energies

Article

Isolated Areas Consumption Short-Term Forecasting Method

Guillaume Guerard *, Hugo Pousseur † and Ihab Taleb

Research Center, Leonard de Vinci Pole Universitaire, 92916 Paris La Défense, France; hugo.pousseur@edu.devinci.fr (H.P.); ihab.taleb@devinci.fr (I.T.)
* Correspondence: guillaume.guerard@devinci.fr
† Student in Master Thesis.

Abstract: Forecasting consumption in isolated areas represents a challenging problem typically resolved using deep learning or huge mathematical models with various dimensions. Those models require expertise in metering and algorithms and the equipment needs to be frequently maintained. In the context of the MAESHA H2020 project, most of the consumers and producers are isolated. Forecasting becomes more difficult due to the lack of external data and the significant impact of human behaviors on those small systems. The proposed approach is based on data sequencing, sequential mining, and pattern mining to infer the results into a Hidden Markov Model. It only needs the consumption and production curve as a time series and adapts itself to provide the forecast. Our method gives a better forecast than other prediction machines and deep-learning methods used in literature review.

Keywords: time series; Hidden Markov Model; short-term forecast

check for updates

Citation: Guerard, G.; Pousseur, H.; Taleb, I. Isolated Areas Consumption Short-Term Forecasting Method. *Energies* **2021**, *14*, 7914. https://doi.org/10.3390/en14237914

Academic Editors: Zbigniew Leonowicz and Michał Jasinski

Received: 25 October 2021
Accepted: 19 November 2021
Published: 25 November 2021

1. Introduction

The power grid became a smart grid [1]. In fact, from the first electric transmission line installed by Thomas Edison in 1882 to nowadays super-large electric grids, power systems evolved with the advance and progress of human industrial civilization. The power grid has evolved in how electricity is generated and distributed.

Over the decade, electricity forecasting (or energy forecasting) focuses not only on the grid side of power grids but also on the client-side to balance load and demand. Short-term consumption forecasting is used to manage, plan, and schedule energy use in various structures, from small-scale houses to vast complexes as factories, malls, or hospitals. It makes it possible to monitor energy consumption, reduce losses, minimize risks, maintain reliability for uninterrupted operation. As a decision-making tool, consumption and production forecasting are used in simulation to plan future investments and where to invest, to schedule maintenance, to manage plants use and battery use.

To optimize and regulate energy flows and prices, the electric system requires the knowledge of future short-term consumption. This means knowing the global consumption and any local consumption, i.e., scaling from a country to a house, to be able to forecast their future. Estimating energy consumption is one of the critical challenges of our time and yet, the interest in this field is still increasing. Besides the energy flow optimization, predicting energy consumption allows us to acquire a more thorough understanding of our modes of consumption.

There is currently no standard defining the types of forecasts. *Hong and Fan* have classified very short term, short term, medium term, and long term forecasts into cut-off horizons of 1 day, 2 weeks, and 3 years [2]. In this paper, we consider only very short-term forecasting with cut-off horizons of 0.5 to 4 h.

Our study's context concerns the European islands, more especially French Mayotte island through the H2020 MAESHA project (https://www.maesha.eu/, accessed on 15 June 2021). MAESHA is the acronym for "deMonstration of smArt and flexible solutions for a decarboniSed energy future in Mayotte and other European islAnds", but also means "future" in Shimaore, the local dialect of the French island of Mayotte that will host the demonstration site. Islands use mainly thermal energies to produce electricity, for example, Mayotte has sixteen diesel engines producing 95% of its energy in 2018 (Source: French energies regulator CRE 2020's report.). Followers include Futuna islands, Saint Barthélémy, Canary islands, Favignana and Gozo.

The goal of the H2020 MAESHA project is to simulate the grid to improve flexibility and include renewable energies and demand-side strategies. Indeed, islands are disconnected from the mainland grid and are primarily dependent on coal. Since the grid is impacted by the geography and activities on the island, coal offers the most flexible and instant source of power to maintain the offer according to the demand. However, the islands tend to cut off the use of coal to include more renewable energies and batteries.

Islands contain many houses, factories, warehouses, farms only connected to the grid by one point. One key challenge is to be able to predict the consumption and production of those isolated areas to manage the offer and demand as well as an appropriate distribution of energy across the islands. To predict isolated areas, we formulate the hypothesis to be able to predict the consumption or production of each device of this area. It will provide a more accurate forecast than a prediction of the whole system.

We define three principal applications of our forecasting tool after being deployed at Mayotte islands:

1. CONSUMPTION FORECAST: The sum of the forecast of each device produces a prediction of an isolated system.
2. CONSUMPTION SCHEDULING: Since each appliance produces its prediction, one can schedule their consumption over time with smart appliances or smart house control.
3. DEMAND-SIDE MANAGEMENT: One may determine the effect of demand-side management strategies on their consumption. It is therefore possible to generate strategies to reduce consumption, switch off certain devices or reduce consumption of appliances, or to plan their renewable energies and batteries with the market.

Concerning the proposed method, it is based on various processes to optimize the accuracy of the forecast and to be adapted to any consumption or production curves. Here are the insights of our paper:

- We provide a method to transform a time series into sequences. Those sequences represent the cycles of consumption/production of devices.
- The sequences are reworked to limit the noise while keeping recurrent motifs as well as outliers. This processing produces a more understandable and practical set of sequences.
- To be able to perform forecasts, we build through ALERGIA algorithm a Hidden Markov Model for each device based on their reworked sequences. A Hidden Markov Model is flexible and can be adapted and updated with new data.
- In our paper, we provide a step-by-step analysis of our method. Moreover, we compare our results with well-known methods used in forecasting.

After presenting the literature review in the following Section 2, we present the materials and the method in the Section 3. The rest paper is organized as follows:

1. DATASET: first of all, time series are decomposed into sequences (See Section 4).
2. MOTIFS DISCOVERY: to increase the quality of the forecast, a motif discovery algorithm is applied to the sequence (See Section 5).
3. GRAMMATICAL INFERENCE: this method builds a sequence prediction machine as a Hidden Markov Model (See Section 6).
4. VALIDATION: we also implemented famous forecasting methods to compare with our method (See Section 7).
5. EXPERIMENTS: the Sections 8 and 9 show a pedagogical instance and the general results with comparison and step by step analysis.

We conclude the paper in Section 10.

2. Literature Review

Since energy consumption is a set of ordered values representing an evolution of a quantity over time, it is handled as a time series forecasting problem. Current contributions in this field use data-driven models to predict consumption over time. In the following literature review, we cite popular articles, based on high-rank journals and conferences (CORE rank). We also privilege current trends, without being exhaustive, by citing less than five years old articles if possible.

2.1. Context and Surveys

Many surveys list forecasting methods. Zhao et al. [3], Daut et al. [4], Wang et al. [5] categorize those methods in three categories: mathematical and statistical methods; Artificial Intelligence (AI) based methods; and hybrid methods, combining the two mentioned. They also provide useful references and compare those methods. About the two last categories, Raza et al. [6] references various AI-based and hybrid methods at various short-term forecasting scales, from building to a country.

Predicting the energy consumption is frequently done in the short term, almost 60% of the studies employing data-driven models make hourly predictions [7]. It is equally interesting to see that most of the studies focus mainly on the Cooling/Heating (HVAC) consumption estimation. Indeed, HVAC represents between 40 and 50% of the overall consumption of vast buildings like offices, faculties or hotels [8]. Forecasting this consumption is often easier since the HVAC is fairly continuous over time and depends on external parameters like local weather or time of the year [9]. Note that no study considers all sockets/appliances of a household to predict the whole system.

Since energy consumption prediction is a time series forecasting problem, it's not surprising to find familiar methods like AutoRegressive Integrated Moving Average (ARIMA), applied to this field. As it is shown in this research work [10], the most used techniques are ARIMA, Support Vector Machines (SVM), and Artificial Neural Networks (ANN). Some models combine neural networks with more classic models of time series forecasting. This combination aims to strengthen predictions and increase performance. They are designated as hybrid models.

2.2. Most-Used Methods

Although getting old, ARIMA is still being used for its simplicity, its accuracy improved by moving average and its confidence interval. ARIMA is unsuited for long-term prediction and cannot deal with nonlinear relationships. Despite more robust models being used, making ARIMA less privileged nowadays, this method can be found combined with other techniques as a hybrid model. Making a hybrid model with ARIMA allows to combine it with techniques that can support non-linear relationships like SVM or ANN [11–13].

ANNs are the most used method in forecasting building energy consumption. They are widely employed in this field for their numerous advantages. Indeed, the complexity of this task is considerable due to several factors/parameters which are a well-suited problem for ANNs and their capacity to deal with non-linear relationships. ANNs are extremely robust and flexible, especially MultiLayer Perceptron (MLP). They are easy to implement but require some specialized knowledge to configure them. They can be employed alone [14], they can also be combined with other models to get a hybrid model [11,13]. One of the disadvantages is the massive amount of data required to train the network. If there aren't enough data to train the ANN, it will have difficulties to generalize and risks to overfit.

