Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 832 – 846

doi:10.3906/elk-1711-394

Classification of generic system dynamics model outputs via supervised time series pattern discovery

Mert EDALI[1,2], **Mustafa Gökçe BAYDOĞAN**[1*], **Gönenç YÜCEL**[1]

[1]Department of Industrial Engineering, Faculty of Engineering, Boğaziçi University, İstanbul, Turkey
[2]Department of Industrial Engineering, Faculty of Mechanical Engineering, Yıldız Technical University, İstanbul, Turkey

Received: 28.11.2017 • **Accepted/Published Online:** 29.11.2018 • **Final Version:** 22.03.2019

Abstract: System dynamics (SD) is a simulation-based approach for analyzing feedback-rich systems. An ideal SD modeling cycle requires evaluating the qualitative pattern characteristics of a large set of time series model output for testing, validation, scenario analysis, and policy analysis purposes. This traditionally requires expert judgement, which limits the extent of experimentation due to time constraints. Although time series recognition approaches can help to automate such an evaluation, utilization of them has been limited to a hidden Markov model classifier, namely the Indirect Structure Testing Software (ISTS) algorithm. Despite being used within several automated model-analysis tools, ISTS has several shortcomings. In that respect, we propose an interpretable time series classification algorithm for the SD field, which also addresses the shortcomings of ISTS. Our approach, which can highlight the regions of a certain time series that are influential in the class assignment, is an extension of the symbolic multivariate time series approach with the use of a local importance measure. We compare the performance of the proposed approach against both ISTS and nearest-neighbor (NN) classifiers. Our experiments on a SD-specific application show that the proposed approach outperforms ISTS as well as conventional NN classifiers on both noisy and nonnoisy datasets. Additionally, its class assignments are interpretable as opposed to the other approaches considered in the experiments.

Key words: System dynamics, model analysis, classification, time series, local importance, nearest-neighbor

1. Introduction

Time series classification is a data mining problem being studied in different fields such as medicine [1], finance [2], and engineering [3]. As a dynamic modeling approach, system dynamics (SD) models produce large sets of time series data and there is therefore high potential for utilization of time series classification tools. Since application-specific approaches are needed due to the characteristics of the dataset at hand, the need for time series classification tools tailored to the SD field is obvious.

Similarity-based approaches such as 1-nearest-neighbor (NN) classifiers with Euclidean or a dynamic time warping (DTW) distance have been widely and successfully used to classify time series [4–6]. Although the Euclidean distance performs well for many real-life applications [7], it is sensitive to the noise, scaling, and translation of the patterns within time series. DTW attempts to compensate for possible time translations and dilations between features, but the capability for DTW is degraded with long time series and relatively short features of interest. More importantly, NN classifiers lack interpretability, which makes it hard to understand

*Correspondence: mustafa.baydogan@boun.edu.tr

what exactly relates to the class [8]. On the other hand, it is of utmost importance to verify results in interdisciplinary teams of researchers. However, it is not easy to determine why an instance is assigned to a certain class when NN classification is used. For example, a cardiologist might be interested in the analysis of ECG signals to identify whether patients have different temporal patterns in their heart signals than a control group [9]. Seismologists aim at discriminating seismic waves to classify events such as earthquakes, mining explosions, or nuclear explosions. However, many successful classifiers fail to provide interpretable results in the time series classification domain [10].

Similar problems exist in the SD field, which is a simulation-supported methodology for studying complex dynamic problems [11]. These problems are "systemic" as the dynamic behavior emerges as a result of the internal feedback structure that is formed by the entangled relationships among system elements [12]. Since the studied problems are dynamic in nature, analyses are typically based on large sets of time series model output. As SD practitioners focus mainly on qualitative pattern features in output analysis, several field-specific tools have been developed to support and automate output analysis [13–15], most of which rely on the same time series classification algorithm, namely ISTS. ISTS is a hidden Markov model (HMM)-based classifier that is trained to classify 25 generic behavior patterns observed in dynamical systems [16, 17]. However, it has several shortcomings. In general, it has low precision and high processing time compared to state-of-the-art methods, and it performs very poorly specifically with output whose dynamics are confined in a narrow time window. More importantly, its class assignments are not directly interpretable as it does not provide any information on why a time series is assigned to a certain class.

Considering the potential problems with the existing methods, Baydogan and Runger [18] proposed symbolic representations that identify potential predictive patterns to classify time series. In this context, symbolic representation for multivariate time series (SMTS) has been shown to be successful in pattern-based representation. After transforming a time series to a pattern-based representation, SMTS trains a tree-based ensemble to classify the time series. However, SMTS also lacks the interpretability aspect since it is hard to understand why a time series is assigned to a certain class because of the multiple decision trees trained in a randomized manner (i.e. tree-based ensembles). We devise a new approach based on the local importance information proposed by the ensembles to facilitate the interpretability. To the best of our knowledge, this is the first study utilizing local information to identify important time series patterns. The novelty of this study stems from the extension of SMTS [18] for interpretability purposes.

This paper also presents a comparative analysis of alternative pattern classification approaches to recommend a novel and improved output classification approach in the SD field. The underlying motivation stems from the deficiencies of the NN classifiers and ISTS algorithm as well as novel developments in the classification field. In that respect, ISTS sets the established benchmark for our inquiry. In the scope of this study, we consider 1-nearest-neighbor (1-NN) classifiers with different distance measures and Baydogan and Runger's [18] tree-based approach for SD model output classification.

2. Background

2.1. ISTS (Indirect Structure Testing Software) algorithm

The ISTS algorithm is a continuous density HMM-based algorithm specifically developed to classify fundamental dynamic behavior patterns studied in the SD field [16]. Basically, ISTS is an ensemble of HMMs, each being trained for a particular behavior class. For a given time series, ISTS converts the series into an observation vector

and classifies it with the label of HMM that is more likely to generate such an observation. The algorithm divides the time series to be classified into 12 equal-length segments and then estimates mean, slope, and curvature values for each segment. In summary, the ISTS relies on an ensemble of HMMs trained on the features to classify 25 principal behavior types. For further information, the reader is referred to [16].

2.2. Nearest-neighbor classification

A nearest-neighbor (NN) classifier assigns a new time series to a class by considering k closest time series in the training dataset. The class of the new time series is the most encountered label among these k neighboring time series [1]. The quantification of "closeness" between time series is another important part of NN classifiers. Wang et al. [7] categorized similarity measures into four different groups as (i) lock-step measures, (ii) elastic measures, (iii) threshold-based measure, and (iv) pattern-based measure. Lock-step measures include the well-known ℓ_p-norms (e.g., Euclidean distance, Manhattan distance). Elastic measures are grouped into two: (i) dynamic time warping and (ii) edit-based distance measures. Edit-based distance measures include longest common subsequence, edit distance on real sequences, edit, and distance with real penalty.

