Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Dynamic Barycenter Averaging Kernel in RBF Networks for Time Series Classification

KEJIAN SHI[1], HONGYANG QIN[1], CHIJUN SIMA[1], SEN LI[1], LIFENG SHEN[1], QIANLI MA[1, 2]
[1]School of Computer Science and Engineering, South China University of Technology, Guangzhou, China(e-mail: qianlima@scut.edu.cn)
[2]Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou 510006, China

Corresponding author: Qianli Ma (e-mail: qianlima@scut.edu.cn).

ABSTRACT Radial basis function (RBF) network has been utilized in many applications due to its simple topological structure and strong capacity on function approximation. The core of RBF network is its static kernel function, which is based on the Euclidean distance and cannot obtain good performance for time series classification (TSC) due to the time-shift invariance, complex dynamics and different length of temporal data. This paper proposed a new temporal kernel, namely, the Dynamic Barycenter Averaging Kernel (DBAK) and introduced it into RBF network. First, we combine k-means clustering with a dynamic time warping (DTW) based averaging algorithm called DTW barycenter averaging (DBA) to determine the center of DBAK. Then, in order to facilitate the stable gradient-training process in the whole network, a normalization term is added into the kernel formulation. By integrating the information of the whole time warping path, our DBAK based RBF network (DBAK-RBF) performs efficiently for TSC tasks.

INDEX TERMS Radial basis function, Dynamic time warping, Kernel function, Time series classification.

I. INTRODUCTION

Time series classification (TSC) is a hot research topic that is attracting wide attention in many fields, including atmospheric monitoring [1], clinical medicine [2], robotics [3], financial stock data analysis [4], etc. Time series in the TSC mission can be divided into two types, the univariate time series and the multivariate ones. This paper focuses on the univariate time series. Recent years have witnessed a growing number of studies on it. These studies can be roughly divided into four categories: 1) the first one is *distance*-based methods which combine standard time series benchmark distance measures with other classifiers for TSC tasks. The mainstream methods include 1NN-DTW model [5], [6] which combines one nearest neighbor (1NN) classifier with dynamic time warping (DTW) and its variant based on derivative distance called DD_{DTW} [7]; 2) the second one is *feature*-based classifiers. They classify time series according to the discriminative features learned from the raw data. The methods in this class include shapelet transform (ST) [8], learned shapelets (LS) [9], bag of SFA symbols (BOSS) [10], time series forest (TSF) [11], time series bag of features (TSBF) [12]; 3) the third category includes *ensemble* methods which aims at ensemble strong TSC baseline results. The three typical methods are Elastic Ensemble (EE) [13], the collection of transformation ensembles (Flat-COTE) [14] and the hierarchical vote COTE (Hive-COTE) [15]. Specifically, EE method is an ensemble classifier with 1NN based on 11 elastic distance measures, while COTE ensembles 35 different classifiers constructed in the domain of time, frequency, change, and shapelet transformation respectively; 4) the fourth one is deep learning methods, such as Multi-Layer Perceptron(MLP), Fully Convolutional Networks(FCN) and ResNet [16]. More deep learning methods are available on [17]. Although some progress has been made, TSC tasks still suffer from the fact that practical time series inevitably exhibit time-shift invariance, high-dimensionality and complex dynamics.

Radial basis function (RBF) network, first introduced by Broomhead and Lowe [18], is a simple, efficient and interpretable tool for modeling time series. It has apply radical basis function into layered network. Benefit from the RBF's kernel, RBF networks are universal approximators. This means an RBF network with enough hidden neurons can approximate any continuous function on a closed, bounded set with arbitrary precision [19]. In this sense, RBF network is versatile, its applications include function approximation, time series prediction, classification, and nonlinear system control. Compared with other neural networks, one main advantage of RBF is the simplicity of the computation of network parameters [20]. Another one is the ability of RBF to generalize the results with high tolerance of input noises [21]. Due to these advantages, RBF network is regarded as a competitive method in contrast to multi layer perceptron (MLP) neural networks. Many existing work focus on optimizing structure [22]–[24], learning mechanism [25]–[27] of RBF networks. Moreover, in recent years, RBF networks are also successfully applied to semi-supervised learning [28] and hyperparameter optimization of deep learning algorithms [29].

However, the kernel function of RBF network has deficiencies as it is based on the Euclidean distance and improper to be used directly for time series classification (TSC). This 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 [30].

In an attempt to adapt RBF kernel to time series classification, DTW distance is a feasible alternative to Euclidean distance. The earliest work is the support vector machine (SVM) with a dynamic time-alignment kernel (DTAK) proposed by Shimodaira et al. [31]. DTAK uses inner product and kernel function to calculate the distance between two aligned points (instead of Euclidean distance) and then minimizes the accumulated distance like DTW. Bahlmann et al. directly replace the Euclidean distance with DTW in another way and propose an invariant of the gaussian radius basis function called gaussian DTW (GDTW) kernel [32]. Most recently, Xue et al. developed an altered Gaussian DTW (AGDTW) [33] kernel function which takes into consideration each of warping path between time series and 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. Therefore this kernel spends vast of time in kernel transformation and cannot be directly applied in gradient training-based RBF network. In contrast to only considering the optimal alignment, global alignment kernels (GAK) [34] uses all of the possible alignments from the cost matrix to compute the soft-minimum of all alignment costs.

In this paper, a new temporal kernel called Dynamic Barycenter Averaging Kernel (DBAK) is proposed and combined 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. [35]. Note that here the kernel center is a time series, not a vector. Furthermore, to ensure the stability of the 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.

Compared with our previous work [36], we extend the experiments to 44 datasets with training sizes smaller than or equal to 200 and also conduct experiments on the whole 85 datasets. Moreover, some non-parametric tests are conducted on these benchmark datasets to test that whether the performance of the proposed model is significantly better than existing methods. For better insight into the model, we conduct components analysis and give some interpretable examples for the new DBAK function.

We will start with preliminaries related to our work in the following section. Section 3 then introduce the the details of the proposed model DBAK-RBF. The experimental results are presented in Section 4. Finally, Section 5 provides conclusions of this paper.

II. PRELIMINARIES

In this section, we first give notation of time series in our paper. Then a brief review of Radial basis function (RBF) network, dynamic time warping (DTW), AGDTW [33] and the DTW barycenter averaging (DBA) algorithm are introduced. More details about RBF network can be found in [18], [19].

A. NOTATION OF TIME SERIES DATASET

Given a time series dataset $\mathcal{D} = \{T_1, T_2, \ldots, T_{N_D}\}$ with N_D samples, $T_i = (x_1, x_2, \ldots, x_{L_{T_i}})$, where $x_n \in \mathcal{R}^d$ $(n = 1, 2, \ldots, L_{T_i})$, L_{T_i} denotes the length of input time series and d denotes the dimension of input time series. In this paper, we focuses on the univariate time series, i.e., d=1.

