A Visual Two-Stage Classification of Electricity Customers Using Growing Hierarchical self-Organizing Maps.

Taha Abdelhalim Nakabi, PhD student, University of Eastern Finland (e-mail: tahanak@uef.fi)
Pekka Toivanen, Professor at University of Eastern Finland (e-mail: pekka.toivanen@uef.fi)

***Abstract*— An efficient management of the new generations of power systems requires a new set of decision making tools that can give insight about the users of the grid. A large number of customers behaving differently and having various electricity demand are using the power system. Therefore, a successful demand-side management program should consider all these variations. Customer classification is an important step to understand the grid, forecast the electricity demand, and implement a good demand-response program that can offer customized prices and incentives. In this paper, we propose a new approach for a visual customers classification, using growing hierarchical self-organizing maps (GHSOM). We present the method and its features and explain its implementation. We test the algorithm against real data including socio-economic information of customers as well as their historic half-hour electricity load meters. We first present basic classifications using the full time series of customers during 208 days and then improve the classification by extracting a daily average load for each customer. This classification gave a good visual separation of loads based on the Euclidian distances between time series of the daily electricity consumption meters. We finally introduce a second stage classification using support vector machine (SVM) to reclassify the load curves according to their shapes using load shape indices. This reclassification grouped more samples in the clusters and transferred some of the samples to other clusters. A comparison between the first and second classification showed that the reclassification could indeed improve the quality of classification. Results of both first and second classifications are visualized using load curves and their distribution on the map.**

***Index Terms*—Customer classification, Customer behavior learning, Demand-response programs, Demand side management, Growing hierarchical self-organizing maps, Smart grid, Support vector machine.**

I. Introduction

Customer classification is an extremely important task for many companies and organizations in different industries. The ability of dividing the customers into groups that have similar characteristics can help understanding their behavior and predicting their demand. It is also useful for any marketing strategy to know what group of customers are more likely to accept certain offers and how they would respond to certain range of prices. The same principle applies to electric power industry and energy distribution. With the large development of smart grids and smart measuring technologies [1], more opportunities are emerging for the utility companies to take advantage of this data in order to develop efficient demand-side management (DSM) strategies and demand-response (DR) programs. The classification of the network's households is a key tool for DR and DSM programs. It gives the decision makers a meaningful insight about the patterns of energy consumption and types of demand. It can also help understanding the customers' behaviors vis-à-vis the electricity consumption, by grouping them into classes with the same load profiles or the same socio-economic characteristics.

The literature contains a variety of methods to classify customers based on their load profiles. In [2] many clustering techniques for electricity customers have been reviewed. According to the review, clustering techniques can be divided into two categories: direct clustering and indirect clustering. Direct clustering methods use data directly collected from smart meters and are based on classification algorithms such as k-means, fuzzy k-means, hieratical clustering, and Self-Organizing Maps (SOM). While indirect clustering consists of applying some dimensionality reduction or time series analysis on the data before clustering the loads. In this paper, we focus on direct clustering of customers. Several papers were published in this context discussing various classification methods to detect electricity usage patterns based on load curves. Such as K-means [3], [4] fuzzy c-means (FCM) [5], [6], hierarchical methods [7], [8], self-organizing maps (SOM) [9] support vector machine (SVM) [10], [11], subspace projection method [12]. Other works have used classification methods based on the socio-economic data rather than historical load curves, such as [13] where a SOM classification and a neural network were combined to generate load profiles based on a customer classification. In [14] a finite mixture model (FMM) was used for the analysis and clustering of residential customers' energy behavioral demand using half-hourly separated smart meters. In [15] a second stage of classification was presented to reclassify the customers taking into consideration their load curves' shapes. The above mentioned works presented valuable methods of customer classification using the benchmark static classification methods that usually require an initial definition of the number of clusters or the size of the model. The main contribution of this paper is the use of

This work was supported by The Jenny and Antti Wihuri Foundation, FINLAND.

dynamic and visual clustering technique. We use growing hierarchical self-organizing maps (GHSOM) algorithm for electricity customers' classification. To the authors knowledge this is the first application of GHSOM algorithm in customer classification in the context of smart grids. The advantages of using GHSOM algorithm are its capabilities to classify complex sets of patterns in an unsupervised way. It is an improved version of SOM where the network adapts dynamically its size and architecture to the data. Therefore, the number of nodes is no longer needed to be specified at the beginning. The algorithm starts from a minimum map size and makes it grow as necessary. It can be successfully used in extracting classes and sub-classes using its hierarchical growing mechanism. In [16], a comparison between SOM and GHSOM showed that GHSOM gives better results for large size unstructured data. It is also extremely efficient in representing data points graphically due to its spatial and hierarchical clustering mechanisms.

In this paper, we perform three separate classifications; the first one is using a dataset of socio-economic information about customers. The dataset contains 1013 customers and their information collected through a survey. This classification is mainly used to test the GHSOM algorithm and its parameters. We run the algorithm several times with different parameters and analyze the variations in map's architecture. The second classification uses the full time series of half-separated electricity meters for the 208 days. The dataset contained 41 customers with 10 000 meters. In this classification, we consider that every point in the time series is important and the consumption patterns were compared point by point. In the third classification we extracted an average daily load from the 208 days for each customer. This is more meaningful considering that customers have daily consumption patterns and that the average daily load is enough to represent a customer. This classification is also more practical for load profiles visualization as we used the load curves to show the similarities in clusters. Finally, we propose a second stage classification of load profiles based on their shapes using SVM algorithm.

