Complexity Measures and Features for Times Series classification

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*

Abstract

Classification of time series is a growing problem in different disciplines due to the progressive digitalization of the world. Currently, the state-of-the-art in time series classification is dominated by The Hierarchical Vote Collective of Transformation-based Ensembles. This algorithm is composed of several classifiers of different domains distributed in five large modules. The combination of the results obtained by each module weighed based on an internal evaluation process allows this algorithm to obtain the best results in state-of-the-art. One Nearest Neighbour with Dynamic Time Warping remains the base classifier in any time series classification problem for its simplicity and good results. Despite their performance, they share a weakness, which is that they are not interpretable. In the field of time series classification, there is a tradeoff between accuracy and interpretability. In this work, we propose a set of characteristics capable of extracting information on the structure of the time series to face time series classification problems. The use of these characteristics allows the use of traditional classification algorithms in time series problems. The experimental results of our proposal show no statistically significant differences from the second and third best models of the state-of-the-art. Apart from competitive results in accuracy, our proposal is able to offer interpretable results based on the set of characteristics proposed.

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

1. Introduction

At present, large amounts of information are recorded from a wide variety of fields. There is a growing need to analyze and classify these data to obtain useful information, for example, to identify different patterns of electricity

*Corresponding author

Email addresses: fjbaldan@decsai.ugr.es (Francisco J. Baldán), J.M.Benitez@decsai.ugr.es (José M. Benítez)

consumption in order to adapt prices to consumers [42], to identify cardiac anomalies characteristics of a pathology [17] or search for anomalies in starlight curves [59].

The field of time series classification (TSC) [5] has historically been dominated by proposals that offer good classification results but are hardly interpretable. For example, a simple approach that achieves good average results in the different types of problems is One Nearest Neighbour with Dynamic Time Warping (1NN+DTW) [11][51]. This approach tells us how similar the time series are to each other, but it does not allow us to extract additional information from the problem. Recently the Collective Of Transformation Ensembles (COTE) [6] has been shown to obtain the best TSC results on the reference time series database collected in the UCR repository [19], in the 2015 version of this repository. This algorithm is composed of 35 classifiers (flat-COTE) which apply cross-validation on the training set. COTE contains reference classifiers in the fields of TSC. These classifiers are evaluated internally with cross-validation, and depending on their results, they are included in the final result. Recently The Hierarchical Vote Collective of Transformation-based Ensembles (HIVE-COTE) [3] has been proposed, which improves the classification process carried out by its previous versions. HIVE-COTE is composed of several classifiers of different domains distributed in five large modules. Each module provides a probability estimate for each class and obtains a weighting proportional to the accuracy achieved over the training set. HIVE-COTE combines these estimates in a second layer and obtains the predicted class from the highest weight over all the modules. The HIVE-COTE proposal provides the best results, but its interpretability is very low, and its high computational cost prevents its application in large datasets.

Other more interpretable approaches as decision trees do not usually obtain competitive results in the field of TSC. This behavior is due to their inability to capture the time relationships between the different time instants that make up a time series. These approaches are successfully used in combination with other proposals, such as shapelets, which extract behavioral patterns from time series [63]. These patterns make it possible to differentiate time series belonging to different classes. These proposals have great interpretability since they allow us to identify, in a graphical way, patterns of interest belonging to the different classes that compose the problem. Although, in this case, there are also proposals such as the Shapelet Transform (ST) [40], which transforms these shapelets into features. ST alter the problem of TSC into a traditional, vector-based, classification problem, on which we can apply traditional algorithms, such as Random Forest (RF) [14], and obtain good results. In this way, there are proposals in Big Data such as Distributed FastShapelet Transform [7] that allows us to face TSC problems in massive data environments where traditional TSC algorithms cannot be applied due to their high computational complexity. There is a more recent proposal, which proposes the creation of a weighted ensemble of standard classifiers, such as Random Forest, Naive Bayes, Support Vector Machines, among others, on the transformed data, obtaining very competitive results. This proposal is named Shapelet Transform Classifier (STC) [13].

In the literature, we can find proposals focused on extracting a large number

of characteristics from time series [30][28]. The main idea of these characteristics is any type of mathematical operation applicable over a time series that provide valuable information. The objective of these proposals is to look for an underlying structure that represents the behavior of a time series. These types of studies are ambitious but difficult to interpret over a specific problem due to the high number of characteristics present. Moreover, these studies are oriented to the unsupervised learning environment. Some proposals make a selection of the main characteristics of a time series, from the theoretical point of view that could explain the origin of their behavior [33]. The objective of the previous work is to generate synthetics time series that represent real problem behaviors, so its main target is far from the problem of TSC. On the other hand, CAnonical Time-series CHaracteristics (catch22) [41] proposes a set of 22 characteristics that have been selected based on the classification results obtained on a large set of datasets. For this proposal, a large number of characteristics and their possible combinations have been tested, measuring the classification results obtained. The main criterion for selecting the characteristics is to provide the best possible results, although the execution time and, in some cases, their interpretability is also taken into account. Recently a method has been proposed in this line. This method, called Feature and Representation Selection (FEARS) [12], is based on obtaining different alternative representations such as derivatives, cumulative integrals, power spectrum, among others, of the time series. This method then extracts characteristics of interest using an automatic variable construction technique. As the last step, a Naive Bayes classifier is in charge of learning about the new extracted characteristics. This procedure is repeated several times to obtain the most informative set of characteristics possible.

There are other proposals based on studying the complexity of time series [65][46]. These proposals use complexity measures that measure the interrelationships between the different values in a time series. A greater number of relationships lead to greater complexity. In the same way that the traditional characteristics of the time series are capable of providing sufficient information about a problem, complexity measures can add useful information to the problem.

In this work, we present a set of characteristics composed of complexity measures and representative features of the time series. This transformation allows the use of traditional learning algorithms on TSC problems. Additionally, this transformation allows interpreting the results obtained by the classification algorithms. The performance of our proposal has been tested on a set of 112 datasets present in the UCR repository. We have applied the most popular and widely used classification algorithms based on trees that allow interpretable results. Our proposal is publicly available as an R package in the online repository [1]

The rest of the work is organized as follows: Section 2 introduces the state-of-the-art in TSC: distance-based methods, feature-based methods, and deep learning methods. In Section 3, we describe in depth our proposal. Section 4

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

shows the experimental design used, the results obtained, and the interpretability of these results. Finally, Section 5 concludes the paper.

2. Related work

There are several ways to group the TSC algorithms. In this work, we group them by the type of data on which each algorithm works and its internal operation. In this way, we have three principals groups with their corresponding subgroups. A first group is composed of the distance-based proposals (Section 2.1), which are strongly related to calculations of similarity and distance between different time series or subsequences of the time series themselves. A second group is composed of features-based proposals (Section 2.2), which are based on the calculation of certain parameters of the time series that transform the original data. After this transformation, traditional classification algorithms are applied to the new dataset. The last group would be made up of the deep learning proposals (Section 2.3), where data entry and processing depend entirely on each proposal.

2.1. Distance-based Classification

Patterns searched for in TSC problems may have their origin in different domains. For this reason, there are different types of approaches depending on multiple factors. There are currently six main approaches for dealing with this kind of TSC problems [5], grouped by the type of discriminatory features that the technique attempts to find:

- Proposals that use all the values of the time series: are linked to the use of similarity measures and different types of distance. The reference algorithm of this group is 1NN+DTW, which is simple to apply but has high computational complexity. This algorithm is often used as a benchmark in TSC problems.

- Those using phase-dependent intervals: they use small subsets from each time series, rather than using the entire time series. Proposals like Time Series Forest (TSF) [25] have been proved that extracting characteristics such as mean, variance, or slope from random intervals, and use them as classifier features, works particularly well. Characteristics such as Fourier, autocorrelation, and partial autocorrelation, which are more complex and related to the time series than those mentioned above, are used by more recent proposals such as contract Random Interval Spectral Ensemble (c-RISE) [27], with very competitive results.

- The independent phases, based on shapelets: the shapelets based ones look for substrings of the time series that allow differentiating the time series belonging to each class. They are closely linked to the use of similarity and distance measurements. The first proposals generated simple classification trees capable of differentiating the belonging of a time series to one class

or another according to the presence or not of a certain subsequence in it [63][50][43]. These approaches offered some interpretability to the results. Recent work on shapelets has shown that they are particularly useful when used as input features to a traditional classification algorithm [40][13][7], rather than as part of the classification tree itself.