B. RADIAL BASIS FUNCTION (RBF) NETWORK

Radial basis function (RBF) network, first formulated by Broomhead and Lowe [18], is a simple and effective tool in the field of mathematical modeling. 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:

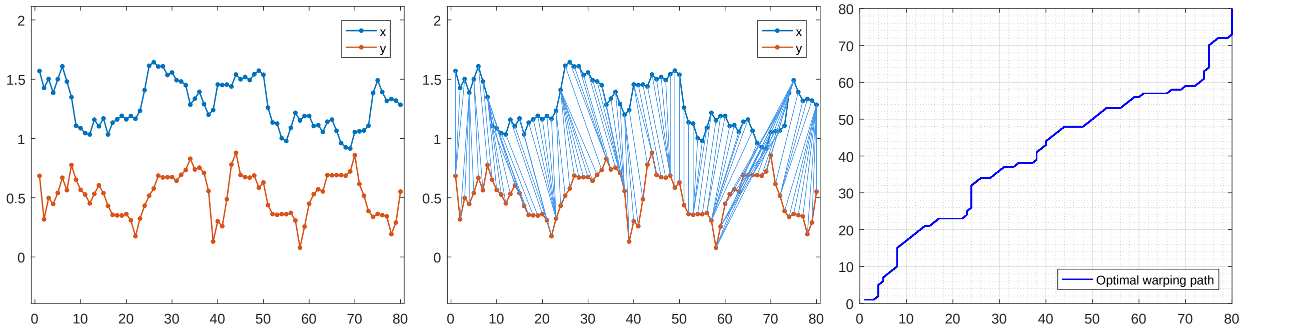
$$\phi_{i,j} = \phi(\|\boldsymbol{x}_i - \boldsymbol{c}_j\|) = \exp\left(\frac{-\|\boldsymbol{x}_i - \boldsymbol{c}_j\|^2}{\sigma_j^2}\right) \quad (1)$$

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

where $\phi_{i,j}$ is the activation value of the j-th hidden neuron, $\phi(\cdot)$ denotes the kernel function of RBF 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 units. $w_{j,k}$ is the corresponding weight and b_k is the bias term.

C. DYNAMIC TIME WARPING (DTW)

In time series analysis, Dynamic time warping (DTW) [5] is an extensively used algorithm for measuring dissimilarity be-



(a)Two original time series (b)The schematic diagram of DTW warping of $\boldsymbol{x}$ and $\boldsymbol{y}$ (c)The cost matrix and the optimal warping path

FIGURE 1: The illustration of DTW.

tween two temporal sequences. It is based on the Levenshtein distance and finds the optimal alignment paths between two sequences. Suppose there are 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,M denotes the length of two sequences. 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 a_i from sequence S and 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) monotonicity: the mapping of the indices from sequence S to indices from sequence T must be monotonically increasing, and vice versa.
3) step size condition: the warping path W satisfies $i_{k+1} - i_k \leq 1$ and $j_{k+1} - j_k \leq 1$;

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 is calculated as follows

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

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

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

To calculate minimum-distance warping path, a dynamic programming approach is used. 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 matrix Γ denotes a cumulative distance matrix. The initial conditions are

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

Fig.1 illustrates the corresponding warping relationship.

D. ALTERED GAUSSIAN DTW (AGDTW)

The GDTW kernel is constructed by introducing DTW distance into kernel mapping to replace the Euclidean distance in the Gaussian RBF kernel. Namely, Given two time series S and T defined in Eq. 3 and Eq. 4, a GDTW kernel [32] has the form

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

where σ denotes the width of the kernel function and satisfies $\sigma \neq 0$.

To take into consideration each of the warping path between time series in kernel function, an altered gaussian DTW (AGDTW) is developed. Let $\{a_{i_k}, b_{j_k}\}$ be the aligned point pair that corresponds to the time series S and T, and K denotes the length of warping path. Formally, AGDTW is defined as

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

E. DTW BARYCENTER AVERAGING (DBA)

DTW Barycenter Averaging (DBA) [35] 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 strategy of DBA is to iteratively optimize an initially (or temporary) average time series to minimize its squared distance (DTW) from averaged sequences. Technically, the DBA is divided into two steps at each iteration:

1) Calculating 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;

Algorithm 1 DBA

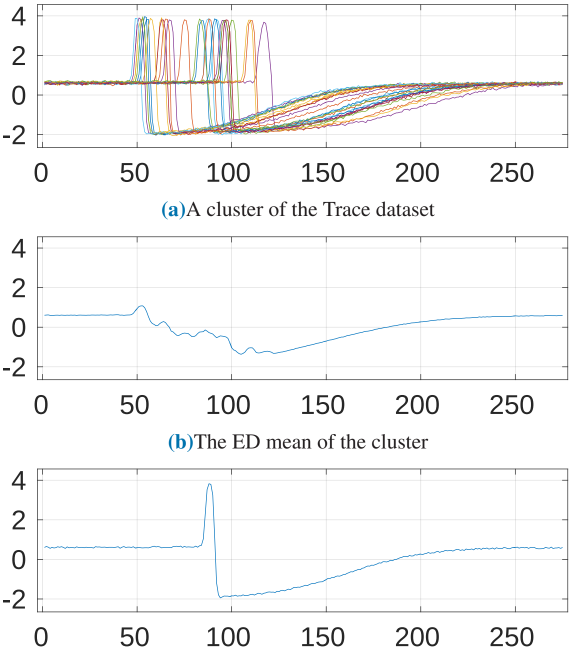
```
Require: $c$ the initial average sequence
Require: $\mathcal{S} = \{T_1, \cdots, T_N\}$ a set of $N$ sequence
Require: $Maxiter$

for iter = 1 to Maxiter do
    $\hat{c}$ is a set to store aligment for each dimension for all signal in $\mathcal{S}$
    for each $T$ in $\mathcal{S}$ do
        Compute DTW warping path $i$ and $j$ for $T$ and $c$.
          //So $T(i_k)$ and $c(j_k)$ are aligned points.
        Add all aligned $T(i_k)$ to its associated subset $\hat{c}(j_k)$

          //$\hat{c}(j_k)$ is a subset of aligned points
          //$\hat{c}$ is a set of subsets.
    end for

    //Compute new average
    for each subset $\hat{c}$(j) in $\hat{c}$ do
        c(j) = barycenter($\hat{c}$(j))
    end for
end for

return c
```

2) Updating each coordinate of the average sequence as the barycenter of coordinates associated to this coordinate during the first step;

The algorithm is shown in Alg. 1, and the source of DBA can be found in [37]. More details about DBA can be found in [35].

