Knowl Inf Syst (2018) 55:695–718
https://doi.org/10.1007/s10115-017-1098-1

REGULAR PAPER

# Time series classification with feature covariance matrices

**Hamza Ergezer[1,2] · Kemal Leblebicioğlu[1]**

Received: 18 January 2017 / Revised: 18 April 2017 / Accepted: 1 September 2017 /
Published online: 9 September 2017


**Abstract** In this work, a novel approach utilizing feature covariance matrices is proposed for time series classification. In order to adapt the feature covariance matrices into time series classification problem, a feature vector is defined for each point in a time series. The feature vector comprises local and global information such as value, derivative, rank, deviation from the mean, the time index of the point and cumulative sum up to the point. Extracted feature vectors for the time instances are concatenated to construct feature matrices for the overlapping subsequences. Covariances of the feature matrices are used to describe the subsequences. Our main purpose in this work is to introduce and evaluate the feature covariance representation for time series classification. Therefore, in classification stage, firstly, 1-NN classifier is utilized. After showing the effectiveness of the representation with 1-NN classifier, the experiments are repeated with SVM classifier. The other novelty in this work is that a novel distance measure is introduced for time series by feature covariance matrix representation. Conducted experiments on UCR time series datasets show that the proposed method mostly outperforms the well-known methods such as DTW, shapelet transform and other state-of-the-art techniques.

**Keywords** Time series classification · Time series representation · Feature covariance matrices

## 1 Introduction

Time series analysis has received great interest over the past decades from several disciplines including biology, medicine, economics, etc. In tandem with the increasing availability of

✉ Hamza Ergezer
hergezer@aselsan.com.tr

1 Department of Electrical and Electronics Engineering, Middle East Technical University (METU), Ankara, Turkey

2 MGEO Division, Department of EO System Design, Aselsan Inc, Ankara, Turkey

digital data, it is highly possible that this problem will gain much more attention. Time series classification is one of the most important problems in time series analysis [17,40]. There are two main issues in time series classification as in other time series analysis problems: representation of time series and similarity between time series [17,40].

Time series classification can be treated as a supervised learning problem. From this perspective, aforementioned common issues related to time series analysis correspond to a feature extraction step and an appropriate classification approach. The feature extraction step is needed to extract the discriminative part of the time series as in problems in computer vision [33] and speech recognition [5]. The time series is then represented with the extracted features for further tasks such as indexing, clustering and classification. In addition, by representing the data in a feature space, dimensionality reduction can also be achieved for most of the cases.

In the last two decades, several feature extraction methods [9,13,28,32,33,46] have been proposed for detection and classification problems in images and videos. Although some visual features are utilized for the problems in time series domain [36,42,44], feature extraction methods have not become popular yet when time series classification is considered. There are recent studies [7,50] applying well-known features in computer vision to the time series classification problem; however, some important techniques remain untouched for the time series domain. Covariance descriptor, which is the utilized technique in this paper, is one of them that can be efficiently applied to time series classification problem.

Previously, covariance descriptor has been used to describe regions for images [15,37,45,46] and video blocks in capturing different types of actions [22]. Covariance descriptor is exploited as a high-level feature in all these applications. It captures the pairwise correlations of the basic image or video features. Similarly, for representing the time series, it captures the pairwise correlations between the pointwise features. Covariance descriptor presents a compact representation due to its symmetry. It provides a low-dimensional and fixed-size covariance matrix which is independent of the length of time series. Also, it is robust to noisy inputs since it includes an inherent averaging in covariance computation.

In this work, we propose a novel approach for time series classification problem. In our previous work [16], feature covariance matrices were used to represent trajectories which are basically 2D time series. It was shown that representing trajectories via feature covariance matrices outperforms the state-of-the-art methods for anomaly detection and activity perception problems. We utilize the same representation for time series classification problem by adapting it to subsequences and defining novel pointwise features. First, feature covariance matrices are determined for overlapping subsequences of both test and training samples. Overlapping subsequences are selected in order to cover all discriminative portions of the time series. While building covariance descriptor, a feature vector is defined for each point in the subsequence. Feature vectors are then combined to determine the feature matrix. Covariance of the feature matrix is utilized to represent the subsequence. In the classification stage, two well-known classifiers are utilized. For 1-NN classifier, the distance between training samples and a test sample is determined by using Log-Euclidean distance metric. For SVM classifier, the upper (or lower) triangular part of the covariance matrices of the subsequences are concatenated. Comparative experiments using 43 different datasets are conducted. We achieve mostly outperforming results compared to well-known methods such as DTW, shapelet transform and two state-of-the-art methods [10,50]. The proposed approach is also very efficient in computation time compared to other methods. The computation time efficiency is due to the fact that overall method is simply based on a feature covariance representation.

The main contribution of the paper is our novel time series classification approach which is based covariance descriptor. To the best of our knowledge, this is the first method where each point of the time series is used for representation and feature extraction. Another important contribution is that a novel distance measure is proposed for time series. The distance obtained by covariance descriptor can be applied to time series of different lengths. Lastly, our results are mostly outperforming the well-known and state-of-the-art methods.

The remainder of this paper is organized as follows. A summary of previous works on time series classification is provided in the next section. Section 3 presents the proposed representation based on covariance descriptor and how it is applied to time series classification. How we utilize two classifiers is summarized in Sect. 4. Conducted experiments are given in Sect. 5. Conclusions and possible future work are provided in Sect. 6.

## 2 Related work

There are several previous works that address the time series classification problem. A review of representations and distance measures for time series is provided in [47]. In our perspective, time series classification is a supervised learning problem. As other supervised learning problems, time series classification is conceived as a two-fold problem in this work: representation and classification. However, there are many studies handling time series classification problem from different perspectives. Studies are divided into three categories according to how they approach the problem. The first category includes the distance-based approaches [24,27,38]. In the second category, approaches seek a better representation for time series [7,10,23,30,35,48,50]. Approaches in the last category mostly focus on the classification part of the problem and apply various classification methods like random forests [14], ensembling [6,21,31].

The most direct solution for time series classification problem is measuring the Euclidean distance between time instances and applying nearest neighbor classifier. However, this solution is susceptible to time distortions such as shifting, stretching and contracting. Dynamic time warping (DTW) [38] is used to mitigate these distortions. DTW measures the similarity between time series by searching the optimum map over the points in a time series. In [27], authors propose a modification of original DTW methodology to handle the singularities by utilizing first-order differences. On the other hand, a weighted version of DTW is used to give different weights to warping distances in [24]. In a very recent work [26], DTW distance is utilized as a feature for time series classification.

Representation-based approaches are divided into three subcategories: shapelet transform, bag-of-words and feature-based approaches. Shapelet is a discriminative subsequence of time series. The main idea behind the approach is to find a subsequence which has a smaller distance to one class of time series compared to other ones. The first shapelet-based approach was proposed in [48] to reduce computation complexity of preceding methods and get more insights about data. In this pioneering work of Ye and Keogh [48], the discovery of the shapelets is embedded into a decision tree classifier. Mueen et al. [35] propose logical shapelets to reduce the computation time of shapelet discovery. They discover the conjunction or disjunction of shapelets instead of discovering a single shapelet. However, the method is still based on an enumerative search. The approach proposed in [23] finds top k shapelets in a single run and utilizes the transformed data onto these top k shapelets while classifying the time series. In a more recent study [6], the data after shapelet transformation are one of the blocks fed into an ensemble classifier.

Another representation-based approach for time series classification is based on bag-of-words or bag-of-features models. Bag-of-words models are used in several tasks including the examples of image retrieval [49], object detection [34] and music classification [20]. An approach called symbolic aggregate approximation (SAX) [30] that attacks several time series data mining problems follows a similar path for time series classification. In this approach, words are generated using the symbols for fixed-size windows and time series is represented by these words. Authors use different distance measures for different time series data mining problems. In [10], some features such as mean, deviation and slope of the fitted regression line are put in a bag to represent a randomly selected subsequence. After extracting features for subsequences, random forest classifier is used to generate the codebooks for subsequences and classify the time series.

Feature-based approaches generally consist of two steps: extraction of local or global features and a classifier. The classifier is trained using the extracted features from a training set. Some recent works adapt the proven descriptors in computer vision to the time series [7,43,50]. Histogram of gradient (HOG) descriptor [13] is applied to 1D time series and is utilized in time series classification in [50]. In [50], there are also descriptors which are used to represent the subsequences. The fused descriptor is the input of a Fisher vector encoding followed by a linear kernel SVM. Scale-invariant feature transform (SIFT) [32] is another popular descriptor used in several computer vision problems. It is applied to time series classification with a bag-of-features approach in [7]. A more recent work [26] uses the DTW distance measure as a feature. A time series is represented in terms of its DTW distances from each of the training examples. Our approach can also be categorized as a feature-based approach. It learns covariance features of subsequences from a training set and inputs these covariance matrices to a classifier to decide the class of test samples.

