Time2Feat: learning interpretable representations for multivariate time series clustering

Angela Bonifati, Francesco Del Buono, Francesco Guerra, Donato Tiano

Time2Feat: Learning Interpretable Representations for Multivariate Time Series Clustering

Angela Bonifati
Lyon 1 University, Liris CNRS
angela.bonifati@univ-lyon1.fr

Francesco Del Buono
University of Modena and Reggio Emilia
francesco.delbuono@unimore.it

Francesco Guerra
University of Modena and Reggio Emilia
francesco.delbuono@unimore.it

Donato Tiano
Lyon 1 University, Liris CNRS
donato.tiano@univ-lyon1.fr

ABSTRACT

Clustering multivariate time series is a critical task in many real-world applications involving multiple signals and sensors. Existing systems aim to maximize effectiveness, efficiency and scalability, but fail to guarantee the interpretability of the results. This hinders their application in critical real scenarios where human comprehension of algorithmic behavior is required. This paper introduces Time2Feat, an end-to-end machine learning system for multivariate time series (MTS) clustering. The system relies on inter-signal and intra-signal interpretable features extracted from the time series. Then, a dimensionality reduction technique is applied to select a subset of features that retain most of the information, thus enhancing the interpretability of the results. In addition, domain experts can semi-supervise the process, by providing a small amount of MTS with a target cluster. This process further improves both accuracy and interpretability, narrowing down the number of features used by the clustering process. We demonstrate the effectiveness, interpretability, efficiency, and robustness of Time2Feat through experiments on eighteen benchmarking time series datasets, comparing them with state-of-the-art MTS clustering methods.

PVLDB Reference Format:
Angela Bonifati, Francesco Del Buono, Francesco Guerra, and Donato Tiano. Time2Feat: Learning Interpretable Representations for Multivariate Time Series Clustering . PVLDB, 16(2): 193 - 201, 2022.
doi:10.14778/3565816.3565822

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/softlab-unimore/time2feat.

1 INTRODUCTION

Time is a dimension that affects many aspects of real-world and digital-world phenomena. Physical environments, industrial machineries, healthcare monitoring, and economic and financial activities are a few examples of scenarios whose elements are regulated and evolve over time. Multivariate time series (MTS), i.e., datasets with more than one time-dependent signal, are widely used data artifacts for encoding collections of sequential observations over the temporal axis. MTS analytics includes supervised and unsupervised tasks, ranging from classification and clustering, to pattern discovery, forecasting, and exploration. Cluster analysis recently gained momentum in many applications and use cases where sensors collect massive amounts of data points.

Research on clustering time series has mainly focused on univariate time series (UTS), i.e., datasets with a single time-dependent variable, addressing issues related to the development of similarity measures to cluster the data (e.g., Dynamic Time Warping –DTW [11, 14, 22], K-Shape [30]). By opposite, research on MTS is still at an early stage. Proposals adapt clustering approaches designed for UTS to MTS after applying dimensionality reduction techniques. Examples of such techniques (CSPCA [19] and MC_2PCA [18]) are based on the Principal Component Analysis (PCA), which enables the conversion of a set of correlated features in the high dimensional space into a set of uncorrelated features in the low dimensional space. Nevertheless, the resulting clusters suffer from poor explainability as the original dimensions are lost. More recently, approaches based on Deep Neural Networks (DNNs) [49], and in particular Variational Autoencoders [13, 23] have been used to generate MTS encodings before applying clustering methods. Although these solutions might exhibit high performance, the resulting clusters are based on latent dimensions that remain unexplainable to the end-users. Limited interpretability can hamper the adoption of a clustering technique in critical real-world scenarios, when experts are asked to provide detailed and trustable explanations of their algorithms' recommendations [4, 27, 32, 35, 37, 40].

We introduce Time2Feat, an open-source system for MTS clustering that adopts an end-to-end semi-supervised feature-based pipeline. Features are automatically extracted from the signals composing the MTS. We exploit both *intra-signal features* characterizing the single signals of MTS, and *inter-signal features* measuring pairwise relatedness (in terms of similarity and correlation) of multiple signals by employing interpretable metrics. For the extraction of the intra-signal features, we rely on the `tsfresh` library [6, 7], which generates features describing the MTS signals according to statistical perspectives (Distribution, Correlation, Information Theory, etc.). Two dataset-dependent techniques are then introduced to select the most important features among the ones describing the MTS. The *unsupervised mode* is an entirely automatic approach based on Principal Features Analysis (PFA) [21]. The *semi-supervised mode* relies on user's annotations on small dataset samples to improve the selection process. Our extensive experimental analysis shows

Proceedings of the VLDB Endowment, Vol. 16, No. 2 ISSN 2150-8097.
doi:10.14778/3565816.3565822

that the number of features is reduced by two orders of magnitude while preserving the accuracy of the results. Finally, a clustering technique is applied to group the MTS.

Time2Feat is *scalable*: it computes efficiently clusters regardless of the main dimensions of the problem (i.e., number of MTS in the dataset, number of generated clusters, number of signals composing the MTS and length of the time series) as the experiments in Section 5.3.1 demonstrate. Time2Feat provides *interpretable features* for the cluster representations. The meaning and the properties associated to interpretable features have not been clearly agreed upon by the literature. As many other approaches, we state that the features are interpretable if humans can understand what they refer to [52], and we consider conciseness as one of the main properties to be satisfied by a set of interpretable features [16, 29, 31]. In Time2Feat, we rely on PFA [21] to select a concise number of features among the ones computed by `tsfresh`. These measures can be interpreted by experts who know the statistical measures used to summarize the time series values. Leveraging interpretable features, users can conduct an in-depth analysis to understand why MTS share the same cluster. They could, for instance, measure the value similarities among the features for MTS in the same cluster or apply techniques for evaluating feature importance during clustering generation.

The key contributions of this paper are summarized as follows:

An interpretable and efficient end-to-end clustering system for multivariate time series. Time2Feat provides a suite of clustering pipelines that leverages the MTS features to make the user aware of the results and internals of the clustering process while at the same time preserving the efficiency of the process.

A human-in-the-loop clustering system allowing for learning-based annotations. Time2Feat allows domain experts to provide small and controllable amounts of labels as input to the process in order to improve the accuracy of the clusters and tame the sizes of extracted feature sets. The feature reduction process allows enhancing the scalability and interpretability of the system further.

A comprehensive evaluation. We evaluate Time2Feat by benchmarking our pipeline against 8 state-of-the-art MTS clustering systems spanning a collection of 18 underlying datasets [3] and reporting their quality and computational performance.

The paper is organized as follows. Section 2 presents a motivating real-world scenario of the usage of the Time2Feat system. Section 3 describes the steps of our clustering pipeline, while Section 4 presents the implementation of Time2Feat. Section 5 presents our experimental setup on real-life and benchmarking data. Section 6 discusses the related work. Finally, Section 7 concludes our work.

2 MOTIVATING REAL-WORLD SCENARIO

The BasicMotions dataset is a real-world dataset belonging to the UEA multivariate time series classification archive [3]. This dataset describes four kinds of activity (i.e., playing badminton, running, standing, and walking) performed by students through two sensors (an accelerometer and a gyroscope) installed in their smartwatches. The sensors gather data in a three-dimensional space, thus producing three different signals (X, Y, Z). The overall dataset comprises 80 MTS, and each signal includes 100 recordings.