- Based on dictionaries: in some cases, the presence of a certain pattern in a time series is not enough to identify whether it belongs to one class or another [39][53]. There are problems in which the number of times the pattern appears in a time series is determinant to classify it correctly. The shapelets are not useful in these cases, and the use of algorithms based on dictionaries is mandatory. These algorithms count both the presence or absence of each subsequence in a time series. They create a classifier based on the histograms obtained from these dictionaries. The way of creating the dictionary is one of the main differences among the proposals of this type. For example, Bag of patterns (BOP) [38] creates the dictionary through the Symbolic aggregate ApproXimation (SAX) [37] words extracted from each window. Symbolic Aggregate approXimation-Vector Space Model (SAXVSM) [56] combines the SAX representation used in BOP with the vector space model commonly used in Information Retrieval and counts the appearance frequencies over the classes and not over the time series. Bag of SFA symbols (BOSS) [53] does not use Piecewise Aggregate Approximation (PAA) [34] in its SAX transformation but uses truncated Discrete Fourier Transform (DFT). Furthermore, it uses the so-called Multiple Coefficient Binning (MCB) technique to discretize the truncated time series, among other differences. Despite the good results, BOSS does not scale well, so it made a proposal called contracted BOSS (cBOSS) [35], which modified the way BOSS classifiers were chosen, indicating construction time limits per classifier and saving the advances during the construction process without significant accuracy changes. Word ExtrAction for time SEries cLassification (WEASEL) [54] is one of the latest proposals made. WEASEL has highly competitive results and differs from the rest by its ability to derive the characteristics obtained, achieving a new, much smaller, and more discriminating set of features.

- Based on models: this approach is mainly oriented to problems with long time series, but with different lengths [4][18]. These proposals usually fit a model to each time series and measure the similarity between the models. It is an approach that is not sufficiently widespread and is applied to particular problems.

- Combinations or ensembles: this approach works both in time series and traditional classification problems, using the results of different models to make a final decision. In the area of time series, HIVE-COTE [3] is the best proposal to date. It uses models from different approaches and offers the best accuracy results. On the other hand, it is the approach with the highest computational complexity due to the high number of algorithms

it uses and its corresponding computational complexities. Moreover, this large number of algorithms leads to low interpretability of results.

Each of these approaches adapts to different types of problems, but they all work on the original values of the time series.

2.2. Feature-based Classification

The feature-based approach is focused on a transformation to the time series dataset, obtaining a new dataset composed of different features that explain the behavior of the original time series [28]. The feature-based approach offers multiple advantages over the distance-based approach for dealing with time-series classification problems. This approach allows analysis of time series on different time domains and with different lengths, being more widely applicable because the stationarity properties of the series are not always required. In addition, this approach allows us to use the standard classification and clustering methods that have been developed for non-time series data. In this approach, we can found two different approximations:

- The first one is based on the use of a reduced set of characteristics with a strong theoretical basis that is easily interpretable. In addition to applying traditional learning algorithms to the problem, this approach offers the possibility to analyze the extracted parameters and to obtain additional information.

  Based on this approach, we can find proposals that, with a minimum of four initial characteristics such as mean, typical deviation, skewness, and kurtosis, are able to face the problem of the classification synthetic control chart patterns used in the statistical process control [44]. There are also proposals, focused on the improvement of accuracy, based on the creation of an ensemble for classification, composed of trained classifiers on different representations of the time series [2]: power spectrum, autocorrelation function, and a principal components space. The final classification is obtained from a weighted voting scheme. In the field of clustering, some proposals use characteristics of time series such as trend, seasonality, non-linearity, among others, which are very appropriate to express the behavior of a time series [60].

  In this approach, we also find proposals that aim to generate synthetic time series with a given behavior as close as possible to a real time series [33]. This work contains a selection of the main characteristics of a time series. Its objective is to use them to generate time series with a real behavior with these controllable parameters.

- The second approach focuses on applying a large number of different operations to obtain a great set of descriptive parameters of the time series analyzed. In this approach, the selection of the characteristics of interest resides in the learning algorithm used on the transformed dataset. Having a much greater set of characteristics than the first approach allows

us to capture a higher number of behaviors of interest, improving the results of the algorithms applied afterward. But it is hard to extract useful information because there are a large number of characteristics to analyze. In addition, it is possible that a large part of the selected characteristics is not as explanatory as the characteristics with a strong theoretical base such as trend, seasonality, among others.

In this area, we can find different proposals. For example, the use of 8,651 operations on a set of 875 time series [30], coming from different fields, with the aim of extracting the different possible structural behaviors. Another of its objectives is to find possible interrelations between time series coming from different fields. Given the rearrangement of the rows (original time series) in the final matrix of characteristics, based on the similarity between the different operations calculated, this work can be included within the field of clustering. Another objective of the previous work would be to find a shared underlying structure between time series belonging or not to the same scope.

In a more controlled environment, within the reference problems of classification of time series, we found a similar proposal to the previous one. In this case, the authors seek to obtain the best classification results by working on the transformed dataset [29]. It has almost 9,000 characteristics, being of special importance the way to select the variables of interest. This proposal opted for the selection of the combination of variables that offers the best classification results, using the following procedure:

In the first place, the proposal selects the variable that obtains the best classification result by itself. Then, one by one, it combines the previously selected variable with the rest of the variables, and the variable that offers the best results is selected as the second variable. This set of two variables is then combined with each of the other variables and evaluated. This process is repeated until the stop criterion is met. However, this proposal entails a high computational complexity due to a large number of combinations available.

2.3. Deep Learning Classification

The approach based on deep learning has gained popularity recently [26]. Although it is usually related to the processing of images, it has very interesting proposals in the field of TSC [62]. We can distinguish between two main groups inside this approach: Generative Models and Discriminative Models.

- In the Generative Models, there is usually a previous step of unsupervised training to the learning phase of the classifier. Depending on the approach, two subgroups can be differentiated: Auto Encoders and Echo State Networks. In the case of Auto Encoders, there are a large number of proposals, for example, to model the time series before the classifier is included in an unsupervised pre-training phase such as Stacked Denoising Auto-Encoders (SDAEs) [10]. A Recurrent Neural Network (RNN) Auto Encoder [49] was designed to generate time series first and then use the

learned representation to train a traditional classifier. After that, it predicts the class of the new input time series. A model based on Convolutional Neural Networks (CNN) [57] was proposed where the authors introduced a deconvolutionary operation followed by an upsampling technique that helps to reconstruct a multivariate time series. In the case of the Echo State Networks, these networks were used to reconstruct time series and use the representation learned in the space reservoir for classification. They were also used to define a kernel on the learned representations and apply an MLP or SVM as a classifier.

- In the case of Discriminative Models, these are a classifier or regressor that learns the mapping between the input values of the time series and returns the probability distribution over the class variable of the problem. In this case, we can differentiate two subgroups: Feature Engineering and End-to-End. The typical case of use of Feature Engineering is the transformation of the time series into images, using different techniques such as recurrence plots [31] and Markov transition fields [61], and introduce that information in a deep learning discriminating classifier [45]. In contrast, the End-to-End approach incorporates feature learning while adjusting the discriminative classifier.

If we look at the TSC problem, we see that the CNNs are the most used architectures, mainly due to their robustness and their relatively short training time, compared to other types of networks. One of the best-known architectures is the Residual Networks (ResNets) [62]. This proposal adds linear shortcuts for the convolutional layers, potentially improving the accuracy of the model.

3. Time series complexity measures and features

The complexity of a time series represents the interrelationship that exists between its different elements. A greater number of interrelations between the elements of a time series indicates a greater complexity. Once these interrelations have been found and understood, we can try to find the mechanisms that produce this complexity. In this way, it is possible to explain the behavior of a time series based on these mechanisms. In other words, these interrelations are characteristic of the time series.

The features of a time series explain certain behavioral characteristics of the time series itself. The features that traditionally have been used in the process of analysis of a time series as seasonality, trend, stationarity, among others, are well documented [33][8]. These types of characteristics can describe the behavior of a time series efficiently. There are other types of characteristics that provide small pieces of information about the behavior of a time series, such as mean, maximum value, minimum value, variance, among others. Although the latter is not usually employed in the analysis process, they are features that may be especially useful depending on the problem. For example, in a classification

Table 1: Complexity measures selected.