For the classification part of the problems, most of the well-known classifiers such as SVM [50], random forests [25], NN [39] are used for time series classification problem. In [3], the authors present a benchmark of approaches for time series forecasting. Since regression and classification are very similar problems, the approaches and their results mentioned in [3] provide a vision for time series classification problem. On the other side, there are some recent works [6,21,31] that utilize the ensembling strategy. Thirty-five classifiers constructed in time, frequency and shapelet transformations are combined in [6]. Classifiers are combined according to their training set cross-validation accuracy. In [21], authors exploit the training set to achieve the most informative features from thousands of interpretable features. The most informative features for each class in a time series are found using greedy feature selection with a linear classifier. Another ensemble method proposed in [31] combines the basic distance measures including two aforementioned variants of DTW and edit distance-based measures. As expected, all three methods that exploit the ensemble strategy achieve significantly better results compared to other methods.

## 3 Representation of time series by feature covariance matrices

In our previous work [16], feature covariance matrices were used to represent trajectories in image sequences, which are basically 2D time series. It was shown that representing trajectories via feature covariance matrices provided state-of-the-art results for anomaly detection and activity perception problems. Similarly, in this work, feature covariance matrices are used to represent 1D time series. A set of analogies is built up between computer vision and time series domains, such that the concept of time series-subsequence-point triplet of a time series signal is inherited from the concept of an image-region-pixel triplet in computer vision. More clearly, covariance descriptor is used to represent an image region [45] or a

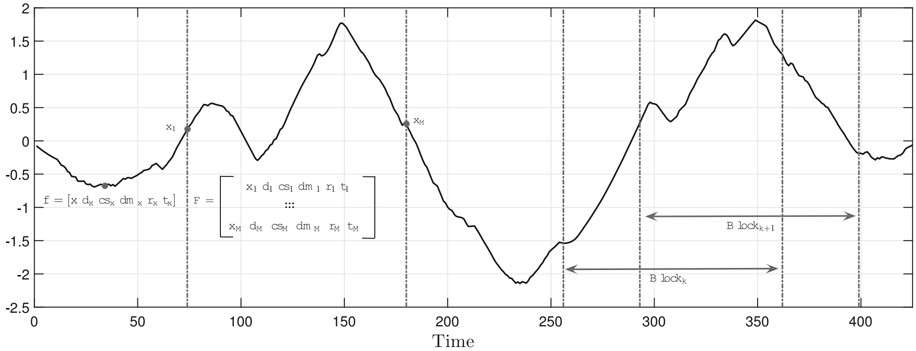


**Fig. 1** Representation of time series with feature covariance matrices is depicted in *yoga* dataset of UCR repository. First, a feature vector is defined for each point in the time series. Feature vectors defined for each point of time series form feature matrices for overlapping subsequences. Each subsequence is represented with the covariance of these feature matrices

video block [22] in computer vision problems. In a similar fashion, in this study, it is utilized to represent the subsequences of a time series as shown in Fig. 1. Covariance descriptor is built by determining the covariance of a feature matrix comprising feature vectors defined for each point in the subsequence. Analogous to the selection of regions in computer vision problems [45,46], overlapping subsequences are selected so that discriminative portions of the data are not missed. A parallel analogy is pursued in the selection of pointwise features. Pointwise features that are used to represent time series are analogous to pixel features such as pixel value, pixel coordinates, optical flow, etc.

Covariance descriptor is based on the covariance of feature matrices. Covariance matrices are semi-positive definite matrices. As other semi-positive definite matrices, covariance matrices lie on Riemannian manifold space. Therefore, before getting into the details of the proposed approach, we provide a brief summary of Riemannian manifolds. Riemannian manifolds are located in the third layer of a manifold hierarchy. Topological manifolds are locally Euclidean spaces which are located in the first layer of the hierarchy. In the second layer, differentiable (or smooth) manifolds are located. Differentiable manifolds are topological manifolds, for which some calculus operations such as the derivative can be defined. Riemannian manifolds are differentiable manifolds, for which distance metrics and angles can be defined.

The critical point is the calculation of distances between covariance matrices while using them for a classification problem. Before explaining the distance measure, a more illustrative definition of the Riemannian manifold can be given as follows: A Riemannian manifold is a differentiable manifold equipped with the inner product on tangent space at each point. The distance measure is based on exponential and logarithmic maps between two points on Riemannian manifolds. It should be noted that these definitions are introductory basics about manifold geometry. More information on Riemannian manifolds can be found in [29] and in the second chapter of [18].

Time series can be considered as a sequential concatenation of several points. Each point has its own features such as value, slope, distance to mean, time index, etc. These can be used for the classification problem since they carry important information about the series. Moreover, the change of these pointwise features with respect to each other carries another important information about time series. The main idea behind the use of covariance matrix as a descriptor is that it captures these pairwise correlations between the pointwise features.

Time series can be defined as a sequential combination of $M$ points or formally as a vector of length $M$ ($[x_1, \ldots, x_M]$). Feature candidates can be merged in a feature vector for a point in time series. Let the number of features, $\{s_i\}$, defined for a point be K. The feature vector for $N$th point of the subsequence can be shown as

$$f_N = [s_{N1}, \ldots, s_{NK}] \tag{1}$$

When feature vectors are combined for all points, we end up with a feature matrix $F$,

$$F = \begin{bmatrix} s_{11} & \ldots & s_{1K} \\ & \cdot & \\ & \cdot & \\ & \cdot & \\ s_{M1} & \ldots & s_{MK} \end{bmatrix} \tag{2}$$

The covariance of the feature matrix is calculated as

$$C = \frac{1}{M-1} \sum_{i=1}^{M-1} (F_i - \mu)(F_i - \mu)^T \tag{3}$$

where $\mu$ is the mean vector of feature vectors $\{f_1, \ldots, f_M\}$ .

This general representation is adapted for time series classification problem by dividing the time series into $L$ overlapping subsequences each of them has a length $M$. A point in a subsequence is defined by its features, namely its value, derivative, cumulative sum, the difference between mean, rank and time index. The features carry both local and global information about the point. Value, derivative and time index features are local features since they depend only on the point itself and in certain cases on its adjacent point. On the other hand, cumulative sum, the difference between mean and rank depends on almost all points in the subsequence, and hence, they can be called as global features. The feature vector with these pointwise features is formed as

$$f = [x_T \; d_T \; \mathrm{cs}_T \; \mathrm{dm}_T \; r_T \; T] \tag{4}$$

where $x_T$ is the value, $d_T$ is the derivative at a specific point at time $T$

$$d_T = x_T - x_{T-1} \tag{5}$$

The third feature in the feature vector is cumulative sum up to a point and is given by Eq. (6) at point $T$

$$\mathrm{cs}_T = \sum_{t=1}^{T} x(t) \tag{6}$$

The next pointwise feature is the difference between the mean of the subsequence and value at point $T$. This pointwise feature indicates the distinctiveness of the point in the subsequence, and it is calculated as

$$\mathrm{dm}_T = x(T) - \frac{1}{M} \sum_{t=1}^{M} x(t) \tag{7}$$

where $M$ is the length of subsequence. The feature of rank, $r_T$, is the $T$th biggest value of the subsequence. We have also inserted time directly to the feature vector to show us how other features change in time. It should be noted that the time feature is normalized within the subsequence. For each subsequence, time index starts from 2, due to the calculation of the derivative, and goes up to $M$. Throughout the experiments, the effects of several other features including the second derivative, the difference between the maximum, etc., are evaluated as well. However, the best classification performance is obtained with the proposed feature set. Besides, a detailed analysis of the effects of selected pointwise features is given in Sect. 5. After determination of all pointwise features for a whole subsequence, feature matrix is defined by

$$F = \begin{bmatrix} x_2 & d_2 & \mathrm{cs}_2 & \mathrm{dm}_2 & r_2 & 2 \\ & & & \cdot & & \\ & & & \cdot & & \\ & & & \cdot & & \\ x_M & d_M & \mathrm{cs}_M & \mathrm{dm}_M & r_M & M \end{bmatrix} \tag{8}$$

Covariance of the feature matrix defined in Eq. 8 is utilized to represent each subsequence of time series. By doing so, time series are carried onto the Riemannian manifold space. Each subsequence of the time series corresponds to a point in the manifold. The covariance of the feature matrix is calculated as in Eq. 3. At this point, a small multiple of the identity matrix is added to covariance matrices for all subsequences in test and training sets. This regularization is performed to ensure the positive definiteness of the covariance matrix. Positive definiteness is crucial for the distance metric which involves a logarithm operation.

The covariance descriptor brings some advantages for time series classification problem. The covariance matrix enables us to combine multiple feature vectors. The dimension of the covariance matrix depends only on the dimension of the feature vector, $f$, given in Eq. 4. The covariance matrix $C$ is a $d \times d$ matrix when $f$ is $d$-dimensional. Also, due to its symmetry, $C$ has only $(d^2 + d)/2$ independent values. As stated before, dimension reduction is one of the main goals of feature extraction. When compared to the dimension of time series or even the dimension of the subsequences, $C$ lies in a lower dimensional space. Last but not least, the novel part of the representation with covariance matrix is that it maps the time series into a fixed-size vector space, which is independent of the series' length.