Figure 2 shows the examples of the average sequences obtained by ED mean and DBA respectively on one dataset from [38]. We can see from Fig. 2 that the shape of sequence obtained by DBA is very similar to the one of the input signals, while the one of ED mean eliminates the original peak of the data. It means that the DBA preserves the ability of DTW with identifying time shifts. Therefore, DBA is more suitable and accurate than ED mean in TSC problem.

In Section 3, we will propose a new temporal kernel based on the AGDTW, which enables a stable gradient-training in the RBF networks.



(a)A cluster of the Trace dataset

(b) The ED mean of the cluster

(c) The DBA of the cluster

FIGURE 2: The examples of the average sequences obtained by ED mean and DBA on one cluster from the "Trace" dataset in [38]

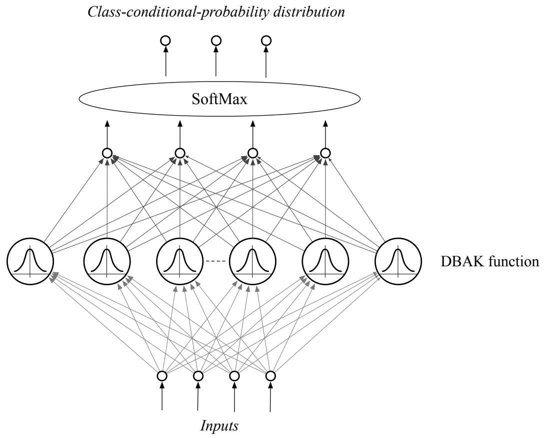


FIGURE 3: The general structure of RBF Network with DBAK.

III. DBAK-RBF NETWORK

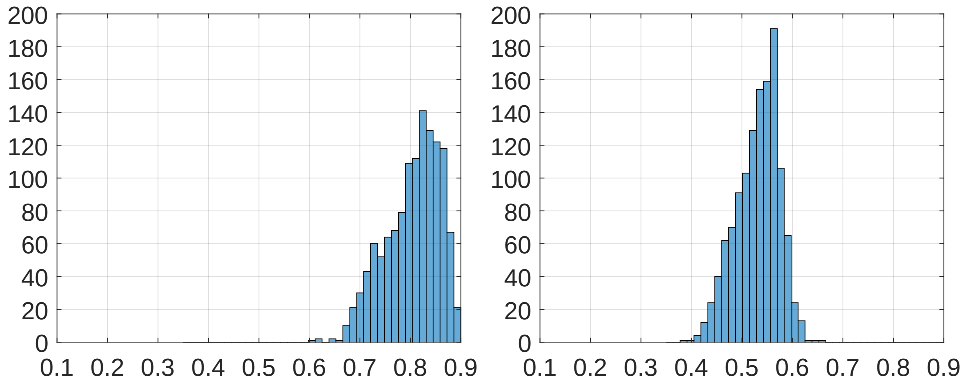
In this section, we combine the RBF network with a new dynamic barycenter averaging kernel (DBAK). We call our model DBAK-RBF for short. The architecture of the DBAK-RBF network is illustrated in Fig.3.

A. FORMULATION

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

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

where C_p denotes the center of p-th hidden neuron's activation function. $\mathcal{M} = \max\{L_{T_n}, L_{C_p}\}$. The center in our DBKA-RBF network is a time series, rather than a vector in the general RBF network. The reason behind this will be explained later. Parameter σ is the kernel parameter. K denotes the length of the warping path between time series T_n and C_p, and 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



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

(b)DBAK with $\mathcal{M}/K$

FIGURE 4: Effects of the scaling term $\mathcal{M}/K$

DBAK introduced a normalization term in Eq. 11, which can be decomposed into two parts:

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

In AGDTW, values of kernel easily turn out to be much larger due to a large K, which will hinder the gradient-training. The part $\frac{1}{K}$ is used to eliminate the effects of the optimal warping length K [39]. The other scaling term is $\frac{\mathcal{M}}{K}$, which can make the outputs of $\mathcal{K}_{DBAK}(T_n, C_p)$ tend to a standard normal distribution, as is shown in Fig. 4. After being multiplied by the first scaling term, namely, $\frac{1}{K}$, the warping path will lose information. The second scaling term, namely, $\frac{\mathcal{M}}{K}$, will compensate for their losses. Their effectiveness will be verified in later experiments.

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

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

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 that corresponds to the n-th time series sample:

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

In this way, we can use the well-known multinomial cross entropy loss to define the classification target function:

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

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

B. TRAINING

The whole process of training our DBAK-RBF consists of 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 $\{b_c\}$.

1) Step 1: Selection of Centers in DBAK

For traditional RBF networks, the kernel's centers $\{C_p\}$ are usually optimized via a clustering algorithm such as k-means. However, because such techniques rely on the Euclidean distance metric, they are not suitable for time series data and cannot be directly used in DBKA-RBF network. Using DTW is a natural way to select the centers when modeling time series data. However, this will generate much higher computation costs because we must compute the DTW distances of all sample pairs. Here, we will estimate our DBAK function's center time series $\{C_p\}$ based on the combination of the DTW-based averaging method, namely, DTW Barycenter Averaging (DBA) [35] and k-means clustering. First, we use the k-means method to roughly divide the training set into different clusters. Then, for each cluster, DBA algorithm is used to estimate an averaged time series as a center of DBAK function. This strategy is also suggested in [40].

2) Step 2: Gradient-based Optimization

After selecting 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 Eq. 11 to Eq. 15, we can easily obtain the corresponding gradient formulations.

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

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

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

where $\delta = \delta(x_{n,i_k}, c_{p,j_k})^2$, $\mathcal{M} = \max\{L_{T_n}, L_{C_p}\}$ and p denotes the width of p-th hidden neuron's activation function.