The rest of this paper is organized as follows: Section III presents a theoretical framework for GHSOMs and their learning process. The case study, data description and numerical results are presented and discussed in section IV. Section V is a conclusion.

II. Unsupervised classification using GHSOM algorithm

In this section, we describe GHSOM algorithm and its learning process. The main idea of SOMs is to have a spatial representation of a multidimensional dataset, allowing each group of data points to be positioned in a node, unit or neuron in the map. The concept of maps is a description of all the nodes or units that contain input data points. Each unit in the map, is characterized by a vector of weights that has the same dimension as the features of data points. These maps and their corresponding nodes are built and organized according to the input data points through an unsupervised learning process, hence the name self-organizing maps. In growing hierarchical SOM, the shape of these maps is also dynamically determined according to the dataset; the learning process searches dynamically for the optimal size of the maps for a given data. It starts by a small size the keeps growing until a convergence criteria is met. Once the first map is complete, if a group of data points in a unit still needs further separation, a new SOM is emerged from that unit and the corresponding data points are spread over the new map that has also a dynamic shape. This process is repeated for all the units in all the maps and sub-maps.

The learning process starts by computing the *mean quantization error (mqe0)* of all inputs. Mqe0 can be seen as the deviation of all the inputs from a unit that represents the mean of all inputs. It can also be considered as a map with only one unit having as weight vector the average values of each attribute in the inputs data points. In all the following equations, we note the mean quantization error as *m*. Let I be the set of data points that we want to classify, each data point is a vector $x_i \in \Re^n$, and $i \in \{1, \dots, N\}$ where N is the number of data points and n the number of attributes. The weight vector of the initial unit is:

$$W_0 = [\mu_1, \mu_2, \dots, \mu_n]^T \quad W_0 \in \Re^n \qquad (1)$$

Using the same notations, the initial mean quantization error m_0 is:

$$m_0 = \frac{1}{N}\sum_{k=1}^{N} \|\mathrm{x_i} - \mathrm{W_0}\| \qquad (2)$$

This measure will be used in the next levels of the learning process. Next step is to create a new map with a small size having I_1 neurons. For each neuron *j* in the map, we associate a randomly initialized weight vector W_j,:

$$W_j = [w_{j1}, w_{j2}, \dots, w_{jn}]^T \quad W_j \in \Re^n \qquad (3)$$

Weights will be adapted iteratively during the learning process. Learning of an SOM is based on competition between neurons for the best approximation of a given input vector. The neuron with weight vector nearest to the input vector is the winner. Consequently, this input point will be associated to it and its weight vector as well as vectors of the neighboring neurons will be adapted in a way that decreases their distances to the input vector. The magnitude of this adaptation is decreasing with each iteration t and controlled by a learning parameter $\alpha(t)$ with: $0 < \alpha(t) < 1$, which is decreasing with time. The neighboring neurons are adapted in less magnitude than the winner, this magnitude is controlled by a neighborhood function $h_{cj}(t)$ based on the distance between the winner c and current neuron j, with $h_{cc}(t) = 1 \ \forall t \in N$ and it decreases with each iteration so that at the beginning of the learning process many neighboring neurons are adapted but at the end, almost only the winner neuron is adapted. Using the same notation as above, a proper form of the neighboring function according to [17] might be:

$$h_{cj}(t) = h_0(\mathrm{t}) \exp\left(-\frac{\left|\left|W_j - W_c\right|\right|^2}{\sigma(t)^2}\right) \qquad (4)$$

Where $h_0(\mathrm{t})$ and $\sigma(t)$ are suitable decreasing functions of time.

As a combination of these principles, we have the following learning rule for computing the weight vector $W_j(t+1)$ using $W_j(t)$:

$$W_j(t+1) = W_j(t) + h_{cj}(t)\left[x_i - W_j(t)\right] \quad (5)$$

Where x_i is the input data point corresponding to the t^{th} iteration. At the end of the learning process, all input points will be distributed on the neurons of the map. We then calculate the mean quantization error m_j for each neuron j using its corresponding input points as:

$$m_j = \frac{1}{d_j}\sum_{i\in j} ||x_i - W_j|| \quad (6)$$

Where d_j is the number of input data corresponding to neuron j. Finally, a mean quantization error for the map is computed as the average of all m_j:

$$M = \frac{1}{I_1}\sum_{j} m_j \quad (7)$$

In GHSOM, this measure is used to decide if the map should grow further or not. In each level, the deviation of input data should be better than the previous layer to a certain extent. A map should keep growing until the deviation measured by M is reduced to at least a fixed percentage τ_m compared to the map in previous level. The smaller the parameter τ_m is chosen, the larger will be the size of SOMs. As long as: $M \geq \tau_m.m_0$, a new row or column is added into the map near to the error neuron (neuron with the largest m_j). Insertion of a row or column depends on the most distant neighbor neuron. Distance is computed in input space (difference between weight vectors) and neighboring neurons are determined on the map. Weights of new neurons are usually initialized at average values of the neighboring neurons. The learning process starts over with the new map architecture. This process is repeated until:

$$M < \tau_m.m_0 \quad (8)$$

The GHSOM grows also vertically by evaluating the deviation in each neuron separately. Neurons, which still have a considerable m_j comparing to m_0 will be expanded by creating a new map in a new hierarchical level. This process is applied for each neuron j having:

$$m_j \geq \tau_u.m_0 \quad (9)$$

With τ_u is a fixed parameter representing the depth of the GHSOM. Neurons that have a m small enough such that:

$$m_j < \tau_u.m_0 \quad (10)$$

will not be developed further. The smaller the parameter τ_u is chosen, the bigger hierarchy of SOMs we will have. For every new map, the same learning process is repeated recurrently, using only the inputs corresponding to each parent neuron. The convergence of GHSOM is reached when both conditions (8) and (10) are satisfied for all maps and all neurons or when a maximal depth or maps size is reached. Fig.1 is an illustration of all the above-mentioned steps of GHSOM. The final model architecture of GHSOM is influenced mostly by the two parameters τ_m and τ_u. The parameter τ_m controls the level of details in data representations, small values of this parameter lead to larger maps while parameter τ_u is responsible for the quality of data representation, small values of this parameter give deep hierarchy of the model.

Additional parameters of GHSOM can also influence the architecture indirectly by influencing the learning rate, such as h_0(t), $\sigma(t)$ and $\alpha(t)$ but those parameters are usually fixed with default values. The maximal depth and size are also useful but mot mandatory parameters that can limit the GHSOM growth. Usually too deep hierarchy is difficult to understand and too large maps can give useless details in classification.

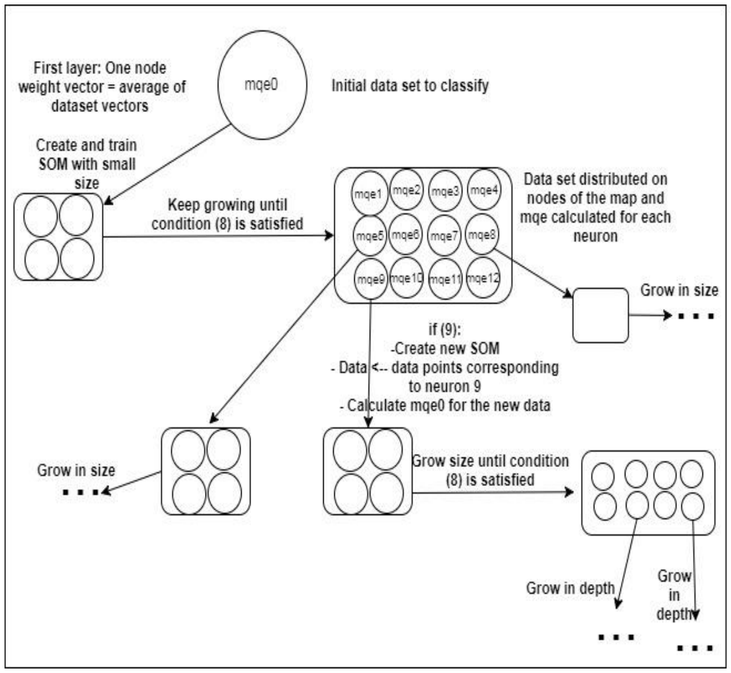

First layer: One node
weight vector = average of dataset vectors
mqe0
Initial data set to classify
Create and train SOM with small size
Keep growing until condition (8) is satisfied
mqe1 mqe2 mqe3 mqe4 mqe5 mqe6 mqe7 mqe8 mqe9 mqe10 mqe11 mqe12
Data set distributed on nodes of the map and mqe calculated for each neuron
Grow in size
if (9):
-Create new SOM
- Data <-- data points corresponding to neuron 9
- Calculate mqe0 for the new data
Grow in size
Grow size until condition (8) is satisfied
Grow in depth
Grow in depth

Fig.1: GHSOM learning process

One problem that can occur is that for some neurons when we emerge a new map, all data points are gathered in one neuron again or only one neuron will have more than one input. This will result in several maps with only one non-empty node, which does not help in classification and can make some convergence problems (condition 10 will never be satisfied for those maps because in the concerned neuron, $m_j = m_0$). To overcome this problem we added a new condition for the maps to be considered. We will only consider the maps with more than one neuron having more than one element.

We choose a learning rate $R(t) = \frac{1-t}{N}$, where N is the number of data points. *R(t)* is a convenient learning rate function that decreases in each iteration until it, vanishes at the end. This learning rate will be used for $\alpha(t) = 0.3 * R(t)$ and for $h(t) = max(m,n) * R(t)$ where m and n are the dimensions of a given map. The algorithm implementation is inspired by [18] and the programming language used is Python. Tensor Flow framework was used in SOM class for creating and training the networks.

III. Simulation data and results

We consider a power grid with N users that we want to cluster, in order to have an insight about their future power demand and their behavior in response to some DR programs. One classification relies on information regarding socio-economic characteristics of the customers such as sex, age, employment status, social class, number of people living in the house, people over 15, people under 15 and type of accommodation. A second classification will use half-hour-separated historic electricity loads for a period of 208 days. The use of GHSOM will give a visual clustering of customers in a way that individuals who are in the same map unit or close to each other will be more likely to behave similarly when it comes to power consumption or response to a given DR program. In the first classification, we used a data set related to 1013 different customers, describing general information about the user such as sex, age, employment status, social class, number of people living in the house, people over 15, people under 15 and type of accommodation. While in the second classification, we used data related to electricity consumption including half-hour separated meters for each household. The data set contains 10 000 meters per household i.e. about 208 days. This dataset contains 41 data points, each one presents a customer and have 10 000 dimension. The data is provided by the Irish social science data archive ISSDA [19] and was collected in Ireland in 2008.