## 4 Time series classification with feature covariance matrices

After representing 1D time series with feature covariance matrices, two well-known classification techniques are utilized. First, the 1-NN classifier is utilized by considering it one of the simplest classification methods. The main goal of the utilization of the 1-NN classifier is to evaluate the effectiveness of the proposed representation and isolate the performance of the overall method from the performance of the classifier. After determining the satisfactory results with the 1-NN classifier, SVM classifier is used as a more complex classification technique. In this section, we give the details about how classifiers are applied to the problem.

### 4.1 Classification with 1-NN classifier

For 1-NN classifier, the most crucial step is the calculation of the distances between the covariance matrices of training samples and a test sample. As stated before, a distance measure that approximate the geodesic distance between two points on Riemannian manifolds must be used. For this purpose, similarly to [22,37,46] that utilize covariance matrices, Euclidean distance metrics must be avoided. We use the log-Euclidean metric which was firstly proposed in [4]. Compared to other distance metrics [19] and divergence functions [11], log-Euclidean metric gives the best performance in our experiments.

Log-Euclidean distance metric is based on matrix logarithm operation. Matrix logarithm operation maps covariance matrices from a Riemannian manifold to Euclidean space. The determination of the log-Euclidean metric starts with the eigenvalue decomposition of the covariance matrix.

$$C = VDV^T \tag{9}$$

After this eigenvalue decomposition, matrix logarithm is obtained as

$$\log(C) \triangleq V\widetilde{D}V^T \tag{10}$$

where $\widetilde{D}$ is a diagonal matrix obtained from $D$ by replacing its diagonal entries by their logarithms. The distance between two covariance matrices is calculated via Frobenius norm of the distance between matrix logarithms.

$$\rho(C_{ik}, C_{jk}) = \left\| \log(C_{ik}) - \log(C_{jk}) \right\|_F \tag{11}$$

where $i$ and $j$ indicate train and test instances of $k$th subsequence of time series.

Now, we have all the distances between subsequences via their feature covariance matrices. The distance between the whole time series is determined by averaging the distances between subsequences.

$$\rho(T_i, T_j) = \frac{1}{L}\sum_{k=1}^{L}\rho(C_{ik}, C_{jk}) \tag{12}$$

where $L$ is the number of subsequences. Here, the average value of the distances between the subsequences is used as the distance between two time series. Also, there are other options like taking minimum or maximum of the distances as the final distance. However, averaging the distances hinders the domination of a single subsequence.

After obtaining the final distance between time series, 1-NN classification is applied to determine the class of a test sample. In other words, the class of a sample in the test set is assigned to the class of its nearest neighbor in the training set using the distance measure explained above.

### 4.2 Classification with SVM classifier

SVM classifier is generally the first solution that comes to mind for a classification problem. In our case, after determining the covariance matrices of the subsequences, the critical question is how we build the model. There are some recent studies [8,41] that aim kernel learning over the manifolds. However, there is a more direct way. Utilizing symmetric property of covariance matrices, lower (or upper) diagonal parts of the covariance matrices are fed into the SVM classifier as in [8]. In a more formal definition, for a time series dataset for which the number of subsequences is $L$ and the dimension of the feature vector is $d$, the dimension of the input vector fed into the SVM classifier is $L * \frac{d(d+1)}{2}$. Again, as in [8], the non-diagonal elements are multiplied with $\sqrt{2}$ to keep the equality of norms.

SVM is originally a binary classification technique. There are some approaches while utilizing SVM in a multi-class classification problem. One-versus-all (or one-against all) is one of these approaches. In this work, we utilize one-versus-all approach for time series classification. In this approach, for each class, a classifier is trained. For the $i$th classifier, let the positive examples be all the points in class $i$, and let the negative examples be all the points not in class $i$. In order to make a decision for an unseen sample $x$, we take the corresponding classifier which reports the highest confidence score

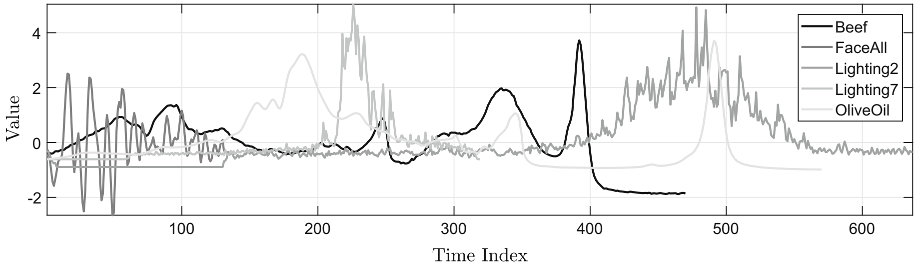


**Fig. 2** Five samples from UCR time series dataset. Diversity in length and variability for the datasets can be observed

$$y = \underset{i=1,\ldots,N}{\arg\max} f_i(x) \tag{13}$$

where $f_i(x)$ is the $i$th classifier and $N$ is the number of classifiers or classes.

## 5 Experiments

In this section, we evaluate the performance of the proposed approach by using the datasets in the UCR repository [2,12]. The UCR repository consists of a diverse set of 85 datasets which grouped into six different types. This diversity in the type of UCR datasets leads to high variability in series length and character as depicted in Fig. 2 and summarized in Table 1. Due to this variability, the number of subsequences, overlap ratio between the subsequences and the regularization constant have been obtained for each dataset by cross-validation. For this purpose, for both classifier, the original training set is split into equally sized validation and training sets. This procedure is performed 100 times by randomly selecting validation and training sets from the original training set. The optimum parameters are based on the average classification errors acquired in these simulations. All the other parameters are kept fixed for training and test sets in a dataset. Lastly, for SVM classifier, LIBSVM [1] is utilized during the experiments.

In our experiments, we have utilized 43 of these 85 datasets for which the classification results of all the compared methods have been previously published. In the following subsection, we first compare the performance of the proposed method with well-known and the state-of-the-art methods and show that the proposed method mostly outperforms the compared methods. In the second subsection, an analysis is presented with different combinations of pointwise features. The computational complexity of the proposed method is analyzed and compared with another feature-based method given in [50] in the last subsection.

### 5.1 Comparative results

The performance of the proposed method is compared to four different methods. Two well-known methods, dynamic time warping (DTW) and shapelet transform (ST), are selected for comparison similarly to state-of-the-art studies [10,50]. These two methods provide the best classification performance for some of the selected datasets in the repository. The other two methods are feature-based methods [10,50] and chosen due to their similarities with the proposed approach. Below, a brief explanation on the reported performance of each compared method is provided. We preferred not to include the Euclidean distance measure

**Table 1** Information about the datasets used in experiments

| Dataset | No. of classes | Train size | Test size | Length | Type |
|---|---|---|---|---|---|
| 50 words | 50 | 450 | 455 | 270 | Image |
| Adiac | 37 | 390 | 391 | 176 | Image |
| Beef | 5 | 30 | 30 | 470 | Spectro |
| CBF | 3 | 30 | 900 | 128 | Simulated |
| ChlorineCon | 3 | 467 | 3840 | 166 | Simulated |
| CinCECGTorso | 4 | 40 | 1380 | 1639 | ECG |
| Coffee | 2 | 28 | 28 | 286 | Spectro |
| CricketX | 12 | 390 | 390 | 300 | Motion |
| CricketY | 12 | 390 | 390 | 300 | Motion |
| CricketZ | 12 | 390 | 390 | 300 | Motion |
| DiatomSizeR | 4 | 16 | 306 | 345 | Image |
| ECG200 | 2 | 100 | 100 | 96 | ECG |
| ECGFiveDays | 2 | 23 | 861 | 136 | ECG |
| FaceAll | 14 | 560 | 1690 | 131 | Image |
| FaceFour | 4 | 24 | 88 | 350 | Image |
| FacesUCR | 14 | 200 | 2050 | 131 | Image |
| Fish | 7 | 175 | 175 | 463 | Image |
| GunPoint | 2 | 50 | 150 | 150 | Motion |
| Haptics | 5 | 155 | 308 | 1092 | Motion |
| InlineSkate | 7 | 100 | 550 | 1882 | Motion |
| ItalyPower | 2 | 67 | 1029 | 24 | Sensor |
| Lightning2 | 2 | 60 | 61 | 637 | Sensor |
| Lightning7 | 7 | 70 | 73 | 319 | Sensor |
| MALLAT | 8 | 55 | 2345 | 1024 | Simulated |
| MedicalImages | 10 | 381 | 760 | 99 | Image |
| MoteStrain | 2 | 20 | 1252 | 84 | Sensor |
| OSULeaf | 6 | 200 | 242 | 427 | Image |
| OliveOil | 4 | 30 | 30 | 570 | Spectro |
| SonyAI | 2 | 20 | 601 | 70 | Sensor |
| SonyAI-II | 2 | 27 | 953 | 65 | Sensor |
| StarLightCurves | 3 | 1000 | 8236 | 1024 | Sensor |
| SwedishLeaf | 15 | 500 | 625 | 128 | Image |
| Symbols | 6 | 25 | 995 | 398 | Image |
| Trace | 4 | 100 | 100 | 275 | Sensor |
| TwoLeadECG | 2 | 23 | 1139 | 82 | ECG |
| TwoPatterns | 4 | 1000 | 4000 | 128 | Simulated |
| SyntheticControl | 6 | 300 | 300 | 60 | Simulated |
| UWaveX | 8 | 896 | 3582 | 315 | Motion |
| UWaveY | 8 | 896 | 3582 | 315 | Motion |
| UWaveZ | 8 | 896 | 3582 | 315 | Motion |
| Wafer | 2 | 1000 | 6174 | 152 | Sensor |

**Table 1** continued

| Dataset | No. of classes | Train size | Test size | Length | Type |
|---|---|---|---|---|---|
| WordSynonyms | 25 | 267 | 638 | 270 | Image |
| Yoga | 2 | 300 | 3000 | 426 | Image |

in the results because of its low performance for all datasets compared to the state-of-the-art methods including the proposed method.