Char.	Name	Description	Ref.
C_1	lempel_ziv	LempelZiv (LZA)	[36]
C_2	aproximation_entropy	Aproximation Entropy	[47]
C_3	sample_entropy	Sample Entropy (DK Lake in Matlab)	[52]
C_4	permutation_entropy	Permutation Entropy (tsExpKit)	[9]
C_5	shannon_entropy_CS	Chao-Shen Entropy Estimator	[16]
C_6	shannon_entropy_SG	Schurmann-Grassberger Entropy Estimator	[55]
C_7	spectral_entropy	Spectral Entropy	[64]
C_8	nforbiden	Number of forbiden patterns	[1]
C_9	kurtosis	Kurtosis, the "tailedness" of the probability distribution	[23]
C_{10}	skewness	Skewness, asymmetry of the probability distribution	[20]

problem where time series of different classes have significant differences in their value ranges, the mean can be very helpful.

This work presents a novel ensemble of complexity measures and features of time series, aimed at solving problems of classification of time series by applying traditional classification algorithms. It also aims to obtain interpretable results. The characteristics selected in this paper, composed of complexity measures and time series features, are based on information theory and seek to provide greater knowledge about the underlying structure of the processed time series. A set of characteristics, based on measures of complexity, is summarized in Table 1.

In addition to the features mentioned above, we have added a set of time series features. It has been selected based on its theoretical basis, also taking into account its historical importance in the field of time series and its interpretability [33]. This set of measures, based mostly on typical characteristics of time series, is summarized in Table 2.

The possible interrelation between the different selected operations has also been analyzed, eliminating those that reached high correlation values.

The objective of using such characteristics is to obtain an alternative and interpretable representation of the behavior of a time series. This representation allows us to use traditional classification algorithms and obtain interpretable results. This way, if a classification algorithm is applied that offers an interpretable model, we can explain the classification based on characteristics that describe the behavior of the processed time series. We can obtain information beyond the simple visual behavior of a time series.

The theoretical explanation of each of the measures has not been included in this paper due to space constraints. For the convenience of the reader, they are available online in the web resource [2] associated with this work.

[2]Complexity Measures and Features for Times Series classification. `http://dicits.ugr.`

Table 2: Time series features selected.

Char.	Name	Description
C_{11}	x_acf1	First autocorrelation coefficient
C_{12}	x_acf10	Sum of squares of the first 10 autocorrelation coefficients
C_{13}	diff1_acf1	Differenced series first autocorrelation coefficients
C_{14}	diff1_acf10	Differenced series sum of squares of the first 10 autocorrelation coefficients
C_{15}	diff2_acf1	Twice differenced series first autocorrelation coefficients
C_{16}	diff2_acf10	Twice differenced series sum of squares of the first 10 autocorrelation coefficients
C_{17}	max_kl_shift	Maximum shift in Kullback-Leibler divergence between two consecutive windows
C_{18}	time_kl_shift	Instant of time in which the Maximum shift in Kullback-Leibler divergence between two consecutive windows is located
C_{19}	outlierinclude _mdrmd	Calculates the median of the medians of the values, while adding more outliers
C_{20}	max_level_shift	Maximum mean shift between two consecutive windows
C_{21}	time_level_shift	Instant of time in which the maximum mean shift between two consecutive windows is located
C_{22}	ac_9	Autocorrelation at lag 9
C_{23}	crossing_points	The number of times a time series crosses the median line
C_{24}	max_var_shift	Maximum variance shift between two consecutive windows
C_{25}	time_var_shift	Instant of time in which the maximum variance shift between two consecutive windows is located
C_{26}	nonlinearity	Modified statistic from Teräsvirta's test
C_{27}	embed2_incircle	Proportion of points inside a given circular boundary in a 2-d embedding space
C_{28}	spreadrandomlocal _meantaul	Mean of the first zero-crossings of the autocorrelation function in each segment of the 100 time-series segments of length l selected at random from the original time series
C_{29}	flat_spots	Maximum run length within any single interval obtained from the ten equal-sized intervals of the sample space of a time series
C_{30}	x_pacf5	Sum of squares of the first 5 partial autocorrelation coefficients
C_{31}	diff1x_pacf5	Differenced series sum of squares of the first 5 partial autocorrelation coefficients
C_{32}	diff2x_pacf5	Twice differenced series sum of squares of the first 5 partial autocorrelation coefficients
C_{33}	firstmin_ac	Time of first minimum in the autocorrelation function
C_{34}	std1st_der	Standard deviation of the first derivative of the time series
C_{35}	stability	Stability variance of the means
C_{36}	firstzero_ac	First zero crossing of the autocorrelation function
C_{37}	trev_num	The numerator of the trev function, a normalized nonlinear autocorrelation, with the time lag set to 1
C_{38}	alpha	Smoothing parameter for the level-alpha of Holt's linear trend method
C_{39}	beta	Smoothing parameter for the trend-beta of Holt's linear trend method
C_{40}	nperiods	Number of seasonal periods (1 for no seasonal data)
C_{41}	seasonal_period	Seasonal periods (1 for no seasonal data)
C_{42}	trend	Strength of trend
C_{43}	spike	Spikiness variance of the leave-one-out variances of the remainder component
C_{44}	linearity	Linearity calculated based on the coefficients of an orthogonal quadratic regression
C_{45}	curvature	Curvature calculated based on the coefficients of an orthogonal quadratic regression
C_{46}	e_acf1	First autocorrelation coefficient of the remainder component
C_{47}	e_acf10	Sum of the first then squared autocorrelation coefficients
C_{48}	walker_propcross	Fraction of time series length that walker crosses time series
C_{49}	hurst	Long-memory coefficient
C_{50}	unitroot_kpss	Statistic for the KPSS unit root test with linear trend and lag one
C_{51}	histogram_mode	Calculates the mode of the data vector using histograms with 10 bins (It is possible to select a different number of bins)
C_{52}	unitroot_pp	Statistic for the PP unit root test with constant trend and lag one
C_{53}	localsimple_taures	First zero crossing of the autocorrelation function of the residuals from a predictor that uses the past trainLength values of the time series to predict its next value
C_{54}	lumpiness	Lumpiness variance of the variance
C_{55}	motiftwo_entro3	Entropy of words in the binary alphabet of length 3. The binary alphabet is obtained as follows: Time-series values above its mean are given 1, and those below the mean are 0

Our proposal consists of a set of characteristics that allow us to classify in a better way the time series and to obtain interpretable results. The pseudocode in Algorithm 1 shows how our proposal works.

Our proposal begins with an individual and independent processing of each time series (line 1). The selected set of characteristics is calculated for each time series, obtaining a set of results with as many values as features applied to the time series. By processing the whole set of input time series, we calculate a matrix of values with as many columns as applied features and as many rows as processed time series. This matrix is a representation of the input time series, free of any time dependency, based on the parameters obtained when applying the operations mentioned above. As there is no time dependency in the new dataset, it is possible to use any traditional classification algorithm on this new dataset.

Although most of the proposed characteristics are specially designed to be applied over time series, in some cases, these characteristics may not be defined for some specific time series. In these cases, undesirable values are produced, and we must process them. In the first place, we differentiate between the cases in which we obtain infinite values and those we do not. For this reason, the results obtained are filtered, detecting the cases of noninfinite values and transforming to the same value (lines 2-5) for subsequent elimination or imputation. On the training set, we check for each column (operation applied) that the amount of these values is less than 20% of the total. In other cases, the column is removed from both the training set and the test set (lines 6-11). Infinite values are identified as positive or negative and replaced by the maximum or minimum value of the corresponding column, respectively, ignoring the infinite values in these calculations (lines 12-22). Imputation of missing values based on the mean is then applied to each column (lines 23-25), eliminating any presence of unwanted values in the datasets.

Since one of our objectives is to obtain interpretable results, in the second part of our proposal, we have selected the main classification algorithms based on trees: C5.0, C5.0 with boosting [48], Rpart [58] and Ctree [32]. We have selected this type of algorithms by the interpretability of the generated models. The accuracy of the models obtained on the test set is an objective indication of the quality or fidelity of the representation obtained by the set of selected features. We initialize a variable that contains the results obtained for each one of the processed models (line 26). In the final part, we calculate each selected model, make the corresponding prediction, and calculate the accuracy. Finally, all these results are stored (lines 27-32). Our proposal returns these results together with the training and test sets with the new calculated characteristics (line 33).

Figure 1 shows, in a graphic form, the process of calculating the characteristics of the time series.

