A Framework for IoT Based Appliance Recognition in Smart Homes

PATRICIA FRANCO1, JOSÉ MANUEL MARTÍNEZ1, (Member, IEEE), YOUNG-CHON KIM2,3, AND MOHAMED A. AHMED1, (Member, IEEE)

1Department of Electronic Engineering, Universidad Técnica Federico Santa María, Valparaíso 2390123, Chile
2Department of Computer Engineering, Jeonbuk National University, Jeonju 54896, South Korea
3Graduated School of Integrated Energy-AI, Jeonbuk National University, Jeonju 54896, South Korea

Corresponding authors: Young-Chon Kim (yckim@jbnu.ac.kr) and Mohamed A. Ahmed (mohamed.abdelhamid@usm.cl)

This work was supported in part by the Agencia Nacional de Investigación y Desarrollo (ANID) through the Proyecto Fondecyt de Iniciación en Investigación 2020 under Project ID11200178, and in part by the National Research Foundation of Korea (NSF) Grant through the Korean Government under Grant 2021R1H1A305872911.

ABSTRACT Internet of Things (IoT) technologies will play an important role in enabling the smart grid achieving its goals in monitoring, protecting, and controlling by incorporating sensors, actuators, and metering devices while supporting various network functions and system automation. In this regard, home energy management systems (HEMS) enable customers efficiently use energy by managing their consumption, providing feedback information and improving control of major appliances. This work proposes a novel framework for IoT based appliance recognition in smart homes. It consists of two parts: training framework and inference framework. The proposed framework allows incorporating different loads in the monitoring system and enables selecting and testing specific parameters related to dataset configuration, feature extraction, and classifier model setting. The work contributes by developing an easy-to-use tool that allows customization of the training/prediction parameters according to the user criterion. Once the data and all its parameters are loaded, a novel feature extraction algorithm is used to obtain a total of ten statistical features. For the classification task, three machine learning models are included: a feed-forward neural network, a long short-term memory and a support vector machine. In addition, the user can apply a set of techniques to handle imbalanced classes, and also measure the influence of the selected features in the classifiers’ prediction by performing a feature importance analysis.

INDEX TERMS Appliance recognition, frameworks, intrusive load monitoring, internet of things, smart grids, smart homes.

I. INTRODUCTION Smart grid is the next generation electric power system, which supports bidirectional energy and information flow between service providers and consumers achieving higher stability, reliability, and efficiency. Internet of Things (IoT) technology is gaining increasing attention in the recent years. This technology can be used for enabling the smart grid to achieve its goals in monitoring, protecting, and controlling through the incorporation of different metering devices such as sensors and actuators, while supporting various network functions and system automation. IoT technology has been applied in smart buildings, healthcare systems, agriculture, smart cities, and smart homes, among others application domains [1]–[4]. In particular, the applications of smart home and home energy management systems (HEMS) are essential towards achieving energy efficiency. To build such management systems, it is necessary to identify and control appliances with higher electrical consumption, i.e., major appliances [5]. The home appliances are mostly used for routine household tasks, such as doing laundry, food preservation, or cooking. Among these common loads, there are the washing machine, the heating, ventilating and air conditioning (HVAC), the dishwasher, the freezer, and the electric vehicles (EVs). In case of electric vehicles, it is expected that EVs will be a key part of the future smart grid as they bring many environmental and economic benefits. However, the bidirectional operation flow of EVs carries complex problems into the distribution
power network, which challenge their integration. This bidirectional energy flow is between electric vehicles and the power grid [6].

By identifying appliance usage it will be possible to deploy multiple useful smart grid applications, such as load planning, demand response (DR) and the classification of activities of daily living (ADLs) [7]. The process which enables the recognition and monitoring the main loads in a household is commonly known as load monitoring [8]. There are two main categories for classifying the methods that manage such process, software based methods (non-intrusive load monitoring, NILM) and hardware based methods (intrusive load monitoring, ILM), as shown in Figure 1.

In Figure 1, from bottom to up, the task is to recognize individual appliance loads through different stages: Data acquisition, Feature extraction, and Classification. Stage one is the process in which data is gathered using a physical device. Then, a further processing (Feature extraction) is performed over the acquired samples to obtain a signature which represents the appliance electrical consumption. Finally, the resultant extraction is frequently classified through machine learning (ML) models. Both methods share some common characteristics. One of them is the use of ML techniques for predicting the behavior of appliances usage and translating raw data (e.g., current, voltage and power) into an easy and understandable form. These techniques also allow to make a deeper analysis of the electricity consumption so that it is possible to build a consumer profile, bringing privacy concerns into account [9].

In software-based methods, measurements come from one single point of sensing (usually the smart meter device). This simplicity for collecting data makes NILM an attractive solution specially for low-cost implementation scenarios. Even though these solutions have been the target of vast majority of research studies in load monitoring during the last decade, the solutions have shown less accuracy and higher difficulty for its deployment in practical scenarios compared to hardware-based methods. The NILM techniques are primarily based on event detection. They consist of sampling the aggregated signal obtained with smart meters, and then identifying particular signatures of electrical appliances. The aggregated signal is characterized for being noisy, which causes that only a few major appliances can be classified, depending on the sampling frequency. Even with the advancement in artificial intelligence (AI), the situation is the same, only some appliances, e.g., oven, air-conditioner, washing machine, and EV can be monitored [7], [10], [11]. Facing these types of scenarios regarding appliance usage, performance is not conclusive on various datasets [12].

On the other hand, hardware-based methods have the characteristic of collecting data from multiple points of sensing. These techniques include two subcategories. The first is known as distributed sensing, in which appliance profiles are obtained from device level using sub-measurement sensors installed next to appliances. The second sub-category is smart appliances (SA). These are devices with integrated and enhanced capabilities, not just to monitor their energy consumption, but to report it as well [10], [11]. The feature extraction consists of computing unique vectors using different procedures (e.g., sliding window) to be set as input of a ML-based classifier [13], [14]. While these solutions are considered relatively expensive, they provide greater reliability and efficiency compared to NILMs. Distributed sensing has great potential as it detects and controls the operation of different devices and appliances due to the fact that they can be collocated. For example, turning off a light when an occupant leaves a room. Another significant benefit is that these methods typically need less complex solution in terms of appliance recognition. An appliance recognition system consist of giving a label to the appliance connected to certain metering device. The label is basically related with to the appliance name.

Due to the elevated market prices and interoperability challenges of smart appliances, these devices are not widely in use, and distributed sensing (ILM) becomes an attractive solution [11]. One important fact to remark is that distributed sensing needs communication technologies to allow the exchange of information and control data among all electrical devices. The communication networks are responsible for carrying control data generated by the metering devices to home appliances, control commands from the home gateway to the appliances, and from the utility to the appliances registered in the home gateway [15]. Analyzing in depth what has been described above, it leads us to think of ILM solutions as an IoT platform used for load monitoring and its various applications.

The access to smart meter measurements is still limited, and the implementation is challenging in some countries due to implementation and regulatory issues. Additionally, data with higher resolution are difficult to obtain using most of commercial smart meters available today. Most smart meters challenge with complexity in setup, data storage, and cost. On the other hand, with the increasing popularity and enhancement of IoT and communication technologies, the ILM solution has become affordable when it comes to overcome the difficulty of NILM methods implementation. ILM promises to be the more effective for the future development of residential load monitoring for different applications such as load forecasting, home automation, energy feedback, demand response, and healthcare systems. In the meantime, different aspects concerning accuracy, data resolution, real-time, and the number of appliances to be covered should be considered [8], [16].

