*modelling*

MDPI

*Article*

# Time Series Clustering: A Complex Network-Based Approach for Feature Selection in Multi-Sensor Data

**Fabrizio Bonacina *, Eric Stefan Miele and Alessandro Corsini**

Department of Mechanical and Aerospace Engineering, Sapienza University of Rome, Via Eudossiana 118, I00184 Rome, Italy; ericstefan.miele@uniroma1.it (E.S.M.); alessandro.corsini@uniroma1.it (A.C.)

* Correspondence: fabrizio.bonacina@uniroma1.it

Received: 15 April 2020; Accepted: 25 May 2020; Published: 28 May 2020


**Abstract:** Distributed monitoring sensor networks are used in an ever increasing number of applications, particularly with the advent of IoT technologies. This has led to a growing demand for unconventional analytical tools to cope with a large amount of different signals. In this scenario, the modeling of time series in similar groups represents an interesting area especially for feature subset selection (FSS) purposes. Methods based on clustering algorithms are very promising for FSS, but in their original form they are unsuitable to manage the complexity of temporal dynamics in time series. In this paper we propose a clustering approach, based on complex network analysis, for the unsupervised FSS of time series in sensor networks. We used natural visibility graphs to map signal segments in the network domain, then extracted features in the form of node degree sequences of the graphs, and finally computed time series clustering through community detection algorithms. The approach was tested on multivariate signals monitored in a 1 MW cogeneration plant and the results show that it outperforms standard time series clustering in terms of both redundancy reduction and information gain. In addition, the proposed method demonstrated its merit in terms of retention of information content with respect to the original dataset in the analyzed condition monitoring system.

**Keywords:** time series clustering; unsupervised feature subset selection; complex network; visibility graph; community detection; multivariate sensor signals; cogeneration plant

## 1. Introduction

The primary goal of industrial Internet of Things (IoT) has been linking the operations and information technology for insight into production dynamics. This potential flexibility entails a floor of technologies made of distributed networks of physical devices embedded with sensors, edge computers, and actuators used to identify, collect, and transfer data among multiple environments. Such IoT-based cyber-physical systems (CPS) establish a direct integration of engineering systems into digital computer-based ones, where the measurement or sensing technologies play an essential role in capturing the real world. Data collected are then the main ingredient to lift efficiency, accuracy, and economic benefits with the added merits of minimum human intervention [1,2].

A consequence of such transformation is that the sensor layer, customarily used to measure, is now the means to map the actual status (of the process) into the cyber-world to derive process information, collect it in databases, and use it as a basis for the models which can be adapted (ideally by self-optimization) to the real situations. In this vein, the CPS provides a holistic view of the engineering systems and enables a bi-directional physical to digital interaction via multimodal interfaces [3,4]. Unlike the classic concepts derived from control theory, CPS forms the basis to describe even complex interactions and thus anticipate process deviations or interpretation and prediction of system behavior,

diagnosis of exceptional events, explanation of causal mechanisms, and reactive response to urgent events [5].

On the other hand, CPS are disclosing large quantities of data to create augmented knowledge about process control. These high-dimensional datasets typically include heterogeneous measures acquired through a large variety of sensors which may have different sampling rates and communication protocols [6]. Following this trend, in recent years, there has been a rapid development of mathematical modeling techniques and analytical tools able to address the aforementioned problems, while meeting the new IoT requirements to assist process decision makers.

In this scenario, Artificial Intelligence (AI) is playing a major role to support development of intelligent systems in process control engineering [7]. Examples are data preprocessing tasks, such as outlier removal [8], the replacement of missing values [9], and the definition of machine learning models for predictive and prescriptive maintenance [10–12]. Moreover, AI can be used to model a system without the computational complexity of a full simulation or a physical analysis, learning functional dependencies between system components from the data [13]. Even though most AI models are data-driven, overabundant high-dimensional data can have a negative impact on model behavior and efficiency, not only because of the computational complexity but also in terms of accuracy [6]. The existence of redundant and noisy data contributes to the degradation of the performance of learning algorithms, making them less scalable and reliable [14].

Remedial strategies must fundamentally select the most representative features of the original dataset, by using feature subset selection (FSS) techniques. FSS, part of the larger family of dimensionality reduction approaches, aims at extracting a smaller set of representative features that retains the optimal salient characteristics of the data, especially in terms of global information content.

The dimensionality problem is particularly evident in data collected from control monitoring systems in engineered processes and machines, due to the strong presence of redundant data related to physical quantities, with similar trends, monitored in different points of the system, or to parameters of different nature but strongly correlated [15]. By detecting and removing irrelevant, redundant, or noisy features, FSS can alleviate the curse of dimensionality from industrial sensor networks, improving model interpretability, speeding-up the learning process, and enhancing model generalization properties [16,17].

FSS techniques can be both supervised and unsupervised depending on the availability of the data class labels used for guiding the search for discriminative features [18]. Recently, unsupervised FSS has been attracting an ever growing interest, especially in the control and monitoring field, because of the lack of ground truth data in many real word applications [19].

Traditional unsupervised FSS methods based on dependence measures, such as correlation coefficients, linear dependence, or statistical redundancy, are already widely used [19]. Recently, feature clustering demonstrated its merit in terms of accuracy with respect to other unsupervised approaches [20]. In addition, clustering algorithms outperform state-of-the-art methods in detecting groups of similar features [21,22], as well as in selecting metrics (one or more features) out of every cluster to reduce the dimensionality of the data with more or less granularity based on the application.

In this paper, we propose an unsupervised FSS algorithm based on a novel feature clustering technique tailored to time series collected from a real industrial IoT sensor networks. The clustering approach complements different tools from complex network theory, which are becoming promising in the field of nonlinear time series analysis for their ability to characterize the dynamical complexity of a system [23]. In particular, we used visibility graphs [24] to map time series in the network domain, then node degree sequences extraction [25] to characterize them, and finally community detection algorithms [26] to perform time series clustering.

The proposed method was tested on the sensor network of a 1 MW Combined Heat and Power Plant (CHP) central monitoring system. The heterogeneous dataset includes measurements from the engine, the auxiliaries, the generator, and the heat recovery subsystem. Finally, we compared with other traditional time series clustering methods in terms of redundancy and information content for FSS.

The rest of the paper is organized as follows. In Section 2, we describe the background and related works. In Section 3, we present the unsupervised FSS method. In Section 4, we describe the case study. In Section 5, we report experimental results to support the proposed approach. In Section 6, we summarize the present work and draw some conclusions.

## 2. Background

In this section we review the main approaches used for FSS and time series clustering, together with some complex network analysis tools that represent the constituent parts of the proposed method.

### *2.1. Feature Subset Selection*

A possible classification of Feature Subset Selection (FSS) methods consists in *embedded*, *wrapper*, and *filter* approaches.

The *embedded* methods [27] are usually related to learning algorithms such as decision trees or neural networks that perform feature selection as part of the their training process. The *wrapper* methods [28], instead, use a predetermined learning algorithm to evaluate the goodness of feature subsets. Since a training and evaluation phase is required for every candidate subset, the computational complexity of these methods is large. Differently, the *filter* methods [29] do not rely on learning algorithms, but select features based on measures such as correlation. By combining *filter* and *wrapper* methods, it is possible to evaluate features through a predetermined learning algorithm without the computational complexity of wrappers thanks to an initial feature filtering [30].

In the context of *filter* methods, it was found that clustering-based methods outperform traditional feature selection methods, and they also reduce more redundant features with high accuracy [21,22]. As a part of filter methods, the clustering algorithms are used to group features according to their similarity. From a dimensionality reduction perspective, one or more representative variables for each cluster must be identified. For example, in [31], the cluster center is used as representative, while, in [32], a single optimal variable is considered based on the $R^2$ correlation coefficient.

FSS approaches can be further categorized into supervised and unsupervised. The former process feature selection by data class labels, while the latter relies on intrinsic properties of the data. Recently, unsupervised FSS are attracting an ever growing interest due to the widespread occurrence of unlabeled datasets in many applications [33,34]. For this reason, the present paper focuses only on unsupervised methods.

#### 2.1.1. Time Series Clustering

Conventional clustering algorithms (partitional, hierarchical, and density-based) are unable to capture temporal dynamics and sequential relationships among data [35]. For these reasons, tremendous research efforts have been devoted to identify time series-specific algorithms.

The most common approaches involve modifying the conventional clustering algorithms to adapt them to deal with raw time series or to convert time series into static data (feature vectors or model parameters) so that conventional clustering algorithms can be directly applied. The former class of approaches includes direct methods, also called *raw data-based* [36–39], while the latter refers to indirect methods that can be distinguished between *model-based* [40–42] and *feature-based* [43–45].

In addition, according to the way clustering is performed, the algorithms can be grouped into *whole time-series clustering* and *subsequence clustering*, a valid alternative to reduce the computational costs by working separately on time series segments. Detailed reviews of clustering algorithms for time series can be found in [46–48].

It has been recently demonstrated that network approaches can provide novel insights for the understanding of complex systems [23,49], outperforming classical methods in the ability to capture arbitrary clusters [50]. In particular, the weakness of conventional techniques resides in the use of distance functions which allow finding clusters of a predefined shape. In addition, they identify only local relationships among neighbor data samples, being indifferent to long distance global relationships

[50]. Examples of network approaches for time series clustering can be found in the literature, making use of DTW and hierarchical algorithms [51] and community detection algorithms [50].

#### 2.1.2. Evaluation Metrics for Unsupervised FSS

To evaluate an unsupervised FSS method for time series, two main indicators are widely used: redundancy and information gain.