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

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

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

Algorithm 2 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 medoid of 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 $\{b_c\}$.
```

Algorithm 2 details the entire process of training our DBAK-RBF.

Generally, a learning rate should be specified for a gradient-based optimization method. And Adam [41] is a popular and practical stochastic gradient descent algorithm that is based on the estimation of 1st and 2nd-order moments. The method computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients.

There are 3 hyper-parameters in Adam, including step size α, exponential decay rates β_1 and β_2. In our experiments, we simply set $\alpha = 0.1, \beta_1 = 0.9, \beta_2 = 0.999$.

C. DISCUSSION OF COMPUTATIONAL COMPLEXITY

In this section, we discuss the computational complexity of our model. In our model, the main computational cost is the calculation of DTW distances. We use DTW for two tasks. 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 quadratic complexity of time and space. For simplicity, the complexity of DTW can be written as $\mathcal{O}(NM)$, where N and M are the lengths of the two input time series. More generally, without loss of generality, assuming that $T = \max\{N, M\}$, the time complexity can be said to be $\mathcal{O}(T^2)$. In this way, the computational complexity in the first part can be given by $\mathcal{O}(mT^2N_{iter})$, which is linear to the number of time series samples m and quadratic to the length of time series T. N_{iter} denotes the number of iteration. The complexity of the second part is $\mathcal{O}(nT^2)$, where n is the number of hidden neurons. According to these analyses, the main cost lies in the computation of DTW, and the orignal DTW in our model can be replaced by any DTW variant such as DTW with a warping window to improve the computational efficiency. For example, the time complexity of DTW with a warping width of w is $\mathcal{O}(T \times wT)$, where w is a percentage of the length of the time series. Since the typical optimal values of w are between 3% and 6%, DTW is effective linear and exact.

IV. EXPERIMENTS

This section validates the performance of our DBAK-RBF. We compare it with other mainstream TSC methods on the classification tasks for time series.

In addition, to verify the effectiveness of the components (e.g. DBAK, normalization term, the number of centers) in our model, we conduct the components analysis.

At last, we will give interpretable explanations for the new DBAK function.

To evaluate the performance, the classification accuracy can be defined by

$$\text{Accuracy} = \frac{\text{\#correct classified data}}{\text{\#testing data}} \tag{20}$$

Additional details about our experimental settings are as follows:

- The distance metric of DTW is square of the Euclidean metric, which consists of the sum of squared differences, which is given by:

$$d_{mn}(X, Y) = \sum_{k=1}^{K}(x_{k,m} - y_{k,n})^2 \tag{21}$$

- We have taken normalization before training and testing [42] [43] [44].
- There are two different ways to select the center of each unit. The first one is to select the center on the whole dataset. In other words, we directly do clustering on the whole dataset and then compute DBA for each cluster. We call them **Global centers**. The second is to divide data according to its category label and compute DBA centers for each class. We first divide data according to its category label and then do clustering for each class. That's to say, we compute DBA centers for each class. We call them **Local centers**. For the experiments on different hidden units, the kind of center is **Global centers**. For the experiments on scaling term and DBAK function, the kind is **Local centers**. For the experiments comparing with other representative TSC models, we combine both of them.
- The initial average sequence of DBA is given as the DTW Median which is shown in Alg. 3. The max number of iterations for DBA is 15 which is default in Petitjean's code [37]. According to our observation, after about 10 times of iterations, the average sequence converges.
- We employ grid search to obtain the number of hidden units. The average number of each category is in the range of [1,4].