*DTW* In this work, the results of the version of DTW with warping window [39] are used. DTW competes with the state-of-the-art methods with the 1-NN classifier. The reported classification success of this combination has the best performing results for some of the datasets as can be seen in Table 2.

*Shapelet Transform (ST)* The results of shapelet transform are based on the results given in University of East Anglia website.[1] The classification result of ECG200 dataset is missing for this approach. Therefore, all comparisons with the shapelet approach are evaluated for 42 datasets only.

*TSBF* In [10], authors report their results for both uniform and random selection of features in 45 datasets. Their results show that random selection of features gives better results for most of the datasets. Therefore, we take the results with randomly selected features similarly to other methods [6,50] that compare their methods with TSBF.

*HOG1D + DTW-MDS* Provided that the approach presented in [50] is achieved by combining a set of features, the authors presented their results for each combination separately. The best results are achieved for the case where all features are used. The results of this combination for 43 datasets are selected for comparison.

In this section, we compare the performance of the proposed method with other approaches. The performance of 1-NN and SVM classifiers is reported excluding the other one. Hereinafter, we use CovNN and CovSVM for the union of the covariance representation with 1-NN and SVM classifiers, respectively. Firstly, as can be seen in Table 2, CovNN provides the best classification performance in 19 of 43 datasets. For 4 of these 19 datasets, the proposed method has identical classification performance with some of the compared methods. For the rest of the repository, the compared methods, DTW, ST, TSBF and HOG-1D + DTW-MDS, provide the best classification performance for 4, 4, 6 and 15 of the datasets, respectively (including the ties). The scatter plots are provided in Fig. 3 for pairwise evaluation of CovNN with the compared methods. CovNN individually outperforms DTW, ST, TSBF and HOG1D + DTW-MDS in 34, 33, 34 and 22 datasets, respectively. The numbers of the datasets for which other methods have better results are 8, 9, 9 and 18 for the same order of the methods. CovNN has the best and same performance as DTW and HOG-1D + DTW-MDS for 1 and 3 of the datasets, respectively.

CovSVM provides the best classification performance in 20 of 43 datasets. For 4 of these 19 datasets, the proposed method has identical classification performance with some of the compared methods. For the rest of the repository, the compared methods, DTW, ST, TSBF and HOG-1D + DTW-MDS, provide the best classification performance for 4, 4, 6 and 14 of the datasets, respectively (including the ties). The scatter plots are provided in Fig. 4 for pairwise evaluation of CovSVM with the compared methods. CovSVM individually outperforms DTW, ST, TSBF and HOG1D + DTW-MDS in 35, 34, 35 and 25 datasets, respectively. The numbers of the datasets for which other methods have better results are 8,

[1] https://www.uea.ac.uk/computing/machine-learning/shapelets/shapelet-results.

**Table 2** Error rates of the methods for a subset of UCR dataset

| Dataset | DTW | ST | TSBF | HOG1D | CovNN | CovSVM |
|---|---|---|---|---|---|---|
| 50 Words | 0.242 | 0.281 | 0.209 | 0.402 | 0.222 | **0.200** |
| Adiac | 0.391 | 0.435 | 0.245 | 0.320 | 0.217 | **0.164** |
| Beef | 0.467 | 0.167 | 0.287 | 0.367 | 0.100 | **0.067** |
| CBF | 0.004 | 0.003 | 0.009 | **0.000** | **0.000** | **0.000** |
| ChlorineCon | 0.350 | 0.300 | 0.336 | 0.307 | 0.294 | **0.255** |
| CinCECGT | 0.070 | 0.154 | 0.262 | 0.249 | 0.003 | **0.000** |
| Coffee | 0.179 | **0.000** | 0.004 | **0.000** | **0.000** | **0.000** |
| CricketX | 0.236 | 0.218 | 0.278 | **0.195** | 0.244 | 0.236 |
| CricketY | **0.197** | 0.236 | 0.259 | 0.205 | 0.210 | 0.251 |
| CricketZ | **0.180** | 0.228 | 0.263 | 0.185 | 0.239 | 0.215 |
| DiatomSizeR | 0.065 | 0.124 | 0.126 | **0.016** | 0.052 | 0.043 |
| ECG200 | 0.310 | | 0.145 | **0.060** | 0.080 | 0.070 |
| ECGFiveDays | 0.203 | **0.001** | 0.183 | 0.012 | 0.116 | 0.002 |
| FaceAll | 0.192 | 0.263 | 0.234 | **0.082** | 0.194 | 0.199 |
| FaceFour | 0.114 | 0.057 | 0.051 | 0.034 | 0.023 | **0.000** |
| FacesUCR | 0.088 | 0.087 | 0.090 | 0.090 | **0.066** | **0.066** |
| Fish | 0.160 | **0.023** | 0.080 | 0.034 | 0.074 | 0.051 |
| GunPoint | 0.087 | 0.020 | 0.011 | 0.007 | **0.000** | **0.000** |
| Haptics | 0.588 | 0.523 | 0.488 | **0.471** | 0.558 | 0.520 |
| InlineSkate | 0.613 | 0.615 | 0.603 | **0.551** | 0.598 | 0.600 |
| ItalyPower | 0.045 | 0.048 | 0.096 | 0.070 | 0.035 | **0.030** |
| Lightning2 | **0.131** | 0.344 | 0.257 | 0.148 | **0.131** | 0.148 |
| Lightning7 | 0.288 | 0.260 | 0.262 | 0.205 | 0.178 | **0.151** |
| MALLAT | 0.086 | 0.060 | 0.037 | **0.035** | 0.042 | **0.035** |
| MedicalImages | 0.253 | 0.396 | 0.269 | **0.230** | 0.262 | 0.258 |
| MoteStrain | 0.134 | 0.109 | 0.135 | 0.090 | **0.074** | 0.084 |
| OliveOil | 0.167 | 0.100 | 0.090 | 0.167 | **0.033** | 0.033 |
| OSULeaf | 0.384 | 0.285 | 0.329 | **0.120** | 0.281 | 0.273 |
| SonyRobot | 0.305 | 0.067 | 0.175 | **0.042** | 0.107 | 0.118 |
| SonyRobotII | 0.141 | 0.115 | 0.196 | 0.084 | 0.084 | **0.078** |
| StarLightC | 0.095 | 0.024 | **0.022** | 0.040 | 0.027 | 0.027 |
| SwedishLeaf | 0.157 | 0.093 | 0.075 | 0.061 | 0.066 | **0.046** |
| Symbols | 0.062 | 0.114 | 0.034 | 0.036 | 0.030 | **0.022** |
| SynthCtrl | 0.017 | 0.017 | 0.008 | 0.007 | **0.000** | **0.000** |
| Trace | 0.010 | 0.020 | 0.020 | **0.000** | **0.000** | **0.000** |
| TwoLeadECG | 0.132 | **0.004** | 0.046 | 0.007 | 0.182 | 0.029 |
| TwoPatterns | 0.002 | 0.059 | **0.001** | 0.004 | 0.042 | 0.024 |
| UWaveX | 0.227 | 0.216 | **0.164** | 0.280 | 0.206 | 0.213 |
| UWaveY | 0.301 | 0.303 | **0.249** | 0.399 | 0.272 | 0.280 |
| UWaveZ | 0.322 | 0.273 | **0.217** | 0.321 | 0.265 | 0.267 |
| Wafer | 0.005 | 0.002 | 0.004 | **0.001** | 0.002 | 0.002 |

**Table 2** continued

| Dataset | DTW | ST | TSBF | HOG1D | CovNN | CovSVM |
|---|---|---|---|---|---|---|
| WordSyn | **0.252** | 0.403 | 0.302 | 0.483 | 0.282 | 0.332 |
| Yoga | 0.155 | 0.195 | 0.149 | 0.182 | **0.134** | 0.175 |