Although lock-step similarity measures are widely used in the literature due to their simplicity, they fail to handle time series of different lengths and local time shifts where similar patterns appear at different portions of two time series being considered [19]. It was also reported that statistical error measurements such as mean squared error (MSE) and sum of squared errors (SSE) are ineffective in determining pattern-wise similarity between SD model outputs [13, 14]. Therefore, ℓ_p-norms are excluded from our study. Only elastic measures are considered in this study as they can handle time series of different lengths and shifts in temporal domain. Namely, these are dynamic time warping (DTW), longest common subsequence (LCSS), edit distance on real sequences (EDR), and edit distance with real penalty (ERP). We refer readers to [7] for further discussion of these measures. We use 1-NN classifiers (i.e. $k = 1$) with the aforementioned similarity measures as competitors. The 1-NN classifier is commonly used to understand whether the selected similarity measure is appropriate for the problem of interest [7].

2.3. Random forest

Decision trees are one of the classification and regression tools, which generate axis-parallel splits on the dataset by using only one feature at each iteration. The number of splits is generally determined by node impurity, a measure of heterogeneity of a node of the tree in terms of class distribution. Each terminal node of a decision tree represents a class label. A random forest (RF) is an ensemble of J decision trees [20], $\{g_j, j = 1, 2, ..., J\}$. Each tree is constructed using a different bootstrap sample from the original data. About one-third of the cases are left out of the bootstrap sample and not used in the construction of the single tree. These are called out-of-bag (OOB) samples. The prediction for instance x from tree g_j is $\hat{y}_j(x) = argmax_c p_j^c(x)$, where $p_j^c(x)$ is the estimated proportion of c in the corresponding leaf of the jth tree, for $c = 1, 2, \ldots, C$. Let $G(x)$ denote the set of all trees in the RF where instance x is OOB. The OOB class probability estimate of x is

$$p^c(x) = \frac{1}{G(x)} \sum_{g_j \in G(x)} I\left(\hat{y}_j(x) = c\right), \tag{1}$$

where $I(\cdot)$ is an indicator function that equals one if its argument is true and zero otherwise. The predicted class is $\hat{y}(x) = argmax_c p^c(x)$. In summary, an instance is labeled through a majority voting approach using

the tree results for which it is OOB. The estimates computed from OOB predictions are known to provide a reliable estimate of generalization error [21].

In the tree growing steps of RF, the best split is determined based on only a random sample of features. Often, the random sample size is $\sqrt{\nu}$, where ν is the number of features. The random selection reduces the variance of the classifier, and also reduces the computational complexity. Therefore, for a large number of features, an RF can be as computationally efficient as a single decision tree.

3. Symbolic time series representation with local importance

3.1. Representation learning

A univariate time series, $x^n = (x^n(1), x^n(2), \ldots, x^n(t), \ldots, x^n(T))$ is an ordered set of T values. We assume time series are measured at equally spaced time points. A time series database, X, stores N univariate time series. Time series are assumed to be of the same length, T, for illustration purposes, although the proposed method is flexible in handling the different length time series.

Most of the time series classifiers transform the raw data to an alternative representation instead of working with the observed values. This strategy allows for noise reduction or obtaining the same number of features for different length series, etc. Considering this fact, SMTS discretizes the signal space in a randomized manner to identify the informative segments related to the class with the help of tree-based learning. To avoid determination of handcrafted features from each time series, SMTS considers each observation of x^n as an instance. This is achieved by creating a matrix of instances $D_{NT\times 1}$ where

$$D_{NT\times 1} = \begin{bmatrix} \mathbf{x^1} \\ . \\ . \\ \mathbf{x^n} \\ . \\ . \\ \mathbf{x^N} \end{bmatrix} \tag{2}$$

Eq. (2) is basically the concatenation of training examples x^n. The label of each observation is assumed to be the same as the time series it belongs to. Then $D_{NT\times M}$ is mapped to a new feature space $\Phi_{NT\times 3}$ by adding the following new information: time index and first differences between consecutive observations. The row of Φ for series n at time index t is

$$[t, x^n(t), x^n(t) - x^n(t-1)]\,. \tag{3}$$

The time index, t, provides information related to the temporal relations. If time of specific observation is important for classification, a tree learner can capture this information. Moreover, the differences can be considered as empirical estimates of the first derivative and they provide trend information. The difference for the first observation of a time series is not available and it is assumed to be missing.

SMTS [18] trains an RF classifier denoted as $RFins$ (for RF applied to the instances) with J_{ins} trees trained on Φ to learn the representation. Each observation (i.e. row) is assumed to have the same class label as its time series. $RFins$ maps each instance of Φ to a terminal node of each tree. The number of terminal nodes of each tree is restricted to R and this determines the alphabet size in the approach. Use of multiple trees trained in a randomized manner is proposed to handle nonlinear boundaries by SMTS [18]. After traversal of each tree, a time series is represented by counting the terminal node assignments in each tree.

A sample discretization is illustrated in Figure 1. The plot of raw data versus time index (i.e. signal space, denoted as S) is schematized in Figure 1a (with the time index plotted on the horizontal axis and the value from a time series plotted on the vertical axis). If there are differences between the classes, there are regions in S where one class of points dominate. A tree learner trained on the signal space assuming that the observation has the same class label as the time series implicitly identifies the level of the observations that differentiates different classes. The terminal nodes for the example tree are shown in Figure 1b to illustrate the partition used for the symbolic representation. These terminal nodes correspond to the partition of signal space shown in Figure 1a. The distribution of the symbols for each time series is schematized in Figure 1c. For example, symbol E implies a region where time series observations from class 3 are frequent. Similarly, time series observations from class 2 are more frequent in the region implied by symbol D. After tree-based discretization of the time series, each time series is represented by the frequency of its observations in the terminal nodes. A classifier trained on this new representation has the potential to perform well in classification.