A. *Classification by customers' socio-economic information*

The simulation is performed using a dataset of 1013 customers and 8 attributes. We run the algorithm several times using different parameters τ_m and τ_u. The architecture as well as the classification of inputs vary when we change one or both parameters. Table 1 shows the results of the simulations. We varied the two parameters between 0.2 and 0.6 and registered information about processing time, depth in terms of number of levels, total number of maps and maximum size of the maps.

Table 1: Comparison of GHSOM architectures for different τm and τu parameters.

τ_u \ τ_m	0.2	0.4	0.6
0.2	89.77 sec. 4 levels 41 maps 12X8 max	48.96 sec. 4 levels 39 maps 4X7 max	43.85 sec. 4 levels 52 maps 2X6 max
0.4	60.23 sec. 3 levels 18 maps 16X6 max	32.22 sec. 3 levels 25 maps 6X5 max	37.54 sec. 5 levels 50 maps 4X2 max
0.6	46.002 sec. 2 levels 3 maps 13X10 max	12.07 sec. 3 levels 8 maps 7X4 max	4.24 sec. 3 levels 4 maps 3X3 max

The results came as mentioned in the previous section, the bigger $\tau_u$and τ_m are, the simpler is the architecture and faster is the convergence. We also notice that for a fixed τ_u, the maximum size of the maps decreases for bigger values of τ_m. The same assumption roughly applies, for the number of levels decreasing with τ_u. As for the number of maps, it seems to be dependent on both parameters. It increases generally with τ_m and decreases with higher values of τ_u. An interpretation of this can be such that with lower values of τ_m maps are growing in size and making accurate classification of the data so that when condition (10) is checked it is more probable that neurons don't need to be developed further, especially with bigger values of τ_u. Which makes the second term of the inequality (10) big enough to help satisfying the condition. Therefore, there won't be many maps emerging from the neurons.

The classification of the 1013 customers was saved in excel files. Each file represents a map and contains columns of customer IDs. Each column represents a node in the map and customers belonging to the same node are considered in the same class. Which can give insight about similarities in their behavior or help design some customized DSM strategies.

B. *Classification by half-hour load information*

The second simulation concerns energy consumption patterns. We want to classify customers according to their amount of energy consumed in each half hour. The data set that we have contains 41 customers' consumption measures during 208 days. We will use every half-hour measure for each customer, the resulting dataset has 41 data point and 10 000 attributes. This simulation is also a test to the robustness of the algorithm to very high dimensional data. We run the algorithm with parameters $\tau_m = 0.5$ and $\tau_u = 0.5$. The processing time is 3.44 seconds and resulted in 2 hierarchical maps, first one is a map of shape 2X2 and the second one is emerging from neuron (1,1) and has a shape of 3X4. In the first map, the mean quantization error is $m_0 = 95.47$, inputs were distributed on two neurons namely neuron (1,1) got 21 input and neuron (1,2) got 20 input. The map has not grown in size because condition (8) was satisfied in the first iteration. However, condition (10) was not satisfied for neuron (1, 1). Therefore, it has to be developed further in a new map.

Table 2: SOM of the first layer

21 samples **6003, 6039, 6030, 6028 6010, 6037, 6035, 6031 6001, 6020, 6017, 6022 6040, 6008, 6029, 6005 6021, 6014, 6041,6032, 6004** $m_1 = 107.92058$	20 samples **6006, 6013, 6025, 6024 6011, 6027, 6002, 6012 6038, 6036, 6026, 6018 6034, 6009, 6023, 6000 6019, 6007, 6016 6033** $m_2 = 65.58465$
No samples $m_3 = 0$	0 samples $m_4 = 0$

In the next map emerging from neuron (1, 1), the mean quantization error is calculated for the 21 customers: $m_0 =$

107.92058. The topology and classification results of this map is given in table 3. The customers are presented by ID and are distributed on the map.

Table 3. Second layer SOM emerging from neuron (1, 1)

6027 **6031** **6026** $m_1 = 88.494$	**6020** $m_2 = 0.00$	**6024** **6038** $m_3 = 70.820$
6001 **6036** $m_4 = 90.289$	**6035** $m_5 = 0.00$	**-** $m_6 = 0.00$
6037 $m_7 = 0.00$	**6013** **6010** $m_8 = 59.774$	**6028** **6002** $m_9 = 90.644$
6003 **6025** $m_{10} = 83.853$	**6006** $m_{11} = 0.00$	**6039** **6030** **6011** **6012** $m_{12} = 69.154$

As shown in table 3, most of neurons have m_j not satisfying condition 10. However, we do not find any sub-maps that have emerged from any of them. The reason is that, as explained in section V, there have been maps that emerged from these neurons but all the data points were gathered in one neuron of these sub maps or there have been only one neuron with more than one element. Therefore, these maps were not considered in the algorithm and consequently were not saved.