Best results are indicated in bold

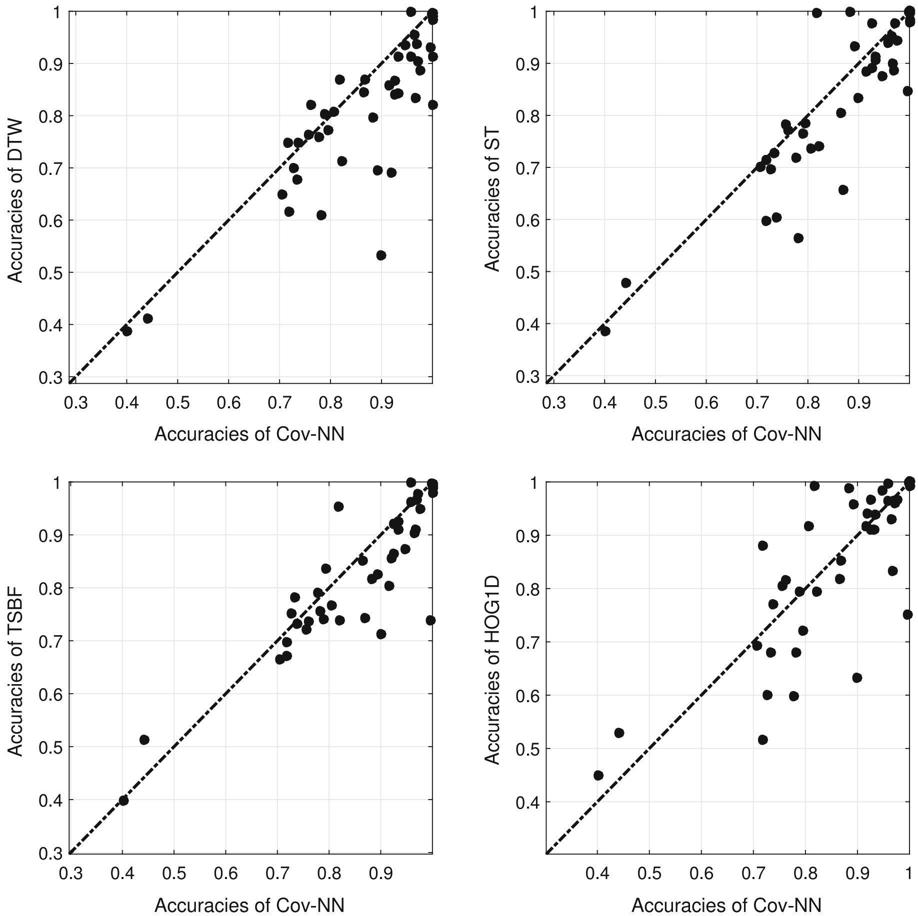


**Fig. 3** Comparison of CovNN with other methods on 43 datasets based on the results presented in Table 2. CovNN is better than DTW, ST, TSBF and HOG1D + DTW-MDS for 34, 33, 34 and 22 of the 43 datasets, respectively

8, 8 and 15 for the same order of the methods. CovSVM has the best and same performance as HOG-1D + DTW-MDS for 3 of the datasets.

Another critical performance parameter is the rank of the proposed approach among other methods. The rank is crucial in the sense that it indicates the consistency of the performance of a method among various results. For this purpose, we calculate the average rank of CovNN and CovSVM in a pool with other 4 methods. As mentioned before, the result of ST is missing for ECG200 dataset and therefore this dataset is excluded for this analysis. As can be seen

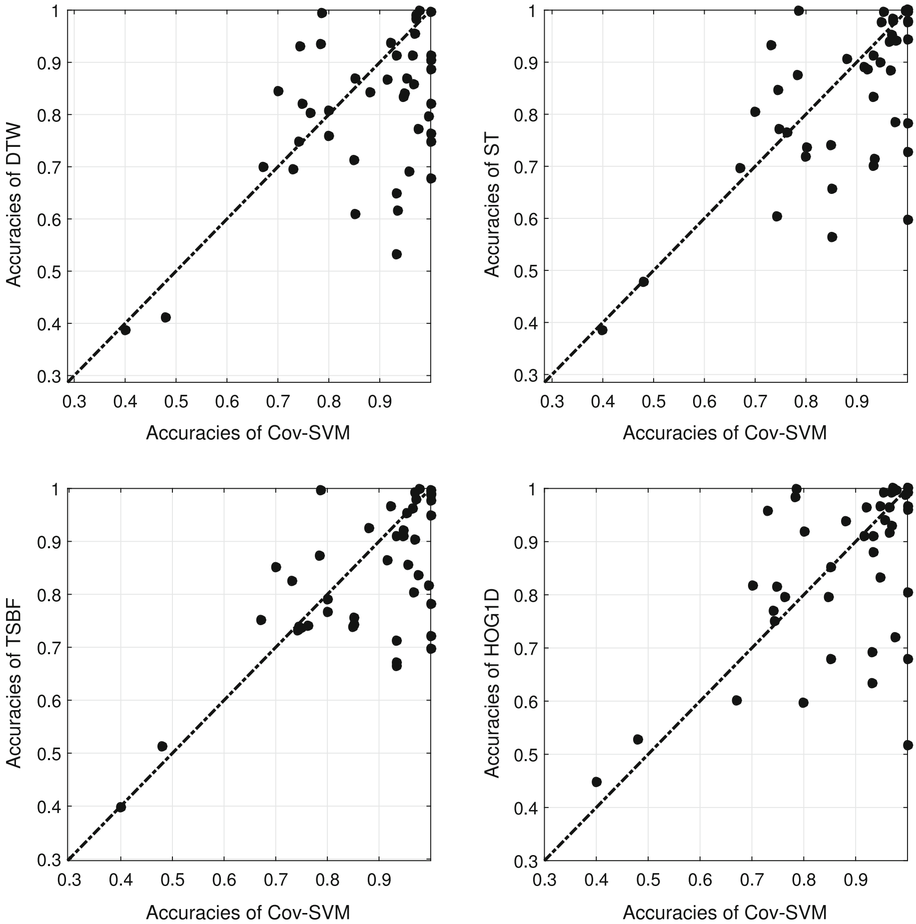


**Fig. 4** Comparison of CovSVM with other methods on 43 datasets based on the results presented in Table 2. CovSVM is better than DTW, ST, TSBF and HOG1D + DTW-MDS for 35, 34, 35 and 25 of the 43 datasets, respectively

in Fig. 5, CovNN and CovSVM significantly outperform the compared four methods. The average ranks of the CovNN and CovSVM are 2.07 and 1.92, whereas the ranks of the nearest competitors are 2.52 and 2.59, respectively.

As can be derived from the comparison with the other methods, CovSVM has better performance than CovNN. A pairwise comparison between CovNN and CovSVM is performed on 43 datasets. The scatter plot for the comparison between CovNN and CovSVM is provided in Fig. 6. The number of the datasets for which CovSVM has better results is 24, whereas CovNN is better for 12 datasets. For remaining 7 datasets, CovNN and CovSVM have equal classification accuracies. As a general observation, CovNN and CovSVM have similar performances. They perform well in the datasets where the covariance representation performs well.

When we conducted a more detailed analysis of the performance of the proposed method, it has been observed that the proposed method with both classifiers performs well except for the motion data. In the UCR repository, there are six different types of data as listed in

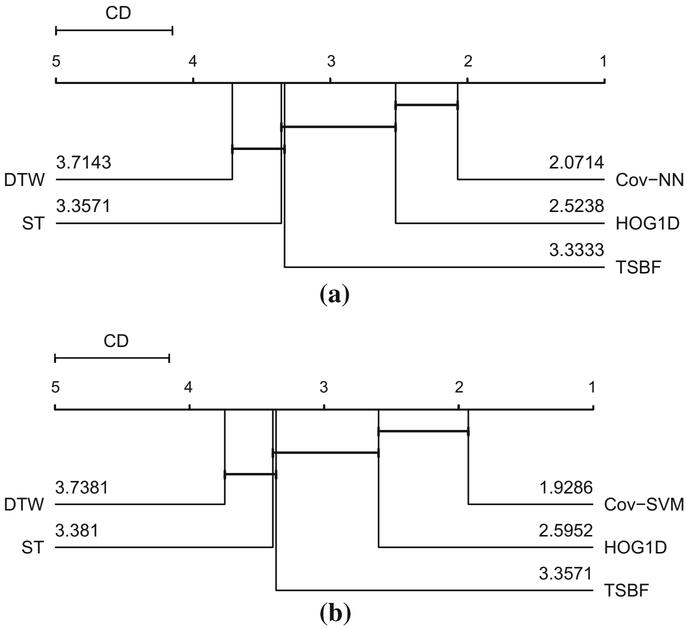


**Fig. 5** Critical difference diagrams for the proposed approach with two classifiers and compared approaches. **a** Critical difference diagram for CovNN. **b** Critical difference diagram for CovSVM

