Feature Interaction in Smart Environments

Luis Emanuel Neves Jesus
Federal University of Bahia Institute of Mathematics: Universidade Federal da Bahia Instituto de Matematica

Daniela Barreiro Claro (dclaro@ufba.br)
Universidade Federal da Bahia Instituto de Matematica https://orcid.org/0000-0001-8586-1042

Tatiane Nogueira Rios
Federal University of Bahia Institute of Mathematics: Universidade Federal da Bahia Instituto de Matematica

Research Article

Keywords: Feature Interaction, Internet of Things, Automate Detection

Posted Date: April 13th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-398452/v1

RESEARCH

Feature Interaction in smart environments

Luis E.N. Jesus[1], Daniela B. Claro[1*] and Tatiane N. Rios[2]

*Correspondence: dclaro@ufba.br
[1] FORMAS Research Group, Computer Science Department, Federal University of Bahia, Salvador-BA, Brazil
Full list of author information is available at the end of the article

Abstract

The Internet of Things (IoT) connects many devices daily together in the same environment. Each device may follow the set of rules from a static environment. A static environment is usually controlled by an expert who knows all the necessary rules to provide this environment. The violation of one rule can cause a feature interaction. A feature interaction occurs when two or more devices generate instability in an environment. In a dynamic environment like IoT, devices' inclusion, and exclusion make it impossible for an expert to maintain all these rules up-to-date. It is necessary to provide an automatic solution to avoid violating these rules and maintain the environment's good performance. Thus, this work introduces a new approach to detect a feature interaction in dynamic environments automatically. Almost all previous work provide static rules defined by an expert in a controlled environment to detect an interaction. However, this is not possible in dynamic environments because of the number of device interactions and the number of device connections in/out, which grow exponentially in IoT environments. We started with a lightweight systematic review to better position our research, and then we identified one gap to provide our solution. Thus, our method learns to detect the interactions based on data analysis and then automatically predict the device detections in IoT environments. Datasets were manually annotated. Experiments were performed, and results provide evidence that automatic detection of a set of device interactions is possible in similar or either in complementary domains.

Keywords: Feature Interaction; Internet of Things; Automate Detection

Introduction

With the growth of the Internet, more devices are plugged and published over the world [25]. Internet infrastructure optimizes some routines and enables users to achieve some requirements outlining a new technological reality. Some envisioning systems are now available through the Internet due to the composition of devices, such as remote lock operation, home appliances operation via a network, and remote control of smart environments. This new reality empowers the connection among devices transparently, providing a new system of devices called the Internet of Things (IoT).

Due to the diversity of smart devices, IoT defines a set of physical objects embedded with sensors and actuators, connected by wireless networks [7]. These devices are plugged into independent domain systems, such as industry, transport, health, and smart environments. Such devices' exponential growth is estimated to reach over 75 billion smart devices in 2025 [1].

The evolution of the Internet has contributed to an increase in complex information systems. Such complexity is related to the communication and interaction between numerous components that occur statically and dynamically, requiring an

effort to provide solutions with total interoperability [32]. However, the disordered growth of IoT allows the combination of devices that can generate unforeseen interactions or some side effects classified as: desirable and undesirable. These effects are called *feature interaction* [56].

Desirable *features interactions* occur when an effect modifies a behavior that improves some functionality [57]. For example, transaction management and database locking system cooperate to ensure data atomicity, consistency, isolation, and durability. Thus, it can be classified as a desirable *feature interaction* [20]. Undesirable *feature interaction* occurs when the effect provides an unstable behavior or an inconsistent data in a system [40]. Kolberg et al [27] treat undesirable *feature interaction* when a resource, when interacting with other resources, generates unexpected behavior in the environment. For example, a residence made up of security and entertainment services that share the DVD player within the security service. The process of recording a camera in the DVD player starts due to security rules. However, the entertainment service is triggered to record a TV program simultaneously, disabling the security setting. Such interaction causes an effect of assumption violation, since the activation of the entertainment service disables the security services, violating some rules, thus causing an undesirable *feature interaction* [56].

Features interactions have been a challenge for a long time [9]. Finding solutions that identify and solve such interactions is crucial to providing stable and intelligent systems capable of matching user behavior. Such solutions are subject to a critical step, which is the management of feature interactions[10]. A *feature interaction* occurs when the resource behavior is influenced by the presence of another resource (or a set of other resources) [6]. It is possible to identify a *feature interaction* when rules and interactions analyze sole resource behaviour. However, such a case can degrade due to resource failure or intervention from other resources. To avoid such scenarios, a FI can occur where there are at least two resources.

Batory et al [8] discuss an intelligent building scenario with fire and flood controls with fire sensors on every ceiling and water sensors on every floor. In a given situation, when a sensor detects a fire, sprinklers are activated. On the other hand, observing the water on the floor, the flood sensor is activated, and sprinklers are deactivated. Such interaction between fire and flood controls generates, in this scenario, a conflict of interest that causes an untreated *feature interaction* [56]. This problem can let the building on fire and, as a consequence, physical and material damage.

Several studies analyze *feature interaction* problems, but no one has done an exhaustive literature review focusing on IoT systems. However, our study is performed with a constraint: time[55]. Other systematic mapping studies have been conducted at the same time constraint. They adjectivally denominate such kind of systematic review as rapid systematic studies [54]. In this work, we follow the definition provided by Turner et al [55], designating our approach as a *Lightweight Systematic Mapping Study* (LSMS). Our LSMS provides a study on feature interactions within IoT to position our research solution in the state of the art.

Seven research questions regarding *feature interactions* in IoT were defined. Our period to retrieve articles was between 2003 and 2019. This period was based on the systematic mapping conducted by Soares et al [53], carried on the software product

line. As a result, 20 studies analyze *feature interactions*, and all of them were classified according to detection, resolution, or general analysis. Our findings identified some gaps on *feature interaction* in smart environments. Such gaps motivated us to focus on the automatic detection of feature interaction within a smart environment.

Our approach concerns a machine learning model to predict a feature interaction in a given environment automatically. Datasets from smart environments were manually annotated to provide a classification of *feature interaction*. We validate our model against a set of manual rules designed by the experts. New desirable and undesirable interactions were found by our model. Such findings were validated against our definition of feature interaction, obtaining satisfactory and valid results.

This paper is organized as follows. Section 2 presents the formal concepts of feature interactions in IoT. Section 3 describes our LSMS. Section 4 discusses some related work. Section 5 identifies some research gaps. Section 6 introduces our *feature interaction* detection model. Section 7 describes our experiments' setup. Section 8 presents our findings. Section 9 discusses some of our results. Finally, section 10 concludes and show some envisioning work.

Formal definitions

The term *feature interaction* emerged in the 1980s to describe everyday situations in telecommunications[8]. Since then, this term has been spread to several areas, such as software engineering, smart grid, automotive, IoT, and smart environments [49].

The literature presents many definitions of *feature interaction*. Such definitions address aspects that vary from resource behavior from the user's point of view. Some definitions are described as follows:

- **A feature interaction** occurs when we integrate two or more resources to produce a new product, but together they do not work as intended [24];
- **A feature interaction** is the way a resource or a set of resources modifies or influences another resource in its behavior in the system [30];
- **A feature interaction** is a situation in which two or more characteristics exhibit unexpected behavior that does not occur when the characteristics are used in isolation [5];
- **A feature interaction** occurs when a combination of specific characteristics has an unexpected influence on performance [52]; and
- **A feature interaction** is the combination of two or more services that perform correctly and individually to obtain unexpected results when combined [31].

As far as there is still no consensus regarding the definition of a *feature interaction*, we redefine it from the point of view of smart environments.

Feature Interaction for smart environments

Considering the previous definitions of *feature interactions* [22] and the characteristics of IoT, the main factor for FI in IoT is the resource [11, 51]. We redefine the resource considering the context of IoT as follows:

Definition 1 (Resource (R)) is a composition of devices (D) with expected behaviors (Ce) that individually meet the needs of the system:

$$D, Ce \models R \tag{1}$$

A *feature interaction* occurs when two or more resources R_1 and R_2 interact considering that R_1 and R_2 satisfies each individual behavior in isolation and R_1 and R_2 do not satisfy the expected behavior when combined, that is:

$$(D_1, Ce_1 \models R_1) \wedge (D_2, Ce_2 \models R_2) \not\Rightarrow D_1 \wedge D_2, Ce_1 \wedge Ce_2, R_1 \wedge R_2 \tag{2}$$

Consider the smart home scenario, with the following features: fire control (**R1**) and window control feature (**R2**). The resource **R1** is composed of two devices (smoke sensor (**D1**) and window (**D2**)), for an expected behavior (fire (**Ce1**)). The expected result of **Ce1** is the opening of **D2** in case of smoke detection by **D1**). The resource **R2** is composed of devices (**D2** and light sensor (**D3**)) and two expected behaviors (airy environment (**Ce2**) and night security (**Ce3**)). The expected result of **Ce2** is opening of **D2** in case of detection of luminosity **D3**. The expected result of **Ce3** is the closing of **D2** in case of lack of light by **D3**.

The resources **R1** and **R2** alone have the expected behaviors achieved. In the event of interaction between the **R1** and **R2** resources, an undesirable *feature interaction* of the type *Multiple Action Interaction* is generated, compromising at least one of the resources and the **D2** is in an undefined state.

Figure 1 exemplifies the occurrence of an undesirable *feature interaction* in the smart home scenario. The fire and window control features work well in isolation. However, in the interaction between resources, a *feature interaction* can be generated. Based on this circumstance, if a fire occurs at night, the fire control triggers the window to open. However, because it is in the night, the window control triggers the window's closing to maintain nighttime safety behavior. This interaction ends up interfering with the functioning of the window and, consequently, with the smart home's security.

Definition 2 (Feature Interaction in Smart environments) A *feature interaction* can occur in a smart environment when two or more resources (R_i) interact, causing unexpected behavior; that is, a resource affects the behavior of the other resources and, consequently, the behavior of the environment.

Type of FI

Regarding the type of interaction, a *feature interaction* can be classified into four types [27]. In this work, only the *Multiple Action Interaction* (MAI) was carried out.

- ***Multiple Action Interaction (MAI)***: occurs when two resources try to control the same device. Depending on the rules from the environment, this interaction can compromise at least one resource or device, which can remain in an indefinite state;

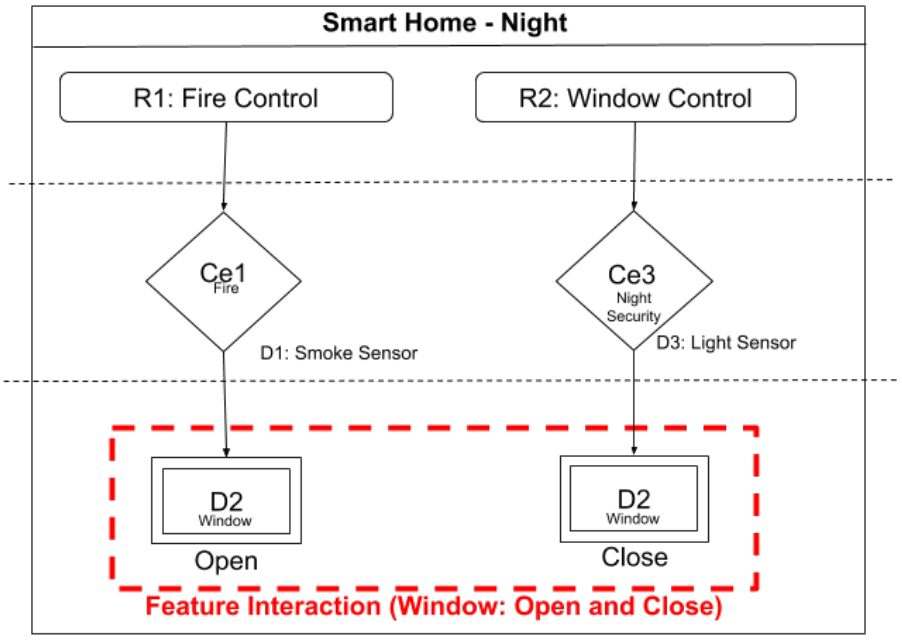

Figure 1 *Undesirable Feature Interaction.*

Considering a dynamic environment with thousands of resources (R_i) and devices (D_j) moving automatically in and out, the detection of a *feature interaction* may be hard to be followed by an expert. In this context, an investigation in the literature was carried out to evaluate methods and techniques adopted to detect a *feature interaction*.

Our Lightweight Systematic Mapping

This systematic mapping study aimed to provide an overview of the detection of FI area and to discover the gaps, authors, research groups, and its trends [44, 45].