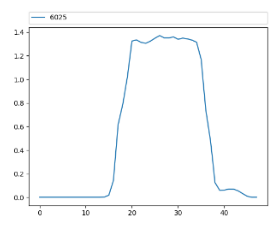
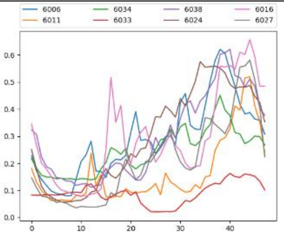
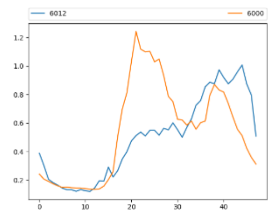
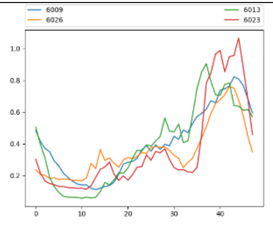
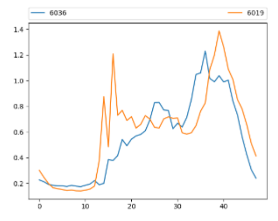
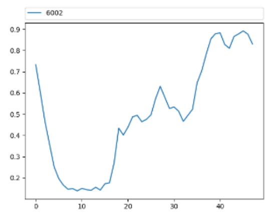
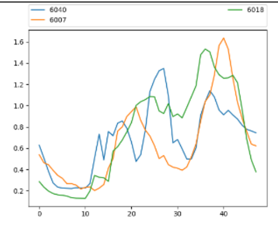
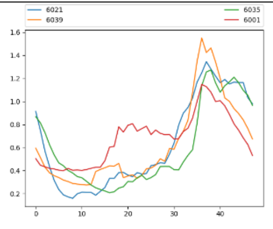
It is essential to note that the spatial distribution of the data points is also important. For example customers in neuron (1,1) are more close to those in (1, 2) and (2,1) than those in (3,4). GHSOM algorithm arranges the neurons that are close to each other on the map, in a way that these neurons will cluster the points that are similar in the input space. This is achieved via the neighbors' activation function that pulls the weights of the "winner's" neighbors towards the input. Which makes them more qualified to win in the case of a similar input.

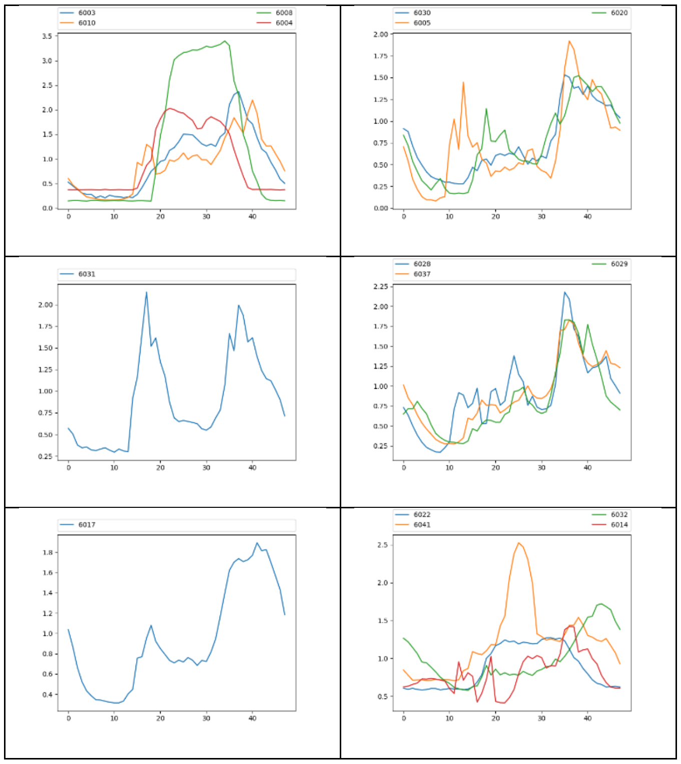
The classification in this case uses time series representing the consumption of each half hour during all the 208 days. It is based a difference between two vectors of 10 000 attribute without even considering the periodicity of the time series or the daily load of a customer. This classification has two weaknesses: The first one is that the distance between two vectors of 10 000 meters will not be able to represent the similarity between two customers because it does not catch the daily habits of customers but only the punctual difference in consumption in each point of time. The second weakness is that using this type of classification we cannot visualize the consumption patterns that is usually visualized by customers' load profiles. To overcome this problems we propose in the next paragraph, a classification based on the average load profiles over the 208 days.

C. Classification by load profile

We transform the previous dataset to a daily average load dataset. Each customer's consumption data will be averaged over all the 208 days to get one load profile for one day. The resulting dataset will have the same number of data points but with 48 attributes representing 24 hours (half-hourly separated) instead of 10 000 attributes. We use the same algorithm to cluster the new data points using parameters such that we can have a simple representation of the clusters. According to the analysis of τ_m and τ_u made in paragraph A of this section we choose a big value of τ_u, and a medium value of τ_m with: $\tau_u = 0.6$ and $\tau_m = 0.4$. The results gave two hierarchic maps with respective shapes 2X7 and 2X2. For a better analysis of the results and for future developments we will only consider the first map. In table 4, we present all the clusters in the map using their load profiles:

Table 4: GHSOM classification results



The map shows that the customers have been classified by their load profiles. It is also a clear visual representation of the customers' daily consumption including peaks and valleys intervals. The map representation shows that some of the nodes contain very homogeneous clusters such as node (2, 2) and node (2, 6), whereas some other clusters have less homogeneity such as node (1, 5). This classification is still considering only the distance between loads at every step. It does not consider the similarities in load shapes. In the next paragraph, we propose a reclassification method that considers the shape of load profiles using load shape indices.