At this point, it is necessary to proceed to the analysis of the trees obtained in search of an interpretable result that, in many cases, is difficult to appreciate

---

`es/papers/CMFTS/`

Algorithm 1 Main procedure

Input:
train: train dataframe with (Ts_class, Ts_values)
test: test dataframe with (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
f_train: characteristics train dataframe
f_test: characteristics test dataframe

```
f_train, f_test ← calc_cmfts((train.Ts_values, test.Ts_values), all)
for each value in (f_train, f_test) do
    if (is.na(value) || is.nan(value)) then value ← NA
    end if
end for
for each column in f_train do
    if (count.na(column) ≥ (length(column)*0.2)) then
        f_train ← f_train[ , -column.index]
        f_test ← f_test[ , -column.index]
    end if
end for
for each column in f_train do
    for each value in (f_train[ , column.index], f_test[ , column.index]) do
        if (is.infinite(value)) then
            if (value ≥ 0) then
                value ← max(f_train[ , column.index], ignore.inf)
            else
                value ← min(f_train[ , column.index], ignore.inf)
            end if
        end if
    end for
end for
for each column in (f_train, f_test) do
    column ← impute.Mean(column)
end for
output_data ← NULL
for each model in models do
    fit ← model.train(f_train, train.Ts_class)
    pred ← fit.predict(f_test)
    acc ← accuracy(pred, test.Ts_class)
    output_data.add(fit, pred, acc)
end for
return (output_data, f_train, f_test)
```

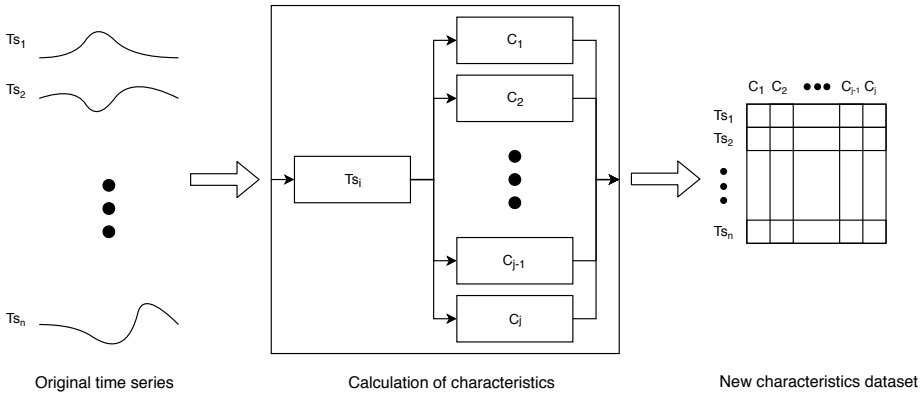


Figure 1: Characteristics calculation workflow.

in the original time series.

4. Empirical Study

In this section we evaluate the performance of our proposal. In order to do this, we first show the experimental design carried out followed by the results obtained with their corresponding analysis.

4.1. Experimental Design

In this section, we show the measures used to evaluate the performance of our proposal, the datasets processed, the classification models selected and the hardware used in the experimentation.

The source code of our proposal and experimentation has been developed in R 3.4.4 and can be found in the online repository [3].

4.1.1. Performance measures

We have chosen accuracy as a basic measure of performance. Accuracy is calculated as the number of correctly classified instances in the test set divided by the total number of cases in the test set. We use the average rank to compare the performance of the different models against each other. Having over a large number of datasets, very different from each other, a relative performance measure like the rank is one of the best options to make the desired comparison. Since the results can vary greatly from one dataset to another we have chosen to use the Critical Difference diagram (CD) [24]. CD allows making a comparison of results, between the different models, from a statistical point of view. In this diagram, the models linked by a bold line can be considered to have no

[3] Complexity Measures and Features for Times Series classification. `https://github.com/fjbaldan/CMFTS/`

statistically significant differences in their results at a given confidence level α. In this paper, we have chosen a 95% confidence level setting an α of 0.05. We have used the R *scmamp* package to calculate average rank and the CD. In addition, we include the Win/Loss/Tie ratio to be able to observe in a direct quantitative way the performance of each model in comparison with the rest.

4.1.2. Datasets

The used datasets have been extracted from the UCR repository [22], which is the reference repository in the field of TSC. It is composed of 128 datasets. The authors of the repository have run the main algorithms of the state of the art of TSC on 112 of the 128 datasets. They eliminated 15 datasets because of containing time series of different lengths and the Fungi dataset because it contains only one instance per class in the training data. Given the great number of algorithms run on these datasets, we can consider the 112 selected datasets as the state of the art in TSC datasets.

4.1.3. Models

The main tree classification algorithms have been selected based on their interpretability: C5.0, C5.0 with boosting (C5.0B) [48], Rpart [58], and Ctree [32]. 1NN+ED, 1NN+DTW(w=100) and 1NN+DTW(w_learned) applied over the original time series have been included as benchmark methods since they are the benchmark TSC methods. The new representation of time series that we propose in this work offers an additional information about these series that can also be used by less interpretable algorithms to improve the obtained results. For this purpose, classification algorithms with greater complexity and better accuracy performance have been selected like RF [14], and SVM [21]. We have also added 1NN+ED applied to the proposed features as a benchmark method. We name the models based on the features proposed in this work following the CMFTS+Model pattern, for example, CMFTS+RF, CMFTS+C5.0, etc.

In order to evaluate our proposal, we have selected only the main algorithms of the state of the art that have been run on the 112 datasets previously mentioned. The algorithms selected are: HIVE-COTE, STC, ResNet, WEASEL, BOSS, cBOSS, c-RISE, TSF, and Catch22. We do not include the model FEARS because there are not public results over the 112 selected datasets, and we were not able to reproduce the results of the original work.

4.1.4. Hardware

For our experiments, we have used a server with the following characteristics: 4 $\times$ Intel(R) Xeon(R) CPU E5-4620 0 @ 2.20GHz processors, 8 cores per processor with HyperThreading, 10 TB HDD, 512 GB RAM. We have used the following software configuration: Ubuntu 18.04, R 3.6.3.

4.2. Results

In this section, we show and evaluate the results obtained by our proposal both in terms of performance (Section 4.2.1) and interpretability (Section 4.2.2).

Since the complete empirical results are too extensive to include in the paper, we have put just a summary. The complete set is available at web resource[4] associated to this work.

4.2.1. Performance results

Table 3 shows the results obtained for the 112 datasets processed. We show the average accuracy, average rank, and Win/Loss/Tie Ratio, for all the feature-based learning models (CMFTS) proposed in this paper and the benchmark models in TSC.

Table 3: Comparative results of the proposed feature-based models (CMFTS) and the TSC benchmark models. The best results are stressed in bold.

Model	Average Acc.	Average Rank	W/L/T Ratio
CMFTS+C5.0	0.724	6.442	3/109/2
CMFTS+C5.0B	0.766	4.263	12/100/3
CMFTS+Rpart	0.682	7.071	4/108/1
CMFTS+Ctree	0.652	7.683	4/108/2
CMFTS+RF	**0.807**	**2.567**	**48/64/4**
CMFTS+SVM	0.764	4.21	14/98/5
CMFTS+1NN-ED	0.737	5.996	8/104/4
1NN-ED	0.694	6.388	9/103/9
1NN-DTW (learned_w)	0.752	4.71	23/89/11
1NN-DTW (w=100)	0.73	5.67	16/96/5

If we look at the results of the average rank, Table 3, we see that the CMFTS+RF model obtains the best results, followed by CMFTS+SVM, CMFTS+C5.0B, and 1NN-DTW (learned_w). This shows that more complex models such as RF, C5.0B, SVM, and 1NN-DTW (learned_w) offer better results than more simple models such as C5.0, Rpart, and Ctree. This behavior is also visible in the Win/Loss/Tie Ratio, where CMFTS+RF is the best model, with 48 wins, followed by 1NN-DTW (learned_w) with 23 wins. The third, fourth and fifth places are taken by 1NNN-DTW (w=100) (16 wins), CMFTS+SVM (14 wins), and CMFTS+C5.0B (12 wins), respectively.

In order to make a statistically robust comparison between the different models, we used the CD shown in Figure 2, with a confidence level of 95%. The CD diagram shows that there is no statistical relationship between CMFTS+RF and the other models, being CMFTS+RF the model most interesting of the tested set. We also see how there are no statistically significant differences between the CMFTS+SVM, CMFTS+C5.0B, and 1NN-DTW (learned_w) models, being the CMFTS+C5.0B model the one with a higher degree of interpretability. Those results allow us to aspire to have interpretable models with competitive results.

