Information Sciences 569 (2021) 596–614

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Multivariate times series classification through an interpretable representation

Francisco J. Baldán [a,*], José M. Benítez [a]

[a] Department of Computer Science and Artificial Intelligence, University of Granada, DICITS, iMUDS, DaSCI, 18071 Granada, Spain

ARTICLE INFO

Article history:
Received 7 September 2020
Received in revised form 15 April 2021
Accepted 11 May 2021
Available online 18 May 2021

Keywords:
Multivariate
Time series features
Complexity measures
Time series interpretation
Classification

ABSTRACT

Multivariate time series classification is a machine learning task with increasing importance due to the proliferation of information sources in different domains (economy, health, energy, crops, etc.). Univariate methods lack the ability to capture the relationships between the different variables that compose a multivariate time series and therefore cannot be directly extrapolated to multivariate environments. Despite the good performance and competitive results of the multivariate proposals published to date, they are hard to interpret due to their high complexity. In this paper, we propose a multivariate time series classification method based on an alternative representation of the time series, composed of a set of 41 descriptive time series features, in order to improve the interpretability of time series and results obtained. Our proposal uses traditional classifiers over the extracted features to look for relationships between the different variables that form a multivariate time series. We have selected four state-of-the-art algorithms as base classifiers to evaluate our method. We have tested our proposal on the complete University of East Anglia repository, obtaining highly interpretable results capable of explaining the relationships between the features that compose the time series and achieving performance results statistically indistinguishable from the best algorithms of the state-of-the-art.

1. Introduction

Nowadays, large amounts of data are generated. Everything is increasingly interconnected, more and more sensors are included in everything around us, and these monitor the behavior of any event of interest over time. These sensors generate lots of data as multivariate time series (MTS). A key task in the analysis and mining of these data is multivariate time series classification (MTSC), which aims to give an accurate response to a large number of problems: e.g. from detecting when a patient is sick or has an anomaly in his heart behavior [28], or if a driver is in optimal condition to drive [26], the recognition of human activities [37], the occupation of an office room based on environmental information [12], the wind speed forecasting [35] or how to adapt energy production based on particular circumstances [24].

The field of MTSC can be divided into two main types of work. Firstly, applied works that seek to obtain a better solution for a given problem, offering ad hoc proposals considering the peculiarities of the treated problem [11,25,31]. Secondly, proposals that deal with MTS in a general way but taking into account possible interrelations between the different variables available [2,17,33,43]. The proposals in the later group are usually based on strong theoretical foundations. A relatively large

* Corresponding author.
E-mail addresses: fjbaldan@decsai.ugr.es (F.J. Baldán), J.M.Benitez@decsai.ugr.es (J.M. Benítez).

https://doi.org/10.1016/j.ins.2021.05.024
0020-0255/

number of proposals for MTSC can be found in the literature [7,16,19,36]. Most of them are guided towards obtaining increasing levels of accuracy. However, eXplainable Artificial Intelligence (XAI) [14] is a topic enjoying a growing level of interest. Its goal is to build accurate intelligent system for complex tasks, but also paying special attention to their interpretability. The built systems or the way they make decisions are required to be easy to understand for human beings. Thus top accuracy is no longer the only objective and interpretability receives higher attention. This also applies to solutions for classification problems.

In the field of MTSC, there are few proposals that pay attention to the interpretability of results [18]. Given the complexity of the problem, most proposals are focused on obtaining the best results in terms of accuracy. Even the proposals based on shapelets [4,42], which are interpretable from their univariate origins, have chosen to use the Transformed Shapelets in multivariate environments [9] or proposals that are even less interpretable [20], giving priority to accuracy results over interpretability. One possible way to pave the path towards easier to understand solutions to MTSC is expressing time series in different domains. Perhaps in terms of descriptive features instead of the raw time domain values.

In this paper, we present a new MTSC approach based on the representation of time series through a set of features and measures. This approach allows transforming the original MTSC problem into a traditional classification problem, enabling to apply the whole set of the traditional classification algorithms. The proposal is mainly focused on obtaining interpretable classifiers, so that, an end user can understand and feel more confident with the obtained systems. In addition, the approach allows to obtain acceptable accuracy results with respect to the main techniques of the state-of-the-art.

The remainder of this paper is organized as follows: Section 2 introduces the state of the art in MTSC. Section 3 describes in depth our proposal. Section 4 shows the experimental study conducted, and the results obtained. Section 5 discusses the interpretability of our proposal. Finally, Section 6 concludes the paper.

2. Related work

In the field of MTSC, proposals from methods that have demonstrated good behavior in univariate cases predominate. Some of the first proposals for MTSC were multivariate extensions of the distance-based algorithm 1NN-DTW [22,40], given its simplicity and good results in univariate environments. These proposals are a good starting point, but they carry the limitations they already had in univariate environments, such as high computational complexity and low interpretability, since they only indicate how much the instances are similar to each other. To these limitations, we must add that in a multivariable environment, the first proposals of 1NN-DTW processed each variable of each time series independently, so they were not able to extract information from the relationship between the different variables that make up each multivariate time series. With this in mind, we can say that multiple proposals for a multivariate DTW have been made, such as dependent (DTW_D) and independent (DTW_I) warping, both having the same performance [38]. Other proposals such as Mahalanobis Distance-based Dynamic Time Warping measure (MDDTW) [32] seek to give a general answer to this problem. MDDTW is able to precisely calculate the relationship between the different variables that compose an MTS. This, together with the alignment obtained by DTW, allows obtaining very competitive results.

The feature-based approach has multiple proposals, giving special importance to the extraction of additional information and to the speed of processing, especially when compared to similarity-based techniques. In this field we can differentiate between proposals based on shapelets and bag-of-words. In the field of shapelets, Generalized Random Shapelets Forests (gRSF) [21] is considered the state-of-the-art, obtaining better results than its direct competitor, Ultra Fast Shapelets (UFS) [41]. gRSF is based on the creation of a set of shapelet-based decision trees from a random extraction of the shapelets. In the field of bag-of-words, Word ExtrAction for time SEries cLassification plus Multivariate Unsupervised Symbols and dErivatives (WEASEL + MUSE) [36] is considered the state-of-the-art, as it obtains the best results against its direct competitors: Learned Pattern Similarity (LPS) [8], AutoRegressive Forests for multivariate time series modelling (mv-ARF) [39], Symbolic representation for Multivariate Time Series classification (SMTS) [7], and gRSF. All of them have been tested on one of the first reference MTS database collected from [6], with a total of 20 MTSC datasets. WEASEL + MUSE extracts a vector of features by applying a sliding-window to each variable of the MTSC and filtering out non-discriminative features, finally a classifier analyses these data.

In the field of deep learning, the extension of the Long Short Term Memory Fully Convolutional Network (LSTM-FCN) and Attention LSTM-FCN (ALSTM-FCN) [19] to a multivariate environment, including a squeeze-and-excitation block in the fully convolutional block that improves accuracy. This proposal improved the WEASEL + MUSE results over the original database of 20 datasets [6] extended with 10 datasets from the UC Irvine Machine Learning Repository (UCI) [15] and 6 datasets used by Pei et al. [34]. Also, we find proposals that pay attention to detect natural features of time series such as trend [27].

A new proposal has recently emerged, Local Cascade Ensemble for Multivariate Data Classification (LCE) and its extension for Multivariate Time Series (LCEM) [16]. LCE and LCEM are a hybrid ensemble method with 2 major objectives. The first one is to handle the bias-variance tradeoff by an explicit boosting-bagging approach. The second one is to individualize classifier errors on different parts of the training data by an implicit divide-and-conquer approach. This proposal is outlined as the new state-of-the-art in MTSC by obtaining better results than the previous state-of-the-art MLSTM-FCN and WEASEL + MUSE, on the University of East Anglia (UEA) repository [3], a new repository for MTSC composed of 30 datasets that is becoming increasingly important.

In contrast to state-of-the-art methods, we propose a method that obtains essential features of each variable and each MTS and applies a transformation to the MTS dataset, obtaining a traditional classification problem based on attributes. All traditional classification algorithms can be applied to this new dataset, and depending on the applied algorithms, interpretable results can be obtained to explain the problem or results of higher accuracy.

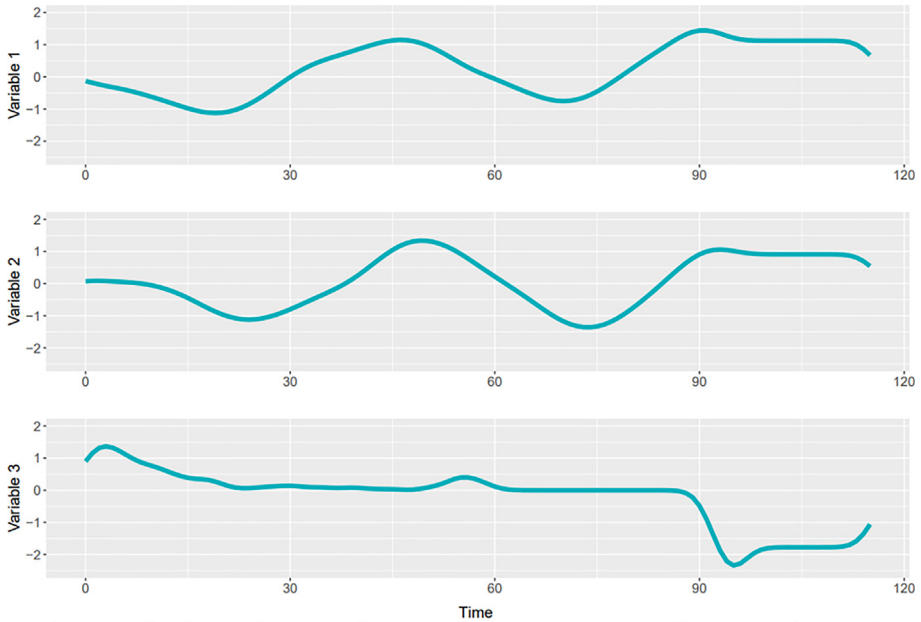
3. Multivariable times series classification through an interpretable representation

In this work we propose a method that allows the calculation of complexity measures to be applied to MTSC problems. Our proposal, namely Complexity Measures and Features for Multivariate Time Series (CMFMTS), is based on the idea that a time series can be faithfully represented with a set of complexity measures and descriptive features [5]. Furthermore, these features preserve most of the information content of the series to such an extend that they can be used to classify the series.

The following is an example of the calculation of some features on an MTS with three variables. In Table 1, we show some features highly related to the nature of the time series and its range of possible values. In Fig. 1, we show a simple example of the feature computation used and its interpretability. In the first place, we can see how variables 1 and 2 are similar, so we can expect values of the features also similar to each other. This is reflected in the values of kurtosis and skewness. If variables 1 and 2 have similar values their probability distribution will be similar and therefore their values of kurtosis and skewness. We can appreciate a significant difference concerning variable 3. In the three variables, we can see the existence

Table 1
Example of some features used.

