A ROBUSTNESS WITH DECREMENT

In this section, we will illustrate the results obtained by the cEnsemble algorithm by decreasing the number of clusters required by the dataset.

Tables 9 and reftab:DecOne show the results of two experiments where the original number of classes was decreased by 2 and 1, respectively. Obviously, not all datasets can be used since some do not have an adequate number of required clusters allowing for non-trivial decrements (e.g. down to 1 cluster).

Decrement by two. In this experiment, we focus on varying the number of initial classes as provided by the ground truth. In the main body of the paper, we address the case of increments whereas we tackle the cases of decrements here. Not all the datasets allowed this decrement (which depends on the initial number of classes) thus we concentrated on those datasets for which this number can be decremented by two. Among all the UCR datasets, it turns out that only *OliveOil*, *Plaid*, *AsphaltObstacles* and *GesturePebble* are used in this experiment, as shown in Table 9.

Dataset	Orig. Class	Test. Class	cEnsemble	Rand
ScreenType	3	2	0,01	0
UMD	3	2	0,32	0,05
ArrowHead	3	2	0,2	0,15
Meat	3	2	0,4	0,35
OliveOil	4	3	0,16	0,13
Plaid	11	10	0,35	0,2
Asphalt Obstacles	4	3	0,38	0,2
GesturePebble	6	5	0,15	0,05

Table 8: Decrement by one

One can notice that across all datasets cEnsemble achieves better results comparing with the baseline, i.e. cEnsemble with randomized features. Indeed, by averaging the results obtained on all 4 datasets, the average AMI is increased of 0.13 compared to the baseline.

Dataset	Orig. Class	Test. Class	cEnsemble	Rand
OliveOil	4	2	0,16	0,03
Plaid	11	9	0,25	0,04
Asphalt Obstacles	4	2	0,29	0,15
GesturePebble	6	4	0,12	0,03

Table 9: Decrement by two

Decrement by one. Finally, in this experiment, we decrease by one the number of original classes and study the behavior of our method. The datasets used in this experiment are those that require three or more classes as shown in Table 8.

Once again, cEnsemble achieves better results across all the tested datasets. In this case the average increase of the AMI for the results obtained with this version of cEnsemble is 0.12 higher than the baseline.

B RESULTS ON DTW VERSUS FEATURES

Table 6 shows the results obtained by replacing the distances between the features, as shown in Figure 4 with the distance computed by the DTW directly on the time series.

The DTW column shows the results obtained by cEnsemble when replacing the distances among features with the DTW.

The obtained results showed that, 14 datasets (out of 15) have a worse behavior if DTW is adopted. Indeed, only in one dataset (*OliveOil*) the usage of DTW outperforms the usage of the features. On average, we have a difference in terms of AMI of 0.31. Hence, the results of this ablation test confirms the importance of using the distances between features instead of merely using distances among raw data.

Dataset	cEnsemble	DTW
ScreenType	0,02	0,01
UMD	0,3	0,27
TwoLeadECG	0,88	0
ECG200	0,32	0,08
Computers	0,09	0
Coffee	0,8	0,01
GunPoint	0,56	0
Arrowhead	0,28	0,2
ItalyPowerDemand	0,57	0
Meat	0,4	0
OliveOil	0,15	0,32
Plaid	0,35	0,01
Asphalt Regularity	0,54	0
Asphalt Obstacles	0,38	0,27
Gesture Pebble	0,25	0,02

Table 10: DTW Ablation Test

C RESULTS ON P-VALUE VERSUS RANDOM.

Table 11 shows the results removing the ordering of the features based on their relevance. We repeated this test 5 times and we averaged the results.

The column Rand represents the average of the results obtained by cEnsemble using Random Features.

The results in Table 6 show that the performance of the algorithm drastically deteriorates for the majority of the datasets (13 out of a total of 15 datasets) if random features are employed. Indeed, computing the average overall the results obtained by the two experiments, we have a difference of 0.35 in terms of AMI.

Therefore, the ordering of the features based on their relevance turns out to be indispensable in order to achieve good results.

Dataset	cEnsemble	Rand
ScreenType	0,02	0
UMD	0,3	0,01
TwoLeadECG	0,88	0,02
ECG200	0,32	0,06
Computers	0,09	0
Coffee	0,8	0,12
GunPoint	0,56	0
Arrowhead	0,28	0,02
ItalyPowerDemand	0,57	0
Meat	0,4	0,16
OliveOil	0,15	0,2
Plaid	0,35	0,11
Asphalt Regularity	0,54	0
Asphalt Obstacles	0,38	0,02
Gesture Pebble	0,25	0

Table 11: Random Features Ablation Test

D RESULTS ON K-MEANS VERSUS cEnsemble.

In Table 7, we show the results obtained capturing the global relationship among the raw time series samples through graph encoding and of the subsequent application of the Community Detection.