[4]Complexity Measures and Features for Times Series classification. `http://dicits.ugr.es/papers/CMFTS/`

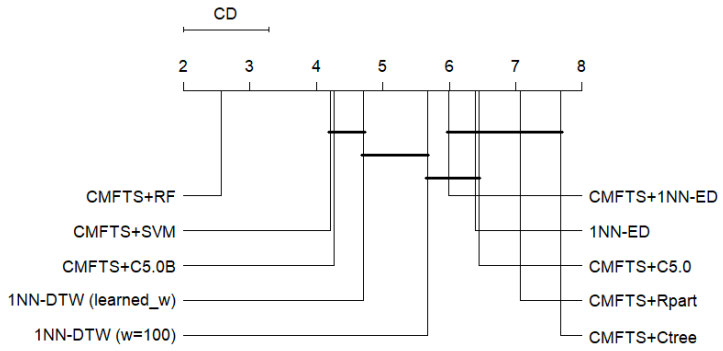


Figure 2: Critical Difference diagram between the proposed feature-based models (CMFTS) and the TSC benchmark models, confidence level of 95%.

Finally, we see how CMFTS+1NN-ED slightly improves the results of its direct competitor 1NN-ED and the remaining of the tree-based models (C5.0, Rpart, and Ctree). But the differences are not significant from a statistical point of view.

Once the best models of our proposal have been identified, we will compare them with the best models of the state of the art. The best models of our proposal selected for this comparison are CMFTS+RF, CMFTS+SVM, CMFTS+C5.0B, and CMFTS+1NN-ED. CMFTS+RF and CMFTS+SVM are the models that obtain the best results, although their interpretability is reduced. CMFTS+C5.0B is the most interpretable model with the best results if we compare it with the rest of the tree-based models. CMFTS+1NN-ED is a simple model that we can use as a benchmark. As in the previous case, for a first analysis, we use a table with the results of average accuracy, average rank, and Win/Loss/Tie Ratio, Table 4. In addition, to carry out an analysis from a statistical point of view we use the CD, Figure 3.

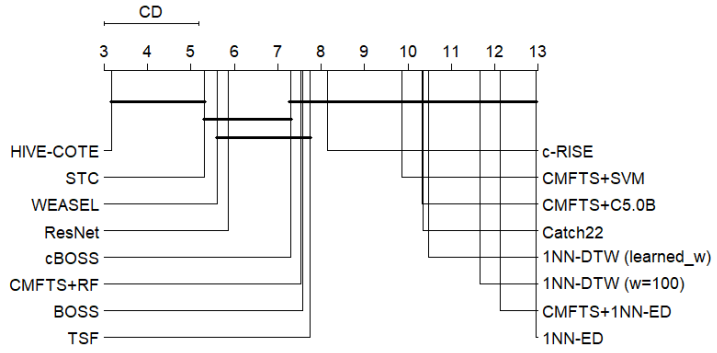
In Table 4, we see the HIVE-COTE algorithm has the best results in average rank, average accuracy, and win/loss/tie ratio. This algorithm should be used whenever possible. STC is the second method with the lowest average rank and higher average accuracy, but the third in the win/loss/tie ratio. STC can obtain good results in a great number of cases, but not the best results. This behavior indicates that STC offers competitive and robust results in different fields. WEASEL has a behavior very similar to STC. It is the third method in the average rank results, and it has a win/loss/tie ratio and average accuracy results lower but very close to the STC results. For the same win/loss ratio values, WEASEL obtains a higher number of ties than STC. Both methods offer a good start point. RestNet is the fourth method in average rank, but the second one on the win/loss/tie ratio. This behavior indicates that it works better in certain cases, obtaining the best results in a higher number of cases in comparison with STC and WEASEL. In another way, RestNet has worse average performance. If we analyze our proposals, we could observe that CMFTS+RF offers the best

results on the average rank, win/loss/tie ratio, and average accuracy.

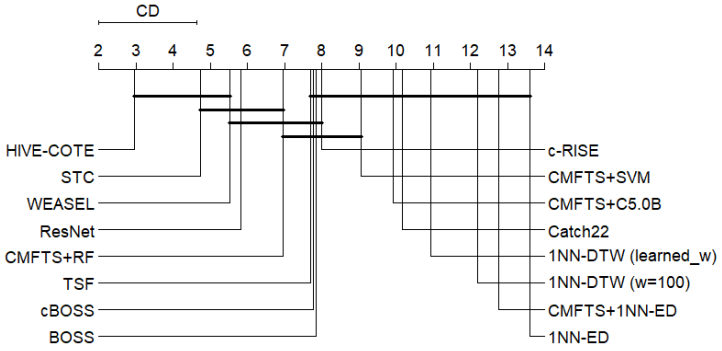
Table 4: Comparative results of the proposed feature-based models (CMFTS) and the TSC state of the art models. The best results are stressed in bold.

Model	Average Acc.	Average Rank	W/L/T Ratio
CMFTS+RF	0.807	7.531	10/102/5
CMFTS+SVM	0.764	9.871	5/107/2
CMFTS+C5.0B	0.766	10.321	1/111/0
CMFTS+1NN-ED	0.737	12.116	4/108/4
BOSS	0.815	7.58	12/100/12
Catch22	0.769	10.353	3/109/2
cBOSS	0.818	7.29	15/97/13
c-RISE	0.79	8.156	7/105/5
HIVE-COTE	**0.864**	**3.17**	**42/70/17**
ResNet	0.82	5.866	33/79/9
STC	0.845	5.308	18/94/9
TSF	0.786	7.741	9/103/7
WEASEL	0.834	5.603	18/94/12
1NN-ED	0.694	12.955	1/111/1
1NN-DTW (learned_w)	0.752	10.473	4/108/3
1NN-DTW (w=100)	0.73	11.665	8/104/5

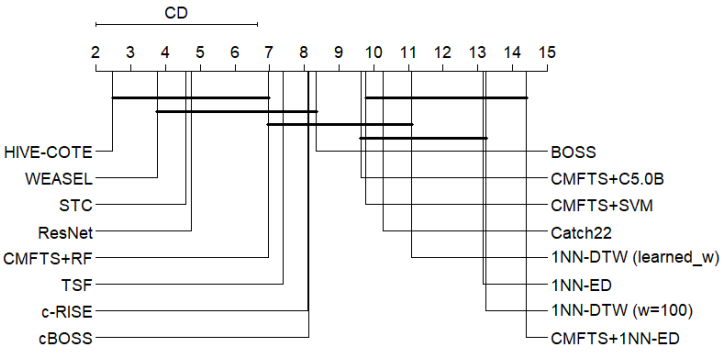
If we analyze Figure 3, we can observe statistical relationships between our proposal CMFTS+RF and the algorithms HIVE-COTE, STC, and WEASEL, with some conditions. In Figure 3a, there are four principal subgroups of proposals without statistical differences between their results over the 112 selected datasets. In this case, HIVE-COTE and STC compose the group with the best results. We can observe that the last group is composed of twelve proposals, which is an interesting behavior. We see how CMFTS proposals are included in this group, but CMFTS+RF is included in another group where its results do not differ statistically from those obtained by WEASEL. If we increase the minimum number of instances per dataset, the observed subgroups can vary significantly. Normally, the features-based approach performs worse in datasets with a low number of instances. In Figure 3b, using datasets with 100 instances or more, we have five different subgroups of models. Now, the best group is composed of HIVE-COTE, STC, and WEASEL. In this case, we see how the results of our best proposal, CMFTS+RF, have not statistical differences with STC and WEASEL models, which are included in the first group. In Figure 3c, using datasets with 500 instances or more, we see how the results of CMFTS+RF have not statistical differences with the best model, HIVE-COTE, since CMFTS+RF has been included in the first group. In this case, we can see how WEASEL is the second best model. Those results support the idea that the number of instances affects the results of the features-based methods.



(a) Full UCR repository, 112 datasets.



(b) Datasets with 100 or more instances, 76 datasets.



(c) Datasets with 500 or more instances, 25 datasets.

Figure 3: Critical Difference diagrams between the proposed feature-based models (CMFTS) and the TSC state of the art models, confidence level of 95%. Different scenarios.

4.2.2. Interpretability

In this section, we analyze the interpretability of the results obtained by our proposals. We also see the advantages of our proposal in terms of the robustness of results.

In Figure 4a, we show an example of each of the classes present in the TSC problem called *GunPoint*. It is a problem that differentiates whether a person has a weapon in his hands or not. The time series that compose this problem comes from the center of mass of the right hand of the person holding or not a weapon. Visually it is appreciated that, in the case of having a gun, the peak present in this temporal series is more pronounced than in the case of not having it.

In Figure 4b, we see the first classification tree C5.0 obtained by our proposal, CMFTS+C5B. In this tree, we observe how two features like the *stability*, as the variance of the means obtained from tiled windows, and the *shannon entropy SG*, with Bayesian estimates of the bin frequencies using the Dirichlet-multinomial pseudo-counting model, can differentiate a large part of the cases that belong to a class. If we compare these results with Figure 4c, where the values of some instants of time are the ones that determine if a case belongs to different classes, we can see how our proposal offers a robust behavior to problems as simple as the desynchronization of the temporal series.