In particular, the redundancy of information among a set of time series $y = (y_1, y_2, \ldots, y_N)$ is quantified by the metric $W_I$, which is defined as:

$$W_I(y) = \frac{1}{|y|^2} \sum_{y_i, y_j \in y} MI(y_i, y_j) \tag{1}$$

where $MI(y_i, y_j)$ is the mutual information between time series $y_i$ and $y_j$ [16]. A low value of $W_I$ is associated to a set of time series that are maximally dissimilar to each other. It is also possible to consider the rate of variation of this metric, represented by the redundancy reduction ratio (RRR):

$$RRR = \frac{W_I(y) - W_I(\bar{y})}{W_I(y)} \tag{2}$$

The information gain, instead, is computed in terms of the Shannon entropy $H$ [52], which reads as:

$$H(X) = \sum_i p(x_i) \log_2(x_i) \tag{3}$$

where $X$ is the data matrix associated to the set of time series $y$, being every row a sample of the observations and every column a different time series, and $x_i$ is the $i$th row of such matrix. The information gain is computed as the variation of entropy between the original time series and the time series after the feature selection $\bar{y}$. If the rate of variation is considered, it is possible to define the information gain ratio (IGR):

$$IGR = \frac{H(X) - H(\bar{X})}{H(X)} \tag{4}$$

where $X$ and $\bar{X}$ are the data matrices associated to $y$ and $\bar{y}$.

### *2.2. Complex Network Analysis*

In this section, we illustrate Complex Network Analysis (CNA) methods used in the present study, namely visibility graphs, local network measures, and community detection algorithms.

#### 2.2.1. Visibility Graph

The visibility graph algorithm is a method to transform time series into complex network representations. This concept was originally proposed in the field of computational geometry for the study of mutual visibility between sets of points and obstacles, with applications such as robot motion planning [53]. The idea was extended to the analysis of univariate time series [24], making it possible to map a time series into a network that inherits several properties of the time series itself. Moreover, visibility graphs are able to capture hidden relations, merging complex network theory with nonlinear time series analysis [23].

In particular, the visibility graph algorithm maps a generic time series segment $y_n^t = ((s_n^t)_1, (s_n^t)_2, \ldots, (s_n^t)_L)$ in a graph by considering a node (or vertex) for every observation $(s_n^t)_i$ for $i = 1, \ldots, L$, where $L$ is the number of observations in the segment. The edges of the graph, instead, can be generated using two different algorithmic variants: the natural visibility graphs and the horizontal visibility graphs.

The natural visibility graph algorithm [24] generates an edge with unitary weight between two nodes if their corresponding observations in the series are connected by a straight line that is not obstructed by any intermediate observation. Formally, two nodes $a$ and $b$ have visibility if their corresponding observations $(s_n^t)_a = (t_a, v_a)$ and $(s_n^t)_b = (t_b, v_b)$ satisfy the condition:

$$v_c < v_b + (v_a - v_b)\frac{t_b - t_c}{t_b - t_a} \tag{5}$$

for any intermediate observation $(s_n^t)_c = (t_c, v_c)$ such that $a < c < b$. $t_a$ and $t_b$ represent the timestamps of the two samples, while $v_a$ and $v_b$ are the actual observed values. A computationally more efficient algorithmic variant is the horizontal visibility graph [54,55], based on a simplified version of Equation (5).

Visibility graphs can be enhanced by considering its weighted version [56], where the weight between any pair of nodes $(s_n^t)_a = (t_a, v_a)$ and $(s_n^t)_b = (t_b, v_b)$ reads as:

$$w_{ab} = \sqrt{(t_b - t_a)^2 + (v_b - v_a)^2} \tag{6}$$

A schematic illustration of the weighted visibility graph construction is shown in Figure 1.

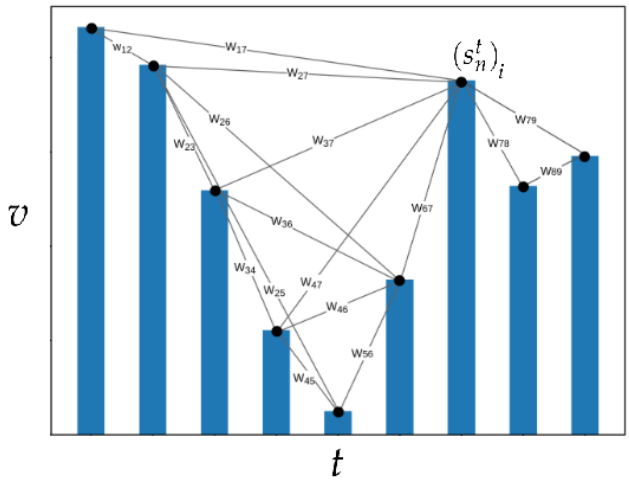



**Figure 1.** A schematic illustration of the algorithm for the construction of the weighted natural visibility graph. Notably, the time series is represented as a bar chart and $(s_n^t)_i$ is the $i$th observation.

### 2.2.2. Local Network Measures

Networks can be composed by many nodes and edges, making the analysis and the unveiling of hidden relationships very challenging. For this reason, global and local network measures are used, respectively, to extrapolate synthetic topological information from the whole network and study the role nodes play in its structure. The latter purpose can be tackled using different centrality measures. The first historically proposed one is the degree centrality [25] which allows detecting the most influential nodes within the network. This measure is based on the simple concept that the centrality of a vertex in a network is closely related to the total number of its connections. In particular, the weighted degree centrality of a node $i$ in a graph reads as:

$$(k_n^t)_i = \sum_{j \in L} w_{ij} \tag{7}$$

where $L$ is the number of nodes, $w_{ij}$ is the weight of the edge connecting nodes $i$ and $j$, and $k_n^t = ((k_n^t)_1, (k_n^t)_2, \ldots, (k_n^t)_L)$ is also called the degree sequence of the graph.

Another measure is the eigenvector centrality [57], which is used for determining elements that are related to the most connected nodes. The betweenness centrality [58], instead, is able to highlight

which nodes are more likely to be in the network communication paths, and, finally, the closeness centrality [58] measures how quickly information can spread from a given node.

#### 2.2.3. Community Detection

Since the last century, networks have been widely used to represent and analyze a large variety of systems, from social networks [59,60] to time series. One of the driver has been the growing interest in detecting groups of nodes (communities) that are strongly connected by high concentrations of edge (relations), sharing common properties or playing similar roles in the network [61–63]. In particular, nodes that are central in a community may be strongly influential on the control and stability of the group, while boundary nodes are crucial in terms of mediation and exchanges between different communities [64].

Many community detection methods have been proposed to date and a possible classification includes traditional, modularity-based, spectral, and statistical inference algorithms.

Traditional methods include graph partitioning [65,66], which selects groups of predefined size by minimizing the number of inter-group edges; distance-based methods [67], where a distance function is minimized starting from local network measures; and hierarchical algorithms [68] that produce multiple levels of groupings evaluating a similarity measure between vertices. Modularity-based methods [69–71], instead, try to maximize the Newman–Girvan modularity [72], which evaluates the strength of division into communities. One of the most popular methods is the Louvain's method [26]. This algorithm is based on a bottom-up approach where iteratively groups of nodes are created and then aggregated in larger clusters. In particular, nodes are initially considered as independent communities and the best cluster partition is identified by moving single nodes to different communities searching for a local maxima of the modularity. Then, a new network is constructed by modeling clusters as graph vertices and by computing edge weights as sum of the connection weights between adjacent nodes belonging to different communities. These steps are iteratively repeated until convergence, corresponding to a maximum of modularity.

Another category of community detection methods are the spectral algorithms [73], which detect communities by using the eigenvectors of matrices such as the Laplacian matrix of the graph. Finally, statistical inference algorithms [74,75] aim at extracting properties of the graph based on hypotheses involving the connections between nodes.

##### Visualization

Community detection algorithms are typically integrated with exploratory network tools in order to improve the network visualization [76]. These tools become essential to give insight into the network structure, by revealing hidden structural relationships that may otherwise be missed. As described in [77], there is a large variety of specialized exploratory network layouts (e.g., force-directed, hierarchical, circular, etc.) based on different criteria. Among them, force-directed layouts are extensively applied in the identification of communities with denser relationships owing to their capability to capture the modular aspect of the network.

An example of force-directed layout is the Frushterman–Reingold algorithm [78], which considers nodes as mass particles and edges as springs between the particles and the goal is to minimize the energy of this physical system in order to find the optimal configuration. This process is only influenced by the connections between nodes, thus, in the final configuration, the position of a node cannot be interpreted on its own, but has to be compared to the others.

##### Evaluation Metrics

Community detection algorithms can be evaluated through several external or internal indicators, customary for clustering or specially designed for community detection in networks [79]. External indicators, which evaluate the clustering results based on ground truth data, include purity [80], Rand index [81], and normalized mutual information [82]. On the other hand, internal indices, relying on

intrinsic information to the clustering and specifically designed for networks. are the modularity [72] and the partition density [83].

## 3. Methods

This section discusses the proposed method, starting from the problem of time series clustering up to the task of unsupervised FSS. Given a set of $N$ time series $y = y_1, y_2, \ldots, y_N$, the main steps of the proposed clustering approach are here summarized.

a. Remove time series noise through a low-pass filter.
b. Segment time series $y_n$ into consecutive non-overlapping intervals $s_n^1, s_n^2, \ldots, s_n^T$ corresponding to a fixed time amplitude $L$, where $T$ is the number of segments extracted for each time series.
c. Transform every signal segment $s_n^t$ ($t = 1, \ldots, T$ and $n = 1, \ldots, N$) into a weighted natural visibility graph $G_n^t$.
d. Construct a feature vector $k_n^t = ((k_n^t)_1, (k_n^t)_2, .., (k_n^t)_L)$ for each visibility graph $G_n^t$, where $(k_n^t)_i$ is the degree centrality of the $i$th node in the graph and $k_n^t$ the degree sequence of the graph.
e. Define a distance matrix $D^t$ for every $t$th segment ($t = 1, \ldots, T$), where the entry $d_{ij}^t$ is the Euclidean distance between the degree centrality vectors $k_i^t$ and $k_j^t$. Every matrix gives a measure of how different every pair of time series is in the $t$th segment.
f. Compute a global distance matrix $D$ that covers the full time period $T$ where the entry $(i, j)$ is computed as $d_{ij} = \frac{1}{T} \sum_{t=1}^{T} d_{ij}^t$, averaging the contributions of the individual distance matrices associated to every segment.
g. Normalize $D$ between 0 and 1, making it possible to define a similarity matrix as $S = 1 - D$, which measures how similar every pair of time series is over the full time period.
h. Build a weighted graph $C$ considering $S$ as an adjacency matrix.
i. Cluster the original time series by applying a community detection algorithm on the graph $C$ and visualize the results through a force-directed layout.