In [17], authors presented an appliance and activity recognition system which operates in the application layer of an IoT architecture. Although this system achieved a high accuracy in the classification process, the procedure in the feature extraction limits the system implementation in practical scenarios. To build a feasible system which can be implemented practically, new data and pre-processing techniques need to be incorporated.
FIGURE 1. Methods of load monitoring. On the left: Software-based methods (NILM). On the right: Hardware-based methods (ILM).
This work proposes an easy-to-use framework for appliance recognition based on distributed sensing techniques. The proposed appliance recognition system belongs to the application layer of the IoT architecture. The framework adds a graphical interface that significantly accelerates and facilitates its use. The user (the person using the tool) will have the possibility of deciding which pre-processing techniques apply to the data, including the features to extract, the classifier to use and its corresponding configuration parameters. Inside a sliding window, the selected features will be computed, and the resulting data frame will be used to train one of three ML classifiers: a feed-forward neural network (FFNN), a long short-term memory (LSTM) and a support vector machine (SVM). In this way, the same pre-processing can be applied in different datasets. To the best of our knowledge, no previous research work has presented a tool which allows to standardize the appliance recognition process, and, at the same time, to customize the pre-processing stage according to a user’s criterion. Therefore, the main contributions of this paper can be summarized in four main aspects:

- A novel ILM solution is developed and analyzed as part of an IoT architecture taking care of the requirements to be applied in a practical scenario.
- The proposed ILM solution consists of an appliance recognition framework with a graphical interface that enables a user to customize the training and prediction stages according to the requirements.
- The system can be trained with different datasets, therefore, standardization is considered.
- Depending on the selected parameters, the system can be adjusted to be working closer or not to real-time operation.

The paper is organized as follows: Section II gives a comprehensive review of the most relevant research work on intrusive load monitoring, giving an insight to the literature concerning appliance recognition and IoT based approaches for ILM. Section III gives an overview of the proposed appliance recognition frameworks. Next, Sections IV and V explain in detail the composition of all the processing available in both frameworks. Section VII provides the results and the performance evaluation, and in Section VIII, the results are discussed. To sum up, in Section IX, the conclusions of this work and the guidelines for future work are given.

II. RELATED WORK

A. ON INTRUSIVE LOAD MONITORING

The vast majority of research work done in load monitoring has been centered on NILM. However, some authors have established the basis of hardware-based methods over the last decades, getting to implement their work in practical scenarios. For example, in [14], authors presented a survey on intrusive load monitoring, in which explain its main characteristics, and implementation requirements. In addition, authors summarized the main NILM techniques given in the literature in a four-layered architecture (Sensors, Gateway, Server, and Vues), and defined the feature extraction process and ML models typically used for ILM applications. Based on the description given in this work, it is possible to envision the ILM systems as an IoT platform with more opportunities to improve various smart home applications.

In another survey presented in [8], the authors stated that with the evolution of power electronics, the accuracy and flexibility of power control is significantly improved, but this causes that the applicability of NILM methods is highly deteriorated. Power converters make possible the continuous adjustment of the device power and remove harmonics and compensate the reactive power. As a consequence, the features being extracted from appliances signatures will become indistinguishable. To conclude, the authors highlighted that future residential load monitoring is expected to adopt a hybrid form, combining NILM, individual appliance signatures collected through smart plugs (distributed sensing), smart appliances and HEMS.

In [9], the author presented a survey that lays the foundations for the development of important applications in the remote and automatic intervention of energy consumption inside homes and buildings. This research work provided a theoretical basis on load monitoring methodologies, concluding that it is feasible to have detailed monitoring and control of appliances using ILM in smart homes to provide energy efficiency, entertainment, healthcare, convenience, and security.

A survey presented in [18] pointed an IoT-based healthcare system. Authors realized the necessity of combining the different IoT services. The result of such applications are a significant amount of data that must be handled correctly during monitoring. In this regard, authors highlighted cloud computing as the hard-core technology, as it is a promising solution for an efficient and centralized management aiming to achieve knowledge processing in the healthcare sector.

A different approach, referenced in [19] gave a general description of sensor fusion techniques and probed the dependency between sensor fusion and dense sensor networks. Authors stated that the so-called multisensory approach can give competitive results considering the complete description of the activities of the sensors deployed in an indoor environment.

B. ON APPLIANCE RECOGNITION IN SMART HOMES

Over the last decade, many proposals came out detailing the foundations of appliance recognition in the context of smart homes [13], [14], [20]–[22]. In [13], the authors proposed to detect and identify household appliances being used by analyzing low-frequency monitoring data collected by meters (e.g., smart plugs) distributed throughout a smart home. For the classification stage, a supervised classification model based on artificial neural networks is used. The model was validated by using real-world power traces collected in home environments. Considering that the main goal was to recognize appliances, authors mainly worked on the application level in the experiments.
In [20], the authors proposed electrical appliance identification technique based on three features. The features are the energy consumption, time usage and location. The information embedded in such features was used to train six different machine learning classifier models. The models are Random Forest (RF), Decision Trees (DT), Bagging, LogitBoost, Naïve Bayes and SVM. The results of this work showed a high accuracy level, which translates into a good performance of the features used. As the main goal was to obtain a neutral assessment of the features, authors only analyzed standard techniques. Therefore, no other application, such as ADL identification, were considered. Although authors conceived their system as part of a smart grid environment, they only focused on the application level-related challenges, leaving any information about the infrastructure or the IoT-based architecture to support the system out of the scope of the research.

A supervised machine learning approach is given in [21] for appliance recognition which mostly bases on the device power signature. In addition to building a customized measurement device, authors aimed to develop what they described as “load library” of individual power signatures for training and recognition. The ML model selected in the classification stage was a K-Nearest Neighbors (KNN). Training results showed that data acquisition timing is crucial, thus it must be considered. After a set of test performed, results showed high accuracy, however these results were obtaining without comparing the KNN with a different ML model. Another recent approach, presented in [12], authors aimed to design and develop a real-time IoT end-to-end system to identify electrical appliances under the premise of keeping hardware costs low. Among the various ML modeled authors benchmarked are K-nearest neighbors (KNN), Decision Tree (DT), and Random Forest (RF). The task was to identify the consumption of a set of operating appliances. The authors refrained to detail the instant at which data acquisition should take place throughout the electrical appliances operating cycle, or the amount of data that should be collected before classification is performed. By using a high resolution current transformer (CT) sensor, authors ensured cost reduction while also delivering satisfactory results. The system implementation in a laboratory was described as a data acquisition system which further processed the data for the classification stage. Even though the system accomplished a high classification accuracy, around 95%, the research did not include details of any load monitoring application, such as ADL classification.

C. ON IoT BASED APPROACHES FOR APPLIANCE RECOGNITION

In [17], the authors presented an IoT based approach for load monitoring and activity recognition in the context of smart homes. The system bases on a proposed architecture of five layer. These are: the physical things layer, perception layer, communication network layer, middleware layer, and application layer, each of them having a specific function. Physical things and perception layers encompass appliances and metering devices, and both are involved in the data acquisition stage. Later, communication network and middleware layers enable the exchange of data between the devices (appliances and metering devices) and the home applications built for consumers. The resultant system has the capabilities of monitoring the major loads in a household, which also benefit the implementation of a series of meaningful applications such as the classification of activities of daily living. The overall architecture frame is illustrated in Figure 2. The application layer consists of an appliance recognition module which plays a key role in the implementation of the activities of daily living (ADL), classification models, and other ILM application. The main objective of the proposed appliance recognition module is to assign a label to the collected data and allow the further processing needed to classify ADLs. The selected ML classifiers were trained and tested using data from UK-DALE dataset. Three models were carried out: a feed-forward neural network (FFNN), a long short-term memory (LSTM), and support vector machine (SVM). The proposed ADL algorithm maps each ADL according appliance used, i.e., each appliance identified. The features extraction algorithm bases on the consumption in Watt-hours and the time instances where appliances were switched on. In case of the FFNN and the LSTM networks, the accuracy was above 0.9, while for the SVM network, this metric behaves around 0.8. The rest of experiments performed aimed to evaluate the classifiers using a new test set. In addition, authors conducted a sensitivity analysis was also to study the impact of one of the feature extraction parameters, the group size, on the classifier accuracy.