Some systematic reviews had imposed restrictions. Turner et al [55] proposed a set of procedures to develop and document the conduction of systematic literature reviews emphasizing stakeholders with time constraints. Such particularity imposes a restriction to conduct the whole SMS process. Such SMS has achieved positive results, and it is called the *Lightweight Systematic Mapping Study*. Other approaches adopted similar restrictions but with different names. *Rapid systematic reviews*[54] aim to offer simplified alternatives to the traditional systematic review process. Some of the peculiarities in the rapid review process are focused on reaching the set of research questions. In contrast to the classic approach that focuses on more rigid steps, these approaches have provided a systematization to carry out an overview if a specific domain. Thus, in this work, we adopted the *Lightweight Systematic Mapping Study (LSMS)*.

Methodology

Our LSMS is divided into 3 phases: *planning*, *conduction* and *documentation* as depicted in Figure 2.

The *planning phase* includes the preliminary scope of the literature and aims to identify and refine the objectives of the study. A protocol is developed, and so the criteria for selecting articles. The protocol serves as the basis for the execution of the *conduction* and *documentation* phases. The protocol design was built and validated into two-fold: a) with researchers from the IoT domain and b) a manual execution of the protocol to observe which papers were retrieved.

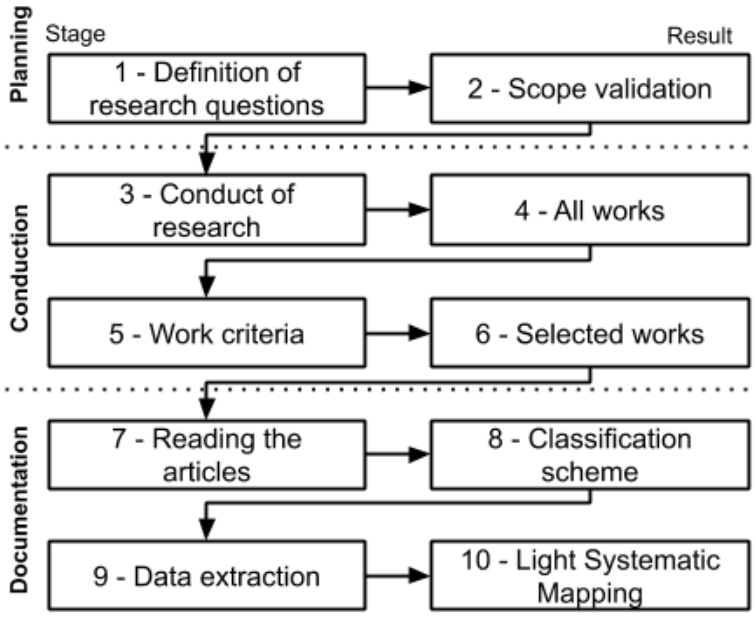

Figure 2 Our phases of LSMS (adapted from [44]).

The *conduction phase* deals with the execution of the mapping, and it is responsible for searching and selecting papers based on the research questions defined in the protocol. For this execution, two strategies for searching articles were defined: automatic search and manual search. Additional filtering was employed according to the inclusion and exclusion criteria [44].

The *documentation phase* defines a classification and information extraction scheme to answer the planning phase's research questions. Results from the detailed analysis of each primary study are presented on this lightweight systematic mapping. Our LSMS results were divided into quantitative and qualitative answers. Due to space reduction, our quantitative results were placed into Appendix .

Research Questions

The methodology followed to construct the research question was based on [47], adapting the PIOC method:

P Population: **IoT**;
I. Intervention: **Detection of feature interaction**;
O Result: **Methods and techniques for detecting features interaction**;
C Application: **Smart environments**.

This LSMS aims to **identify feature interaction approaches in IoT**, defined by the combination of the characteristics in PIOC. Consequently, seven secondary questions were identified, grouped into two-folds: three questions with *quantitative aspects* and four questions with *qualitative aspects*. The *quantitative aspects* focus on providing some statistical knowledge of FI on IoT. On the other hand, *qualitative aspects* aims to analyze the detection methods already proposed in the literature, how they were evaluated, and the domains in which FI in IoT was employed.

RQ1. *Quantitative Questions*

- **RQ1a.** *Which countries are researching feature interaction in IoT?* Conduct a survey of the countries working with feature interaction in IoT;
- **RQ1b.** *How many articles have been published in conferences and journals?* Collect the mediums which are getting more publicity on FI in IoT;
- **RQ1c.** *How many universities and/or research groups are working on FI in IoT?* Find out which universities are researching FI in IoT.

RQ2. *Qualitative Questions*

- **RQ2a.** *What feature interaction solutions have been proposed on IoT environments?* Investigate the solutions that involve the detection and management of feature interaction in IoT to bring together research communities and industries focused on such solutions;
- **RQ2b.** *What are the methods employed to detect feature interaction?* Categorize the methods for the detection of FI in IoT, addressing the levels of syntactic interoperability, semantic or pragmatic [50], being formalized through rules, access policies, formal notation, among others;
- **RQ2c.** *What are the domains that apply FI in IoT?* Examine which domains are usually provided to detect FI in IoT. This question was subdivided into three extraction possibilities: (*i*) the domain, (*ii*) the detection of specific or generic feature interaction, that is if the solution can be applied in other domains, and (*iii*) in which domains the solutions need to consider specific contexts;
- **RQ2d.** *What are the evaluation methods of feature interaction in IoT?* In this question, the objective is to investigate applied empirical research (case study, controlled experiment, and formal validation). Allowing to carry out the evaluation and validate the area's maturity in detecting feature interaction in IoT.

Research Strategy

A three-stage strategy to select the papers: **automatic search**, **manual search**, and **full paper** (Figure 4).

The first stage **automatic search** was against a set of search engines presented in Table 1 [53]. All studies published from 2003 to 2019, including conferences, newspapers, and workshops in the field of *feature interaction* in the IoT were retrieved.

The search *string* is composed of keywords according to the Kitchenham and Charters [26] method: (*i*) Feature Interaction and (*ii*) Internet Of Things. These words served as a basis to build the search *string*, adding synonyms and alternative words, applying the operators **AND** and **OR** to perform the joining of the terms, according to Table 2.

Table 1 Digital search engines.

ACM Digital Library	http://dl.acm.org/
IEEE Xplore	http://ieeexplore.ieee.org/
Scopus	http://www.scopus.com/
Engineering village	http://engineeringvillage.com/
Science direct	http://www.sciencedirect.com/
Springer	http://springer.com/

Table 2 Search string.

("feature interaction" **OR** "feature-interaction") **AND** ("Smart" **OR** "Smart City" **OR** "Smart House" **OR** "Sensor" **OR** "Actuator" **OR** "Smart Home" **OR** "Intelligent Industry" **OR** "Intelligent Home" **OR** "Internet of Things" **OR** "IoT" **OR** "FoT" **OR** "IIoT" **OR** "FoG" **OR** "Sensors" **OR** "Actuators")

A set of 661 publications was gathered after applying the *search string* (Table 1). Figure 3 presents the distribution of the articles per vehicles. Most articles were from *Springer*, followed by *Science Direct* with 56.6% and 29.5%, respectively.

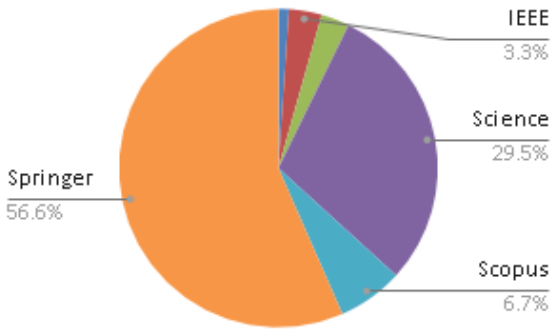

Figure 3 Percentage of publications by search engines.

All 661 publications were submitted to the inclusion and exclusion criteria, as described in Figure 4.

.

- Inclusion criteria
  - Written in English (F6);
  - Publications with search *strings* in the title, abstract or keywords (F4);
  - Publications at conferences or in newspapers (F3).
- Exclusion criteria
  - Gray literature (such as tutorials and manuals) (F2);
  - Short articles (less than six pages) (F5);
  - Studies that do not have *feature interaction* on the IoT (F7);
  - Duplicate studies (F1).

The analysis of the first filter was related to duplicate articles (F1). Due to the indexing process used by search engines, the F1 filter removed 52 articles. The second (F2) and the third filter (F3) removed 266 articles, 262 of which were removed from F2 and 4 from F3, composed of gray literature and articles published in workshops. Then, F4 filter was applied which corresponds to reading the title, abstract

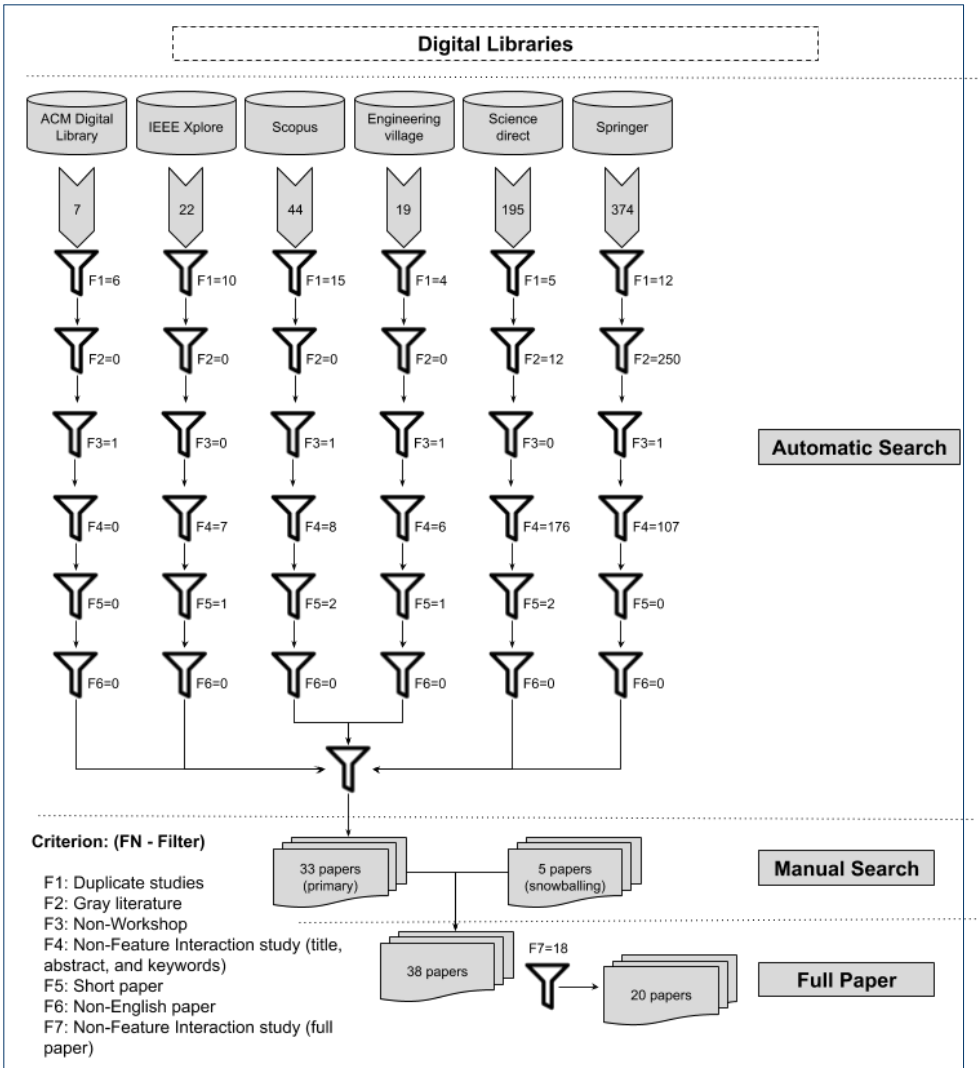

Figure 4 Filters from LSMS.

and keywords that do not have feature interactions in IoT, resulting in an exclusion of 304 publications. The F5 filter, concerning short publications contributes to the removal of 6 publications. The last filter (F6) was related to publications that were not written in the English language, where there were no removals. At the end of the first stage (**automatic search**), 95.1% of the publications were discarded, resulting in 33 articles for the second stage (**manual search**).

The second stage (**manual search**) was applied to obtain a sample of the references from the set of filtered articles. This stage was divided into (*i*) manual search in conferences and (*ii*) snowballing. The conferences were chosen by IoT domain. More 3 new publications were added to our approach. The snowballing [58], a manual search on the references of relevant studies, was conducted based on 36 pre-filtered publications (33 from the automatic search and 3 from the manual search). This process resulted in the selection of 2 more publications to be added. Thus, the **manual search** and **automatic search** summarized 38 articles for the third stage (**full read**).