3.2. Classification

The trees in $RFins$ map each observation of time series in a terminal node (i.e. symbol) as illustrated in Figure 1. The final representation is obtained by concatenation of symbol distribution from each tree. Specifically, let

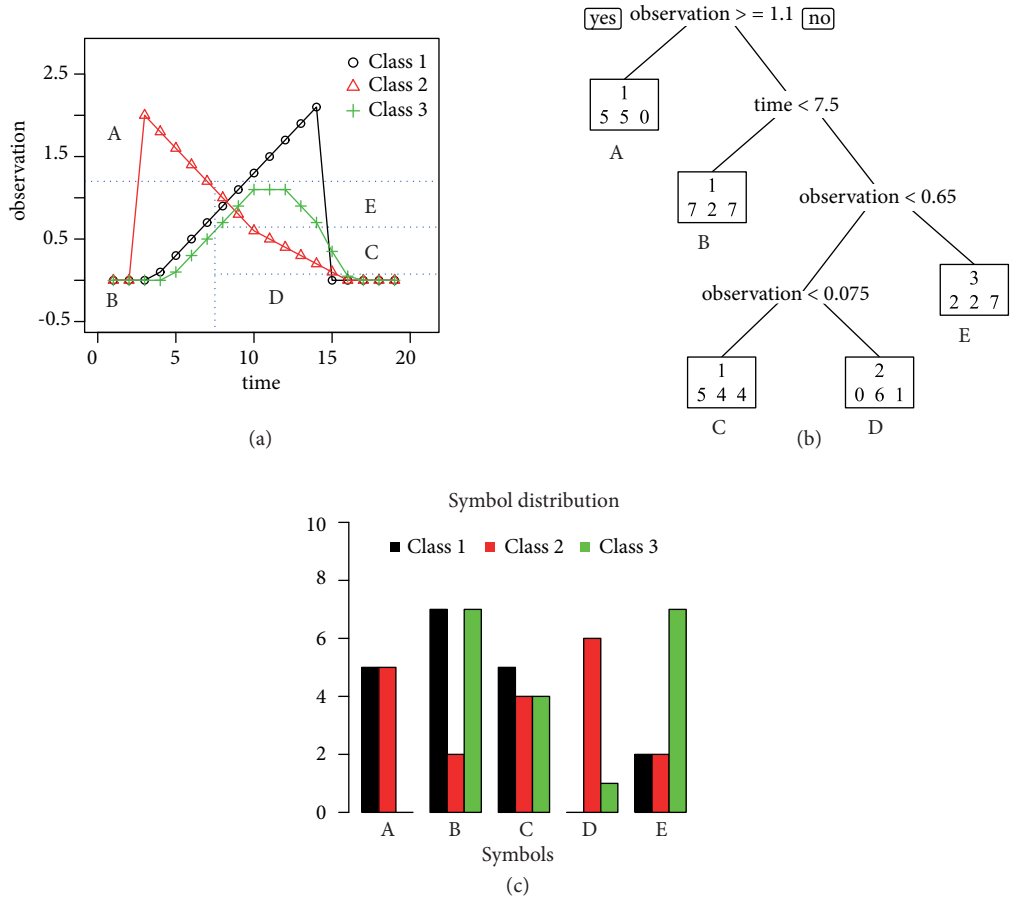

Figure 1. (a) Signal space plot of time series data from three classes. (b) A decision tree fitted to the time series data in signal space partitions the space and the terminal nodes provide symbols. (c) Symbol distribution for each time series [18].

$F_j(x^n)$ be the $R \times 1$ frequency vector of the terminal nodes from tree g_j for time series x^n. Let the function $z_j(\cdot)$ assign a row of Φ to a terminal node of tree g_j. Each entry of $F_j(x^n)$ is the proportion of time indices from time series x^n assigned to a terminal node (say r) by tree g_j. That is, the rth element of $F_j(x^n)$ is

$$\frac{\sum_{t=1}^{T} I[z_j(\Phi_t^n) = r]}{T} \tag{4}$$

for $r = 1, 2, \ldots, R$, where $I(\cdot)$ is the indicator function and Φ_t^n denotes the row t of time series x^n in Φ.

Frequency vectors of time series x^n from each of the J_{ins} trees, $F_j(x^n)$, are concatenated to represent the time series. This mapping results in a final representation, namely $H(x^n)$, which is of length $R \times J_{ins}$. Table 1 illustrates the representation for symbol frequencies with the number of terminal nodes $R = 4$. A classifier is then trained on the $H(X^n)$. SMTS [18] suggests training another RF, denoted as $RFts$ on the $H(x^n)$, since the number of features in $H(x^n)$ might be large based on the setting of R and J_{ins}. $RFts$ not only provides the classification result but also the information about the important time series patterns can be identified by exploiting the structure of $RFts$. To classify a test time series, x^0, $H(x^0)$ is obtained and $RFts$ is used to assign the class label. To understand why a time series is assigned to a certain class, we introduce a local importance measure, which is further discussed in Section 3.3.

3.3. Interpretation via local importance measures

A random forest classifier is not directly interpretable since it is a combination of multiple unpruned trees built on the random subspaces of the features. However, certain measures can be derived from the forest structure to facilitate interpretability. To identify the symbols that are informative, we utilize the local importance information from $RFts$. Each feature in the frequency-based representation refers to a symbol and the local importance of each feature provides information about the effectiveness of the symbol (i.e. a partition in the signal space) in classification for individual series. Based on the structure of $RFts$, local importance was obtained by the change in the accuracy of the classifier when the variables (i.e. symbols) were perturbed.

RF local importance for variable k (i.e. a symbol) of a time series n, $LI_k(n)$, is defined as follows. For each tree g_j in a $RFts$, consider the associated OOB sample represented by $OOB(g_j)$ (time series not included in the bootstrap sample used to train g_j). For time series n, let the proportion of votes for the correct class be v_n based on the trees in which time series n is OOB. Now, randomly permute the values of the variable k in

Table 1. A visual example of the representation based on symbol frequencies [18]. Each column denotes a symbol from a tree of $RFins$ (with $R = 4$ terminal nodes each), and each row denotes a multivariate time series. Each table entry is the proportion of time series indices (for the multivariate time series corresponding to the row) in the terminal node.

	Tree 1				Tree 2				.	.	.	Tree J_{ins}			
	A_1	B_1	C_1	D_1	A_2	B_2	C_2	D_2	.	.	.	.	.	.	.
1	0.26	0.37	0.26	0.11	0.12	0.53	0.35	0.00	.	.	.	0.63	0.00	0.12	0.25
2	0.25	0.10	0.20	0.45	0.39	0.39	0.00	0.22	.	.	.	0.11	0.53	0.21	0.15
3	0.35	0.37	0.23	0.05	0.15	0.15	0.22	0.48	.	.	.	0.42	0.42	0.00	0.16
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
N	0.40	0.11	0.19	0.30	0.16	0.37	0.21	0.26	.	.	.	0.25	0.10	0.25	0.40

$OOB\left(g_j\right)$ to get a perturbed sample denoted by $\widetilde{OOB}_k\left(g_j\right)$. The prediction based on the perturbed sample provides a new proportion of votes $\tilde{v}_n^k$ for time series n. Local importance for variable k of time series n is then equal to $LI_k\left(n\right) = v_n - \tilde{v}_n^k$. If the number of votes for the correct class decreases with the perturbed OOB data for a particular variable and time series n, that means the variable (i.e. symbol) plays an important role in the classification of this particular time series. Conversely, if the number of votes increases (or remains approximately the same), the variable is not found to be informative for instance n. For each time series, the local importance of the symbols is sorted and the top q of them are visualized.

