RBF Networks with Dynamic Barycenter Averaging Kernel for Time Series Classification

Hongyang Qin[1], Lifeng Shen[1], Chijun Sima[1], and Qianli Ma[1,2(✉)]

[1] School of Computer Science and Engineering,
South China University of Technology, Guangzhou 510006, China
scuterlifeng@foxmail.com, qianlima@scut.edu.cn
[2] Guangdong Key Laboratory of Big Data Analysis and Processing,
Guangzhou 510006, China

Abstract. Radial basis function (RBF) network has received many considerable realistic applications, due to its simple topological structure and strong capacity on function approximation. However, the core of RBF network is its static kernel function, which is based on the Euclidean distance and cannot be directly used for time series classification (TSC). In this paper, a new temporal kernel called Dynamic Barycenter Averaging Kernel (DBAK) is introduced into RBF network. Specifically, we first determine the DBAK's centers by combining k-means clustering with a faster DTW-based averaging algorithm called DTW Barycenter Averaging (DBA). And then, to allow the stable gradient-training process in the whole network, a normalization term is added to the kernel formulation. By integrating the warping path information between the input time series and the centers of kernel, DBAK based RBF network (DBAK-RBF) can efficiently work for TSC tasks. Experimental results demonstrate that DBAK-RBF can achieve the better performance than previous models on benchmark time series datasets.

Keywords: Radial basis function · Dynamic time warping
Kernel function · Time series classification

1 Introduction

Time series classification (TSC) is a hot research topic drawing wide attention in many fields, including atmospheric monitoring [17], clinical medicine [22], robotics [25], financial stock data analysis [23], etc. Recent years have witnessed a growing body of research works on time series classification (TSC). These research works can be roughly divided into three categories: (1) the first one is *distance*-based methods which aim to combine standard time series benchmark distance measures with other classifiers for TSC tasks. The mainstream approaches include 1NN-DTW model [4,9] combining one nearest neighbor (1NN) classifier with dynamic time warping (DTW) and its variant based

Z.-H. Zhou et al. (Eds.): ICAI 2018, CCIS 888, pp. 139–152, 2018.
https://doi.org/10.1007/978-981-13-2122-1_11

on derivative distance called DD_{DTW} [10]; (2) the second one is *feature*-based classifiers. This category aims to extract representative features from raw time series and classify these time series by the learned discriminative features. The methods in this category include shapelet transform (ST) [12], learned shapelets (LS) [11], bag of SFA symbols (BOSS) [20], time series forest (TSF) [8], time series bag of features (TSBF) [3]; (3) the third category includes *ensemble* methods which present strong TSC baseline results. The two representative methods are Elastic Ensemble (EE) [14] and the collection of transformation ensembles (COTE) [1]. To be more specific, EE method is an ensemble classifier with 1NN based on 11 elastic distance measures, while COTE ensembles 35 different classifiers constructed in the time, frequency, change, and shapelet transformation domains. Although the above works all have made progress, TSC tasks remain to be challenging due to the fact that practical time series inevitably presents time-shift invariance, high-dimensionality and complex dynamics.

Radial basis function (RBF) network, first introduced by Broomhead and Lowe [5], is a simple and efficient tool for modeling time series. RBF networks typically have three layers: an input layer, a hidden layer with a non-linear RBF kernel function and a linear output layer. Given a D-dimensional sample $\boldsymbol{x}_i$, a RBF network with P hidden neurons can be defined by the following equations:

$$\phi_{i,j} = \phi(\|\boldsymbol{x}_i - \boldsymbol{c}_j\|) = \exp\left(\frac{-\|\boldsymbol{x}_i - \boldsymbol{c}_j\|^2}{\sigma_j^2}\right) \tag{1}$$

$$y_{i,k} = f(\sum_{j=1}^{P} w_{j,k}\phi_{i,j} + b_k), \quad k = 1, 2, \cdots, K \tag{2}$$

where $\phi_{i,j}$ is the activation value of the j-th hidden neuron of this RBF network. $\phi(\cdot)$ denotes the RBF's kernel function and $\boldsymbol{c}_j$ is the center vector of the kernel function in the j-th hidden neuron. $y_{i,k}$ is the k-th output unit corresponding to the i-th input. f denotes the activation function. K is the number of output nodes. $w_{j,k}$ is the weight and b_k is the bias term.

Due to the existing of the RBF's kernel, RBF network is an universal approximator and can approximate any continuous function on a closed, bounded set with arbitrary precision when it has enough hidden neurons [15]. In this sense, RBF network is versatile, whose functions include function approximation, time series forecasting and nonlinear system identification, etc. However, RBF network has deficiencies in its kernel function as it is based on the Euclidean distance and cannot be directly used for time series classification. That is because the Euclidean distance assumes that each sample pair has been aligned with the same length, and time shifts or time distortions occur frequently and unpredictably in time series data.