Char.	Name	Description	Range
F_1	curvature	Calculated based on the coefficients of an orthogonal quadratic regression	$(-\infty, \infty)$
F_2	kurtosis	The "tailedness" of the probability distribution	$(-\infty, \infty)$
F_3	linearity	Calculated based on the coefficients of an orthogonal quadratic regression	$(-\infty, \infty)$
F_4	shannon_entropy_cs	Chao-Shen entropy estimator	$[0, \infty)$
F_5	skewness	Asymmetry of the probability distribution	$(-\infty, \infty)$
F_6	trend	Strength of trend	$[0, 1]$



Variable	curvature	kurtosis	linearity	shannon_entropy_cs	skewness	trend
1	0.0862	−1.4715	5.5504	2.1907	−0.1657	0.9958
2	1.9031	−1.2722	2.4158	2.1899	−0.359	0.9959
3	−3.3814	0.5522	−8.2628	1.5649	−1.269	0.9813

Fig. 1. Example of feature extraction from an MTS with 3 variables.

of a single trend, for this reason, the trend values are close to 1 in all cases. The oscillations present in the variables 1 and 2 seem more typical of seasonal patterns that do not affect the trend of the time series. To evaluate the Chao-Shen shannon entropy (shannon_entropy_cs) we have to appreciate the evolution of variables 1, 2, and 3. Variables 1 and 2 show a certain pattern, while variable 3 shows a long period without changes with a final reduction of the value never seen before. For this reason, it obtains a higher value in shannon entropy very far from the one obtained by variables 1 and 2. We also analyze the values of curvature and linearity. Given the evolution and shape of the three variables and the perceptible linear relationship between the current values of variables 1 and 2 with their corresponding past values, it is logical to expect positive and similar values of curvature and linearity for variables 1 and 2. On the other hand, variable 3 does not show these forms or a linear relationship between its present and past values, so it obtains negative values that are far from curvature and linearity concerning what is obtained by variables 1 and 2.

Fig. 2 shows the workflow of our proposal. First, a set of n multivariate time series is assumed, each consisting of m variables (Fig. 2.1). Individually, each one of the variables that compose each time series is processed, obtaining the j features for each variable (Fig. 2.2). A dataset is obtained with $n \times m$ rows and j columns, where each row is composed of the set of features processed on each time series that compose each MTS (Fig. 2.3). Finally, this dataset is processed by placing all the variables of the same MTS in the same row (Fig. 2.4). In this way, a new (transformed) dataset is obtained where all the features of all the variables that compose the same MTS are placed in the same row, forming part of the same instance. This enables the search for patterns and relationships of interest among the different variables that compose the same MTS (Fig. 2.5).

Although the feature calculation based approach can be applied to all types of automatic learning problems as well as supervised, unsupervised, semi-supervised learning, etc. In the supervised case, simple and fast comparisons can be made with respect to the main state-of-the-art algorithms. Due to the great variety of the processed time series, it is possible that undesired values are obtained for some of the proposed features. For example, to calculate the autocorrelation function (ACF) [5] concerning the values delayed 10 instants of time it is necessary that our time series has a minimum length of 11, otherwise, we will obtain an Not Available (NA). Time series with a single value are another problematic case since features like kurtosis and skewness are not defined for these cases and would return Not a Number (NaN) values. Time series containing NA generate problems internally in some of the features used (acf, kurtosis, skewness, shannon_entropy_cs, etc.) returning NA values in those features. Finally, there are features that can obtain values in the range $(-\infty, \infty)$. Extreme values close to the limits are considered as undesirable since they generate several problems in the different algorithms applied later. To deal with these cases, we have specified a preprocessing stage, following the calculation of the features and their correct ordering, which solves the possible inconveniences generated by these cases. The whole process is depicted in Algorithm 1.

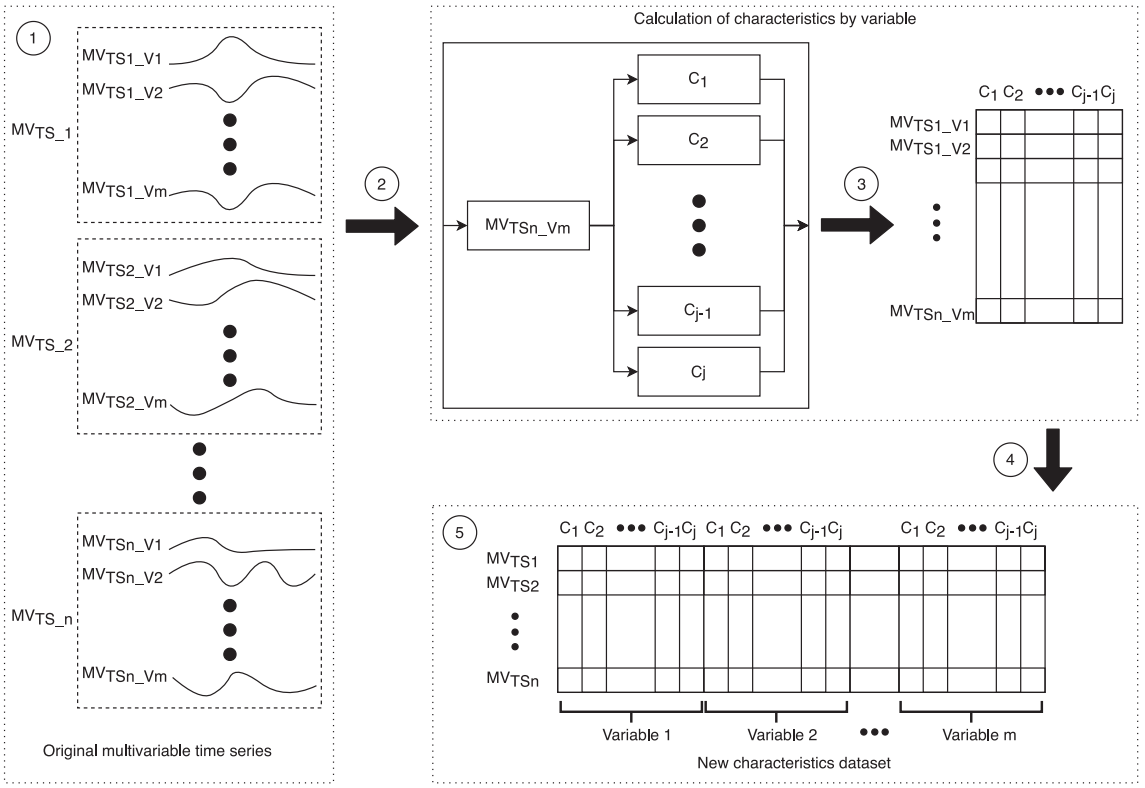


Fig. 2. Features calculation workflow.

Algorithm 1 Preprocessing procedure

```
Input:
  train: train dataframe with (Ts_id, Ts_dimId, Ts_class, Ts_values)
  test: test dataframe with (Ts_id, Ts_dimId, Ts_class, Ts_values)
  models: list of models to be processed
Output:
  output_data: a triplet that contains the fitted models, the vectors with the predicted labels and the accuracies
 obtained
  mvf_train: features train dataframe
  mvf_test: features test dataframe
 1: mvf_train, mvf_test ← calc_mvcmfts((train, test), all)
 2: for each value in (mvf_train, mvf_test) do
 3:     if (is.na(value) || is.nan(value) || is.infinite(value)) value then ← NA end if
 4: end for
 5: for each column in mvf_train do
 6:     if sum(!is.na(colum.values)) == 0) then
 7:         mvf_train.delete(column)
 8:         mvf_test.delete(column)
 9:     end if
10: end for
11: for each column in (mvf_train, mvf_test) do
12:     for each value in column do
13:         if is.na(value) value then ← mean(column) end if
14:     end for
16: end for
16: for each column in mvf_train do
17:     if (length(unique(column)) <= 1) then
18:         mvf_train.delete(column.index)
19:         mvf_test.delete(column.index)
20:     end if
21: end for
22: output_data ← NULL
23: for each model in models do
24:     fit ← train.model(mvf_train, train.Ts_class)
25:     pred ← fit.predict(mvf_test)
26:     acc ← accuracy(pred, test.Ts_class)
27:     output_data.add(fit, pred, acc)
28: end for
29: return (output_data, mvf_train, mvf_test)
```

The starting point is the calculation of the proposed features in the training and test sets (Line 1). For any of the cases mentioned above in which an undesired value has been obtained, these values are unified under a single NA identifier (Lines 2–4). We check on the training set if any column lacks interest because it is full of undesired values. If so, this column is removed from both the training set and the test set (Lines 5–10). In order to simplify the treatment of missing values, we have chosen to impute these values with the average of their respective column (Lines 11–15). There are better imputation techniques, but we do not address that task in this paper and the considered one has proved to be effective enough. To avoid the use of variables without information, we analyzed the training set looking for variables with a single value. If any variable with this condition is found, it is eliminated from both the training set and the test set (Lines 16–21). Finally, each of the specified models is processed, obtaining the desired model fit, its prediction on the test set and the accuracy achieved (Lines 23–28). These data are returned to the user, together with the datasets transformed to the features of our proposal (Line 29).

Finally, once we have explained our proposal and its application in a real environment, we can list the main advantages offered by this approach:

- Allows the use of the application of any vector-based classification method, since after the applied transformation, we obtain a traditional dataset where each instance is represented by its corresponding attributes (features).
- Allows the use of machine learning methods based on different paradigms: supervised, semi-supervised, self-supervised, unsupervised, etc., since it obtains a vector-based dataset, where each instance is composed of different attributes.
- Handles easily datasets of time series with varying lengths, as it processes each time series individually.

- Decisions made can be easily understood by human experts, since the features used explain the behavior of the time series. In addition, the represented concepts by the selected features are interpretable by the users.

3.1. Computational complexity

Our proposal is composed of two main steps: the calculation of the feature value set and the construction of the classifier. Since the steps have to be executed sequentially, the complexity of the whole process can be computed by simply adding the complexity of each one. For the sake of convenience, the variables used in the formulas are: f, number of features computed; n, number of time series in a dataset; v, the number of variables in a multivariate time series; and l, the length of the time series.

The computing of the feature values is defined as a sequential process, each feature computing at one time, although if enough threads are available, all of them can be computed in parallel. Thus the overall complexity, stated in O notation, is equal to the feature with the highest complexity. The computation of the complexity of all of them is quite straightforward. Two of them have the greatest complexity, namely approximation entropy and sample entropy. For a univariate time series, those features have a computational time complexity of $O(l^2)$ [29,30], each one. Based on the feature computation process shown in Fig. 2, we can conclude that our proposal has a computational complexity of $O(n \cdot v \cdot f \cdot l^2)$. As for the classification models used in our proposal, each one has a different computational complexity. In Table 2, we show the computational complexity of typical models. The final computational complexity of our proposal results from the addition of the feature computation complexity, seen above, and the complexity of the model used, as shown in Table 2.