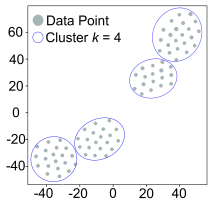
Suppose we are asked to analyze the dataset and no detail on the activities that MTS describes are provided to us. This lack of information frequently happens in business scenarios where trade secrets or simply the costs for labeling make it necessary to work with unlabeled datasets. Clustering is one of the main exploration techniques we can apply on unlabeled data.

	Intra-Signal						*Inter-Signal*		
	AccX		AccY		AccZ	GirX	AccX-AccY		AccX-AccZ
	AccX	...	PACF 9	...	...	...	L_1 dist.	Chebyshev	L_1 dist.
MTS_R	80.12	...	0.16	...	...	...	981.78	49.56	1115.71
MTS_B	139.75	...	0.26	...	...	...	1121.45	30.71	1423.28
MTS_S	0.09	...	-0.23	...	...	...	85.05	1.93	35.60
MTS_W	2.26	...	0.14	...	...	...	301.46	8.45	174.99
...	...	...	...	...	...	...	...	...	...

(a) Excerpt of the features (46) extracted in the unsupervised mode.

	Intra-Signal		*Inter-Signal*
	AccX	GirX	AccY-AccZ
	Variance	Quantile 0.3	L_1 dist.
MTS_R	80.12	-0.94	764.99
MTS_B	139.75	-3.57	1083.85
MTS_S	0.09	-0.14	97.05
MTS_W	2.26	-1.02	325.82
...	...	...	...

(b) The features (3) extracted in the semi-supervised mode.



(c) Clusters generated with the unsupervised mode.

Figure 1: Time2Feat **on the BasicMotion dataset.**

Generating clusters for MTS is a non-trivial task. From a data structure perspective, they are third-order tensors, i.e., a dataset includes many MTS, each one containing multiple signals, and each signal is composed of several timestamps. From a numerical perspective, it is frequent to work with datasets composed of thousands of MTS and tens of signals with thousands of records (see, for example, the datasets used in the experiments in Section 5). This problem gives rise to a first challenge to address: **(C1):** *Analyzing MTS datasets requires the application of scalable techniques capable of dealing with the high dimensionality of the data.*

We address this challenge by proposing Time2Feat, which computes the clusters based on features extracted from the signals of the MTS. This operation allows us to reduce the problem's dimensionality: from the many timestamps constituting the time series to the single values of the features. We rely on external specialized software libraries to extract *intra and inter-signal* interpretable features from the MTS. The former describes particular properties of the signals in isolation (e.g., the mean value, the autocorrelation, etc.). The latter evaluates pairwise the signals measuring for example distances, and correlations. Feature extraction can give rise to a large number of features describing the same MTS under different (but also possibly close) perspectives. For example, this operation overall generates 4842 features in the BasicMotion Dataset. The high dimensionality could both be the cause of inefficiencies in the generation of clusters and could lead to the creation of clusters difficult to interpret by the users. The large amount of features would make them non-interpretable by humans who would not understand what clusters they represent and the reasons why the data points were grouped together in the same clusters. This problem introduces a second challenge: **(C2):** *providing an interpretable clustering technique is of paramount importance for data analysis.*

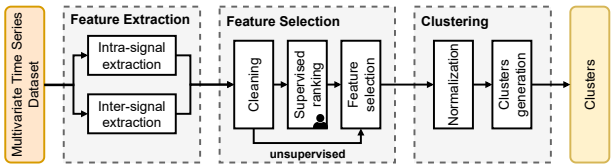


Figure 2: The Time2Feat **pipeline.**

State-of-the-art approaches for MTS clustering [13, 18, 19] suffer from low interpretability. Time2Feat addresses this problem by relying on a reduced number of interpretable features. In particular, Time2Feat adopts a mechanism for feature selections based on the PFA, that ranks the features according to their importance in the process and selects only the meaningful ones. Figure 1a shows the features reduced by PFA to 46 from the 4842 initially extracted.

Although Time2Feat does not rely on a specific clustering technique, the number of clusters to generate is another critical parameter to select. We adopt the well-known Elbow Method to automatically compute the number that better fits the computed statistical measures. The application of the Elbow method to BasicMotion generates 4 clusters, as shown in Figure 1c in blue circles (t-SNE [48] was applied to reduce the dimensionality). We point out that this result is obtained through a completely automatic *unsupervised procedure* where the pipeline starts with the BasicMotion dataset and generates 4 clusters based on 46 features. Data analytics processes are typically the result of several iterations where users gain more and more insights from the data that enable them the application of deeper analytical functions. This intuition is the third challenge to address: **(C3):** *clustering techniques for data analytics need to put the human in the loop.*

Time2Feat addresses this challenge by supporting a *semi-supervised procedure* allowing users to select samples of elements representative of the clusters they want to generate. Our experiments demonstrate that selecting (i.e. labeling) a few elements per cluster improves the accuracy and significantly reduces the number of features adopted by the clustering technique, thus improving their interpretability. Going back to our example, suppose that a user decides to manually provide four elements per cluster. For instance, the user can analyze some elements from the top-right cluster of Figure 1c and observe that they refer to people playing badminton. Indeed, the high variance of the acceleration recorded for the cluster elements (as shown in Figure 1a), caused by the sudden and irregular movements in badminton can clearly support the decision. The user can also observe the values of the acceleration variance for the elements from the second top-right cluster, and decide that the cluster refers to people doing running. Through a similar analysis, the user can notice that the variance of the other clusters suggests less intensive activities, typical of walking and standing people. Time2Feat exploits the user annotations by further reducing the number of features used for the clustering (from 46 down to 3), as shown in Figure 1b, and improving the accuracy of the clusters.

Finally, we would like to point out that cluster analysis offered by Time2Feat is exceptionally flexible and facilitates precise and fruitful data explorations. For instance, in case the user would like to obtain two clusters instead of four, Time2Feat would derive a first cluster with elements from the first two clusters representing badminton and running activities and a second cluster with elements representing standing and walking activities.

3 CLUSTERING MULTIVARIATE TIME SERIES

Figure 2 depicts the main components of our MTS clustering pipeline. The pipeline can be formally defined as follow.

Definition 3.1 (Multivariate Time Series). A multivariate time series M is a set of univariate time series (a.k.a. *signals*). In particular, $M = (u_1, u_2, \ldots, u_S)$, where S is the number of signals, and $u_j = (t_{1j}, t_{2j}, \ldots, t_{Nj})$ is a time series of length N. More generally, a multivariate time series can be represented as a matrix $\mathbb{R}^{NxS}$, where the signals are described as column vectors.

Definition 3.2 (Multivariate Time Series Dataset). A dataset D of multivariate time series is a set of V multivariate time series $D = (M_1, M_2, \ldots, M_V)$. A dataset is represented as a tensor $\mathbb{R}^{VxNxS}$.

In the pipeline, we consider both intra-signal features $\mathcal{F}$, describing the signal constituting the MTS in isolation, and inter-signal $\mathcal{F}'$, describing pairs of signals, as defined below.

Definition 3.3 (Set of intra-signal features). Given a multivariate time series M, composed of S signals, $M = (u_1, u_2, \ldots, u_S)$ and a set of functions for intra-signal feature extraction $\mathcal{F} = (f_1, f_2, \ldots f_F)$, a set of intra-signal feature is the set of values resulting from the application of the functions to the signals. $F_{intra} = (e_{11}, ..., e_{1F}, ..., e_{S1}, ..., e_{SF})$, where $e_{ij} = f_j(u_i)$.