Figure 2 illustrates the flowchart of the methodology.

After the initial stages of data filtering (Step a) and time series segmentation (Step b), for the transformation of every signal into network domain (Step d), we used the natural weighted visibility graphs. The natural variant was preferred to the horizontal one because it is able to capture properties of the original time series with higher detail, avoiding simplified conditions. The weighted variant, on the other hand, is used to magnify the spatial distance between observations that have visibility and thus avoid binary edges in favor of weighted edges in the visibility graph.

Since we used natural weighted visibility graphs to map time series into networks, for the extraction of a feature vector for each signal segment (Step e), we considered the weighted degree centrality sequence of the network, as suggested in [84], because it is able to fully capture the information content included in the original time series [25,85].

Then, after the construction of the segment distance matrices $D^t$ and the normalized global similarity matrix $S$ together with its graph representation $C$ (Steps f–h), we used the modularity-based Louvain's method in step (Step i) for community detection since extremely fast and well performing in terms of modularity.

To achieve a modular visualization of the clusters detected by the discussed method and their mutual connections, we used a force-directed algorithm, namely the Frushterman–Reingold layout, as a graphical representation.

Finally, for specific unsupervised FSS purposes, we considered a representative parameter for each cluster. Such parameters were identified based on their importance within the communities, by considering the signals with highest total degree centrality in their respective groups.

Every part of the proposed approach was developed in Python 3.6 [86], using the Numpy [87] and NetworkX [88] packages.

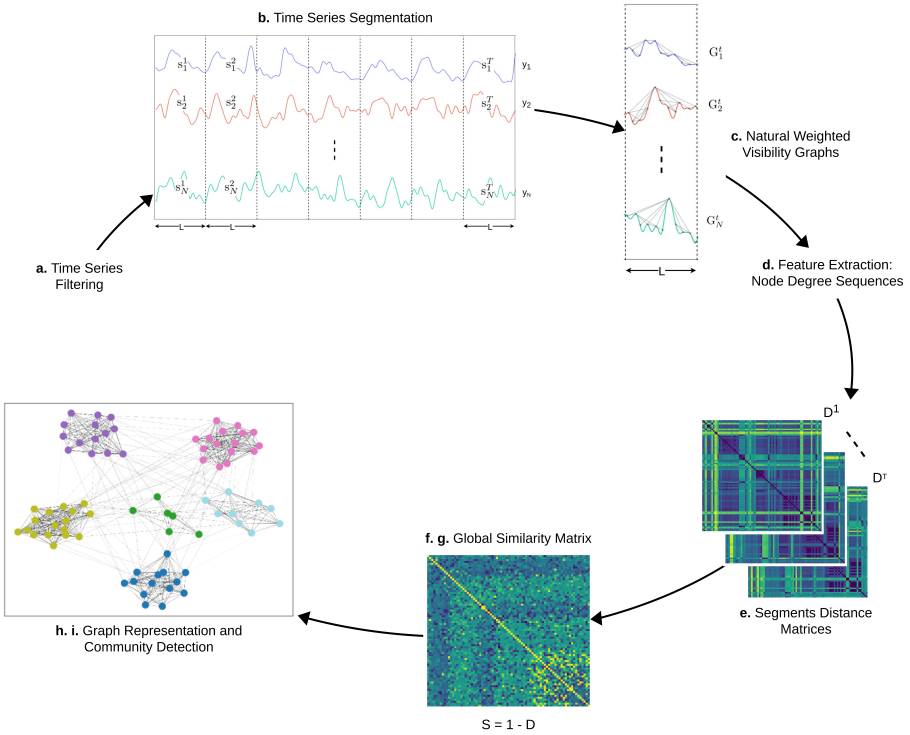



**Figure 2.** Flowchart of the proposed time series clustering methodology.

## 4. Case Study

This section deals with the case study considered for the applications of the proposed method, which is an internal combustion engine used in industrial cogeneration (or CHP).

The CHP system consists of a four-stroke ignition engine ($P = 1032$ kW) fueled with vegetable oil, coupled to a three-phase synchronous generator. The electricity produced is used to meet the self-consumption of the plant and the production surplus is fed into the grid.

The heat recovery takes place both from the engine cooling water circuit and from the exhaust gases. In particular, the heat exchange with engine cooling water ($t = 65$–$80$ °C) is used both to meet part of the plant heating requirement and for the preheating of the fuel before the injection phase. The return water from the plant is cooled by a ventilation system consisting of four fans ($P = 15$ kW).

The exhaust gases, after being treated in a catalyst, are conveyed inside a boiler of 535 kW thermal power, which is used to produce steam at about 90 °C useful for different production lines. A schematic representation of the system is shown in Figure 3.

The system is equipped with a sensor network for condition monitoring and control purposes that samples every minute for a total of 90 physical quantities at different points. The data used for the case study go from 25 June 2014 to 5 May 2015. The early preprocessing phase involved the removal of the parameters that were constantly flat and the cumulative signals, thus reducing the number of the starting parameters to 78. The final list of monitored CHP plant variables considered for the analysis is reported in Table 1. In the preprocessing phase, the outliers caused by sensor errors were also removed. To deal with unusual dynamics linked to system shutdowns, observations where the active power of the system was zero were filtered out. Afterwards, we resampled the data every 15 min to filter constant signal intervals and reduce the amount of measurements processed by the algorithm. The resulting data matrix used as input for the analysis had 30,240 rows and 78 columns. Finally, we built time series segments including 24 h of observations to capture the typical daily cycle of the plant.

