Automating the Interactions among IoT Devices using Neural Networks

Javier Rojo*, Daniel Flores-Martin*, Jose Garcia-Alonso*, Juan M. Murillo* and Javier Berrocal*
*University of Extremadura
Caceres, Spain
Email:{javirojo, dfloresm, jgaralo, juanmamu, jberolm}@unex.es

***Abstract*—The number of Internet of Things (IoT) devices is growing at an unstoppable pace. In most cases, the proper functioning of these devices requires human intervention. Due to this growth, people will have to configure more and more devices leading to the investment of a considerable effort and time. Nowadays, some works try to automate the user actions with IoT devices by using machine learning algorithms based on the relationships among devices or the previous human behaviour. However, these proposals do not make use of contextual information obtained from the devices themselves or they employ complex sets of prediction models. This paper proposes a neural network-based solution to predict the devices behaviour by using previous interactions and contextual information as input variables. Thanks to this, device interactions can be predicted and automated, even without having previous records, by knowing behaviours of devices of the same type in similar environment.**

***Index Terms*—IoT Device interactions, Context-awareness, machine learning, neural networks.**

I. Introduction

IoT devices make people's lives easier by offering them improvements in their daily routines [1]. This is one of the reasons why these devices have a high penetration in many environments such us *smart homes*, *smart cities* or *smart offices*, for instance. However, although in so many cases these devices simplify people's lives, nowadays, they usually require a direct command from the user each time a person needs to perform an action or to change the defined configuration.

This supposes an effort that increases when the number of devices in the environment raises, to the point that the perceived benefit of applying the IoT paradigm is not the expected one. To manually change a single configuration is not hard but, when the number of devices and the interactions among them increase, the required effort increases overwhelmingly. Based on some predictions [2], in 2025 will be 75 billion IoT devices, which means an average of 9 devices per person, and even more in the developed countries. Therefore, managing these devices and their interactions will require a great effort [3].

In order to reduce this effort, different approaches have been proposed for automating some of these tasks. For instance, by analyzing the relationships between devices [4] [5] or by defining custom operating models for each device [6], but again each time an interaction changes, the models have to be reconfigured. Other works are based on machine learning algorithms that are able to predict and adapt future interactions with the devices based on the previous interactions [7] [8]. Nevertheless, these proposals make use of complex systems that combine different techniques and employ each one depending on the environment and the number of the devices, reducing the responsiveness and increasing the computational load.

Therefore, in this paper we propose a neural network-based solution for predicting the interactions among IoT devices that can be used in almost any environment. To that end, it makes use of contextual information provided by the devices themselves such as its kind, how they interact, the status of the environment, etc. From this information, the neural network is able to predict actions and the interaction among devices when the contextual information changes (users, time, devices, specific values, etc.).

One of the advantages of this solution is that the predictions can be made with a small amount of information and, every day, the model can be refitted with new records. Thanks to this, the system can re-adapt the predictions to the changes and user's needs. The solution has been tested with semi-synthetic data and in a real environment for three weeks obtaining a probability of success of 85.34%, highly reducing the effort required to interact and reconfigure these devices when the user's needs change.

The rest of the document is structured as follows. Section II shows the motivations of this work and some related works. Then, Section III describes the proposed neural network-based solution. Section IV details the validation performed to evaluate the proposed solution. Finally, in Section V the conclusions of this work are exposed.

II. Motivations and Related Works

IoT devices allow users to automate their daily tasks. There are IoT devices for a wide range of tasks. However, when the number of IoT devices with which the user must interact grows a lot, the number of actions and configurations to be performed also grows, being difficult to manage all these devices. To better understand this problem, an example is proposed below:

Suppose an user named Peter, who is an enthusiastic of technology and interacts with IoT devices in their common environments. Every time Peter arrives home, he turns on the light and the television in the kitchen. Peter feels the need for these devices to recognize that he arrived home and automatically do these tasks for him. He tried to program their actions at a certain time, but since his work schedule

is flexible, he does not always arrive home at the same time, which means that the devices are turned on before or after the appropriate time. After dinner, he usually goes to his living room, where he has a bulb with a dim light and turns on the television to watch his favourite program. However, during the last few weeks he is starting to read, so that every day he has to manually turn the television off and increase the luminosity (or change the configuration). These manual interactions make Peter to start thinking that the deployment of these intelligent devices does not provide any advantage.