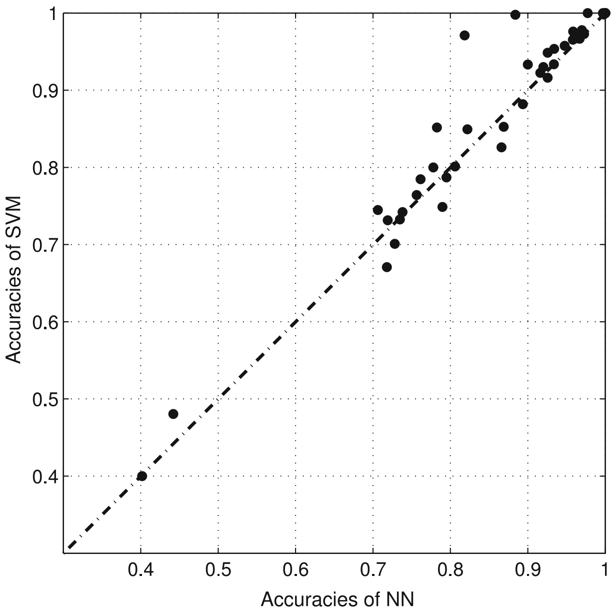


**Fig. 6** Comparison of NN and SVM classifiers on 43 datasets based on the results presented in Table 2. SVM classifier is better than the 1-NN classifier on 23 of the 43 datasets. For 13 of the datasets, 1-NN classifier has better results and for remaining 7 datasets, two classifiers have same classification accuracies

**Table 3** Error rates for remaining motion datasets

| Dataset | DTW | ST | CovNN | CovSVM |
|---|---|---|---|---|
| ToeSeg1 | 0.250 | 0.044 | 0.1711 | 0.206 |
| ToeSeg2 | 0.092 | 0.146 | 0.1231 | 0.146 |

Table 1. These types are image outline, motion, ECG, spectro, sensor reading and simulated. There are 11 motion datasets (*CricketX, CricketY, CricketZ, GunPoint, Haptics, InlineSkate, ToeSegmentation1, ToeSegmentation2, UWaveX, UWaveY, UWaveZ*). For 9 of them, except *ToeSegmentation1* and *ToeSegmentation2*, we have already presented the results in Table 2. CovNN and CovSVM have the best result for only *GunPoint*. For remaining 2 datasets, among the compared methods, only DTW and Shapelet Transform have the results. TSBF and HOG1D have not reported their results in these datasets. We obtained the error rates for these 2 datasets and reported in Table 3.

As can be seen from Table 3, CovNN and CovSVM do not have the best classification accuracies in these 2 datasets. When we analyze the motion datasets in UCR repository, we have observed that the motion data might be shifted in the time domain. In more detail, same portions of the motions can occur in different time intervals. These time intervals are generally close; however, in some circumstances, the proposed method cannot handle such situations.

To the best of our knowledge, in time series classification literature, there is only one study [6] which outperforms our results in UCR datasets. The work in [6] outperforms CovNN in 24 of the 43 datasets, while CovNN outperforms [6] for 17 datasets and presents identical performance for 2 datasets. On the other hand, CovSVM has better classification accuracies for 19 of 42 datasets. For 21 datasets, the method given in [3] outperforms the CovSVM. For 2 datasets, we have a tie. We want to remind that the classification accuracy for ECG200 dataset has not been reported in [3]. These results show that the proposed method gives classification accuracies that are comparable to a state-of-the-art ensemble method. It should be reminded that the main purpose of this work is to introduce a novel representation for time series and present its results with basic classifiers.

### 5.2 Analysis on pointwise features

In the previous subsection, classification performances of CovNN and CovSVM are presented using the complete set of pointwise features as given in Eq. 4. In Eq. 4, we define a feature vector of six pointwise features. As already mentioned, the covariance descriptor is based on the second-order correlations between the pointwise features. Therefore, this feature vector represents the information of the relative changes within the time series. The value, derivative and time index are local features, and the rank, cumulative sum and the difference between the mean are global features. Using the local features, we emphasize on the characteristics of a point locally. Therefore, in the proposed approach, we consider the local features as the core features, by which the time series is principally represented as in our previous work [16]. In addition to the local features, we add the global (or auxiliary) features into the feature vector in this work. The global features are derived from the whole subsequence and give information about the prominence of the point within the subsequence. In the following, an analysis is performed to capture the performance of the proposed method with reference to a different combination of global (auxiliary) features by concatenating them to the complete set of local (core) features.

**Table 4** Analysis on pointwise features

| Features | NN Average accuracy | NN Standard deviation | SVM Average accuracy | SVM Standard deviation |
|---|---|---|---|---|
| LF + Rank | 0.838 | 0.153 | 0.842 | 0.152 |
| LF + CS | 0.832 | 0.147 | 0.848 | 0.149 |
| LF + DM | 0.834 | 0.151 | 0.835 | 0.155 |
| LF + Rank + CS | 0.855 | 0.139 | 0.867 | 0.141 |
| LF + Rank + DM | 0.837 | 0.154 | 0.839 | 0.154 |
| LF + CS + DM | 0.830 | 0.146 | 0.850 | 0.146 |
| LF + Rank + CS + DM | 0.858 | 0.137 | 0.867 | 0.143 |

Average accuracies for seven combinations of pointwise features are listed. *LF* three local features together, *CS* cumulative sum, *DM* difference between mean

When overall accuracy in all of 43 datasets is considered, the feature vector that includes all pointwise features as defined in Eq. 4 gives the best results as listed in Table 4. However, the feature vector of six pointwise features does not give the best results for some datasets as can be seen in Figs. 7 and 8. For 1-NN and SVM classifiers, better results are achieved, when some of the global features are omitted in 15 and 14 of 43 datasets, respectively. This result implies that the covariance representation may show better performance with an appropriate selection of pointwise features for other time series analysis problems. In CovNN case, when the best performing feature set is selected separately for each dataset, the average rank moves from 2.07 to 2. The number of datasets for which the proposed approach has the best performance remains as 18. On the other side, for CovSVM, when the best performing feature set is selected separately for each dataset, the average rank moves from 1.93 to 1.86. The number of datasets for which the proposed approach has the best performance increases from 20 to 21. Moreover, for both classifiers, we observe a correlation between the number of features used and the stability of the performance when we observe the standard deviations of the classification accuracies (see the last column of Table 4). The minimum standard deviation is achieved when the complete set of (six) features are utilized.

### 5.3 Analysis on computational complexity

In the first subsection, we compare the proposed method with well-known and state-of-the-art methods. It has been shown that the proposed method mostly outperforms the compared methods. In the previous subsection, we show that better results can be acquired with a different selection of pointwise features. In this subsection, we analyze the computational complexity of the CovNN and CovSVM and then compare it with the HOG-1D + DTW-MDS method given in [50]. The reasons why we choose of HOG-1D + DTW-MDS for comparison are that it is the closest competitor and is feature-based similarly to the proposed method. In Fig. 9, the total computational times reported separately for each dataset are given for CovNN, CovSVM and HOG-1D + DTW-MDS. Since there are remarkable differences in total computational times between our method and compared method, the computational times are depicted using a log-axis. The reported computation times are obtained in MATLAB 2016a with a desktop machine with 4 cores, Intel i5-6500 CPU, 8GB RAM.

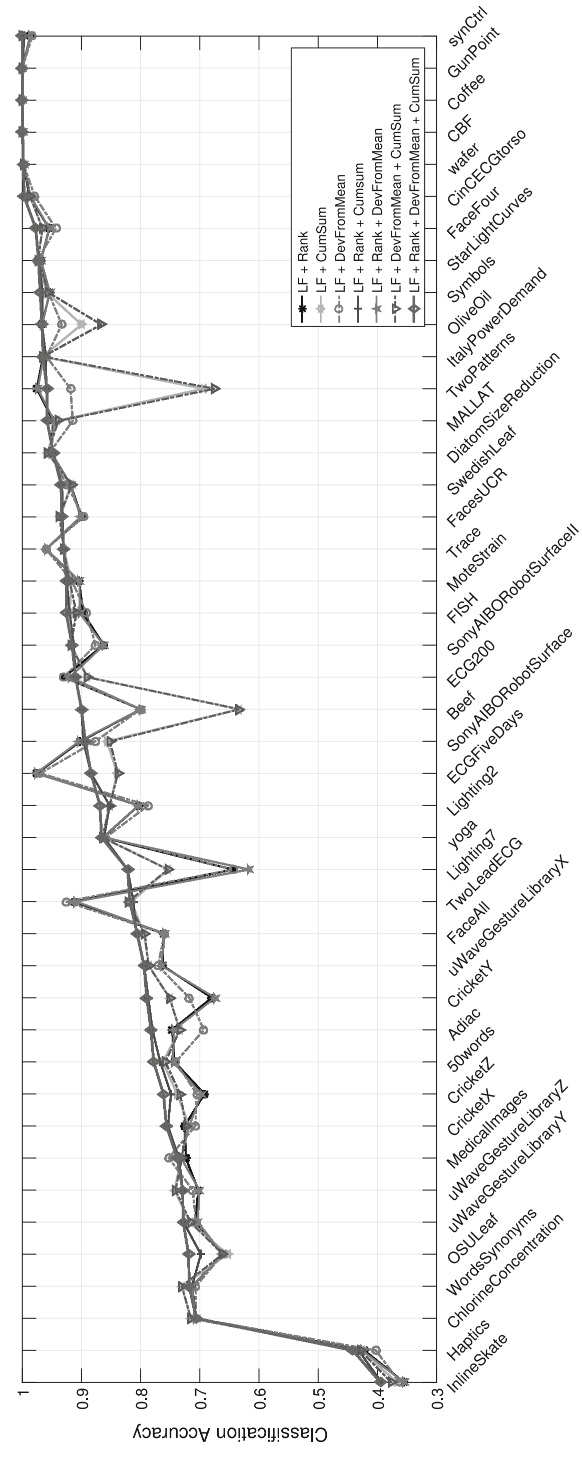


**Fig. 7** Accuracy values for different combinations of global pointwise features for 1-NN classifier. LF stands for local features, namely *value*, *derivative* and *time index*. The datasets are sorted according to accuracy values obtained in the case of six pointwise features