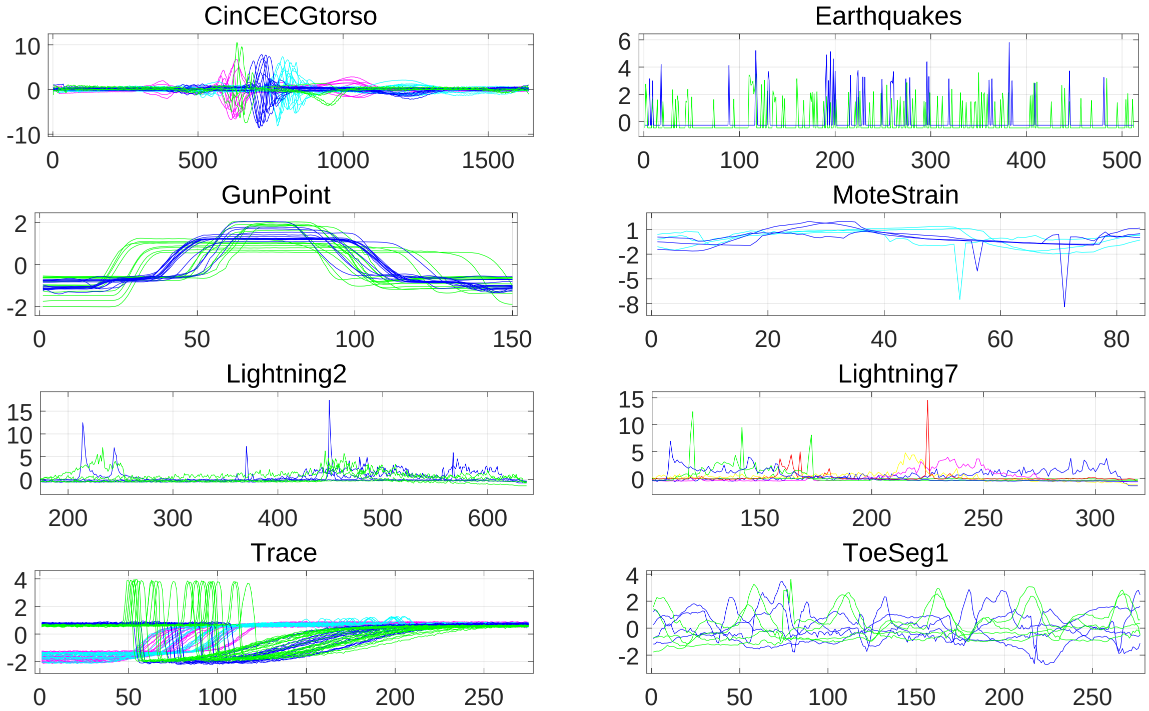


FIGURE 5: Visualization of UCR Datasets

- In addition, we conduct non-parametric tests (Wilcoxon signed-rank test [45]) in all experiments to make statistical comparisons.

A. COMPARISONS ANALYSIS

To evaluate the performance of the proposed DBAK-RBF, we perform a series of experiments on publicly available Time Series datasets from the "UCR Time Series Data Mining Archive" [38]. 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.

The 8 representative datasets are showed in Fig. 5. 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 first verify the performance of our model on 44 datasets with training size smaller than or equal to 200. Table 1 summarizes the details of the 44 datasets. We then conduct experiments on the whole 85 UCR datasets and the detailed results are in Appendix.

We select the current state-of-the-art classifiers as the compared baselines. They come from four categories we introduced in Section 1, including 1) distance-based methods; 2) feature-based methods; 3) ensemble-based methods and 4) deep learning methods.

In more details, for the distance-based methods, we compare with DTW$_{1NN}$ [6] and DD$_{DTW}$ [7]. For the feature-based methods, we select shapelet transform (ST) [8], learned shapelets (LS) [9], bag of SFA symbols (BOSS) [10], time series forest (TSF) [11] and time series bag of features (TSBF) [12]. For the ensemble-based methods, Elastic Ensemble (EE) [13] and the collection of transformation ensembles (Flat-COTE and HIVE-COTE) [14] [15] are chosen. For deep learning methods, we compare with Multi-Layer Perceptron(MLP), Fully Convolutional Networks(FCN) [46] and ResNet [16].

The results of MLP, FCN and ResNet are collected from [17], and the rest of the results come from [47].

The comparison results are shown in Table 2. As shown in Table 2, DBAK-RBF achieves much higher accuracies on some TSC tasks. For example, on the wine dataset, DBAK-RBF obtains an accuracy of 0.907, while other classifiers achieve accuracies of 0.648 by Flat-COTE, 0.778 by HIVE-COTE and 0.796 by ST. DBAK-RBF also has an accuracy of 0.883 on the Lightning2, and feature-based methods have accuracies of 0.738 (ST), 0.820 (LS), 0.836 (BOSS), 0.803 (TSF) and 0.738 (TSFB), respectively.

In addition, we conduct non-parametric tests (Wilcoxon signed-rank test with Holm's alpha correction at the 5% level, Friedman test) to perform statistical comparisons [48]. The statistic τ_F of Friedman test is 15.0208, which is larger than the critical value 1.738 (p = 0.05). Thus, the null-hypothesis (the performances of all algorithms on the groups of data are similar) is rejected.

Wilcoxon signed-rank test also compares the results of DBAK-RBF with other classification methods. The adjusted p-value under the Holm's alpha correction at the 5% level are shown in the last row of Table2. If cor-p-value is lower

TABLE 2: Accuracy of 13 mainstream TSC classifiers and our DBAK-RBF

DATASETS	DTW_{1NN}	DD_{DTW}	ST	LS	BOSS	TSF	TSBF	EE	Flat -COTE	HIVE -COTE	MLP	FCN	ResNet	DBAK-RBF
ArrowHead	0.703	0.789	0.737	0.846	0.834	0.726	0.754	0.811	0.811	**0.863**	0.778	0.843	0.845	0.750
Beef	0.633	0.667	0.900	0.867	0.800	0.767	0.567	0.633	0.867	**0.933**	0.720	0.697	0.753	0.767
BeetleFly	0.700	0.650	0.900	0.800	0.900	0.750	0.800	0.750	0.800	**0.950**	0.870	0.860	0.850	0.850
BirdChicken	0.750	0.850	0.800	0.800	0.950	0.800	0.900	0.800	0.900	0.850	0.775	**0.955**	0.885	0.900
Car	0.733	0.800	0.917	0.767	0.833	0.767	0.783	0.833	0.900	0.867	0.767	0.905	**0.925**	0.667
CBF	0.997	0.997	0.974	0.991	0.998	0.994	0.988	0.998	0.996	**0.999**	0.872	0.994	0.995	0.996
CinCECGtorso	0.651	0.725	0.954	0.870	0.887	0.983	0.712	0.942	0.995	**0.996**	0.840	0.824	0.826	0.675
Coffee	**1.000**	**1.000**	0.964	**1.000**	**1.000**	0.964	**1.000**	**1.000**	**1.000**	**1.000**	0.996	**1.000**	**1.000**	**1.000**
DiatomSizeReduction	0.967	0.967	0.925	0.980	0.931	0.931	0.899	0.944	0.928	0.941	0.910	0.313	0.301	**0.996**
DistalPhalanxOutlineCorrect	0.717	0.732	0.775	0.779	0.728	0.772	0.783	0.728	0.761	0.772	0.726	0.760	0.771	**0.860**
DistalPhalanxTW	0.590	0.612	0.662	0.626	0.676	0.669	0.676	0.647	0.698	0.683	0.617	0.690	0.665	**0.800**
Earthquakes	0.719	0.705	0.741	0.741	0.748	0.748	0.748	0.741	0.748	0.748	0.717	0.727	0.712	**0.835**
ECG200	0.770	0.830	0.830	0.880	0.870	0.870	0.840	0.880	0.880	0.850	**0.916**	0.889	0.874	0.880