As shown above, there is a need for approaches learning from the user behaviour and from the contextual information in order to automate these actions without the need for the user to configure and reconfigure them whenever her/his needs change. Thus, the user can focus on performing actions that are not usual or do not follow a pattern of behaviour over time. This would reduce the number of actions done by the user, something important in environments with a growing number of IoT devices [2]. Therefore, technological alternatives for predicting future actions of the user, based on their previously registered behaviour and the contextual information, should be developed.

During the last few years, for the device interactions prediction, different techniques and methodologies have been proposed. This is an important topic in researches linked with IoT, especially in Ambient Assisted Living (AAL) [9], a paradigm oriented to improve people quality of life using Information and Communication Technologies (ICT). In this sense, Machado et al. [4] propose a solution using ontologies for establishing relationships between devices of the same environment. Ciocarlie et al. [5] also propose a solution for predicting behaviours using action links between devices in the same environment. In addition, Paternò et al. [6] propose the definition of customizable rules that can be used in trigger-action programming paradigms. These rules can be customized by any user and they will be triggered by a machine learning system to recommend future rules for the device. Other proposals are focused on the prediction of device interactions by using *context-awareness* information instead of the relationship between devices. Alhafidh et al. [7] [8] propose some comparisons about different machine learning classification models, by using them in an isolated way or assembling them. However, chaining different models require for these proposals to find a balance between the system responsiveness and the accuracy of prediction. Both dimensions are crucial for IoT applications [10].

The work presented in this paper is focused on the use of contextual information to predict the interactions that will be performed among IoT devices; using a single model. Besides, the proposed solution has been evaluated in five real scenarios obtaining very good accuracy in all of them.

III. Predicting the interactions

The proposed solution is divided into four sections: a) first, the synthetic data were generated to evaluate the technological alternatives to make the predictions; b) then, each alternative was evaluated with the generated data in order to know which one provides the best results in a simulated environment; c) next, the best technique was selected and adapted in order to get the best predictions regarding the responsiveness and the accuracy; and d) finally, the developed prototype was evaluated in five real scenarios.

A. Generation of the synthetic data

To measure the available algorithms and techniques, it is required a dataset to train the models and evaluate them. Currently, in the moment of developing this work, there is no available dataset with the contextual characteristics that this proposal requires for automating the IoT interactions (users present, type of devices, connection type, environment status, etc.). So that, it is necessary to generate our own data. Generating synthetic data is the quickest method to start collecting data that serve us to identify the technique or the set of techniques more appropriate to solve our problem [11].

Synthetic data was generated to make a first validation in order to choose the best prediction model. These data are generated with TheONE Simulator [12]. Initally, The is aimed to to simulate message routing between nodes with various Delay-tolerant Networking (DTN) routing algorithms and sender and receiver types. Nevertheless, although its objective is different, this tool was chosen because it can simulate different movement patterns, among which are the movement patterns of a person in his day-to-day life through different environments (office, home, car,...). This tool allowed us to simulate the interaction among IoT devices in different environments. In order to generate the synthetic data its *core* library has been modified to make the simulation more realistic.

In the TheONE Simulator, a series of environments were defined (such as the one defined in Section II), in which there were different types of devices that can be used to perform different actions. The action to be performed is selected based on a probability and also is based on the time of day, the context of the environment and the context of the user.

In a first version of the data generation, a relationship between the probabilities of performing each action with a device and the time, was not stipulated –only some contextual information about the environment and the devices were considered– (Fig. 1.a). For instance, the probability of turning on the kitchen oven was the same at mealtime than at 3 in night. This resulted in a poor generation of synthetic data, which had to be modified

to obtain more realistic data. In a second step, the time was considered as another contextual characteristic together with the device information and the contextual data (Fig. 1.b).