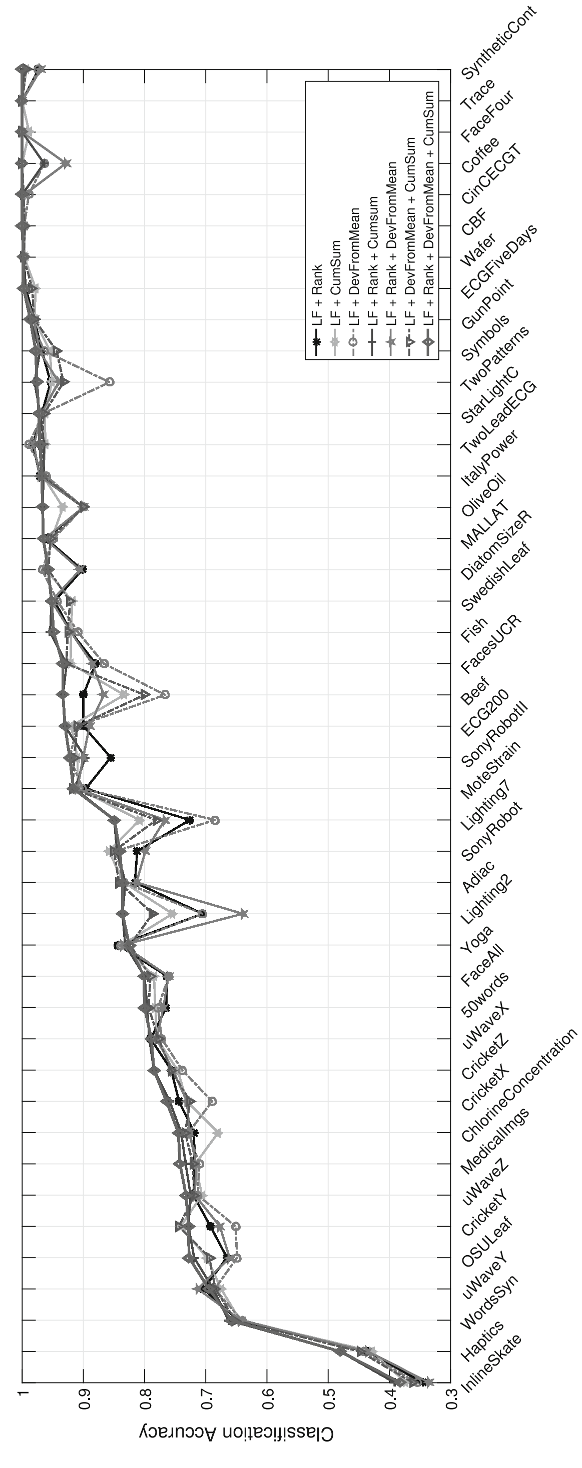


**Fig. 8** Accuracy values for different combinations of global pointwise features for SVM classifier. LF stands for local features, namely *value*, *derivative* and *time index*. The datasets are sorted according to accuracy values obtained in the case of six pointwise features

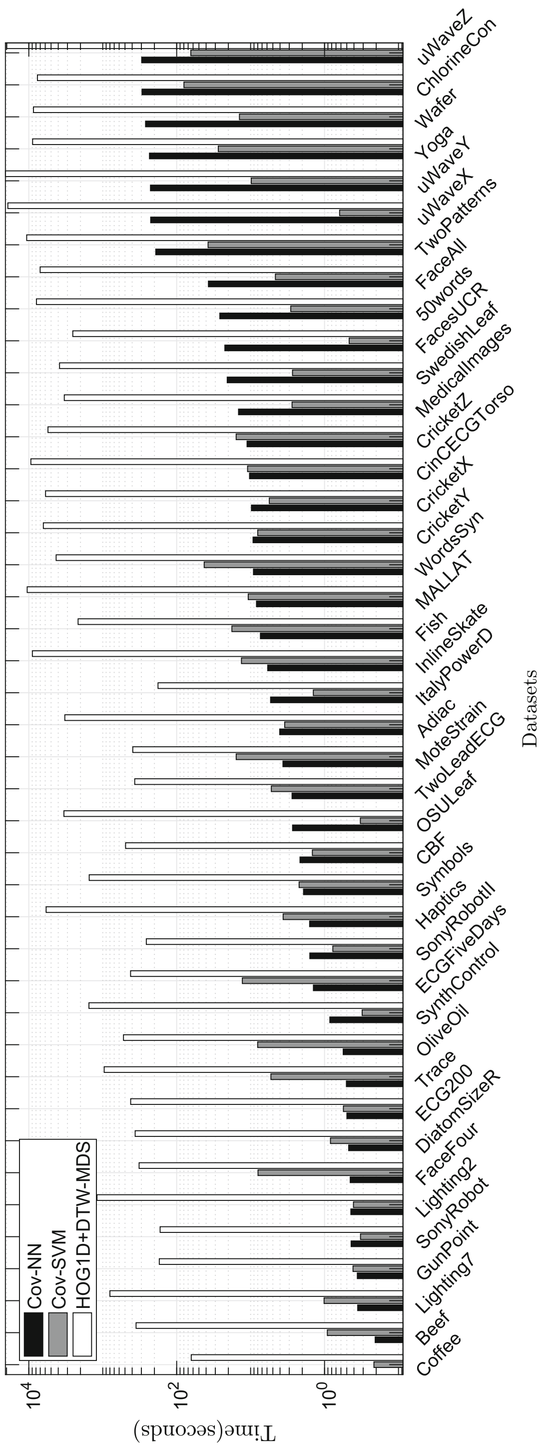


**Fig. 9** Comparison of the computational times of CovNN, CovSVM and HOG1D+DTW-MDS

The reported values for HOG1D + DTW-MDS [50] are obtained by running the publicly available code.[2] The code does not include some steps such as determination of DTW-MDS features in the method. For this reason, the computational time for the *StarLightCurves* dataset is not reported. Therefore, the comparison for the computational times is based on 42 datasets. The reported computational times for our approach include all the calculations from start to finish for both classifiers.

As can be seen from Fig. 9, CovNN finishes the classification within 10 seconds for most of the datasets. Moreover, for 12 datasets, it takes less than one second. Therefore, y-axis starts from $10^{-1}$ in Fig. 9. In the maximum case, the proposed method takes less than five minutes which occurs on *yoga* dataset. 1-NN classification takes 92.1 percent of the total calculation time on average. The average computational time for the proposed method is 46.6 seconds, while it is 4674 seconds for HOG1D + DTW-MDS. Consequently, it can be asserted that CovNN is 100 times faster than HOG1D + DTW-MDS technique. These results indicate that the proposed method is not only significantly accurate in classification, but also computationally efficient when compared to its competitor.

For CovSVM case, the computational times are still better than the work given in [50]. It is even better than CovNN in average. It begins to become faster when datasets become larger. This is due to the use of LIBSVM. Since LIBSVM is written in C++, it is normal to handle the larger datasets faster. The average time of CovSVM is about 6.98 seconds. When we examine the results more closely, we see that classification takes about 6.73% of the total time. Since we use MATLAB for 1-NN classifier, it takes so much time compared to the use of LIBSVM.

## 6 Conclusion

A novel approach is proposed for time series classification. Moreover, a novel distance calculation approach for time series is proposed by combining the covariance descriptor with a distance metric defined for covariance matrices. Conducted experiments on UCR time series datasets show that the proposed method yields results which are mostly outperforming some established methods such as DTW and shapelet transform and some state-of-the-art techniques such as TSBF and HOG-1D + DTW-MDS. Besides, in terms of computation time, the proposed method is very efficient when compared to a similar feature-based technique.

In addition to these satisfactory results with an efficient computation time, some challenging issues for time series such as missing data can also be addressed by using the covariance descriptor. Although the datasets in UCR repository do not include any missing data or time series of different lengths, these two issues can occur in a real time series classification problem. The covariance descriptor of the data with missing points will be again a square matrix with a length of number of features used. The missing points are inherently occupied in feature vectors since time index is used as a feature. Besides, the distance calculation between the time series of different lengths is generally problematic. In the proposed approach, invariant of the lengths of the time series, the dimensions of covariance matrices remain the same. Therefore, the proposed method is capable of measuring the distance between time series of different lengths.