The interpretability of the results is strongly linked to the importance given by each algorithm to each of the input features, whenever it is possible. For this reason, we have selected our best proposal, CMFTS+RF, that measures the importance of each feature through the Gini Index [15]. We have analyzed the accumulated importance of each feature over the 112 datasets and the importance of each feature in each dataset.

Figure 5 shows the mean results of the importance of the features obtained on the 112 datasets used. We see how characteristics related to entropy, such as *sample_ entropy* and *aproximation_ entropy* achieve the highest valuation in importance. Interpretable characteristics such as *linearity*, *curvature*, *spike*, and *skewness* would occupy the following positions of importance. On the other hand, we can see two characteristics that have zero importance: *nperiods* and *seasonal_ period*. Given the high number of datasets, a no-preprocessing of the data approach has been chosen, specifying a zero frequency for every time series. This causes the calculation of *nperiods* and *seasonal_ periods* to always get the same value. In a real case, different parameters can be specified that would allow different values to be obtained in these characteristics. The previous characteristics are especially interesting in the field of time series, so we have decided to keep them in the CMFTS package.

We use a heat map to be able to analyze the importance of each feature on each dataset, Figure 6. As we can see in Figure 6, there are a lot of differences in the feature importance scores between different datasets. It means that each problem has very several characteristics and behaviors, so we need different features to extract the right information on each dataset. We can differentiate into two big groups of datasets. The first one which we need a small number

of features to obtain the desired information. So, our models can obtain good enough results with this small subset of features, even if these results are not the best. The second one which our model uses a lot of features. In this case, it might be because the problem is very complex, and we need a lot of information to obtain good results. Or the features are not good enough to obtain the needed information to resolve the problem, and the model uses a lot of them trying to obtain good results. If we sort the datasets from Figure 6 in an increasing way based on the accumulated importance of the features, we can observe both groups in an easy way, Figure 7. At the top of the heat map, we can see the datasets in which our model uses a small set of features. At the bottom, we are able to see the datasets in which our proposal needs to use a lot of features. On the datasets in the order of Figure 7, if we calculate the difference between the best case of each dataset and our best model (CMFTS+RF), Figure 8, we see that this difference is lesser in the datasets at the top of Figure 7. That means that in the cases in which our model uses a small subset of features, it is able to obtain very close results to the best algorithm. These results reinforce the original idea of this proposal to obtain competitive results with simple and interpretable models.

5. Conclusion

In this work, we have presented a set of characteristics, composed of measures of complexity and representative features of time series, capable of extracting important information from the time series on which they are applied. The proposed set of features makes it possible to tackle TSC problems with traditional classification algorithms, allowing them to obtain useful and interpretable results.

We have published our proposal software to make it accessible and usable for any practitioner or researcher to use. We have published all the results obtained throughout the work to make it fully reproducible. The functioning of our proposal has been tested on 112 datasets obtained from the UCR repository. We have used tree-based classification algorithms due to their high interpretability, and they have been compared with the state of the art TSC algorithms. The results obtained by our proposal have not statistical differences with the third best algorithm of the state of the art of TSC, with a confidence level of 95%. If we focus our analysis on datasets with more than 500 time series, our proposal obtains results statistically indistinguishable from those obtained by the best state-of-the-art algorithm. This result reinforces the original idea that feature-based methods require a larger number of time series to perform correctly.

Extracting characteristics of interest from time series that are robust and interpretable provides more understandable and even better classification results in some cases. Our proposal demonstrates a robust behavior against typical TSC problems by extracting descriptive characteristics from the time series rather than working on the original series itself. In this way, additional interpretability is achieved, which is especially useful in some problems.

6. Acknowledgment

This research has been partially funded by the following grants: TIN2013-47210-P and TIN2016-81113-R both 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.

We do warm fully acknowledge insightful comments from Rob. J. Hyndman on previous versions of this paper that without doubt greatly contributed to the enhancement of this paper.

The authors would like to thank Prof. Eamonn Keogh and all the people who have contributed to the UCR TSC archive for their selfless work.

References

[1] J. Amigó, Permutation complexity in dynamical systems: ordinal patterns, permutation entropy and all that, Springer Science & Business Media, 2010.

[2] A. Bagnall, L. Davis, J. Hills, J. Lines, Transformation based ensembles for time series classification, in: Proceedings of the 2012 SIAM international conference on data mining, SIAM, 2012, pp. 307–318.

[3] A. Bagnall, M. Flynn, J. Large, J. Lines, M. Middlehurst, On the Usage and Performance of The Hierarchical Vote Collective of Transformation-based Ensembles version 1.0 (HIVE-COTE 1.0), arXiv preprint arXiv:2004.06069.

[4] A. Bagnall, G. Janacek, A run length transformation for discriminating between auto regressive time series, Journal of classification 31 (2) (2014) 154–178.

[5] A. Bagnall, J. Lines, A. Bostrom, J. Large, E. Keogh, The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances, Data Mining and Knowledge Discovery 31 (3) (2017) 606–660.

[6] A. Bagnall, J. Lines, J. Hills, A. Bostrom, Time-series classification with COTE: the collective of transformation-based ensembles, IEEE Transactions on Knowledge and Data Engineering 27 (9) (2015) 2522–2535.

[7] F. J. Baldán, J. M. Benítez, Distributed FastShapelet Transform: a Big Data time series classification algorithm, Information Sciences 496 (2019) 451 – 463.

[8] F. J. Baldán, S. Ramírez-Gallego, C. Bergmeir, F. Herrera, J. M. Benítez, A Forecasting Methodology for Workload Forecasting in Cloud Systems, IEEE Transactions on Cloud Computing 6 (4) (2018) 929–941.

[9] C. Bandt, B. Pompe, Permutation entropy: a natural complexity measure for time series, Physical review letters 88 (17) (2002) 174102.

[10] Y. Bengio, L. Yao, G. Alain, P. Vincent, Generalized denoising auto-encoders as generative models, in: Advances in neural information processing systems, 2013, pp. 899–907.

[11] D. J. Berndt, J. Clifford, Using Dynamic Time Warping to Find Patterns in Time Series, in: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, AAAIWS'94, AAAI Press, 1994, pp. 359–370.

[12] A. Bondu, D. Gay, V. Lemaire, M. Boullé, E. Cervenka, FEARS: a Feature and Representation Selection approach for Time Series Classification, in: Asian Conference on Machine Learning, 2019, pp. 379–394.

[13] A. Bostrom, A. Bagnall, Binary shapelet transform for multiclass time series classification, in: Transactions on Large-Scale Data-and Knowledge-Centered Systems XXXII, Springer, 2017, pp. 24–46.

[14] L. Breiman, Random Forests, Machine Learning 45 (1) (2001) 5–32.

[15] 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.

[16] A. Chao, T.-J. Shen, Nonparametric estimation of Shannon's index of diversity when there are unseen species in sample, Environmental and Ecological Statistics 10 (4) (2003) 429–443.

[17] S. Chauhan, L. Vig, S. Ahmad, ECG anomaly class identification using LSTM and error profile modeling, Computers in biology and medicine 109 (2019) 14–21.

[18] H. Chen, F. Tang, P. Tino, X. Yao, Model-based kernel for efficient time series analysis, in: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2013, pp. 392–400.

[19] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, The UCR Time Series Classification Archive, `www.cs.ucr.edu/~eamonn/time_series_data/` (2015).

[20] P. Čisar, S. M. Čisar, Skewness and kurtosis in function of selection of network traffic distribution, Acta Polytechnica Hungarica 7 (2) (2010) 95–106.

[21] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297.

[22] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, E. Keogh, The UCR time series archive, IEEE/CAA Journal of Automatica Sinica 6 (6) (2019) 1293–1305.

[23] L. T. DeCarlo, On the meaning and use of kurtosis, Psychological methods 2 (3) (1997) 292.

[24] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine learning research 7 (Jan) (2006) 1–30.

[25] H. Deng, G. Runger, E. Tuv, M. Vladimir, A time series forest for classification and feature extraction, Information Sciences 239 (2013) 142–153.

[26] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, P.-A. Muller, Deep learning for time series classification: a review, Data Mining and Knowledge Discovery 33 (4) (2019) 917–963.

[27] M. Flynn, J. Large, T. Bagnall, The contract random interval spectral ensemble (c-RISE): the effect of contracting a classifier on accuracy, in: International Conference on Hybrid Artificial Intelligence Systems, Springer, 2019, pp. 381–392.