The general structure of an ILM system is shown in Figure 3. There are a series of devices that can be used to collect data: either smart appliances or a sensor/actuator attached to target appliances. Although smart appliances have much better performance in terms of applicability, generalizability and accuracy, their elevated commercial prices restrict their massive deployment. Hence, smart appliances remain out of the scope of this research. With the use of sensors and actuators, the main idea is to detect the appliance’s electrical consumption based on their activations. Therefore, some computational procedures are necessary. First, features are extracted, and they can be used to show certain statistics, such as the maximum, minimum and mean value of an electrical measurement (e.g., power, voltage, current). Then, the statistical features are classified by using ML learning techniques. The latter process varies from supervised to unsupervised learning techniques (e.g., Support Vector Machines and K-means, respectively) [13], [14]. As the first stages of ILM involve hardware devices (sensors and actuators) and further processing is software-based, performed in a central data server (middleware), a communication network is needed to allow data exchange between sensors and the server. Middleware solutions are used to integrate and coordinate the nodes, thus achieving a real-time status and management of the household [9]. For appliance recognition, standardization is an important issue. If the appliance
recognition is not standardized, several adjustments have to be made (if different data structures are applied), especially in the feature extraction part. In the UK-DALE dataset, records are structured in two columns of float values. One corresponding to the sample timestamp, and the other one belonging to the active power. For feature extraction, the data can be divided into groups and then used in the same proportion of feature vectors for each training class. This balance prevent the need to work with imbalanced classes, which would impose a great challenge in the performance of the machine learning classifier. As a result, the training can be performed using as many feature vectors per class as the number of vectors in the minority class, leaving information behind. Since only the appliances activation are considered for extracting the features, in a practical scenario it might conduct to counterproductive results. In a practical use-case, it is likely that the system has to deal with a different number of samples for each class, thus most proposals did not handle the imbalance problem. In addition, for a practical use-case in which the system must be retrained in front of new data, the fact of training with the whole appliance profile dividing it into fixed groups may impose a restriction since it makes the system less capable of working in real-time. Hence, there is a need to make some adjustments, forcing the system to be...
more resistant to real-time processing. Another big constraint imposed by the pre-processing stage is the extracted features. The majority of the features depend on the scale of the data, i.e., how many samples above 30? This implies that the data need to be delivered in a specific format, which is not possible with some sensing devices.

On the other hand, the smart grid is now supporting many new applications in the distribution power system. Among these applications, electric vehicles are expected to play an important role. The electric vehicles operation modes adopt a bidirectional energy flow between EVs and the power grid, therefore their integration into the power grid is considered a highly complex task. Many research studies have been conducted to investigate the influence of electric vehicle charging on the distribution system from different power perspectives [10], [11], [23]. The impact of electric vehicle charging is expected to be significant in view of power losses, power quality, voltage deviations, harmonics and frequency shift. To overcome the new peak demand with the integration of electric vehicles, solutions include increasing generation capacity, upgrading the existing distribution system infrastructure or considering demand response techniques [24]. Demand response (DR) strategies will play an important role in load shaping in order to prevent the distribution transformer overloading. Furthermore, load monitoring techniques are vital to develop efficient DR applications. In [10], [11], [23], the authors proposed event-detection algorithms for NILM systems based on low complexity statistical features. Although they achieved successful simulation results, the efficiency in practical scenarios has not been proved yet. As it was previously discussed, NILM techniques are rather unreliable in these situations. However, EVs are considered major loads in the context of smart homes and their monitoring has to be highly prioritized.

All in all, to make the system capable of working in nearly real-time and to include the monitoring of major loads, a novel appliance recognition module is proposed. This proposed work emerges as a framework which facilitates not only the pre-processing and training, but the inference too, becoming a flexible system which allows to apply the same principle using data from different sources.

D. SUMMARY OF RELATED WORK
Table 1 shows the summary of the main aspects for relevant research proposals described in this section. First column reveals the reference. The second and third, contain the type of publication (survey/application) and the load monitoring method, respectively. In the case of references which have specified the method as IoT-based system, this means that the authors do not explicitly describe their system as an ILM solution but as an IoT-based approach instead. As it was described in Section I, ILM solutions can be considered analogous to IoT applications for load monitoring and remote healthcare. The fourth column indicates the dataset considered in the studies. If the dataset is not described or mentioned, the space is left in blank. In cases where the dataset was detailed as collected data, it means that authors used their own collected data to train their classifier models. Finally, the last column highlights opportunities which benefit the proposal of this paper, since these are remarkable facts (features extracted, datasets, architecture).

III. FRAMEWORKS FOR IoT BASED APPLIANCE RECOGNITION
In this work, two frameworks are developed: a training framework and an inferences framework, to work in the application layer of the IoT architecture. The general description of both frameworks is given in the next subsections. Both frameworks include a graphical interface which easily allows the setup of the system, giving the possibility for the user of configuring it by applying different pre-processing and features extraction techniques, as well as choosing among the classifier models.

The result is a flexible and complete system that could be scaled in the future to include more data and as many classifier models and pre-processing techniques as desired.

A. TRAINING FRAMEWORK
Figure 4 shows the general composition of the training framework which has been implemented in Google Colaboratory (Colab). The upper part of Figure 4 shows the dataset configuration. First, it is necessary to input the number of classes, the location of the dataset files, the selection of the target appliances and to choose how to input the activation threshold. If the thresholds are known, then the user (the person training the system) can manually introduce their values. On the contrary, if the manual setup is hard to obtain, thresholds will be computed automatically. In this case, the only requirement is to introduce a value that represents the limit of the neighborhood of the minimum power measurement registered by the appliances. Then, the thresholds values will be between the global minimum and the limit imposed by the user. The reason to have an activation threshold is to compute the stand-by value of every appliance. Therefore, only the activations or the values of the active power when an appliance is ON will be considered for feature extraction. To compute the activation threshold, first, the minimum is obtained; later, the algorithm looks for all power measurements between the minimum and the minimum plus the limit inputted by the user. After that, the maximum of all the filtered values is chosen as the activation threshold. Once the initial setting is completed, the user has to choose the name of the dataset to load and optionally to check missing samples in the chosen dataset. The method to fill missing values can also be selected by the user. The possible datasets to work with will be described in the following subsections. At the end of this part, the target appliances signatures are plotted.

The middle part of Figure 4 represents the pre-processing and feature extraction. Once the selected dataset is uploaded and activation thresholds are computed, the user can select...
In the next step, a set of ten possible features can be selected to extract. These features represent statistical computations that describe the appliances’ profiles. One important aspect to remark is that the user can train to extract the whole set...
of features or either choose a subset of them. Samples will be processed inside a sliding window. This sliding window operates similarly to a 1D-convolutional layer in a convolutional neural network but without the convolution operation. Therefore, the window will have a size from which statistical features will be calculated. The size value is one of the parameters that is required to be provided by the user at this stage, along with the sliding window stride and mode. The stride parameter reveals how far the window should move at each step, and the mode describes what to do when the size of the window is larger than the number of remaining samples. For the latter operation, the user can choose among padding, no_padding and dynamic. Then, the true labels that correspond with the appliance name, must be provided for the selected appliances.

After feature extraction, data have to be prepared to enter the classifier model. In this case, the complete set of vectors is divided into three subsets: training, validation and testing. The user can set the proportion of the dataset to be included in the three subsets, for example, 80% of the feature vectors in the training set and 10% in the validation and test sets, respectively.