Definition 3.4 (Set of inter-signal features). Given a multivariate time series M, composed of S signals, $M = (u_1, u_2, \ldots, u_S)$ and a set of functions for inter-signal feature extraction $\mathcal{F}' = (f'_1, f'_2, \ldots f'_{F'})$, a set of inter-signal feature is the set of values resulting from the application of the functions to pairs of signals. $F_{inter} = (e_{(1,2)1}, ..., e_{(1,S)F'}, ..., e_{(S-1,S)F'})$, where $e_{(i,j)k} = f'_K(u_i, u_j)$.

Given the large scale of feature sets, the feature selection step is devoted to pruning features with null values or low variance and ranking the remaining ones in order only to keep the meaningful ones for the clustering step. This leads to obtaining significant and semantically rich features, thus achieving interpretability and scalability. Contrarily to raw data points, features are interpretable and understandable for end-users. The third pipeline step then executes clustering as defined below.

Definition 3.5 (Clustering MTS). Given a multivariate time series dataset D and a set of clusters C with cardinality k, the goal of MTS clustering is to map a cluster to each series in D. This corresponds to define a surjective function $m : D \longrightarrow C$ that maps each time series $M \in D$ into a cluster $k \in C$.

4 THE TIME2FEAT SYSTEM

Time2Feat implements the components of the data analysis pipeline as illustrated in Figure 2. Time2Feat takes a MTS dataset D as input and the number of clusters to generate (provided by the user or via some heuristic). It can run under *unsupervised mode*, i.e., no further input is required, and under *semi-supervised mode*, i.e., the users specify a subset of clustered samples. The three steps of the pipeline

Algorithm 1: `feature_extraction`

Input : $D \in \mathbb{R}^{VxNxS}$ Multivariate time series dataset.
Output: $F \in \mathbb{R}^{VxE}$ Matrix of extracted features.

```
   // Extracting intra-signal features
1  F[] ← 0; // list of extracted features
2  foreach V ∈ D; // For each MTS in the dataset
3  do
4      foreach S_i ∈ V; // For each signal in the MTS
5      do
6          F ← intra_feature_extraction(S_i)
   // Extracting inter-signal features
7  foreach V ∈ D do
8      foreach S_i ∈ V do
9          V = V − S_i;
10         foreach S_j ∈ V; // For pairs of signals in the MTS
11         do
12             F ← inter_feature_extraction(S_i, S_j)
13 return F;
```

Algorithm 2: `feature_selection`

Input : $F \in \mathbb{R}^{VxE}$ Matrix of extracted features.
labels Optional labels
Output: $t \in \mathbb{R}^{VxF}$ Matrix of signals and top features.

```
   // Remove features with constant, null and/or infinite values
1  T ← clean(F);
   // Semi-supervised step if the user labels some records
2  if labels then
3      T ← auto_anova_selection(T, labels);
4  T ← pfa(T); // Extract best feature with PFA
5  return T;
```

are implemented by the *feature_extraction* function (described in Section 4.1), the *feature_selection* function (in Section 4.2) and the *cluster* function (in Section 4.3).

4.1 Feature Extraction

The goal is to generate an exhaustive representation of an MTS dataset via a large spectrum of features, each describing the MTS signals (in isolation or pairs).

Intra-signal Features Extraction. The computation of statistical features describing the signals of the MTS relies on the library `tsfresh` [6, 7], already in many time series analysis tasks [36, 38, 50]. Each of the 700+ features computed by `tsfresh` encodes the signal description from the perspective offered by a specific analysis method, such as Distribution Analysis, Statistical Analysis, etc. In this way, the features are interpretable for users who know how the statistical measure summarizes the time series values. Lines 2-6 of Algorithm 1 shows a simplified procedure where a nested for-cycle iterates over the MTS and the `intra_feature_extraction` function generates the features for each composing signal. In the actual implementation, we leverage the efficient parallelization of the feature extraction function provided by `tsfresh` that can compute features in batches of univariate series.

Inter-signal Features Extraction Many works [20, 42] highlight the importance of inter-signal relationships in the analysis of time series. Nevertheless, they typically extract the features employing neural network architectures, obtaining uninterpretable descriptions. We adopt a straightforward approach by conceiving inter-signal features as the measure of the relatedness (in terms of similarity and correlation) between pairs of signals that we measure through 8 metrics (e.g., correlation, Euclidean distance, etc.). The pipeline firstly generates the pairs of signals per time series (lines 7-10 of Algorithm 1), then applies (line 12 of Algorithm 1) the function `inter_feature_extraction` in charge of the extraction.

4.2 Feature Selection

The feature extraction procedure generates a large number of features per dataset. Reducing the dimensionality of such representation improves the interpretability and increases (as the experiments in Section 5.1 show) the performance of the clustering procedure.

As a first operation, in line 1 of Algorithm 2, we `clean` the matrix of the features by removing all zero-variance features and the features that have missing or infinite values (they would be useless for the cluster generation). If Time2Feat is running with the *semi-supervised mode*, the available labels are used to rank the relevance of the features for identifying a subset capable of generating clusters. We rely on the *Analysis of Variance (ANOVA)* [44] for computing the p-value associated with each feature and quantifying its significance. Then, we apply a grid search analysis identifying the subset of features that maximizes the quality of the generated clusters. To evaluate the quality, we use the *Homogeneity Score* [34], the Adjusted Mutual Information (AMI) [28] and *Adjusted Rand Index* [45] which are specific measures evaluating the agreement and similarity of pairs of cluster elements. The joint application of the ANOVA and grid search analysis is referred to `auto_anova_selection` in line 3 of Algorithm 2. Finally, both in the presence and in the absence of labels, we apply the PFA technique to select the most meaningful features. We observe that the PFA technique not only guarantees *conciseness* but also *diversity*[1] of the features by choosing "the principal features which retain most of the information in the sense of maximum variability of the features in the lower dimensional space" [21].

4.3 Clustering

Time2Feat can work with any clustering algorithm. In Section 5, we show that among the experimented approaches, the hierarchical technique achieved the best accuracy. Concerning the number of clusters, the Time2Feat system leverages state-of-the-art heuristics (e.g., applying the well-known Elbow method) or user preferences. Finally, the clustering operation includes a normalization step that avoids the dominance of features due to large-scale domain ranges.

5 EXPERIMENTAL EVALUATION

The evaluation addresses four main research questions:

- RQ1 How effective is Time2Feat in solving MTS clustering tasks (Section 5.1);
- RQ2 To what extent the representations of the generated clusters are interpretable (Section 5.2);
- RQ3 How efficient is the cluster computation (Section 5.3);

[1] Diversity is another properties of interpretable features as reported in [16, 29, 31]

Table 1: The datasets evaluated in the experiments. V is the number of MTS, S the number of signals, N the length of the series, C the number of classes in the ground truth, E_O the number of elements per dataset and E_M the number of elements per MTS.