In the data collection, certain data from the environment have been collected (*context-awareness* information), such as the day of the week, the time at which the action is performed (rounding to the nearest o'clock time) and a timestamp (which are not used still in the selected model); and device data, such as its identifier and its device type. Also, the performed actions were always recorded. In total, we simulated 163.039

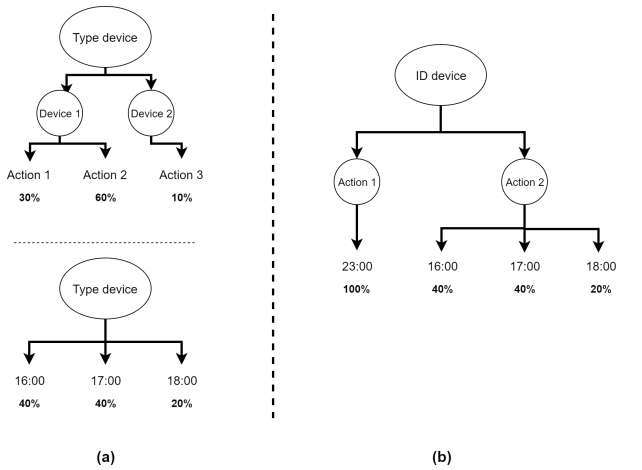


Fig. 1: Synthetic data generation: (a) Without relating actions and hours; (b) Relating actions and hours.

interactions among IoT devices, in four different environments, with a total of 14 devices. We simulated the interactions of an user in his/her everyday environments with devices such as smart bulbs, speakers or plugs, for example.

B. Machine Learning algorithms

Different machine learning techniques have been analyzed to predict and automate the device interactions, based on contextual data and previous human behaviours with the same devices or devices of the same type.

Before analyzing the different machine learning models, it is necessary to understand the data pre-processing. Thanks to this, better performance in predictions can be obtained. In this process, all fields were serialized, based on their different values, in binary fields.

Once the data were pre-processed, different techniques were evaluated to identify their accuracy and responsiveness. To increase the responsiveness and reduce computation load, we are focused on using a single model. The evaluated models and their results can be seen in Table I. Please, note that these results are with synthetic data. Once they are applied in real scenarios, the accuracy can be slightly lower since some actions can be executed without following a predefined pattern.

TABLE I: Table of accuracy per model.

ML Model	Accuracy	Delay (s)
Logistic Regression	0.5329949	10.92
SVM	0.9999182	∞(>600)
Decision Tree	0.6666667	0.0019
Random Forest (80)	0.9999386	22.58
KNN (n=17)	0.9999182	247.60
Neural Network	0.9996013	163.73

As can be seen in Table I, Random Forest, KNN and Neural Networks provide promising results. For KNN, we evaluated the optimal number of neighbours (from K=1 to K=30) in terms of accuracy, obtaining K=17. However, the algorithm was quite slow (it took 30 minutes to find the optimal number of neighbours and be trained with the synthetic dataset), without offering so much superior accuracy to other alternatives studied. Therefore, it was discarded.

Because of the performance and the short time needed to train the model and make predictions, we decided to use neural networks. Random Forest also offers good times, however, in other tests performed, where the number of devices increased and the relationships were more complex, its accuracy decreased with respect to the neural network. The definition of this neural network will be explained in section III-C. Combined models will not be used, as proposed by other solutions, because the performance obtained with this single model is good enough and it reduces the computational load.

C. Neural Network for predicting interactions

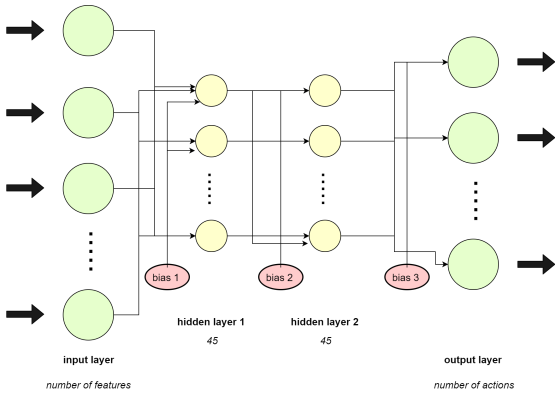


Fig. 2: Neural network definition.