The bottom of Figure 4 shows the training configuration. The user can choose between standardization and normalization as the feature scaling method to apply. Feature scaling is the process of converting all the features into a given range. Depending on the operation selected, the limits of this range will be established [25]. To complete the pre-processing stage, the true labels are converted into a numerical value. Then, the user must decide which classifier model to train, for example, the feed-forward neural (FFNN) network, the long short-term memory (LSTM) or the support vector machine (SVM) classifier. In this work, we apply a penalization or so-called kernel regularizer to the model. In this regard, three options are available: L1 norm, L2 norm or a combination of both (L1_L2). A penalization can be helpful in the presence of imbalanced data. Regularizers allow to apply penalties on layer parameters during optimization. These penalties are summed into the loss function that the network optimizes [26].

Once the selected classifier is trained, the model is evaluated using the validation data. In addition, a new tool was deployed to assess the performance of the model given the chosen features. It is a process called feature importance and it allows us to understand how the features in our model contribute to prediction. Now, it is possible to know if a given feature has more or less relevance to the system behavior, and in the negative case to counteract it. The best model configuration can be saved to make future inferences. The last part of training is configured to predict with the test set. To assess the system’s generalization, a set of metrics are used. These are the precision, recall, F1-score, cohen’s kappa coefficient and confusion matrices. The cohen’s kappa coefficient is the classification accuracy normalized by the imbalance of classes in data.

B. INFERENCE FRAMEWORK

The inference framework is used to test the system generalization which predicts with unseen data, similar to what is expected in a practical use-case. The general structure of the framework is shown in Figure 5. In the same way, as the training framework, first, it is necessary to configure the dataset to use, entering the number of classes (equivalent to the number of appliances), the appliances to use, the location of their profiles and their activation thresholds. In this case, as the same appliances are used, and their activation thresholds are computed during training, the user (i.e., the person making the predictions) only needs to input the obtained value to make inferences. Then, the user must select the dataset to analyze and optionally check missing values.

Similar to the training framework, the next step is the preprocessing and feature extraction. The user needs to provide a time range to infer, it can either select the same or a different time range from training, although predicting with the same data used in training has no meaning when assessing the system generalization. Next, there are the feature selection and the parameter setting for the extraction. Obviously, this is done identically to the training. Inside a sliding window, configured with the size, stride and mode determined by the user, the set of selected features (among a group of ten) are computed. After finishing the feature extraction, the true labels need to be provided to the feature vectors.

Since there is no need to perform data splitting for inferences, the next step is to concatenate the resultant vector in a unique dataframe. Once completed, it is only required to load the model and prepare the data to input the trained classifier model (pre-processing). In that sense, the user has to provide the model’s location and to select the feature scaling function to apply and as well as the model type (FFNN, LSTM and SVM).

The last step, shown at the bottom of Figure 5, is to perform predictions and to obtain the evaluation metrics. These are also the precision, recall, F1-score, cohen’s kappa coefficient and confusion matrices.

IV. DATASETS

Two datasets are considered in the frameworks: the UK-DALE dataset [27] and the Pecan Street Dataport [28]. Both datasets are similar as both provide individual power consumption and aggregated signal for a set of houses in a certain period of time. However, they differ in structure, appliances included, the scale of the measurements, features (active power, reactive power, etc.) and sampling frequency.

The UK-DALE dataset involves the consumption profile of five houses in the United Kingdom (UK). It is organized in a hierarchy of file folders in which each house has its own folder, and inside, there are separated files for each appliance in the house and their aggregated power consumption. Each file is structured in two columns: one for samples’ timestamps and the other one for active power measurements.
The data used for training were collected at a 6 s sampling frequency. The house (1) appliances are washing machine, iron, oven, hair dryer and dishwasher. The general structure of the UK-DALE dataset is represented in Figure 6.

In contrast, the Dataport dataset is the world’s largest residential energy and water research database which has been used by a large number of researchers. Dataport contains unique, high-resolution data collected from instrumentation that Pecan Street installed in approximately 1000 homes, from which about 50 are located in Colorado, 100 in New York, 750 in Texas, 50 in California, and 50 in other states. Over the course of 2020, an additional number of 100 homes in the New York State and 100 homes in California have been added to Dataport [28]. Pecan Street provides access to static time-series datasets (1-second energy, 1-minute energy and 15-minute energy) for 25 homes in three regions (Austin, California, New York). The provided data on the New York region contains 6 months of data with 100% completeness across all intervals for individual homes, and 99% for California and Austin.

The objective pursued by including these data in the appliance recognition framework is to include the electric vehicle...
among the target appliances, as it is considered a major load in a smart home. Although plug-in vehicles have not been adopted at a scale in some countries, governments, utilities and automobile companies, like Tesla [29], are corroborating the opportunities that arise from reduced emissions and gasoline consumption. Apart from its many benefits, the inclusion of electric vehicles into the power grid creates along serious challenges to utilities, as this load adds stress on the power grid, which might cause voltage instabilities and blackouts [15]. The vehicles’ charging consumption is assumed as the analog load introduced by connecting another house into the power grid. Therefore, load monitoring can massively contribute to avoid overload in the grid, overcoming the aforementioned challenges.

The free data files of Dataport can be heavy to process. Thus, to efficiently use the available computation resources, a subset of the dataset was used for training the proposed frameworks. The data of three appliances (electric vehicle, oven and microwave) from House 2335 of Austin’s residential data was filtered in a separate csv file. The resultant file contains a column for the time and date of the collected samples, and another column for each appliance profile. In this case, the sampling frequency of data is 1 minute.

V. PRE-PROCESSING

The pre-processing stage represents all operations conceived to prepare the data to input the classifier model. Once the appliances’ data are read, the feature extraction, some techniques to handle the imbalance in the data, dataset splitting and feature scaling take place. All these operations will be described below.

Optionally, when the data are loaded, the user can check the existence of missing values in the profiles. In this case, the available methods to fill the gaps are:

- **pad**: propagates last valid observation forward to next valid backfill.
- **bfill**: uses the next valid observation to fill the gap.

The complete procedure in the training of pre-processing stage is represented in Figure 7. Inside a sliding window, with size, stride and mode set by the user, a number of statistical features (10 or less) are extracted. Once the feature vectors for every target appliance profile are obtained, the user has to provide a label for each of them that corresponds with the appliance name. Then, to have a better understanding of the classes’ distribution, the number of vectors in each class is plotted. If the number of vectors in every class is not the same, optionally, the user can apply a set of techniques to handle the dataset imbalance. Applied these techniques or not, the next step is to split the dataset into three subsets that will be used for training: the training, validation and test subsets. Next, an operation to take the values to the same scale is applied to the three subsets previous to input in the classifier model. When the training is completed, the user can evaluate the impact of the selected features in the system predictions by performing a process called feature importance. All these steps will be described in the subsections below.

A. FEATURE EXTRACTION

In this work, a new dataset, in addition to the UK-DALE, is considered in the appliance recognition frameworks. Both datasets are different in structure and format. Based on the analysis performed in [17], a 10-feature vector showed to be more than enough to capture the appliance behavior and to make the classifier model distinguish one from the other. Therefore, a set of ten new features was formed. It reflects a series of statistics behind the appliance power measurements. These features are:

1) Minimum power value
2) Maximum power value
3) Mean power value
4) Standard deviation
5) Skewness
6) Kurtosis
7) Coefficient of variation (or relative standard deviation)
8) Median absolute deviation (MAD)
9) Number of values above the mean
10) Number of stand-by values

To understand the formulation of each possible feature, see Ref. [30]. The flowchart of the feature extraction algorithm is given in Figure 8. It needs to be provided with four parameters in addition to the appliance profile: **stride**, **window size**, **mode** and **activation threshold** of each appliance. The first two (stride and window size) can take integer values ranging from 1 to 50. The mode reflects what to do in the case when the window size (w) is greater than the number of remaining samples (n) in the profile. Mathematically, we know that \( w > n \) by:

\[
n = \text{length(profile)} - i
\]

where \( \text{length(profile)} \) is the size of the complete appliance profile and \( i \) is the position of sample being analyzed in the appliance profile. Then, three mode options are available when \( w \geq n \):

- **padding**: it completes the missing values in the window by simply concatenating zeros at the end of the profiles. This option is set by default.
no_padding: the algorithm discards the samples from which \( w > n \). It just analyzes the value where the window fits.

dynamic: it adapts the size of the window to be equal to the number of remaining samples (\( n \)). In other words, it makes \( w = n \).