In an attempt to adapt RBF kernel to time series classification, using DTW distance to replace the Euclidean distance is a feasible choice. The earliest work is the support vector machine (SVM) with a dynamic time-alignment kernel (DTAK) proposed by Shimodaira et al. [21]. DTAK uses inner product and kernel function to compute the distance between two aligned points (rather than

Euclidean distance) and then minimizes the accumulated distance like DTW. In a different manner, Bahlmann et al. directly replace the Euclidean distance with DTW and propose an invariant of the gaussian radius basis function called gaussian DTW (GDTW) kernel [2], while GDTW ignored the warping path information. Most recently, Xue et al. developed an altered Gaussian DTW (AGDTW) [24] kernel function to obtain corresponding transformed kernel features for time series classification. The main idea of AGDTW is to align time series at first and then calculate the values of the corresponding kernel function over the warping path. However, the AGDTW kernel is unnormalized and dependent on the length of warping path. Thus this kernel cannot be directly applied in gradient training-based RBF network.

In this paper, we propose a new temporal kernel called Dynamic Barycenter Averaging Kernel (DBAK) and combine it with RBF networks. DBAK is based on the AGDTW. Specifically, we first determine the DBAK's centers by k-means clustering with a DTW barycenter averaging (DBA) algorithm developed by Petitjean et al. [18]. Note that here the kernel center is a time series rather than a vector. Furthermore, to keep a stable gradient-training process in the whole network, a scaled term is added to normalize the proposed kernel. By integrating the warping path information between the input time series and the kernel's time series center, DBAK based RBF network can be efficiently applied in the TSC task.

The rest of this paper is organized as follows. In Sect. 2, we provide preliminaries related to our work. And then in Sect. 3, we describe the details of the proposed model DBAK-RBF. The experimental results are reported in Sect. 4. Finally, we make conclusions in Sect. 5.

2 Preliminaries

In this section, we will provide a brief review of dynamic time warping (DTW), AGDTW [24] and the dtw barycenter averaging (DBA) algorithm. More details about RBF network can be found in [5,15].

2.1 Dynamic Time Warping (DTW)

Dynamic time warping (DTW) [4] is a widely-used similarity measure for time series. It is based on the Levenshtein distance and aims to find the optimal alignment paths between two sequences. Given two sequences S and T defined by:

$$S = (a_1, a_2, \ldots, a_N) \tag{3}$$

$$T = (b_1, b_2, \ldots, b_M) \tag{4}$$

where N denotes the length of S and M denotes the length of T. And then, these two sequences can be arranged to an N-by-M grid $\mathcal{G}$, where the grid point $\mathcal{G}(i, j)$ denotes an alignment between the elements a_i from sequence S and elements b_j from sequence T. A warping path $W = (w_1, w_1, \ldots, w_K)$ between these two

sequences is a sequence of points in grid $\mathcal{G}$ (K denotes the length of this path). This warping path satisfies three conditions:

1. boundary condition: this condition restricts endpoints of path by $w_1 = \mathcal{G}(1,1)$ and $w_K = \mathcal{G}(N,M)$;
2. continuity: for each pair $w_k = \mathcal{G}(i_k, j_k)$ and $w_{k+1} = \mathcal{G}(i_{k+1}, j_{k+1})$, the warping path satisfies $i_{k+1} - i_k \leq 1$ and $j_{k+1} - j_k \leq 1$;
3. monotonicity: the points in the path must be monotonically ordered by time, e.g., $i_{k+1} - i_k \geq 0$ and $j_{k+1} - j_k \geq 0$;

In this way, searching for a DTW path is equivalent to minimizing all potential alignment paths in terms of cumulative distance cost. The DTW distance can be formally formulated by

$$DTW(S,T) = \min_{W} \sum_{k=1}^{K} \delta(w_k) \tag{5}$$

where δ denotes a distance between two aligned points a_{i_k} and b_{j_k}. In this work, we use the squared Euclidean distance:

$$\delta(a_{i_k}, b_{j_k}) = \|a_{i_k} - b_{j_k}\|_2^2 \tag{6}$$

Actually, the problem (5) can be solved by dynamic programming over a cumulative distance matrix Γ. The recurrence relation in DTW problem can be formulated by

$$\Gamma(i,j) = \delta(a_{i_k}, b_{j_k}) + \min\{\Gamma(i-1, j-1), \Gamma(i-1, j), \Gamma(i, j-1)\} \tag{7}$$

where $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, M$. The initial conditions are

$$\Gamma(0,0) = 0;\ \Gamma(i,0) = \infty;\ \Gamma(0,j) = \infty \tag{8}$$

2.2 Altered Gaussian DTW (AGDTW)

Given two time series S and T defined in Eqs. 3 and 4, a GDTW kernel [2] can be formulated by

$$\mathcal{K}_{GDTW}(S,T) = \exp(-\frac{DTW(S,T)^2}{\sigma^2}) \tag{9}$$

where σ indicates the width of the kernel function and satisfies $\sigma \neq 0$. Although Eq. (9) is very similar to a general gaussian RBF kernel, it uses the DTW distance in the kernel rather than the Euclidean measure.