A type of neural network widely used in the field of time series forecasting is the Long Short-Term Memory (LSTM). It's an artificial recurrent neural network architecture that can process entire sequences of data, making it a privileged model for handwriting recognition, speech recognition, and time-series data. It can be implemented alone [15,16] or with a convolutional layer for better results [17].

2.3. Forecasting Isolated Areas

One problem is frequently overlooked or ignored: how to predict the energy consumption of an isolated house, with only its consumption data. Indeed, a single household is strongly dependant on the inhabitants. Building a good forecasting model is hard due to the human factor, the lack of external context (no sensors), and the small or null amount of energy consumed most of the day.

Our study is based on sequence prediction machines. They build a prediction or decision tree from the time series. Those methods are mostly ignored in energy forecasting because of the massive number of heterogeneous data. However, those methods provide outstanding results with a homogeneous dataset like Deoxyribonucleic acid (DNA) forecasting. The key challenge of our method is to build an understandable and useful dataset to be used by sequence prediction machines.

3. Materials and Methods

3.1. Materials

Mayotte island contains various cities and villages close to the coast and rivers as shown in the density map Figure 1. The mainland is essentially mountains, quickly going up to 900 m above sea level. The economy of Mayotte is essentially based on agriculture, fishing, and aquaculture.

The island has only one high voltage line, energy circulates across the island through low voltage. The network forms the shape of a bubble, but inland areas are mainly served by linear lines which made the network unstable. Moreover, the two diesel plants are in the north of the island; in the south many brownouts and blackouts happen every day (Figure 2). Isolated areas frequently cause discrepancies between demand and distribution, triggering local outages or even cascading over part of the network. Our study is a necessity for the network, before considering building new plants and new batteries.

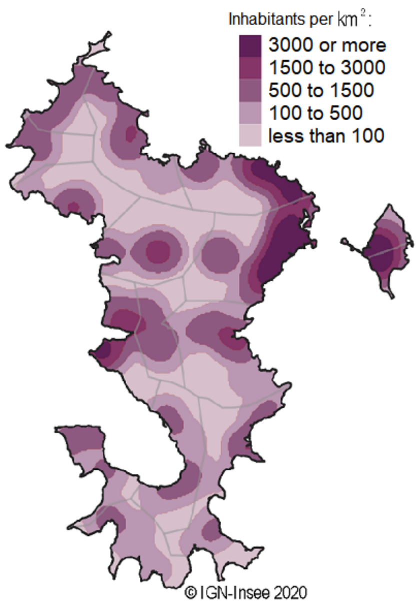

Figure 1. Density map of Mayotte from IGN-Ensee 2020©.

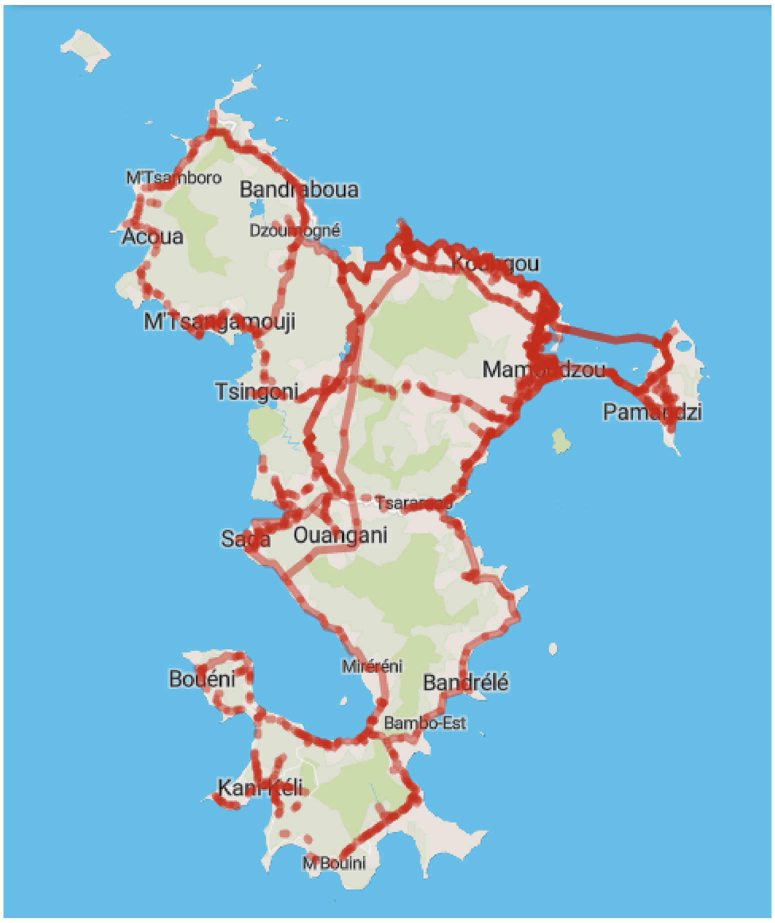

Figure 2. Electric low voltage network of Mayotte from EDM Open Data 2021©.

Since there are no current data for isolated areas in the Mayotte island, the proposed method has been evaluated on the SMART* (http://traces.cs.umass.edu/index.php/Smart/Smart, (accessed on 8 May 2021)) databases. This dataset is composed of seven instrumented and isolated smart homes gathering real-life data each minute for more than two years. Each device, appliance, or socket gets its time series. Those data are consolidated, i.e., they do not have missing data or abnormal values. Since we aim to forecast isolated areas in many European islands, this dataset perfectly suits our goal. The Table 1 presents a dataset overview. The columns refer respectively to the home ID, the number of meters, the percent of total energy from renewable energy.

Table 1. Analysis of the dataset.

HomeID	# Meters	%RE
A	31	0
B	26	0
C	20	10
D	70	39
F	22	73
G	45	0
H	18	0

3.2. Methods

The goal of our method is to build a prediction machine from a time series. Indeed, in isolated areas, it's easy to add meters to any socket, but it becomes harder and more expansive to instrumentalize the area and locally compute a deep-learning method. After preprocessing the time series into a set of sequences, we implement clustering methods, sequence mining, and motif mining (common subsequences) to transform those sequences into meaningful sequences. They are inferred into a Hidden Markov Model, a data structure well-known to generate the next items given a sequence. This flexible prediction machine can learn if the forecast encounters new patterns.

The proposed method contains various steps explained in the following sections. To be more understandable, we provide in this section an overview of the method with the goals of each part. Figure 3 presents the steps of the proposed method. We briefly explain each part with its purpose:

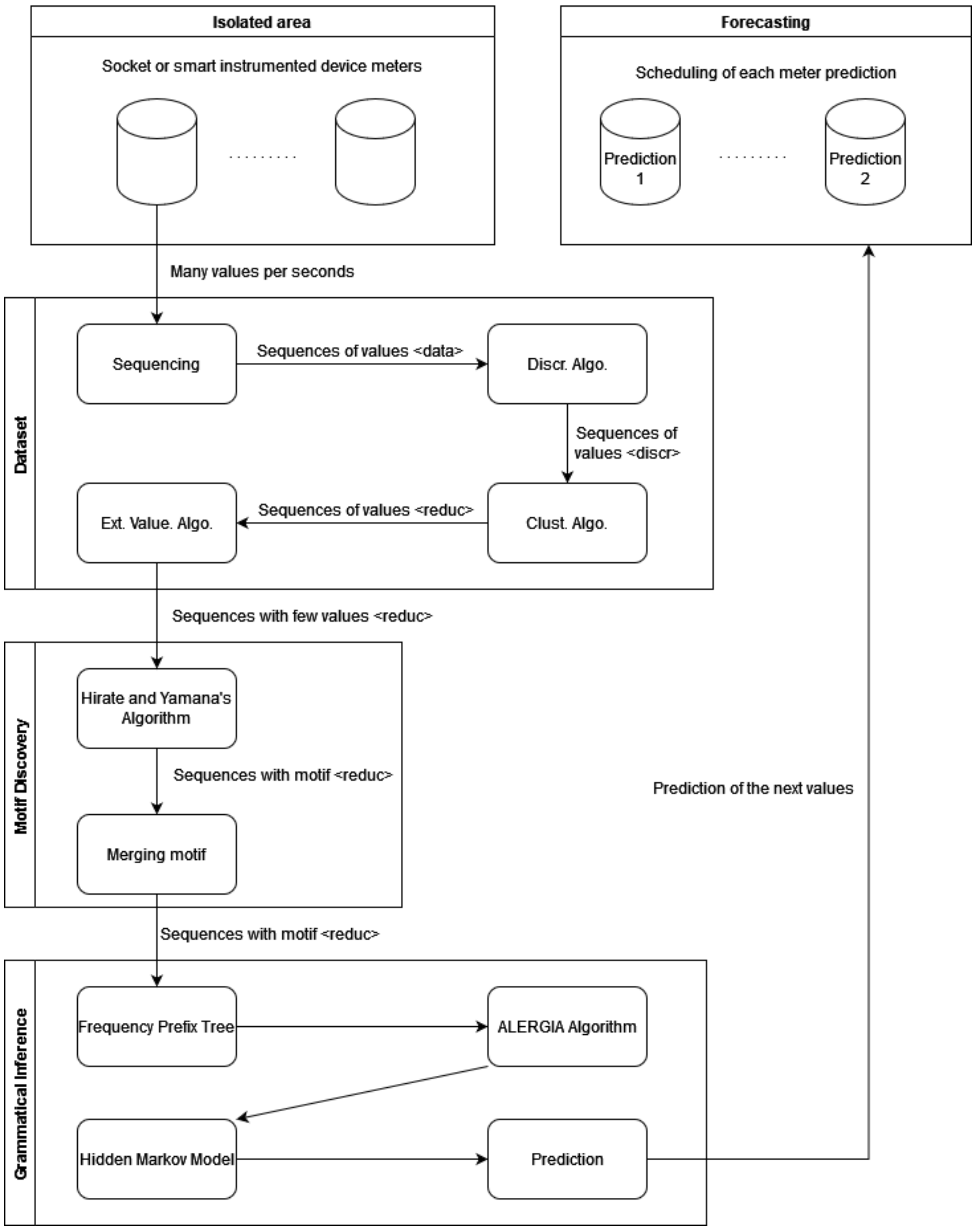
Isolated area
Socket or smart instrumented device meters
Forecasting
Scheduling of each meter prediction
Prediction 1
Prediction 2
Many values per seconds
Dataset
Sequencing
Sequences of values <data>
Discr. Algo.
Sequences of values <discr>
Clust. Algo.
Sequences of values <reduc>
Ext. Value. Algo.
Sequences with few values <reduc>
Motif Discovery
Hirate and Yamana's Algorithm
Sequences with motif <reduc>
Merging motif
Sequences with motif <reduc>
Prediction of the next values
Grammatical Inference
Frequency Prefix Tree
ALERGIA Algorithm
Hidden Markov Model
Prediction