ECGFiveDays	0.768	0.769	0.984	**1.000**	**1.000**	0.956	0.877	0.820	0.999	**1.000**	0.970	0.987	0.975	0.826
FaceFour	0.830	0.830	0.852	0.966	**1.000**	0.932	**1.000**	0.909	0.898	0.955	0.840	0.928	0.955	0.966
FacesUCR	0.905	0.904	0.906	0.939	0.957	0.883	0.867	0.945	0.942	**0.963**	0.833	0.946	0.955	0.824
Fish	0.823	0.943	**0.989**	0.960	**0.989**	0.794	0.834	0.966	0.983	**0.989**	0.848	0.958	0.979	0.829
GunPoint	0.907	0.980	**1.000**	**1.000**	**1.000**	0.973	0.987	0.993	**1.000**	**1.000**	0.927	**1.000**	0.991	0.927
Ham	0.467	0.476	0.686	0.667	0.667	0.743	**0.762**	0.571	0.648	0.667	0.691	0.718	0.757	0.714
Haptics	0.377	0.399	0.523	0.468	0.461	0.445	0.490	0.393	**0.523**	0.519	0.433	0.480	0.519	0.477
Herring	0.531	0.547	0.672	0.625	0.547	0.609	0.641	0.578	0.625	**0.688**	0.528	0.608	0.619	0.641
InlineSkate	0.384	**0.562**	0.373	0.438	0.516	0.376	0.385	0.460	0.495	0.500	0.337	0.339	0.373	0.307
ItalyPowerDemand	0.950	0.950	0.948	0.960	0.909	0.960	0.883	0.962	0.961	0.963	0.954	0.961	**0.963**	0.955
Lightning2	0.869	0.869	0.738	0.820	0.836	0.803	0.738	**0.885**	0.869	0.820	0.670	0.739	0.770	0.883
Lightning7	0.726	0.671	0.726	0.795	0.685	0.753	0.726	0.767	0.808	0.740	0.630	0.827	0.845	**0.863**
Mallat	0.934	0.949	0.964	0.950	0.938	0.919	0.960	0.940	0.954	0.962	0.918	0.967	0.972	**0.982**
Meat	0.933	0.933	0.850	0.733	0.900	0.933	0.933	0.933	0.917	0.933	0.897	0.853	**0.968**	0.933
MiddlePhalanxOutlineCorrect	0.698	0.732	0.794	0.780	0.780	0.828	0.814	0.784	0.804	**0.832**	0.770	0.801	0.809	0.683
MiddlePhalanxTW	0.506	0.487	0.519	0.506	0.545	0.565	0.597	0.513	0.571	0.571	0.534	0.512	0.484	**0.647**
MoteStrain	0.835	0.833	0.897	0.883	0.879	0.869	0.903	0.883	**0.937**	0.933	0.858	**0.937**	0.928	0.890
OliveOil	0.833	0.833	0.900	0.167	0.867	0.867	0.833	0.867	0.900	0.900	0.667	0.723	0.830	**0.933**
OSULeaf	0.591	0.880	0.967	0.777	0.955	0.583	0.760	0.806	0.967	**0.979**	0.557	0.977	**0.979**	0.607
Plane	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.978	**1.000**	**1.000**	**1.000**
ShapeletSim	0.650	0.611	0.956	0.950	**1.000**	0.478	0.961	0.817	0.961	**1.000**	0.503	0.724	0.779	0.672
SonyAIBORobotSurface1	0.725	0.742	0.844	0.810	0.632	0.787	0.795	0.704	0.845	0.765	0.672	**0.960**	0.958	0.933
SonyAIBORobotSurface2	0.831	0.892	0.934	0.875	0.859	0.810	0.778	0.878	0.952	0.928	0.834	**0.979**	0.978	0.845
Symbols	0.950	0.953	0.882	0.932	0.967	0.915	0.946	0.960	0.964	0.974	0.832	0.955	0.906	**0.975**
ToeSegmentation1	0.772	0.807	0.965	0.934	0.939	0.741	0.781	0.829	0.974	**0.982**	0.583	0.961	0.963	0.943
ToeSegmentation2	0.838	0.746	0.908	0.915	0.962	0.815	0.800	0.892	0.915	**0.954**	0.745	0.880	0.906	0.907
Trace	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.990	0.980	0.990	**1.000**	**1.000**	0.807	**1.000**	**1.000**	**1.000**
TwoPatterns	**1.000**	**1.000**	0.955	0.993	0.993	0.991	0.976	**1.000**	**1.000**	**1.000**	0.946	0.871	**1.000**	0.997
Wine	0.574	0.574	0.796	0.500	0.741	0.630	0.611	0.574	0.648	0.778	0.565	0.587	0.744	**0.907**
Worms	0.584	0.584	0.740	0.610	0.558	0.610	0.688	0.662	0.623	0.558	0.457	0.765	**0.791**	0.624
WormsTwoClass	0.623	0.649	**0.831**	0.727	**0.831**	0.623	0.753	0.688	0.805	0.779	0.601	0.726	0.747	0.735
Avg.Rank	10.784	9.534	6.841	7.398	6.375	9.045	8.000	7.739	4.909	**4.045**	11.523	6.568	5.943	6.295
p-value	0.000	0.006	0.754	0.628	0.655	0.003	0.142	0.273	0.230	0.138	0.000	0.791	0.253	-
cor-p-value	0.000	0.060	1.965	2.512	2.512	0.033	1.242	1.518	1.610	1.242	0.000	1.508	1.610	-

than 0.05, the null-hypothesis (the performance of the two algorithms on the groups of data is similar) is rejected. In this sense, we can conclude that although our model ranks fourth but it has no statistically significant difference with HIVE-COTE, Flat-COTE, FCN and ResNet on these 44 datasets, and it significantly outperforms the rest of the compared models.

Furthermore, although the results of our model on the whole 85 UCR datasets in Appendix can not beat FCN, Flat-COTE, HIVE-COTE and ResNet, it has no statistically significant difference with other methods including some ensemble models e.g. EE and BOSS.

We must emphasize here that although the deep learning methods and ensemble methods outperform most of the other models, there are specific problems with each method. For example, as a representative ensemble-based method, COTE will suffer from high computational complexity due to employing 35 different classifiers. And the deep learning methods inevitably have the overfitting problem on small datasets. This point has been verified to some extent by