Dataset	V	S	N	C	E_O $(VxSxN)$	E_M (SxN)
Li — Libras	360	2	45	15	32400	90
AF – AtrialFibrillation	30	2	640	3	38400	1280
BM – BasicMotions	80	6	100	4	48000	600
RS – RacketSports	303	6	30	4	54540	180
ER – ERing	300	4	65	6	78000	260
Ep – Epilepsy	275	3	206	4	169950	618
PD – PenDigits	10992	2	8	10	175872	16
SW – StandWalkJump	27	4	2500	3	270000	10000
UW – UWaveGestureLibrary	440	3	315	8	415800	945
Ha – Handwriting	1000	3	152	26	456000	456
AW – ArticularyWordRecognition	575	9	144	25	745200	1296
HM – HandMovementDirection	234	10	400	4	936000	4000
LS – LSST	4925	6	36	14	1063800	216
Cr – Cricket	180	6	1197	12	1292760	718
EC – EthanolConcentration	524	3	1751	4	2752572	5253
S1 – SelfRegulationSCP1	561	6	896	2	3015936	5376
S2 – SelfRegulationSCP2	380	7	1152	2	3064320	8064
PS – PhonemeSpectra	6668	11	217	39	15916516	2387

RQ4 How robust is the pipeline, i.e. to what extent do the components in the pipeline contribute to the task. (Section 5.4)

Baselines. We selected 18 benchmark datasets from the UEA multivariate time series classification archive [3]. For each dataset, Table 1 reports the number of MTS (V), the number of signals (S), the length (N) of the series, and the clusters (C), where the MTS can be grouped according to the baselines. In addition, we computed the overall number of elements in the dataset (E_O) that provides a yardstick for measuring the scalability of the approach. Finally, we estimate the complexity of generating the clusters by computing the number of elements per MTS (E_M). Intuitively, the lower the value, the lower the ability to extract descriptive features. The datasets represent different scenarios as their overall number of elements E_O spans over three orders of magnitudes, and E_M ranges from 16 elements for the PD dataset to 10000 for SW.

We compared Time2Feat with eight approaches: Hierarchical, KMeans, and Spectral are straightforward applications of these classical clustering techniques to MTS datasets. CSPCA and MC_2PCA introduce a PCA-based mechanism to reduce the data dimensionality before the clustering. DETSEC and IT-TSC leverage neural networks [2]: the former by creating embeddings for the series through autoencoders, the latter by combining a multi-path neural network with variable association graphs to determine the importance of the signals for each cluster. Finally, we created a variant of the KMeans clustering technique by introducing DTW to measure the similarity between two temporal sequences. We refer the reader to Section 6 for an extensive discussion of these baselines.

Setup. The experiments are executed on a machine with a 12 cores Intel Xeon Processor, 64GB of RAM, and 324GB of local (SSD) storage. The machine runs Ubuntu version 18.04. All experiments have been executed ten times, and the average result plus standard deviation is reported (whenever significant).

[2]Our technical report [2] includes experiments with other neural network techniques.

5.1 Effectiveness

We evaluated the effectiveness of Time2Feat by adopting the AMI [28] to measure the accuracy of the generated clusters with respect to the baselines. The AMI evaluates to 1 when the two clusterings are identical, and to roughly 0 (negative values are allowed) in case of random partitions. Table 2 shows the results of this experiment. Time2Feat has been evaluated by executing the *unsupervised mode* (column $\mathbf{T2F}_0$) and by simulating the *semi-supervised mode* through stratified random samples composed of 20% (column $\mathbf{T2F}_2$), 40% (column $\mathbf{T2F}_4$), 50% (column $\mathbf{T2F}_5$) of labels per cluster from the baseline datasets. The remaining columns show the competing approaches (discussed in Section 6). Among them, Hierarchical, KMeans, and Spectral can be considered as reference baselines for their simplicity[3].

Discussion. The experiment results clearly show that Time2Feat outperforms its competitors. In particular, in the unsupervised mode, the accuracy of the clusters generated by Time2Feat is higher than the other approaches in 11 out of 18 datasets. Among them, in 3 datasets, it obtains the best accuracy score. By providing 20% labels per cluster, Time2Feat outperforms the other approaches in 13 datasets (obtaining in 2 datasets the best accuracy score). The performances generally improve by adding more labels, as in the configuration $\mathbf{T2F}_5$, where Time2Feat outperforms the other approaches in 15 out of 18 datasets (showing the best accuracy value in 9 out of 18 datasets). This experiment helped us derive the following insights: (1) Time2Feat's pipeline is highly efficient as at least one configuration of Time2Feat outperforms the other approaches in all datasets, except in the UW dataset, where it performs slightly worse than some competing approaches. The reason is that UW describes trajectories. One kind of trajectory is the composition of two other trajectories. The features extracted by Time2Feat cannot recognize these three different movements. (2) Time2Feat is highly scalable as it obtains high accuracy results both for small (the ones at the top of Table 2) and for large datasets (the ones at the bottom of Table 2). These results do not hold for the competitors, where the accuracy drops as the number of elements in the dataset increases (and in some cases, marked N/A in the Table 2, no cluster is generated due to timeout or memory exceptions). (3) The semi-supervised procedure improves the accuracy. By labeling a small number of elements per dataset, the accuracy steadily increases.

5.2 Interpretability

We provide a measure of the interpretability of the clusters by analyzing the number of features that Time2Feat uses for their computation. A limited number of features facilitating human comprehension and conciseness is one of the main properties of interpretable features (see Section 6). The column All in Table 3 shows the overall amount of features extracted after the feature extraction step of the pipeline (Section 4.1). The other columns report the number of features retained with the unsupervised mode (column $\mathbf{T2F}_0$) and with increasing levels of supervision as in the previous experiment. The values represent the average number of features across ten runs of each experiment.

[3]We rely on the `sklearn` implementations of these algorithms with default parameters.

Table 2: Effectiveness (AMI). In bold, the best value per dataset. ↑ shows Time2Feat settings overcoming all competing approaches.

Dataset	Semi-supervised			Unsupervised	Competing approaches							
	$T2F_2$	$T2F_4$	$T2F_5$	$T2F_0$	Hierarchical	KMeans	Spectral	DTW	CSPCA	DETSEC	MC_2PCA	IT-TSC
Li	0.728±0.02↑	0.722 ± 0.016↑	**0.730±0.020**	0.716±0.012↑	0.563	0.545	0.492	0.503	0.311	0.416	0.069	0.483
AF	0.028±0.046	0.123 ± 0.066↑	**0.238±0.059**	0.038±0.027↑	-0.002	-0.002	-0.002	0.005	-0.07	-0.001	-0.056	-0.06
BM	0.977±0.034↑	**1.000 ± 0.000**	**1.000±0.000**	**1.000±0.000**	0.347	0.23	0.002	0.832	0.7	**1.000**	0.189	0.676
RS	0.559±0.038↑	0.666 ± 0.049↑	**0.710±0.047**	0.35±0.006	0.192	0.194	0.0	0.215	0.221	0.224	0.094	0.41
ER	0.801±0.016	0.823 ± 0.014	0.826±0.023	**0.921±0.011**	0.859	0.91	0.0	0.775	0.5	0.646	0.115	0.315
Ep	0.896±0.025↑	**0.913 ± 0.019**	0.882±0.007↑	0.792±0.04↑	0.135	0.167	-0.001	0.25	0.258	0.213	0.08	0.68
PD	0.752±0.022↑	0.771 ± 0.028↑	**0.784±0.013**	0.437±0.02	0.728	0.682	N/A	0.6	N/A	0.431	0.065	0.716
SW	0.038±0.036	0.101 ± 0.01	**0.23±0.046**	0.048±0.079	0.131	-0.002	0.0	-0.005	-0.072	-0.097	0.045	-0.02
UW	0.555±0.026	0.554 ± 0.036	0.59±0.035	0.587±0.055	**0.752**	0.712	0.0	0.611	0.236	0.414	0.111	0.749
Ha	0.325±0.023↑	**0.353 ± 0.019**	0.349±0.009↑	0.161±0.006	0.226	0.193	0.0	0.235	0.165	0.271	-0.004	N/A
AW	0.921±0.007	0.931 ± 0.01↑	0.927±0.005↑	**0.963±0.007**	0.926	0.902	0.0	0.781	0.716	0.794	0.182	0.752
HM	0.021±0.011↑	**0.045 ± 0.007**	0.07±0.012	0.015±0.008	-0.006	-0.002	0.001	0.01	0.002	-0.004	0.018	-0.01
LS	0.293±0.013↑	0.317 ± 0.011↑	**0.333±0.002**	0.156±0.009↑	0.028	0.018	0.001	N/A	0.047	0.152	0.048	N/A
Cr	**0.984±0.021**	0.975 ± 0.018↑	0.974±0.021↑	0.946±0.021↑	0.756	0.719	0.0	N/A	0.876	0.865	0.361	0.274
EC	0.065±0.017↑	0.097 ± 0.006↑	**0.121±0.04**	0.052±0.002↑	0.009	0.01	-0.003	N/A	0.013	N/A	0.002	0
S1	**0.397±0.047**	0.374 ± 0.025↑	0.382±0.012↑	0.007±0.001	0.212	0.194	-0.001	N/A	N/A	0.18	0.022	0.08
S2	0.008±0.003↑	**0.015 ± 0.004**	**0.015±0.006**	0.003±0.001 ↑	-0.002	-0.001	0.01	N/A	0.001	0.007	0.005	0.002
PS	0.2±0.006↑	**0.202 ± 0.002**	0.201±0.002↑	0.121±0.007↑	0.093	0.096	0.0	N/A	N/A	N/A	0.058	N/A