D. *Reclassification using load shape indices*

In [15] a second stage classification was presented to reclassify the customers by taking into consideration the load profiles shapes in the classification. The method is based on five indices called load shape indices. They are derived from load curves and can represent the load shape information and the key information such as peak load value, valley time value, and load level at different levels of pressure time: peak load time, valley load time, flat load time. These indices can provide meaningful information about the load curve shape and they can be used to find the similarities in load profiles between the customers. Authors in [15] defined the five load shape indices as follows:

Load factor: represents the peak load information and the whole load curve shape, and can be defined as,

$$I_{LF} = \frac{P_{av}}{P_{max}} \qquad (11)$$

where P_{av} is the average load of all the day, P_{max} is the maximum load of all the day.

Peak valley factor: denotes variation interval of the load and is defined by,

$$I_{PV} = \frac{P_{max} - P_{min}}{P_{max}} \qquad (12)$$

Peak load factor: represents the average load level at system peak time, and can be defined as,

$$I_P = \frac{P_{av,peak}}{P_{av}} \qquad (13)$$

where $P_{av,peak}$ is the average load during system's peak load time. In the study, we choose 9:00-18:00 as the system's peak load time.

Flat load factor: expresses the average load level at system flat time, represented by,

$$I_F = \frac{P_{av,flat}}{P_{av}} \qquad (14)$$

where $P_{av,flat}$ is the average load during system's flat load time. In the study, 6:00-9:00 and 18:00-21:00 are the system's flat load times.

Valley load factor: represents the average load level at system valley time, which is defined by,

$$I_V = \frac{P_{av,valley}}{P_{av}} \qquad (15)$$

where $P_{av,valley}$ is the average load during system's flat load time. In the study, 0:00-6:00 and 21:00-24:00 are the system's valley load times.

The reclassification method consists of using a supervised classification of each sample using similarities in the load shape indices' space. The idea is that if a given sample has high similarity (in terms of load shape) with samples of a different cluster than its own, it should be reclassified in that cluster. The reclassification algorithm as expressed in [15], consists of the following steps:

1. Select the i^{th} sample (x_i, y_i), where x_i is the input feature vector (load shape indices), y_i is the class the sample belongs to at the first stage.
2. Use all the other loads without the *ith* sample as the training sample 1, $\{(x_k, y_k)\}_{k=1,k\neq i}^{n}$ to train a classifier.
3. Reclassify the i^{th} load in to the new class using the classifier in step 2.
4. Repeat the reclassification until all the loads are reclassified.

We implemented the reclassification algorithm considering each node as a class. In addition, we propose to implement a modified version of the algorithm taking in consideration the features of GHSOM classification. In contrast to classification in [15], we do not have classes but only nodes in a map. A line and column number defines each node. Additionally, the classification has a spatial value; neighbor nodes are more likely to cluster similar inputs. For this reason, we use a modified version of the algorithm: We use lines and columns as two independent classes for each point. We make two separate classifications using lines' numbers then columns' numbers. A third modification concerns the reclassification method in step 3: whenever a sample has to be classified in a different line or column, we put it in the line or column halfway between the new and the old one.

$$position = Old + \frac{New - Old}{2}$$

This process will enable us to consider both the first and second classification and it is valid thanks to the spatial distribution of SOMs.

We therefore have 3 reclassification algorithms to test. The reclassification algorithms can be performed using any supervised classification algorithm such as KNN, SVM, Naïve Byes, Random forests…etc. In [15] SVM was chosen for reclassification due to its reliable performance in classification. Consequently, we used the same algorithm for our case. To determine how much the reclassification was useful, and to compare the three reclassification algorithms performance, we have to compare one of the cluster evaluation indices. We will use the mean index adequacy (MIA) defined as:

$$MIA = \sqrt{\frac{1}{K}\sum_{j=1}^{K}\sum_{l\in\Omega_j}\frac{d^2(\omega_j,l)}{N_j}}$$

Where d is the Euclidian distance, Ω_j is the set of all load of class j, ω_j is the average of the vector of class j and N_j is the total number of class j. K represents the number of clusters. Smaller MIA value shows better clustering effect. We compute two types of MIA: MIA1 is based on normalized load measures where MIA2 is based on normalized load shape indices.

We compare results of the original classification and the reclassification algorithms mentioned above. The results are given in table. R1 is the original reclassification algorithm, R2 is the modified the first modified version and R3 is the second modified version.