Figure 3. Overview of the method.

1. METERING: Some meters report the consumption or production (we use consumption for the following explanation) of devices or sockets in the isolated area. Each meter provides a time series with a timestep in milliseconds.
2. DATASET: Since meters give several values per second, data are preprocessed. The first step is to cut the time series into sequences of consumption. Then, values are summed together at a common timestep. Values of consumption are replaced by the

slope between two values to better reflect the behavior of the device. To limit the number of unique values, we use two clustering algorithms and a new algorithm to retrieve extreme values hidden in the clusters. This step produces a meaningful dataset for any prediction algorithm, especially those based on sequence mining.

3. MOTIF DISCOVERY: To increase the knowledge contained in the sequences, we implement two algorithms to determine recurrent motifs of consumption inside the sequences. This step increases the range of prediction. Indeed, since a motif starts to be predicted, the whole motif of consumption is already known.
4. GRAMMATICAL INFERENCE: Most of the sequence prediction machines can only predict known sequences or cannot predict further than one recognized sequence. To build a more accurate and complete prediction machine, we transform the sequence into a Hidden Markov Model, a combination of Markov Process and Automaton. This model possesses an algorithm to update the probability and to predict future values.
5. FORECASTING: Since we can forecast the consumption for each meter, we can schedule or sum the future consumption and production of the isolated area.

4. Dataset's Building

To know the consumption of an appliance, a meter is plugged. Each appliance produces a curve of consumption. This one saves a time series where each tuple delivers the following information: ID_device, consumption (in Wh), and date. Each time series (one per appliance) is cut into various sequences as described below. Those sequences form the dataset used by our method to produce a forecast.

4.1. Sequence

A sequence of consumption starts from a zero value of consumption to a non-zero value of consumption on two consecutive time steps. As soon as there is a zero-value of consumption at the following step, the sequence stops. Some devices may have programmed cycles of consumption with small break times. The following method merges two sequences of consumption if it is considered on the same cycle:

$$\Delta B \leqslant \Delta T_i \text{ and } \Delta B \leqslant \Delta T_j$$

with the following variables: ΔB the break duration; ΔT_i the ith sequence of consumption duration; ΔT_j: jth sequence of consumption duration. The Figure 4 presents an example of merging multiple sequences (separated by vertical lines) to constitute a unique cycle of consumption.

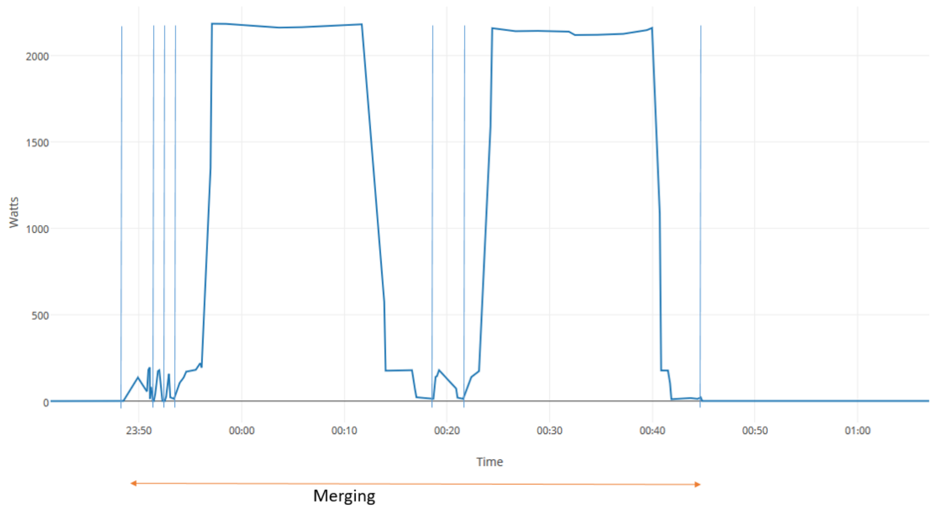

Figure 4. A washing machine consumption's cycle.

Since most of the appliance consumes a little amount of energy in idle mode, the time series must be studied to determine how to set a "zero" value, below this threshold, all values are considered as zero. The consumption curve is studied to set the "zero" value thanks to some mathematical tools as histogram, distribution function before the method, and square error after appreciating the results. They provide an overview of the data to find a good threshold for the "zero" value.

Given the enormous number of values that all the sequences of an appliance may produce, it is relevant to smooth the data by the following methods: a data discretizing named *Discr. Algo.*, a data clustering named *Clust. Algo.*, an algorithm to retrieve extreme values named *Ext. Value. Algo*.

4.2. Data Discretizing

Since most meters provide several values per second, we need to provide a suitable structure for any time series. *Discr. Algo.* provide a discretized dataset at a constant timer which starts at the value of 5 min. Put differently, we sum the instant consumption of the device for 5 min. Thus, the curve of consumption only shows the consumption value every 5 min.

Once all the consumption's values are computed in the following dataset *discretized data*, the integral of the *data* curve is compared to the integral of the *discretized data* curve. The ratio between those two curves provides an estimation of the loss of information. This ratio must not exceed a value α.

As long as this ratio is strictly less than α, it is possible to consider the data with a greater timer between two values. On the contrary, if the ratio is greater than α, one needs to shorten the timer between two values. The algorithm stops when the ratio is close to α. The process is described in Algorithm 1.

Algorithm 1: *Discr. Algo.*

```
Input  : dataset of consumption named $\langle data \rangle$
Output: a discretised dataset of consumption named $\langle discr \rangle$
$\alpha \in [0, 1]$
$\epsilon << \alpha$
$timer \leftarrow 5\ min$
Function Discretizing(data α) is
$\langle discr \rangle \leftarrow\ empty$
Integral_data ← integral of the curve generated by $\langle data \rangle$
test ← $true$
while test do
    $Integral\_discr \leftarrow 0$
    $x_i \leftarrow$ first value of $\langle data \rangle$
    insert $x_i$ in $\langle discr \rangle$
    foreach $x_j$ from $\langle data \rangle$ at a timer ahead $x_i$ do
        insert $x_j$ in $\langle discr \rangle$
        $Integral\_discr+ = (\frac{(x_j - x_i)}{2} + min(x_i, x_j)) * timer$
        $x_i \leftarrow x_j$
    $incr$ is the incremental difference between $\langle discr \rangle$ and $\langle data \rangle$
    if $\frac{incr}{Integral\_data} < \alpha - \epsilon$ then
        increase the value of $timer$
        reset $\langle discr \rangle$
    if $\frac{incr}{Integral\_data} > \alpha + \epsilon$ then
        decrease the value of $timer$
        reset $\langle discr \rangle$
    else
        test ← $false$
```

4.3. Slope between Two Values

We transform the value in each sequence. We don't consider the value of consumption but compute the slope between two values. The slope provides a more thorough understanding of the device's behavior than its consumption in Wh. Indeed, some smart appliances have similar cycles of consumption but at various scales of consumption.

Each data of a sequence is referenced with: its value, its sequence ID, its position in the sequence, and from the second position the value refers to the slope between two consecutive values.

4.4. Data Clustering

The data may contain millions of distinctive slope values. Reducing the noise in each sequence is useless since sequences have less than one hundred values. In consequence, we need to reduce the noise in the whole dataset by a clustering method.

Clustering represents the process of determining typical groups, called clusters, in a dataset. The objective is to find homogeneous clusters that are as distinct as possible from other clusters. More formally, the grouping should maximize inter-cluster variance while minimizing intra-cluster variance.