Table 3: Number of intra-signal / inter-signal features.

Dataset	All	$T2F_0$	$T2F_2$	$T2F_4$	$T2F_5$
Li	1574/8	55/1	7.17/0	8.33/0	9.4/0
AF	1574/8	21/0	2.83/0.17	5.67/0	5.67/0
BM	4722/120	44.33/1.67	2.33/1.5	2.0/0.67	2/0.17
RS	4722/120	141.2/9.8	12.8/2.6	15.4/3.4	21/4.2
ER	3148/48	125.83/3.17	7/1.67	6.83/1.33	7.17/1.17
Ep	2361/24	163.33/3.67	12.67/1.83	15.67/1.17	15.33/1.33
PD	1574/8	98/1	16.4/0.6	13.8/0.6	18.8/0.8
SW	3148/48	20/0	1.8/0.4	3.4/0	6.4/0
UW	2361/24	124/3	4.4/0.2	4.4/0.2	4.4/0
Ha	2361/24	309.83/3.17	23.83/1.5	25/2.17	30.83/2.67
AW	7083/288	283.67/30.33	10/5	9.5/4	10.5/4
HM	7870/360	167/11	15.17/1	28.17/1.33	24.83/2.17
LS	4722/120	217/5	6.6/3.6	9.4/3.2	12.8/4.4
Cr	4722/120	113.17/3.83	4.5/3.83	4.17/4.17	4.17/4
EC	2361/24	122.83/5.17	9.33/0.33	8.5/0	3/0
S1	4722/120	222.2/1.8	2/0.2	2/0	3/0.2
S2	5509/168	183/3	26.4/0.4	20.8/0.4	20.2/0
PS	8657/440	293/8	4/0	4.4/0	4.2/0

Table 4: Runtime execution , in seconds (- timeout exception fixed in 10 hours, × memory exception).

Dst.	$T2F_0$	Hier.	KMeans	Spec.	DTW	CSPCA	MC_2PCA	DETSEC	IT-TSC
Li	20.31	0.2	0.28	0.37	366	2	0.53	500	31
AF	31.01	0.04	0.06	0.31	350	0.01	0.12	876	57
BM	58.45	0.09	0.16	0.23	175	0.03	209	260	31
RS	50.11	0.2	0.39	0.39	474	0.317	356	270	60
ER	34.2	0.18	0.24	5.27	914	0.21	559	555	85
Ep	47.95	0.24	0.42	7.71	3667	0.31	682	1673	173
PD	198.03	9.0	3.0	×	24713	50	6	3395	7410
SW	559.69	0.16	0.25	0.34	10768	0.01	1	10142	255
UW	58.42	0.45	0.74	15.6	20639	0.47	3063	4229	600
Ha	44.74	0.86	2.0	7.19	27018	0.19	25	4301	-
AW	135.18	1.0	1.0	1.35	23611	1	18811	220	57
HM	220.78	0.83	1.0	0.89	30251	0.62	3162	3175	783
LS	300.23	6.0	3.0	6.49	-	0.35	16591	4666	-
Cr	737.61	0.8	1.0	0.91	-	0.14	13642	12145	2478
EC	876.3	2.0	2.0	2.01	-	0.26	9708	-	2865
S1	727.37	2.0	2.0	2.37	-	-	12	19317	1831
S2	952.58	2.0	2.0	2.22	-	0.36	11	20898	1831
PS	1219.88	1.0	1.0	34.73	-	×	×	-	-
Avg	348.49	1.50	1.14	5.19	11912.17	3.52	3931.69	5537.88	1390.73

Discussion. The feature extraction generates a large number of features that increases as the number of elements per dataset E_O (the Pearson correlation coefficient – Pcc is 0.61) and the number of elements per MTS E_M (Pcc = 0.39) increases. The unsupervised approach drastically reduces the selected features while maintaining a strong correlation (Pcc = 0.51) with the overall number of features. Not always an increase in the supervision corresponds to a reduction in the features. We explain this as a sort of "overfitting" that forces better accuracy results by adding features. However, we observe the small number of features (they can be managed by humans) retained in all semi-supervised settings. Finally, the experiment shows the importance of the inter-signal features. Despite the low number, the extracted inter-signal features are preserved in almost all Time2Feat settings, thus showing their importance in the clustering process.

5.3 Efficiency

We evaluate the efficiency of our approach by computing the overall time required to complete the pipeline (Section 5.3.1), evaluating the time breakdown (Section 5.3.2) and introducing a simple heuristic to optimize the parallelism of the feature extraction (Section 5.3.3).

5.3.1 Time Performance. Table 4 shows the maximum time to complete the cluster computations for all the datasets in the 10 repetitions of the experiment. We show only the time measured in the unsupervised mode ($T2F_0$): the semi-supervision does not change the value significantly. The last row shows the average time computed on all datasets (excluding the ones raising the exceptions).

Discussion. The clustering techniques that reach the best time performance are KMeans and Hierarchical, which finish the pipelines in a few seconds. However, this comes at the cost of accuracy and interpretability loss. CSPCA also shows a low time performance, but the algorithm cannot handle large datasets, where time and memory exceptions occur and the accuracy is poor. The other approaches report an average time greater than Time2Feat of at least an order of magnitude and, in some cases, time and memory exceptions. Finally, we observe that Time2Feat's execution time ranges from a few seconds to a few thousands of seconds, having the performance correlated with the overall number of elements (Pcc=0.75).

5.3.2 Time breakdown. We analyze the breakdown of the computation time (Figure 3a) into the main pipeline components (feature extraction, feature selection, and cluster generation).