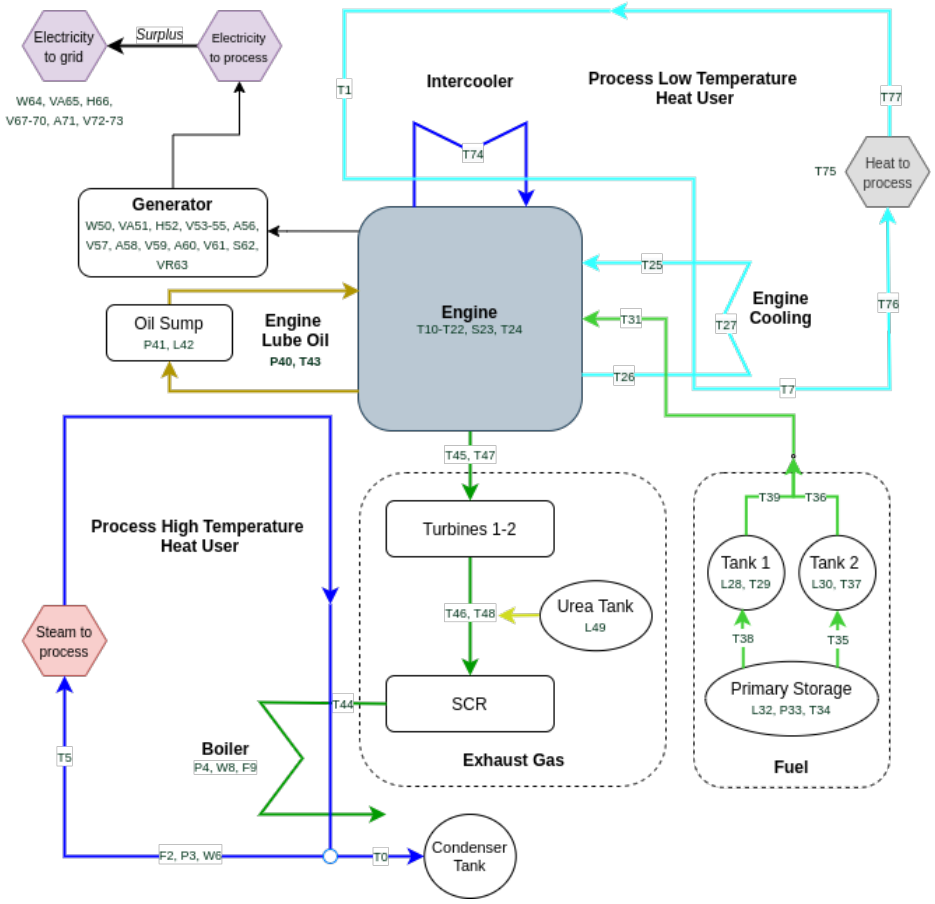



**Figure 3.** Schematic block diagram of the CHP system with measuring points.

**Table 1.** List of monitored CHP plant parameters.

| ID | Variable | ID | Variable | ID | Variable |
|---|---|---|---|---|---|
| T0 | Condense Temp. [°C] | T26 | Engine 1 Out Temp. [°C] | H52 | Gen Frequency [Hz] |
| T1 | Hot Water Temp. [°C] | T27 | Engine 2 Out Temp. [°C] | V53 | Gen L1-L2 Concat. Volt. [V] |
| F2 | Steam Flow Rate [m3/h] | L28 | Tank 1 Level [%] | V54 | Gen L2-L3 Concat. Volt. [V] |
| P3 | Steam Out Pressure [bar] | T29 | Tank 1 Temp. [°C] | V55 | Gen L3-L1 Concat. Volt. [V] |
| P4 | Steam Pressure [bar] | L30 | Tank 2 Level [%] | A56 | Gen Phase 1 Current [A] |
| T5 | Steam Temp. [°C] | T31 | Zone 4 Temp. [°C] | V57 | Gen Phase 1 Volt. [V] |
| W6 | Steam Thermal Power [kW] | L32 | Tank Level [lt] | A58 | Gen Phase 2 Current [A] |
| T7 | Cold Water Temp. [°C] | P33 | Tank Pressure [bar] | V59 | Gen Phase 2 Volt. [V] |
| W8 | Thermal Power [kW] | T34 | Zone 1 Temp. [°C] | A60 | Gen Phase 3 Current [A] |
| F9 | Water Flow Rate [m3/h] | T35 | Zone 2 Temp. [°C] | V61 | Gen Phase 3 Volt. [V] |
| T10 | Casing Out Temp. [°C] | T36 | Zone 3 Temp. [°C] | S62 | Gen Power Factor |
| T11 | Cylinder 1A Temp. [°C] | T37 | Tank 2 Temp. [°C] | VR63 | Gen Reactive Power [Var] |
| T12 | Cylinder 1B Temp. [°C] | T38 | Zone 5 Temp. [°C] | W64 | Grid Active Power [W] |
| T13 | Cylinder 2A Temp. [°C] | T39 | Zone 6 Temp. [°C] | VA65 | Grid Apparent Power [VA] |
| T14 | Cylinder 2B Temp. [°C] | P40 | Carter Pressure [mbar] | H66 | Grid Frequency [Hz] |
| T15 | Cylinder 3A Temp. [°C] | P41 | Oil Pressure [bar] | V67 | Grid L1-L2 Concat. Volt. [V] |
| T16 | Cylinder 3B Temp. [°C] | L42 | Sump Level [%] | V68 | Grid L2-L3 Concat. Volt. [V] |
| T17 | Cylinder 4A Temp. [°C] | T43 | Oil Temp. [°C] | V69 | Grid L3-L1 Concat. Volt. [V] |
| T18 | Cylinder 4B Temp. [°C] | T44 | SCR Out Temp. [°C] | V70 | Grid Phase 1 Volt. [V] |
| T19 | Cylinder 5A Temp. [°C] | T45 | DX Turbine In Temp. [°C] | A71 | Grid Phase 2 Current [A] |
| T20 | Cylinder 5B Temp. [°C] | T46 | DX Turbine Out Temp. [°C] | V72 | Grid Phase 2 Volt. [V] |
| T21 | Cylinder 6A Temp. [°C] | T47 | SX Turbine In Temp. [°C] | V73 | Grid Phase 3 Volt. [V] |
| T22 | Cylinder 6B Temp. [°C] | T48 | SX Turbine Out Temp. [°C] | T74 | Intercooler In Temp. [°C] |
| S23 | Speed [rpm] | L49 | Urea Tank Level [%] | T75 | Plant Delta Temp. [°C] |
| T24 | Supercharger Temp. [°C] | W50 | Gen Active Power [W] | T76 | Plant In Temp. [°C] |
| T25 | Engine 1 In Temp. [°C] | VA51 | Gen Apparent Power [VA] | T77 | Plant Out Temp. [°C] |

## 5. Results

This section provides a detailed discussion about the experimental results obtained by the proposed approach, followed by a comparison with two traditional time series clustering methods.

Figure 4 shows the plot of the 78 standardized signals during a representative period of about two months. Data were extracted during a total measuring time of almost 11 months.

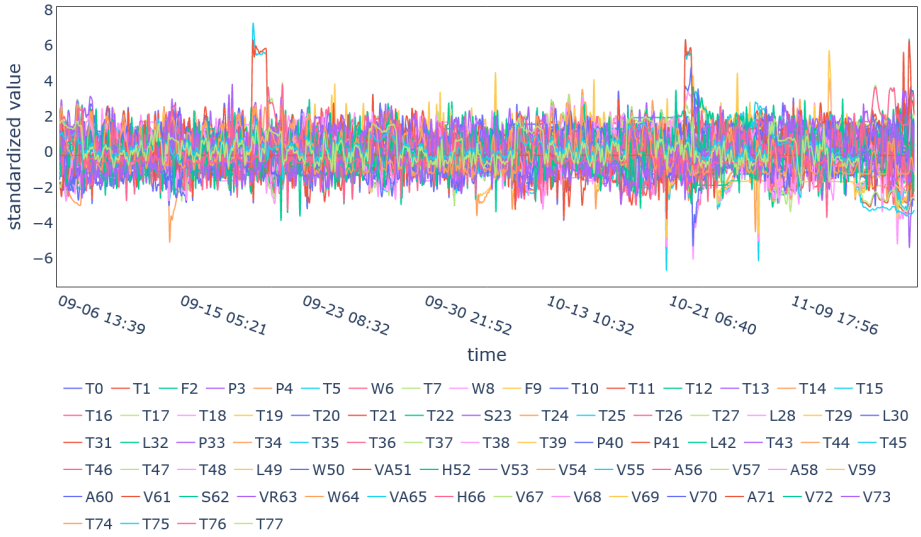



**Figure 4.** Overall standardized time series sampled from 6 September 2014 to 21 November 2014.

The total dataset was then analyzed by applying the method described in Section 3. After the application of a low-pass filter for noise removal, Steps b–d of the workflow, time series were segmented into non-overlapping intervals $s_n^t$, then mapped into natural visibility graphs $G_n^t$, and finally feature vectors were extracted in terms of degree sequences $k_n^t$. Afterwards, in Steps e–g, a global distance matrix $D$ was computed by combining the contribution of all the distance matrices $D^t$, followed by the definition of the similarity between all the pairs of time series. The resulting similarity matrix $S$ is shown in Figure 5.

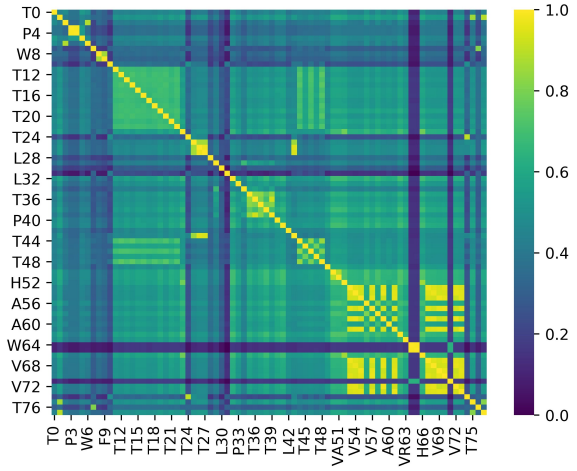



**Figure 5.** Similarity matrix represented as a heat map where the entry $(i, j)$ quantifies the similarity between signals $i$ and $j$.

As per Step h, the similarity matrix $S$ is represented in the form of a weighted graph, also called similarity graph $C$, where each node corresponds to a specific signal and the edge weights quantify

pairwise similarities between time series. To carry out the community detection phase, only the most important edges were taken into account. In particular, we performed edge pruning by filtering the pairwise similarities lower than the second quantile of their probability distribution.

Then, as for Step i, by means of the Louvain's algorithm (see Section 2.2.3), we identified 12 different communities within the filtered similarity graph, which globally cover 70 parameters. Table 2 provides the detail of the variables contained in each cluster with reference to the parameter IDs presented in Table 1.

**Table 2.** Clusters obtained through the Louvain's algorithm.

| Cluster ID | Variable ID |
|---|---|
| Cluster 1 | T29, T34-T39 |
| Cluster 2 | L28, L30 |
| Cluster 3 | T11-T22, T44-T48 |
| Cluster 4 | T7, T10, T24-T27, T31, T43, T74, T76 |
| Cluster 5 | W8, F9 |
| Cluster 6 | T0, F2, P3, P4, T5, W6 |
| Cluster 7 | T1, T75, T77 |
| Cluster 8 | S23, H52, H66 |
| Cluster 9 | V53-V55, V57, V59, V61, V67-V70, V72, V73 |
| Cluster 10 | W50, VA51 |
| Cluster 11 | A56, A58, A60 |
| Cluster 12 | W64, VA65, A71 |

The eight signals shown in Figure 6 were not clustered since they were characterized by independent dynamics. This subset includes engine lube oil parameters, i.e., carter pressure, sump level, and pressure; generator parameters, i.e., power factor and reactive power; parameters in the fuel primary storage, i.e., tank level and pressure; and parameters in the exhaust gas treatment, i.e., urea tank level.

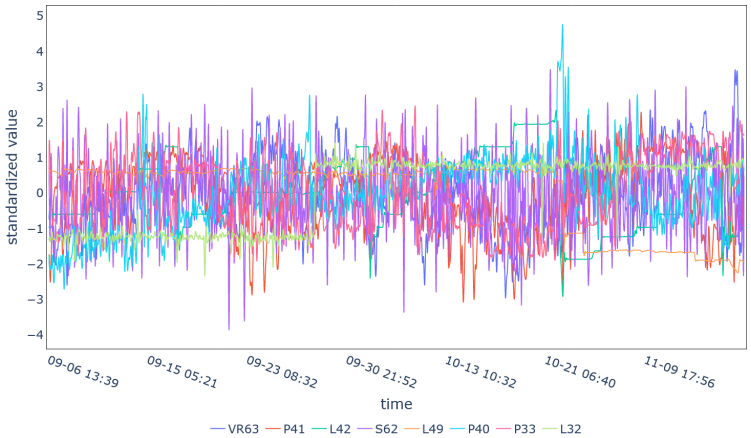
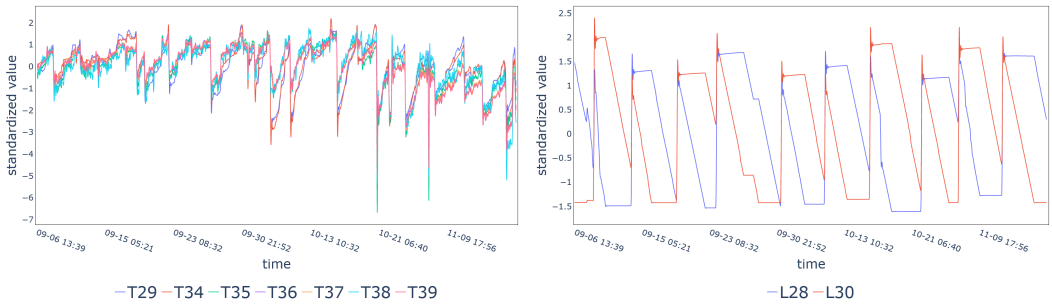



**Figure 6.** Signals that were not included in clusters: lube oil carter pressure (P40), lube oil sump level and pressure (L42 and P41), generator power factor (S62) and reactive power (VR63), fuel tank level and pressure (L32 and P33) in the primary storage, and urea tank level (L49).