The column k-Means shows the results obtained by k-Means among the features selected by each dataset used in our experiment. We have highlighted in bold the best results obtained between k-Means and cEnsemble for each dataset.

The results show that cEnsemble outperforms k-Means in the majority of the cases. Indeed, there are only four datasets for which k-Means shows slightly better results, namely UMD, $Meat$, $Coffee$ and $OliveOil$. On average, the results obtained by cEnsemble outperforms the results obtained by k-Means of 0,08 in terms of AMI.

Therefore, capturing the global relationships among the various time series allows to achieve better results in terms of performance.

Dataset	k-Means	cEnsemble
ScreenType	0,01	**0,02**
UMD	**0,42**	0,33
TwoLeadECG	0,75	**0,88**
ECG200	0,16	**0,33**
Computers	0	**0,09**
Coffee	**0,9**	0,89
GunPoint	0,26	**0,56**
ArrowHead	0,24	**0,3**
ItalyPowerDemand	0,51	**0,57**
Meat	**0,48**	0,42
OliveOil	**0,35**	0,18
Plaid	0,03	**0,35**
Ashpalt Regularity	0,35	**0,54**
Asphalt Obstacles	0,37	**0,38**
Gesture Pebble	0,16	**0,25**

Table 12: k-Means Ablation Test results

E DETAILED STUDY ABOUT CHRONIC AND ACUTE DIALYSIS.

As described in section 5.3, patients under chronic dialysis need to go regularly at the hospital to cure their kidney chronic problem. Chronic kidney diseases can be caused by diabetes, high blood pressure, glomerulonephritis (which can be caused by ANCA-associated vasculitis or lupus), and polycystic kidney disease.

Concerning instead acute kidney injury, it is an abrupt loss of kidney function that develops within 7 days. They are often caused by an inflammatory process. Thus, ANCA-associated vasculitis and lupus, which are autoimmune diseases, can also cause an acute kidney injury. Basically, an acute kidney injury occurs when kidneys are exposed to something harmful, or which is considered harmful by our system. It can be pathogens, antibodies (ANCA antibodies for examples), antibiotics, bad chemical substances. An Acute kidney disease can cause a future chronic kidney disease.

In the Figure 9 we show the signals of two patients treated with Acute Dialysis (a) and Chronic Dialysis (b) extracted from our results in Section 5.3.In the patient treated with an Acute Dialysis we have just one single drop of GFR Signal. Once the patient went through dialysis, the signal tends to stabilize. Instead, the patient with Chronic Dialysis has multiple drops in the GFR signal, therefore indicating that the patient has remained on dialysis for a longer time.

Algorithm Parameters	
Th	0.8
$Perc.ofClass$	0.2
$Community\ Detection\ Algorithm$	$Greedy\ Modularity$
$Number\ of\ Features$	20
$Metric\ Pairwise\ Distance$	$Euclidean$
$Tolerance\ kMedoid$	0.000001

Table 13: Hyperparameters Algorithm of Clustering

F REPRODUCIBILITY

F.1 Hardware and Software

All the experiments have been executed on a Server running Linux with 64GB of RAM, Intel Xeon CPU Skylake,IBRS @ 2.6GHz, 12 CPUs.

The code is implemented using Python 3.5 and is available at the fommowing url: https://github.com/DonaTProject/cEnsemble.

The feature extraction and the feature have been implemented with TSFresh 0.13. For creating the graph and the application of community detection, NetworkX 2.2 has been used, while for the clustering on the Co-Occurrence Matrix we have used the K-Medoid algorithm as in Pyclustering 0.9.2. Finally, for computing all the Cluster Quality Metrics, we have used Sklearn.

Our experiments are fully reproducible. In the following, we report the parameter settings.

F.2 Hyper-parameter Settings

The first hyper-parameter to set is the threshold, that represents the choice of 80% of the distances accepted for each feature. Greedy Modularity has been chosen as Community Detection algorithm for detecting the groups of nodes in the graph.

Another important parameter to choose is the maximum number of features that the algorithm has to use. This has been set to 20.

Another hyper-parameter is the choice of the percentage of the number of class that we want to use in order to select the features, for default we have setted to 20%.

In order to apply the clustering algorithm, we have computed the pairwise distance between the rows in the Co-Occurrence matrix, where we used as metric the Euclidean distance. For what concerns the K-Medoid algorithm, we have set as tolerance a value of 0.000001. Table 13 summarizes all the used parameters by our method for the sake of reproducibility.

F.3 Reproducibility of the Real-World Dataset

The real-life dataset describing GFR time series cannot be used for reproducibility due to data protection restrictions as enacted by the French hospital under the European GDPR regulations. For that reason, no reproducility guidelines for this dataset are included in the paper and in the git repository.