The computational complexity of our proposal is very similar to the one offered by the main state-of-the-art algorithm. LCEM has a computational complexity $O(N \cdot s \cdot d \cdot D \cdot 2^D \cdot T_{Base})$, where d is the number of attributes, d' is the number of attributes in RF subset of attributes, D is the maximum depth of a tree, s is the number of samples, N is the number of trees, and T_{Base} is the time complexity of the base classifier. To this complexity, we must add the complexity of the applied transformation, which linearly grows in complexity with the number of samples.

One of the main advantages of our proposal is that the time complexity of the feature extraction process scales linearly with the number of time series to be processed and is trivial to parallelize. In addition, we have not considered hyperparameter optimization for feature computation, unlike LCEM proposal or the models used, so the procedure is kept simple and does not increase the computational complexity. Furthermore, the feature extraction process is applied independently to each multivariate time series, so we can compute these characteristics directly as the time series are received, and we do not need to have a complete set of time series to start processing them. Again, the process can be trivially parallelized. This allows us, in the best case, to reduce the computational complexity of our proposal to the complexity of the classification model used.

4. Empirical study

To assess the effectiveness of our proposal, we have developed a detailed empirical study. We start by explaining the experimental design (Section 4.1), followed by the results obtained (Section 4.2). The analysis of the interpretability of the models is important enough so that a complete section is devoted to them, namely Section 5.

4.1. Experimental design

We describe the performance measures used to evaluate our proposal (Section 4.1.1), followed by the datasets used (Section 4.1.2) and the machine learning models selected (Section 4.1.3). Finally, we describe the hardware and software used in the development of our proposal (Section 4.1.4).

4.1.1. Performance measures

Since the datasets come from very different fields, we have opted for a ranking performance measure. We have selected the average rank as a comparative method from the accuracy calculation on the original training and test sets. The accuracy has been calculated as the number of instances correctly classified divided by the total number of instances of the test set. To obtain robust results from a statistical point of view, we have executed each experiment 100 times, —using different random seeds—, and calculated the mean of the obtained values. Also, we have included the Win/Loss/Tie ratio to quantify the number of cases in which each model and approach wins, loses, or ties concerning the best case. Since the range of possible results is wide, we have opted to include a Critical Difference diagram (CD) [13], and the Wilcoxon Signed-Rank test. CD shows the results of a statistical comparison between all models in pairs based on average ranks. Models that are connected by a bold line do not have a statistically significant difference for a particular confidence level. The Wilcoxon Signed-Rank test allows us to assess whether two models offer statistically distinguishable results at a particular confidence level, compared between all models in pairs based on the accuracies. In our case, we have set an α of 0.05 for a 95% confidence level. The average rank and the CD were obtained using the R *scmamp* package. The Wilcoxon Signed-Rank test results were obtained using the R *stats* package.

Table 2
Computational complexity of typical models. Notation: n, number of samples, t, number of randomized trees; k number of attributes randomly included at each node; h, height of a tree; m, number of attributes; c, number of classes.

Models	Time complexity
Random Forest	$O(0.632 \cdot n \cdot t \cdot k \cdot \log(0.632 \cdot n))$ [4]
C5.0 with Boosting	$O(h \cdot m \cdot (n \cdot c + n \cdot \log(n))$ [23]
Support Vector Machine	$O(n^3)$ [1]
1-Nearest Neighbor	$O(n \cdot v \cdot f)$

4.1.2. Datasets

To evaluate the performance of our proposal on problems of all kinds, we have selected the main repository of MTSC problems, the UEA multivariate time series classification archive. In Table 3, we show the characteristics of the 30 datasets of the UEA repository: number of instances of the training and test sets, length of the time series, number of variables of each MTS, and number of classes. Some of these datasets are composed of time series of different lengths, so the repository chose to pad with NA values. In our case, we have removed those values. We have processed the values of the time series that contain information without affecting the original values of each time series.

4.1.3. Models

For our proposal, we have selected a set of traditional models with two main approaches: to obtain interpretable results and to obtain the best classification results by sacrificing interpretability [5]. These models are C5.0 with Boosting (C5.0B), Random Forest (RF), Support Vector Machine (SVM), and 1-Nearest Neighbors with Euclidean Distance (1NN-ED). For this last model, we have applied a normalization between [0, 1]. This set of models will be applied to the set of time series features obtained by our proposal. The final models of our proposal are obtained from the union of the transformed datasets with the four models previously commented. These proposals are: CMFMTS + C5B, CMFMTS + RF, CMFMTS + SVM, and CMFMTS + 1NN-ED. We have simplified the CMFMTS nomenclature by CMFM due to space limitations in later tables. On the other hand, we have selected the main state-of-the-art MTSC models:

- 1-Nearest Neighbor classifier with Euclidean distance (1NN-ED), with and without normalization.
- 1-Nearest Neighbor classifier based on multi-dimensional points (DTW-1NN-D) [38], with and without normalization.
- 1-Nearest Neighbor classifier based on the sum of DTW distance for each dimension (DTW-1NN-I) [38], with and without normalization.
- Multivariate LSTM Fully Convolutional Networks for Time Series Classification (MLSTM-FCN) [19] with the settings specified by their authors: 128–256-128 filters, 250 training epochs, a dropout of 0.8, and a batchsize of 128.
- Word ExtrAction for time SEries cLassification plus Multivariate Unsupervised Symbols and dErivatives (WEASEL + MUSE) [36] with the settings specified by their authors: SFA word lengths l in [2,4,6], windows length in [4:max(MTSlength)], chi = 2, bias = 1, p = 0.1, c = 5 and a solver equals to L2R LR DUAL.
- Local Cascade Ensemble for Multivariate data classification (LCEM) [16], optimized hyper-parameters for each dataset (Windows (%), Trees, and Depth). The results have been obtained from the published work of the authors.
- Random Forest for Multivariate (RFM) algorithm, from the sklearn library, applied to the transformation proposed in the LCEM paper [16].
- Extreme Gradient Boosting for multivariate (XGBM), Extreme Gradient Boosting algorithm, from the xgboost library, applied to the transformation proposed in the LCEM paper [16].

The results of the algorithms mentioned above have been obtained from [16].

4.1.4. Hardware and software

The experimentation carried out in this work was performed in a server with the following hardware: 4 × Intel(R) Xeon (R) CPU E5-4620 0 @ 2.20 GHz processors, 8 cores per processor with HyperThreading, 10 TB HDD, 512 GB RAM. The server software configuration comprises Ubuntu 18.04 and R 3.6.3.

The source code of our proposal can be found in the online repository.[1]

4.2. Results

We start by analyzing the accuracy and the average rank results on the 30 processed datasets. Table 4 shows the accuracy results obtained by our proposal against the main state-of-the-art algorithms. The NA values refer to cases in which for any reason (memory overflow, libraries limitations, etc.), a model has not been obtained correctly, and it has been impossible to

[1] Complexity Measures and Features for Multivariate Times Series classification.https://github.com/fjbaldan/CMFMTS/

Table 3
Datasets information from the UEA repository.

Dataset	Train	Test	Length	Dims	Class
ArticularyWordRecognition	275	300	144	9	25
AtrialFibrillation	15	15	640	2	3
BasicMotions	40	40	100	6	4
CharacterTrajectories	1422	1436	60–182	3	20
Cricket	108	72	1197	6	12
DuckDuckGeese	50	50	270	1345	5
EigenWorms	128	131	17984	6	5
Epilepsy	137	138	206	3	4
ERing	30	270	65	4	6
EthanolConcentration	261	263	1751	3	4
FaceDetection	5890	3524	62	144	2
FingerMovements	316	100	50	28	2
HandMovementDirection	160	74	400	10	4
Handwriting	150	850	152	3	26
Heartbeat	204	205	405	61	2
InsectWingbeat	25000	25000	2–22	200	10
JapaneseVowels	270	370	7–29	12	9
Libras	180	180	45	2	15
LSST	2459	2466	36	6	14
MotorImagery	278	100	3000	64	2
NATOPS	180	180	51	24	6
PenDigits	7494	3498	8	2	10
PEMS-SF	267	173	144	963	7
PhonemeSpectra	3315	3353	217	11	39
RacketSports	151	152	30	6	4
SelfRegulationSCP1	268	293	896	6	2
SelfRegulationSCP2	200	180	1152	7	2
SpokenArabicDigits	6599	2199	4–93	13	10
StandWalkJump	12	15	2500	4	3
UWaveGestureLibrary	120	320	315	3	8

perform the desired classification. As we can see in Table 4, our proposal CMFMTS + RF, called CMFM + RF for simplification, obtains the best results among the four models we have proposed: CMFM + C5.0B, CMFM + RF, CMFM + SVM, and CMFM + 1NN-ED. The CMFM + C5.0B model is especially interesting for cases where a simple and easy to interpret classifier is required, and that offers results close to the optimum ones as it happens in the datasets: Epilepsy and LSST. We can find cases in which CMFM + RF does not offer the best results among these four models, and it may be interesting to try other combinations as it happens in the datasets: AtrialFibrilation, EthanolConcentration, FaceDetection, HandMovementDirection, etc.

If we compare our proposal with the rest of the state-of-the-art algorithms, we can see how CMFM + RF obtains an average rank of 7.1, close to the one obtained by LCEM (4.23), RFM (5.18), MLSTM-FCN (5.33), and WEASEL + MUSE (5.93). We have included two decimals for the average rank so that the differences shown in Fig. 3 can be better appreciated. If we observe the Win/Loss/Tie ratio, we can see that MLSTM-FCN obtains the best results in 11 datasets, followed by LCEM, which wins in 8 datasets, and CMFM + RF and RFM, which obtains the best results in 5 datasets. These behaviors are reflected in the CD diagram shown in Fig. 3. This diagram shows that there is no statistically significant difference, for an α of 0.05, between the previously mentioned models, in addition to the DTW-1NN-D model. In Table 5, we include the p-values obtained by the Wilcoxon Signed-Rank test of all the accuracies pairs of our models and the best models of the state-of-the-art in the first union line in Fig. 3. For our CMFM + RF model, all the p-values are higher than the significance level 0.05, so we cannot reject the null hypothesis, in which the results of each model come from the same population. Those results indicate that our CMFM + RF proposal offers results that are statically indistinguishable from those obtained by the main state-of-the-art algorithms. For the rest of our models, we can see that the null hypothesis is rejected except in the cases CMFM + C5.0B with DTW-1NN-D and DTW-1NN-D (norm), and CMFM + SVM with WEASEL + MUSE, DTW-1NN-D, and DTW-1NN-D (norm). These results reinforce CMFM + RF as our best model. If we analyze the median and average accuracy values of Table 4, we can see that our proposal obtains competitive results. The NA values have been transformed to 0 for the calculation made.

Analyzing the results obtained for some specific cases, we can appreciate significant differences between the different proposals. For example, in the DuckDuckGeese dataset, the MLSTM-FCN algorithm obtains 7.5 points of difference with the next best result. LCEM and similar proposals obtain results with significant differences concerning the rest of the methods, as can be seen in the HandMovementDirection dataset. In the ERing dataset, we can see a big difference between our CMFM + Any proposals and the rest of the algorithms. These cases confirm the idea that in the field of CMTS, the results are strongly linked to the data itself and the approach used. It is especially complicated to find an approach that is able to face all kinds of problems with optimal results or close to them.