The third stage (**full read**) was tackled by the F7 filter. Publications were full reading, removing manuscripts that: (*i*) are not about *feature interaction* in IoT; (*ii*) do not have any discussion on *feature interaction*; or (*iii*) do not provide an assessment of the approach. At the end of the full reading, 20 publications were selected, and made available at `https://www.bibsonomy.org/user/fi-smart-envirn`.

Our findings

As we described previously, our LSMS results were divided into two-fold: quantitative and qualitative results [16]. Due to space purposes, all *Quantitative results* were placed into Appendix .

Qualitative results

The qualitative assessment stage focuses on the interpretation of the study, with an emphasis on subjectivity [14]. In addition to allowing the understanding and context that influence the results from each article, it provides a reflection based on questions regarding the solutions, methods, domains and evaluation of each article.

Feature interaction solutions. Different solutions were developed to deal with feature interactions in different environments on IoT. In this study, five categories were identified:

- *framework* is a reusable mini-architecture that applies to the development of solutions in a specific and limited domain [48];
- *method* means the definition of systematized procedures for the description and explanation of phenomena [14];
- *algorithm* a sequence of computational steps that transforms inputs into outputs [12];
- *architecture* is a process that defines a conceptual framework of the elements of a process [41]; and
- *model* defines the flow of activities, actions, artifacts and organization of activities to be carried out [48].

Among these categories, framework and method stood out, with 70% of the solutions (Figure 5). In a framework approach, Maternaghan and Turner [35] provided a mini-architecture by a set of rules based on access policies to the set of resources in a smart home. Pedersen et al [43] describes the method through model checking, applied in a home automation environment to finding feature interactions.

Methods for detecting feature interactions. Our LSMS retrieved three main groups of methods to detect a FI in IoT:

- *algorithms* present a sequence of computational steps that transforms input(s) into output(s) [12];
- *formal notation* is a set of rigorous engineering practices, along with well-defined rules that are based on mathematical theory [19]; and
- *rules* are rules that require something to be done within the established conditions [21].

Figure 6 shows the distribution of each group. The *algorithm* [37] details the actions to perform a task.

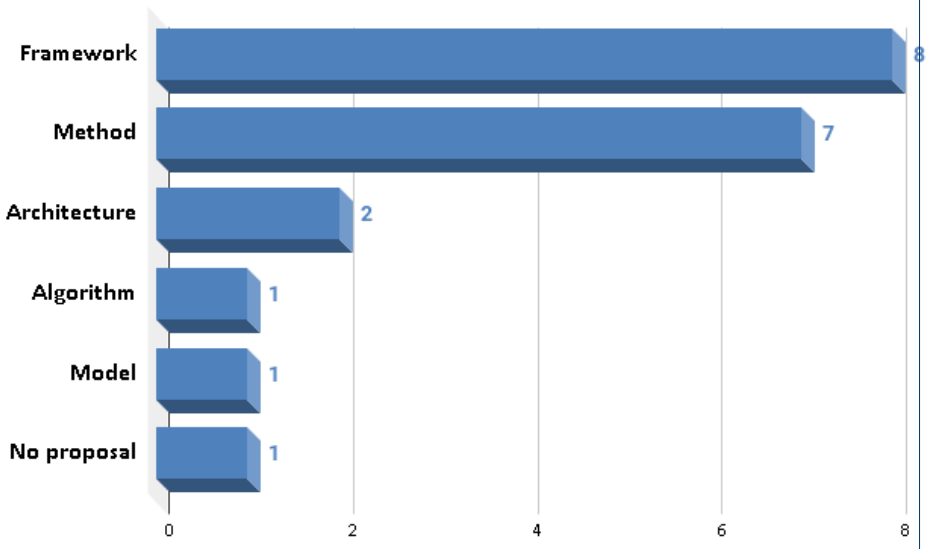

Figure 5 Solutions to detect features interactions.

The *formal notation* group corresponds to methods that adopted a formal method or a Z notation to describe their approach. The *rules* group is based on the conceptualization of standards defined by experts. Among the papers, those that presented a variation in the application of the rules were either grouped, as they follow the same concept.

IoT Domain. With the development of intelligent components and devices, several domains are perceived from the papers:

- *automotive* is a feature pack that provides advanced functionality for a vehicle [23];
- *home automation* consists of managing a set of resources for a home [43];
- *sensors* are devices that detect signals from physical phenomena (such as thermal, electrical or magnetic radiation) and convert them into digital values [33];
- *embedded systems* consist of any system that has a built-in microprocessor, with the exception of equipment easily identified as computers. This definition of embedded systems includes smart objects [29];
- *smart building* is composed of equipment to automate constructions, with the objective of facilitating the real-time monitoring and control of each function [42];
- *smart grid* is an enhanced electrical grid that collaborates with information technologies for efficient electricity distribution [15]; and
- *smart home* is a residence equipped with smart technologies designed to provide personalized services to users [34].

Figure 7 presents the most addressed domains in comparison with the solutions highlighted in the selected works. Smart home obtained 37.5% of incidences, followed by home automation with 25%.

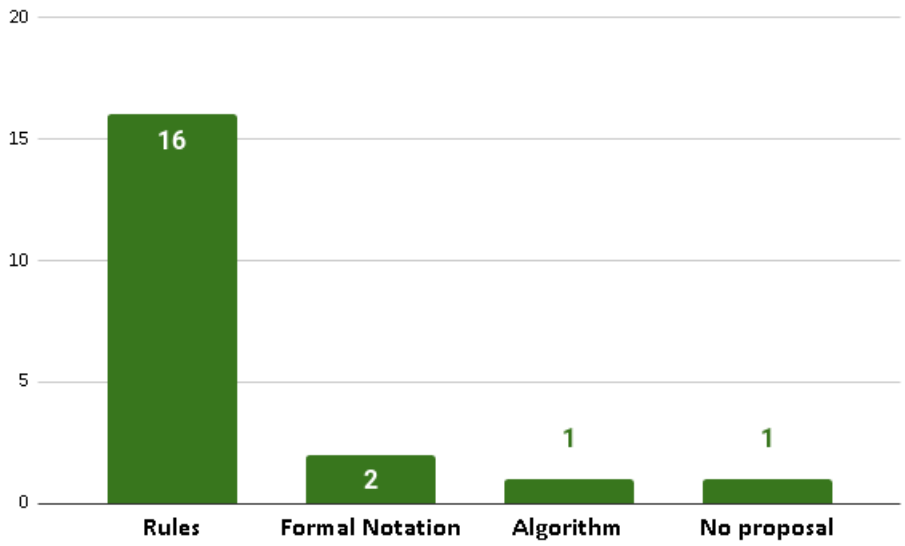

Figure 6 Detection methods of features interactions.

Among the solutions, framework and method are concentrated in the domains of home automation and smart homes. The other approaches were spread into other domains.

With the diversity of solutions and domains, it is possible to provide new approaches for detecting *feature interaction* in specific domains that can have significant results. These solutions can work with the applicability of knowledge acquisition from data.

Evaluation of feature interaction. Regarding the evaluation of feature interaction, two categories were mapped based on the 20 studies:

- *experiment* is used to support the formulation of new theories. An experiment is characterized by the composition and execution of several activities that include changing input data, parameters, programs or even their combination [36]; and
- *case study* is an empirical investigation and comprises a comprehensive method, with the logic of data planning, collection and analysis [59].

The predominant method was the case study with 13 publications, while the experiments had 5 appearances. Two studies did not evaluate their method regarding FI aspects.

Research Opportunities

With the evidence from our LSMS, the methods for detecting *feature interactions* were mostly focused on the applicability of rules and the absence of data analysis.

Looking at the scenario of smart environments, a large number of devices are connected at the same time. According to [17], in 2025, it is projected that 41.6 billion devices will be connected to the internet and will generate 79.4 zetabytes. This considerably increases the complexity of determining priority policies manually.

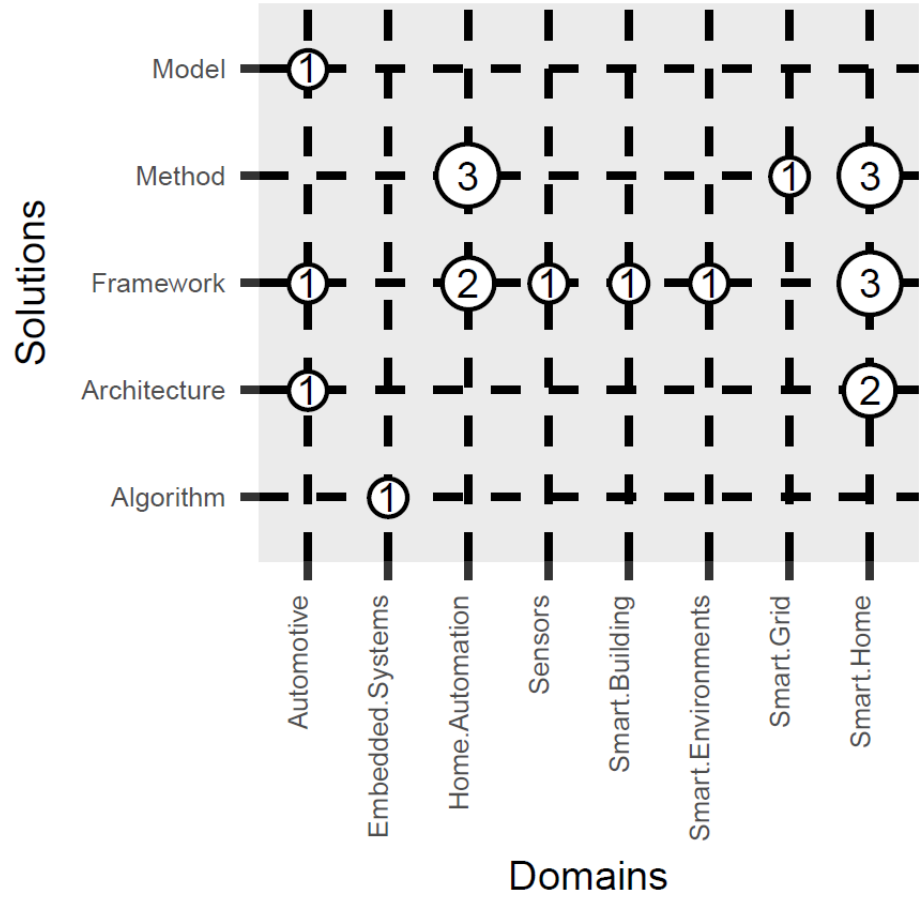

Figure 7 Amount of solution per domain.

In dynamic environments such as the Web and IoT, the inclusion and exclusion of devices make it hard to maintain manual rules up-to-date, increasing the potential for *feature interactions*. In such environments, with the inclusion and exclusion of devices and resources dynamically, there is a need to treat *feature interactions* in an automated manner. One possible way to automate is to identify patterns from the IoT datasets to detect a *feature interaction*[46] [3] [39] [28].

We observed other transversely gaps that must have an attention from the research community. The lack of basic artifacts for replicating the experiments, such as a benchmark of feature interaction is an important envisioning work to enable the evolution of FI area. Another aspect is concerning a general solution to different domains instead of having specific approaches to solve an isolated problem. Such solutions can carry out an automate manner to detect FI due to the exponential growth of devices in IoT systems.

Finally, there is no measure concerning temporal detection of a feature interaction in IoT. This is an important measure due to the mobility in IoT environments.

Concerning these research opportunities, we develop a new model to detect patterns of feature interactions in IoT datasets. We faced some challenges related to

the annotated smart environments data set, the definition of FI in the context of the IoT, as well as experts to annotate the dataset.

Feature Interaction Detection Model

Considering our previous definition of FI in IoT, we introduce our approach with the focus on data to automate the detection of patterns. This detection enables an assessment of the existence of *feature interaction* and its associations.

We follow the knowledge discovery process from [18] and we depicted all three stages in Figure 8.