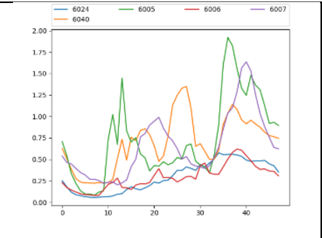
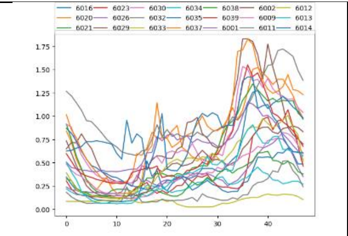
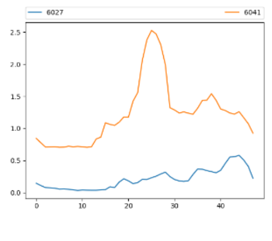
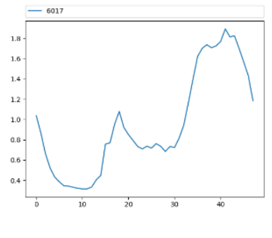
Table 5: Comparison of MIA indices

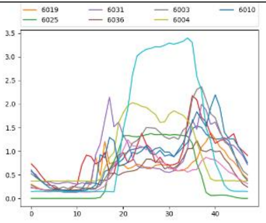
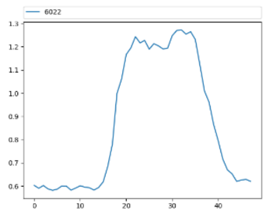
	MIA1	***MIA2***	***MIA1*MIA2***
First classification using GHSOM	*0.6841*	*0.2303*	*0.1576*
R1	*0.8680*	*0.1805*	*0.1567*
R2	*0.8263*	*0.1792*	*0.1481*
R3	*0.7908*	*0.2231*	*0.1764*

The MIA values in table. Show that the first classification is giving the best MIA1 that is based only on load measures. In the second stage classification, MIA1 is getting worse for the all the reclassification algorithms. R3 was the least divergent from the first classification, followed by R2 then R1 in the last place. In terms of MIA2, which represents the load shape indices, The first classification gave the worst value. MIA2 was improved in the second stage classification. R2 has the best MIA2 value followed by R1 then R3 in the last place. These results were expected because the first classification algorithm is considering only the load measures without any consideration of the load shape indices measured by MIA2. Therefore, it is normal that MIA1 is smaller and MIA2 bigger in the first classification. R1 is considering the nodes as classes and changing completely the classes of samples without considering the spatial distribution. Its MIA1 was the worst because of the weak consideration of original classes and positions on the maps. However, it gives more importance to the load shape indices, which made it, come in the second place in terms of MIA2. R2 reclassifies samples based on load shape indices, but also takes into consideration their spatial distribution on the map, which made it the best reclassification algorithm in terms of MIA2 but also not too divergent in terms of MIA1. R3 Is relying on a middle position of the samples between the old and the new lines/columns. Which made it the closest algorithm to the first one in terms of MIA1 and MIA2. Considering both the load shape index and the whole dimension load curve information, the final clustering results can be evaluated by MIA1*MIA2. The product is not as sensitive to the order of magnitude difference of different variables as summing. It can be seen from Table 5, that R2 is giving the best results followed by R1, the original classification algorithm and then R3. It can be concluded that the second stage classification can indeed improve the classification of the first algorithm if a proper reclassification algorithm is chosen.

We finally present in Table 6 the results of the winner algorithm R2 by the graphs of the resulting clusters:

Table 6: Results of reclassification using SVM

	
No samples	
No samples	No samples
No samples	

	
No samples	
No samples	

The first impressions about the results are that more samples are grouped in the clusters and we start having empty nodes. This can be explained by the similarities in curves' shapes that could not be caught by the first classification. In node (1, 2), 21 samples were clustered together whereas only eight samples were originally in that node in the first classification. Samples 6011, 6034, 6033, 6038, and 6016 remained in the same node whereas 6006 and 6024 have moved to the next column in the same line and sample 6027 have moved to the next line in the same column. So 16 new samples were added to this cluster in the second classification. This can give a general overview about what samples are more likely to have the same load curve shape and the ones who have similarities in terms of punctual consumption. Both first and second classifications are useful for the utility company. The first one can provide information about the real time consumption and the groups of customers having similar consumptions. The second classification tells more about the peaks and valleys in the load curves and similarities in the load shape.

IV. Conclusion

This work shows the potential of GHSOM in customer clustering and electricity consumption patterns recognition. It can be used to detect groups of customers who have the same socio-economic properties, the same consumption habits or the same load curves, in order to make targeted offers regarding the electricity pricing or DR incentives. Smart grid technologies such as smart meters and two ways communication systems have been emerging with a fast pace in the recent years. A successful implementation of these technologies can increase the potential of the new electric power system by providing a robust infrastructure for the implementation of DR programs. Additionally, several decision making and business intelligence tools can be developed based on the data collected by the smart meters. Customer classification is one of the most important decision making tools. It can give the utility company a meaningful understanding of consumption patterns in the grid. This information can be used in several practices such as market segmentation and targeted DR programs. GHSOM algorithm was presented in this paper and proposed as a new visual method for customer classification, based on socio-economic information as well as electricity meters. We discussed the details and learning process of the algorithm. The parameters of the algorithm were analyzed and tested using a real dataset of socio-economic information of 1013 customers. Additionally, the algorithm was tested against a dataset of half-hour separated electricity meters for 41 customer throughout 208 days. Due to the very high dimension of the inputs (10 000 measures for each costumer), the results could not provide a good visual representation of the samples. The load curves of 208 days will not be representative of the customers and their consumption habits. Therefore, a new classification was performed based on the average daily loads. This classification gave a good visual separation of the load curves, emphasized the similarities and differences between loads time series. Finally, we presented a second stage classification to take into consideration the load shapes using load shapes indices. The results were compared using MIA index, the comparison showed that the first classification is always better in terms of full time series clustering without considering the shape of load curves. The reclassification based on the lines and columns gave better results in terms of the combination of load time series and load curve shapes criteria. The clusters were redistributed according to the second stage classification and the new visual representation was shown and analyzed. In this paper, the classification using socio-economic data and the one using electricity meters were discussed separately, because the datasets do not contain information about the same customers. A combined classification of the two datasets would be very meaningful in understanding the correlation between the socio-economic situation of customers and their energy consumption patterns.