Based on the results shown in this section, we can conclude that:

Table 4
Accuracy results on the UEA repository datasets: accuracy (%), average accuracy, median, average rank, and Win/Loss/Tie Ratio. The best results are stressed in bold.

Part 1:

Datasets	CMFM + C5.0B	CMFM + RF	CMFM + SVM	CMFM + 1NN-ED	LCEM	XGBM	RFM	MLSTM-FCN
ArticularyWordRecognition	92.0	98.8	97.3	98.7	**99.3**	99.0	99.0	98.6
AtrialFibrillation	6.7	19.1	26.7	26.7	**46.7**	40.0	33.3	20.0
BasicMotions	90.0	98.2	97.5	97.5	**100.0**	**100.0**	**100.0**	**100.0**
CharacterTrajectories	94.2	97.0	97.0	93.3	97.9	98.3	98.5	**99.3**
Cricket	86.1	97.7	95.8	98.6	98.6	97.2	98.6	98.6
DuckDuckGeese	54.0	51.0	42.0	46.0	37.5	40.0	40.0	**67.5**
EigenWorms	84.7	89.5	84.7	77.9	52.7	55.0	**100.0**	80.9
Epilepsy	99.3	**99.9**	97.8	96.4	98.6	97.8	98.6	96.4
ERing	80.7	**93.1**	93.0	90.4	20.0	13.3	13.3	13.3
EthanolConcentration	22.1	26.0	26.6	22.8	37.2	42.2	**43.3**	27.4
FaceDetection	55.7	55.7	58.3	50.3	61.4	**62.9**	61.4	55.5
FingerMovements	50.0	50.1	46.0	49.0	59.0	53.0	56.0	**61.0**
HandMovementDirection	17.6	24.5	28.4	17.6	**64.9**	54.1	50.0	37.8
Handwriting	17.2	27.4	18.7	23.4	28.7	26.7	26.7	54.7
Heartbeat	76.1	76.8	72.7	62.4	76.1	69.3	**80.0**	71.4
InsectWingbeat	NA	**67.7**	10.0	26.6	22.8	23.7	22.4	10.5
JapaneseVowels	79.5	83.7	77.8	69.5	97.8	96.8	97.0	**99.2**
Libras	82.2	84.7	81.7	80.0	77.2	76.7	78.3	**92.2**
LSST	65.2	**67.3**	65.6	50.4	65.2	63.3	61.2	64.6
MotorImagery	49.0	50.3	50.0	40.0	**60.0**	46.0	55.0	53.0
NATOPS	87.2	83.5	80.0	76.1	91.6	90.0	91.1	**96.1**
PEMS-SF	91.3	**99.9**	66.5	78.0	94.2	98.3	98.3	65.3
PenDigits	93.7	95.2	95.9	93.7	97.7	95.1	95.1	**98.7**
PhonemeSpectra	22.8	28.4	24.7	17.1	**28.8**	18.7	22.2	27.5
RacketSports	73.0	80.6	80.9	69.7	**94.1**	92.8	92.1	88.2
SelfRegulationSCP1	81.6	82.0	77.1	70.0	83.9	82.9	82.6	**86.7**
SelfRegulationSCP2	48.3	41.8	45.0	46.1	**55.0**	48.3	47.8	52.2
SpokenArabicDigits	93.3	97.6	97.9	91.5	97.3	97.0	96.8	**99.4**
StandWalkJump	26.7	36.3	26.7	20.0	40.0	33.3	**46.7**	**46.7**
UWaveGestureLibrary	65.0	77.5	72.8	74.4	89.7	89.4	90.0	85.7
Mean	62.8	**69.4**	64.5	61.8	69.1	66.7	69.2	68.3
Median	74.6	79.1	72.8	69.6	70.7	66.3	**79.2**	69.5
Average Rank	10.25	7.1	9.25	11.37	**4.23**	6.67	5.18	5.33
Win/Loss/Tie Ratio	0/30/0	5/25/0	0/30/0	0/30/0	8/22/2	2/28/1	5/25/2	**11/19/2**

Part 2:

Datasets	WEASEL + MUSE	ED-1NN	DTW-1NN-I	DTW-1NN-D	ED-1NN (norm)	DTW-1NN-I(norm)	DTW-1NN-D(norm)
ArticularyWordRecognition	**99.3**	97.0	98.0	98.7	97.0	98.0	98.7
AtrialFibrillation	26.7	26.7	26.7	20.0	26.7	26.7	22.0
BasicMotions	**100.0**	67.5	**100.0**	97.5	67.6	**100.0**	97.5
CharacterTrajectories	99.0	96.4	96.9	99.0	96.4	96.9	98.9
Cricket	98.6	94.4	98.6	**100.0**	94.4	98.6	**100.0**
DuckDuckGeese	57.5	27.5	55.0	60.0	27.5	55.0	60.0
EigenWorms	89.0	55.0	60.3	61.8	54.9	NA	61.8
Epilepsy	99.3	66.7	97.8	96.4	66.6	97.8	96.4
ERing	13.3	13.3	13.3	13.3	13.3	13.3	13.3
EthanolConcentration	31.6	29.3	30.4	32.3	29.3	30.4	32.3
FaceDetection	54.5	51.9	51.3	52.9	51.9	NA	52.9
FingerMovements	54.0	55.0	52.0	53.0	55.0	52.0	53.0
HandMovementDirection	37.8	27.9	30.6	23.1	27.8	30.6	23.1
Handwriting	53.1	37.1	50.9	**60.7**	20.0	31.6	28.6
Heartbeat	72.7	62.0	65.9	71.7	61.9	65.8	71.7
InsectWingbeat	NA	12.8	NA	11.5	12.8	NA	NA
JapaneseVowels	97.8	92.4	95.9	94.9	92.4	95.9	94.9
Libras	89.4	83.3	89.4	87.2	83.3	89.4	87.0
LSST	62.8	45.6	57.5	55.1	45.6	57.5	55.1
MotorImagery	50.0	51.0	39.0	50.0	51.0	NA	50.0
NATOPS	88.3	85.0	85.0	88.3	85.0	85.0	88.3
PEMS-SF	NA	70.5	73.4	71.1	70.5	73.4	71.1
PenDigits	96.9	97.3	93.9	97.7	97.3	93.9	97.7
PhonemeSpectra	19.0	10.4	15.1	15.1	10.4	15.1	15.1
RacketSports	91.4	86.4	84.2	80.3	86.8	84.2	80.3

Table 4 (*continued*)

				Part 1:			
SelfRegulationSCP1	74.4	77.1	76.5	77.5	77.1	76.5	77.5
SelfRegulationSCP2	52.2	48.3	53.3	53.9	48.3	53.3	53.9
SpokenArabicDigits	98.2	96.7	96.0	96.3	96.7	95.9	96.3
StandWalkJump	33.3	20.0	33.3	20.0	20.0	33.3	20.0
UWaveGestureLibrary	**90.3**	88.1	86.9	**90.3**	88.1	86.8	**90.3**
Mean	64.3	59.1	63.6	64.3	58.5	57.9	62.9
Median	67.8	58.5	63.1	66.5	58.5	61.7	66.5
Average Rank	5.93	10.25	8.83	7.65	10.6	9.27	8.08
Win/Loss/Tie Ratio	3/27/3	0/30/0	1/29/1	3/27/2	0/30/0	1/29/1	2/28/2

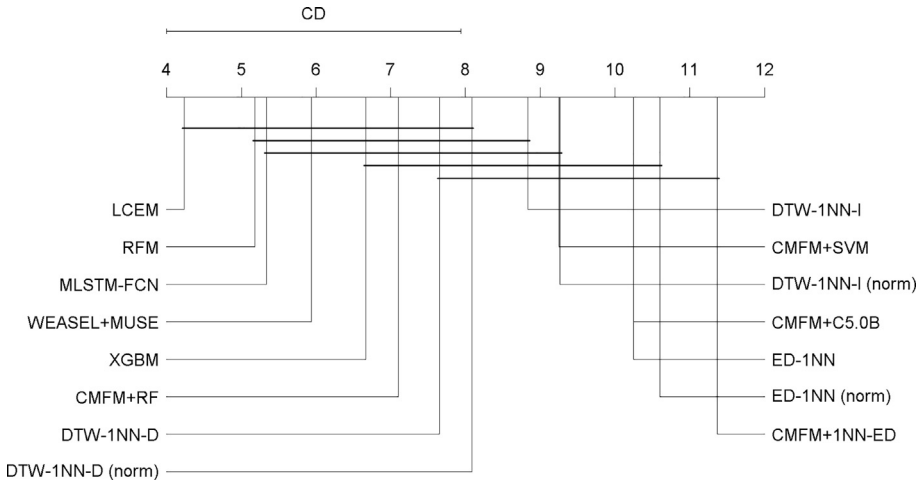


Fig. 3. Critical Difference diagram, $\alpha = 0.05$.

Table 5
Wilcoxon Signed-Rank test *p*-values obtained by the Wilcoxon Signed-Rank test of all the accuracies pairs of our models and the best models of the state-of-the-art in the first union line in Fig. 3.

Models	LCEM	XGBM	RFM	MLSTM-FCN	WEASEL + MUSE	CMFM + RF	DTW-1NN-D	DTW-1NND(norm)
CMFM + RF	0.1650	0.9672	0.1294	0.0977	0.2848	–	0.5237	0.4224
CMFM + C5.0B	0.0061	0.0410	0.0012	0.0012	0.0111	0.0001	0.1442	0.2177
CMFM + SVM	0.0027	0.0379	0.0011	0.0047	0.0643	0.0045	0.4732	0.7499
CMFM + 1NN	0.0020	0.0036	0.0004	0.0032	0.0049	$6.9e^{-06}$	0.0373	0.0397

- CMFM + RF is the model of our proposal that offers the best results among the four proposed models.
- The CMFM + RF model offers, with its default configuration, competitive results that are statistically indistinguishable from the main state-of-the-art algorithms, which optimize their parameters for each dataset.
- The CMFM + C5.0B model offers for some datasets results close to the best ones. This behavior turns it into a very interesting model because of its high interpretability.
- The accuracy of the classifiers obtained with the proposal of this work has proven to be competitive over a wide range of datasets.

5. Analysis of the interpretability

As clearly stated by many authors, interpretability has become a key property of machine learning systems. It relates to the easiness of understanding the decision-making process of the system.

In this section, we analyze the interpretability of the classification models that can be built following our proposal and how relevant knowledge can be derived from them. We begin by discussing the interpretability contributed by the consid-

ered features (Section 5.1). Then we consider the interpretability provided by trees (Section 5.2). Finally, we analyze how to extract knowledge from the models built upon the features (Section 5.3). First, we study how to assess the overall importance of the selected features as well seeking even more simplified models. Then the variable importance within the multivariate time series is addressed.