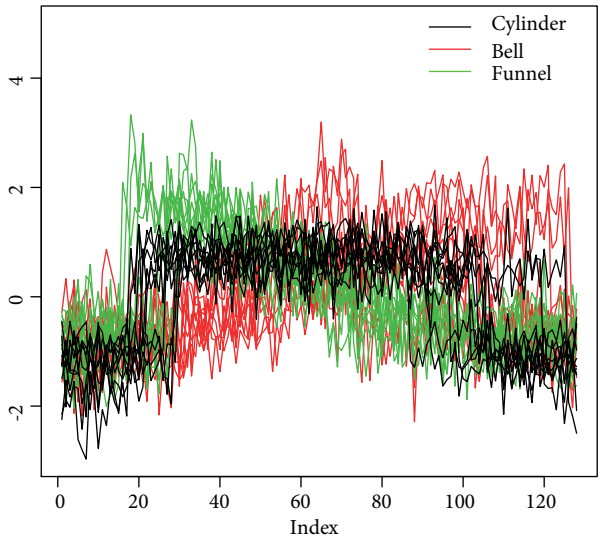

Figure 2. Cylinder-Bell-Funnel dataset [8].

We illustrate the idea of local importance using the time series of each class from the Cylinder-Bell-Funnel (CBF) dataset [8] illustrated in Figure 2. After obtaining the representation with $J_{ins} = 20$ and $R = 25$, local importance values from $RFts$ with 500 trees are visualized in Figure 3 for selected time series from each class. Symbols with the highest local importance values corresponding to the discriminative observations of the time series match with class definitions. For the Bell class, increasing intervals between time indices 60 and 80 are found to be informative. The Cylinder class is defined by the plateau between time indices 20 and 40. The decreasing intervals between time indices 30 and 50 are important for the Funnel class. Further discussion on the visualization and interpretation together with illustrative examples is provided in Section 4.2.

4. Experimental setup and results

4.1. Experimental design

We use a noisy and a nonnoisy dataset for our comparative analysis. Both datasets consist of time series from 11 different fundamental dynamic pattern classes (Figure 4) from the SD practice, for which the ISTS algorithm has individual HMMs. Each class includes 100 instances with varying lengths from 50 to 500 data points. The noisy versions of the time series are generated by adding random uniform numbers between –0.2 and +0.2 (i.e. $U\left[-0.2, +0.2\right]$) to each data point after applying z-normalization to the time series. Samples from the nonnoisy dataset can be seen in Figure 4.

We use the `TSdist` package [22] for LCSS, EDR, and ERP distance calculations and the `dtw` [23, 24]

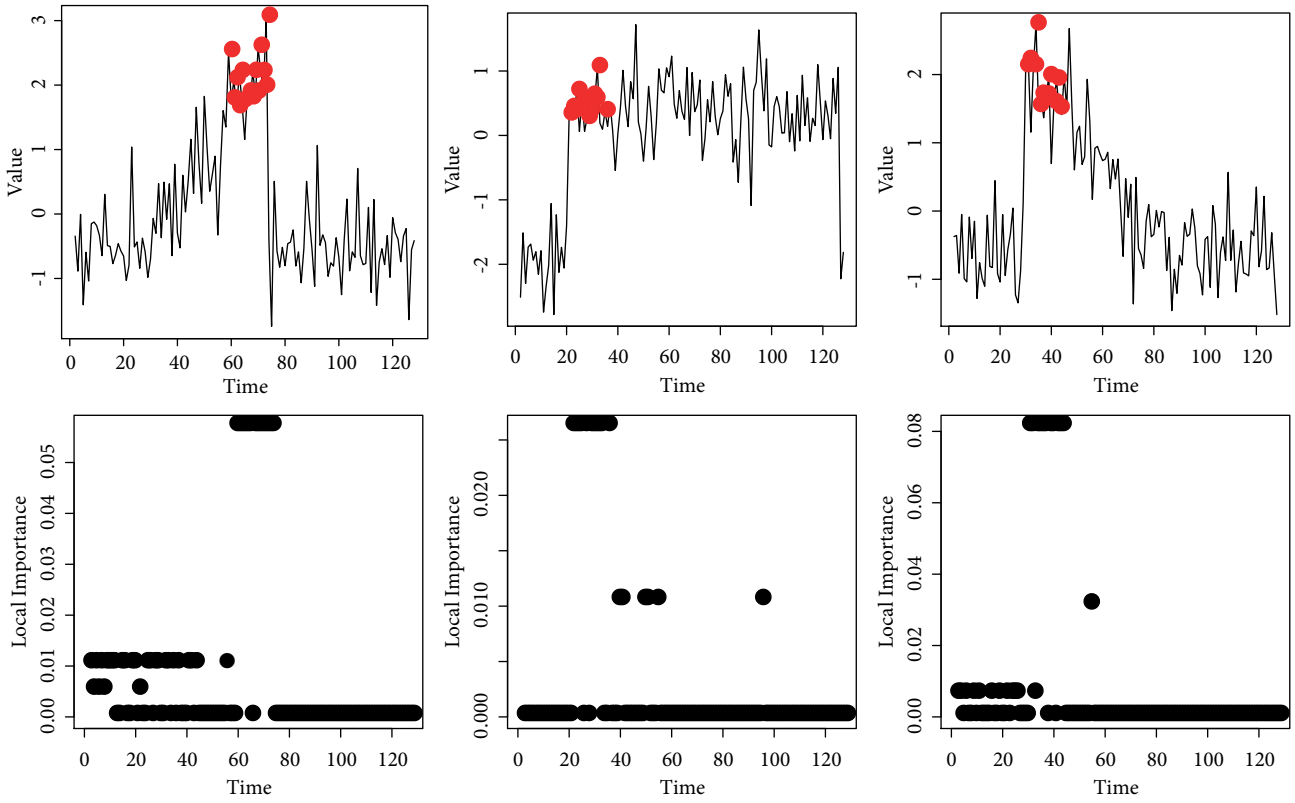

Figure 3. Three time series from CBF dataset with local importance given under each time series.

package for DTW distance calculations in R [25]. Our experiments use the Windows 8.1 system with 8 GB RAM, dual core CPU (i5-3230M 2.60 GHz). We perform 10 stratified replications of 10-fold cross-validation on the dataset to calculate misclassification errors. In k-fold cross-validation, the dataset is split into k disjoint subsets. Then the classifier is trained on the dataset obtained by taking the union of $k-1$ subsets and tested on the remaining subset. In this way, the performance of the classifier will be tested k times and the average of the misclassification errors over k folds will determine the performance of the classifier. Repeating this process several times (e.g., 10 times) and taking the average yields more reliable misclassification error estimates [26]. In addition, the stratification process ensures that the class distribution in each fold represents the class distribution of the entire dataset [27]. For LCSS and EDR, the matching threshold ε is 0.1 and 0.5, respectively. For ERP, g is set to 0 as suggested in [19]. For DTW distance calculations, we employ a symmetric dynamic programming algorithm with no slope constraint and no normalization. For all of the distance measures, no warping window constraint is used.