Time series clustering results are illustrated with reference to the functional groups shown in the block diagram in Figure 3.

Most of the fuel parameters were grouped into two distinct homogeneous clusters (see Figure 7). Fuel temperatures from the primary storage to the output of the tanks 1 and 2 are included in Cluster 1 (Figure 7a), while Cluster 2 (Figure 7b) groups the fuel levels in the two tanks.

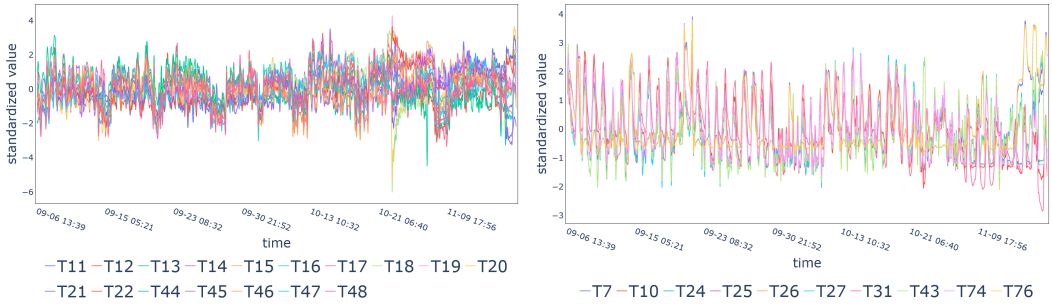



(**a**) Cluster 1 (**b**) Cluster 2

**Figure 7.** Fuel parameters: (**a**) the trends of the time series included in Cluster 1, i.e., temperatures of the fuel in the storage area (T29 and T34–T39); and (**b**) the trends of the time series included in Cluster 2, i.e., fuel levels in tank 1 (L28) and tank 2 (L30).

Engine sensor signals fall, together with other strictly related parameters, into two distinct clusters (see Figure 8). In particular, Cluster 3 (Figure 8a) includes all the cylinder temperatures and the exhaust temperatures, while Cluster 4 (Figure 8b) includes the casing temperatures, the supercharger temperatures, and the temperatures monitored at the engine auxiliaries, e.g., cooling water, lube oil, and intercooler subsystems. Cluster 4 also contains some parameters by the heat exchange with the engine cooling, such as water inlet temperatures of the process heat circuit and inlet fuel temperature.

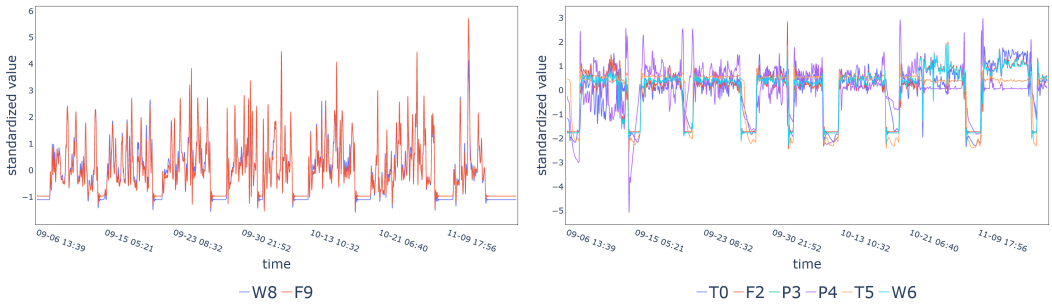
All the parameters of the high temperature heat recovery circuit (process steam demand) were, instead, separated into two distinct groups (see Figure 9). In detail, Cluster 5 (Figure 9a) includes the thermal power and hot water flow rate, monitored at the boiler inlet, while, in Cluster 6 (Figure 9b), all the specific steam parameters are grouped together, such as steam flow rate, pressure, and thermal power, as well as the temperature of the condensed water.




(**a**) Cluster 3 (**b**) Cluster 4

**Figure 8.** Engine sensor signals and strictly related parameters: (**a**) the trends of the time series included in Cluster 3, i.e., temperatures of the cylinders (T11–T22) and temperatures of the exhaust gases (T44=-T48); and (**b**) the trends of the time series included in Cluster 4, i.e., temperatures referred to the external casing of the engine (T10 and T24), engine auxiliaries (T25–T27, T43, and T44)m and parameters influenced by the heat exchange with the cooling water (T7, T76, and T31).

As mentioned above, low temperature heat circuit sensor signals, measured at the plant inlet, are part of Cluster 4 together with other engine and auxiliaries signals (see Figure 8b), while the water temperatures at the plant outlet and the delta in–out temperature are in Cluster 7 (see Figure 10).



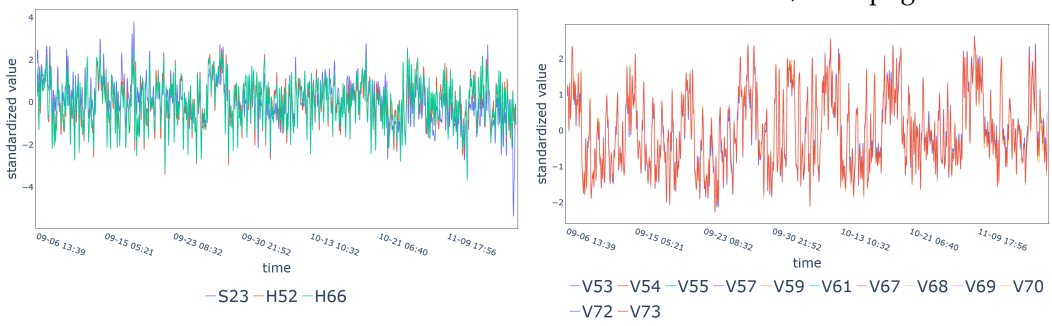

(**a**) Cluster 5 (**b**) Cluster 6

**Figure 9.** Process high temperature user parameters: (**a**) the trends of the time series included in Cluster 5, i.e., thermal power (W8) and flow rate (F9) of the hot water at the boiler inlet; and (**b**) the trends of the time series included in Cluster 6, i.e., steam parameters such as flow rate (F2), pressure (P3), temperature (T5), and thermal power (W6) as well as the condensed wster temperature (T0).


**Figure 10.** Process low temperature user parameters: the trends of the time series included in Cluster 7, i.e., water temperatures at the plant outlet (T1, T77) and delta in–out temperature (T75).

The two principal properties of the electric power supply, frequencies and voltages, were divided into two clusters (see Figure 11). Notably, in Figure 11a, it is possible to note how the engine speed was included in Cluster 8 together with the generator and grid frequencies. On the other and, Cluster 9 includes all the generator and grid voltages.

and Grid/volts.png




(**a**) Cluster 8 (**b**) Cluster 9

**Figure 11.** Generator and Grid frequencies and voltages: (**a**) the trends of the time series included in Cluster 11, i.e., respectively engine speed (S23), and frequencies monitored both at the generator (H52) and at the grid (H66); and (**b**) the trends of the time series included in Cluster 12, i.e., voltages monitored both at the generator (V53–V55, V57, V59, and V61) and at the grid (V67–V70, V72, and V73).

Other electrical parameters, such as powers and currents, have instead been divided into three different clusters (see Figure 12). In particular, Cluster 10 (Figure 12a) and Cluster 11

(Figure 12b) distinguish, respectively, the generator powers from the generator currents, while Cluster 12 (Figure 12c) groups together the grid powers and currents. The latter refer only to Phase 2 current, because the Phase 1 and 3 currents were removed in the preprocessing phase due to sensor malfunctions.

The clustering results show that the proposed approach is independent from the nature of the monitored parameters and their functionality within the system.

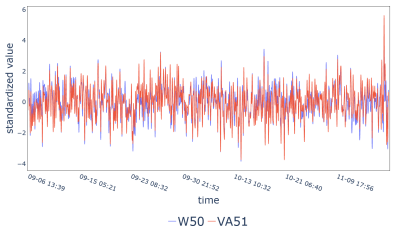
For example, Clusters 1, 2, 7, 9, 10, and 11 (Figures 7a, 10a, 11b, and 12a,b) include only homogeneous variables (e.g., temperatures) belonging the same functional area (e.g., engine). Among those, it is interesting to note how the parameters within Cluster 2, i.e., the fuel levels in the tanks for primary storage, seem to be very different from the Euclidean point of view, but the method identified a similarity in their global trends.

On the other hand, Clusters 5, 6, and 12 (Figures 9a,b, and 12c) represent some examples of communities populated by heterogeneous physical parameters recorded in the same functional area.

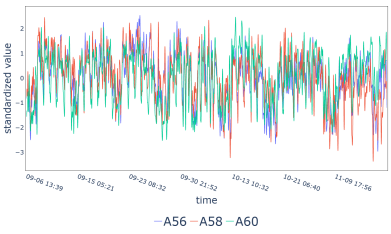
Finally, a particular interest derives from the hidden relationships identified between parameters characteristic of different functional areas. Examples are Cluster 3 (Figure 9a), which includes temperatures of cylinder and exhaust; Cluster 4 (Figure 9b), which groups together temperatures referred to the engine external casing, the engine auxiliaries, heat recovery, and fuel pre-heating systems and the inlet fuel; and Cluster 8 (Figure 11a), which is composed by frequencies and voltages related to both the generator and the grid.

After the identification of clusters, exploratory network analysis was used to render a graphical representation of their degree of similarity (the higher the similarity between nodes, the smaller their spatial distance), thus improving the cluster visualization. The Frushterman–Reingold layout applied to the similarity graph $C$, after edge pruning, provided the results shown in Figure 13.