5.1. Interpretability of our proposal

The interpretation of a classifier depends mainly on features upon which it is based (inputs) and the actual algorithm (the technique). The representation features have been chosen following, among other criteria, interpretability. These features express different behaviors of each time series. It is expected that the time series that belong to the same class show similar behaviors. Representing the time series through these features allows us to evaluate, numerically, these behaviors, compare them, and draw conclusions. For example, the value distribution of the time series can define the difference between the two classes of one problem that compose it. Features such as kurtosis and skewness allow us to differentiate time series with different value distribution. Also, the variability level or chaos of the time series may allow us to differentiate between the distinct classes. In this case, entropy related features allow us to quantify and compare this behavior between different time series. Usually, we find these differentiating patterns based on different features, obtaining a greater expressiveness and interpretability of the results. The overall point is that most of these features are easier to understand than the original time series values. The user can grasp a better understanding of the time series behavior through these well stated features. The interpretability of them is usually greater than a vector of lagged-values. Furthermore, they are independent of the classifier to be used.

On the other hand, classifiers offer different interpretability levels depending on the structure and nature of the connections between inputs and outputs that they build and on their number. For example, it is more useful to know that the difference between two classes depends on the linearity of the time series, on its distribution of values, or the stability of its values, among others, than on particular values in certain instants of time. So, for example, classification trees are easier to understand than deep neural networks. In addition, they can be easily translated into a set of rules. Analyzing the models considered in this work, we can observe that the C5.0B model offers us a simple decision tree based on the features used, although as we saw in Table 4, its accuracy results are not the best. On the other hand, an RF offers competitive results in exchange for sacrificing part of their interpretability, although RF is able to offer an assessment of the importance of each feature in the final model that can be very useful. In contrast, models such as 1NN-ED lack interpretability since they work on how much one instance resembles another, and SVMs are really complex to interpret since weights can be affected by external components unrelated to the underlying importance of each variable. Since tree-based models offer different interpretability tools, we will focus on them in this section.

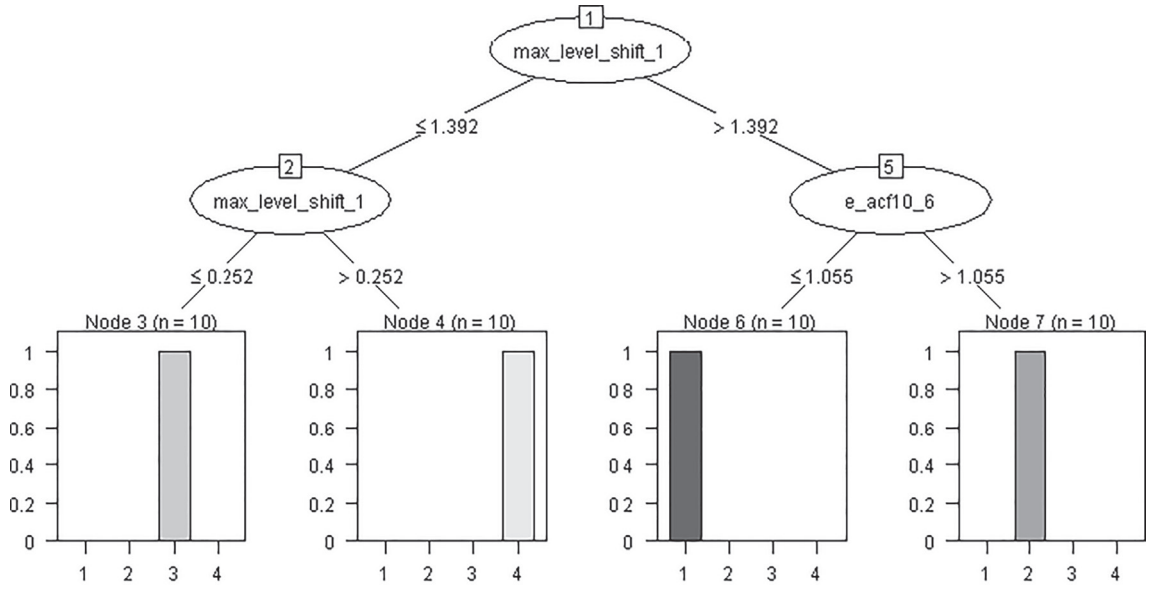
5.2. C5.0 with Boosting model interpretability

Decision trees offer very simple and straightforward interpretability. The C5.0 model with boosting allows us to explain its results using the rules included in the tree. This, together with the use of well-known features, such as those used in our proposal, allows us to understand the decisions of the model based on well-defined behaviors of the time series.

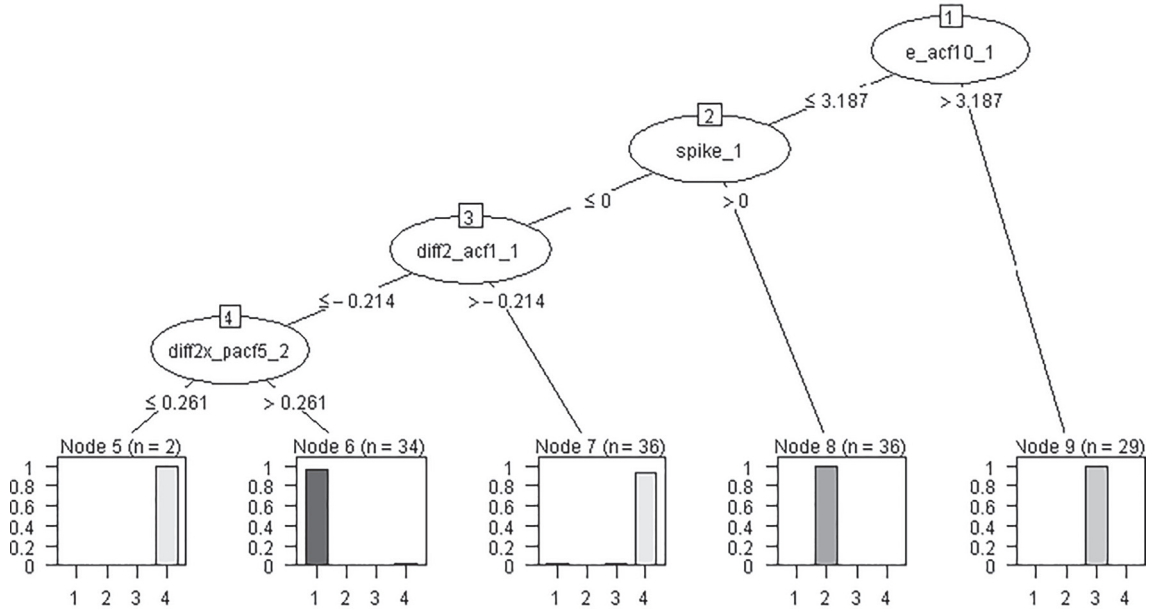
For illustrative purposes, we have included two simple examples. In Fig. 4, we show two examples of a single C5.0B tree for the BasicMotions and Epilepsy datasets. BasicMotions is a dataset with 4 classes and MTS with 6 variables. As we can see in Fig. 4a, our approach allows us to solve this problem with a simple tree composed of 3 nodes. Two of these nodes refer to features of variable 1, and the remaining one refers to variable 6. According to this tree, we can say that a time series belongs to class 3 if the maximum mean shift between two consecutive windows, for variable 1, is less than 0.252. On the other hand, we can know that a time series belongs to class 2 if the maximum mean shift between two consecutive windows, for variable 1, is greater than 1.392 and the sum of the first ten squared autocorrelation coefficients, for variable 6, is greater than 1.055. Although the results obtained for this dataset are not the best possible ones, it is remarkable how you can obtain acceptable results with such a simple decision tree. Next, we consider the dataset Epilepsy which has 4 classes and MTS with 3 variables. In Fig. 4b, we see a tree with 4 nodes: three of them refer to variable 1, and another to variable 2. In this dataset, if the sum of the first ten squared autocorrelation coefficients, for variable 1, is higher than 3.187, we know that the time series belongs to class 3. Otherwise, if the spikiness variance of the leave-one-out variances of the remainder component of the time series is positive, we can say that the time series belongs to class 2. Another relevant conclusion is that the number of features actually used to describe the classifier is rather small, making it easier to comprehend the process.

5.3. Feature and variable importance

The set of 41 features selected for the representation of time series enable a diverse and extensive description of the characteristics of the time series. For the purpose of classification, however, not all of them are always necessary. This fact has been observed in the classifiers obtained in the empirical analysis detailed in Section 4 and is clearly illustrated in the examples depicted in Section 5.2. It is obvious that simplicity in the classifiers leads to enhanced interpretability. This simplification can be achieved for each classifier by including a feature selection stage in the preprocessing of each workflow. However, we endeavor to seek a more general approach. So, in this section, we describe the work —and attained results—



(a) BasicMotions example of a single C5.0B tree with time series features.



(b) Epilepsy example of a single C5.0B tree with time series features.

Fig. 4. Interpretability CMFM example.

reached in the trek of a smaller subset of features. This trek has been guided through the relevance of the proposed features. An analogue approach can be followed to identify the importance of the different variables composing an MTS.

While different definitions of feature relevance are published, and the interpretability of the features has already been discussed, we turn an eye now towards accuracy. Since Random Forest is the model of our proposal that obtains the best accuracy results, we are going to analyze the importance given to each variable in this section. The importance measure used for each variable by this model is the total decrement with respect to the node impurities, resulting from splitting on the variable, averaged over all trees. In classification problems, the node impurity is measured by the Gini Index [10]. We have performed three different analyses that allow us to extract the desired information:

1. Analysis of the importance of each feature in each dataset (Section 5.3.1). This allows us to know which features have a greater contribution to the final result. Based on the features with the highest contribution, we can determine which behaviors, represented in those features, are the ones that define each type of time series.
2. Analysis of the accumulated importance of each feature over a large set of datasets (Section 5.3.2). This analysis allows us to identify which features are representative of most problems and which ones are uninteresting.
3. Analysis focused on the cumulative importance of features for each variable composing the MTS (Section 5.3.3). In this way, we could identify which variables contain a greater amount of information about a given problem. These variables would be the most interesting ones to solve the problem.

5.3.1. Feature importance by dataset

Since we are working with MTS, each feature is calculated for each variable of a MTS. Therefore, each feature of the original set offers a different result for each variable of the same MTS. For this, it is normal obtaining different values of *importance* for the same feature in different variables. To easily compare the importance of the original features in MTS, we need to simplify the importance values of each feature over each variable to a global importance value per feature. For this reason, we have calculated the mean of the importance of each feature over all the variables. For example, for the approximation entropy feature in a 7-variables MTS, we get 7 different values of importance (1 for each variable to which its corresponding approximation entropy feature is calculated). We add these 7 values and divide them by the number of variables of our MTS. In this way, we also penalize the importance of any features that could not be calculated in any variable. Finally, we normalize these last values between 0 and 1 for each dataset. This produces a normalized measure that is best explored in a graphical way. Fig. 5 shows a heatmap of the importance of each feature in the RF classifier for each dataset, where the datasets have been ordered by the accumulated importance of the 41 features.