A possible future work can be the comparison of the proposed method with standard methods after creating a dataset which includes missing points and time series of different

[2] https://github.com/jiapingz/TSClassification, lastly accessed in 04/09/2016.

lengths. Moreover, the proposed method can be utilized in other time series analysis problems such as anomaly detection and clustering.

## References


1. Libsvm library (2017). https://www.csie.ntu.edu.tw/~cjlin/libsvm/
2. Time series classification website (2017). www.timeseriesclassification.com
3. Ahmed N, Atiya AF, Gayar N, El-Shishiny H (2010) An empirical comparison of machine learning models for time series forecasting. Econom Rev 29(5–6):594–621
4. Arsigny V, Fillard P, Pennec X, Ayache N (2006) Log-euclidean metrics for fast and simple calculus on diffusion tensors. Magn Reson Med 56(2):411–421
5. Ayadi ME, Kamel M, Karray F (2011) Survey on speech emotion recognition: features, classification schemes, and databases. Pattern Recogn 44(3):572–587
6. Bagnall A, Lines J, Hills J, Bostrom A (2015) Time-series classification with cote: the collective of transformation-based ensembles. IEEE Trans Knowl Data Eng 27(9):2522–2535
7. Bailly A, Malinowski S, Tavenard R, Guyet T, Chapel L (2015) Bag-of-temporal-sift-words for time series classification. In: ECML/PKDD workshop on advanced analytics and learning on temporal data
8. Barachant A, Bonnet S, Congedo M, Jutten C (2013) Classification of covariance matrices using a riemannian-based kernel for bci applications. Neurocomputing 112:172–178
9. Bay Herbert TT, Gool LV (2006) Surf: speeded up robust features. In: European conference on computer vision, pp 404–417
10. Baydogan MG, Runger G, Tuv E (2013) A bag-of-features framework to classify time series. IEEE Trans Pattern Anal Mach Intell 35(11):2796–2802
11. Chebbi Z, Moakher M (2012) Means of hermitian positive-definite matrices based on the log-determinant alpha-divergence function. Linear Algebra Appl 436(7):1872–1889
12. Chen Y, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015) The UCR time series classification archive. www.cs.ucr.edu/~eamonn/time_series_data/
13. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. IEEE Conf Comput Vis Pattern Recogn 1:886–893
14. Deng H, Runger G, Tuv E, Vladimir M (2013) A time series forest for classification and feature extraction. Inf Sci 239:142–153
15. Erdem E, Erdem A (2013) Visual saliency estimation by nonlinearly integrating features using region covariances. J Vis 13(4), article no. 11. doi:10.1167/13.4.11
16. Ergezer H, Leblebicioğlu K (2016) Anomaly detection and activity perception using covariance descriptor for trajectories. In: European conference on computer vision, Springer, Berlin, pp 728–742
17. Esling P, Agon C (2012) Time-series data mining. ACM Comput Surv (CSUR) 45(1), article no. 12
18. Freifeld O (2014) Statistics on manifolds with applications to modeling shape deformations. Ph.D. Thesis, Brown University
19. Förstner W, Moonen B (2003) A metric for covariance matrices. Geodesy-The Challenge of the 3rd Millennium. Springer, Berlin
20. Fu Z, Lu G, Ting K, Zhang D (2011) Music classification via the bag-of features approach. Pattern Recogn Lett 32(14):1768–1777
21. Fulcher BD, Jones NS (2014) Highly comparative feature-based time-series classification. IEEE Trans Knowl Data Eng 26(12):3026–3037
22. Guo K, Ishwar P, Konrad J (2013) Action recognition from video using feature covariance matrices. IEEE Trans Image Process 22(6):2479–2494
23. Hills J, Lines J, Baranauskas E, Mapp J, Bagnall A (2014) Classification of time series by shapelet transformation. Data Min Knowl Disc 28(4):851–881
24. Jeong Y, Jeong M, Omitaomu O (2011) Weighted dynamic time warping for time series classification. Pattern Recogn 44(9):2231–2240
25. Karlsson I, Papapetrou P, Boström H (2016) Generalized random shapelet forests. Data Min Knowl Disc 30(5):1053–1085
26. Kate RJ (2016) Using dynamic time warping distances as features for improved time series classification. Data Min Knowl Disc 30(2):283–312
27. Keogh EJ, Pazzani MJ (2001) Derivative dynamic time warping. In: Proceedings of the 2001 SIAM international conference on data mining
28. Laptev I (2005) On space-time interest points. Int J Comput Vis 64(2–3):107–123
29. Lee JM (2006) Riemannian manifolds: an introduction to curvature, vol 176. Springer, Berlin

30. Lin J, Keogh E, Wei L, Lonardi S (2007) Experiencing sax: a novel symbolic representation of time series. Data Min Knowl Disc 15(2):107–144
31. Lines J, Bagnall A (2015) Time series classification with ensembles of elastic distance measures. Data Min Knowl Disc 29:565–592
32. Lowe D (1999) Object recognition from local scale-invariant features. Proc Seventh IEEE Int Conf Comput Vis 2:1150–1157
33. Mikolajczyk K et al (2005) A performance evaluation of local descriptors. IEEE Trans Pattern Anal Mach Intell 27(10):1615–1630
34. Mohan A, Papageorgiou C, Poggio T (2001) Example-based object detection in images by components. IEEE Trans Pattern Anal Mach Intell 23(4):349–361
35. Mueen A, Keogh E, Youngin N (2011) Logical-shapelets: an expressive primitive for time series classification. In: Proceedings of 17th ACM SIGKDD internatinal conference knowledge discovery data mining, pp 1154–1162
36. Pongpaichet S, Tang M, Jalali L, Jain R (2016) Using photos as micro-reports of events. In: In Proceedings of the 2016 ACM on international conference on multimedia retrieval, pp 87–94. ACM
37. Porikli F, Tuzel O, Meer P (2006) Covariance tracking using model update based on lie algebra. In: CVPR, pp 728–735
38. Ratanamahatana C, Keogh E (2005) Three myths about dynamic time warping data mining. In: In Proceedings of SIAM international conference on data mining (SDM'05), pp 506–510
39. Ratanamahatana CA, Keogh E (2004) Making time-series classification more accurate using learned constraints. In: Proceedings of SIAM international conference on data mining (SDM04), pp 11–22
40. Ratanamahatana CA, Lin J, Gunopulos D, Keogh E, Vlachos M, Das G (2005) Mining time series data. In: Data mining and knowledge discovery handbook 1069–1103, Springer, US
41. Sadatnejad K, Ghidary SS (2016) Kernel learning over the manifold of symmetric positive definite matrices for dimensionality reduction in a bci application. Neurocomputing 179:152–160
42. Tang M, Agrawal P, Nie F, Pongpaichet S, Jain R (2016) A graph based multimodal geospatial interpolation framework. In: In 2016 IEEE international conference on multimedia and expo (ICME), pp 1–6
43. Tang M, Nie F, Pongpaichet S, Jain R (2017) From photo streams to evolving situations. arXiv preprint arXiv:1702.05878
44. Tang M, Wu X, Agrawal P, Pongpaichet S, Jain R (2017) Integration of diverse data sources for spatial PM2. 5 data interpolation. IEEE Trans Multimedia 19(2):408–417.
45. Tuzel O, Porikli F, Meer P (2006) Region covariance: a fast descriptor for detection and classification. In: European conference on computer vision. Springer, Berlin, pp 589–600
46. Tuzel O, Porikli F, Meer P (2008) Pedestrian detection via classification on riemannian manifolds. IEEE Trans Pattern Anal Mach Intell 30(10):1713–1727
47. Wang X, Mueen A, Ding H, Trajcevski G, Scheuermann P, Keogh E (2013) Experimental comparison of representation methods and distance measures for time series data. Data Min Knowl Disc 26(2):275–309
48. Ye L, Keogh E (2009) Time series shapelets: A new primitive for data mining. In: Proceedings of 15th ACM SIGKDD international conference on knowledge discovery data mining, pp 947–956
49. Zhang Q, Goldman S, Yu W, Fritts J (2002) Content-based image retrieval using multiple-instance learning. ICML 2:682–689
50. Zhao J, Itti L (2016) Classifying time series using local descriptors with hybrid sampling. IEEE Trans Knowl Data Eng 28(3):623–637

**Hamza Ergezer** received B.Sc. and M.Sc. degrees in Electrical and Electronics Engineering from Hacettepe University, 2004 and Middle East Technical University (METU), 2007, respectively. He is a Ph.D. student at Electrical and Electronics Engineering of METU. Also, since 2004, he has been working as a research engineer in MGEO division of Aselsan Inc. His main research interests include computer vision, machine learning and image processing.

**Kemal Leblebicioğlu** received his Ph.D. from the Department of Mathematics of the Middle East Technical University (METU), in 1988. He has been a full professor at the Department of Electrical and Electronics Engineering of METU since 1999. He has a background in optimization, optimal control theory, computer vision, intelligent systems, flight control, walking robots and unmanned vehicles. Dr. Leblebicioğlu has been a member of numerous international program committees and has been author of more than 40 journal papers, book chapters and numerous conference papers.