TABLE 1: The details of 44 selected UCR datasets

DATASETS	♯ CLASSES	♯ TRAIN	♯ TEST	LENGTH
ArrowHead	3	36	175	251
Beef	5	30	30	470
BeetleFly	2	20	20	512
BirdChicken	2	20	20	512
Car	4	60	60	577
CBF	3	30	900	128
CinCECGtorso	4	40	1380	1639
Coffee	2	28	28	286
DiatomSizeReduction	4	16	306	345
DistalPhalanxOutlineCorrect	3	139	400	80
DistalPhalanxTW	6	139	400	80
Earthquakes	2	139	322	512
ECG200	2	100	100	96
ECGFiveDays	2	23	861	136
FaceFour	4	24	88	350
FacesUCR	14	200	2050	131
Fish	7	175	175	463
GunPoint	2	50	150	150
Ham	2	109	105	431
Haptics	5	155	308	1092
Herring	2	64	64	512
InlineSkate	7	100	550	1882
ItalyPowerDemand	2	67	1029	24
Lightning2	2	60	61	637
Lightning7	7	70	73	319
Mallat	8	55	2345	1024
Meat	3	60	60	448
MiddlePhalanxOutlineCorrect	3	154	400	80
MiddlePhalanxTW	6	154	399	80
MoteStrain	2	20	1252	84
OliveOil	4	30	30	570
OSULeaf	6	200	242	427
Plane	7	105	105	144
ShapeletSim	2	20	180	500
SonyAIBORobotSurface1	2	20	601	70
SonyAIBORobotSurface2	2	27	953	65
Symbols	6	25	995	398
ToeSegmentation1	2	40	228	277
ToeSegmentation2	2	36	130	343
Trace	4	100	100	275
TwoPatterns	2	23	1139	82
Wine	2	57	54	234
Worms	5	77	181	900
WormsTwoClass	2	77	181	900

the experiments on 44 datasets where our single model is no statistical significant difference with FCN and ResNet. Moreover, FCN has 5 layers and ResNet has 11 layers, while DBAK-RBF only has one hidden layer and is a single elegant model that we test as a stand-alone classifier. That means our model has many fewer parameters to learn.

B. IMPACT OF THE DBAK FUNCTION

The DBAK is a novel DTW-based kernel for RBF network for dealing with time series data. For components analysis, we firstly compare DBAK-RBF with a regular RBF network on 85 UCR datasets.

By regarding an input time series as an L_{T_i}-dimension input vector, we allow RBF network to be directly applied in TSC task. For the number of hidden units, only one center is selected for each class.

For each of the UCR Datasets, we obtain the accuracies of the two models. We combine these two accuracies as a point then plot them together. In Fig.6a, the straight line represents equal accuracy of the two models. The points under the line represent the datasets on which the DBAK-RBF obtained a higher accuracy. It verifies that our DBAK outperforms the original RBF kernel based on Euclidean distance(ED) (Wins/Ties/Losses are 59/0/26 in Fig.6a). The p-value of Wilcoxon signed-rank test is 0.000038, which is smaller than the critical value 0.05. Thus, the null-hypothesis (the two algorithms have the same performance) is rejected. Therefore, with the advantage of DTW, DBAK is more suitable for time series, and it can also be applied in data that has different lengths.

C. SUPERIORITY OF DBA AVERAGE

In most cases, time series problems suffer from time-shift invariance, high-dimensionality, and complex dynamics. No specific meaning is associated with a specific dimension. Hence, it is unreasonable to directly employ ED-Mean as an average center.We conduct an experiment to verify this. We use the k-means algorithm that is based on ED to select centers in our DBAK-RBF, other than using the DBA. In other words, the means that are obtained from the k-means clustering process are set as the centers and each class has only one selected center. The result in Fig.6b demonstrate that our DBAK outperforms ED-based k-means. The p-value of Wilcoxon signed-rank test is 0.00000089, which is smaller than the critical value 0.05. Thus, the null-hypothesis (the two algorithms have the same performance) rejected. It verifies that our DBAK is significantly outperforms the case of using ED-based k-means.

Furthermore, the DBA average is compared with the DTW median to determine whether the DBA average is more suitable than the DTW median under the DTW distance. Here,we compute DTW Median after clustering. The DTW Median is the median using DTW to compute the distance and the detail can be found in Alg. 3.

The p-value of Wilcoxon signed-rank test is 0.0000013, which is smaller than the critical value 0.05. Thus, the null-hypothesis (the two algorithms have the same performance) is rejected. Fig. 6c shows the results comparing DTW median and DBA in DBAK. DBAK-RBF with DBA outperforms the DTW median.

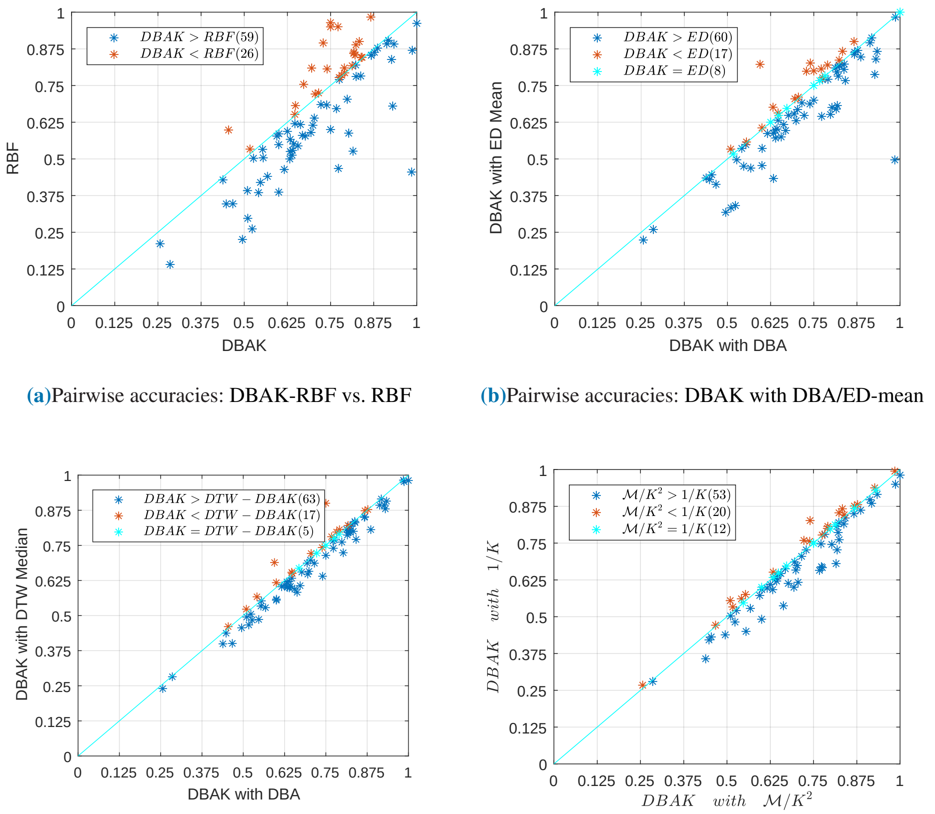
Algorithm 3 DTW Median