A classification of clustering methods for various static data is proposed in [18]. In the context of our dataset, agglomerative hierarchical-based clustering is suitable because it considers the distance between each value. It runs the distance matrix and plots a dendrogram that displays a hierarchical relationship among the data based on the nearest neighbors for each value. The metric is the Euclidian distance between each value.

To determine the number of cuts, the Elbow method is employed. The total Within-cluster Sum of Squares (WSS) in the dataset is computed for various cuts (i.e., in the function of the number of the cluster). The WSS curve forms an elbow, the number of cuts is equal to the value where the WSS starts to be stabilized. Then, two hierarchical algorithms are implemented:

1. DIvise ANAlysis clustering: DIANA is completely described in Kaufmann and Rousseeuw's book [19]. At each iteration, the cluster with the largest dissimilarity between any two of its observations is selected. The algorithm selects its most disparate observation. Those values initiate the splintering process, and the algorithm reassigns closer observations to both splinters.
2. AGglomerative NESting: AGNES is thoroughly described in Kaufmann and Rousseeuw's book [19]. It constructs a hierarchy of clusterings. Initially, each observation remains a small cluster by itself. Clusters are merged until only one large cluster remains which contains all the observations. At each stage, the two nearest clusters are combined to form one larger cluster.

Various methods are implemented as Ward, Complete linkage, or Flexible linkage to offer various hierarchies to maximize the Silhouette and minimize the WSS.

The Silhouette refers to a method of interpretation and validation of consistency in data clustering and provides a graphical representation of how well each point has been classified.

Let $a(i)$ be the average distance between a point i to all other data points in the same cluster; $b(i)$ be the smallest average distance of point i to any other cluster of which i is not in the set. The Silhouette of point i is defined as follows:

$$s(i) = \begin{cases} 1 - a(i)/b(i), \text{if } a(i) < b(i) \\ 0, \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, \text{if } a(i) > b(i) \end{cases}$$

the Silhouette provides a result between -1 and 1. An $s(i)$ close to one means that the data is appropriately clustered. A value of -1 means the data is in the wrong cluster. In other words, the Silhouette qualifies if the data fit in its cluster or not. A mean Silhouette close to 1 induced the lowest WSS.

After the hierarchical cut, the data are assimilated to their respective nearest cluster center (as the mean of its data). The data with a small Silhouette or with a negative Silhouette means the value will be replaced with an inappropriate one. To decrease the loss of information due to a small Silhouette's value, a K-means algorithm adjusts the observations to the good cluster. Note that the mean of each cluster and the affectation of the observations may differ from the hierarchical clustering.

Since the clustering methods computed the distance between data, in consequence, the dataset must be sampled to avoid memory shortage. The sample must be representative of the diversity of the values. A Monte-Carlo procedure is employed to re-sample a given observation at each level of sampling. The expectation is the mean of the re-sampling will approach Shannon's diversity index at that sample level.

Since we minimize the WSS, each data is now as close to its cluster center as possible. At this step, all values of the sequence are now changed to their respective cluster means. The number of unique values is drastically reduced.

4.5. Extreme Points Retrieval

Once the K-means converges, the Silhouette is computed again. If the Silhouette of some points is not higher than a threshold $0 < silh < 1$, then those points are considered as *noise*. Indeed, the more the Silhouette is close to zero, the more the WSS increase. Put differently, the values with a Silhouette under the threshold *silh* decrease the quality of the clustering, they better fit with a new cluster. A new cluster is produced with each *noise* point.

Otherwise, if the Silhouette provides a negative value, the point is conceded to the corresponding cluster. Then, the Silhouette are updated. This process stops when there are no more *noise* points. Note that through iterations, some *noise* points may be grouped.

This process is useful to keep extreme and isolated values that have been erased through the clustering process. Indeed, those values refer to extreme slope values (mostly very high values) and we need those to accurately predict the consumption.

The proposed method is linear on numbers of clusters and numbers of iteration, and log-linear on the number of points. The Algorithm 2 describes how to retrieve the extreme points *Ext. Value. Algo.* Following this process, each value is changed to their cluster mean.

Algorithm 2: *Ext. Value. Algo.*

```
Input  : Kmeans results $\langle reduc \rangle$, a Silhouette threshold $silh$.
Output : $\langle reduc \rangle$ with extremal values.
Function Retrieval($\langle reduc \rangle$, $silh$) is
while A cluster has a point with a Silhouette $< silh$ do
    foreach Points of this cluster do
        if The silhouette is negative then
            Give the point to the corresponding cluster
        if The silhouette is $< silh$ then
            Put the point in a new cluster $C$
    Compute Silhouette
```

5. Motifs Discovery

At this point, the sequences are meaningful but can include various values. We formulate the hypothesis that some subsequences appear multiple times in the dataset. A device may have consumption patterns/schemes which happen in several sequences. To limit the length of the sequences and to improve the effectiveness of the prediction, those schemes should be considered as a unique symbol, i.e., once the pattern is detected or suggested, the prediction will provide all the concerned motifs.

This section is composed of *Disco. Motif. Algo.*, an algorithm designed to discover a frequent motif inside the dataset.

Sequential pattern mining is an important data mining method that can extract similar subsequences while maintaining their order. However, it is critical to identify data intervals of sequential patterns extracted by sequential pattern mining. *Disco. Motif. Algo.* is based on *Hirate and Yamana* [20] generalized sequential pattern mining with data interval.

Those intervals can be a data gap and/or a time interval. The motif on the consumption scheme must be continuous and contiguous. The generalized sequential pattern mining is set on contiguous data with no time interval.

Besides the common subsequence, its Support is indicated. The Support provides the frequency of this subsequence among the dataset but does not give if the motif is recurrent inside the same sequence. After finding the motifs, only the ones with the most significant length and Support are considered without any conflict on data.

To resume, the *Disco. Motif. Algo.* is as follows:

1. Implement *Hirate and Yamana*'s Algorithm.
2. Use the algorithm on the dataset $\langle reduc \rangle$ from Section 4.4.
3. Return a list of motifs and their support.
4. Merge the symbol of the motif in the dataset from the top motif's length and Support to the lowest. Do not merge if data is already in a better motif.

Let's observe the process through an example. We consider three sequences where each letter refers to a cluster's mean: *aaabacc*, *bacaaa*, *ccbacaaa*. *Hirate and Yamana*'s Algorithm provides, for example, subsequences *bac*, *aaa*, *ba*, *aa* with a support of 100% because those sequences are observed in the three sequences. For example, the subsequence *cc* has support of 66% because of its presence in the first and third sequence. In conclusion, we rewrite the three sequences considering in braces the motif: $(aaa)(bac)c$, $(bac)(aaa)$ and $cc(bac)(aaa)$.

Time series motifs are pairs of individual sequences, or subsequences of a longer sequence, which are extremely similar to each other. One hypothesis is made after *Disco. Motif. Algo.*: some recurrent motifs may differ by at least one symbol.

The goal of *Ext. Motif. Algo.* is to discover extended versions of the motifs in the sequences. For example, the sequences *aaaeaaa* and *aaafaaa* only differ about one symbol. If the symbol e and f have close values, the two symbols are considered equal with a new symbol $g = \frac{e+f}{2}$.

The Figure 5 represents the sequences in Figure 4 after *Ext. Motif. Algo.* Each motif is identified by a number and a color (from 1 to 4).

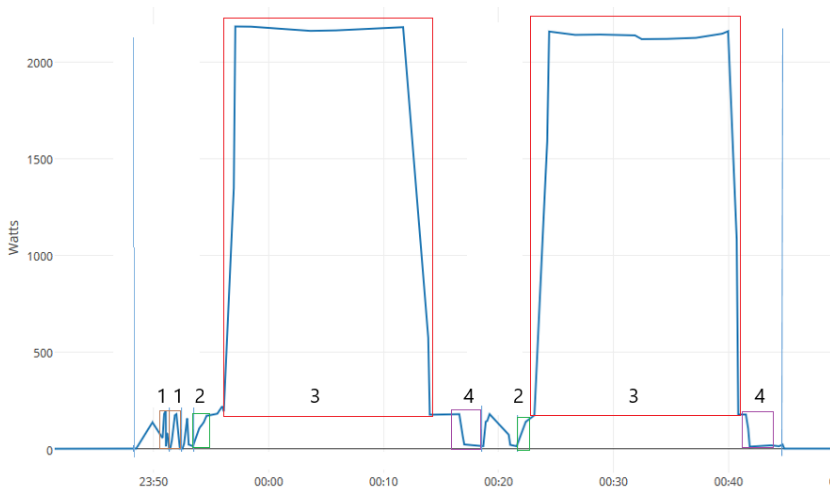

Figure 5. A washing machine consumption's curve after motif discovery.

6. Grammatical Inference

Following the previous steps, the cycles of consumption are now composed of cluster's means and motifs. The goal of this section is to build a prediction machine from those sequences. First of all, the sequences are displayed into a frequency prefix tree, also known as a trie.