We compare the considered approaches on the basis of computation time (i.e. speed) and classification performances. Regarding speed, we use average time required to label a time series as the indicator. For comparing the classification performances, we rely on averages of the misclassification errors over all replications and folds and use the test procedure developed by Demšar [28]. This procedure first conducts a Friedman test [29]. If there is a significant difference among the methods, then the Nemenyi test [30] is performed. In the Nemenyi test, the performances of two classifiers are said to be different if the difference between their average ranks is greater than the critical difference (CD).

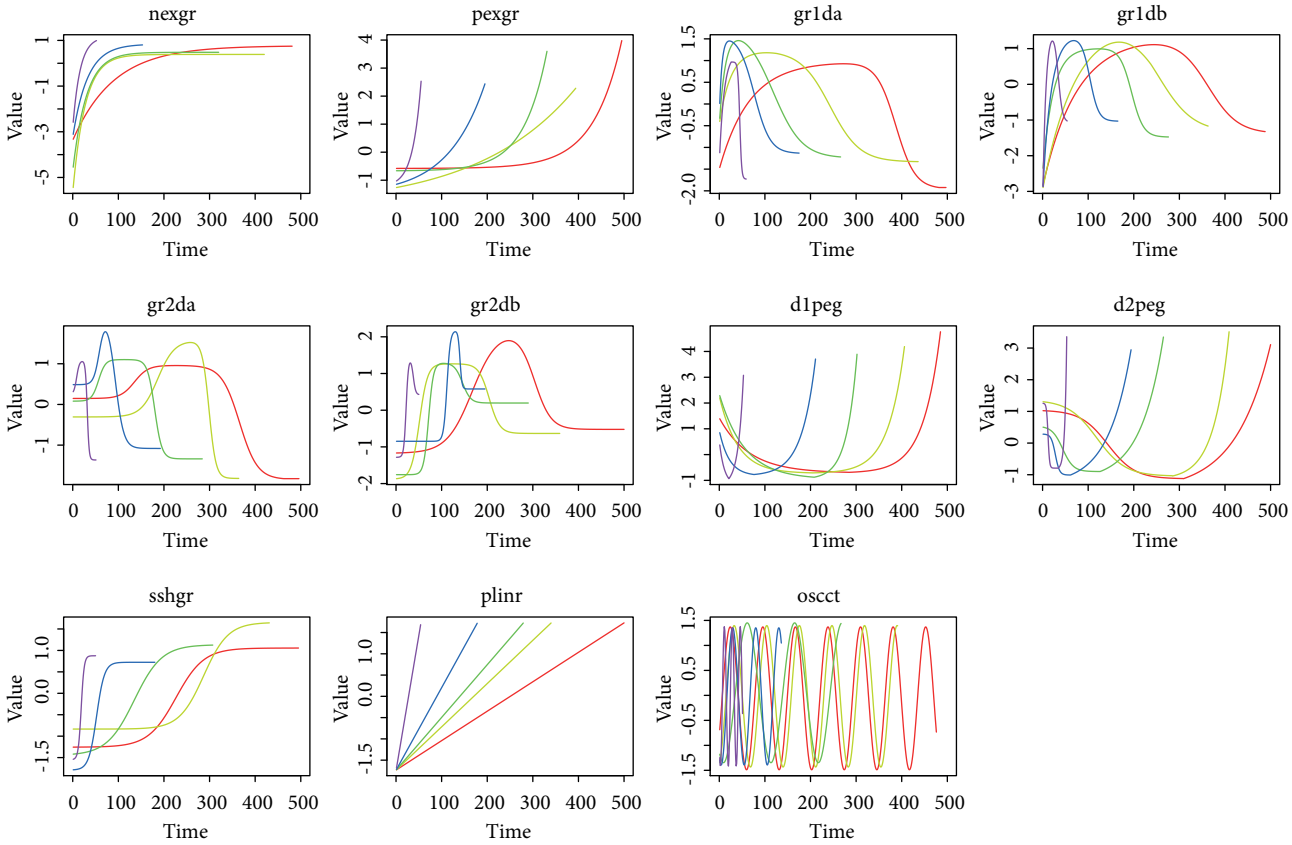

Figure 4. Example time series from the nonnoisy dataset.

4.2. Results on raw data

The experimental results for the NN classifiers (Table 2) constitute the benchmark for our comparison. For each NN classifier, we report the average required time to classify one single time series, and the error rate, which is the average of the misclassification errors over all replications and folds.

The results with the proposed approach and ISTS are given in Table 3. For the proposed approach, we report the average training and test times over all replications and folds. The test time is the time required to classify one single test instance. We are unable to train the ISTS algorithm with our dataset. Therefore, we only report the required time to classify one single test instance.

The DTW-based NN classifier performs best in terms of error rates on both datasets, and its run time is similar to other NN classifiers. The performance of ISTS on the nonnoisy dataset is comparable to some of the NN classifiers (e.g., LCSS, EDR), but it lags behind the DTW one. On the noisy dataset, the performance of ISTS is poor (i.e. error rate = 0.7518), which makes it impractical to use on noisy time series without prior filtering/denoising operations. Considering the computation time, ISTS seems to have a slight advantage over the NN classifiers.

When we compare the proposed approach against both the best NN classifier (i.e. DTW) and ISTS, SMTS clearly outperforms both on the nonnoisy dataset. On the noisy dataset, SMTS is on par with DTW. Apart from the superiority in error rates, the proposed approach also has a clear advantage in classification speed.

Table 2. Error rates and runtimes of 1-NN classifier with different distance measures.