On the other hand, the activation threshold, as it was described before, can be manually configured or either computed looking for the maximum value in a range given by the minimum power measurement in the appliance profile, and a limit which depends on the value inputted by the user. Then, the selected features are computed to finally return a dataframe of feature vectors.

**B. DATASET SPLITTING**

Separating data into different subsets is an important part of evaluating data in machine learning models. Typically, the data are split into a training set and testing set where most data are used for training, and a smaller portion of data is used for testing [26]. It is often used an additional set called validation set to give an estimate of model skill while tuning model’s hyperparameters. The validation dataset is different from the test dataset that is also held back from the training of the model [26].

For training the proposed frameworks, the dataset is split into:

- a **training** set: a set of data used for learning, that is to fit the parameters of the classifier.
- a **validation** set: a set of data used to tune the parameters of a classifier, for example, to choose the number of hidden units in a neural network.
- a **test** set: a set of data used only to assess the performance of a fully-specified classifier.

The proportion of data for each subset is determined by the user. In addition, the validation set is useful in the feature selection. This impact on the selected features will be further discussed in the next subsections.

**C. TECHNIQUES TO HANDLE THE CLASS IMBALANCE**

Since the proposed appliance recognition frameworks only consider the activations (i.e., power measurements when the appliances are on), and the activation pulses can have different time duration. Then, the resultant number of feature vector in each class is different. Hence, we are in the presence of imbalanced data to train.

Data imbalance is a challenging problem which has not got clear solution yet. However, some techniques can help to reduce the impact of this disproportion in the dataset.
FIGURE 8. Flowchart of the feature extractor algorithm.
It is important to highlight that all these techniques are only applied to the training set. The techniques implemented in the proposed training framework are:

- **Drop duplicates**: it executes a method that removes all the duplicated feature vectors in the training set. It can serve as an undersampling procedure to reduce the number of elements in the majority classes (i.e., the classes with a higher number of vectors).

- **Oversampling**: these are synthetically generated samples making use of the Synthetic Minority Over-sampling Technique (SMOTE) algorithm [31]. This algorithm, as its name indicates, generates synthetic samples for the minority classes. It focuses on the feature space to generate new instances with the help of interpolation between the positive instances that lie together.

- **Undersampling**: it reduces the number of samples in the minority class following a certain strategy (i.e., the proportion of samples in each class). In the case of the proposed framework, the strategy has to be included in system as a dictionary. The implemented algorithm takes advantage of the RandomUnderSampler class available in scikit-learn [32].

- **Penalized models**: penalized classification causes an additional cost of the model for making classification mistakes on the minority class during training. These penalties can bias the model to pay more attention to the minority class [26]. In the proposed training framework, penalization is performed by setting regularizers to the classifier models. The penalties are added to the loss function that the network optimizes. Three different regularizers are available. The three regularizers may be transferred to a layer as a string identifier:

  1) **L1 norm**: a regularizer that applies a L1 regularization penalty. The user needs to provide the coefficient to use.

  2) **L2 norm**: a regularizer that applies a L2 regularization penalty. The user needs to provide the coefficient to use.

  3) **L1_L2 norm**: a regularizer that applies a combination of both L1 and L2 regularization penalties. The user needs to provide the coefficient to use.

- **Cohen’s Kappa coefficient**: Accuracy can be misleading when working with an imbalanced dataset. It is the case in which accuracy is high (such as 90%), but the accuracy is only reflecting the underlying class distribution. Including a new metric, such as cohen’s kappa coefficient can help to describe better the behavior of the system. Classification accuracy is normalized by the imbalance of classes in data. Cohen’s kappa coefficient represents the classification accuracy normalized by the imbalance of the classes in the data. It is based on comparing the concordance observed in a set of data, with respect to what could occur by mere chance [33].

**D. FEATURE SCALING**

Feature scaling is a technique to standardize the independent features present in data in a fixed range. Feature scaling is performed during the data pre-processing in order to handle highly varying magnitudes or values or units. If feature scaling is not done, then the machine learning model tends to weigh greater values, and to consider smaller values as the lower values, regardless of the unit of the values [26].

The two available techniques in the proposed frameworks are: **standardization** and **normalization**. Standardization takes the values to be centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation. It is computed using the Z-score:

\[
Z = \frac{x - \mu}{\sigma}
\]

(2)

where \(x\) is the raw score, \(\mu\) is the mean, and \(\sigma\) is the standard deviation. Normalization is a scaling technique in which values are shifted and rescaled so that they range between 0 and 1 [26]. It is implemented by applying the Min-Max scaler formula:

\[
X' = \frac{X - \text{min}(X)}{\text{max}(X) - \text{min}(x)}
\]

(3)

Here, \(\text{max}(X)\) and \(\text{min}(X)\) are the maximum and the minimum values of the feature, respectively.

**E. FEATURE IMPORTANCE**

Feature importance reflects the relative contributions of features to predictions made by a model. They are a set of techniques that assign a score to input features based on how useful they are at predicting a target variable. The feature importance scores play an important role in a predictive modeling project, providing an insight into the data, an insight into the model, and the basis for dimensionality reduction and feature selection. This can improve the effectiveness and efficiency of a predictive model on the problem [34].

In the proposed training framework, a feature’s importance is calculated as the difference between the baseline score and the average score obtained by permuting the corresponding column of the test set. If the difference is small, then the model is insensitive to permutations of the feature, thus its importance is low. Conversely, if the difference is large, then the feature’s importance is high. The parameter permutations, set by the user, controls the number of permutations per feature. More permutations imply better estimates, at the cost of computation time. The user can decide which score metric to use among a set of four possibilities:

- **cohens_kappa_score**: computes the cohen’s kappa coefficient.

- **accuracy_score**: uses the classification accuracy.

- **balanced_accuracy_score**: defined as the average of recall obtained on each class.

- **f1_score**: computes the classification F1-score.
VI. CLASSIFIER MODELS
In the case of the classifiers, this work considers three models:

- **Support vector machines (SVM):** this model is based on developing a hyperplane or set of them in a high or infinite dimensional space. A good distance between the hyperplanes implies a greater distance to the closest training data points of any class, which is often called the functional margin. If the margin is large, generalization error of the classifier will be smaller [35], [36]. SVM was used as a classifier model in [36].

- **Feed forward neural networks:** or multilayer perceptron (MLP) is a ML model in which information goes from the input through intermediate calculations to the output, without any feedback connection. This means that none of the output layers is fed back on itself. At the time of deciding the configuration of an FFNN model, no specific procedures are mandatory for selecting the number of hidden layers and neuron units. Too many parameters will lead to overfitting, which affects the generalizability and performance of the model. On the contrary, a simple model tends to underfitting, and therefore, it is necessary to extract more characteristics from the data [26]. This model was implemented in [13] to classify a vector of ten features.

- **Long short-term memory (LSTM):** this is a type of recurrent neural network (RNN) model that includes a memory cell with gated inputs, outputs, and feedback loops. The main benefit of this model is that it overcomes the vanishing gradient problem, that commonly affects RNNs, and in which gradient information disappears or explodes, as it is propagated back through time [26]. This is one of the main reasons why LSTM are known to be better suited for time series data [37] analytics. Particularly in classification problems, LSTM networks has reported to be capable of successfully achieving competitive results. LSTM was another model tested in [12] for the classification stage.

The main differences among the three models lie in the inputs and outputs of the system, in addition to the regularization criteria that the user might choose to apply. The input size is determined by the number of features selected by the user. The outputs, on the other hand, correspond to the number of classes being classified. If the regularization checkbox is selected, then a penalization will be applied to the chosen classifier. The penalization methods to apply were described in subsection V-C. On the contrary, if regularization is unchecked, then kernel_regularizer=None will be passed on to the model. This means that no penalization will be applied.