The neural network (Fig. 2) has as many input neurons as characteristics our data have (after obtaining their *dummies* or serializing the attributes to binary fields). The number of outputs offered by the network was equal to the number of different actions that exist in the devices. When new devices are added, the number of outputs also change, and the network is redefined. The network gives as output the probability that each action is the action to be performed, after applying the *softmax* function, that convert output values to probability values (softmax output sum of values must be 1, that indicates that the probability of every outputs is 100%). A *argmax* function is applied to that output to obtain the most probable action. To the rest of the layers, a *sigmoid* function as activation function is applied. The *sigmoid* functions are more specifically defined as a squashing function. Squashing functions limit the output to a range between 0 and 1, making these functions useful in the prediction of probabilities. All these functions are used, therefore, to be able to predict the action or the interaction that should be executed among IoT devices.

The number of hidden layers and the number of neurons for these layers were decided after testing different combinations, and trying to reduce the probability of *overfitting*. Also, it has been added a *bias* neuron in each layer, to reduce the bias

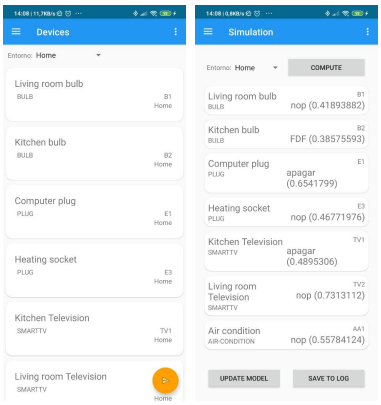
of data. This was determined by using the synthetic dataset generated.

To train the model, it was decided to divide the data with a *batch_size* of 100, making 3000 epochs. This number of iterations is necessary, as it has been stipulated a very low learning rate (*0.000125*) for the selected learning algorithm (*AdamOptimizer*). This algorithm was used because it is well suited for problems with large quantity of data and it requires little memory resources. A *sparse softmax cross-entropy with logits* was used for training. This is a method employed when the prediction is based on a classification problem with probabilities. The value of the coefficient has been set based on the graph of the loss coefficient generated with the loss value of each epoch [13].

To do the implementation of this neural network the technology used was *TensorFlow*. This is due to the generated models are compatible with their execution in mobile devices with *TersorFlow Lite* [14].

D. Prototype

A prototype has been developed that allows us to validate the predictions performed by the selected model to automate the device interactions. In this sense, a client-server architecture has been chosen for the prototype, consisting of an Android client and a Python server. A server in the *cloud* is used to define and train the model. The trained model is downloaded by the Android application to be able to predict the interaction of that device with the rest of IoT devices in the environment. This allows us to predict and automate the interaction of the owner of the mobile device.



(a) (b)

Fig. 3: App views: (a) IoT devices of each environment; (b) Simulation of device interactions predictions.

In this sense, a specific model is created for each user/mobile device. The generated model is linked to the user using her/his *Google Advertising ID*.

The developed application can execute the defined model and simulate the interactions with IoT devices. Currently, it can simulate different environments with different not-real IoT devices (Fig. 3a). With these devices, the different actions can be performed manually and saved in a registry. This registry is used for refitting the model every day. Then, the model is used to automate the interactions with the devices. If an interaction is incorrectly predicted, the user can correct it (Fig. 3b). This information is used to provide feedback to the model in order to improve future predictions. All the information generated by the application is stored in a log (such as the *context* of the prediction, the predicted action, if the action was corrected, the correct action, etc.). That log was used to evaluate the performance of the selected techniques.

IV. Validation

To validate the system, the developed application has been used by five users in their daily routine. Five different scenarios and IoT devices were defined where each user had to use the application in those devices depending on their usual behaviour.

A. Scenarios definition

Each of the defined scenarios was used to generate metrics separately. On the one hand, three of the five scenarios were used to record information on tasks that are performed when the user arrives at a specific environment (Scenarios 1 to 3). On the other hand, the other two scenarios were used to record all the actions that the user performs when s/he already is in a specific situation (Scenarios 4 and 5). The scenarios are defined as follows:

- **Scenario 1:** A scenario that represents the behavior of a person who works from Monday to Friday morning. In the afternoon, s/he is usually at home watching TV.
- **Scenario 2:** A scenario that represents the behavior of a person who works from Monday to Friday (morning and afternoon). On Saturday s/he is at home and on Sunday s/he goes to her/his parents' home.
- **Scenario 3:** A scenario that represents the behavior of a person who works from Monday to Friday morning. In the afternoon, s/he does not follow a pattern of behavior.
- **Scenario 4:** A scenario that represents the behavior of a person who works from Monday to Friday, morning and afternoon (except Fridays). On weekends s/he goes to her/his parents' home.
- **Scenario 5:** A scenario that represents the behavior of a person who works from Monday to Friday morning. In the afternoon, s/he goes to her/his parents' home by car.