		No interpolation		Linear interpolation	
Distance	Dataset	Error	Runtime	Error	Runtime
LCSS ($\varepsilon = 0.1$)	Noisy	0.1140	13.6065	0.0866	4.4546
	Nonnoisy	0.1648	13.9162	0.0727	4.4394
LCSS ($\varepsilon = 0.5$)	Noisy	0.1516	13.7450	0.0506	4.3846
	Nonnoisy	0.2001	13.5328	0.0600	4.4087
EDR ($\varepsilon = 0.1$)	Noisy	0.1114	13.8286	0.0748	4.5216
	Nonnoisy	0.1500	13.9908	0.0689	4.4784
EDR ($\varepsilon = 0.5$)	Noisy	0.1584	14.2040	0.0540	4.4676
	Nonnoisy	0.2153	14.1784	0.0578	4.4706
ERP ($g = 0$)	Noisy	0.1394	13.9658	0.0608	4.4692
	Nonnoisy	0.1505	14.2031	0.0561	4.4037
DTW	Noisy	0.0866	13.2865	0.0731	4.4965
	Nonnoisy	0.1001	12.8535	0.0821	4.4520

Table 3. Error rates, training, and test times of SMTS and ISTS.

		No interpolation			Linear interpolation		
Method	Dataset	Error	Training time	Test time	Error	Training time	Test time
SMTS	Noisy	0.0895	62.5382	0.0022	0.0432	28.9982	0.0010
	Nonnoisy	0.0412	52.9031	0.0022	0.0227	28.1526	0.0010
ISTS	Noisy	0.7518	-	0.0964	0.8182	-	0.0964
	Nonnoisy	0.2109	-	0.0965	0.2155	-	0.0939

When we compare the classification performance of the considered approaches as described in the previous section, we obtain the comparative plots given in Figure 5 and Figure 6 for nonnoisy and noisy datasets, respectively. Methods are sorted with respect to their average ranks and there is a line segment between two methods if the difference between them is less than the critical difference (CD). The proposed approach (SMTS) significantly outperforms all of the methods at 0.05 and 0.1 level in nonnoisy data, but there is no significant difference between DTW and SMTS at 0.05 and 0.1 level in the presence of noisy data. In Figure 6, the critical differences (CDs) are 0.963 and 1.050 for 0.1 and 0.05 levels, and in both cases DTW is connected to SMTS. This means that the difference between the average ranks of DTW and SMTS is less than the corresponding CDs. Considering the two main outcomes of interest (speed and precision), we conclude that the proposed approach performs superior not only to ISTS, but also to NN classifiers on nonnoisy datasets.

Apart from these two main criteria, this study aims to propose an interpretable classifier. Neither ISTS nor NN classifiers provide any information about the reasons behind individual label assignments. By extending SMTS with the procedure suggested in [21], it is possible to report time series segments that are most influential in the class assignment. For example, Figure 7 illustrates sample time series with characteristic segments marked with red dots. We observe that SMTS can capture original class definitions by appropriately determining the behavioral characteristics of classes. This provides insights especially when one deals with time series data with different complex behavioral characteristics, which may also be generated by SD models.

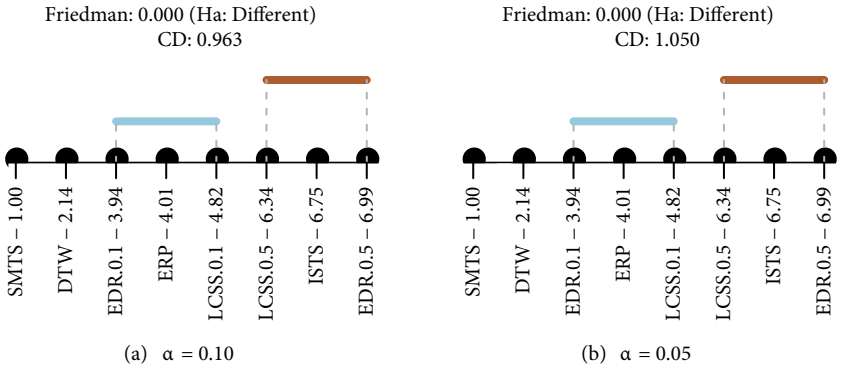

Figure 5. Average ranks of classifiers on the nonnoisy dataset (no interpolation).

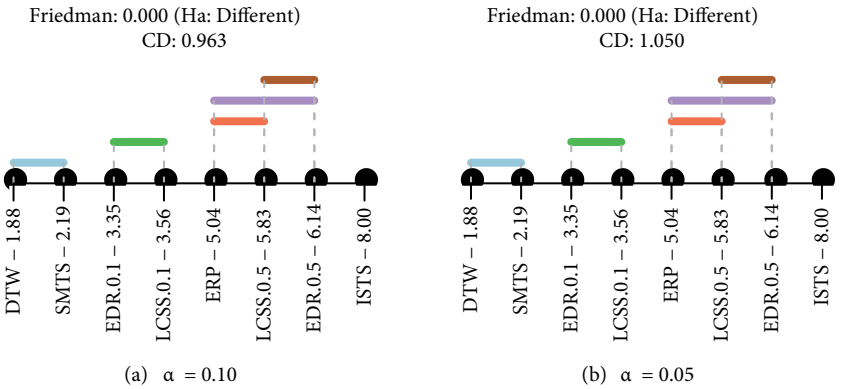

Figure 6. Average ranks of classifiers on the noisy dataset (no interpolation).

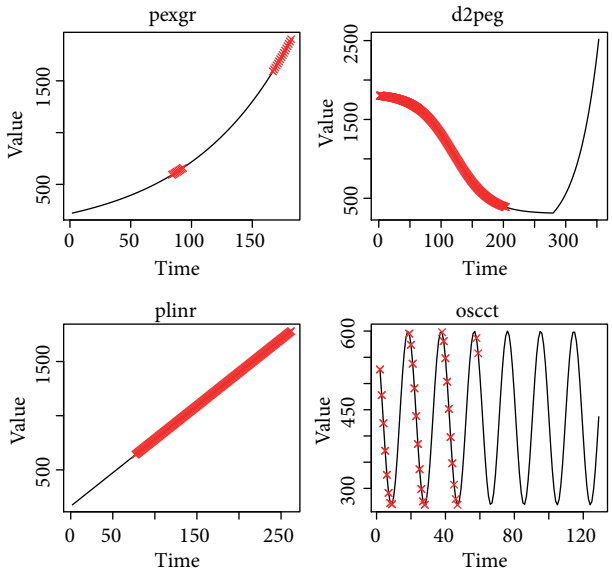

Figure 7. Sample time series from four different classes with local importance information.

4.3. Results on interpolated data

Although elastic similarity measures are proposed as an effective way of calculating the distance between time series of different lengths, the elastic nature of the measure also creates certain disadvantages. For example, consider the three series given in Figure 8. A1 and A2 are from the same behavior class, with a small difference between their plateau levels. B1 belongs to another class, but its y-range (y_{min}, y_{max}) covers the y-range of A1. In the case where the lengths of A2 and B1 are relatively shorter than that of A1, DTW tends to measure A1 to be closer to B1. In other words, the shape-wise misfit of A1 and B1 is underpenalized due to the elastic nature of the measure, and the scale-wise difference of A1 and A2 is relatively overpenalized. This eventually leads to a misclassification. Triggered by this observation, we introduce a preprocessing stage in order to equate the lengths of the time series through interpolation. Since ISTS already incorporates such a preprocessing to transform the length of the time series to 120 by linear interpolation, we employ the same for all classifiers considered in this study.