[28] B. D. Fulcher, Feature-based time-series analysis, in: Feature Engineering for Machine Learning and Data Analytics, CRC Press, 2018, pp. 87–116.

[29] B. D. Fulcher, N. S. Jones, Highly comparative feature-based time-series classification, IEEE Transactions on Knowledge and Data Engineering 26 (12) (2014) 3026–3037.

[30] B. D. Fulcher, M. A. Little, N. S. Jones, Highly comparative time-series analysis: the empirical structure of time series and their methods, Journal of the Royal Society Interface 10 (83) (2013) 20130048.

[31] N. Hatami, Y. Gavet, J. Debayle, Classification of time-series images using deep convolutional neural networks, in: Tenth international conference on machine vision (ICMV 2017), vol. 10696, International Society for Optics and Photonics, 2018, p. 106960Y.

[32] T. Hothorn, K. Hornik, A. Zeileis, ctree: Conditional inference trees, The Comprehensive R Archive Network (2015) 1–34.

[33] Y. Kang, R. J. Hyndman, F. Li, et al., Efficient generation of time series with diverse and controllable characteristics, Tech. rep., Monash University, Department of Econometrics and Business Statistics (2018).

[34] E. Keogh, K. Chakrabarti, M. Pazzani, S. Mehrotra, Locally adaptive dimensionality reduction for indexing large time series databases, in: Proceedings of the 2001 ACM SIGMOD international conference on Management of data, 2001, pp. 151–162.

[35] J. Large, A. Bagnall, S. Malinowski, R. Tavenard, On time series classification with dictionary-based classifiers, Intelligent Data Analysis 23 (5) (2019) 1073–1089.

[36] A. Lempel, J. Ziv, On the complexity of finite sequences, Information Theory, IEEE Transactions on 22 (1) (1976) 75–81.

[37] J. Lin, E. Keogh, L. Wei, S. Lonardi, Experiencing SAX: a novel symbolic representation of time series, Data Mining and knowledge discovery 15 (2) (2007) 107–144.

[38] J. Lin, R. Khade, Y. Li, Rotation-invariant similarity in time series using bag-of-patterns representation, Journal of Intelligent Information Systems 39 (2) (2012) 287–315.

[39] J. Lin, Y. Li, Finding structural similarity in time series data using bag-of-patterns representation, in: International Conference on Scientific and Statistical Database Management, Springer, 2009, pp. 461–477.

[40] J. Lines, L. M. Davis, J. Hills, A. Bagnall, A shapelet transform for time series classification, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2012, pp. 289–297.

[41] C. H. Lubba, S. S. Sethi, P. Knaute, S. R. Schultz, B. D. Fulcher, N. S. Jones, catch22: CAnonical Time-series CHaracteristics, CoRR abs/1901.10200.

[42] R. Markovič, M. Gosak, V. Grubelnik, M. Marhl, P. Virtič, Data-driven classification of residential energy consumption patterns by means of functional connectivity networks, Applied energy 242 (2019) 506–515.

[43] A. Mueen, E. Keogh, N. Young, Logical-shapelets: an expressive primitive for time series classification, in: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2011, pp. 1154–1162.

[44] A. Nanopoulos, R. Alcock, Y. Manolopoulos, Feature-based classification of time-series data, International Journal of Computer Research 10 (3) (2001) 49–61.

[45] H. F. Nweke, Y. W. Teh, M. A. Al-Garadi, U. R. Alo, Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges, Expert Systems with Applications 105 (2018) 233–261.

[46] A. R. S. Parmezan, G. E. Batista, A study of the use of complexity measures in the similarity search process adopted by knn algorithm for time series prediction, in: Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on, IEEE, 2015, pp. 45–51.

[47] S. M. Pincus, Approximate entropy as a measure of system complexity., Proceedings of the National Academy of Sciences 88 (6) (1991) 2297–2301.

[48] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[49] D. Rajan, J. J. Thiagarajan, A generative modeling approach to limited channel ECG classification, in: 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE, 2018, pp. 2571–2574.

[50] T. Rakthanmanon, E. Keogh, Fast shapelets: A scalable algorithm for discovering time series shapelets, in: proceedings of the 2013 SIAM International Conference on Data Mining, SIAM, 2013, pp. 668–676.

[51] C. A. Ratanamahatana, E. Keogh, Making time-series classification more accurate using learned constraints, in: Proceedings of the 2004 SIAM International Conference on Data Mining, SIAM, 2004, pp. 11–22.

[52] J. S. Richman, J. R. Moorman, Physiological time-series analysis using approximate entropy and sample entropy, American Journal of Physiology-Heart and Circulatory Physiology 278 (6) (2000) H2039–H2049.

[53] P. Schäfer, The BOSS is concerned with time series classification in the presence of noise, Data Mining and Knowledge Discovery 29 (6) (2015) 1505–1530.

[54] P. Schäfer, U. Leser, Fast and Accurate Time Series Classification with WEASEL, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 637–646.

[55] T. Schürmann, P. Grassberger, Entropy estimation of symbol sequences, Chaos: An Interdisciplinary Journal of Nonlinear Science 6 (3) (1996) 414–427.

[56] P. Senin, S. Malinchik, Sax-vsm: Interpretable time series classification using sax and vector space model, in: 2013 IEEE 13th international conference on data mining, IEEE, 2013, pp. 1175–1180.

[57] W. Song, L. Liu, M. Liu, W. Wang, X. Wang, Y. Song, Representation learning with deconvolution for multivariate time series classification and visualization, in: International Conference of Pioneering Computer Scientists, Engineers and Educators, Springer, 2020, pp. 310–326.

[58] T. M. Therneau, E. J. Atkinson, et al., An introduction to recursive partitioning using the RPART routines (1997).

[59] N. Twomey, H. Chen, T. Diethe, P. Flach, An application of hierarchical Gaussian processes to the detection of anomalies in star light curves, Neurocomputing 342 (2019) 152–163.

[60] X. Wang, K. Smith, R. Hyndman, Characteristic-based clustering for time series data, Data mining and knowledge Discovery 13 (3) (2006) 335–364.

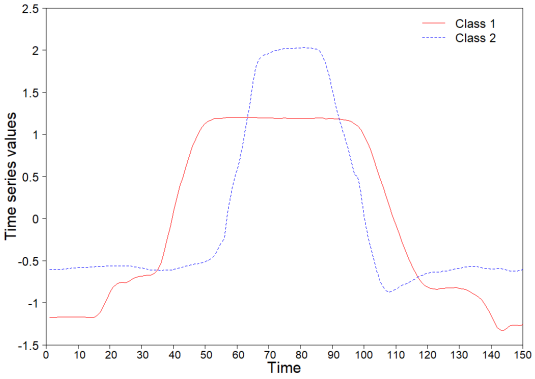
[61] Z. Wang, T. Oates, Spatially encoding temporal correlations to classify temporal data using convolutional neural networks, arXiv preprint arXiv:1509.07481.

[62] Z. Wang, W. Yan, T. Oates, Time series classification from scratch with deep neural networks: A strong baseline, in: 2017 International joint conference on neural networks (IJCNN), IEEE, 2017, pp. 1578–1585.

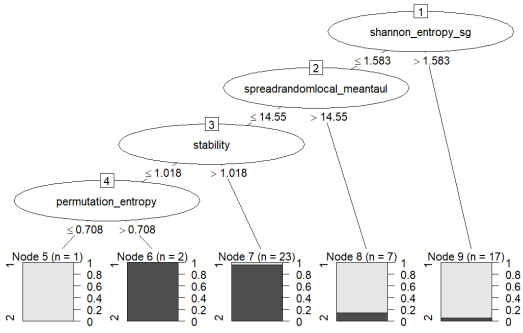
[63] L. Ye, E. Keogh, Time series shapelets: a new primitive for data mining, in: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2009, pp. 947–956.

[64] A. Zhang, B. Yang, L. Huang, Feature Extraction of EEG Signals Using Power Spectral Entropy, in: 2008 International Conference on BioMedical Engineering and Informatics, vol. 2, 2008, pp. 435–439.

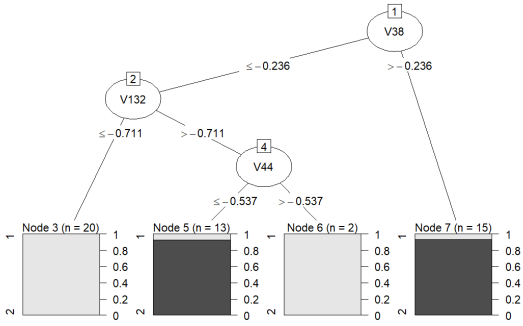
[65] R. Zhou, C. Yang, J. Wan, W. Zhang, B. Guan, N. Xiong, Measuring complexity and predictability of time series with flexible multiscale entropy for sensor networks, Sensors 17 (4) (2017) 787.