Since the trie is PAC-learnable, for Probably Approximately Correct, it is presently possible to make grammatical inference on it [21]. The grammatical inference has been significantly developed by *Colin de la Higuera* whose works are presented in these books [22,23]. Grammatical inference represents the process of learning a formal grammar of a set of data. Note that the trie is pre-parsed, the grammatical inference algorithms will be faster and more efficient.

Many methods of grammatical inference exist, often competing in the PAutomaC [24] contest, a competition of PAC-learning. Among the algorithms, ALERGIA and MDI are non-deterministic algorithms allowing the reduction of a probabilistic deterministic automaton.

6.1. ALERGIA

The ALERGIA algorithm is detailed in the following articles [25,26]. ALERGIA has been applied to information extraction from text or structured documents and speech-language modeling. ALERGIA follows the same process as the well-known RPNI algorithm: some sequences, a compatibility test, and a merging operation.

The compatibility test is based on Hoeffding bounds. In the trie, let n_i be the number of subsequences arriving at node i; f denotes the count of the transitions: $f_i(a)$ be the number of subsequences following arc with the value a; $f_i(.)$ be the number of sequences terminating at node i. Calculate the following probabilities: $p_i(a) = \frac{f_i(a)}{n_i}$ for each transition.

Two nodes i and j are compatible if for each transition (including termination) and their corresponding children are going to be checked recursively:

$$\left|p_i(a) - p_j(a)\right| < \sqrt{0.5 \ln \frac{2}{\epsilon}} \left(\frac{1}{\sqrt{n_i}} + \frac{1}{\sqrt{n_j}} \right)$$

where ϵ is called the acceptance range.

The ALERGIA algorithm checks all pairs of nodes, if no pairs are compatible then the algorithm stops. ALERGIA produces a deterministic probabilistic finite automaton. As a compromise between the automaton reduction and the Kullback-Leiber divergence [27], a value $\epsilon = \frac{1}{N^3}$ is taken with N the number of states.

6.2. Prediction and Updating

To be able to use efficiently the probabilistic automaton, we transform it into a Hidden Markov Model (HMM).

The process to transform a stochastic automaton into an HMM is described and proved by Dupont et al. [28] and Harbrard et al. [29]. The steps of the process are not discussed in our paper. This process has been developed in previous works and published in Demessance et al. [30]. The algorithm has been adapted to our context. Let us describe the structure of an HMM and how to perform a prediction.

An HMM is a graph where nodes and arcs possess probabilities. A set of couples $[item, probability]$ are referenced on each node where *item* is a cluster's means, an outlier, or a motif; and *probability* refers to its probability to be generated on this node. The item # refers to the end of a sequence. By convention, all items are represented in each node, even if its probability is equal to zero.

The root nodes of an HMM are characterized by an ongoing arc without a start node. The root nodes are nodes where we can start to read the sequence. They are computed through the ALERGIA algorithm. Last but not least, arcs possess a *probability* to go from a node to another node.

To perform a prediction, we introduce two operations on an HMM: *jump* and *observation*. The jump process is similar to a random walk in a Markov chain. Then, one can generate an *item* on the last node visited. The combination of those two processes is called observation. The probability to observe an item is equal to the product of the jump and item generation.

Since we need to predict an item from a given sequence, the first step is to start the observation from the right node. Indeed, we need to read the given sequence in the HMM before performing a new observation. This process is done thanks to the *Viterbi* algorithm [31]. Given an input sequence, this algorithm computes the most probable sequence of observations.

Finally, the prediction process occurs at the node of the last observation of this sequence. A set of all predicted observations are generated from this node. This means all jumps from the start node to another node and all generated items at these nodes are computed.

Each observation produces a couple of *(suffix, probability)* where $suffix$ refers to the generated item with its $probability$ to be observed. Some couple has the same $suffix$, thus its probability is summed.

To observe at a range of 2, we produce a new observation starting at the last visited node of each $suffix$. The new $suffix$ is the concatenation of the two observed items. And so on for a larger range of predictions.

In our context, we want to predict the next slope value (consumption or production) of a device knowing its past. Following the prediction process, the forecast sequence is equal to the most probable $suffix$.

Let us show the prediction process with an example (see Figure 6). The HMM has four nodes named 11, 12, 21, 22. The item and probability in brackets are items for each node. The weighted arcs between nodes represent the probability to jump. For this example, we start at node 11 and build the $suffixes$ at two steps forward.

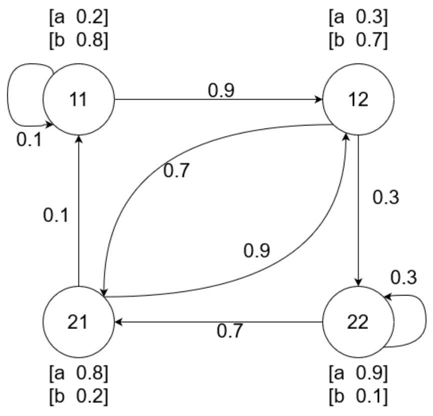

Figure 6. Example of Hidden Markov Model.

Concerning the prediction of the next item, there are two possibilities to jump: to 12 with a probability 0.9 and to 11 with a probability 0.1. To perform an observation, each of these probabilities are multiplied by the probability of each item at nodes. On node 11, we generate item a at 0.2 and b at 0.8. The probabilities of observations from node 11 to node 11 are: a with a probability $0.1 * 0.2 = 0.02$ and b with a probability $0.1 * 0.8 = 0.08$. For the observation from node 11 to node 12, we have: a with a probability $0.9 * 0.3 = 0.27$ and b with a probability $0.9 * 07 = 0.63$. Since both a and b are common $suffix$es, we add their probability to be observed: a with a probability $0.02 + 0.27 = 0.29$ and b with a probability $0.08 + 0.63 = 0.71$ (note the sum of probabilities of $suffix$es is always equal to 1).

To perform a prediction of two items, we need to compute the next observation considering we start at node 11 with a probability 0.1 and at node 12 with a probability 0.9. The predicted item is the $suffix$ with the highest probability.

In our model, the predicted item can be a single value or a motif. If we consider a timestamp of 5 min, i.e., 5 min between each value, an observation represents a forecast of at least 5 min. Indeed, predicting a motif extends the range of the forecast. In our model, we consider a forecast of at least 20 minutes, i.e., at most a prediction of 4 items.

6.3. Update

If the HMM provides not good enough results, or when we face new behaviors, we must update the HMM.

The update process adapts the probabilities to jump and the probabilities to generate items for a sequence set. To update the HMM, we use the *Baum-Welch algorithm* [31,32]. In our method, we implement this algorithm without any modification, so it isn't discussed in this paper.

7. Validation and Comparison

To validate our approach, we compare our results with famous sequence prediction machines and deep-learning methods.

7.1. Sequence Prediction Machines

The task of sequence prediction consists of predicting the following data of a sequence based on the previously observed data. Most of the sequence prediction machines are based on Markov properties as Prediction by Partial Matching (PPM) [33], Dependancy-Graph (DG) [34], All-K-Order-Markov (AKOM) [35], Transition Directed Acyclic Graph (TDAG) [36], Probabilistic Suffix Tree (PST) [37], Context Tree Weighting (CTW) [37].

The drawback of these algorithms is the increasing of temporal and spatial complexity when one wants to increase their accuracy. If the Markovian hypothesis doesn't hold, the accuracy severely decrease. Some models are based on compression such as LZ78 [38], Active Lezi [39] and more recently CPT [40] and CPT+ [41].

To validate our model, we implemented the following methods with the same sequence preprocessing: TDAG, CPT+, AKOM. Note that those sequence prediction machines do not possess an update method.

7.2. Deep-Learning Methods

To validate our method, we implement four deep-learning methods, trained and tested directly on the time series of the whole house after discretizing. The four methods are the most known concerning consumption forecasting as seen in the literature review: CNN-LSTM, CNN, LSTM, and MLP [42].

8. Pedagogical Instance

The data processing (sequencing, evolution of consumption) and the hierarchical clustering are done with the R Studio software. The sequence prediction machines are implemented with JAVA using the SPMF Library developed by Fournier-Viger et al. [43]. The deep-learning methods, ALERGIA algorithm, and HMM methods are implemented in Python language. A Python main script regroups all the processes.

Our method takes one minute to run from the data gathering to the prediction. Tests are executed on a personal computer running on Windows 10 with a processor Intel Core i5-6400 CPU at 2.70 Ghz with 8Go DDR RAM, with 250k-1M observations per appliance. The method requires a similar time with any dataset since it's sampled.

In pedagogical terms, we consider the dishwasher from Home C, Meter 1.

8.1. Data Analysis

First of all, one must analyze the general trends of the time series shown in Figure 7. The consumption values present 503,910 observations from 0.0001 Wh to 1.4018 Wh. Over 98% of the values are under 0.5 Wh and represent the idle mode of the dishwasher. In consequence, a threshold of 0.5 Wh is used as "zero". Close to 90% of the total consumption is kept.