The validation lasted three weeks. The two first weeks were used for training the model with information about the different actions that the users performed with the present devices in each environment. And the last week was used for the testing phase in order to get prediction and evaluate them. Some predictions were also made during the two weeks of training, to check how much data were needed for the system to start learning new behaviours and how the predictions improved with new records. However, this depended a lot on how frequent the behaviour is.

B. Results

First, regarding the responsiveness, it is interesting to indicate that on average **11.05 seconds** are required to refit the neural network with the inputs of each user every day and **1.35 milliseconds** on average are required to make a prediction. As can be seen, the responsiveness is very good for almost any IoT application. Other similar proposals [8], are getting prediction times of 896.1 milliseconds (when the number of devices is small) and 1.21 seconds (when the number of devices is higher). Training times are not included in these proposals.

Second, regarding the accuracy, the results obtained for each scenario can be seen in Table II. In this table, the number of predictions, the correct and the incorrect ones are detailed. This table also shows the *total* row that represents the joint results for every scenario.

TABLE II: Table of predictions.

Scenario	Predictions	Correct	Failed	Accuracy
Scenario 1	109	83	26	0,76
Scenario 2	245	212	33	0,86
Scenario 3	69	64	5	0,92
Scenario 4	78	67	11	0,85
Scenario 5	72	63	9	0,87
Total	573	489	84	0,85

By analyzing the joint results for every scenario, we have to notice that 84 predictions were not correct. According to this, we detected that 76 were produced by trying to predict actions that do not have a specific behaviour pattern, since there were records of different actions for the same device in the same context, and for the same user. For example, imagine a person who sets a different television channel every day when arriving at home. If the system tries to predict which channel the person will choose, the probability of getting a correct prediction is low. For the other 8 incorrect predictions, the model was not able to learn the action. Therefore, it can be said that only 8 out of the 573 predictions were not correct.

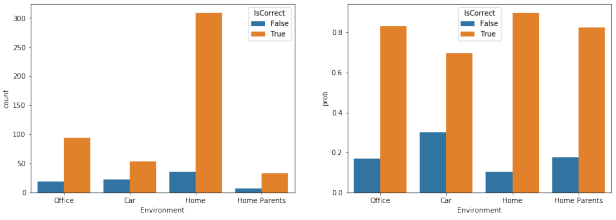


Fig. 4: Probability of predictions per environment.

Besides, if it is assumed that the number of failures is mainly due to actions that cannot be predicted, it can be observed in which environments (Fig. 4) and for what types of devices (Fig. 5) the behaviour is more changing and our model can obtain worse results. As can be seen in Fig. 4 and Fig. 5, the environment with a higher probability of failure is the *"Car"* and the type of device with the most failures

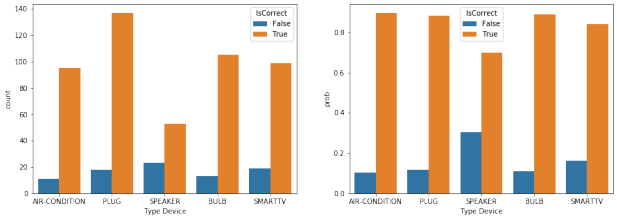


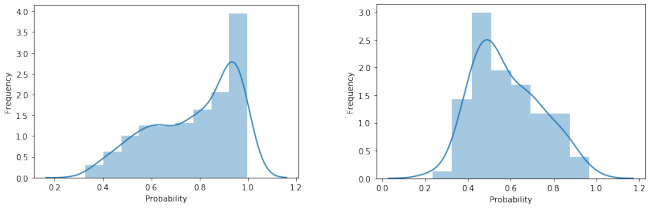
Fig. 5: Probability of predictions per type of device.

is the *"Speaker"*. Evaluating the obtained logs, we identified that this is because the users make short trips by car and the speaker is turned on and off in a short time and because they do not listen to the same type of music on the same route. In these situations, the presented system is more likely to fail.