Discussion. The time required for extracting the features dominates the other components: it takes between 88% and 99% of the overall time needed for completing the pipeline. The average time to complete the feature extraction is around 337 seconds, and the one to complete features selection is 9 seconds, whereas, for clustering, it amounts to 1 second. The correlation between the time spent in clustering and V is strong (Pcc=0.99). This is why in three datasets (PD, LS, and PS, the ones with the largest V), the clustering time takes more than 1 second but less than 8 seconds. The feature extraction and selection show a different behavior: they are correlated with the overall number of elements in the datasets E_O (the Pcc is more than 0.95 for both the tasks). In order to further confirm these trends, we have also used 27 synthetic datasets generated by varying the number of MTS $V \in (100, 1000, 2500)$, the number of signals $S \in (2, 8, 16)$ and the length of the series $N \in (100, 1000, 2000)$ by means of GRATIS tool [15]. The results (omitted for space reasons and reported in [2]) show that by fixing V and varying S and N, the only time that increases is the time due to feature extraction, while the other times (feature selection and clustering) remain constant. As a conclusion, despite the increase, the approach remains overall scalable for several datasets, whereas for larger ones alternative strategies can be devised, as shown in the next experiment.

5.3.3 Workload balancing. We evaluate a straightforward heuristic to improve the time performance by optimizing the computational workload on the processors. Feature extraction performs the computation using batches of time series, which are not balanced by default. Time2Feat allows to balance the workload by customizing the number of batches per dataset by dividing the total number of MTS (V) by the number of available processors, rounding for excess to the upper integer. Figure 3b shows the time reduction obtained.

Discussion. By balancing the workload on the processors, the time performance essentially improves in almost all datasets (the average time computed on all datasets decreases from 348 to 242 seconds. In 5 datasets (AF, BM, Ep, SW, and HM), the time reduction is more than 60%. In only two cases (PD and LS), the heuristic does not affect the time performance, and the time slightly increases.

5.4 Robustness

This Section assesses the robustness of the pipeline components by evaluating: the importance of feature selection (Section 5.4.1); alternative options to the hierarchical algorithm for performing the final cluster computations (Section 5.4.2); the importance of the features in the clustering task (Section 5.4.3).

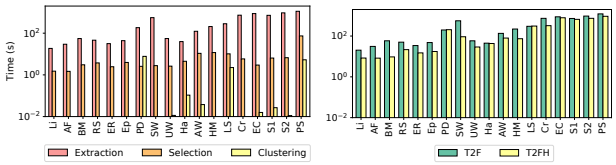
5.4.1 Importance of feature selection. We evaluate how the accuracy (AMI) in Figure 4a, the interpretability (number of features) in Figure 4b, and the efficiency in Figure 4c vary with and without feature selection along the pipeline.

Discussion. The experiment shows that the removal generally has a large impact on the accuracy and interpretability. The AMI largely decreases in almost all datasets when the clusters are computed with all features. Moreover, feature selection reduces the number of features of two orders of magnitude. Finally, the time required for performing the feature selection is negligible, especially compared to the feature extraction step as shown in Figure 3a. In summary, the feature selection improves the accuracy and the interpretability, and only slightly affects the time efficiency of the pipeline.

5.4.2 Importance of the clustering technique. We experimented with 3 techniques (*Hierarchical*, *KMeans*, *Spectral*) for generating the clusters, as shown in Table 5. For each technique, we computed the AMI of the clusters obtained with three settings: the unsupervised procedure ($\mathbf{T2F}_0$), the semi-supervised procedure with 20% and 50% labeled elements per cluster ($\mathbf{T2F}_2$ and $\mathbf{T2F}_5$, respectively).

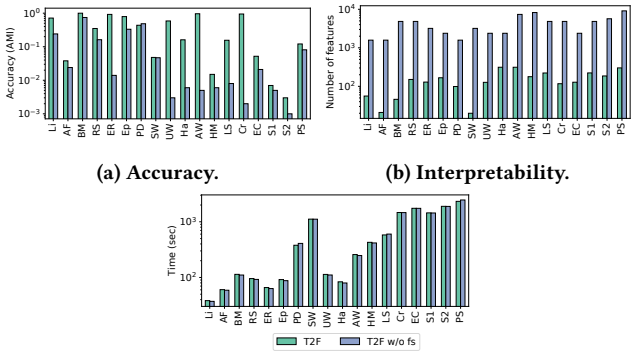
Table 5: Accuracy (AMI) varying the clustering techniques. In bold, the best result per dataset.

Dataset	$\mathbf{T2F}_0$			$\mathbf{T2F}_2$			$\mathbf{T2F}_5$		
	Hier.	KMeans	Spec.	Hier.	KMeans	Spec.	Hier.	KMeans	Spec.
Li	0.716	0.711	0.627	0.728	0.691	0.709	**0.73**	0.722	0.705
AF	0.038	0.007	-0.001	0.028	0.047	0.047	**0.238**	0.192	0.188
BM	**1**	**1**	0.992	0.977	0.961	0.902	**1**	**1**	0.993
RS	0.35	0.359	0.371	0.559	0.578	0.612	**0.71**	0.649	0.663
ER	0.921	**0.955**	0.925	0.801	0.819	0.79	0.826	0.824	0.804
Ep	0.792	0.874	0.528	**0.896**	0.839	0.8	0.882	0.867	0.793
PD	0.437	0.639	0.377	0.752	0.727	0.651	**0.784**	0.715	0.688
SW	0.048	0.071	-0.004	0.038	0.074	0.167	0.231	**0.327**	0.256
UW	0.587	0.566	0.454	0.555	0.541	0.511	**0.59**	0.539	0.537
HM	0.161	0.153	0.014	0.325	0.302	0.277	**0.349**	0.325	0.289
AW	**0.963**	0.945	0.807	0.921	0.918	0.89	0.927	0.903	0.899
HM	0.015	0.011	0.005	0.021	0.048	0.037	0.069	**0.089**	0.062
LS	0.156	0.146	0.041	0.293	0.315	0.037	**0.333**	0.332	0.051
Cr	0.946	0.907	0.787	**0.984**	0.956	0.95	0.974	0.96	0.967
Ec	0.052	0.056	0.049	0.065	0.066	0.06	**0.121**	0.094	0.106
S1	0.007	0.019	0.002	0.397	**0.419**	0.387	0.382	0.391	0.379
S2	0.003	0	0	0.008	0.015	0.021	0.015	0.024	**0.035**
PS	0.121	0.143	0.143	0.2	**0.211**	**0.211**	0.201	0.208	0.208



(a) Pipeline Breakdown. **(b) Custom parallelization.**

Figure 3: Efficiency analysis.



(a) Accuracy. **(b) Interpretability.**

(c) Efficiency.

Figure 4: Removing the Features Selection from the pipeline.

Discussion. The results show that Hierarchical clustering obtains the best performance. The KMeans technique has similar accuracy. The Spectral clustering is competitive only for the largest datasets when it achieves the best results, but inline with other approaches.

5.4.3 Importance of the features in the clustering task. This experiment evaluates whether a feature-based clustering approach is more effective than an approach based on raw data. To this end, we run Time2Feat in the unsupervised mode by performing the clustering computation with the same techniques used in the previous experiment (Hierarchical, KMeans, and Spectral), and we compare the accuracy obtained (in terms of AMI) with the one obtained by the application of the same clustering technique to the raw datasets.

Discussion. The results of the experiment reported in the technical report [2] show that the feature-based clustering approach usually obtains the best performance. Only in three datasets (PD, UW, and S1) the approach based on raw data performs slightly better.