To integrate the warping path information in kernel function, an altered gaussian DTW (AGDTW) is developed. Let $\{a_{i_k}, b_{j_k}\}$ be the aligned point pair corresponding to the time series S and T, and the length of warping path is K, that is $k = 1, 2, \ldots, K$. Formally, AGDTW can be formulated by

$$\mathcal{K}_{AGDTW}(S,T) = \sum_{k=1}^{K} \exp(-\frac{\delta(a_{i_k}, b_{j_k})^2}{\sigma^2}) \tag{10}$$

2.3 DTW Barycenter Averaging (DBA)

DTW Barycenter Averaging (DBA) [18] is a global technique for averaging a set of time series. In our work, this technique will be used for estimating the kernel's centers. The main idea of DBA is to iteratively refine an initially (or temporary) average time series, in order to minimize its squared distance (DTW) to averaged sequences. Technically, the DBA works in two steps at each iteration:

1. Compute DTW distance between each time series sample and a temporary average sequence, and find associations between coordinates of the average sequence and coordinates of the set of time series;
2. Update each coordinate of the average sequence as the barycenter of coordinates associated to this coordinate at the first step;

More details about DBA can be found in [18].

In the next section, we will propose a new temporal kernel based on this AGDTW, which allows a stable gradient-training in the RBF networks.

3 DBAK-RBF Network

In this section, we explicate on our RBF network with a new dynamic barycenter averaging kernel (DBAK). We call our model DBAK-RBF for short.

3.1 Formulation

Assume that a time series dataset we used is $\mathcal{D} = \{T_1, T_2, \ldots, T_{N_D}\}$, which is a collection of N_D samples. Let a sample be $T_i = (x_1, x_2, \ldots, x_{L_{T_i}})$, where $x_n \in \mathcal{R}^d$ $(n = 1, 2, \ldots, L_{T_i})$, and L_{T_i} denotes the dimension of input time series. In this way, we can apply our DBAK-RBF in the time series classification on $\mathcal{D}$.

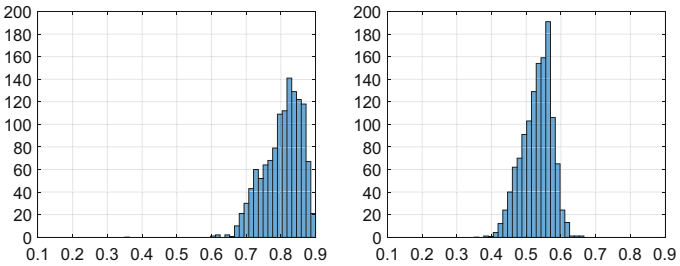
Our DBAK-RBF network is based on the topological structure of general RBF network and has a single hidden layer. The core of DBAK-RBF network is the DBAK function, which is formulated by

$$\mathcal{K}_{DBAK}(T_n, C_p) = \frac{\mathcal{M}}{K^2} \cdot \sum_{k=1}^{K} \exp\left(-\frac{\delta(x_{n,i_k}, c_{p,j_k})^2}{\sigma_p^2}\right) \tag{11}$$

where C_p denotes the center of p-th hidden neuron's activation function. $\mathcal{M} = \max\{L_{T_n}, L_{C_p}\}$. It is worth mentioning that in the case of the general RBF network, the center is a vector; while in our DBKA-RBF network, the center is a time series. The reason behind this selection will be explained later. The parameter σ denotes the width of the kernel function. K is the length of the warping path between time series T_n and C_p. L_{T_n} and L_{C_p} are the length of T_n and C_p, respectively. Compared with AGDTW kernel in Eq. 10, our DBKA has a normalization term in Eq. 11, which can be decomposed into two parts:

$$\frac{\mathcal{M}}{K^2} = \frac{1}{K} \cdot \frac{\mathcal{M}}{K} \tag{12}$$

Here, the part $\frac{1}{K}$ is used to eliminate the effects of the warping length K. In AGDTW, values of kernel easily turn out to be much larger due to a large K, which will hinder the gradient-training. The $\frac{\mathcal{M}}{K}$ is the other scaling term and can make the outputs of $\mathcal{K}_{DBAK}(T_n, C_p)$ tend to a standard normal distribution, as is shown in Fig. 1. Its effectiveness will be verified in later experiments.