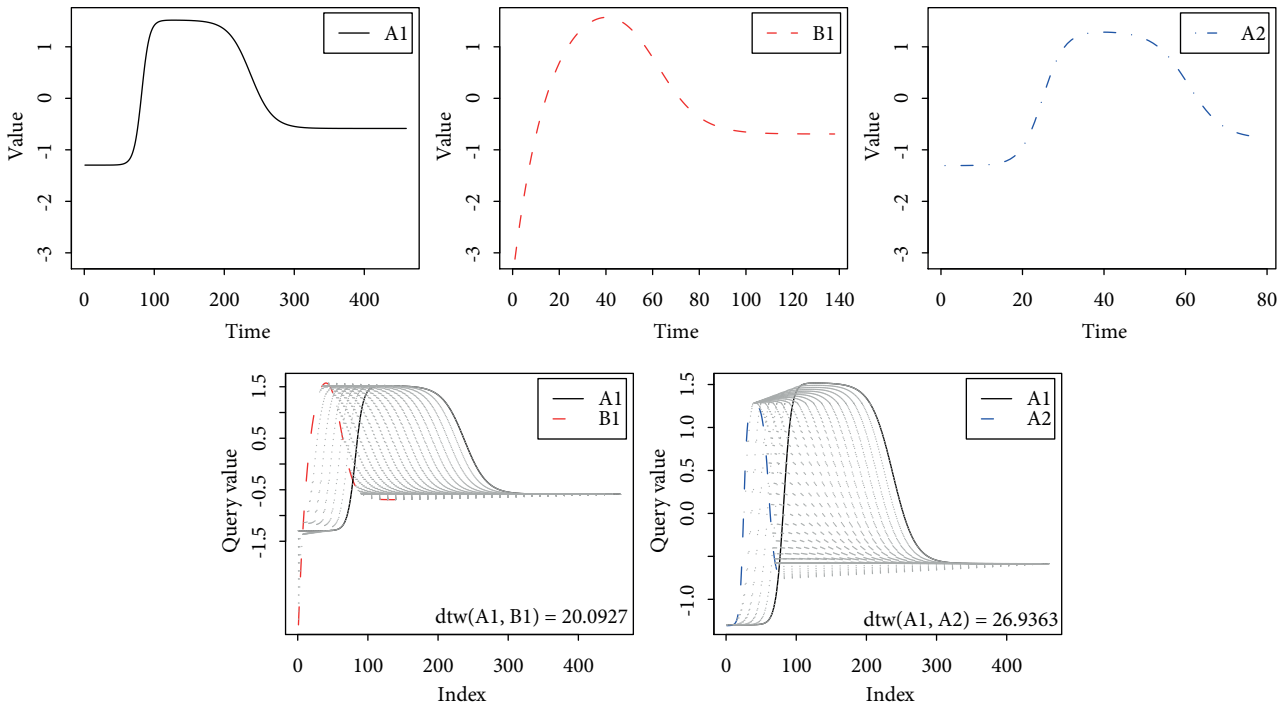

Figure 8. An example case where DTW distance calculation yields a misclassification.

Results show that equating the lengths of the time series by interpolation yields similar ranks of the methods. In that respect, our main observation regarding the superiority of the proposed approach still holds. However, on average, we observe a 68% decrease in runtimes and a 50% increase in accuracy except for ISTS. Ratanamahatana and Keogh [31] also reports that, in the case of 1-NN with DTW, there is no statistically significant difference between working on the original dataset and equating the lengths of the time series by linear interpolation in terms of classification accuracy. Even if the increase in accuracy values might not be significant, interpolation reduces the required time for classification since most of the time series become shorter than their original lengths, yielding faster distance calculations. Results show that SMTS significantly outperforms all the methods in the presence of a nonnoisy dataset having time series of the same lengths (Figure 9). In a noisy

dataset having time series of the same lengths, SMTS still performs best, but not significantly better than 1-NN with LCSS ($\varepsilon = 0.5$) (Figure 10).

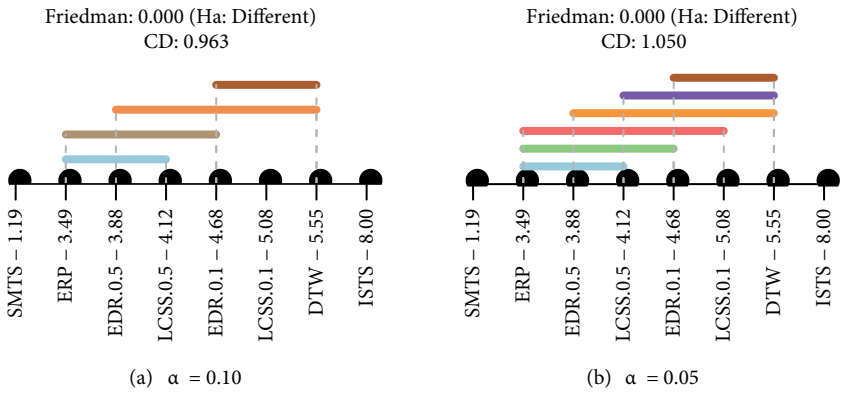

Figure 9. Average ranks of classifiers on the nonnoisy dataset (linear interpolation).

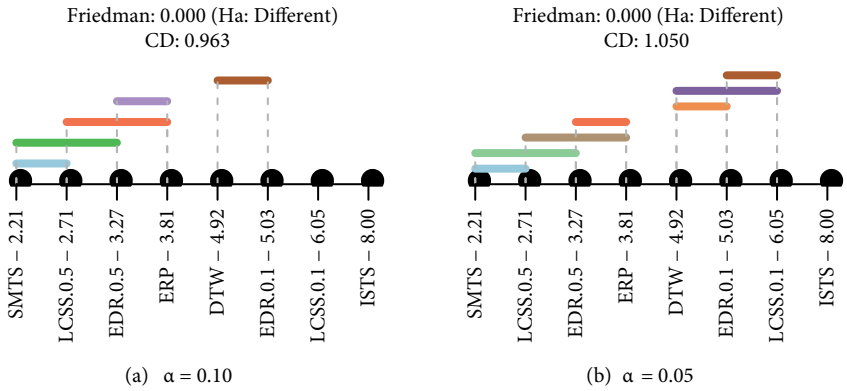

Figure 10. Average ranks of classifiers on the noisy dataset (linear interpolation).