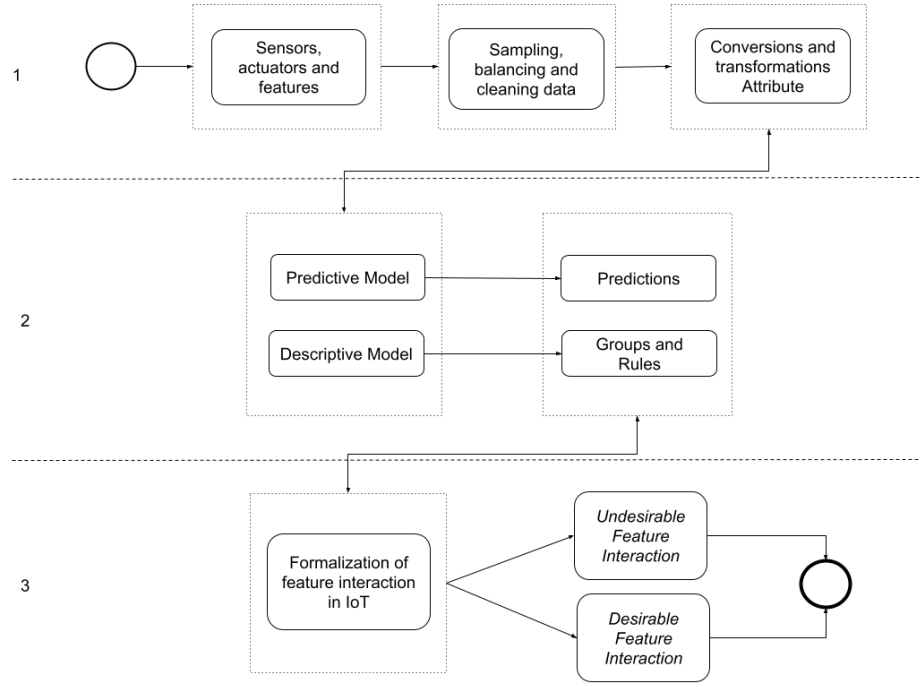

Figure 8 Method for detecting *feature interaction* in smart environments.

The first step consists of pre-processing the data. Firstly, we identified the attributes referring to devices and resources. Secondly we analysed the sampling, the balancing and data cleaning, applying techniques to incomplete and inconsistent data. Afterwards, we did a data transformation to convert and normalized some of them[18].

The second stage identifies patterns based on interpreting the predictive and descriptive model algorithms. The predictive model aims to learn from previously labeled instances by creating a model capable of predicting unknown values. The descriptive model, on the other hand, has the function of finding patterns and trends interpretable by humans to describe the data [13].

The third step aims to consolidate the knowledge discovered with the validation and verification of conflicts related to the detected *features interactions*, (“desirable”

or "undesirable"), based on the formalization adopted by the method or with the help of the rules defined by an expert.

Next sections describe our experiments to evaluate our approach.

Experiment Setup

Two experiments were carried out to evaluate the detection model thru a dataset. The first experiment consists of detecting *features interactions*, and the second of identifying new FI and validating existing ones.

The experiments were carried out in a synthetic database in the context of smart environments, in an environment capable of providing temperature, safety and pleasant energy efficiency for the possible residents and with the presence of undesirable *features interactions*.

Synthetic Dataset

Our dataset simulates a smart home made up of smart devices. An algorithm was built to input events for each device with other random events to add undesirable *feature interaction* data sets.

Based on the LSMS, all datasets were made up of binary attributes. Alemdar et al [2] explored the data to identify the pattern behavior of their residents and served as a parameter for defining the data domain of our dataset.

Our dataset consists of devices arranged in a smart home: (*i*) fan, (*ii*) curtain, (*iii*) air conditioning, (*iv*) temperature sensor, (*v*) window, (*vi*) door, (*vii*) smoke detector, (*viii*) photosensor, (*ix*) anemometer, and (*x*) rain sensor, totaling 10 devices, with random data events generated by the algorithm. The dataset contains 11 attributes, 10 of which refer to devices and resources, and one 'target' attribute, labeled by an expert. The values of the 'target' attribute of a *feature interaction* are determined to be desirable or undesirable.

The dataset consists of 12.931 observations, divided into 6.422 by events with desirable *feature interactions* and 5.969 with undesirable *feature interactions*. Seven rules were defined by the expert C_i, ($i = 1...7$), as follows, to meet the user's need.

- C_1 - *Fan*: **IF** temperature between 15°C and 24°C **THEN** Turn on the Fan;
- C_2 - *Curtain*: **IF** Wind speed greater than 20 km/h **THEN** Close the curtain;
- R_3 - *Air conditioning*: **IF** temperature greater than or equal 24°C **THEN** Turn on the Air Conditioning;
- C_4 - *Window*: **IF** rain **THEN** Close the window;
- C_5 - *Curtain*: **IF** temperature greater than or equal to 20°C **THEN** Open the curtain;
- C_6 - *Door*: **IF** smoke **THEN** Open the Door; and
- C_7 - *Window*: **IF** smoke **THEN** Open the window.

Machine Learning Algorithms

Two experiments were carried out: identification of FI (experiment **A**); and detection of rules for FI (experiment **B**). Both experiments generated a generic model to detect *feature interaction*.

To identify FI, algorithms for the classification task were employed [38]: (*i*) KNN, (*ii*) *Naive Bayes*, (*iii*) Decision tree, (*iv*) *Random Forest* and (*v*) SVM. Such algorithms are commonly employed to identify patterns from IoT data [3, 28, 39, 46].

To detect the rules, an algorithm for the association task were employed [18]: Apriori. This algorithm was adapted to finding new interactions of characteristics and validating the rules defined by the expert.

The evaluation of the **A** experiment occurred through the analysis of the confusion matrix, precision, recall and F-measure. On the other hand, in experiment **B**, the evaluation process was based on the rules generated by the algorithm and on the comparison of them with the expert's rules. Results are presented and discussed in the following sections.

Results

We present our results considering each experiment performed. Firstly, we describe the Experiment A considering the classification task and then we present our second experiment B which aims to discover the rules based on an associate task.

Experiment **A**

This experiment aims to predict the class considering if it is a desirable or undesirable FI. We provide a generic model based on a 10 fold cross-validation. Results are presented in Table 3 and we can observe that the Decision Tree based algorithm gathered the best accuracy.

Table 3 Accuracy, recall and F-measure of predictive methods from our dataset.

Algoritmos	Accuracy	Recall	F-measure
Random Forest	**0,9883**	**0,9997**	**0,9939**
Decision Tree	**0,9899**	**0,9952**	**0,9920**
KNN	0,9720	0,9725	0,9722
SVM	0,881	0,9517	0,9188
Naive Bayes	0,8658	0,8635	0,8646

We provided additional experiments to guarantee no overfitted approach. We analyze the distribution of the data in each fold of the cross-validation. Our findings make evidence that no overfitting was performed due to medians close to 1 and with low dispersion.

Results from the decision tree algorithms were above 98%, and a new investigation of data distribution within a cross-validation approach was employed. Figure 9 depicts the boxplot graph with the data distribution, which the average is close to 1, and there is a little dispersion, thus achieving a good distribution.

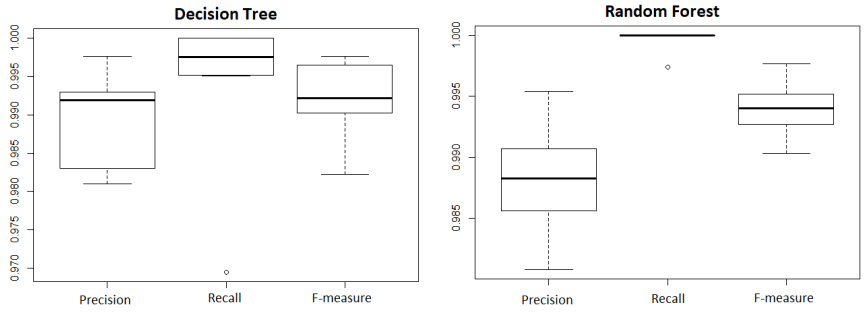

Figure 9 Distribution of the 10-folds of the algorithm Random Forest e Decision Tree

Experiment **B**

Experiment B was performed by an Associate algorithm, Apriori, which is based on finding a subset of the most frequent items in a data set [18]. The main goal of such an experiment was to compare the rule set from the dataset and the rule set defined by the expert. This algorithm enables the discovery of new *feature interactions* that have not been previously defined by the expert.

As it is a balanced dataset, built to apply the predictive model, the extraction of the rules adopted a threshold of 0.4 and 0.8 for support and confidence parameters, respectively. Nine (9) rules (Table 4) were generated taking these thresholds, of which two of these rules are the same defined by the expert (R_6 and R_7).

- R_6 - *Door*: **IF** smoke **THEN** Open the Door (associated with rule 8 in Table 4); and
- R_7 - *Window*: **IF** smoke **THEN** Open the Window (associated with rules 1 in Table 4) of the association algorithm.

Table 4 Rules generated by the association algorithm with 0.4 support and 0.8 confidence. Lines 1 and 8 indicate the aforementioned rules defined by the expert (R_6 and R_7).

	Rules
1	Smoke =>Window
...	...
8	Smoke =>Door
9	Door =>Smoke

We experimented with decreasing the threshold to catch more rules, adopting support at a rate of 0.2 and confidence at a rate of 0.6. Thus, 40 rules were generated with this new threshold. From this set of 40 rules, 3 of them were the same as those defined by the expert (R_3, R_6,R_7) which are (25, 32, 39) in Table 5.

- R_3 - *Air conditioning*: **IF** temperature greater than or equal 24°C **THEN** turn on the Air Conditioning (associated with rule 25 in the Table 5);
- R_6 - *Door*: **IF** smoke **THEN** Open the Door (associated with rule 39 in Table 5); and
- R_7 - *Window*: **IF** smoke **THEN** Open the Window (associated with rule 32 in Table 5).

Table 5 Rules generated by the association algorithm with 0.2 support and 0.6 confidence. Rule 3 indicates an undesirable *feature interaction* and rules 25, 32 and 39 indicate the rules defined by the expert (R_3, R_6 and R_7).

	Rules
1	Rain, Smoke =>Door
2	Door, Rain =>Smoke
3	Rain, Smoke =>Window
...	...
25	Temperature = Warm =>Air conditioning
...	...
32	Smoke =>Window
...	...
39	Smoke =>Door
40	Door =>Smoke

In addition, the model detects a new undesirable *feature interaction* at rule 3 (in green) in Table 5, involving rain and smoke sensors. The interaction between these sensors generates a *feature interaction* of the type *Multiple Action Interaction*, since the smoke detection service triggers the window opening and the rain detection service triggers the window closure.

Discussion

The association and decision tree algorithms validated the rules defined by an expert. In addition to providing the detection of *feature interaction* in a smart environment, this approach envisions the possibility to create a generic model of *feature interaction* detection for different domains. Such models provided interoperability in smart environments based on data through machine learning algorithms, without relying on architecture or a set of rules defined by an expert to obtain a stable environment.

Results were based on a synthetic dataset that may have generated an undesirable bias. It is important to note that the origin of this dataset is real. The problem is that the generation of real datasets must cause undesirable *feature interaction*, so it is necessary to set up a real environment with inclusion and exclusion of devices dynamically and then evaluate more complex rules.

The dataset analysis indicated that the occurrence of *feature interaction* is low, in which the detection of *feature interaction* can be addressed in data anomaly.

Another relevant aspect is related to dataset agglutination, that is, the union of different domain datasets from smart environments. The challenge stands out in combining the attributes of different domains by building a model capable of detecting *feature interaction* without reapplying the learning process and adjustments in the model.

Conclusion

We provide a lightweight systematic mapping with an overview of the *feature interaction* approaches in the IoT domain, delimiting the construction of a *feature interaction* detection model from data.

The detection model used classification and association algorithms to detect *feature interaction* in IoT environments through data analysis. The model can help domain experts adjust the rules, mainly detecting undesirable *feature interaction*.

The analysis for the detection of *feature interaction* was performed manually. In a real environment, this analysis would be carried out by an expert, a complex task when dealing with millions of thousands of devices working together. As future work, we intend to improve our approach to consider rules with a significant probability and provide a benchmark dataset to guide other works on this challenging task.

Regarding the aspects inherent to the detection of *feature interaction* in smart environments, the present work demonstrated that it is possible, with the acquisition of knowledge from a set of data, to detect *feature interaction* without the presence of an expert. In addition, other transversal contributions can be listed: (*i*) a detection model of *feature interaction* based on the data, (*ii*) a data set annotated with *feature interaction* undesirable, and (*iii*) a rule detection model with the purpose of assisting domain experts to improve the addressed environment.

Appendix A

Quantitative results

Authors in [4] define that the data in a quantitative search must be expressed with numerical measures focusing on the relationship between variables. Based on these

variables, issues related to countries, conferences, journals, universities, and research groups investigating feature interactions in the IoT were analyzed.

Regarding the countries that are developing research with features interactions in IoT, 60.65% of the publications are from Asia. Each country is counted from the researcher's publication. These results condition Japan and China (with 18 and 10 publications, respectively) as references in this research area (Figure 10).