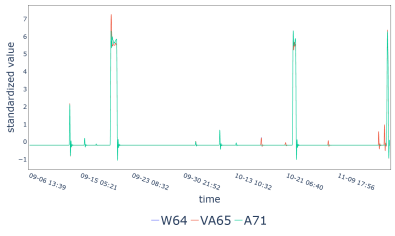
The force-directed layout gives the evidence of a central core of strongly connected parameters, which includes, respectively, most of the fuel temperatures in the storage area (Cluster 1), all the temperatures of cylinders and exhaust (Cluster 3), all the process low temperature parameters (Cluster 7), and most of the generator and grid parameters (Clusters 8–11).




(**a**) Cluster 10




(**b**) Cluster 11




(**c**) Cluster 12

**Figure 12.** Generator and Grid electric powers and currents: (**a**) the trends of the time series included in Cluster 10, i.e., generator parameters such as active power (W50) and reactive power (VA51); (**b**) the trends of the time series included in Cluster 11, i.e., generator electric currents Phase 1 (A56), Phase 2 (A58), and Phase 3 (A60); and (**c**) the trends of the time series included in Cluster 12, i.e., grid parameters such as active power (W64), reactive power (VA65), and electric current Phase 2 (A71).

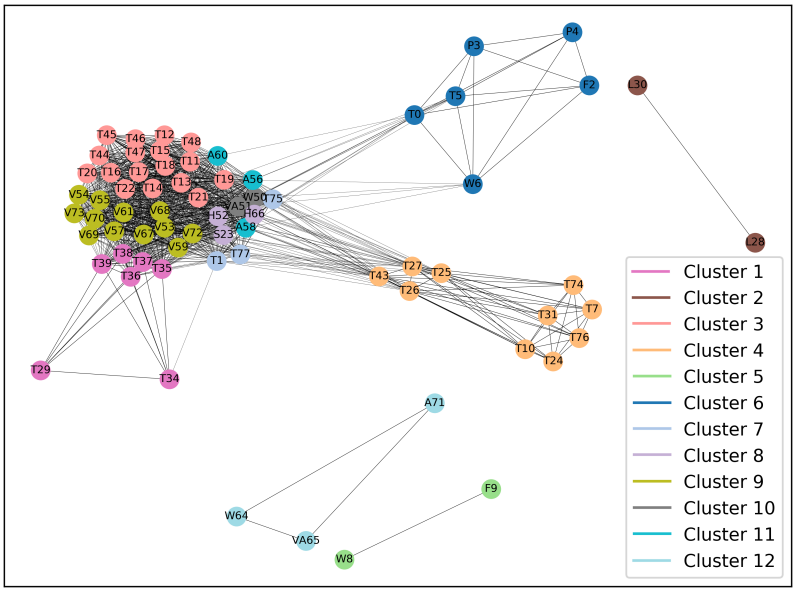



**Figure 13.** Frushterman–Reingold layout applied to the similarity graph after edge pruning.

Notably, only two parameters of Cluster 3 are outside the central core, namely T29 and T34, measuring, respectively, the fuel temperature in the primary storage and in the tank 2 (the latter being a backup tank). It is also possible to notice how the temperatures of engine cooling water (T25–T27) and lube oil (T43) subsystems represent a key group in bridging the central core to the other variables of Cluster 4. Similarly, the steam parameters in the high temperature heat recovery (Cluster 6), although not directly included in the central core, appear to be strictly connected to it. As expected, no correlation is active among the fuel levels inside the tanks (Cluster 2), the power and flow rate of the hot water at the boiler inlet (Cluster 5), the grid power and currents (Cluster 12), and the rest of the network. To improve the interpretation of the results by adding quantitative information to the exploratory analysis, we calculated the cumulative percentage distribution of the average degree centrality of each cluster (see Figure 14).

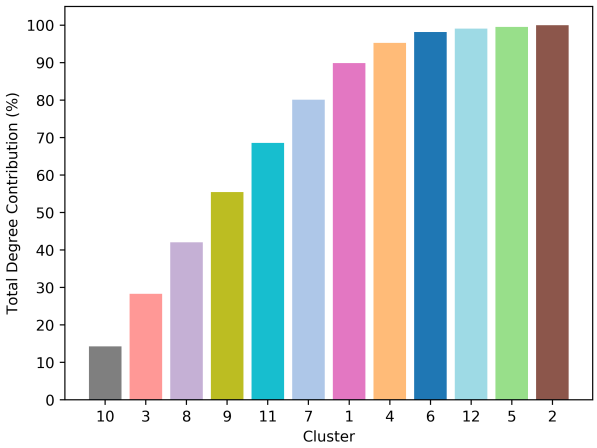



**Figure 14.** Cumulative percentage distribution of the average degree centrality of clusters.

The bar chart in Figure 14 attributes a specific ranking to the clusters according to their average contribute to the degree centrality of the network. Overall, the results confirm the considerations made so far in relation to the core communities (Clusters 1, 3, 7, and 8–11), to the boundary communities (Clusters 4 and 6), and to the communities totally unrelated to the network (Clusters 2, 5, and 12).

As for the communities included in the central core, it is possible to obtain a distinction between the roles played in the network. In detail, Cluster 10, which groups engine speed and generator and grid frequencies together, is the most influential on the control and stability of the global systems, followed by Cluster 3, which includes cylinder temperatures and exhaust gases.

Finally, after clusters identification and analysis, FSS was performed by selecting in each cluster the representative signal as the one with the highest degree contribution in its group. Table 3 shows the selected variables associated to each cluster, together with their degree centrality in the similarity graph, and their share contribution to the sum of the degree centralities within the reference cluster.

The representative parameters shown in the table are visually confirmed by the force-directed layout in Figure 13. For example, variable T0 (condense temperature) appears to be the most influential node of Cluster 6 (process high temperature user parameters), having a high number of connections not only with variables of its own cluster, but also with those belonging to the central core of strongly connected signals. Another example is the parameter T43 (oil temperature) with respect to Cluster 4 (parameters strictly related to the engine).

As reported in the case study, the data matrix considered as input for the analysis has 30,240 $\times$ 78 dimensions. After the application of the proposed method, by considering the 12 representative cluster variables, listed in Table 3, together with the 8 independent signals shown in Figure 6, we obtained a final data matrix of size 30,240 $\times$ 20, thus reducing the dimensionality by 74.4%.

**Table 3.** Representative signals chosen for every cluster according to their node degree centrality.

| Cluster ID | Number of Elements | Most Representative Variable ID | Absolute Degree Value | Within Cluster Degree Contribution |
|---|---|---|---|---|
| Cluster 1 | 7 | T38 | 29.44 | 19.51% |
| Cluster 2 | 2 | L30 | 1.00 | 50.00% |
| Cluster 3 | 17 | T19 | 34.62 | 6.61% |
| Cluster 4 | 10 | T43 | 18.20 | 15.34% |
| Cluster 5 | 2 | W8 | 1.00 | 50.00% |
| Cluster 6 | 6 | T0 | 9.03 | 23.50% |
| Cluster 7 | 3 | T75 | 25.83 | 33.96% |
| Cluster 8 | 3 | H66 | 31.06 | 34.18% |
| Cluster 9 | 12 | V72 | 30.91 | 8.70% |
| Cluster 10 | 2 | VA51 | 31.67 | 50.30% |
| Cluster 11 | 3 | A56 | 30.67 | 35.31% |
| Cluster 12 | 3 | W64 | 2.00 | 33.33% |

*Performance Metrics*

An exhaustive evaluation of the proposed method can be obtained by appropriate measures of clustering partitioning and FSS information content.

The lack of ground truth data in the present condition monitoring application precluded the evaluation of the clustering results through classical external indices. In addition, since we used a modularity-based method for the community detection, modularity was identified as the most appropriate metric of the final clustering. A first evaluation of the clustering results was performed using the modularity measure, which quantifies the goodness of the communities on a scale that goes from $-1$ to 1. In particular, we obtained a modularity index of 0.72, representative of good quality results.

Since the proposed approach belongs to the category of unsupervised FSS methods, the final evaluation was performed in terms of redundancy reduction ratio (RRR) and information gain ratio (IGR), defined, respectively, in Equations (2) and (4).

The proposed method was compared with standard approaches for time series clustering (see Table 4). In particular, a *raw data-based* method was considered, which uses the Euclidean distance as time series similarity measure and a partitioning clustering, namely K-Means, for grouping variables. In addition, we included a *feature-based* method in the comparison, which involves the extraction of statistical parameters characteristic of the time series (i.e., average, median, standard deviation, skewness, and kurtosis) and the subsequent application of the K-Means algorithm for clustering.

**Table 4.** Comparison of the FSS performances between the proposed approach and two standard methods: a *raw data-based* method and a *feature-based* one. The evaluation was performed by considering the redundancy reduction ration (RRR) and information gain ratio (IGR) indices.

| Method | Optimal Number of Clusters | RRR | IGR |
|---|---|---|---|
| Proposed approach | 12 | 29.05% | 10.60% |
| *Raw-data* based | 12 | 9.52% | 8.39% |
| *Feature* based | 10 | 20.96% | 7.90% |

Table 4 shows that the time series clustering approach seems to be particularly efficient in terms of FSS, allowing a total redundancy reduction of 29.05% in the starting dataset by obtaining, at the same time, a global information gain of 10.60%.

It is also interesting to note that both performance metrics are better than those obtained with the standard approaches considered. In particular, the proposed method outperforms the *raw data-based* clustering approach in terms of both RRR and IGR indices, with an overall performance improvement of 19.53% and 2.21%, respectively. Looking at the results obtained with the *feature-based* method, also in this case the proposed approach provides better results with an increase of 8.09% and 2.70% for the RRR and IGR indices, respectively.

## 6. Conclusions

With the advent of Industry 4.0, the increasing availability of sensor data is leading to a rapid development of models and techniques able to deal with it. In particular, data-driven AI models are becoming essential to conduct the analysis of complex systems based on large data streams.