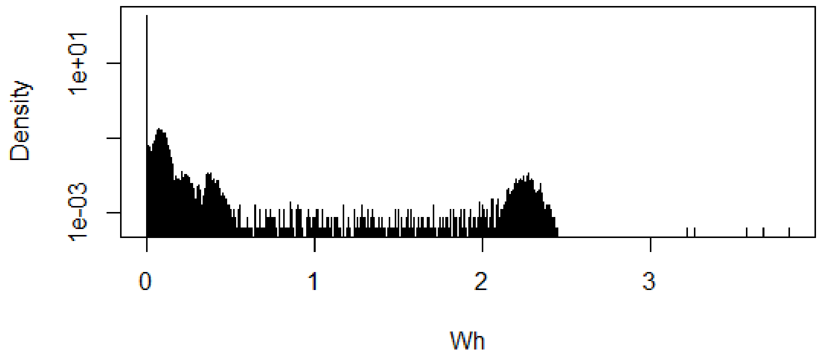

Figure 7. Density of the data.

After sequencing the time series, the dataset provides 748 sequences with an average of 10.6 values of slopes per sequence for a total of 10,834 values where 92.7% are singletons. The slope's percentiles are shown in Figure 8. A representative sample of 1000 slope's values is taken from the dataset and is used to reduce the noise through clustering.

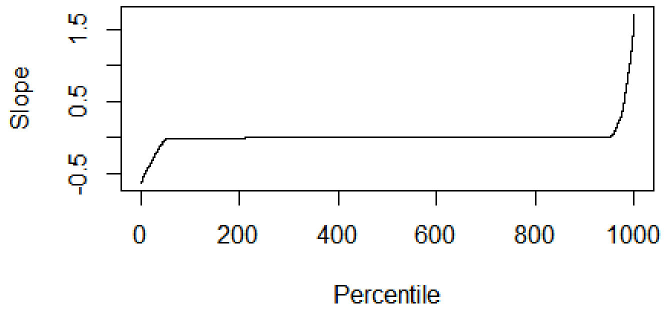

Figure 8. Percentile of the slope.

8.2. Method Step by Step Analysis

The first step of clustering is to determine the optimal number of clusters employing the Elbow method to the sample. The Elbow method executed on the sample gives an optimal number of clusters of 4 clusters and an estimated WSS at 35.8 by extrapolating to the whole dataset.

The Figure 9 presents the inertia of AGNES-flexible for 1 to 20 clusters. The inertia tells how far away the points within a cluster are. As seen with the inertia of the hierarchical clustering, 4 clusters are a good choice. Indeed, the inertia jumps from a maximal value of 1.8 to 3.8. With 5 clusters, the inertia will be at 1.6 which has no change compared to 4 clusters.

Subsequently, the method determines the best hierarchical clustering among DIANA and AGNES algorithms with several metrics. Table 2 presents the average Silhouette, total WSS, and WSS for each cluster for all the hierarchal clustering algorithms. The suffix of AGNES denotes the method: Complete, Ward, Flexible. In this example, the highest average silhouette is given by AGNES with the flexible method.

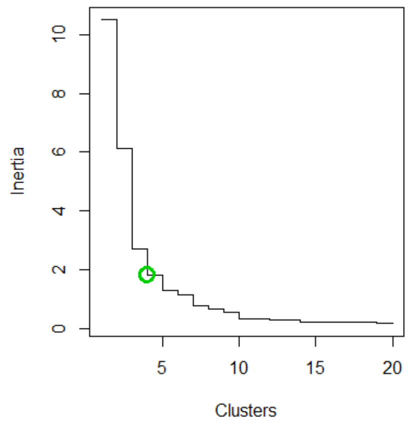

Figure 9. Inertia of AGNES-flexible on the sample.

Table 2. Comparison of WSS for each hierarchical clustering.

Algo	Silh	TWSS	WSS
DIANA	0.937	12.8	2.85 2.36 3.46 4.16
AGNES-c	0.94	12.4	1.90 3.01 3.34 4.16
AGNES-w	0.945	12.1	1.64 2.29 3.34 4.83
AGNES-f	0.945	12.1	1.64 2.29 3.34 4.83

The Figure 10 shows the Silhouette after merging the dataset. The average Silhouette before merging is equal to 0.95. The difference between the Silhouette before and after merging is −0.01. It may vary depending on the sample. The more the dataset contains a vast diversity of values, the more the Silhouette will decrease after merging the dataset. Indeed, the sample is representative of the diversity, not of the distribution.

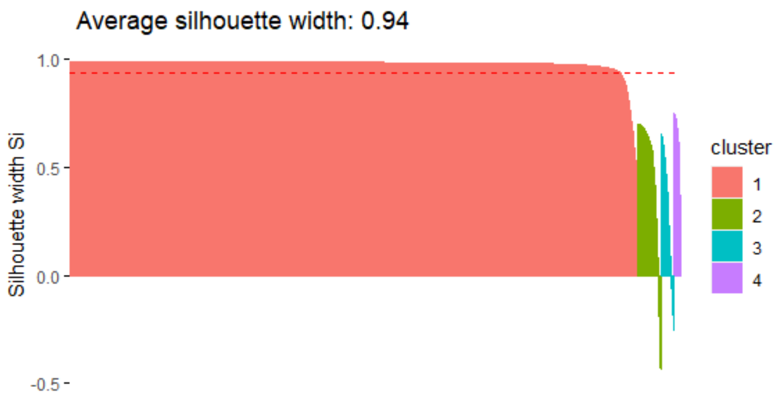

Figure 10. Silhouette after merging the dataset.

Finally, the K-means algorithm adjusts the data in the right cluster to maximize the silhouette. The Figure 11 presents the Silhouette of the dataset after implementing the K-means algorithm. The average Silhouette is close to the sample. Among the dataset, 586 different data (5.5% of the dataset) have changed their cluster.

Table 3 compares the WSS at each step of our method: after the "zero" threshold (*Zero*), after the hierarchical clustering (*Hcl*), after the K-means (*End*). The total WSS (TWSS) slightly decreases between the hierarchic clustering results and the K-means algorithm results. The WSS is similar for each cluster, and the mean is more adequate to the range of values.

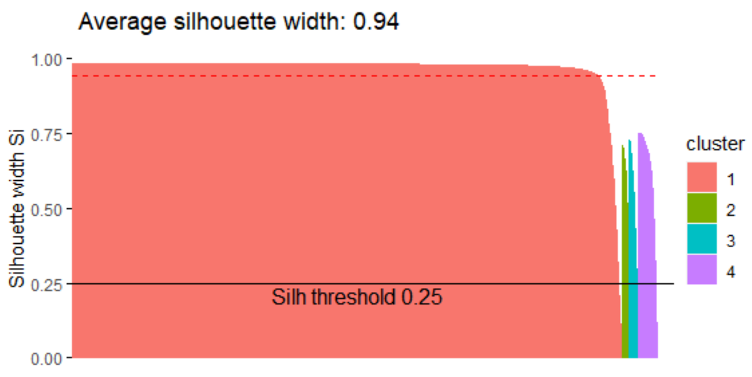

Figure 11. Silhouette of the dataset after K-means algorithm with 4 clusters.

Table 3. Step by step comparison of the proposed method.

	Zero	**Hcl**	**End**
TWSS	35.8	37.7	35.4
WSS	none	4.02 9.45 14.11 10.11	8.42 8.92 8.64 9.45
Elements	10,834	10,075 405 232 122	10,179 348 173 134
Mean	0.012	0.001 −0.358 0.057 1.256	0.001 −0.39 0.551 1.225
Min/max	−0.631/1.711	−0.19/0.22 −0.63/−0.13 0.22/0.92 0.93/1.71	−0.19/0.27 −0.63/−0.20 0.28/0.89 0.89/1.71

At this point, we retrieve the extreme points with a *Silh* threshold at 0.25 as shown in Figure 11. The number of outliers is 11 (0.10% of the dataset), each outlier has its cluster. Thus, in the beginning, we got 10,834 different values. They are transformed into 15 values, 4 from the means of the clusters, and 11 outliers. We reduce the number of different values by 99.89%.

Concerning the motif discovery, we start with 748 sequences with an average of 10.6 values. We discover several motifs, from length 16 to length 2. The average number of values drops to 3.4 values per sequence, reducing the length by 67.9%.

Finally, we transform the dataset into a stochastic automaton thanks to ALERGIA algorithm. We do not provide the graph of the HMM since it's understandable for a human's eye (18 nodes, an average of 3 arcs per node, 15 items per node).

We perform some metrics to compare the true consumption to the predicted consumption. First of all, we transform the predicted sequence into a curve that constitutes the predicted consumption. The HMM prediction has an accuracy (the probability to predict the right value) of 99.8%. The Mean Square Error (MSE) is 0.0183 Wh and the Mean Absolute Error (MAE) is 0.0956 Wh which is low compared to the data values: the minimal consumption is 0.5016 Wk; the maximal consumption is 1.4018Wh; the mean is 1.3047 Wh. The Mean Absolute Percentage Error (MAPE) is 7.4%. The MAPE is relevant since it quickly increases with small values.

8.3. Impact of Zero Threshold

The accuracy with a value's threshold at 0.5 Wh is 99.8 with a MAPE at 7.4%. To determine how the Zero value impacts the results, we perform the same test with a Zero threshold at 0.1 Wh. Our model's accuracy drops to 78.8 with a MAPE at 12.8%. As shown, the threshold must be representative of a device in operation to provide relevant results.