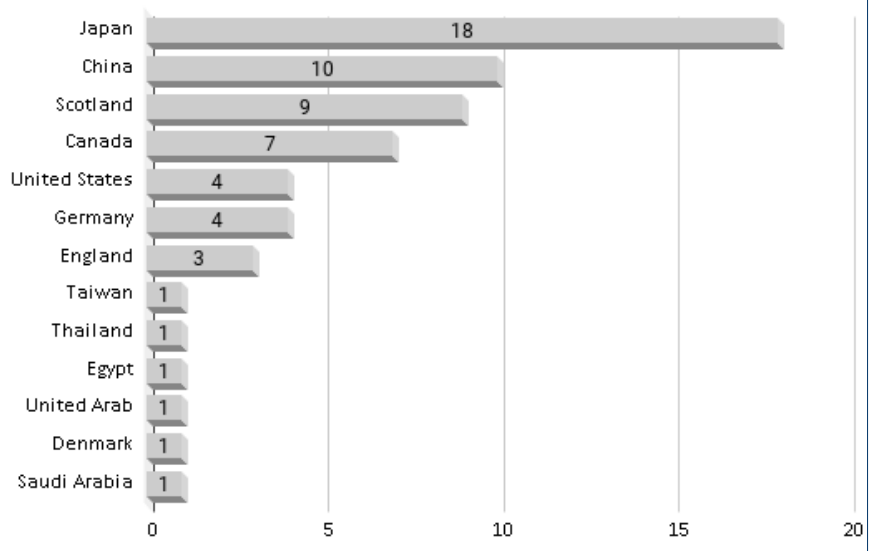

Figure 10 Number of countries with research in the area of feature interaction.

Regarding the year of publication, Figure 11 shows the distribution of articles over the years. Studies were published in journals, magazines, and conferences from 2003 to 2019.

In 2008 there was a higher incidence of publications, followed by the years 2007, 2011, 2015, and 2017 with two publications per year; that is, the research area is evolving. However, this evolution is timid since only 1 or 2 articles were published annually among those selected. Considering the paper venues, we observed that 65% of the publications occur in journals (Table 6).

Table 6 Journals of selected articles

Feature Interactions in Software and Communication Systems IX
Electronic Communications of the EASST 2008
Computer Networks 2004
IEICE TRANSACTIONS on Information and Systems
IET communications
arXiv
IEEE Transactions on Emerging Topics in Computing
IEEE Communications Magazine
Pervasive and Mobile Computing
Computer Networks
Science of Computer Programming
Transactions on Emerging Telecommunications Technologies

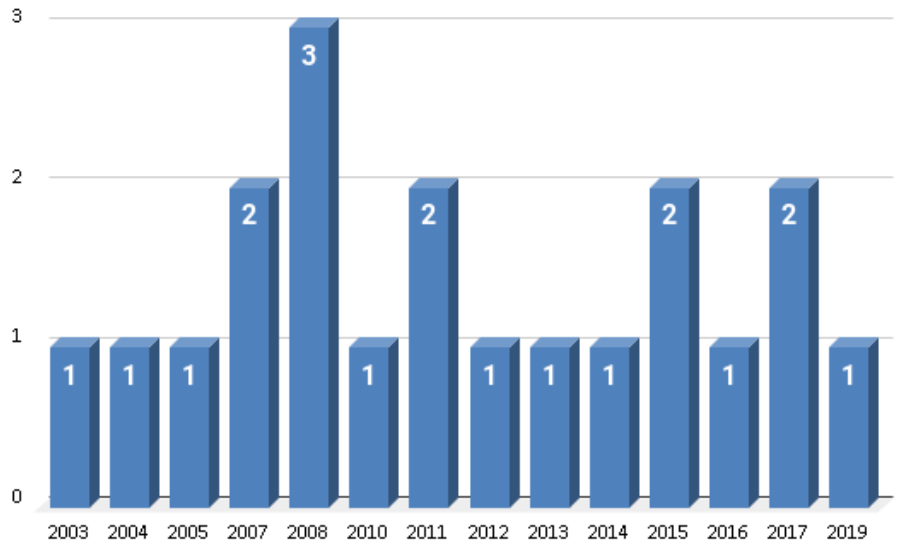

Figure 11 Number of publications per year.

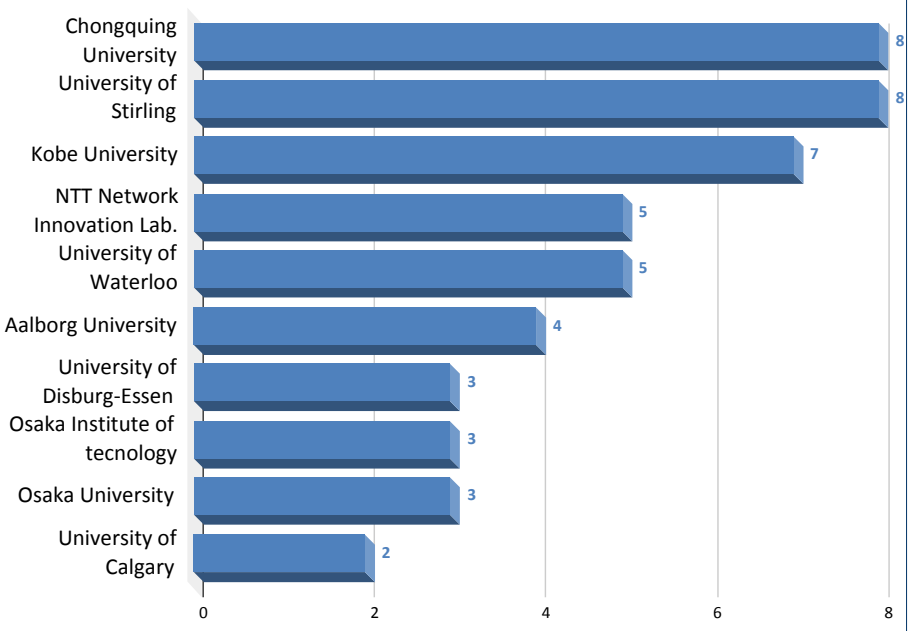

Figure 12 Universities and Group researching feature interaction.

In publications by universities or research groups, quantitative aspects were explored to obtain an overview of where the techniques and solutions are being developed and evolved. The evaluation process to define the institutions was based on the number of researchers linked to each institution. Figure 12 stands out: *Chongquing University* and *University of Stirling*.

In the quantitative aspects, an annual variation of publications, particularly in journals, the concentration on the Asian continent, and either their universities and laboratories consolidating the evolution of the *feature interaction* research area in the context of IoT.

Acknowledgements
Not applicable

Funding
Not applicable

List of Abbreviations
IoT - Internet of Things
FI - Feature Interaction
LSMS - Lightweight Systematic Mapping Study
MAI - Multiple Action Interaction

Availability of data and materials
The datasets generated and analysed during the current study are available in the FORMAS Github repository, `https://docs.google.com/spreadsheets/d/1XC0k4oUniha63_TvrKxehhsMqdFkfe_s/edit#gid=1596292566`

Ethics approval and consent to participate
Not applicable

Competing interests
The authors declare that they have no competing interests.

Consent for publication
All authors consent for publication.

Authors' contributions
LENJ analyzed and interpreted data regarding the feature interaction. DBC and LENJ discussed the formal definitions and the LSMS. TNR and LENJ performed the evaluation from machine learning algorithms. DBC and LENJ was a major contributor in writing the manuscript. All authors read and approved the final manuscript.

Authors' information
Not applicable

Author details
[1] FORMAS Research Group, Computer Science Department, Federal University of Bahia, Salvador-BA, Brazil.
[2]CINO - Computational Intelligence and Optimization Research Lab, Computer Science Department, Federal University of Bahia, Salvador-BA, Brazil.

References

1. Alam T (2018) A reliable communication framework and its use in internet of things (iot). International Journal of Scientific Research in Computer Science, Engi-neering and Information Technology (IJSRCSEIT), ISSN 13(10):2456–3307
2. Alemdar H, Ertan H, Incel OD, Ersoy C (2013) Aras human activity datasets in multiple homes with multiple residents. In: 7th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth), IEEE, Venice, Italy
3. Alsouda Y, Pllana S, Kurti A (2019) Iot-based urban noise identification using machine learning: Performance of svm, knn, bagging, and random forest. In: Proceedings of the International Conference on Omni-Layer Intelligent Systems, pp 62–67
4. de Andrade Marconi M, Lakatos EM, et al (2002) Técnicas de pesquisa. São Paulo: Atlas
5. Apel S, Kästner C (2009) An overview of feature-oriented software development. Journal of Object Technology
6. Apel S, Kolesnikov S, Siegmund N, Kästner C, Garvin B (2013) Exploring feature interactions in the wild: the new feature-interaction challenge. In: 5th International Workshop on Feature-Oriented Software Development (FOSD), pp 1–8
7. Atzori L, Iera A, Morabito G (2010) The internet of things: A survey. Computer networks 54(15):2787–2805
8. Batory D, Höfner P, Kim J (2011) Feature interactions, products, and composition. In: 10th ACM International Conference on Generative Programming and Component Engineering (GPCE), pp 13–22
9. Bowen TF, Dworack F, Chow CH, Griffeth N, Herman GE, Lin YJ (1989) The feature interaction problem in telecommunications systems. In: 7th International Conference on Software Engineering for Telecommunication Switching Systems (SETSS), pp 59–62
10. Cameron EJ, Velthuijsen H (1993) Feature interactions in telecommunications systems. IEEE Communications Magazine 31(8):18–23
11. Classen A, Heymans P, Schobbens PY (2008) What's in a feature: A requirements engineering perspective. In: 11th International Conference on Fundamental Approaches to Software Engineering (FASE'08), Springer, Berlin, Heidelberg
12. Cormen TH, Leiserson CE, Rivest RL, Stein C (2002) Algoritmos: teoria e prática. Elsevier