5. Conclusion

During a typical SD modeling cycle, an analyst faces the task of visually evaluating a large set of time series output in terms of the dynamic pattern features. As it has been shown in former studies, a field-specific time series classification algorithm may prove very useful. In that respect, we propose a novel field-specific classifier for the SD studies.

The performance of the proposed approach (SMTS with local importance) is compared against both established field-specific (ISTS) and state-of-the-art NN classifiers that perform well on time series classification in general. Experimental results show that SMTS performs significantly better than all of the considered alternatives on nonnoisy test datasets. For noisy data, the 1-NN classifier with DTW and SMTS performs best with no significant difference between them. Although it is claimed that ISTS can handle noisy data, its performance is the worst among other distance measures and methods. Based on the experimental results, SMTS stands not just as a new approach better than the currently used ISTS, but also as an approach that outperforms state-of-the-art classifiers in the time series classification domain. Moreover, the performance of the proposed approach on noisy time series is on par with NN classifiers, and the former is much faster than

the latter. Additionally, SMTS also has an advantage over NN classifiers regarding the training data. As NN classifiers need to store all the training data in order to classify a new test instance, they are considered as "lazy learners". However, as a tree-based approach, SMTS is an "eager learner", which does not require to store the training data once trained.

Considering the interpretability aspect, the proposed approach performs successfully in informing the user about the important segments of a time series with respect to its class assignment. In that respect, SMTS successfully utilizes local variable importance in conjunction with a feature-based classifier. This is a methodological contribution of our work, which goes beyond the SD field and relates to the time series data mining domain.

Acknowledgment

This research was supported by the Boğaziçi University Research Fund (Grant No: 12560-17A03D1).

References

[1] Chaovalitwongse WA, Fan YJ, Sachdeo RC. On the time series k-nearest neighbor classification of abnormal brain activity. IEEE T Syst Man Cy A 2007; 37: 1005–1016.

[2] Zeng Z, Yan H. Supervised classification of share price trends. Inform Sciences 2008; 178: 3943–3956.

[3] Vergara A, Vembu S, Ayhan T, Ryan MA, Homer ML, Huerta R. Chemical gas sensor drift compensation using classifier ensembles. Sensor Actuat B-Chem 2012; 166–167: 320–329.

[4] Fu T. A review on time series data mining. Eng Appl Artif Intel 2011; 24: 164–181.

[5] Jeong YS, Jeong MK, Omitaomu OA. Weighted dynamic time warping for time series classification. Pattern Recogn 2011; 44: 2231–2240.

[6] Keogh E, Kasetty S. On the need for time series data mining benchmarks: a survey and empirical demonstration. Data Min Knowl Disc 2003; 7: 349–371.

[7] Wang X, Mueen A, Ding H, Trajcevski G, Scheuermann P, Keogh E. Experimental comparison of representation methods and distance measures for time series data. Data Min Knowl Disc 2013; 26: 275–309.

[8] Ye L, Keogh E. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. Data Min Knowl Disc 2011; 22: 149–182.

[9] Khadra L, Al-Fahoum AS, Binajjaj S. A quantitative analysis approach for cardiac arrhythmia classification using higher order spectral techniques. IEEE T Bio-Med Eng 2005; 52: 1840-1845.

[10] Kakizawa Y, Shumway RH, Taniguchi M. Discrimination and clustering for multivariate time series. J Am Stat Assoc 1998; 93: 328-340.

[11] Sterman JD. Business Dynamics: Systems Thinking and Modeling for a Complex World. Boston, MA, USA: Irwin McGraw-Hill, 2000.

[12] Forrester JW. Principles of Systems. Cambridge, MA, USA: Wright-Allen Press, 1968.

[13] Sücüllü C, Yücel G. Behavior Analysis and Testing Software (BATS). In: Proceedings of the 32nd International System Dynamics Conference; 20–24 July 2014; Delft, the Netherlands.

[14] Yücel G, Barlas Y. Automated parameter specification in dynamic feedback models based on behavior pattern features. Syst Dynam Rev 2011; 27: 195–215.

[15] Bog S, Barlas Y. Automated dynamic pattern testing, parameter calibration and policy improvement. In: Proceedings of the 23rd International System Dynamics Conference; 17–21 July 2005; Boston, MA, USA.

[16] Barlas Y, Kanar K. Structure-oriented behavior tests in model validation. In: Proceedings of the 18th International System Dynamics Conference; 6–10 August 2000; Bergen, Norway.

[17] Soylu S. Generic dynamic patterns: Testing by empirical evidence. MSc, Boğaziçi University, İstanbul, Turkey, 2006.

[18] Baydogan MG, Runger G. Learning a symbolic representation for multivariate time series classification. Data Min Knowl Disc 2015; 29: 400–422.

[19] Chen L, Ng R. On the marriage of ℓ_p-norms and edit distance. In: Proceedings of the Thirtieth International Conference on Very Large Databases – Volume 30; 31 August–3 September 2004; Toronto, Canada. pp. 792–803.

[20] Breiman L, Friedman J, Stone CJ, Olshen RA. Classification and Regression Trees. Wadsworth, NY, USA: CRC Press, 1984.

[21] Breiman L. Random forests. Mach Learn 2001; 45: 5–32.

[22] Mori U, Mendiburu A, Lozano JA. Distance measures for time series in R: the TSdist package. R J 2016; 8: 451–459.

[23] Tormene P, Giorgino T, Quaglini S, Stefanelli M. Matching incomplete time series with dynamic time warping: An algorithm and an application to post-stroke rehabilitation. Artif Intell Med 2009; 45: 11–34.

[24] Giorgino T. Computing and visualizing dynamic time warping alignments in R: the dtw package. J Stat Softw 2009; 31: 1-24.

[25] R Core Team. R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing, 2016.

[26] Alpaydin E. Introduction to Machine Learning. Cambridge, MA, USA: MIT Press, 2014.

[27] Delen D, Walker G, Kadam A. Predicting breast cancer survivability: a comparison of three data mining methods. Artif Intell Med 2005; 34: 113-127.

[28] Demšar J. Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 2006; 7: 1–30.

[29] Friedman M. A comparison of alternative tests of significance for the problem of m rankings. Ann Math Stat 1940; 11: 86–92.

[30] Nemenyi P. Distribution-free multiple comparisons. PhD, Princeton University, NJ, USA, 1963.

[31] Ratanamahatana CA, Keogh E. Three myths about dynamic time warping data mining. In: Proceedings of the 2005 SIAM International Conference on Data Mining; 21–23 April 2005; Newport Beach, CA, USA. pp. 506–510.