In Fig. 5, we can see significant differences among the datasets. We can differentiate two bands: the upper band where only some features accumulate great importance, and the lower band where multiple features have great importance. In the upper band, we can see datasets for which the classifiers are *dominated* by up to four features with high importance (SelfRegulationSCP1, InsectWingBeat, SpokenArabicDigits, NATOPS, BasicMotions, DuckDuckGeese, PEMS-SF, among others). In these cases, two categories are observed: the set of features used is sufficiently expressive to address the problem satisfactorily with competitive results (RF: InsectWingBeat, and PEMS-SF), or the selected features are not sufficient and other approaches achieve significantly better results (MLSTM-FCN: BasicMotions, DuckDuckGeese, NATOPS, SelfRegulationSCP1, and SpokenArabicDigits). In the lower band, we can identify cases where all the features are necessary (HandMovementDirection, PhonemeSpectra, Handwriting, FingerMovements, MotorImagery, EthanolConcentration, ArticularyWordRecognition, among others). In these cases, we can observe two behaviors. On the one hand, the differentiating capabilities of the features are not enough for some of them to stand out from the rest, so the classifier assigns similar importance to a large number of features. On the other hand, in datasets with a complex problem, it is not possible to find a reduced subset of features capable of explaining the problem. In these cases, more complex solutions are obtained, with a high number of features, capable of offering results very close to the best ones (ArticularyWordRecognition and PhonemeSpectra).

5.3.2. Accumulated feature importance over set of datasets

Another particularly interesting analysis to be carried out is related to the importance at the feature level. In Fig. 6, we show the average importance of each feature throughout all the datasets. We have ordered the features in decreasing order of the Average Gini Index. These values have been obtained from the results shown in Fig. 5. The average value of the importance of each feature has been calculated over the 30 datasets processed. We can see that there is a group of three distinguished features that frequently reach the highest importance values, namely, *curvature*, *linearity*, and *spike*. This group has values of importance far superior to the rest. As a fast check experiment, we have built a C5.0B model for every dataset restricted to use only these three variables and have observed an average improvement of accuracy of 2.2%. A second distinguished breakpoint in the importance curve leads to a second group of relevant features: *max_var_shift*, *max_level_shift*, *skewness*, *spectral_entropy*, and *trend*. All of them are assigned importance values greater than 0.45. Even further, it is interesting to realize that the features related to the complexity of a time series get the highest importance values. Higher values achieved by features such as *trend* confirm that the components of the time series are very descriptive and useful when extracting information from them. Other features such as *curvature*, *linearity*, and *spike*, shown as characteristic behaviors of the time series, are especially useful in describing them.

On the other hand, there are also features with particularly low values of importance: *nperiods*, *seasonal_periods*, and *length*. The feature *length* is not of high importance because, in the UEA repository, the vast majority of datasets are composed of MTS of equal length. The features *nperiods* and *seasonal_periods* have importance values of 0 because we have processed all the time series with a frequency of 1, making the processing as general as possible. If all the time series have the same frequency, the features *nperiods* and *seasonal* always return the same value, which does not provide distinguishing information for the problem. In the case that the best results are sought, and a detailed analysis of the time series is carried out in which data on seasonality is available, these measures can be very useful. Furthermore, features such as *max_kl_shift* and *lempel_ziv* have obtained low average importance values, although they are particularly explanatory. If we look at Fig. 5, we find some datasets like PEMS-SF and EigenWorms in which *max_kl_shift* and *lempel_ziv* have a high importance, respectively. In this case, even if we identify features that are generally not interesting, they may be relevant for specific problems.

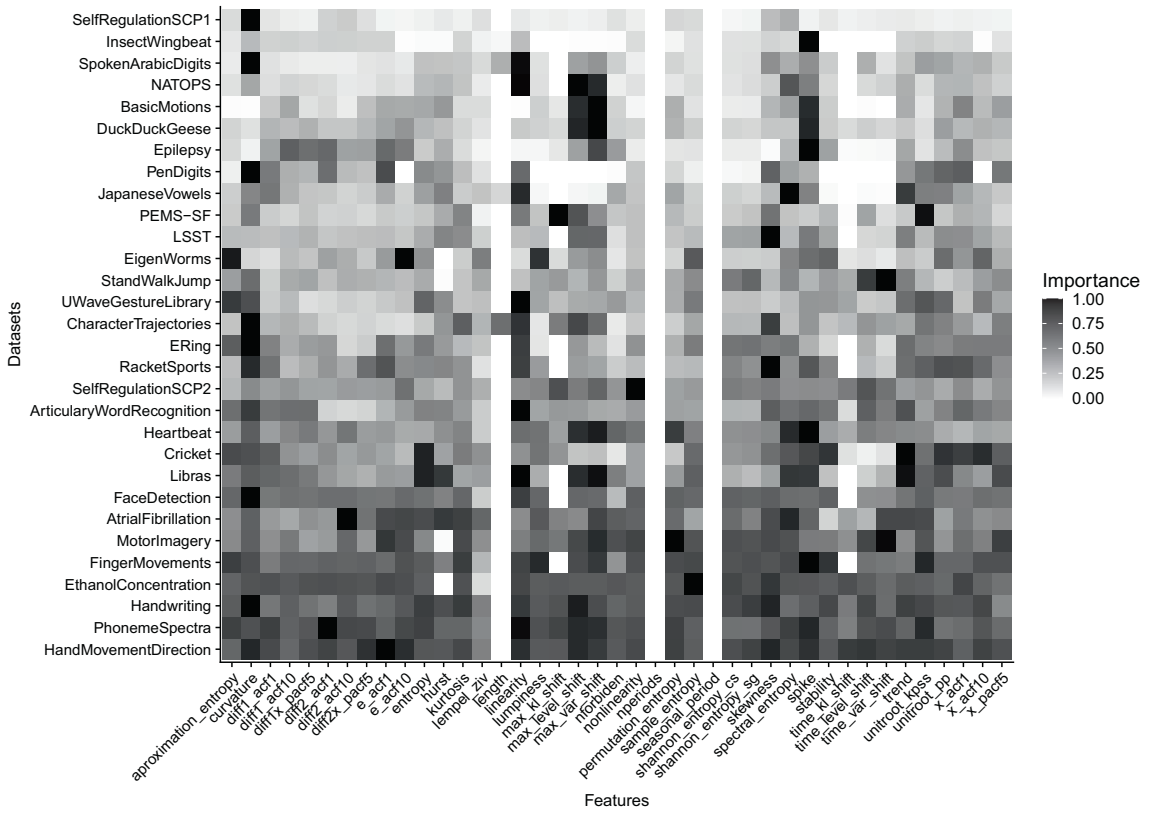


Fig. 5. Ordered features importance heatmap.

These cases reinforce the idea that the selection of a representative set of features must be supported by theoretical knowledge about the structure of time series and by different analyses of results performed on large sets of datasets.

5.3.3. Variable importance

Finally, we analyze an important point in the field of MTSC, the existence of components or variables that contain a major part of the information on the problem. To assess the importance of a variable in a given dataset, we have computed the sum of the importance of the 41 features for each variable of the problem in question. Then, this value is normalized by dividing it by the maximum value of each sum. In this way, the importance value of any variable belongs in $[0, 1]$, with 1 being the maximum. Several statistics for the variable importance of the considered datasets have been gathered in Table 6. "Sum" refers to the addition of the variable importance of all the variables of the dataset. Conversely, "Max", "Min", "Mean", "Median", and "SD" corresponds to the maximum, minimum, mean, median, and standard deviation of the variable importance.

Some observations can be made from the table. For example, in the PhonemeSpectra dataset, all 11 variables are of similar importance, and CMFM + RF was close to the best results obtained. This means that all variables contain information of interest. On the other hand, in NATOPS, there is a relevant dispersion among variable importance. To further illustrate diversity in variable importance, we have selected some datasets and plotted a histogram of their respective variable importance —see Fig. 7.

For the BasicMotions dataset, in Fig. 7a, we can see 2 variables with much higher importance than the rest, together with a third variable that also stands out. These variables are, in decreasing order of importance: 2, 6, 1, 3, 5, and 4. If we compare these values of importance —derived from RF— against the C5.0B tree, shown in Fig. 4a, we can realize that two of the tree nodes have features of the variable 1, and the remaining node has a feature of the variable 6, which are the second and third most important variables. In the case of dataset Epilepsy —see Fig. 7b—, we can see 2 variables significantly distinguished from the rest. These variables are, in decreasing order of importance: 1, 2, and 3. Fig. 4b shows a C5.0B tree for this dataset. Three of its nodes have features of the variable 1, the most important variable, while the remaining node has a feature of the variable 2, the second most important variable according to the RF. These examples show a certain relation between the variables with more importance according to the RF and those used by a simple classifier such as C5.0B.

In the case of the NATOPS dataset, the most relevant information is expressed in just 24 of the variables. In Fig. 7c, we can see that it is more difficult to obtain well-differentiated groups of variables according to their importance. In this case, we can see that the 3 variables with the greatest importance are significantly distanced from the rest, with importance values