State-of-the-art models fail when dealing with overfitting in the data and suffer from performance loss when variables are highly correlated between each other. Many FSS methods have been introduced to address these problems. Notably, it has been demonstrated that clustering-based methods for unsupervised FSS outperform traditional approaches in terms of accuracy.

The complexity of nonlinear dynamics associated to data streams from sensor networks make standard clustering methods unsuitable in this context. For these reasons, in this paper, we propose a new clustering approach for time series useful for unsupervised FSS, exploiting different complex network tools. In particular, we mapped time series segments in the network domain through natural weighted visibility graphs, extracted their degree sequences as feature vectors to define a similarity matrix between signals, used a community detection algorithm to identify clusters of similar time series, and selected a representative parameter for each of them based on the variable degree contributions.

The analysis of the results highlights two advantages deriving from the proposed method. The first is the ability to group together both homogeneous and heterogeneous physical parameters even when related to different functional areas of the system. This is obtained by capturing time series similarities not necessarily linked to signal Euclidean distance. In the FSS perspective, the approach, by considering 12 representative variables for the identified clusters and 8 independent signals that were not clustered, reduced the dimensionality of the dataset by 74.4%.

Second, as an additional advantage with respect to FSS purposes, the method allows discovering hidden relationships between system components enriching the information content about the signal roles within the network.

Since the construction of a natural weighted visibility graph has time complexity $O(L^2)$, being $L$ the number of samples in a time series interval, the proposed approach was intended as an offline filtering tool. In particular, being the visibility graph the bottleneck of the algorithm, the global time complexity is in the order of $O(TL^2)$, where $T$ is the number of consecutive non-overlapping segments. Running the algorithm on a dataset of 11 months with time windows of 24 h took approximately 15 min. The idea is to consider the whole dataset at disposal in order to identify the overall most relevant signals, by averaging the contributions of all intervals. Thus, the resulting reduction in the dimensionality of data streams opens the possibility to simplify the condition monitoring system and its data.

If, instead, a real time tool for FSS or time series clustering is of interest, it is possible to imagine the integration of the proposed algorithm into sensor network now-casting models, e.g., on a sliding window of 24 h the algorithm runs in less than 3 s.

**Author Contributions:** Conceptualization, F.B. and A.C.; methodology, F.B. and E.S.M.; software, E.S.M.; validation, F.B. and E.S.M.; formal analysis, F.B. and E.S.M.; writing—original draft preparation, all authors; writing—review and editing, all authors; and supervision, A.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

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

## References

1. Mohammadi, M.; Al-Fuqaha, A.; Sorour, S.; Guizani, M. Deep Learning for IoT Big Data and Streaming Analytics: A Survey. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2923–2960. [CrossRef]
2. Asghari, P.; Rahmani, A.M.; S.; Javadi, H.H. Internet of Things applications: A systematic review. *Comput. Netw.* **2019**, *148*, 241–261. [CrossRef]
3. Imkamp, D.; Berthold, J.; Heizmann, M.; Kniel, K.; Manske, E.; Peterek, M.; Schmitt, R.; Seidler, J.; Sommer, K.D. Challenges and trends in manufacturing measurement technology—The "Industrie 4.0" concept. *J. Sensors Sensor Syst.* **2016**, *5*, 325. [CrossRef]
4. Lu, Y. Industry 4.0: A survey on technologies, applications and open research issues. *J. Ind. Inf. Integr.* **2017**, *6*, 1–10.
5. Hayes-Roth, B.; Washington, R.; Hewett, R.; Hewett, M.; Seiver, A. Intelligent Monitoring and Control. In Proceedings of the IJCAI, Detroit, MI, USA, 20–25 August 1989; Volume 89, pp. 243–249.
6. Verleysen, M.; François, D. The curse of dimensionality in data mining and time series prediction. In *International Work-Conference on Artificial Neural Networks*; Springer: New York, NY, USA, 2005; pp. 758–770.
7. Uraikul, V.; Chan, C.W.; Tontiwachwuthikul, P. Artificial intelligence for monitoring and supervisory control of process systems. *Eng. Appl. Artif. Intell.* **2007**, *20*, 115–131.
8. Tian, H.X.; Liu, X.J.; Han, M. An outliers detection method of time series data for soft sensor modeling. In Proceedings of the 2016 Chinese Control and Decision Conference (CCDC), Yinchuan, China, 28–30 May 2016; pp. 3918–3922.
9. Kaiser, J. Dealing with missing values in data. *J. Syst. Integr.* **2014**, *5*, 42–51. [CrossRef]
10. Liu, R.; Yang, B.; Zio, E.; Chen, X. Artificial intelligence for fault diagnosis of rotating machinery: A review. *Mech. Syst. Signal Process.* **2018**, *108*, 33–47. [CrossRef]
11. Ruiz-Sarmiento, J.R.; Monroy, J.; Moreno, F.A.; Galindo, C.; Bonelo, J.M.; Gonzalez-Jimenez, J. A predictive model for the maintenance of industrial machinery in the context of Industry 4.0. *Eng. Appl. Artif. Intell.* **2020**, *87*, 103289. [CrossRef]
12. Ansari, F.; Glawar, R.; Nemeth, T. PriMa: A prescriptive maintenance model for cyber-physical production systems. *Int. J. Comput. Integr. Manuf.* **2019**, *32*, 482–503.
13. Jin, X.; Shao, J.; Zhang, X.; An, W.; Malekian, R. Modeling of nonlinear system based on deep learning framework. *Nonlinear Dyn.* **2016**, *84*, 1327–1340. [CrossRef]
14. Chormunge, S.; Jena, S. Correlation based feature selection with clustering for high dimensional data. *J. Electr. Syst. Inf. Technol.* **2018**, *5*, 542–549. [CrossRef]
15. Frolik, J.; Abdelrahman, M. Synthesis of quasi-redundant sensor data: A probabilistic approach. In Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No. 00CH36334), Chicago, IL, USA, 28–30 June 2000; Volume 4, pp. 2917–2921.
16. Acid, S.; De Campos, L.M.; Fernández, M. Minimum redundancy maximum relevancy versus score-based methods for learning Markov boundaries. In Proceedings of the 2011 11th International Conference on Intelligent Systems Design and Applications, Cordoba, Spain, 22–24 November 2011; pp. 619–623.
17. Chandrashekar, G.; Sahin, F. A survey on feature selection methods. *Comput. Electr. Eng.* **2014**, *40*, 16–28. [CrossRef]
18. You, D.; Wu, X.; Shen, L.; He, Y.; Yuan, X.; Chen, Z.; Deng, S.; Ma, C. Online Streaming Feature Selection via Conditional Independence. *Appl. Sci.* **2018**, *8*, 2548. [CrossRef]

19. Pal, S.K.; Mitra, P. *Pattern Recognition Algorithms for Data Mining: Scalability, Knowledge Discovery, and Soft Granular Computing*; Chapman & Hall, Ltd.: London, UK, 2004.
20. Song, Q.; Ni, J.; Wang, G. A fast clustering-based feature subset selection algorithm for high-dimensional data. *IEEE Trans. Knowl. Data Eng.* **2011**, *25*, 1–14. [CrossRef]
21. Baker, L.D.; McCallum, A.K. Distributional clustering of words for text classification. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, 24–28 August 1998; pp. 96–103.
22. Slonim, N.; Tishby, N. The power of word clusters for text classification. In Proceedings of the 23rd European Colloquium on Information Retrieval Research, Darmstadt, Germany, 4–6 April 2001; Volume 1, p. 200.
23. Zou, Y.; Donner, R.; Marwan, N.; Donges, J.; Kurths, J. Complex network approaches to nonlinear time series analysis. *Phys. Rep.* **2018**, *787*, 1–97. [CrossRef]
24. Lacasa, L.; Luque, B.; Ballesteros, F.; Luque, J.; Nuño, J.C. From time series to complex networks: The visibility graph. *Proc. Natl. Acad. Sci. USA* **2008**, *105*, 4972–4975.
25. Koschützki, D.; Schreiber, F. Centrality analysis methods for biological networks and their application to gene regulatory networks. *Gene Regul. Syst. Biol.* **2008**, *2*, GRSB–S702.
26. Blondel, V.D.; Guillaume, J.L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**, *2008*, P10008. [CrossRef]
27. Lal, T.N.; Chapelle, O.; Weston, J.; Elisseeff, A. Embedded methods. In *Feature Extraction*; Springer: New York, NY, USA, 2006; pp. 137–165.
28. Kohavi, R.; John, G.H. Wrappers for feature subset selection. *Artif. Intell.* **1997**, *97*, 273–324. [CrossRef]
29. Sánchez-Maro no, N.; Alonso-Betanzos, A.; Tombilla-Sanromán, M. Filter methods for feature selection—A comparative study. In Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning, Birmingham, UK, 16–19 December 2007; pp. 178–187.
30. Brahim, A.B.; Limam, M. A hybrid feature selection method based on instance learning and cooperative subset search. *Pattern Recognit. Lett.* **2016**, *69*, 28–34.
31. Hauskrecht, M.; Pelikan, R.; Valko, M.; Lyons-Weiler, J. Feature selection and dimensionality reduction in genomics and proteomics. In *Fundamentals of Data Mining in Genomics and Proteomics*; Springer: New York, NY, USA, 2007; pp. 149–172.
32. Sanche, R.; Lonergan, K. Variable reduction for predictive modeling with clustering. In *Casualty Actuarial Society Forum*; Casualty Actuarial Society: Arlington, VA, USA, 2006; pp. 89–100.
33. Fritzke, B. Unsupervised clustering with growing cell structures. In Proceedings of the IJCNN-91-Seattle International Joint Conference on Neural Networks, Seattle, WA, USA, 8–12 July 1991; Volume 2, pp. 531–536.
34. Clarkson, B.; Pentland, A. Unsupervised clustering of ambulatory audio and video. In Proceedings of the 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing, Phoenix, AZ, USA, 15–19 March 1999; Volume 6, pp. 3037–3040.
35. Popat, S.K.; Emmanuel, M. Review and comparative study of clustering techniques. *Int. J. Comput. Sci. Inf. Technol.* **2014**, *5*, 805–812.
36. Fujita, A.; Sato, J.; Demasi, M.; Sogayar, M.; Ferreira, C.; Miyano, S. Comparing Pearson, Spearman and Hoeffding's D measure for gene expression association analysis. *J. Bioinf. Comput. Biol.* **2009**, *7*, 663–84. [CrossRef] [PubMed]
37. Iglesias, F.; Kastner, W. Analysis of Similarity Measures in Times Series Clustering for the Discovery of Building Energy Patterns. *Energies* **2013**, *6*, 579–597. [CrossRef]
38. Jing, L.; Ng, M.; Huang, J. An Entropy Weighting K-Means Algorithm for Subspace Clustering of High-Dimensional Sparse Data. *IEEE Trans. Knowl. Data Eng.* **2007**, *19*, 1026–1041. [CrossRef]
39. Huang, X.; Ye, Y.; Zhang, H. Extensions of Kmeans-Type Algorithms: A New Clustering Framework by Integrating Intracluster Compactness and Intercluster Separation. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *25*, 1433–1446. [CrossRef] [PubMed]
40. Baragona, R. A simulation study on clustering time series with metaheuristic methods. *Quad. Stat.* **2001**, *3*, 1–26.
41. Ramoni, M.; Sebastiani, P.; Cohen, P.R. Multivariate Clustering by Dynamics. In Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, Austin, TX, USA, 30 July–3 August 2000; pp. 633–638.
42. Tran, D.; Wagner, M. Fuzzy c-means clustering-based speaker verification. In *Proceedings of the AFSS International Conference on Fuzzy Systems*; Springer: New York, NY, USA, 2002; pp. 318–324.