8.4. Impact of the Number of Clusters

The Table 4 presents the accuracy (Acc.), average Silhouette (Sil.), MSE (in 10^{-3} Wh), MAE (in 10^{-3} Wh) and MAPE (in %) in the function of the number of clusters, we don't count outliers as clusters. Note that MSE, MAE, and MAPE are computed considering only true predictions. The average Silhouette drops after 10 clusters because their center becomes too close to each other. The MSE, MAE, and MAPE decrease with the number of clusters, but the accuracy becomes quickly low.

Table 4. Prediction efficiency in function of the number of clusters.

# Cluster	Acc.	Sil.	MSE	MAE	MAPE
4	99.8	0.94	18	96	7.4
5	92.6	0.93	17	96	7.3
6	85.4	0.94	8	64	4.9
7	73.5	0.94	5	50	3.9
10	40.8	0.92	3	42	3.2
15	30.1	0.78	1	31	2.4
20	18.7	0.76	1>	21	1.16

9. Overall Results

This subsection presents the results for various households.

Table 5 presents the results for a sample of 10 appliances taken randomly. The table shows in order the number of optimal clusters, the accuracy, the average Silhouette, and the MAPE in percent. Some devices are more suitable for our method than others. The appliances with the highest accuracy are the most used and those with distinct cycles of consumption (dishwasher as an example). The appliances with the lowest accuracy are the less used and those which depend on external parameters or human interaction (ovens for example).

Table 5. Results for a sample of 10 appliances.

Test	# Cluster	Acc	MAPE %
1	4	99.8	7.4
2	5	96.1	4.7
3	5	98.3	1.8
4	7	99.9	6
5	9	95.5	1.5
6	3	99.9	5.2
7	5	98.9	0.9
8	4	98.1	1.3
9	4	99.2	4.2
10	8	96.8	1.1

9.1. Comparison with Deep-Learning Methods

To validate our method, we implement four deep-learning methods, trained and tested on the whole Home C discretized time series. The four methods are CNN-LSTM, CNN, LSTM, and MLP. Their architecture is shown in Table 6. The window size is optimized to keep the lowest error. All models are trained to give a 10 min forecast, similar to our method with a range of 2.

Table 6. Models parameters & architecture.

Models	CNN_LSTM	CNN	LSTM	MLP
Architecture	1x Conv1D 2x LSTM 2x Dense Output	1x Conv1D 1x Flatten 2x Dense Output	2x LSTM 1x Dense Output	1x Flatten 2x Dense Output
Epochs	17	25	30	50
Optimizer	Adam	Adam	Adam lr = 0.003	Adam

The MAPE results are: 40% for the CNN-LSTM; 47% for CNN; 54% for LSTM and 48% for MLP. As said in the introduction, deep-learning methods fail to forecast isolated areas due to the low values and the huge impact of human behavior on the system.

9.2. Comparison with Sequence Prediction Machines

We compare our results with sequence prediction machines: TDAG at order 4, CPT+, AKOM at order 4. They show similar results as our method with an equal or less MAPE with no more than 10 points less due to lower accuracy than our method. Note that the coverage is at 100%, i.e., all the sequences are used to train the model.

In this case, all prediction machines provide outstanding accuracy due to the quality of the sequence's processing we propose. Note those prediction machines can only predict already known sequences, and cannot predict when there are at the end of a sequence. Videlicet, those prediction machines are useless if they encounter a new behavior and for a long forecast.

9.3. Algorithms and Best Results

We calculate the number of times an algorithm in the method provides the best result. Indeed, depending on the distribution function of the values, the clustering methods (Dividing, agglomerative, Ward, Complete linkage, Flexible linkage) provide different dendrograms. Thus, the distribution and clusters change, which implies a variation in WSS and Silhouette and the remaining methods. About the prediction machines, they implement various algorithms to reduce the decision tree. Depending on the trie in input, they may provide another prediction from each other.

We compute the percentage of presence of each clustering method and prediction machines to obtain the best prediction based on 232 tests in the Table 7. We don't include deep-learning methods since they reveal their limits for forecasting isolated areas.

When the values follow a Gaussian distribution, AGNES-ward and AGNES-flexible are the most used; else DIANA or AGNES-complete give a better clustering. Concerning the prediction, our method based on HMM gives 91% of time the best accuracy. Sequence machines have most of the time a lesser or equal accuracy than the HMM due to the outliers. Those results validate our approach.

Table 7. Used of each algorithms for an optimal solution.

Algo	% Use	Algo	% Use
DIANA	9	TDAG	>1
AGNES-c	11	CPT+	1
AGNES-w	41	AKOM	8
AGNES-f	39	HMM	91

9.4. Forecasting of an Isolated Area

Another application of our method is the consumption schedule. One identifies the sequences associated with the use of their appliances or the recurrent consumption cycles. Next, one manipulates them for a job scheduling with knapsack constraints [44].

Since the time series is decomposed into sequences, one can analyze the use of devices to compute a frequency or recurrence of sequences along days. More information is added to each sequence: the time when the sequence starts with its frequency as a stochastic process. Thanks to this, we can schedule the sequences of each device having a recurrent behavior over the day. The Figure 12 presents the common patterns of consumption of appliances in Home F during 500 min where each color refers to one appliance. The user can add more appliances consumption cycle according to its need or according to the most probable use time.

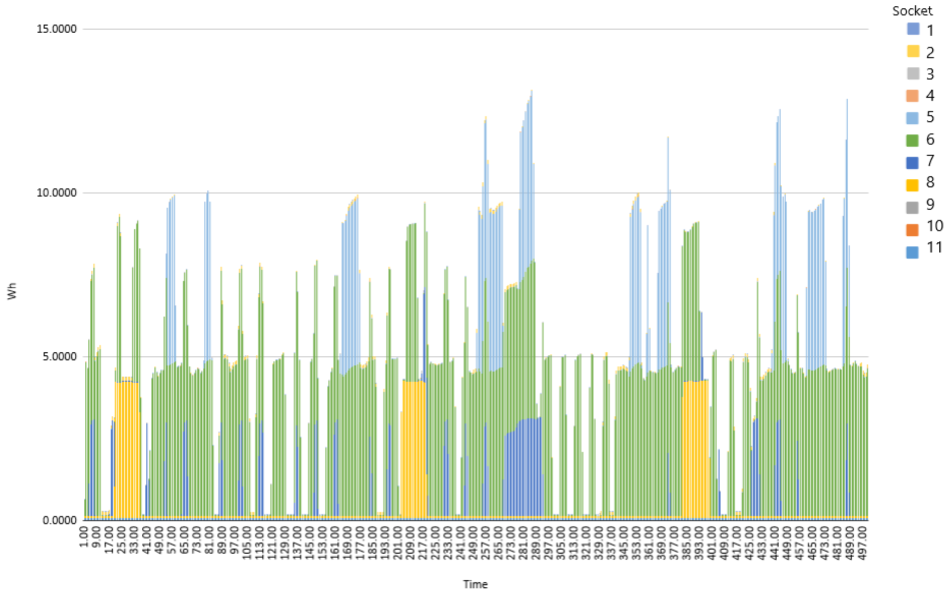

Figure 12. Manual consumption scheduling of recurrent consumption cycle of 11 appliances.

9.5. Discussion

Our method works for any time series, without time, season, weather contexts. Those external factors influence human behavior. Since we determine sequences of consumption reflecting those behaviors, our method is relevant in any context.

The drawbacks are the lack of flexibility of a prediction machine when it encounters unknown schemes of consumption and fails to forecast the good motif. The ability to adapt the HMM thanks to *Baum-Welch*'s algorithm makes our model more flexible to new behaviors.

Note that our model performs better than the most used methods in the literature for isolated areas. This statement becomes false when we need to forecast for a large area as a region. In this case, deep-learning methods with external factors are the most effective. Indeed, our method needs a huge amount of knowledge for each place and cannot compete with some deep-learning methods. Those run with the time series and the weather and provide results with less than 1% of MAPE over a region.

10. Conclusions

Consumption's prediction of isolated households represents a complex problem. Some deep learning methods or mathematical models generate substantial results but are unsuitable to isolated small prosumers.

We adopt a method principally employed to predict DNA to the energy forecasting problem. Our method works efficiently on any device (household appliances, industrial appliances, industrial motors). Moreover, the method is not a black or grey box and it can be embedded easily in any smart building by anyone.

The result of the sequence processing is used to build an HMM. The latter can provide a short-term forecast for the device. The sum of the forecast forms the forecast of the area. The sequences are also used by the owner to schedule its consumption and to be able to adapt its scheduling according to the demand-side management program.

Our method will be used in Mayotte Island, through the H2020 MAESHA's project. It will be included in a multi-agent system to simulate the behavior of the whole island. This sandbox is useful to test new strategies, renewable plants, and flexible markets on the island before considering real-world changes.

Author Contributions: Conceptualization, G.G. and H.P.; methodology, G.G.; software, H.P. and I.T.; validation, G.G., H.P. and I.T.; resources, I.T.; writing—original draft preparation, G.G. and H.P.; writing—review and editing, G.G.; visualization, G.G.; supervision, G.G.; project administration, G.G. All authors have read and agreed to the published version of the manuscript.