5.5 Lessons Learned

We conclude by pinpointing how our feature-based clustering pipeline addresses the aforementioned questions.

(RQ1) Thanks to the features, we gain in effectiveness. The experiment in Section 5.1 demonstrates that Time2Feat provides more accurate clusters than its competitors. The ablation test in Section 5.4.3 further confirms that feature-based clustering techniques generate more effective results than techniques using raw data.

(RQ2) The cluster representations are concise. The experiment in Section 5.2 shows that Time2Feat relies on a small number of features for generating the clusters, and shows the importance of inter-signal features retained during cluster generation.

(RQ3) Feature-based clustering achieves a trade-off between accuracy and performance. Time2Feat achieves the best accuracy in the majority of the datasets along with good performance. The approach is among the fastest ones and performs in seconds, thus making it efficiently usable for batch analyses (Section 5.3). The studied time breakdown of the pipeline components shows that the feature extraction phase is the most expensive (Section 5.3.2). Nevertheless, heuristics for optimizing the workload balance, e.g., concerning the available processors (Section 5.3.3), can be quickly developed to reduce the overall time execution considerably.

(RQ4) The pipeline is robust and scalable. The pipeline is highly modular and then scalable concerning the specificities of real-world environments. Our robustness analysis shows the importance of all pipeline components. Feature selection (Section 5.4.1) improves both accuracy and interpretability. The comparison of clustering techniques shows the adaptativeness of the pipeline, allowing for striking a balance between accurate results in small and large datasets depending on the use case at hand (Section 5.4.2).

6 RELATED WORK

Clustering of multivariate time series. Dimensionality reduction is one of the research questions addressed by previous work on MTS clustering. Principal component analysis (PCA) [5, 17, 39, 41, 43] has been adopted to transform MTS into a new dimensional space to find the most critical features representative of the original MTS. Among the approaches, we recall the Covariance Sequence-based Principal Component Analysis (CSPCA) [19] that builds a matrix representing the pairwise covariance of the MTS. Then, a PCA-based transformation is applied to the matrix to reduce the dimensions, followed by clustering techniques. MC_2PCA [18] is a similar technique, that applies a cycle of feature transformation based on an internal component, the Common Principal Analysis (CPCA) or clustering (based on KMeans) until the reconstruction error is small. The main problem of PCA-based approaches is the poor explainability of the generated clusters building on latent space dimensions with no semantics for the end-users.

Building time series representations (embeddings) based on Neural Networks is another line of research to generate clusters [1, 9]. DETSEC [13] is a state-of-the-art embedded solution, based on an encoder-decoder architecture built with GRU components. IT-TSC [49] is an approach based on neural networks to build variable association graphs for clusters creation. Approaches based on Neural Networks have shown high performance for detecting the clusters, but they suffer from poor interpretability. Moreover, a large body of work has been devoted to UTS clustering (such as Dynamic Time Warping (DTW) [26], KShape [30] and FeatTS [46, 47]). All these methods are not directly applicable to MTS clustering due to the inherent differences between univariate and multivariate time series, leading to poor scalability.

Interpreting the clusters. Providing insights for cluster membership is a real need in many scenarios [4, 32, 37, 40]. Explainable AI (XAI) is one of the current hottest topics [8, 24], usually approached in two ways [10, 51]: 1) by exploiting post-hoc analysis or 2) by designing intrinsically explainable systems. Time2Feat falls into the second category, being based on interpretable features. More specifically, interpretability of the clustering techniques is typically addressed: (1) by applying dimensionality reduction techniques (e.g., PCA) to be able to visualize clusters through two or three dimensions; (2) by identifying the centroid or a selected set of points to represent the cluster [33]; and (3) by relying on an interpretable model (usually a decision tree) that learns how to classify the generated clusters [4, 12, 32, 37]. In our work, we mainly deal with interpretable representations, that can support users in understanding cluster's contents. Whereas there is no consensus in the literature on the meaning of interpretable features [52] and on what properties interpretable features should satisfy [25], several approaches indicate conciseness as one of the key properties for interpreting algorithm behaviors [16, 29, 31].

7 CONCLUSION

We have presented an end-to-end feature-based clustering pipeline for multivariate time series, leveraging state-of-the-art machine learning components and making them interact with each other. We have empirically studied Time2Feat under the lenses of its effectiveness, interpretability, efficiency, and robustness, comparing it with existing clustering methods on several real-world and benchmarking datasets. The results show that the combination of interpretable features and weakly labeled MTS lead to better quality and explainability of the obtained clusters.

ACKNOWLEDGMENTS

This work was partially supported by ANR (grant nr. 18-CE23-0002 QualiHealth).

REFERENCES

[1] Ali Alqahtani, Mohammed Ali, Xianghua Xie, and Mark W Jones. 2021. Deep Time-Series Clustering: A Review. *Electronics* 10, 23 (2021), 3001.

[2] Angela Bonifati, Francesco Del Buono, Francesco Guerra, Donato Tiano. 2022. Time2Feat: Evaluation (Technical report). https://github.com/softlab-unimore/time2feat. Technical Report.

[3] Anthony J. Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn J. Keogh. 2018. The UEA multivariate time series classification archive, 2018. *CoRR* abs/1811.00075 (2018).

[4] Dimitris Bertsimas, Agni Orfanoudaki, and Holly M. Wiberg. 2021. Interpretable clustering: an optimization approach. *Mach. Learn.* 110, 1 (2021), 89–138.

[5] Lianfang Cai, Nina F Thornhill, Stefanie Kuenzel, and Bikash C Pal. 2018. Wide-area monitoring of power systems using principal component analysis and k-nearest neighbor analysis. *IEEE Transactions on Power Systems* 33, 5 (2018), 4913–4923.

[6] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr. 2018. Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh - A Python package). *Neurocomputing* 307 (2018), 72–77.

[7] Maximilian Christ, Andreas W. Kempa-Liehr, and Michael Feindt. 2016. Distributed and parallel time series feature extraction for industrial big data applications. *CoRR* abs/1610.07717 (2016).

[8] Roberto Confalonieri, Ludovik Coba, Benedikt Wagner, and Tarek R Besold. 2021. A historical perspective of explainable Artificial Intelligence. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 11, 1 (2021), e1391.

[9] Francesco Del Buono, Francesca Calabrese, Andrea Baraldi, Matteo Paganelli, and Francesco Guerra. 2022. Novelty Detection with Autoencoders for System Health Monitoring in Industrial Environments. *Applied Sciences* 12, 10 (2022). https://doi.org/10.3390/app12104931

[10] Mengnan Du, Ninghao Liu, and Xia Hu. 2020. Techniques for interpretable machine learning. *Commun. ACM* 63, 1 (2020), 68–77.

[11] Duarte Folgado, Marília Barandas, Ricardo Matias, Rodrigo Martins, Miguel Carvalho, and Hugo Gamboa. 2018. Time Alignment Measurement for Time Series. *Pattern Recognition* 81 (2018), 268–279. https://doi.org/10.1016/j.patcog.2018.04.003

[12] Ricardo Fraiman, Badih Ghattas, and Marcela Svarc. 2013. Interpretable clustering using unsupervised binary trees. *Adv. Data Anal. Classif.* 7, 2 (2013), 125–145.