13. da Costa Côrtes S, Porcaro RM, Lifschitz S (2002) Mineração de dados-funcionalidades, técnicas e abordagens. PUC, Rio de Janeiro
14. Dalfovo MS, Lana RA, Silveira A (2008) Métodos quantitativos e qualitativos: um resgate teórico. Revista interdisciplinar científica aplicada 2(3):1–13
15. Desai S, Alhadad R, Chilamkurti N, Mahmood A (2019) A survey of privacy preserving schemes in ioe enabled smart grid advanced metering infrastructure. Cluster Computing 22(1):43–69
16. Diehl AA, Tatim DC (2004) Pesquisa em ciências sociais aplicadas: métodos e técnicas. Pearson Brasil
17. Eden Estopace (2019) Idc forecasts connected iot devices to generate 79.4zb of data in 2025. URL `https://futureiot.tech/idc-forecasts-connected-iot-devices-to-generate-79-4zb-of-data-in-2025/`
18. Faceli K, Lorena AC, Gama J, de Leon Carvalho ACP, et al (2011) Inteligência Artificial: Uma abordagem de aprendizado de máquina. LTC, Rio de Janeiro
19. Gibbins P (1990) What are formal methods?, Butterworths, pp 278–290
20. Haerder T, Reuter A (1983) Principles of transaction-oriented database recovery. ACM computing surveys (CSUR)
21. Heck LA (2000) O modelo das regras e o modelo dos princípios na colisão de direitos fundamentais. Direito e democracia 1(1):10
22. Hu H, Yang D, Fu L, Xiang H, Fu C, Sang J, Ye C, Li R (2011) Semantic web-based policy interaction detection method with rules in smart home for detecting interactions among user policies. IET communications
23. Juarez-Dominguez AL, Day NA, Joyce JJ (2008) Modelling feature interactions in the automotive domain. In: 11th International Workshop on Models in Software Engineering (MODELS), pp 45–50
24. Khoshmanesh S, Lutz RR (2018) The role of similarity in detecting feature interaction in software product lines. In: 29th International Symposium on Software Reliability Engineering Workshops (ISSREW), IEEE, Memphis, TN, USA
25. Kiljander J, D'elia A, Morandi F, Hyttinen P, Takalo-Mattila J, Ylisaukko-Oja A, Soininen JP, Cinotti TS (2014) Semantic interoperability architecture for pervasive computing and internet of things. IEEE access 2(1):856–873
26. Kitchenham B, Charters S (2007) Guidelines for performing systematic literature reviews in software engineering. Tech. rep., Keele University and Durham University
27. Kolberg M, Magill EH, Wilson M (2003) Compatibility issues between services supporting networked appliances. IEEE Communications Magazine 41(11):136–147
28. Kopp M, Pevnỳ T, Holeňa M (2020) Anomaly explanation with random forests. Expert Systems with Applications 149:113,187
29. Kozuch M, Wolfe A (1994) Compression of embedded system programs. In: 2nd International Conference on Computer Design: VLSI in Computers and Processors (ICCD), pp 270–277
30. Lettner M, Tschernuth M, Mayrhofer R (2011) Feature interaction analysis in mobile phones: on the borderline between application functionalities and platform components. In: 9th International Conference on Advances in Mobile Computing and Multimedia (MoMM '11), ACM, New York, NY, USA
31. Liu Y, Meier R (2008) Feature interaction in pervasive computing systems. Electronic Communications of the EASST 11(1):7
32. Maciel RSP, David JMN, Claro DB, Braga R (2017) Full Interoperability: Challenges and Opportunities for Future Information Systems, SBC, pp 107–118
33. Magill E, Blum J (2016) Exploring conflicts in rule-based sensor networks. Pervasive and Mobile Computing 27(3):133–154
34. Marikyan D, Papagiannidis S, Alamanos E (2019) A systematic review of the smart home literature: A user perspective. Technological Forecasting and Social Change 138(1):139–154
35. Maternaghan C, Turner KJ (2013) Policy conflicts in home automation. Computer Networks 57(12):2429–2441
36. Mattoso M, Werner C, Travassos G, Braganholo V, Murta L, Ogasawara E, Oliveira F, Martinho W (2009) Desafios no apoio à composição de experimentos científicos em larga escala. Seminário Integrado de Software e Hardware, SEMISH 9(1):36
37. Medina M, Ferting C (2006) Algoritmos e programação: teoria e prática. Novatec Editora
38. Mitchel T (1997) Machine Learning. McGraw-Hill
39. Nakhodchi S, Upadhyay A, Dehghantanha A (2020) A comparison between different machine learning models for iot malware detection. In: Security of Cyber-Physical Systems, Springer, pp 195–202
40. Nhlabatsi A, Laney R, Nuseibeh B (2008) Feature interaction: The security threat from within software systems. Progress in Informatics
41. de Pádua Paula Filho W (2003) Engenharia de software. LTC
42. Park H, Rhee SB (2018) Iot-based smart building environment service for occupants' thermal comfort. Journal of Sensors 2018(1):10
43. Pedersen T, Guilly TL, Ravn AP, Skou A (2015) A method for model checking feature interactions. In: 10th International Joint Conference on Software Technologies (ICSOFT), pp 1–10
44. Petersen K, Feldt R, Mujtaba S, Mattsson M (2008) Systematic mapping studies in software engineering. In: 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), pp 68–77
45. Petersen K, Vakkalanka S, Kuzniarz L (2015) Guidelines for conducting systematic mapping studies in software engineering: An update. Information and Software Technology 64:1–18
46. Petnik J, Vanus J (2018) Design of smart home implementation within iot with natural language interface. IFAC-PapersOnLine 51(6):174–179
47. Petticrew M, Roberts H (2008) Systematic reviews in the social sciences: A practical guide. John Wiley & Sons
48. Pressman R, Maxim B (2016) Engenharia de Software-8ª Edição. McGraw Hill Brasil
49. Reiff-Marganiec S, Ryan MD (2005) Feature interactions in telecommunications and software systems VIII. IOS Press
50. Ribeiro ELF, Monteiro EL, Claro DB, Maciel RSP (2019) A conceptual framework for pragmatic interoperability. In: Proceedings of the XV Brazilian Symposium on Information Systems, Association for

Computing Machinery, New York, NY, USA, SBSI'19, , URL https://doi.org/10.1145/3330204.3330246
51. Shehata M, Eberlein A, Fapojuwo A (2007) Using semi-formal methods for detecting interactions among smart homes policies. Science of Computer Programming
52. Siegmund N, Kolesnikov SS, Kästner C, Apel S, Batory D, Rosenmüller M, Saake G (2012) Predicting performance via automated feature-interaction detection. In: 34th International Conference on Software Engineering (ICSE), IEEE, Zurich, Switzerland
53. Soares LR, Schobbens PY, do Carmo Machado I, de Almeida ES (2018) Feature interaction in software product line engineering: A systematic mapping study. Information and Software Technology 98(6):44–58
54. Taylor-Phillips S, Geppert J, Stinton C, Freeman K, Johnson S, Fraser H, Sutcliffe P, Clarke A (2017) Comparison of a full systematic review versus rapid review approaches to assess a newborn screening test for tyrosinemia type 1. Research synthesis methods 8(4):475–484
55. Turner M, Kaur R, Brereton P (2008) A lightweight systematic literature review of studies about the use of pair programming to teach introductory programming. In: PPIG, p 21
56. Weiss M, Esfandiari B, Luo Y (2007) Towards a classification of web service feature interactions. Computer networks 51(2):359–381
57. Wilson M, Magill EH, Kolberg M (2005) An online approach for the service interaction problem in home automation. In: 2th IEEE Consumer Communications and Networking Conference (CCNC), IEEE, Las Vegas, NV, USA
58. Wohlin C (2014) Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: 18th International Conference on Evaluation and Assessment in Software Engineering (EASE), p 38
59. Yin RK (2015) Estudo de Caso-: Planejamento e métodos. Bookman editora

Figures

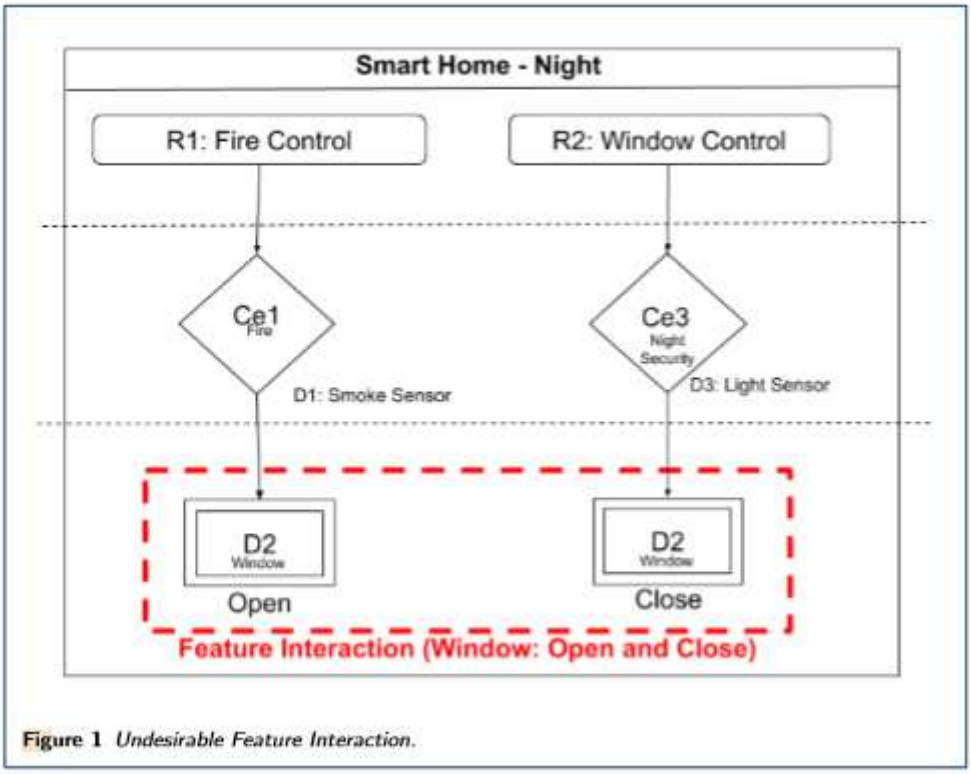

Figure 1 *Undesirable Feature Interaction.*

Figure 1

Undesirable Feature Interaction.

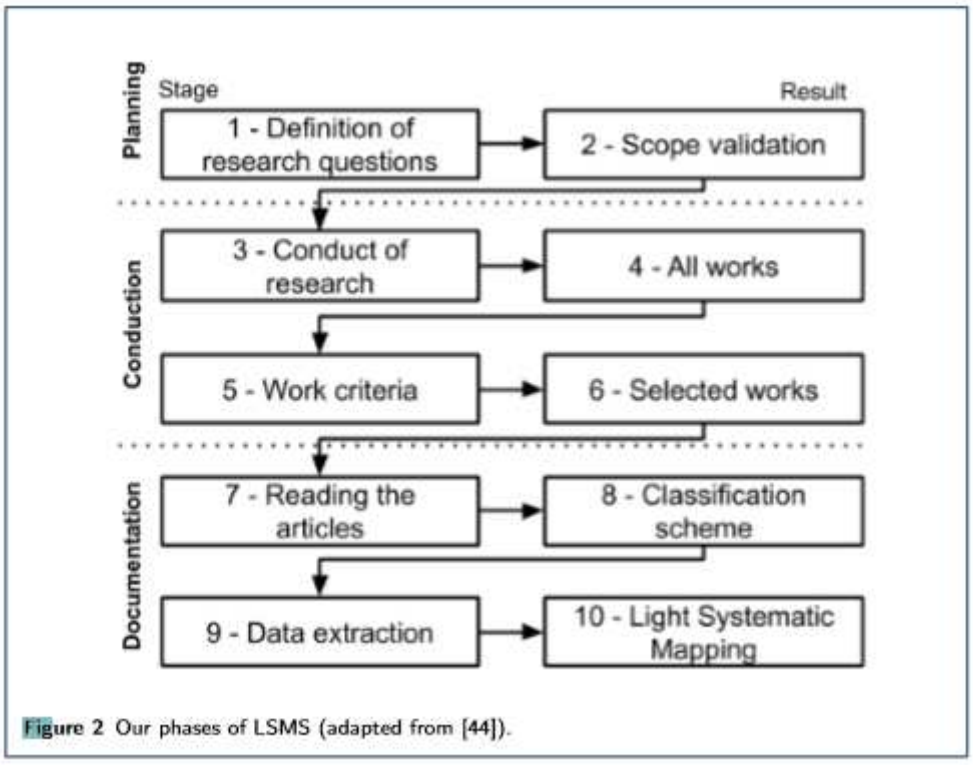

Figure 2 Our phases of LSMS (adapted from [44]).

Figure 2

Our phases of LSMS (adapted from [44]).

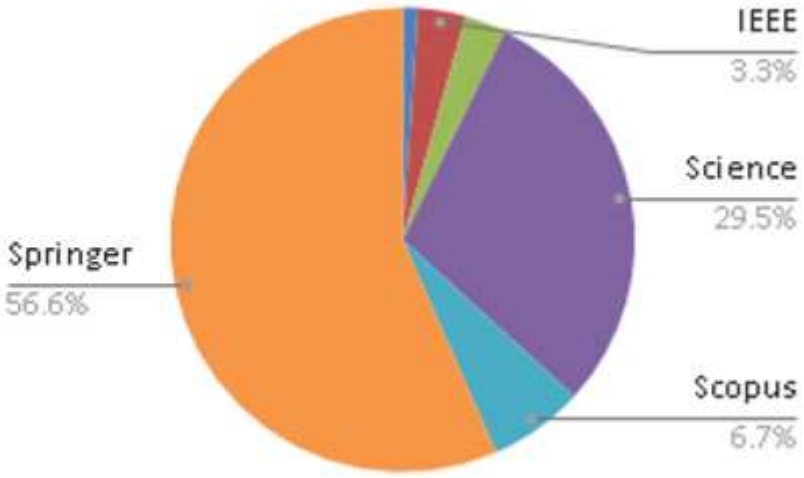

Figure 3

Percentage of publications by search engines.