V. References

[1] Q. Li, Z. Xu, and L. Yang, "Recent advancements on the development of microgrids," *J. Mod. Power Syst. Clean Energy*, vol. 2, no. 3, pp. 206-211, Sep. 2014

[2] Y. Wang, Q. Chen, C. Kang, M. Zhang, K. Wang and Y. Zhao, "Load profiling and its application to demand response: A review," *Tsinghua Science and Technology*. vol. 20, no. 2, pp. 117-129, Apr. 2015.

[3] G. J. Tsekouras, N. D. Hatziargyriou, and E. N. Dialynas, "Two-stage pattern recognition of load curves for classification of electricity customers," *IEEE Trans. Power Syst.*, vol. 22, no. 3, pp. 1120–1128, Aug.2007.

[4] N. M. Kohan, M. P. Moghaddam, S. M. Bidaki, and G. R. Yousefi, "Comparison of modified k-means and hierarchical algorithms in customers load curves clustering for designing suitable tariffs in electricity market," in *Proc. 43rd Int. Universities Power Engineering Conf.*, Padova, Italy, pp. 1–5, Sep. 2008.

[5] Z. Zakaria, K. L. Lo, and M. H. Sohod, "Application of fuzzy clustering to determine electricity consumers' load profiles," in Proc. *IEEE Int. Power*

and Energy Conf., Putra Jaya, Malaysia, pp.99–103, Nov. 28–29, 2006.

[6] B. Stephen, A. J. Mutanen, S. Galloway, G. Burt, and P. Jarventausta,“ Enhanced load profiling for residential customers,” *IEEE Trans. Power Del.*, vol. 29, no. 1, pp. 88–96, Feb. 2014.

[7] G. Chicco, R. Napoli, F. Piglione, M. Scutariu, P. Postolache, and C. Toader, “Emergent electricity customer classification,” *Proc. Inst.Elect. Eng., Gen., Transm. Distrib.*, vol. 152, no. 2, pp. 164–172, Mar. 2005.

[8] G. Chicco, R. Napoli, and F. Piglione, “Comparisons among clustering techniques for electricity customer classification,” *IEEE Trans. Power Syst.*, vol. 21, no. 2, pp. 933–940, May 2006.

[9] S. V. Verdu, M. O. Garcia, C. Senabre, A. G. Marin, and F. J. G. Franco, “Classification, filtering, and identification of electrical customer load patterns through the use of self-organizing maps,” *IEEE Trans. Power Syst.*, vol. 21, no. 4, pp. 1672–1682, Nov. 2006.

[10] G. Chicco and I. S. Ilie, “Support vector clustering of electrical load pattern data,” *IEEE Trans. Power Syst.*, vol. 24, no. 3, pp. 1619– 1628,Aug. 2009.

[11] J. Nagi, K. S. Yap, S. K. Tiong, S. K. Ahmed, and M. Mohamad, “Nontechnical loss detection for metered customers in power utility using support vector machines,” *IEEE Trans. Power Del.*, vol. 25, no. 2, pp.1162–1171, Apr. 2010.

[12] M Piao, HS Shon, JY Lee, KH Ryu, "Subspace Projection Method Based Clustering Analysis in Load Profiling," *IEEE Trans. Power Syst.*, vol. 29, no. 6, pp. 2628-2635, Nov. 2014.

[13] J. Llanos, D. Sáez, R. Palma-Behnke, A. Núñez, G. Jiménez-Estévez, “Load Profile Generator and Load Forecasting for a Renewable Based Microgrid Using Self Organizing Maps and Neural Networks,” *IEEE World Congress on Computational Intelligence*, June 2012 - Brisbane, Australia

[14] S. Haben, C. Singleton, P. Grindrod, “Analysis and Clustering of Residential Customers Energy Behavioral Demand Using Smart Meter Data,” *IEEE Trans. on Smart Grid*, vol. 7, no. 1, pp. 136 – 144, Jan. 2016.

[15] B. Peng, C. Wan, S. Dong, J. Lin, Y. Song, Y. Zhang, J. Xiong, “A two-stage pattern recognition method for electric customer classification in smart grid,” *IEEE Int. Conf. on Smart Grid Comm.* NSW, Australia. pp. 1 - 6, Dec. 2016.

[16] M. Chattopadhyaya, P. K. Danb, S. Mazumdar, “Comparison of visualization of optimal clustering using self-organizing map and growing hierarchical self-organizing map in cellular manufacturing system,” *Applied Soft Computing*, Vol. 22, pp. 528-543. Sep. 2014.

[17] T. Kohonen, “The self-organizing map,” *Proceedings of the IEEE*, Vol. 78, No. 9, Sep 1990.

[18] T. Y. Lin, clusting algorithm implement - numpy version, https://github.com/10yung/GHSOM.

[19] Irish social science data archive ISSDA, http://www.ucd.ie/issda/data/commissionforenergyregulationcer/