(a) DBAK without $\mathcal{M}/K$ (b) DBAK with $\mathcal{M}/K$

Fig. 1. Effects of the scaling term $\mathcal{M}/K$

After computing the DBAK kernel's values in the hidden layer, we can formulate the c-th activation value $z_{n,c}$ as following

$$z_{n,c} = \sum_{p=1}^{P} w_{p,c} \cdot \mathcal{K}_{DBAK}(T_n, C_p) + b_c \tag{13}$$

where $c = 1, 2, \cdots, C$ and $p = 1, 2, \cdots, P$. Using the **Softmax** function, we can have the posterior distribution $y_{n,c}$ of the c-th class corresponding to the n-th time series sample:

$$y_{n,c} = \mathbf{Softmax}(z_{n,c}) = \frac{\exp(z_{nc})}{\sum_{\hat{c}=1}^{C} \exp(z_{n\hat{c}})} \tag{14}$$

In this way, the classification target function can be defined by the well-known cross entropy loss:

$$\mathcal{L}_{\mathcal{D}} = -\frac{1}{N_D} \sum_{n=1}^{N_D} \sum_{c=1}^{C} \hat{y}_{n,c} \log y_{n,c} \tag{15}$$

where $\hat{y}_n$ denotes the true label of the n-th time series.

3.2 Training

The whole process of training our DBAK-RBF includes two steps: (1) the selection of the centers $\{C_p\}$ in Eq. 11 and (2) the estimation of other parameters, including the width $\{\sigma_p\}$, the weights $\{w_{p,c}\}$ and the biases $\{b_c\}$.

Algorithm 1. DBAK-RBF

Require: $\mathcal{D} = \{T_1, \cdots, T_{N_D}\}$: a set of time series; $\mathcal{Y} = \{y_1, \cdots, y_{N_D}\}$ corresponding labels of $\mathcal{D}$; C: the number of kernel's centers.

1. Clustering the given data $\mathcal{D}$ by k-means;
2. Using DBA algorithm to obtain the centeral time series in each cluster;
3. Initialize parameters $\{\sigma_p\}$, $\{w_{p,c}\}$, and $\{b_c\}$;
4. Forward-propagation by using Eq. 11, Eq. 13, Eq. 14 and Eq. 15;
5. Back-propagation by using gradients formulated by Eq. 16-19;

return the centers $\{C_p\}$; the optimized width $\{\sigma_p\}$, the weights $\{w_{p,c}\}$ and the biases $\{b_c\}$.

Step 1: Selection of Centers in DBAK. For traditional RBF networks, the kernel's centers $\{C_p\}$ usually are optimized by clustering algorithm like k-means. However, these techniques cannot be directly used in our DBKA-RBF network since they depend on the Euclidean distance metric and thus are not suitable for time series data. Using Dynamic Time Warping (DTW) is a natural way to select the centers when modeling time series data. However, this change will generate much higher computation costs because we need to compute the DTW distances of all sample pairs. Here, we combine the DTW-based averaging method called DTW Barycenter Averaging (DBA) [18] with k-means clustering to estimate our DBAK function's center time series $\{C_p\}$. Firstly, we use the k-means method to roughly divide the training set into different clusters. And then, for each cluster, DBA algorithm is used to learn an averaged time series as a center of DBAK function. This strategy is also suggested by the work [16].

Step 2: Gradient-Based Optimization. After fixing the centers in DBAK function, we can use gradient-based optimization techniques to learn other parameters, including $\{\sigma_p\}$, $\{w_{p,c}\}$ and $\{b_c\}$. According to Eqs. 11 to 15, we can easily obtain the corresponding gradient formulations.

The gradient of the width σ of DBAK function is given by

$$\frac{\partial \mathcal{L}_{\mathcal{D}}}{\sigma_p} = \frac{1}{N_D} \sum_{n=1}^{N_D} \sum_{c=1}^{C} (y_{n,c} - \hat{y}_{n,c}) w_{p,c} \cdot \frac{\partial \mathcal{K}_{DBAK}(T_n, C_p)}{\sigma_p} \tag{16}$$

$$\frac{\partial \mathcal{K}_{DBAK}(T_n, C_p)}{\sigma_p} = \frac{2\mathcal{M}}{\sigma_p^3 K^2} \sum_{k=1}^{K} \exp\left(-\frac{\delta(x_{n,i_k}, c_{p,j_k})^2}{\sigma_p^2}\right) \delta(x_{n,i_k}, c_{p,j_k})^2 \tag{17}$$

where $\mathcal{M} = \max\{L_{T_n}, L_{C_p}\}$ and p denotes the width of p-th hidden neuron's activation function.