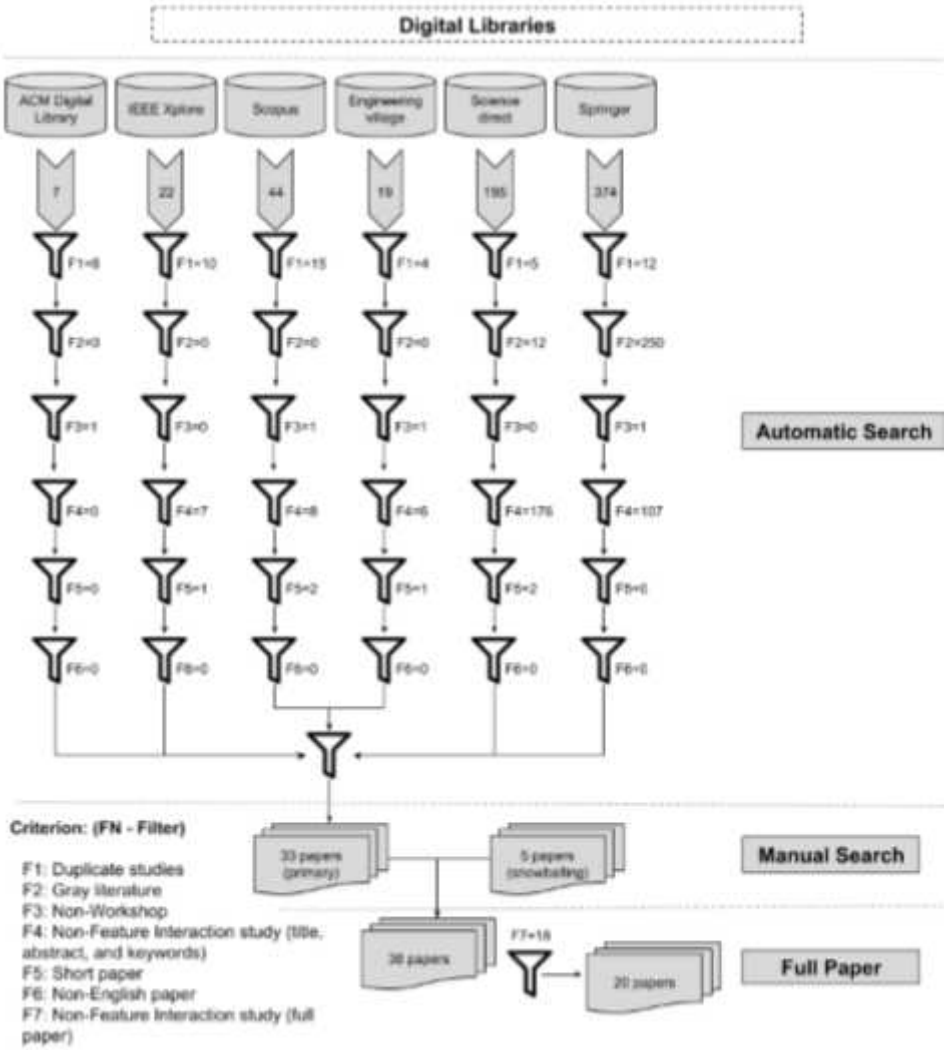

Figure 4

Filters from LSMS.

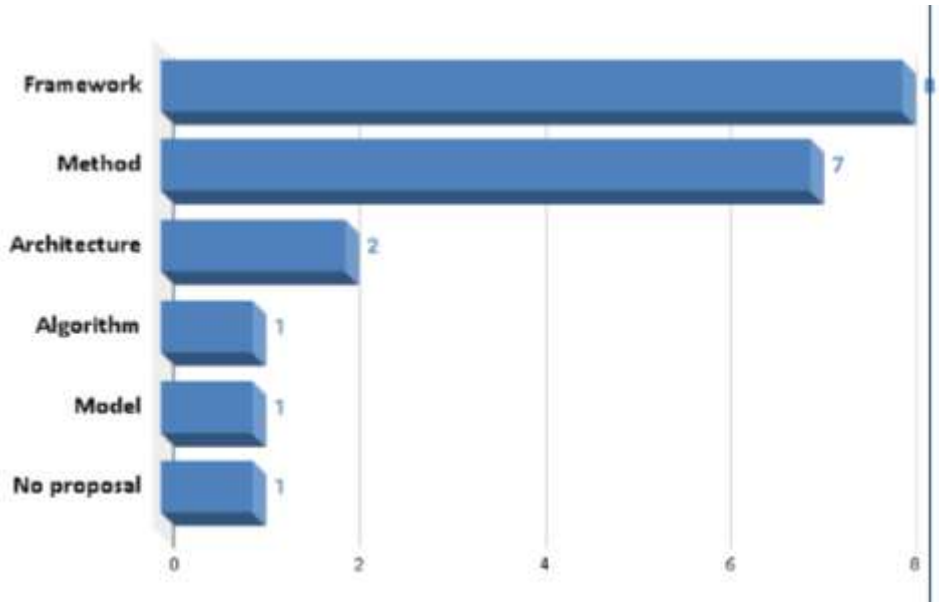

Figure 5

Solutions to detect features interactions.

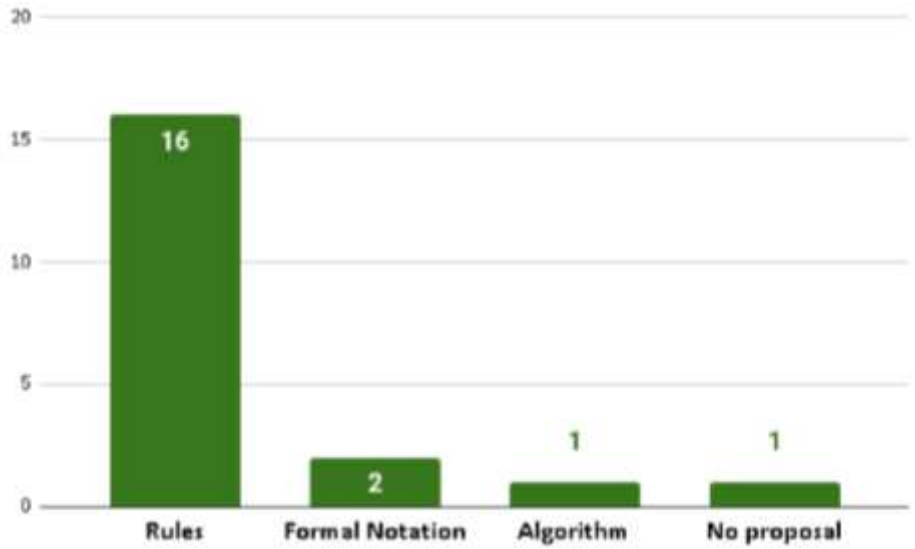

Figure 6

Detection methods of features interactions.

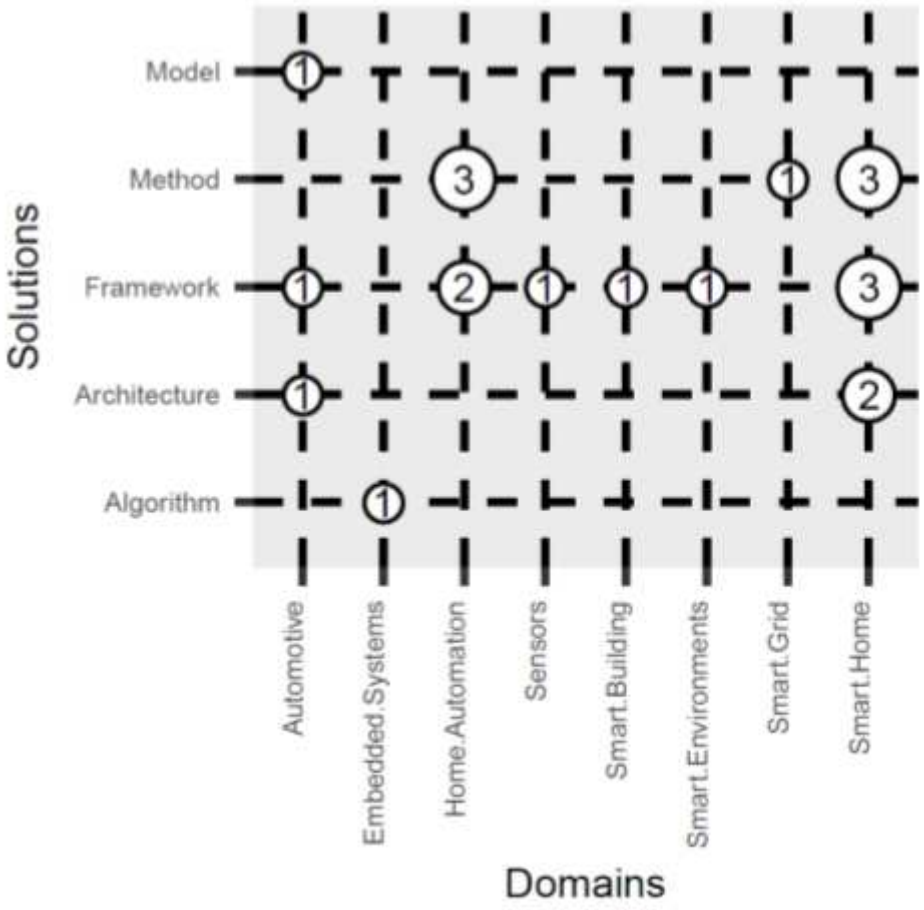

Figure 7

Amount of solution per domain.

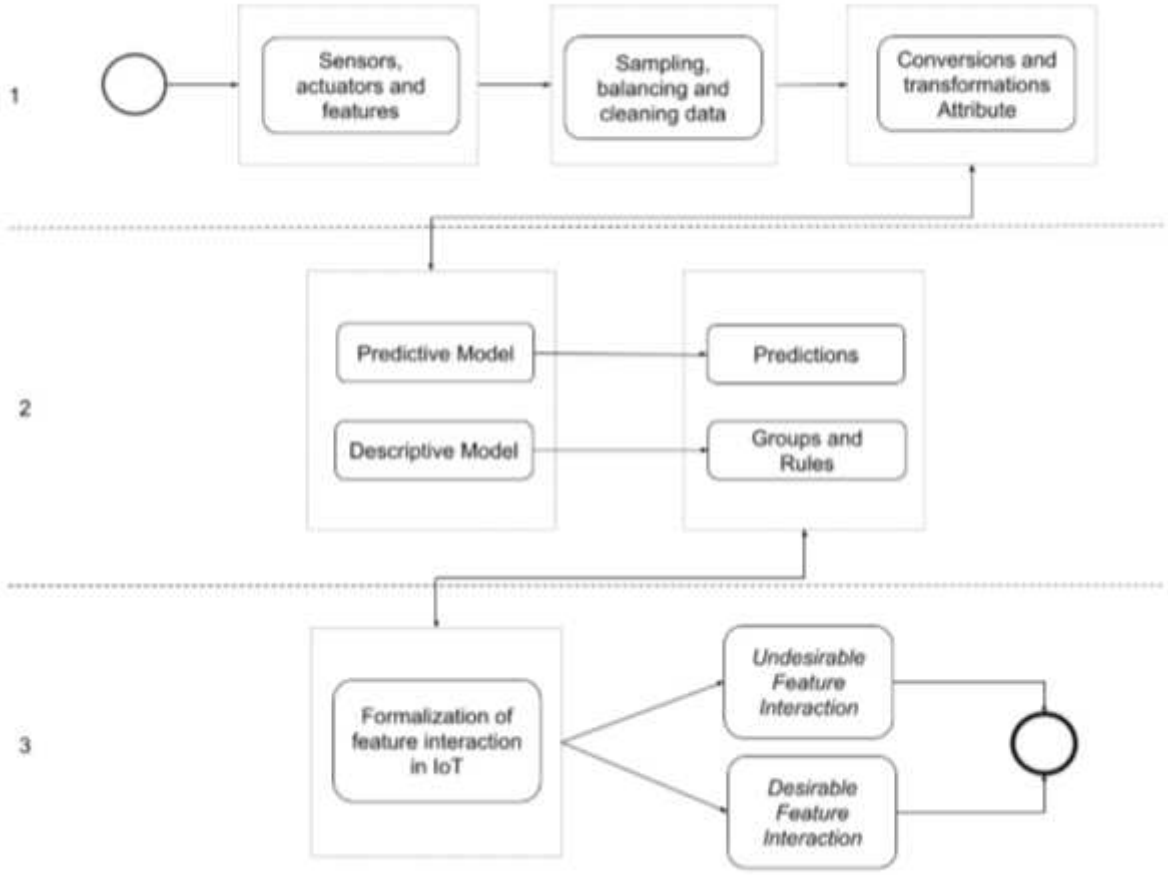

Figure 8

Method for detecting feature interaction in smart environments.