Funding: This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 957843.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable

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

References

1. Amin, M. The smart-grid solution. *Nature* **2013**, *499*, 145–147. [CrossRef]
2. Hong, T.; Fan, S. Probabilistic electric load forecasting: A tutorial review. *Int. J. Forecast.* **2016**, *32*, 914–938. [CrossRef]
3. Zhao, H.X.; Magoulès, F. A review on the prediction of building energy consumption. *Renew. Sustain. Energy Rev.* **2012**, *16*, 3586–3592. [CrossRef]
4. Daut, M.A.M.; Hassan, M.Y.; Abdullah, H.; Rahman, H.A.; Abdullah, M.P.; Hussin, F. Building electrical energy consumption forecasting analysis using conventional and artificial intelligence methods: A review. *Renew. Sustain. Energy Rev.* **2017**, *70*, 1108–1118. [CrossRef]
5. Wang, Z.; Srinivasan, R.S. A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models. *Renew. Sustain. Energy Rev.* **2017**, *75*, 796–808. [CrossRef]

6. Raza, M.Q.; Khosravi, A. A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings. *Renew. Sustain. Energy Rev.* **2015**, *50*, 1352–1372. [CrossRef]
7. Amasyali, K.; El-Gohary, N.M. A review of data-driven building energy consumption prediction studies. *Renew. Sustain. Energy Rev.* **2018**, *81*, 1192–1205. [CrossRef]
8. Pérez-Lombard, L.; Ortiz, J.; Pout, C. A review on buildings energy consumption information. *Energy Build.* **2008**, *40*, 394–398. [CrossRef]
9. Liu, T.; Tan, Z.; Xu, C.; Chen, H.; Li, Z. Study on deep reinforcement learning techniques for building energy consumption forecasting. *Energy Build.* **2020**, *208*, 109675. [CrossRef]
10. Deb, C.; Zhang, F.; Yang, J.; Lee, S.E.; Shah, K.W. A review on time series forecasting techniques for building energy consumption. *Renew. Sustain. Energy Rev.* **2017**, *74*, 902–924. [CrossRef]
11. Zhang, G. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing* **2003**, *50*, 159–175. [CrossRef]
12. Nie, H.; Liu, G.; Liu, X.; Wang, Y. Hybrid of ARIMA and SVMs for Short-Term Load Forecasting. *Energy Procedia* **2012**, *16*, 1455–1460. [CrossRef]
13. Wang, X.; Meng, M. A Hybrid Neural Network and ARIMA Model for Energy Consumption Forcasting. *J. Comput.* **2012**, *7*. [CrossRef]
14. Platon, R.; Dehkordi, V.R.; Martel, J. Hourly prediction of a building's electricity consumption using case-based reasoning, artificial neural networks and principal component analysis. *Energy Build.* **2015**, *92*, 10–18. [CrossRef]
15. Karevan, Z.; Suykens, J.A. Transductive LSTM for time-series prediction: An application to weather forecasting. *Neural Netw.* **2020**, *125*, 1–9. [CrossRef]
16. Wang, J.Q.; Du, Y.; Wang, J. LSTM based long-term energy consumption prediction with periodicity. *Energy* **2020**, *197*, 117197. [CrossRef]
17. Kim, T.Y.; Cho, S.B. Predicting residential energy consumption using CNN-LSTM neural networks. *Energy* **2019**, *182*, 72–81. [CrossRef]
18. Jaiwei, H.; Kamber, M. *Data Mining: Concepts and Techniques*; Morgan Kaufmann: San Francisco, CA, USA, 2006.
19. Rousseeuw, P.J.; Kaufman, L. *Finding Groups in Data*; Wiley Online Library: Hoboken, NJ, USA, 1990.
20. Hirate, Y.; Yamana, H. Generalized Sequential Pattern Mining with Item Intervals. *JCP* **2006**, *1*, 51–60. [CrossRef]
21. Haussler, D. *Probably Approximately Correct Learning*; University of California, Santa Cruz, Computer Research Laboratory: Santa Cruz, CA, USA, 1990.
22. De la Higuera, C. *Grammatical Inference: Learning Automata and Grammars*; Cambridge University Press: Cambridge, UK, 2010.
23. Eyraud, R.; De La Higuera, C.; Kanazawa, M.; Yoshinaka, R. Introduction to the Grammatical Inference Special Issue of Fundamenta Informaticae. 2016. Available online: https://hal.archives-ouvertes.fr/hal-01399434/document (accessed on 20 July 2021).
24. Verwer, S.; Eyraud, R.; De La Higuera, C. PAutomaC: A probabilistic automata and hidden Markov models learning competition. *Mach. Learn.* **2014**, *96*, 129–154. [CrossRef]
25. Carrasco, R.C.; Oncina, J. Learning stochastic regular grammars by means of a state merging method. In *International Colloquium on Grammatical Inference*; Springer: Berlin/Heidelberg, Germany, 1994; pp. 139–152.
26. Thollard, F.; Dupont, P.; De La Higuera, C. Probabilistic DFA inference using Kullback-Leibler divergence and minimality. In Proceedings of the Seventeenth International Conference on Machine Learning (ICML), Standord, CA, USA, 29 June–2 July 2000; pp. 975–982.
27. Kullback, S. *Information Theory and Statistics*; Courier Corporation: North Chelmsford, MA, USA, 1997.
28. Dupont, P.; Denis, F.; Esposito, Y. Links between probabilistic automata and hidden Markov models: Probability distributions, learning models and induction algorithms. *Pattern Recognit.* **2005**, *38*, 1349–1371. [CrossRef]
29. Habrard, A.; Denis, F.; Esposito, Y. Using pseudo-stochastic rational languages in probabilistic grammatical inference. In *International Colloquium on Grammatical Inference*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 112–124.
30. Demessance, T.; Bi, C.; Djebali, S.; Guérard, G. Hidden Markov Model to Predict Tourists Visited Places. In Proceedings of the 2021 22nd IEEE International Conference on Mobile Data Management (MDM), Toronto, ON, Canada, 15–18 June 2021; pp. 209–216.
31. Kriouile, A.; Mari, J.F.; Haon, J.P. Some improvements in speech recognition algorithms based on HMM. In Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Albuquerque, NM, USA, 3–6 April 1990; pp. 545–548.
32. Baggenstoss, P.M. A modified Baum-Welch algorithm for hidden Markov models with multiple observation spaces. *IEEE Trans. Speech Audio Process.* **2001**, *9*, 411–416. [CrossRef]
33. Cleary, J.; Witten, I. Data compression using adaptive coding and partial string matching. *IEEE Trans. Commun.* **1984**, *32*, 396–402. [CrossRef]
34. Padmanabhan, V.; Mogul, J. Using prefetching to improve world wide web latency. *Comput. Commun.* **1998**, *16*, 358–368. [CrossRef]
35. Pitkow, J.; Pirolli, P. Mininglongestrepeatin g subsequencestopredict worldwidewebsurfing. In Proceedings of the UsENIX Symposium on Internet Technologies and Systems, Boulder, CO, Canada, 11–14 October 1999; p. 1.
36. Laird, P.; Saul, R. Discrete sequence prediction and its applications. *Mach. Learn.* **1994**, *15*, 43–68. [CrossRef]

37. Begleiter, R.; El-Yaniv, R.; Yona, G. On prediction using variable order Markov models. *J. Artif. Intell. Res.* **2004**, *22*, 385–421. [CrossRef]
38. Ziv, J.; Lempel, A. Compression of individual sequences via variable-rate coding. *IEEE Trans. Inf. Theory* **1978**, *24*, 530–536. [CrossRef]
39. Gopalratnam, K.; Cook, D.J. Online sequential prediction via incremental parsing: The active lezi algorithm. *IEEE Intell. Syst.* **2007**, *22*, 52–58. [CrossRef]
40. Gueniche, T.; Fournier-Viger, P.; Tseng, V.S. Compact prediction tree: A lossless model for accurate sequence prediction. In Proceedings of the International Conference on Advanced Data Mining and Applications, Hangzhou, China, 14–16 December 2013; pp. 177–188.
41. Gueniche, T.; Fournier-Viger, P.; Raman, R.; Tseng, V.S. CPT+: Decreasing the time/space complexity of the Compact Prediction Tree. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Ho Chi Minh City, Vietnam, 19–22 May 2015; pp. 625–636.
42. Lim, B.; Zohren, S. Time-series forecasting with deep learning: A survey. *Philos. Trans. R. Soc. A* **2021**, *379*, 20200209. [CrossRef]
43. Fournier-Viger, P.; Gomariz, A.; Gueniche, T.; Soltani, A.; Wu, C.W.; Tseng, V.S. SPMF: A Java Open-Source Pattern Mining Library. *J. Mach. Learn. Res.* **2014**, *15*, 3389–3393.
44. Charikar, M.; Khuller, S. A robust maximum completion time measure for scheduling. In Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm, Miami, FL, USA, 22–26 January 2006; pp. 324–333.