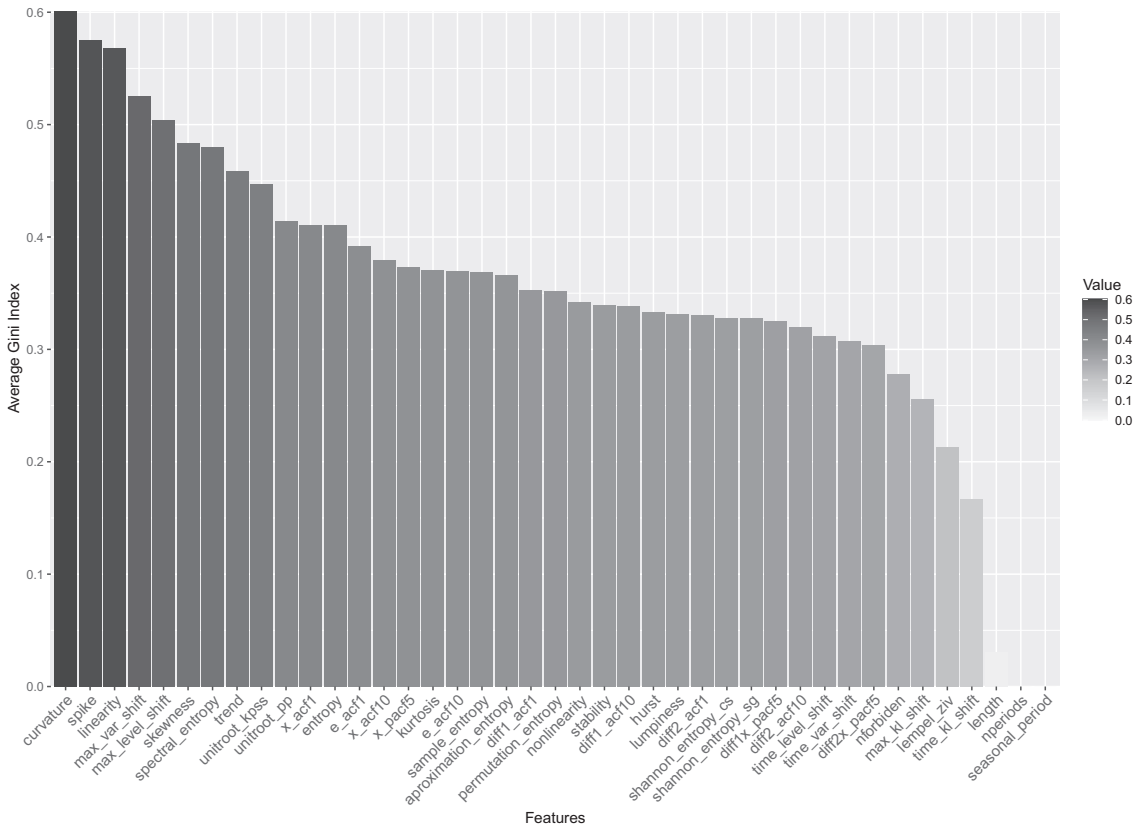


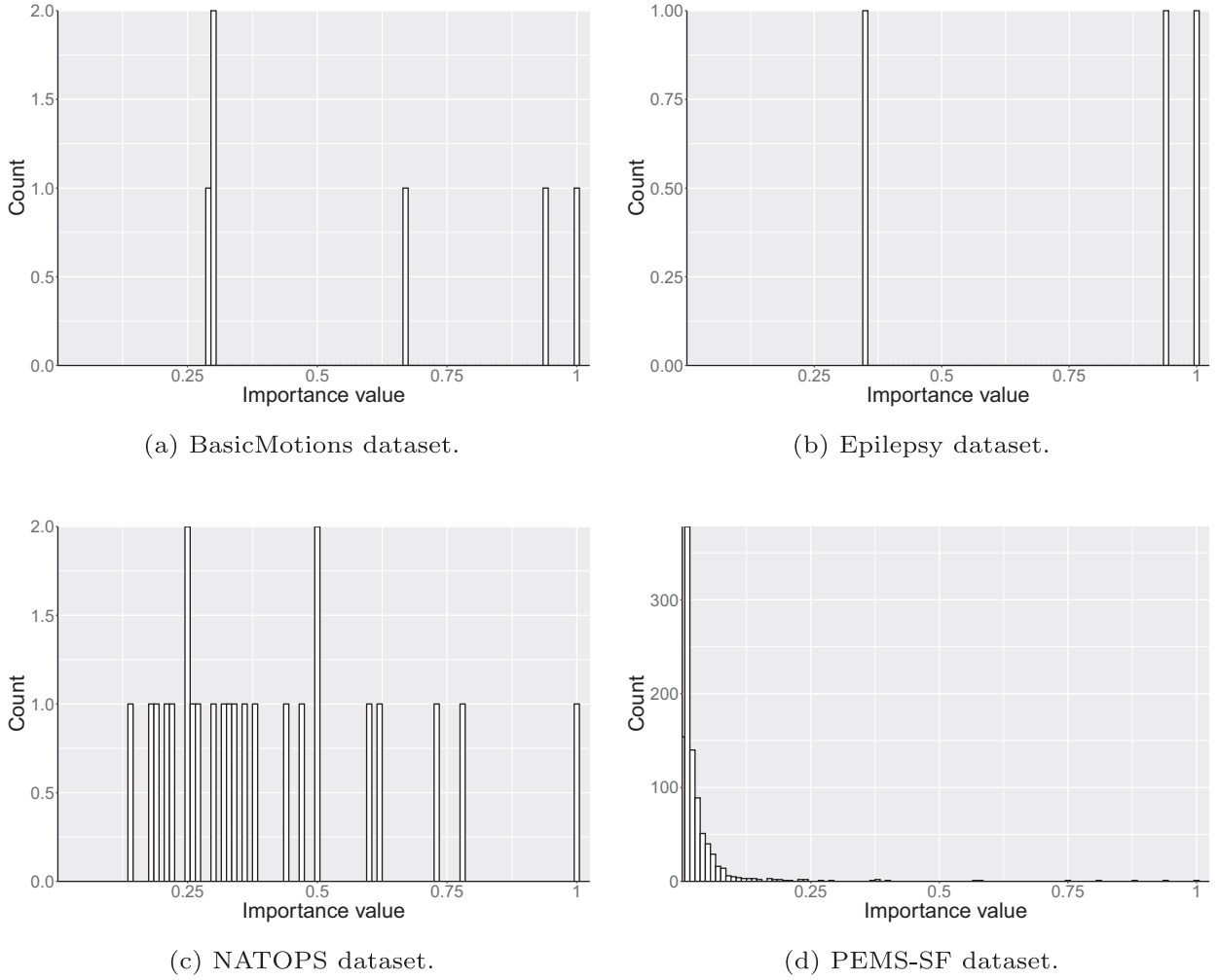
Fig. 6. Average importance of each feature in the UEA repository.

Table 6
Statistics of the variable importance.

Datasets	Sum	Max	Min	Mean	Median	SD	Variables
ArticularyWordRecognition	4.816	1	0.190	0.535	0.504	0.263	9
AtrialFibrillation	1.720	1	0.720	0.860	0.860	0.198	2
BasicMotions	3.495	1	0.291	0.582	0.485	0.332	6
CharacterTrajectories	2.120	1	0.359	0.707	0.762	0.324	3
Cricket	4.726	1	0.627	0.788	0.797	0.131	6
DuckDuckGeese	143.003	1	0	0.106	0.074	0.115	1345
EigenWorms	3.739	1	0.367	0.623	0.594	0.208	6
Epilepsy	2.295	1	0.352	0.765	0.943	0.359	3
ERing	3.356	1	0.659	0.839	0.848	0.152	4
EthanolConcentration	2.971	1	0.980	0.990	0.991	0.010	3
FaceDetection	100.234	1	0.557	0.696	0.692	0.061	144
FingerMovements	19.528	1	0.547	0.697	0.676	0.097	28
HandMovementDirection	9.141	1	0.795	0.914	0.927	0.079	10
Handwriting	2.701	1	0.818	0.900	0.883	0.092	3
Heartbeat	30.558	1	0.307	0.501	0.470	0.162	61
InsectWingbeat	59.976	1	0.195	0.300	0.221	0.159	200
JapaneseVowels	9.279	1	0.563	0.773	0.789	0.138	12
Libras	1.901	1	0.901	0.950	0.950	0.070	2
LSST	5.213	1	0.718	0.869	0.873	0.138	6
MotorImagery	50.752	1	0.588	0.793	0.791	0.097	64
NATOPS	9.634	1	0.140	0.401	0.334	0.215	24
PEMS-SF	30.718	1	0	0.032	0.013	0.078	963
PenDigits	1.684	1	0.684	0.842	0.842	0.223	2
PhonemeSpectra	10.917	1	0.984	0.992	0.993	0.004	11
RacketSports	4.493	1	0.304	0.749	0.810	0.263	6
SelfRegulationSCP1	4.044	1	0.532	0.674	0.629	0.175	6
SelfRegulationSCP2	6.673	1	0.901	0.953	0.950	0.037	7

Table 6 (*continued*)

Datasets	Sum	Max	Min	Mean	Median	SD	Variables
SpokenArabicDigits	5.962	1	0.162	0.459	0.325	0.302	13
StandWalkJump	3.614	1	0.758	0.904	0.928	0.112	4
UWaveGestureLibrary	2.782	1	0.877	0.927	0.904	0.064	3



(a) BasicMotions dataset.

(b) Epilepsy dataset.

(c) NATOPS dataset.

(d) PEMS-SF dataset.

Fig. 7. Histograms of variable importance values.

higher than 0.70. Depending on the information sought and the difficulty of the problem, we could decide to lower the threshold, create different groups of variables, etc. These histograms are especially interesting for datasets with a large number of variables. For example, in the PEMS-SF dataset, Fig. 7d, only some of the 963 variables have a high importance, giving residual importance to the rest. In this case, there are only 5 variables with an accumulated importance value higher than 0.75. These variables are, in decreasing order of importance: 212, 55, 604, 172, and 187.

To better understand the variable importance distribution, we have calculated the percentage of cumulative importance of each variable in each dataset. For this, we have calculated the cumulative importance of the set of features for each variable and dividing these values by the sum of the importance of all the features. In this way, we can see which variables contain a greater amount of useful information, e.g., for a dataset composed of MTS with 3 variables, we have calculated the sums of the importance of the 41 features used for each variable and divided those values by the total of the sum of the importance of the 41 features for the 3 variables. In Fig. 8, we show the percentage of cumulative importance for each variable for some processed datasets: it is easy to spot the differences in importance by comparing the relative lengths of the

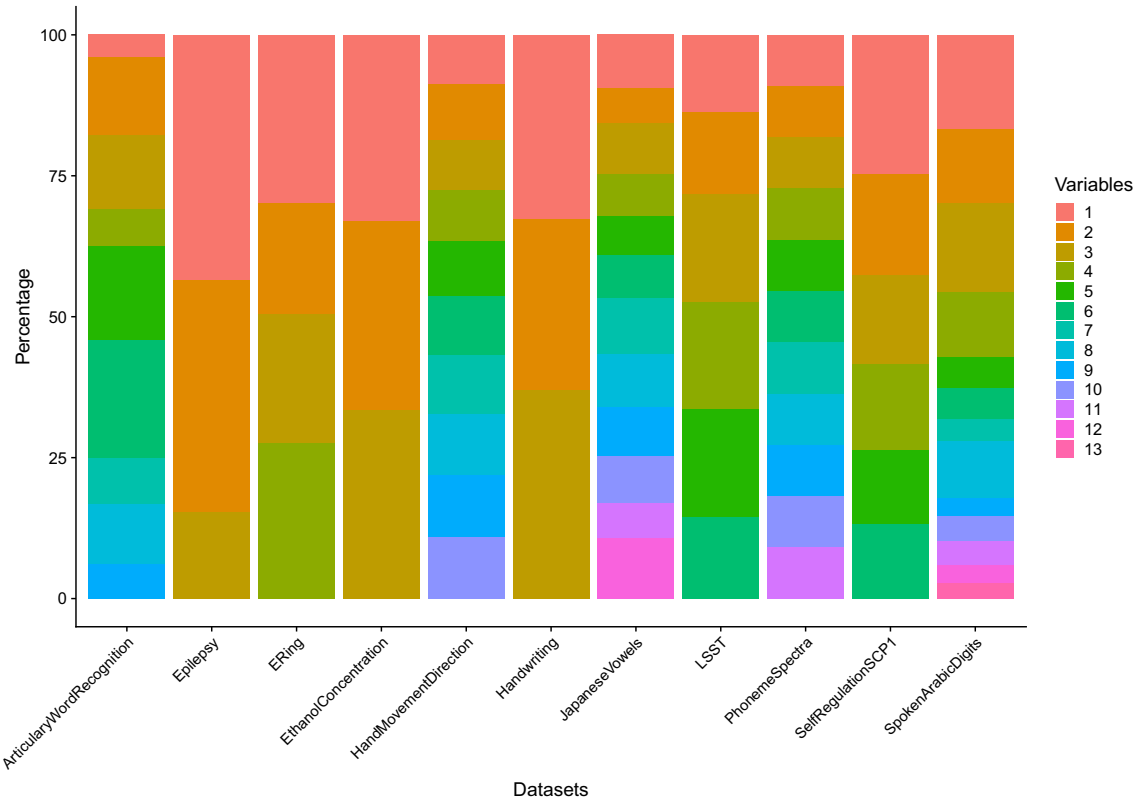


Fig. 8. Accumulated importance by variable.

colored pieces into which each bar is divided. Each of these pieces represents a single variable. For example, in the ArticularyWordRecognition dataset, variables 1, 4, 7, and 9 are of less importance; ERing dataset shows a great importance accumulated in variables 1 and 4; Epilepsy dataset has much of its useful information in variables 1 and 2; and so forth. With these results, the preprocessing of the data can be modified in such a way as to improve the recording of the data of these variables or to give them greater importance in the learning process.