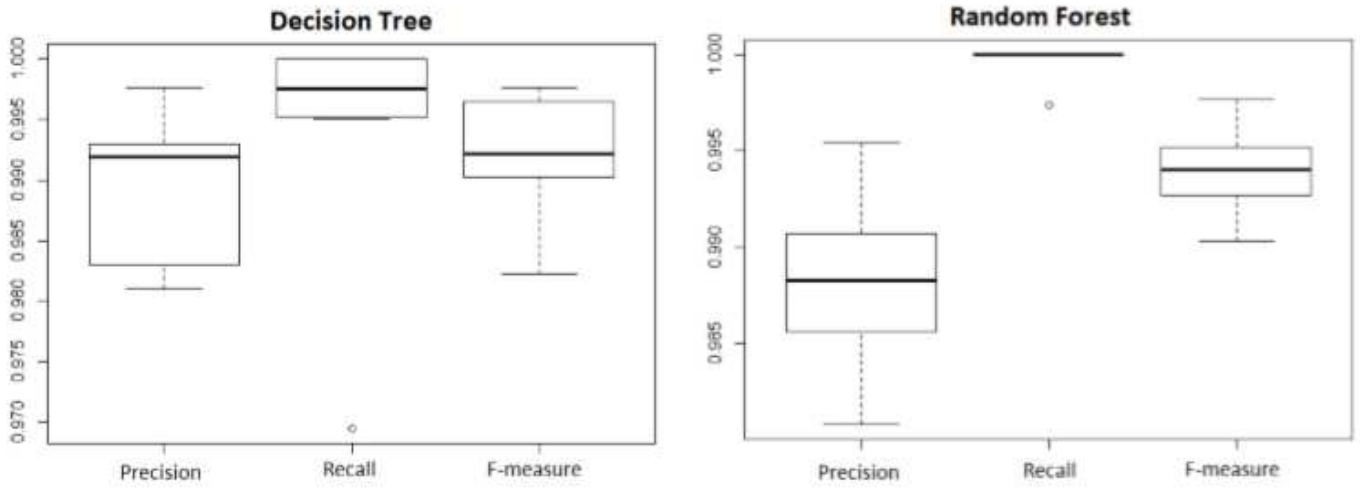

Figure 9

Distribution of the 10-folds of the algorithm Random Forest e Decision Tree

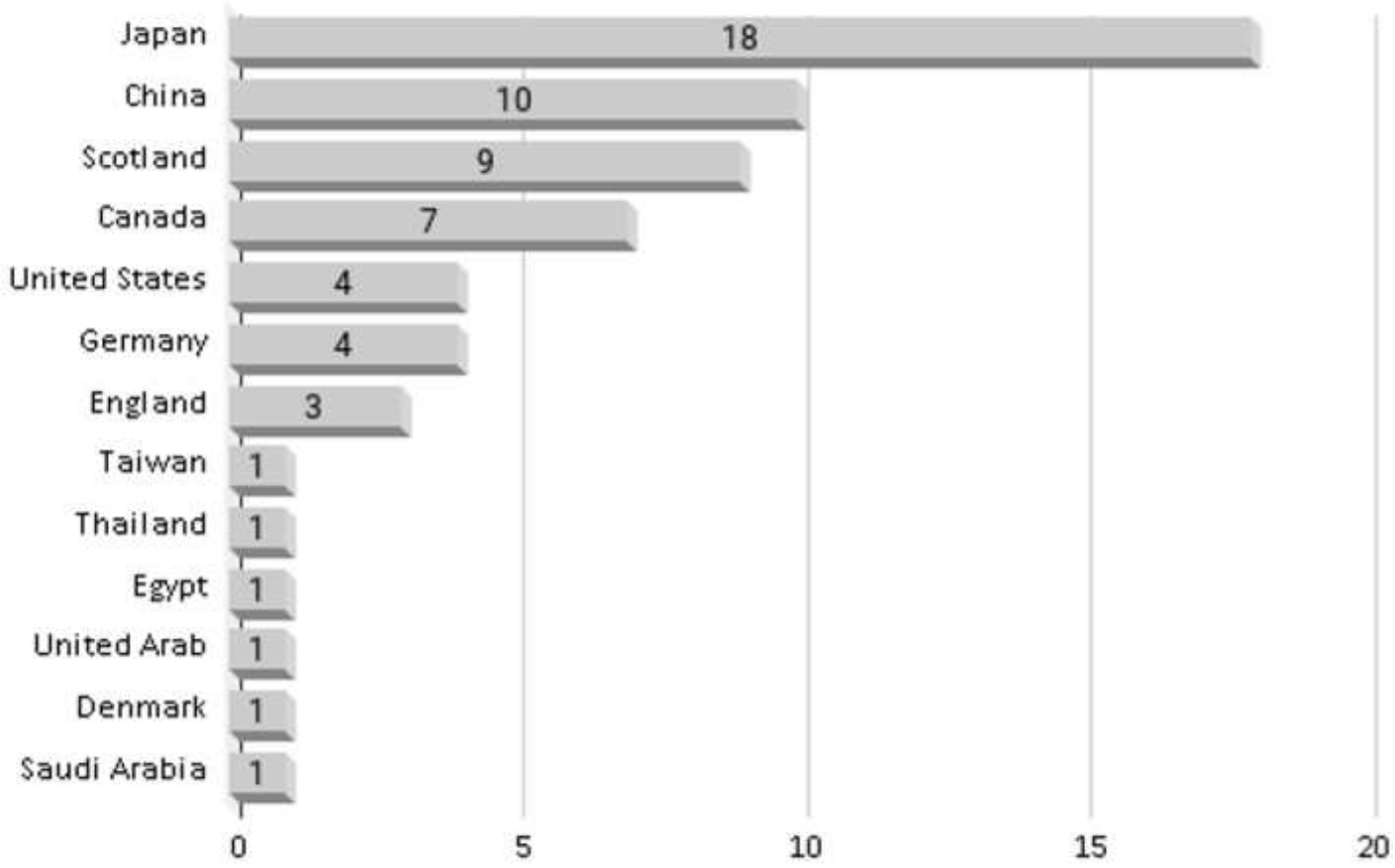

Figure 10

Number of countries with research in the area of feature interaction.

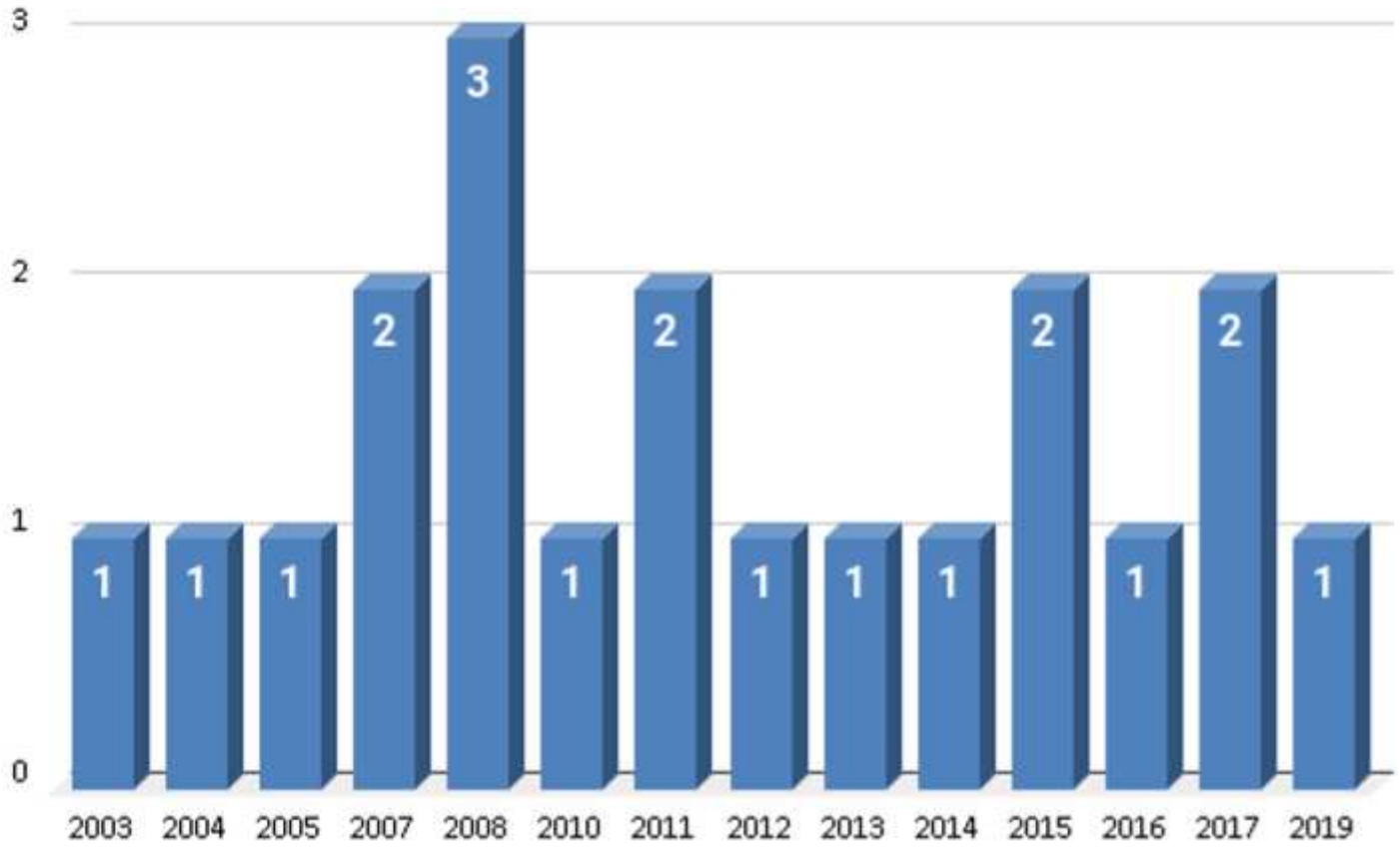

Figure 11

Number of publications per year.

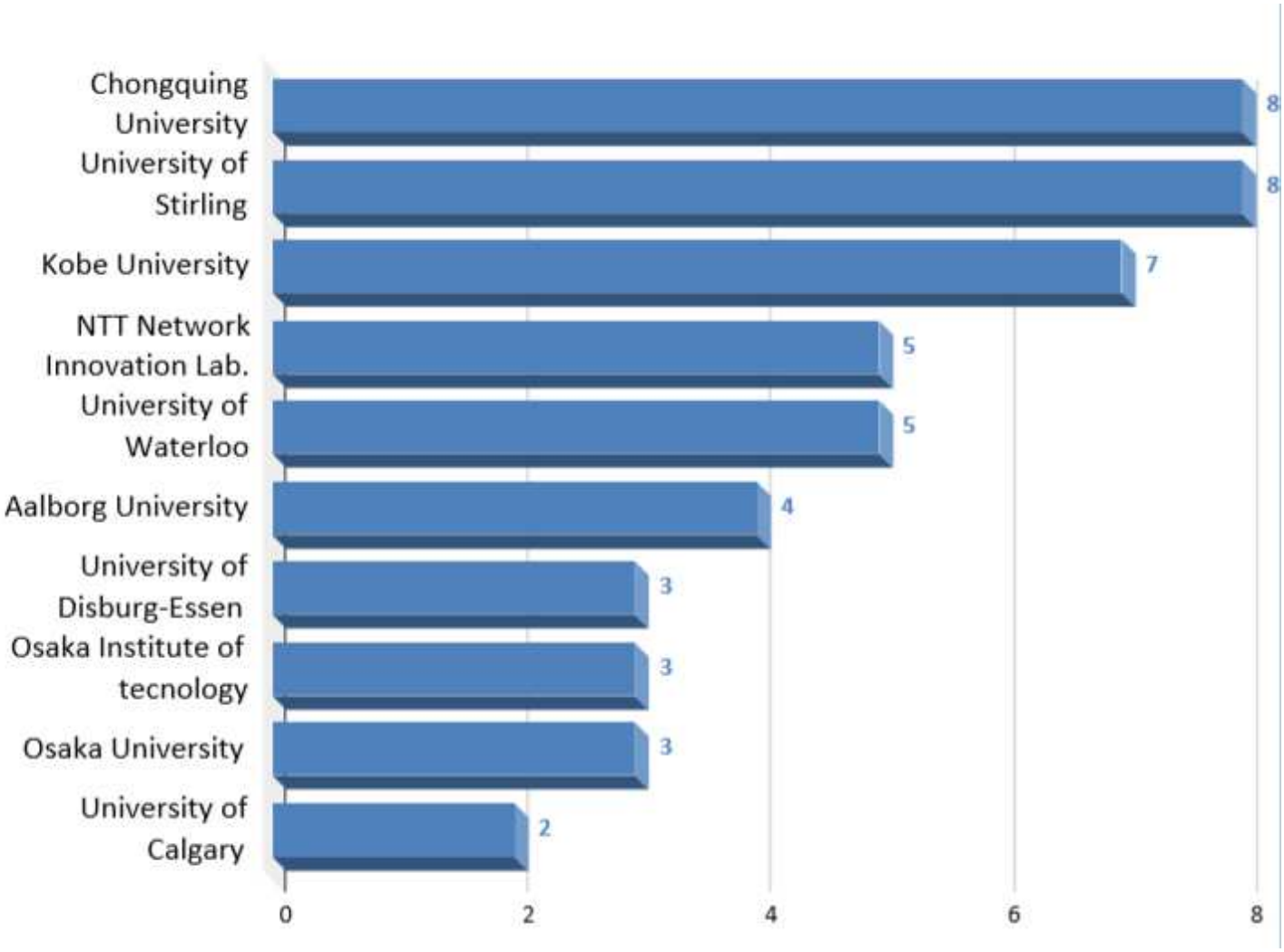

Figure 12

Universities and Group researching feature interaction.