VII. EVALUATION AND RESULTS
Several tests were performed in order to prove the reliability of the proposed frameworks using two datasets (UK-DALE and Dataport). The first step was to set the training parameters. A group of them, e.g., the activation threshold, the test set size and the classifier’s hyperparameters (learning rate, batch size and epochs) remained unvaried among the experiments. Others, like the stride value and window size, were gradually incremented to study their influence on the system performance.

A. PARAMETERS SETTING
In Figure 3, from top to bottom, it is first needed to determine the dataset-related parameters. Appliances activation thresholds were manually configured; thus, the remaining parameter to set in this part is the number of classes. Appliance activation thresholds were established after analyzing the data of all the appliances in each dataset and observing their standby pulses. The resultant values are shown in Table 2. The number of classes corresponds to the name of appliances selected in both datasets, i.e., 5 for UK-DALE and 3 for Dataport datasets.

When checking for missing values, the method for filling the gaps was fill. On the other hand, the time range to extract the features was determined by analyzing the appliances signatures and searching for an interval in which all selected appliances had activations. This interval needed to be as short as possible, since in a possible practical use-case, this selection implies the minimum amount of time required for collecting data, but at the same time, the classifiers need enough data for training. The selected ranges are also shown in Table 2. A total of two weeks data for UK-DALE and one week data for Dataport were used for training.

Regarding the feature extractor parameters, the window size and stride were gradually increased to assess the system behavior through several metrics. For this analysis, all the features were selected in padding mode. However, some experiments used a different value of the last two, to study their influence on the model’s predictions.

The rest of the pre-processing and training parameters: test size, feature scaling method, coefficients of regularizers and number of permutations in the feature importance remained unvaried over the experiments, and the values are summarized in Table 3. The classifiers hyperparameters and initial settings are shown in Table 4.

The target appliances in both datasets are listed as follows:

*From UK-DALE dataset:*
- Washing Machine
- Dishwasher
- Oven
- Iron
- Hair dryer

*From Dataport dataset:*
- Electric vehicle
- Microwave
- Oven

Therefore, there is a total of five classes in UK-DALE, and 3 in the case of Dataport’s. Selecting a class with a common name (label) in two different datasets, ensures that despite being the same type of appliance, their behavior can be totally
different, depending on the vendor or even on the usage mode of the householders.

B. ANALYSIS OF THE INFLUENCE OF THE FEATURE EXTRACTOR PARAMETERS

Figures 9 and 10 show the results obtained with the two datasets after varying the feature extractor parameters, stride and window size, to analyze their influence in the FFNN model. The metrics to evaluate the model’s performance are the accuracy and the cohen’s kappa coefficient. Since we are facing imbalance data, both metrics need special attention.

Accuracy and cohen’s kappa are obtained from three different values of window size: 5, 10 and 15, by gradually increasing the stride. With the UK-DALE, represented in Figure 9, for a low value of window size, such as 5, the accuracy since to have a random behavior ranging from 0.86 with a stride of 2 to 0.93 with a stride of 4. For high values of window size, the accuracy is higher for low values of stride. The best performance was obtained for a value of 10 in window size and stride. In the Dataport dataset, the accuracy reaches 0.99, showing a very competitive and solid performance. In this case, the best result in this case is obtained from a window size of 15 and a stride of 13.

In terms of cohen’s kappa, the values obtained from the UK-DALE dataset are, in most cases, around 0.8. The highest values and the best performance for this dataset is obtained from a window size of 10 and a stride of 1. For the Dataport dataset, the kappa coefficient is, in most cases, around 0.98, achieving a maximum of 0.993148.

The difference in performance in both datasets could rely on the number of classes (5 in UK-DALE and only 3 for Dataport) and the number of features extracted from each class. In different configurations of the parameters, there is a lower imbalance in the data of the Dataport dataset. All the above experiments were ran using all possible features and the model which gave the best results in the previous version: the FFNN. The mode used in all cases was the default padding and the selected regularizer was l1_l2.

The confusion matrices obtained from the best parameter configuration in both datasets are shown in Figure 11.
In the Dataport, all the outliers are related to the microwave, 71 features vector were incorrectly classified as an oven. Both appliances have a pretty similar behavior in terms of consumption. One of the most probable causes of these misclassifications may be related to the features selected (all of them in this case).

With the UK-DALE dataset, the most visible problems are related to the dishwasher. The vast majority of misclassifications in this class are labeled as either an oven or a washing machine. More attention should be paid to the minority class, the iron, in which 100% of their feature vectors were incorrectly classified. This could be highly related to the features selected. Therefore, an analysis of the importance of the selected features may be helpful to overcome this issue.

C. FEATURE IMPORTANCE ANALYSIS
This section shows the results of analyzing the feature importance of those cases in which the best result was achieved in terms of accuracy and Cohen’s kappa coefficient. Figure 12 shows the importance obtained when the complete set of features were selected for training using both datasets.

Retraining with only relevant features in both cases, results show an improvement compared with Figure 11.a and Figure 11.b for the UK-DALE and Dataport, respectively.

In the case of UK-DALE, more samples in the minority class are correctly classified when training with the most relevant features. For the Dataport dataset, improvements are evidence in the majority classes. Figure 13 shows the confusion matrices obtained after retraining with only relevant features. Figure 13.a shows the one obtained from the UK-DALE data, and Figure 13.b from the Dataport data. This analysis reveals that depending on the data, the selected features will give better results or not, i.e., the performance of the features varies depending on the data. The selection must be specific for the different datasets.

D. PREDICTING WITH THE THREE CLASSIFIER MODELS
After the analysis performed on the feature extractor parameters, the combination which gave the best results was used to predict the rest of the classifier models. This is in view of testing the performance of the classifiers in the implementation frameworks for appliance recognition. Table 5 summarizes
TABLE 5. Evaluation metrics for FFNN, LSTM and SVM, computed in both datasets and using the best configuration parameters: Stride(1) Widow size(10) for UK-DALE and Stride(13) and window size(15).

| DATASETS | CLASSES             | FFNN | LSTM | SVM |
|----------|---------------------|------|------|-----|
|          | PRECISION           | RECALL | F1-SCORE | PRECISION | RECALL | F1-SCORE | PRECISION | RECALL | F1-SCORE |
|          |                     |       |       |      |       |       |       |       |       |       |
| UK-Dale  | washing machine     | 0.98577 | 0.96377 | 0.97611 | 0.97555 | 0.96728 | 0.97189 | 0.98382 | 0.95231 | 0.96998 |
|          | oven                | 0.82091 | 0.82335 | 0.89364 | 0.81929 | 0.98183 | 0.89333 | 0.81427 | 0.98133 | 0.89011 |
|          | hair dryer          | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
|          | dishwasher          | 0.99582 | 0.83233 | 0.90156 | 0.99532 | 0.83281 | 0.86710 | 0.93414 | 0.84812 | 0.88450 |
|          |                       | 0.99737 |       |       | 0.99737 |       |       | 0.99737 |       |       |
|          | COHEN’S KAPPA       | 0.90156 | 0.90156 |       | 0.89185 | 0.89185 |       | 0.88807 |       |       |
| Dataport | electric vehicle    | 0.99952 | 1.00000 | 0.99960 | 0.99900 | 1.00000 | 0.99960 | 0.91322 | 1.00000 | 0.67744 |
|          | oven                | 0.99582 | 0.99849 | 0.99735 | 0.99148 | 0.99999 | 0.99932 | 0.99937 | 0.99924 | 0.99660 |
|          | microwave           | 0.94895 | 0.85833 | 0.89256 | 0.94453 | 0.90000 | 0.80659 | 1.00000 | 0.933739 | 0.52653 |
|          |                       | 0.99599 |       |       | 0.99599 |       |       | 0.73865 |       |       |
|          | COHEN’S KAPPA       | 0.902145 |       |       | 0.902145 |       |       | 0.648253 |       |       |

(a) Feature importance selecting all features for the best configuration parameters in UK-DALE.