As for the weights $\{w_{p,c}\}$ and the biases $\{b_c\}$, we have

$$\frac{\partial \mathcal{L}_{\mathcal{D}}}{w_{p,c}} = \frac{1}{N_D} \sum_{n=1}^{N_D} (y_{n,c} - \hat{y}_{n,c}) \mathcal{K}_{DBAK}(T_n, C_p) \tag{18}$$

$$\frac{\partial \mathcal{L}_{\mathcal{D}}}{b_c} = \frac{1}{N_D} \sum_{n=1}^{N_D} (y_{n,c} - \hat{y}_{n,c}) \tag{19}$$

With these gradients, we use the well-known ADAM method to do the stochastic optimization. More details about ADAM can be found in [13]. Algorithm 1 details the complete process of training our DBAK-RBF.

3.3 Discussion of Computational Complexity

In this section, we discuss the computational complexity of our model. The main computational cost in our model is calculation of the DTW distances. There are two parts where we use DTW. The first one is to compute center vectors with DBA. Here we need to compute the DTW distance between each time series sample and a temporary average sequence in an iterative way. The second is to compute the warping path between the given input time series and the centroids in the hidden neurons.

The original DTW algorithm has a quadratic time and space complexity. To accelerate the computation of DTW, FastDTW is an efficient and effective DTW variant, which can provide a linear time and space complexity [19]. For simplicity, given two time series with same length T, and then the worst-case behavior of FastDTW can be given by $\mathcal{O}(T \times (8r+14)) \approx \mathcal{O}(T)$, where r denotes the limit radius in FastDTW (a small positive constant value).

In this way, the computational complexity in the first part can be given by $\mathcal{O}(mTN_{iter})$, which is linear to the number of time series samples m and the length of time series T. N_{iter} denotes the number of iteration. The complexity of the second part is $O(nT)$ where n is the number of hidden neurons, whose complexity is also linear to the length T. According to these analysis, we can conclude that our model do not bring much more computational cost and can be efficiently used in linear time.

4 Experiments

In this section, we will thoroughly evaluate the performance of our DBAK-RBF by comparing it with other mainstream TSC methods.

The classification accuracy is used as performance measures and can be defined by

$$\text{Accuracy} = \frac{\text{the number of correctly classified data}}{\text{the total number of testing data}} \tag{20}$$

4.1 Comparisons Analysis

To examine the performance of the proposed DBAK-RBF, we perform a series of experiments on a publicly available Time Series datasets from the "UCR Time Series Data Mining Archive" [6]. UCR datasets exhibit different background knowledge in a wide range of application domains, which is very effective in testing the learning and classification ability of time series classifiers.

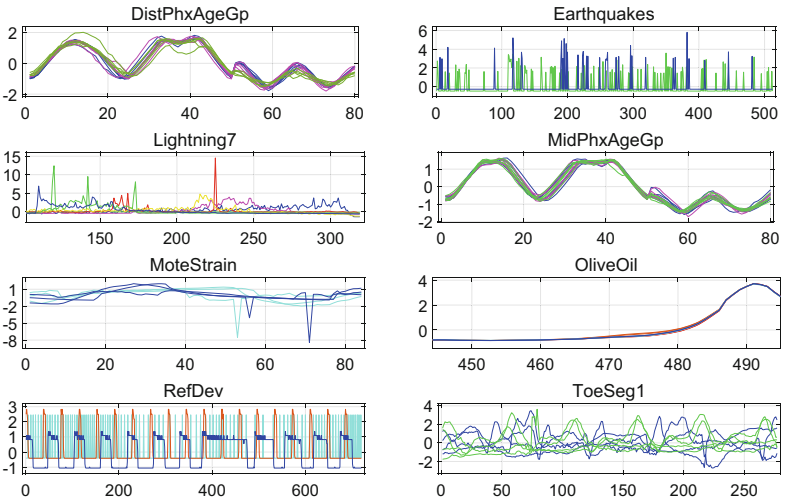


Fig. 2. Visualization of Six UCR Datasets

Table 1. The details of 15 selected UCR datasets

DATASETS	♯ CLASSES	♯ TRAIN	♯ TEST	LENGTH
DistalPhalanxOutlineAgeGroup	3	139	400	80
DistalPhalanxOutlineCorrect	2	276	600	80
DistalPhalanxTW	6	139	400	80
Earthquakes	2	139	322	512
Lightning2	2	60	61	637
Lightning7	7	70	73	319
MiddlePhalanxOutlineAgeGroup	3	154	400	80
MiddlePhalanxTW	6	154	399	80
MoteStrain	2	20	1252	84
OliveOil	4	30	30	570
ProximalPhalanxOutlineAgeGroup	3	400	205	80
ProximalPhalanxTW	6	205	400	80
RefrigerationDevices	3	375	375	720
ToeSegmentation1	2	40	228	277
Wine	2	57	54	234