(a) GunPoint classes example.



(b) GunPoint example, first C5.0B tree with time series measures.



(c) GunPoint example, first C5.0B tree with time series original values.

Figure 4: Interpretability GunPoint dataset example.

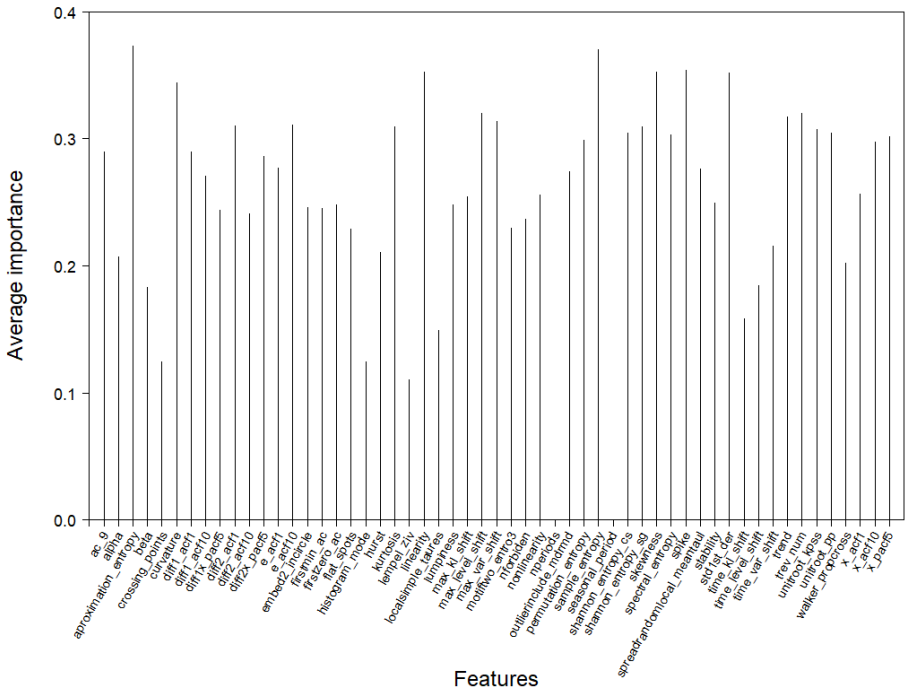


Figure 5: Average importance of features above all datasets.

Figure 6: Heat map of the importance of characteristics by dataset.

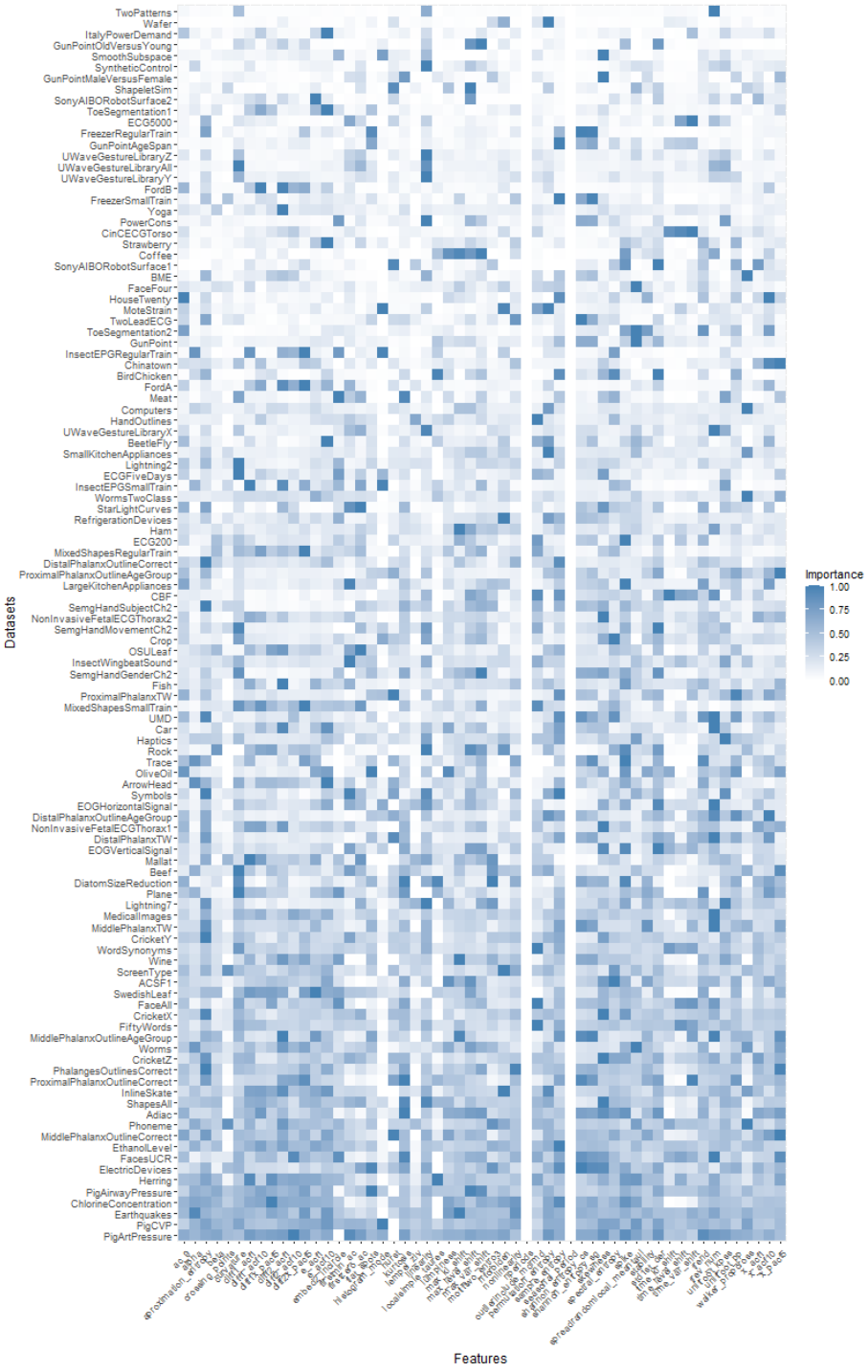


Figure 7: Heat map of the importance of characteristics by dataset, sorted by accumulated importance.

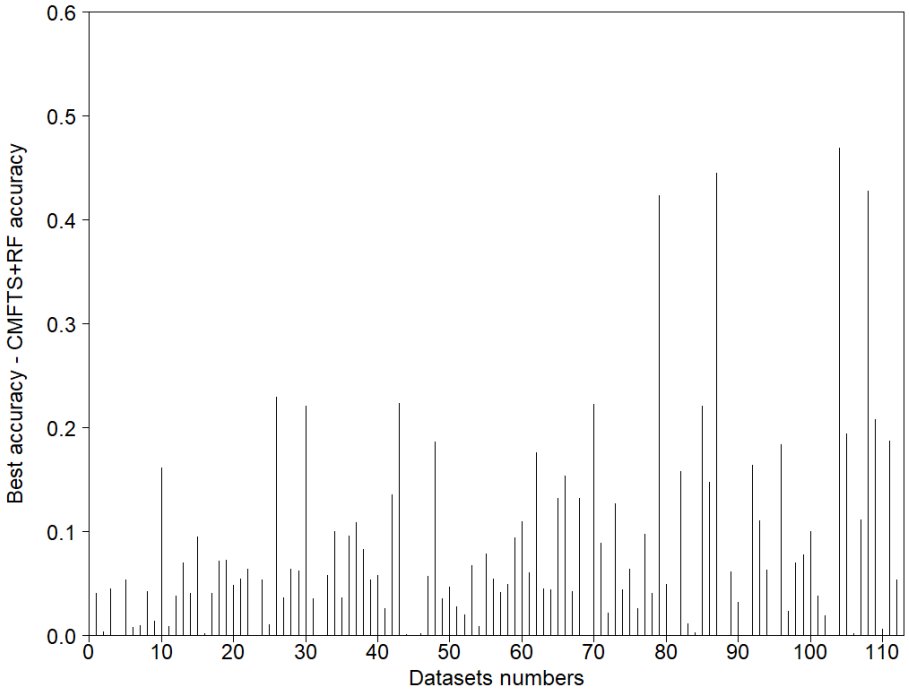


Figure 8: Accuracy differences between CMFTS+RF and the best algorithm on each dataset. The datasets are sorted like Figure 7.