[13] Dino Ienco and Roberto Interdonato. 2020. Deep Multivariate Time Series Embedding Clustering via Attentive-Gated Autoencoder. In *PAKDD 2020 - 24th Pacific-Asia Conference on Knowledge Discovery and Data Mining (Advances in Knowledge Discovery and Data Mining)*. Singapore, Singapore. https://hal.inrae.fr/hal-02923636

[14] Gaoxia Jiang, Wenjian Wang, and Wenkai Zhang. 2019. A novel distance measure for time series: Maximum shifting correlation distance. *Pattern Recognition Letters* 117 (2019), 58–65.

[15] Yanfei Kang, Rob J. Hyndman, and Feng Li. 2020. GRATIS: GeneRAting TIme Series with diverse and controllable characteristics. *Stat. Anal. Data Min.* 13, 4 (2020), 354–376.

[16] Thai Le, Suhang Wang, and Dongwon Lee. 2020. GRACE: Generating Concise and Informative Contrastive Sample to Explain Neural Network Model's Prediction. In *KDD*. ACM, 238–248.

[17] Hailin Li. 2014. Asynchronism-based principal component analysis for time series data mining. *Expert systems with applications* 41, 6 (2014), 2842–2850.

[18] Hailin Li. 2019. Multivariate time series clustering based on common principal component analysis. *Neurocomputing* 349 (2019), 239–247.

[19] Hailin Li, Chunpei Lin, Xiaoji Wan, and Zhengxin Li. 2019. Feature representation and similarity measure based on covariance sequence for multivariate time series. *IEEE Access* 7 (2019), 67018–67026.

[20] Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei. 2021. Multivariate Time Series Anomaly Detection and Interpretation using Hierarchical Inter-Metric and Temporal Embedding. In *KDD*. ACM, 3220–3230.

[21] Yijuan Lu, Ira Cohen, Xiang Sean Zhou, and Qi Tian. 2007. Feature Selection Using Principal Feature Analysis. In *Proceedings of the 15th ACM International Conference on Multimedia* (Augsburg, Germany) *(MM '07)*. Association for Computing Machinery, New York, NY, USA, 301–304. https://doi.org/10.1145/1291233.1291297

[22] Ruizhe Ma and Rafal Angryk. 2017. Distance and density clustering for time series data. In *2017 IEEE international conference on data mining workshops (ICDMW)*. IEEE, 25–32.

[23] Laura Manduchi, Matthias Hüser, Julia Vogt, Gunnar Rätsch, and Vincent Fortuin. 2019. DPSOM: Deep probabilistic clustering with self-organizing maps. *arXiv preprint arXiv:1910.01590* (2019).

[24] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence* 267 (2019), 1–38.

[25] Christoph Molnar. 2019. *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/.

[26] Meinard Müller. 2007. Dynamic time warping. *Information retrieval for music and motion* (2007), 69–84.

[27] W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. 2019. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences* 116, 44 (2019), 22071–22080.

[28] Xuan Vinh Nguyen, Julien Epps, and James Bailey. 2009. Information theoretic measures for clusterings comparison: is a correction for chance necessary?. In *ICML (ACM International Conference Proceeding Series)*, Vol. 382. ACM, 1073–1080.

[29] Matteo Paganelli, Paolo Sottovia, Antonio Maccioni, Matteo Interlandi, and Francesco Guerra. 2020. Explaining data with descriptions. *Inf. Syst.* 92 (2020), 101549.

[30] John Paparrizos and Luis Gravano. 2015. k-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*. 1855–1870.

[31] Jayneel Parekh, Pavlo Mozharovskyi, and Florence d'Alché-Buc. 2021. A Framework to Learn with Interpretation. In *NeurIPS*. 24273–24285.

[32] Claudia Plant and Christian Böhm. 2011. INCONCO: interpretable clustering of numerical and categorical objects. In *KDD*. ACM, 1127–1135.

[33] Dragomir R. Radev, Hongyan Jing, Magorzata Sty, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Inf. Process. Manag.* 40, 6 (2004), 919–938.

[34] Andrew Rosenberg and Julia Hirschberg. 2007. V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure. In *EMNLP-CoNLL*. ACL, 410–420.

[35] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1, 5 (2019), 206–215.

[36] Martin Rätz, Amir Pasha Javadi, Marc Baranski, Konstantin Finkbeiner, and Dirk Müller. 2019. Automated data-driven modeling of building energy systems via machine learning algorithms. *Energy and Buildings* 202 (2019), 109384.

[37] Sandhya Saisubramanian, Sainyam Galhotra, and Shlomo Zilberstein. 2020. Balancing the Tradeoff Between Clustering Value and Interpretability. In *AIES*. ACM, 351–357.

[38] Davi Alberto Sala, Azarakhsh Jalalvand, Andy Van Yperen-De Deyne, and Erik Mannens. 2018. Multivariate Time Series for Data-Driven Endpoint Prediction in the Basic Oxygen Furnace. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 1419–1426. https://doi.org/10.1109/ICMLA.2018.00231

[39] S Yaser Samadi, Lynne Billard, MR Meshkani, and A Khodadadi. 2017. Canonical correlation for principal components of time series. *Computational Statistics* 32, 3 (2017), 1191–1212.

[40] Mattia Setzu, Riccardo Guidotti, Anna Monreale, Franco Turini, Dino Pedreschi, and Fosca Giannotti. 2021. GLocalX - From Local to Global Explanations of Black Box AI Models. *Artif. Intell.* 294 (2021), 103457.

[41] Han Lin Shang. 2014. A survey of functional principal component analysis. *AStA Advances in Statistical Analysis* 98, 2 (2014), 121–142.

[42] Shun-Yao Shih, Fan-Keng Sun, and Hung-Yi Lee. 2019. Temporal pattern attention for multivariate time series forecasting. *Mach. Learn.* 108, 8-9 (2019), 1421–1441.

[43] Ashish Singhal and Dale E Seborg. 2005. Clustering multivariate time-series data. *Journal of Chemometrics: A Journal of the Chemometrics Society* 19, 8 (2005), 427–438.

[44] Lars Stahle and Svante Wold. 1989. Analysis of variance (ANOVA). *Chemometrics and Intelligent Laboratory Systems* 6, 4 (1989), 259–272.

[45] Douglas Steinley. 2004. Properties of the Hubert-Arable Adjusted Rand Index. *Psychological methods* 9, 3 (2004), 386.

[46] Donato Tiano, Angela Bonifati, and Raymond Ng. 2021. FeatTS: Feature-based Time Series Clustering. In *SIGMOD 2021*. 2784–2788.

[47] Donato Tiano, Angela Bonifati, and Raymond Ng. 2021. Feature-driven Time Series Clustering.. In *EDBT*. 349–354.

[48] Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605.

[49] Chenxiao Xu, Hao Huang, and Shinjae Yoo. 2021. A Deep Neural Network for Multivariate Time Series Clustering with Result Interpretation. In *IJCNN*. IEEE, 1–8.

[50] Wei Zhang, Xiaowei Dong, Huaibao Li, Jin Xu, and Dan Wang. 2020. Unsupervised Detection of Abnormal Electricity Consumption Behavior Based on Feature Engineering. *IEEE Access* 8 (2020), 55483–55500.

[51] Zijian Zhang, Koustav Rudra, and Avishek Anand. 2021. Explain and Predict, and then Predict Again. In *WSDM*. ACM, 418–426.

[52] Alexandra Zytek, Ignacio Arnaldo, Dongyu Liu, Laure Berti-Équille, and Kalyan Veeramachaneni. 2022. The Need for Interpretable Features: Motivation and Taxonomy. *SIGKDD Explor.* 24, 1 (2022), 1–13.