Table 2. Accuracy of 9 mainstream TSC classifiers and our DBAK-RBF

DATASETS	DTW_{1NN}	DD_{DTW}	ST	LS	BOSS	TSF	TSBF	EE	COTE	DBAK-RBF
DistPhxAgeGp	0.770	0.705	0.770	0.719	0.748	0.748	0.712	0.691	0.748	**0.855**
DistPhxCorr	0.717	0.732	0.775	0.779	0.728	0.772	0.783	0.728	0.761	**0.845**
DistPhxTW	0.590	0.612	0.662	0.626	0.676	0.669	0.676	0.648	0.698	**0.800**
Earthquakes	0.719	0.705	0.741	0.741	0.748	0.748	0.748	0.741	0.748	**0.835**
Lightning2	0.869	0.869	0.738	0.820	0.836	0.803	0.738	**0.885**	0.869	0.869
Lightning7	0.726	0.671	0.726	0.795	0.685	0.753	0.726	0.767	0.808	**0.863**
MidPhxAgeGp	0.500	0.539	0.643	0.571	0.546	0.578	0.578	0.558	0.636	**0.805**
MidPhxTW	0.507	0.487	0.520	0.507	0.546	0.565	0.597	0.513	0.571	**0.647**
MoteStrain	0.835	0.833	0.897	0.883	0.879	0.869	0.903	0.883	**0.937**	0.890
OliveOil	0.833	0.833	0.900	0.167	0.867	0.867	0.833	0.867	0.900	**0.933**
ProxPhxAgeGp	0.805	0.800	0.844	0.834	0.834	0.849	0.849	0.805	0.854	**0.878**
ProxPhxTW	0.761	0.766	0.805	0.776	0.800	0.815	0.810	0.766	0.781	**0.825**
RefDev	0.464	0.445	0.581	0.515	0.499	**0.589**	0.472	0.437	0.547	**0.589**
ToeSegmentation1	0.772	0.807	0.965	0.934	0.939	0.741	0.781	0.829	**0.974**	0.943
Wine	0.574	0.574	0.796	0.500	0.741	0.630	0.611	0.574	0.648	**0.907**
Avg. Rank	8.067	8.533	4.367	6.533	5.667	5.000	5.167	6.900	3.233	1.533
Difference	6.533	7.000	2.833	5.000	4.133	3.467	3.633	5.367	1.700	-

We visualize the spectrogram of 6 datasets in Fig. 2. We can find that time series data from different categories present time-shift invariance, high-dimensionality (time direction), and complex dynamics. Thus, classifying time series data is not a trivial task.

In our experiments, we choose 15 datasets to verify the performance of our model. Table 1 summarizes the details of these selected datasets.

As for the compared baselines, we select 9 mainstream representative TSC models. They come from three categories we introduced in Sect. 1, including (1) distance-based methods; (2) feature-based methods and (3) ensemble-based methods.

In more details, for the distance-based methods, we compare with DTW_{1NN} [9] and DD_{DTW} [10]. For the feature-based methods, we select shapelet transform (ST) [12], learned shapelets (LS) [11], bag of SFA symbols (BOSS) [20], time series forest (TSF) [8] and time series bag of features (TSBF) [3]. For the ensemble-based methods, Elastic Ensemble (EE) [14] and the collection of transformation ensembles (COTE) [1] are chosen.

We report our comparison results in Table 2. As shown in this table, DBAK-RBF outperforms other TSC models with the average rank of 1.533 and achieves the best performance on 12 of the 15 datasets. The ensemble-based COTE and the feature-based ST model achieve the second and the third best performance, respectively. On the whole, ensemble-based and feature-based methods are better than the distance-based methods. Notably, our DBAK-RBF achieves much higher accuracy on some TSC tasks. For example, on the MidPhxAge dataset, DBAK-RBF can achieve an accuracy of 0.805, while the second best one ST only has an accuracy of 0.643. This is a huge improvement. For Wine, our DBAK-RBF works very well and achieves an accuracy of 0.907, while other classifiers

achieve accuracies of 0.6481 by $COTE$, 0.574 by DTW_{1NN}, DD_{DTW}, and EE. Moreover, there are also some datasets like Lightning2 on which feature-based methods do not perform well. On the Lightning2, DBAK-RBF has an accuracy of 0.8689, but feature-based methods have accuracies of 0.7377 (ST), 0.8197 (LS), 0.8361 ($BOSS$), 0.8033 (TSF) and 0.7377 ($TSFB$), respectively.