(b) Feature importance selecting all features for the best configuration parameters in Dataport.

FIGURE 12. Feature importance analysis in UK-DALE and Dataport.

the results obtained from this experiment using both datasets. It is evident in Table 5 that in both cases (UK-DALE and Dataport), the best performing model involving Stride=1 and Window size=10 for UK-DALE, and Stride=13 and Window size=15 for Dataport represents the feed-forward neural network. However, repeating the analysis shown in Figure 9 for the LSTM model, the best parameter configuration turned out to be: Stride=8 and Window size=10, giving an accuracy of 0.94602 and a cohen’s kappa coefficient of 0.906754.

FIGURE 13. Results after applying feature importance.

Both metrics are pretty closed to the values obtained from the best parameter configuration for the FFNN; therefore, it is possible to state that the best performance of the machine learning classifiers directly depends on the feature extractor parameter configuration.
Special attention should be paid to the minority class in UK-DALE (an iron) as none of the 14 samples in the test set was correctly classified. This is most likely due to the problems that entail the fact of working with imbalanced data. Support vector machine is very sensitive to class imbalance [38]. Hence, a similar analysis as the one performed in section VII-C is required to improve the FFNN model’s behavior.

With the Dataport dataset, the support vector machine gives significantly poor results in comparison with the rest of the models. Thus, it is not recommended to choose the SVM as the preferred classifier model in this occasion.

E. PREDICTING WITH NEW DATA
To test the behavior of the model in front of new data, the Inference framework was configured to predict using data from a new house. In this case, the experiment was carried out in House 5 of UK-DALE. The parameters set were the same as in training. The window size and the stride are 10 and 1, respectively. The selected time range to analyze was also determined by looking for a period of time with activations of every target appliance. The resultant range is from September 1st, 2014 to September 8th, 2014. The confusion matrix obtained from this prediction is shown in Figure 14. As it can be seen, the model behaves poorly in front of unseen data having a very low accuracy of 0.46241. Therefore, in order to apply this system in a new house, first, the classifier models have to be retrained.

A second prediction was made, but this time using new data from House 1 of UK-DALE dataset and having the same configuration of feature extraction parameters (Stride=1 and Window size=10). In this case, the system achieved an accuracy of 0.93385, which is not far from the 0.9426 obtaining from training. The confusion matrix for this experiment is shown in Figure 15.

VIII. DISCUSSION
In this work, two frameworks were developed to present a tool that facilitates appliance recognition. With the new considerations, especially in the feature extraction process, the feature extraction relies on a sliding window. Depending on the size and stride programmed by a user, the system will be closer or not to real-time operation. For example, if the sampling frequency is 6 seconds (as in UK-DALE), and the window size is 10, this means that it is necessary to wait a minute to gather all the samples in the window and extract the features. If the sampling frequency is 1 minute (case of Dataport) and the window size is 15, this means that 15 minutes are necessary to collect the numbers of samples considered in the window, thus, it will be further from real-time. Therefore, although 10 and 15 are the sizes which gave the best results in UK-DALE and Dataport, respectively, the analysis for choosing the values of these parameters for a practical operation has to consider their relation with real-time necessarily. The sensitivity analysis on both, the stride and window size, leads to determining these parameters efficiently.

On the other hand, new data were introduced, including new appliances as an electric vehicle or a microwave. Although the oven is a target appliance in both datasets, they behave differently. Consequently, this ratifies the fact that even if two appliances are of the same type, consumption may differ significantly. The microwave and the oven exhibit pretty similar signatures; however, the classifier is able to distinguish between them accurately.

With regard to predictions using unseen data, to apply the system in a different house, the classifier models still need to be retrained. This is due to low accuracy and poor generalization shown in the experiments performed in House 5 of UK-DALE. When predicting new data from the same house but in a different time range, the accuracy does not vary much from training. Although there are some challenges that need to be overcome, the proposed frameworks facilitates the application of the same principle (pre-processing and classification) to data organized in different structures, introducing certain standardization in this process.

IX. CONCLUSION AND FUTURE WORK
In this work, two novel frameworks were developed for appliance recognition: a Training framework and an
Inference framework. Both frameworks operate in the application layer of the IoT architecture. In addition to bringing an easy-to-use tool to the user for training or predicting through a graphical interface, it allowed to incorporate some major loads in the monitoring system such as an electric vehicle and a microwave. The available data with these two loads, the Dataport dataset showed a notable difference in structure with regard to the UK-DALE dataset. Therefore, standard statistical features were proposed in order to apply the same pre-processing principle with both datasets.

The proposed frameworks allowed the user to select and to test specific parameters related with dataset configuration, feature extraction and classifier model setting. Feature extraction relies on a sliding window which is similar to a 1D convolutional layer, but without the convolution operation. Depending on the size and stride established by a user, this system will be able to operate in real-time. A sensitivity analysis on stride and window size was performed aiming to find the values that gave higher accuracy. This metric was about 0.99 and 0.94 for the best configuration parameters when evaluating this system using both datasets. Another aspect to remark is the analysis on feature importance. The user not only has the possibility to select which feature to extract but also carry out an analysis to quantify the influence of selected features in the models prediction. The main limitation is the behavior of the classifier models in front of the presence of selected features in the models prediction. The main limitation is the behavior of the classifier models in front of new data, which still shows low accuracy. This means that to apply the system in a new house, it has to be retrained first. However, for new data of the same house, the performance is stable with regard to the training process. The fact of including multiple datasets, and the possibility of choosing the right parameter values, give certain standardization to the proposed frameworks, since it is possible to adapt the system configuration according to the problem needs. In the future, more datasets could be included, converting the frameworks into a very useful tool for researchers. In addition, our work will be focused on designing and implementing the complete IoT platform in a laboratory environment.