Based on the study conducted in this section, we can conclude that:

- Combining easily interpretable models with understandable time series features, the user can better understand and explain the decision-making process. The CMFM+C5.0B model is especially interesting in this respect.
- CMFM+RF allows us to define a feature importance measure that can be used to reduce the set of considered features, leading to simpler and more interpretable models.
- The feature importance can be used to assess variable importance within multivariate time series. In addition, based on the feature importance, we have also defined a variable importance measure that allows us to identify the most relevant variables and thus guide or prioritize the time series caption, storage, and processing.

6. Conclusions

In this paper, we have presented a method to represent multivariate time series in terms of a set of interpretable features. This method enables the use of conventional classification algorithms on MTSC problems, considerably expanding the tools available to deal with this type of problem. The main benefit of this approach is to obtain interpretable classifiers so that the decision-making process can be better understood. We have designed and executed a thorough empirical study, based on the main repository of the state-of-the-art, composed of 30 datasets. The accuracy results of the built classifiers remain competitive with respect to the state-of-the-art results. In particular, no statistically relevant differences can be found between our CMFMTS+RF proposal and the most accurate already known methods.

The interpretability of the built classifiers has been extensively analyzed. Measures for feature importance and variable importance have been defined, allowing to derive relevant knowledge for each particular problem addressed with the proposed method. In addition, we have verified the existence of a set of features that maintains high importance throughout

different datasets. However, there are also certain cases where other features that are less important on average offer the best results. These features are usually related to characteristic behaviors of the time series.

The method has been implemented in the R programming language. The code is publicly available.

CRediT authorship contribution statement

Francisco J. Baldán: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing - original draft, Writing - review & editing, Visualization. **José M. Benítez:** Conceptualization, Supervision, Investigation, Resources, Writing - review & editing, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research has been partially funded by the following grants: TIN2016-81113-R from the Spanish Ministry of Economy and Competitiveness, and P12-TIC-2958 from Andalusian Regional Government, Spain. Francisco J. Baldán holds the FPI grant BES-2017–080137 from the Spanish Ministry of Economy and Competitiveness.

References

[1] A. Abdiansah, R. Wardoyo, Time complexity analysis of support vector machines (SVM) in LibSVM, International Journal Computer and Application 128 (3) (2015) 28–34.
[2] A. Antonucci, R. De Rosa, A. Giusti, F. Cuzzolin, Robust classification of multivariate time series by imprecise hidden Markov models, International Journal of Approximate Reasoning 56 (2015) 249–263.
[3] A. Bagnall, H.A. Dau, J. Lines, M. Flynn, J. Large, A. Bostrom, P. Southam, E. Keogh, The UEA multivariate time series classification archive, 2018, arXiv preprint arXiv:1811.00075.
[4] F.J. Baldán, J.M. Benítez, Distributed FastShapelet Transform: a big data time series classification algorithm, Information Sciences 496 (2019) 451–463.
[5] F.J. Baldán, J.M. Benítez, Complexity Measures and Features for Times Series classification, arXiv preprint arXiv:2002.12036..
[6] M. Baydogan, Multivariate Time Series Classification Datasets, URL:http://www.mustafabaydogan.com, accessed: 2020-07-01 (2017)..
[7] M.G. Baydogan, G. Runger, Learning a symbolic representation for multivariate time series classification, Data Mining and Knowledge Discovery 29 (2) (2015) 400–422.
[8] M.G. Baydogan, G. Runger, Time series representation and similarity based on local autopatterns, Data Mining and Knowledge Discovery 30 (2) (2016) 476–509.
[9] A. Bostrom, A. Bagnall, A shapelet transform for multivariate time series classification, arXiv preprint arXiv:1712.06428..
[10] L. Ceriani, P. Verme, The origins of the Gini index: extracts from Variabilità e Mutabilità (1912) by Corrado Gini, The Journal of Economic Inequality 10 (3) (2012) 421–443.
[11] W.A. Chaovalitwongse, R.S. Pottenger, S. Wang, Y.-J. Fan, L.D. Iasemidis, Pattern-and network-based classification techniques for multichannel medical data signals to improve brain diagnosis, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 41 (5) (2011) 977–988.
[12] M. Das, M. Pratama, A. Ashfahani, S. Samanta, Fernn: A fast and evolving recurrent neural network model for streaming data classification, in: 2019 International Joint Conference on Neural Networks (IJCNN), IEEE, 2019, pp. 1–8.
[13] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (Jan) (2006) 1–30.
[14] F.K. Došilović, M. Brčić, N. Hlupić, Explainable artificial intelligence: A survey, in: 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2018, pp. 0210–0215..
[15] D. Dua, C. Graff, UCI Machine Learning Repository (2017). URL:http://archive.ics.uci.edu/ml.
[16] K. Fauvel, É. Fromont, V. Masson, P. Faverdin, A. Termier, Local Cascade Ensemble for Multivariate Data Classification, arXiv preprint arXiv:2005.03645..
[17] T. Górecki, M. Łuczak, Multivariate time series classification with parametric derivative dynamic time warping, Expert Systems with Applications 42 (5) (2015) 2305–2312.
[18] J. Grabocka, M. Wistuba, L. Schmidt-Thieme, Fast classification of univariate and multivariate time series through shapelet discovery, Knowledge and Information Systems 49 (2) (2016) 429–454.
[19] F. Karim, S. Majumdar, H. Darabi, S. Harford, Multivariate LSTM-FCNs for time series classification, Neural Networks 116 (2019) 237–245.
[20] S. Karimi-Bidhendi, F. Munshi, A. Munshi, Scalable classification of univariate and multivariate time series, in: 2018 IEEE International Conference on Big Data (Big Data), IEEE, 2018, pp. 1598–1605.
[21] I. Karlsson, P. Papapetrou, H. Boström, Generalized random shapelet forests, Data Mining and Knowledge Discovery 30 (5) (2016) 1053–1085.
[22] E.J. Keogh, M.J. Pazzani, Derivative dynamic time warping, in: Proceedings of the 2001 SIAM International Conference on Data Mining, SIAM, 2001, pp. 1–11.
[23] K. Khadiev, I. Mannapov, L. Safina, The Quantum Version Of Classification Decision Tree Constructing Algorithm C5. 0, arXiv preprint arXiv:1907.06840..
[24] M. Khan, N. Javaid, M.N. Iqbal, M. Bilal, S.F.A. Zaidi, R.A. Raza, Load prediction based on multivariate time series forecasting for energy consumption and behavioral analytics, Conference on Complex, Intelligent, and Software Intensive Systems, Springer (2018) 305–316.
[25] B. Klockl, Multivariate time series models applied to the assessment of energy storage in power systems, in: Proceedings of the 10th International Conference on Probablistic Methods Applied to Power Systems, IEEE, 2008, pp. 1–8.
[26] Z. Li, X. Jin, X. Zhao, Drunk driving detection based on classification of multivariate time series, Journal of Safety Research 54 (2015) 61–67.
[27] T. Lin, T. Guo, K. Aberer, Hybrid neural networks for learning the trend in time series, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, AAAI Press, 2017, pp. 2273–2279.
[28] E.A. Maharaj, A.M. Alonso, Discriminant analysis of multivariate time series: Application to diagnosis based on ECG signals, Computational Statistics & Data Analysis 70 (2014) 67–87.
[29] G. Manis, Fast computation of approximate entropy, Computer Methods and Programs in Biomedicine 91 (1) (2008) 48–54.
[30] G. Manis, M. Aktaruzzaman, R. Sassi, Low computational cost for sample entropy, Entropy 20 (1) (2018) 61.

[31] A. McGovern, D.H. Rosendahl, R.A. Brown, K.K. Droegemeier, Identifying predictive multi-dimensional time series motifs: an application to severe weather prediction, Data Mining and Knowledge Discovery 22 (1–2) (2011) 232–258.
[32] J. Mei, M. Liu, Y.-F. Wang, H. Gao, Learning a mahalanobis distance-based dynamic time warping measure for multivariate time series classification, IEEE Transactions on Cybernetics 46 (6) (2015) 1363–1374.
[33] C. Orsenigo, C. Vercellis, Combining discrete SVM and fixed cardinality warping distances for multivariate time series classification, Pattern Recognition 43 (11) (2010) 3787–3794.
[34] W. Pei, H. Dibeklioğlu, D.M. Tax, L. van der Maaten, Multivariate time-series classification using the hidden-unit logistic model, IEEE Transactions on Neural Networks and Learning Systems 29 (4) (2017) 920–931.
[35] S. Samanta, M. Pratama, S. Sundaram, N. Srikanth, A dual network solution (DNS) for lag-free time series forecasting, in: 2020 International Joint Conference on Neural Networks (IJCNN), 2020, pp. 1–8.
[36] P. Schäfer, U. Leser, Multivariate time series classification with WEASEL+ MUSE, arXiv preprint arXiv:1711.11343..
[37] S. Seto, W. Zhang, Y. Zhou, Multivariate time series classification using dynamic time warping template selection for human activity recognition, in: 2015 IEEE Symposium Series on Computational Intelligence, IEEE, 2015, pp. 1399–1406..
[38] M. Shokoohi-Yekta, B. Hu, H. Jin, J. Wang, E. Keogh, Generalizing DTW to the multi-dimensional case requires an adaptive approach, Data Mining and Knowledge Discovery 31 (1) (2017) 1–31.
[39] K.S. Tuncel, M.G. Baydogan, Autoregressive forests for multivariate time series modeling, Pattern Recognition 73 (2018) 202–215.
[40] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, E. Keogh, Indexing multi-dimensional time-series with support for multiple distance measures, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003, pp. 216–225.
[41] M. Wistuba, J. Grabocka, L. Schmidt-Thieme, Ultra-fast shapelets for time series classification, arXiv preprint arXiv:1503.05018..
[42] L. Ye, E. Keogh, Time series shapelets: a new primitive for data mining, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009, pp. 947–956.
[43] C. Yu, L. Luo, L.L.-H. Chan, T. Rakthanmanon, S. Nutanong, A fast LSH-based similarity search method for multivariate time series, Information Sciences 476 (2019) 337–356.