TABLE III: Table of measures over probabilities.

Measure	Correct predictions	Failed predictions
Min. value	0.3266	0.2347
Max. value	0.9970	0.9667
Average	0.7762	0.5817
Typical Deviation	0.1763	0.1563

In addition, we also evaluated the probability of success for every prediction in order to identify a *threshold* that allows us to know when a predicted action should be performed or not. In Table III can be appreciated the average, the maximum and the minimum value and standard deviation for the probability of the correct and the failed actions.



(a) (b)

Fig. 6: Univariate distribution of probabilities: (a) for correct predictions; (b) for failed predictions.

TABLE IV: Table of measures over threshold.

Threshold	False neg.	False pos.	Correct	% Success
0.40	41	55	477	0.832461
0.42	54	44	475	0.828970
0.44	67	42	464	0.809773
0.46	76	41	456	0.795812
0.48	84	36	453	0.790576
0.50	99	34	440	0.767888

Finally, to determine the *threshold*, an *univariate* distribution has been created with the value of the probability of success of the correct (Fig. 6a) and failed (Fig. 6b) predictions. Thanks to these data, and reducing the number of false

positives even if the accuracy is reduced, it can be determined that the best *threshold* is over 0.4 and 0.5. Besides, in Table IV it can be observed that a *threshold* of 0.42 is better, because a lower *threshold* include more false positives.

In conclusion, it can be stated that the algorithm has been able to correctly predict 85% of the interactions, failing only in 1% of the predictions or when the performed actions do not follow a pattern.

V. Discussion and Conclusions

In this paper, a system that allows us to predict interactions with the IoT devices in an environment has been proposed. This approach makes use of contextual information from the environment, previous interactions and the devices' information. To do this, a neural network was used after evaluating the performance in accuracy and prediction time.

These solutions are required in order to be able to implement a true IoT paradigm in which devices are used to automate tasks instead of leading the user to be constantly interacting and re-configuring the different devices depending on people' needs. In addition, the presented system is able to learn and adapt the prediction to the new needs.

We are currently working on integrating the defined model with real and heterogeneous devices. To that end, we are combining the presented model with a framework already defined to improve the interactions among heterogeneous devices by means of semantic web [15]. It is also intended to be able to define and to train the model in the final device, without the need for a server to train the model and send it to the device to make the predictions.

Finally, we are also working on deploying this model in social environments, in which the behaviour is not the same depending on whether a person is accompanied or not, who accompanies them and what are the needs of each one.

For this experiment, the model is executed in the user's mobile device, which will be the only one performing the interactions with the real IoT devices. However, this solution will not work in social environments, where we will need a network of mobile devices that communicate with each other or a master device to which all IoT devices and smartphones are connected.

Resources

- Full validation document [16].
- Dataset of synthetic data [17].
- Datasets of actions performed by each person in validation phase [18].
- Logs of results of predictions in validation phase [19].

Acknowledgment

This work was supported by the project 0499 _4IE_PLUS_4_E funded by the Interreg V-A España-Portugal 2014-2020 program, by the project RTI2018-094591-B-I00 (MCIU/AEI/FEDER, UE) FPU17/02251 grant, by the Department of Economy and Infrastructure of the Government of Extremadura (GR18112, IB18030), and by the European Regional Development Fund.

References

[1] "How Internet of Things (IoT) effects on our daily lifestyle." [Online]. Available: https://www.futurescope.co/internet-of-things-iot/

[2] "• IoT: number of connected devices worldwide 2012-2025 — Statista." [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[3] D. Flores-Martín, J. Berrocal, J. García-Alonso, C. Canal, and J. M. Murillo, "Enabling the interconnection of smart devices through semantic web techniques," in *Web Engineering - 19th International Conference, ICWE 2019, Daejeon, South Korea, June 11-14, 2019, Proceedings*, ser. Lecture Notes in Computer Science, M. Bakaev, F. Frasincar, and I. Ko, Eds., vol. 11496. Springer, 2019, pp. 534–537.