REFERENCES
[1] Y. Kabalci, E. Kabalci, S. Padmanaban, J. B. Holm-Nielsen, and F. Blaabjerg, “Internet of Things applications as energy internet in smart grids and smart environments,” *Electronics*, vol. 8, no. 9, p. 972, Aug. 2019.
[2] S. Tan, Y. Wu, P. Xie, J. M. Guerrero, J. C. Vasquez, and A. Abusorrah, “New challenges in the design of microgrid systems: Communication networks, cyber-attacks, and resilience.” *IEEE Electr. Mag.*, vol. 8, no. 4, pp. 98–106, Dec. 2020.
[3] Y. Wu, Y. Wu, J. M. Guerrero, J. C. Vasquez, E. J. Palacios-Garcia, and J. Li, “Convergence and interoperability for the internet energy: From ubiquitous connection to distributed automation,” *IEEE Ind. Electron. Mag.*, vol. 14, no. 4, pp. 91–105, Dec. 2020.
[4] J. Leitao, P. Gil, B. Ribeiro, and A. Cardoso, “A survey on home energy management,” *IEEE Access*, vol. 8, pp. 5669–5722, 2020.
[5] B. Hartono, S. P. Mursid, and S. Prajogo, “Review: Home energy management system in a smart grid scheme to improve reliability of power systems,” *IOP Conf. Ser., Earth Environ. Sci.*, vol. 105, Jan. 2018, Art. no. 012081.
[6] C. Liu, K. T. Chau, D. Wu, and S. Gao, “Opportunities and challenges of vehicle-to-home, vehicle-to-vehicle, and vehicle-to-grid technologies,” *Proc. IEEE*, vol. 101, no. 11, pp. 2409–2427, Nov. 2013.
[7] M. A. Devlin and B. P. Hayes, “Non-intrusive load monitoring and classification of activities of daily living using residential smart meter data,” *IEEE Trans. Consum. Electron.*, vol. 65, no. 3, pp. 339–348, Aug. 2019.
[8] X. Yuan, P. Han, Y. Duan, R. E. Alden, V. Rallabandi, and D. M. Ionel, “Residential electrical load monitoring and modeling–state of the art and future trends for smart homes and grids,” *Electr. Power Compon. Syst.*, vol. 48, pp. 1–19, Nov. 2020.
[9] D. Burbano, “Intrusive and non-intrusive load monitoring (a survey) inference and learning approach.” *Latin Amer. J. Comput.*, vol. 2, no. 1, p. 9, 2015.
[10] A. U. Rehman, T. T. Lie, B. Vallès, and S. Tito, “Low complexity event detection algorithm for non-intrusive load monitoring systems,” in *Proc. IEEE Innov. Smart Grid Technol. Asia (ISGT Asia)*, May 2018, p. 751.
[11] A. U. Rehman, T. T. Lie, B. Vallès, and S. R. Tito, “Event-detection algorithms for low sampling nonintrusive load monitoring systems based on low complexity statistical features,” *IEEE Trans. Instrum. Meas.*, vol. 69, no. 3, pp. 751–759, Mar. 2020.
[12] R.-C. Mihailescu, D. Hurtig, and C. Olsson, “End-to-end anytime solution for appliance recognition based on high-resolution current sensing with few-shot learning.” *Internet Things*, vol. 11, Sep. 2020, Art. no. 100263.
[13] F. Paradiso, F. Paganelli, A. Luchetta, D. Giuli, and P. Castrogiovanni, “ANN-based appliance recognition from low-frequency energy monitoring data,” in *Proc. IEEE 14th Int. Symp. World Wireless, Mobile Multimed. Netw. (WoWMoM)*, Madrid, Spain, Jun. 2013, pp. 1–6.
[14] A. Ridi, C. Gisler, and J. Hennebert, “A survey on intrusive load monitoring for appliance recognition,” in *Proc. 22nd Int. Conf. Pattern Recognit.*, Stockholm, Sweden, Aug. 2014, pp. 3702–3707.
[15] F. Bouhafs, M. Mackay, and M. Merabti, “Links to the future: Communication requirements and challenges in the smart grid,” *IEEE Power Energy Mag.*, vol. 10, no. 1, pp. 24–32, Jan. 2012.
[16] C. Chalimers, P. Fergus, C. A. Curbelo Montanez, S. Sikdar, F. Ball, and B. Kendall, “Detecting activities of daily living and routine behaviours in dementia patients living alone using smart meter data disaggregation,” *IEEE Trans. Emerg. Topics Comput.*, early access, May 11, 2020, doi: 10.1109/TETC.2020.2993177.
[17] P. Franco, J. M. Martinez, Y.-C. Kim, and M. A. Ahmed, “IoT based approach for load monitoring and activity recognition in smart homes,” *IEEE Access*, vol. 9, pp. 45325–45339, 2021.
[18] R. Alekya, N. D. Boddeti, K. S. Monica, D. R. Prabha, D. V. Venkatesh, and K. Ashton, “IoT based smart healthcare monitoring systems: A literature review,” *Clin. Med.*, vol. 7, no. 11, p. 9, 2020.
[19] S. Deep, X. Zheng, C. Karmakar, D. Yu, L. G. C. Hamey, and J. Jin, “A survey on anomalous behavior detection for elderly care using dense-sensing networks,” *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 352–370, 1st Quart, 2020.
[20] A. Bündis, A. Faizan, and M. Mühlhäuser, “A feature-based model for the identification of electrical devices in smart environments,” *Sensors*, vol. 19, no. 11, p. 2611, Jun. 2019.
[21] J. R. Cuñado and N. B. Linsangan, “A supervised learning approach to appliance classification based on power consumption traces analysis,” *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 517, Apr. 2019, Art. no. 012011.
[22] L. G. Fahad and S. F. Tahir, “Activity recognition and anomaly detection in smart homes,” *Neurocomputing*, vol. 423, pp. 362–372, Jan 2021.
[23] A. U. Rehman, “Load disaggregation: Towards energy efficient systems,” M.S. thesis, Auckland Univ. Technol., Auckland, New Zealand, 2021.
[24] P. Brandstetter, J. Vanek, and T. Verner, “Electric vehicle energy consumption monitoring,” in *Proc. 15th Int. Symp. Conf. Electr. Power Eng. (EPE)*, Brno-Bystro, Czech Republic, May 2014, pp. 589–592.
[25] K. G. Kim, “Book review: Deep learning,” *Healthcare Informat. Res.*, vol. 22, no. 4, p. 351, 2016.
[26] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[27] UK Domestic Appliance-Level Electricity (UK-DALE) Dataset. Accessed: Feb. 2021. [Online]. Available: https://jack-kelly.com/data/ Dataport | Residential Data. Accessed: Apr. 2021. [Online]. Available: https://dataport.pecanstreet.org/. Accessed: Feb. 2021. [Online]. Available: https://www.tesla.com/.
[28] Electric Cars, Solar & Clean Energy Dataport | Residential Data. Accessed: May 2021. [Online]. Available: http://www.tesla.com/.
[29] R. Shanmugam and R. Chatamvelli, *Statistics for Scientists and Engineers*. Hoboken, NJ, USA: Wiley, 2015.
[30] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002. [Online]. Available: https://arxiv.org/abs/1106.1813
PATRICIA FRANCO received the bachelor’s degree in telecommunications and electronics engineering from the Universidad Tecnológica de La Habana José Antonio Echeverría, Cuba, in 2017. She is currently pursuing the Ph.D. degree in electronic engineering with Universidad Técnica Federico Santa María, Chile. Her research interests include smart grids, human activity recognition, and load monitoring.

JOSÉ MANUEL MARTÍNEZ (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in electronic engineering from Universidad Técnica Federico Santa María (UTFSM), Chile, in 2000, 2000, and 2014, respectively. He is currently working as an Assistant Professor with the Department of Electronic Engineering, UTFSM. From 2013 to 2018, he was a Visiting Assistant Professor with the Department of Electrical and Computer Engineering, Duke University, NC, USA. His research interests include performance-dependability modeling and analysis of ICT systems, Markovian and semi-Markovian stochastic processes, stochastic Petri nets, the Internet of Things, smart grids, and blockchain technology.

YOUNG-CHON KIM received the B.S., M.S., and Ph.D. degrees from Korea University, Seoul, South Korea, in 1980, 1982, and 1987, respectively. He joined the Department of Computer Engineering, Chonbuk National University, Jeonju, South Korea, in 1986, where he is currently a Professor with the School of Information Technology. From 1989 to 1990, he was a Postdoctoral Fellow with the University of California at Irvine. From 1998 to 2000, he worked as a Visiting Researcher with the Network Research Laboratory, University of California at Davis. His research interests include high-speed optical communication networks, next generation IP networks, and broadband wireless communication networks. He was awarded the Best Professor from the College of Engineering, Chonbuk National University, in 2002, and Motorola Academic Excellence Award, in 2003.

MOHAMED A. AHMED (Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from Minia University, Minya, Egypt, in 2003 and 2007, respectively, with a focus on electronics and communications, and the Ph.D. degree in electronics and information engineering from Chonbuk National University, Jeonju, South Korea, in 2014. He is currently working as an Assistant Professor with the Department of Electronic Engineering, Universidad Técnica Federico Santa María (UTFSM), Chile. He is also an Assistant Professor with the Department of Communications and Electronics, Higher Institute of Engineering and Technology-King Marouf, Alexandria, Egypt. Prior to joining UTFSM, he worked as a Postdoctoral Research Fellow with the Advanced Communication and Network Laboratory (ADCAN), Chonbuk National University, from 2014 to 2018. His research interests include wireless sensor networks, wind energy, electric vehicles, smart grid, and cyber-physical systems.