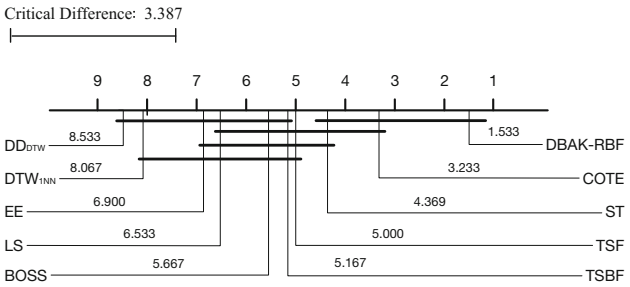


Fig. 3. Comparison of DBAK-RBF with other 9 classifiers on the Nemenyi test. Groups of classifiers that are not significantly different ($p = 0.05$) are connected.

In addition, we conduct non-parametric tests (Friedman test, Nemenyi post-hoc test) to make statistical comparisons [7]. The statistic τ_F of Friedman test is 15.08, which is larger than the critical value 1.950 ($p = 0.05$). Thus, the null-hypothesis (all the algorithms are equivalent) is rejected and we can proceed with a post-hoc Nemenyi test. The critical difference diagram compares the results of DBAK-RBF to other classification methods with the Nemenyi test, which is shown in Fig. 3. The performance of two classifiers is significantly different if the corresponding average ranks differ by at least a critical difference (CD) value. As seen in Fig. 3, the CD value here is 3.387. From this result, we can conclude that although our DBAK-RBF is only slightly better than COTE and ST, it significantly outperforms most of the other models.

4.2 Components Analysis

To further verify the effectiveness of the components (e.g. DBAK, normalization term) in our model, we conduct the components analysis in this section. We firstly compare DBAK-RBF with two baseline models on UCR datasets.

The first model is a regular RBF network. In this case, we allow this network to be directly applied in TSC task by regarding an input time series as an L_{T_i}-dimension input vector. As seen in Fig. 4a, we verify that our DBAK is better than the Euclidean distance (ED) based original RBF kernel (as there are more blue points in the figure).

In Fig. 4b, we use the k-means based on ED to fix centers in our DBAK-RBF, without using the DBA. Again, the result shows that our DBAK is much better than the case of using ED-based k-means.

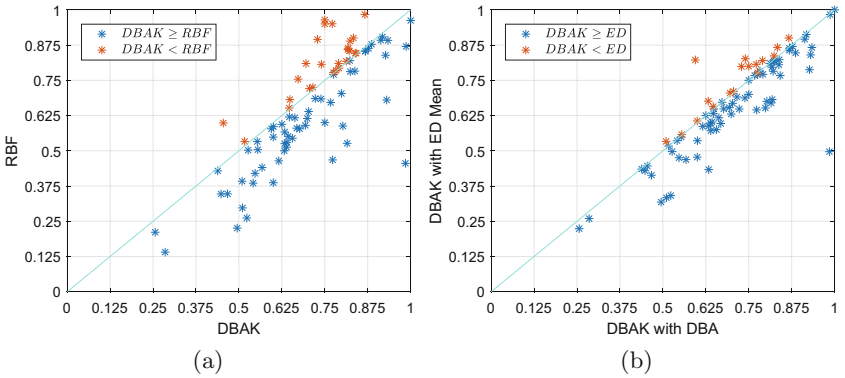


Fig. 4. (a) DBAK-RBF vs. RBF; (b) DBAK with DBA/ED-mean

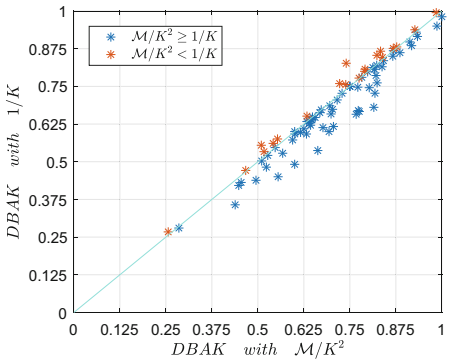


Fig. 5. DBAK with $\mathcal{M}/K^2$ vs. DBAK with $1/K$

Furthermore, we also investigate the performance of our normalization term $\mathcal{M}/K^2$ in Eq. 11. We compare our DBAK with $\mathcal{M}/K^2$ and the case with $1/K$ in Fig. 5. This result shows that the scaling term $\mathcal{M}/K$ significantly improves the performance on time series classification.

5 Conclusion

In this work, we develop a new DTW-based kernel, named Dynamic barycenter averaging kernel (DBAK), in RBF network to address the problem of time series classification. The main characteristics of the DBAK-based RBF network include: (1) we use a two-step k-means clustering based on a DTW-Barycenter-Averaging (DBA) algorithm to estimate the centers of RBF network's kernel function. (2) we add a normalization term to the kernel formulation to allow a stable gradient-training process. Experiment results demonstrate that our DBAK-RBF has a better or more competitive TSC performance than previous

TSC methods, including some strong baselines, such as ensemble-based COTE and feature-based ST.

