detecting Privacy Threats in IoT Neighborhoods". 3rd ACM International Workshop on IoT Privacy, Trust, and Security. pp. 23-30, Abu Dhabi, Apr. 2017.

[12] V. Michell, J. Olwen, "The Human-IoT Ecosystem: An Approach to Functional Situation Context Classification". In The Internet of Things in the Modern Business Environment (Ed. T. Ochs, U. Riemann), IGI Global, pp. 223-248, 2017.

[13] D. Goel, S. Chaudhury, H. Ghosh "An IoT approach for context-aware smart traffic management using ontology". International Conference on Web Intelligence, pp. 42-49. Leipzig, Germany, Aug. 2017.

[14] Y. Meidan, M. Bohadana, A. Shabtai, J. Guarnizo, M. Ochoa, N. Tippenhauer, Y. Elovici "ProfilIoT: A Machine Learning Approach for IoT Device Identification Based on Network Traffic Analysis", Symposium on Applied Computing, pp. 506-509, Marrakesh, Morocco, Apr. 2017.

[15] O. El Mouaatamid, M. Lahmer, M. Belkasmi, "Internet of Things Security: Layered classification of attacks and possible Countermeasures", Electronic Journal of Information Technology, No. 9, pp. 25-37, 2016.

[16] J. Kumar M. Zaveri, "Clustering Approaches for Pragmatic Two-Layer IoT Architecture", Wireless Communications and Mobile Computing, Vol. 2018.

[17] J. Kumar, M. Zaveri, "Hierarchical Clustering for Dynamic and Heterogeneous Internet of Things", Procedia Computer Science, Vol. 93, pp. 276-282, 2016.

[18] S. Zhao, L.Yu, B. Cheng, J. Chen, "IoT Service Clustering for Dynamic Service Matchmaking", Sensors. Vol 17, No 8, pp 1727, 2017

[19] J. Kumar M. Zaveri. "Clustering for collaborative processing in IoT network". Second International Conference on IoT in Urban Space, pp. 95-97, Tokyo Japan, May 2016.

[20] R. Azevedo, R. Ribeiro. "Distributed Data Clustering in the Context of the Internet of Things: A Data Traffic Reduction Approach". 23rd Brazilian Symposium on Multimedia and the Web, pp. 313-316, Gramado, Brazil, October 2017.

[21] P. Mahalle, N. Prasad, R. Prasad, "Novel Context-Aware Clustering with Hierarchical Addressing (CCHA) for the Internet of Things (IoT)," Fifth International Conference on Advances in Recent Technologies in Communication and Computing, pp. 267-274, Bangalore, India, Sept. 2013.

[22] S. Medjiah and C. Chassot, "On the enhancement of non-functional requirements for cloud-assisted middleware-based IoT and other applications (invited paper)" To be publish in the Second Workshop on Adaptive Service-Oriented and Cloud Application 30ovember 13, Malaga, Spain 2017.

[23] Open Networking Foundation, "OpenFlow Switch Specification Version 1.5.1" , March 2015.

[24] J. Medved, et al. "Opendaylight: Towards a model-driven sdn controller architecture." World of Wireless, IEEE 15th International Symposium on a Mobile and Multimedia Networks (WoWMoM), Sydney, Australia, June 2014.

[25] Floodlight [Online] Available: http://floodlight.openflowhub.org

[26] C Isaza, J Aguilar-Martin, MV Le Lann, J Aguilar, A Rios-Bolivar, "An optimization method for the data space partition obtained by classification techniques for the monitoring of dynamic processes", Frontiers in Artificial Intelligence and Applications, Vol. 146, pp. 80-87. 2006

[27] L. Morales J. Aguilar D. Chavez C. Izasa, LAMDA-HAD, an extension to the LAMDA classifier in the context of supervised learning. Submitted to publication, International Journal of Information Technology and Decision Making, pp. 1–26, 2018.

[28] IBM, "An Architectural Blueprint for Autonomic Computing", IBM White Paper 3th Ed., June 2005.

[29] J Vizcarrondo, J Aguilar, E Exposito, A Subias "MAPE-K as a service-oriented architecture", IEEE Latin America Transactions, Vol. 15, No 6, pp. 1163-1175, 2017.

[30] J. Vizcarrondo, J. Aguilar, E. Exposito, A. Subias, "ARMISCOM: Autonomic Reflective Middleware for management Service COMposition".Proceedings of the 4th Global Information Infrastructure and Networking Symposium, IEEE Communication Society, Choroni, Venezuela, Dec. 2012.

[31] P. Jianli, M. James. Future edge cloud and edge computing for internet of things applications. IEEE Internet of Things Journal, 2018, vol. 5, no 1, p. 439-449.

[32] J. F. Botía and D. Botía, On LAMDA clustering method based on typicality degree and intuitionistic fuzzy sets. Expert Syst. Appl. Vol. 107, pp. 196–221, 2018.

[33] LAMDA-RD, an extension to the LAMDA classifier in the clustering context. Submitted to publication, Expert Syst. Appl, pp. 1–38, 2018.

[34] Q. Zhang, F. Fitzek. Mission Critical IoT Communication in 5G. Social Informatics and Telecommunications Engineering, vol 159. Springer, pp 35-41, 2015

[35] G. White, V. Nallur, S. Clarke, Quality of service approaches in IoT: A systematic mapping, Journal of Systems and Software, Vol 132, pp. 186-203, 2017

[36] M, Sandesh, P. P. Railkar, P. Mahalle, Mathematical Representation of Quality of Service (QoS) Parameters for Internet of Things (IoT), International Journal of Rough Sets and Data Analysis, Vol. 4, No. 3, 2017.