43. Bandara, K.; Bergmeir, C.; Smyl, S. Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. *Exp. Syst. Appl.* **2020**, *140*, 112896. [CrossRef]
44. Shaw, C.; King, G. Using cluster analysis to classify time series. *Phys. D Nonlinear Phenom.* **1992**, *58*, 288–298. [CrossRef]
45. Vlachos, M.; Lin, J.; Keogh, E.; Gunopulos, D. A Wavelet-Based Anytime Algorithm for K-Means Clustering of Time Series. In Proceedings of the Workshop on Clustering High Dimensionality Data and Its Applications, San Francisco, CA, USA, 3 May 2003; pp. 23–30.
46. Kavitha, V.; Punithavalli, M. Clustering Time Series Data Stream—A Literature Survey. *arXiv* **2010**, arXiv:1005.4270.
47. Rani, S.; Sikka, G. Recent Techniques of Clustering of Time Series Data: A Survey. *Int. J. Comput. Appl.* **2012**, *52*, 1–9. [CrossRef]
48. Aghabozorgi, S.; Shirkhorshidi, A.S.; Wah, T. Time-series clustering—A decade review. *Inf. Syst.* **2015**, *53*, 16–38. [CrossRef]
49. Zanin, M.; Papo, D.; Sousa, P.A.; Menasalvas, E.; Nicchi, A.; Kubik, E.; Boccaletti, S. Combining complex networks and data mining: why and how. *Phys. Rep.* **2016**, *635*, 1–44. [CrossRef]
50. Ferreira, L.; Zhao, L. Time Series Clustering via Community Detection in Networks. *Inf. Sci.* **2015**, *326*. [CrossRef]
51. Zhang, X.; Liu, J.; Du, Y.; Lu, T. A novel clustering method on time series data. *Exp. Syst. Appl.* **2011**, *38*, 11891–11900. [CrossRef]
52. Shannon, C.E. A mathematical theory of communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
53. Lozano-Pérez, T.; Wesley, M.A. An Algorithm for Planning Collision-Free Paths among Polyhedral Obstacles. *Commun. ACM* **1979**, *22*, 560–570. [CrossRef]
54. Luque, B.; Lacasa, L.; Ballesteros, F.; Luque, J. Horizontal visibility graphs: Exact results for random time series. *Phys. Rev. E* **2009**, *80*, 046103. [CrossRef]
55. Lacasa, L.; Toral, R. Description of stochastic and chaotic series using visibility graphs. *Phys. Rev. E* **2010**, *82*, 036120. [CrossRef]
56. Bianchi, F.M.; Livi, L.; Alippi, C.; Jenssen, R. Multiplex visibility graphs to investigate recurrent neural network dynamics. *Sci. Rep.* **2017**, *7*, 1–13. [CrossRef]
57. Bonacich, P. Some unique properties of eigenvector centrality. *Soc. Netw.* **2007**, *29*, 555–564. [CrossRef]
58. Freeman, L.C. Centrality in social networks conceptual clarification. *Soc. Netw.* **1978**, *1*, 215–239.
59. Scott, J. Social Network Analysis. *Sociology* **1988**, *22*, 109–127. [CrossRef]
60. Freeman, L. The development of social network analysis. *Study Sociol. Sci.* **2004**, *1*, 687.
61. Rice, S.A. The identification of blocs in small political bodies. *Am. Pol. Sci. Rev.* **1927**, *21*, 619–627. [CrossRef]
62. Weiss, R.S.; Jacobson, E. A method for the analysis of the structure of complex organizations. *Am. Sociol. Rev.* **1955**, *20*, 661–668. [CrossRef]
63. Homans, G.C. *The Human Group*; Routledge: London, UK, 2013.
64. Fortunato, S. Community detection in graphs. *Phys. Rep.* **2010**, *486*, 75–174.
65. Kernighan, B.W.; Lin, S. An Efficient Heuristic Procedure for Partitioning Graphs. *Bell Syst. Tech. J.* **1970**, *49*, 291–307. [CrossRef]
66. Barnes, E. An algorithm for partitioning the nodes of a graph. In Proceedings of the 1981 20th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes, San Diego, CA, USA, 16–18 December 1981; pp. 303–304.
67. Lloyd, S.P. Least Squares Quantization in PCM. *IEEE Trans. Inf. Theory* **1982**, *28*, 129–137. [CrossRef]
68. Friedman, J.; Hastie, T.; Tibshirani, R. *The Elements of Statistical Learning*; Springer: New York, NY, USA, 2001; Volume 1.
69. Newman, M.E. Fast algorithm for detecting community structure in networks. *Phys. Rev. E* **2004**, *69*, 066133. [CrossRef]
70. Guimera, R.; Sales-Pardo, M.; Amaral, L.A.N. Modularity from fluctuations in random graphs and complex networks. *Phys. Rev. E* **2004**, *70*, 025101.
71. Duch, J.; Arenas, A. Community detection in complex networks using extremal optimization. *Phys. Rev. E* **2005**, *72*, 027104. [CrossRef] [PubMed]
72. Newman, M.E.; Girvan, M. Finding and evaluating community structure in networks. *Phys. Rev. E* **2004**, *69*, 026113.
73. Donath, W.E.; Hoffman, A.J. Lower bounds for the partitioning of graphs. In *Selected Papers Of Alan J Hoffman: With Commentary*; World Scientific: Singapore, 2003; pp. 437–442.
74. Hastings, M.B. Community detection as an inference problem. *Phys. Rev. E* **2006**, *74*, 035102.

75. Newman, M.E.; Leicht, E.A. Mixture models and exploratory analysis in networks. *Proc. Natl. Acad. Sci. USA* **2007**, *104*, 9564–9569. [CrossRef] [PubMed]
76. Shannon, P.T.; Grimes, M.; Kutlu, B.; Bot, J.J.; Galas, D.J. RCytoscape: tools for exploratory network analysis. *BMC Bioinf.* **2013**, *14*, 217. [CrossRef] [PubMed]
77. Sakkalis, V. Review of advanced techniques for the estimation of brain connectivity measured with EEG/MEG. *Comput. Biol. Med.* **2011**, *41*, 1110–1117. [CrossRef]
78. Fruchterman, T.M.; Reingold, E.M. Graph drawing by force-directed placement. *Softw. Pract. Exp.* **1991**, *21*, 1129–1164. [CrossRef]
79. Liu, X.; Cheng, H.M.; Zhang, Z.Y. Evaluation of community detection methods. *IEEE Trans. Knowl. Data Eng.* **2019**, in press.
80. Manning, C.D.; Raghavan, P.; Schütze, H. *Introduction to Information Retrieval*; Cambridge University Press: Cambridge, UK, 2008.
81. Rand, W.M. Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.* **1971**, *66*, 846–850.
82. Danon, L.; Diaz-Guilera, A.; Duch, J.; Arenas, A. Comparing community structure identification. *J. Stat. Mech. Theory Exp.* **2005**, *2005*, P09008. [CrossRef]
83. Ahn, Y.Y.; Bagrow, J.P.; Lehmann, S. Link communities reveal multiscale complexity in networks. *Nature* **2010**, *466*, 761–764. [CrossRef]
84. Luque, B.; Lacasa, L. Canonical Horizontal Visibility Graphs are uniquely determined by their degree sequence. *Eur. Phys. J. Spec. Top.* **2016**, *226*. [CrossRef]
85. Corsini, A.; Bonacina, F.; Feudo, S.; Marchegiani, A.; Venturini, P. Internal Combustion Engine sensor network analysis using graph modeling. *Energy Procedia* **2017**, *126*, 907–914. [CrossRef]
86. Van Rossum, G.; Drake, F.L., Jr. *Python Reference Manual*; Centrum voor Wiskunde en Informatica Amsterdam: Amsterdam, The Netherlands, 1995.
87. Oliphant, T.E. *A Guide to NumPy*; Trelgol Publishing USA: Spanish Fork, UT, USA, 2006; Volume 1.
88. Hagberg, A.; Swart, P.; Chult, D. Exploring Network Structure, Dynamics, and Function Using NetworkX. In Proceedings of the 7th Python in Science Conference, Pasadena, CA, USA, 19–24 August 2008; pp. 11–15.