Acknowledgment. The work described in this paper was partially funded by the National Natural Science Foundation of China (Grant No. 61502174), the Natural Science Foundation of Guangdong Province (Grant No. 2017A030313355, 2017A030313358, 2015A030313215), the Science and Technology Planning Project of Guangdong Province (Grant No. 2016A040403046), the Guangzhou Science and Technology Planning Project (Grant No. 201704030051), the Opening Project of Guangdong Province Key Laboratory of Big Data Analysis and Processing (Grant No. 2017014), and the Fundamental Research Funds for the Central Universities (Grant No. D2153950).

References

1. Bagnall, A., Lines, J., Hills, J., Bostrom, A.: Time-series classification with cote: the collective of transformation-based ensembles. IEEE Trans. Knowl. Data Eng. **27**(9), 2522–2535 (2015)
2. Bahlmann, C., Haasdonk, B., Burkhardt, H.: Online handwriting recognition with support vector machines - a Kernel approach. In: Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition, pp. 49–54 (2002)
3. Baydogan, M.G., Runger, G., Tuv, E.: A bag-of-features framework to classify time series. IEEE Trans. Pattern Anal. Mach. Intell. **35**(11), 2796–2802 (2013)
4. Berndt, D.J.: Using dynamic time warping to find patterns in time series. In: KDD Workshop, pp. 359–370 (1994)
5. Broomhead, D.S., Lowe, D.: Multivariable functional interpolation and adaptive networks. Complex Syst. **2**(3), 321–355 (1988)
6. Chen, Y., et al.: The UCR time series classification archive, July 2015
7. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
8. Deng, H., Runger, G., Tuv, E., Vladimir, M.: A time series forest for classification and feature extraction. Inf. Sci. **239**, 142–153 (2013)
9. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: experimental comparison of representations and distance measures. Proc. VLDB Endow. **1**(2), 1542–1552 (2008)
10. Górecki, T., Łuczak, M.: Using derivatives in time series classification. Data Min. Knowl. Discov. **26**(2), 310–331 (2013)
11. Grabocka, J., Schilling, N., Wistuba, M., Schmidt-Thieme, L.: Learning time-series shapelets. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2014, pp. 392–401. ACM, New York (2014)
12. Hills, J., Lines, J., Baranauskas, E., Mapp, J., Bagnall, A.: Classification of time series by shapelet transformation. Data Min. Knowl. Discov. **28**(4), 851–881 (2014)
13. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2014)
14. Lines, J., Bagnall, A.: Time series classification with ensembles of elastic distance measures. Data Min. Knowl. Disc. **29**(3), 565–592 (2015)
15. Park, J., Sandberg, I.W.: Universal approximation using radial-basis-function networks. Neural Comput. **3**(2), 246–257 (1991)

16. Petitjean, F., Forestier, G., Webb, G.I., Nicholson, A.E., Chen, Y., Keogh, E.: Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm. Knowl. Inf. Syst. **47**(1), 1–26 (2016)
17. Petitjean, F., Inglada, J., Gançarski, P.: Satellite image time series analysis under time warping. IEEE Trans. Geosci. Remote Sens. **50**(8), 3081–3095 (2012)
18. Petitjean, F., Ketterlin, A., Ganarski, P.: A global averaging method for dynamic time warping, with applications to clustering. Pattern Recogn. **44**(3), 678–693 (2011)
19. Salvador, S., Chan, P.: Toward accurate dynamic time warping in linear time and space. Intell. Data Anal. **11**(5), 561–580 (2007)
20. Schfer, P.: The boss is concerned with time series classification in the presence of noise. Data Min. Knowl. Discov. **29**(6), 1505–1530 (2015)
21. Shimodaira, H., Noma, K., Nakai, M., Sagayama, S.: Dynamic time-alignment kernel in support vector machine. In: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, pp. 921–928. MIT Press, Cambridge (2001)
22. 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. **26**(2), 275–309 (2013)
23. Wei, L.Y.: A hybrid ANFIS model based on empirical mode decomposition for stock time series forecasting. Appl. Soft Comput. **42**, 368–376 (2016)
24. Xue, Y., Zhang, L., Tao, Z., Wang, B., Li, F.: An altered Kernel transformation for time series classification. In: Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, E.-S.M. (eds.) ICONIP 2017 Part V. LNCS, vol. 10638, pp. 455–465. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70139-4_46
25. Zheng, Y., Liu, Q., Chen, E., Ge, Y., Zhao, J.L.: Time series classification using multi-channels deep convolutional neural networks. In: Li, F., Li, G., Hwang, S., Yao, B., Zhang, Z. (eds.) WAIM 2014. LNCS, vol. 8485, pp. 298–310. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08010-9_33