[4] G. M. Lunardi, F. A. Machot, V. A. Shekhovtsov, V. Maran, G. M. Machado, A. Machado, H. C. Mayr, and J. P. M. de Oliveira, "IoT-based human action prediction and support," *Internet of Things*, vol. 3-4, pp. 52–68, oct 2018.

[5] G. F. Ciocarlie, B. Copos, I. Agadakos, G. Ciocarlie, T. Lepoint, U. Lindqvist, and M. Locasto, "Butterfly Effect: Causality from Chaos in the IoT," Tech. Rep., 2018. [Online]. Available: https://www.researchgate.net/publication/327515023

[6] F. Paternò and S. Alawadi, "Towards Intelligent Personalization of IoT Platforms End User Development; Internet of Things; Trigger-Action Programming ACM Reference format," Tech. Rep., 2019. [Online]. Available: https://www.researchgate.net/publication/331876007

[7] B. M. Alhafidh, A. I. Daood, and W. H. Allen, "Poster abstract: Comparison of classifiers for prediction of human actions in a smart home," in *Proceedings - ACM/IEEE International Conference on Internet of Things Design and Implementation, IoTDI 2018*. Institute of Electrical and Electronics Engineers Inc., may 2018, pp. 287–288.

[8] B. M. Hasan Alhafidh, A. I. Daood, and W. H. Allen, "Prediction of Human Actions in a Smart Home Using Single and Ensemble of Classifiers," in *SoutheastCon 2018*. IEEE, apr 2018, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/8479270/

[9] "Descripción general - Programa "Active Assisted Living" (AAL) - Portal de Ayudas del Mº de Industria, Comercio y Turismo." [Online]. Available: http://www.mincotur.gob.es/PortalAyudas/programaAAL/Descripcion/Paginas/descripcionPrograma.aspx

[10] P. Bellavista, J. Berrocal, A. Corradi, S. K. Das, L. Foschini, and A. Zanni, "A survey on fog computing for the internet of things," *Pervasive and Mobile Computing*, vol. 52, pp. 71–99, 2019. [Online]. Available: https://doi.org/10.1016/j.pmcj.2018.12.007

[11] "Generación de datos sintéticos: una habilidad imprescindible para los nuevos científicos de datos." [Online]. Available: https://towardsdatascience.com/synthetic-data-generation-a-must-have-skill-for-new-data-scientists-915896c0c1ae

[12] "Information — The ONE." [Online]. Available: https://akeranen.github.io/the-one/

[13] "CS231n Convolutional Neural Networks for Visual Recognition." [Online]. Available: http://cs231n.github.io/neural-networks-3/

[14] "How to Deploy TensorFlow to Android (feat. TensorFlow Lite)." [Online]. Available: https://medium.com/datadriveninvestor/how-to-deploy-tensorflow-to-android-feat-tensorflow-lite-957a903be4d

[15] D. Flores-Martín, J. Berrocal, J. García-Alonso, and J. M. Murillo, "Towards a runtime devices adaptation in a multi-device environment based on people's needs," in *IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2019, Kyoto, Japan, March 11-15, 2019*. IEEE, 2019, pp. 304–309. [Online]. Available: https://doi.org/10.1109/PERCOMW.2019.8730859

[16] J. Rojo, "Full validation document," 2019. [Online]. Available: https://drive.google.com/file/d/1fYYtRZuXTqkeualkG-HVdIpbqj1Koe30/view?usp=sharing

[17] ——, "Dataset of synthetic data," 2019. [Online]. Available: https://drive.google.com/file/d/1WzhHA8G0pkiCI3TqiRuw6FC5MxWNzkVZ/view?usp=sharing

[18] ——, "Datasets of actions performed by each person in validation phase," 2019. [Online]. Available: https://drive.google.com/file/d/1VBzUSrx3Vzl_JTo5XK-Klb9HUHs_FXh7/view?usp=sharing

[19] ——, "Logs of results of predictions in validation phase," 2019. [Online]. Available: https://drive.google.com/file/d/1-WlX8l-H82QqrWlmoGurWXQ0qPLCXGDW/view?usp=sharing