```
Require: Data: Data to compute DTW Median

Compute the sum of distances with other signals:
    $d_i = \sum_{j=1}^{N} dist(i, j)$
$k = \arg\min_{i} d_i$

return k-th signals of Data
```

D. INTERPRETABLE EXAMPLES OF NORMALIZATION TERM

In Section III, we stated that the factor $\frac{1}{K}$ is used to eliminate the effects of the warping length K to make the model trainable but leads to the loss of warping path information. Therefore, we add a $\frac{\mathcal{M}}{K}$ term to make up for these losses.

Assume we have the warping path with a length of more than 200 and the activation value that is computed via Eq.



(a)Pairwise accuracies: DBAK-RBF vs. RBF

(b)Pairwise accuracies: DBAK with DBA/ED-mean

(c)Pairwise accuracies: DBAK with DTW Median/DBA

(d)Pairwise accuracies: DBAK with $\mathcal{M}/K^2$ vs. DBAK with $1/K$

FIGURE 6: Accuracy pairwise plots on 85 datasets(the number of Wins/Ties/Losses is between parentheses in the legend)

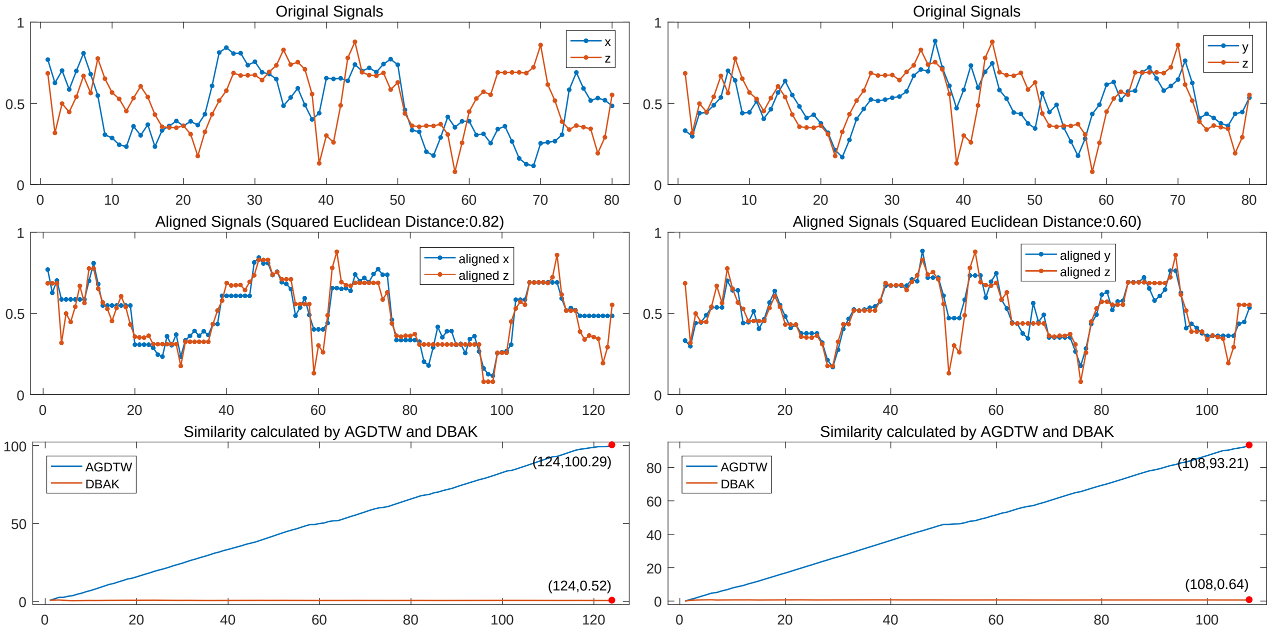


FIGURE 7: There are 3 signals:x, y, z; x, y is the blue signal in the left and right respectively, and the third one is labeled as z plotted in red in both left and right; In the first column, the first row is the signal of x and z. The second row is the aligned signal and its DTW distance. The third row is the similarity calculated by AGDTW and DBAK with respect to the length of the warping path and the same center signal z; The second column is similar to the first one.

10 is 200. In the following forward-propagation step, we obtain a number that is equal to e^{200}, which may lead to the **inf/NaN** Error and hinder the training process. Therefore the normalization term is necessary.

We investigate the performance of our normalization term, namely, $\mathcal{M}/K^2$ in Eq. 11 as an example to explore the effect of normalization term. The DBAK with $\mathcal{M}/K^2$ and the case with $1/K$ is compared in Fig.6d. The p-value of Wilcoxon signed-rank test is 0.000038, which is smaller than the critical value 0.05. Thus, the null-hypothesis (the two algorithms have the same performance) is rejected. It verifies that the scaling term $\mathcal{M}/K$ significantly improves the performance on time series classification compared with the model with $1/K$ only.

Furthermore, we visualize the activation computed by Eq. 10 and Eq. 11 in Fig. 7. Intuitively, if the distance between two signals is very large, their similarity will be small. As shown in Fig. 7, the DTW distance is given as:

$$d_{dtw}(\boldsymbol{x}, \boldsymbol{z}) = 0.82 \tag{22}$$

$$d_{dtw}(\boldsymbol{y}, \boldsymbol{z}) = 0.60 \tag{23}$$

and the final similarity is given as:

$$K_{AGDTW}(\boldsymbol{x}, \boldsymbol{z}) = 100.29 \tag{24}$$

$$K_{AGDTW}(\boldsymbol{y}, \boldsymbol{z}) = 93.21 \tag{25}$$

$$K_{DBAK}(\boldsymbol{x}, \boldsymbol{z}) = 0.52 \tag{26}$$

$$K_{DBAK}(\boldsymbol{y}, \boldsymbol{z}) = 0.64 \tag{27}$$

Since $d_{dtw}(\boldsymbol{x}, \boldsymbol{z}) > d_{dtw}(\boldsymbol{y}, \boldsymbol{z})$, $K(\boldsymbol{x}, \boldsymbol{z}) < K(\boldsymbol{y}, \boldsymbol{z})$ should be satisfied. We can see that DBAK satisfies this inequality. However, AGDTW does not. In contrast, the warping length reflects the distortion. The results in the last row of Fig. 7 demonstrates that with the growth of the warping length in the process of dynamic programming, the similarity computed via Eq. 10(AGDTW) will also increase, while the similarity from Eq. 11(DBAK) is stable in the range from 0 to 1. DBAK outperforms AGDTW since it is unreasonable for the similarity to increase when the warping path becomes longer.

E. EFFECT OF THE WEIGHT INITIALIZATIONS

The effect of the initial weight values has received a significant amount of interest from many researchers in the field [17]. Therefore, we conduct experiments to explore whether the random initialization will impact the performance of DBAK or not.

We conduct experiments on 44 datasets, and the average accuracies and the standard deviations for 10 runs by random weight initialization are shown in Fig. 1 in Appendix. As we can see from it, the performance of DBAK is stable and does not suffer from significant impacts of different initial weight values.

F. EXPERIMENTS ON THE DIFFERENT NUMBER OF HIDDEN UNITS

At last, the number of centers will be tested which is shown in Fig. 8. For each dataset, we choose 20 numbers evenly over the length of time series as the number of centers. The performance of DBAK is stable, instead of that working better in a specific size of the network. We can also find out that a small number of centers is enough for the model to reach the stable accuracy, which is consistent with [40].

V. CONCLUSIONS AND FUTURE WORK

In this paper, a new DTW-based kernel, named Dynamic barycenter averaging kernel (DBAK), has been proposed in RBF network to solve the time series classification problem. The main characteristics of the DBAK-based RBF network include : 1) a two-step k-means clustering based on a DTW-Barycenter-Averaging (DBA) algorithm is used to determine the centers of RBF network's kernel function. 2) a normalization term is added to the kernel formulation to facilitate the stable gradient-training process. It retains advantages of the original RBF and performs efficiently for temporal data. Experimental results indicate that DBAK-RBF outperforms most of the distance-based methods and feature-based methods, and has no statistically significant difference with ensemble-based methods and deep learning methods on 44 datasets. Moreover, even on the whole 85 UCR datasets shown in Appendix, DBAK-RBF is a competitive model that has no statistically significant difference with some ensemble models e.g. EE and BOSS.

In the future, DBAK-RBF could be integrated into other methods and extended to other areas as a variant of RBF network. The scaled term we use to normalize the proposed kernel could be also applied to alleviate the overfitting problem in deep neural networks. Moreover, regularization techniques specific for neural networks such as data augmentation [49] could be employed to our model.

ACKNOWLEDGMENT

First of all, we would like to express my sincere heartfelt thanks to anonymous reviewers for their valuable advice to improve this paper. We appreciate F. Petitjean for the code of DBA. Then we would like to show our gratitude to Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen and G. Batista for their work on creating and maintaining all the UCR data. The work described in this paper was funded by the National Natural Science Foundation of China (Grant No. 61502174, 61872148), the Natural Science Foundation of Guangdong Province (Grant No. 2017A030313355, 2017A030313358), the Guangzhou Science and Technology Planning Project (Grant No. 201704030051), the National Training Program of Innovation and Entrepreneurship for Undergraduates (201810561117), and the Opening Project of Guangdong Province Key Laboratory of Big Data Analysis and Processing (Grant No. 2017014).

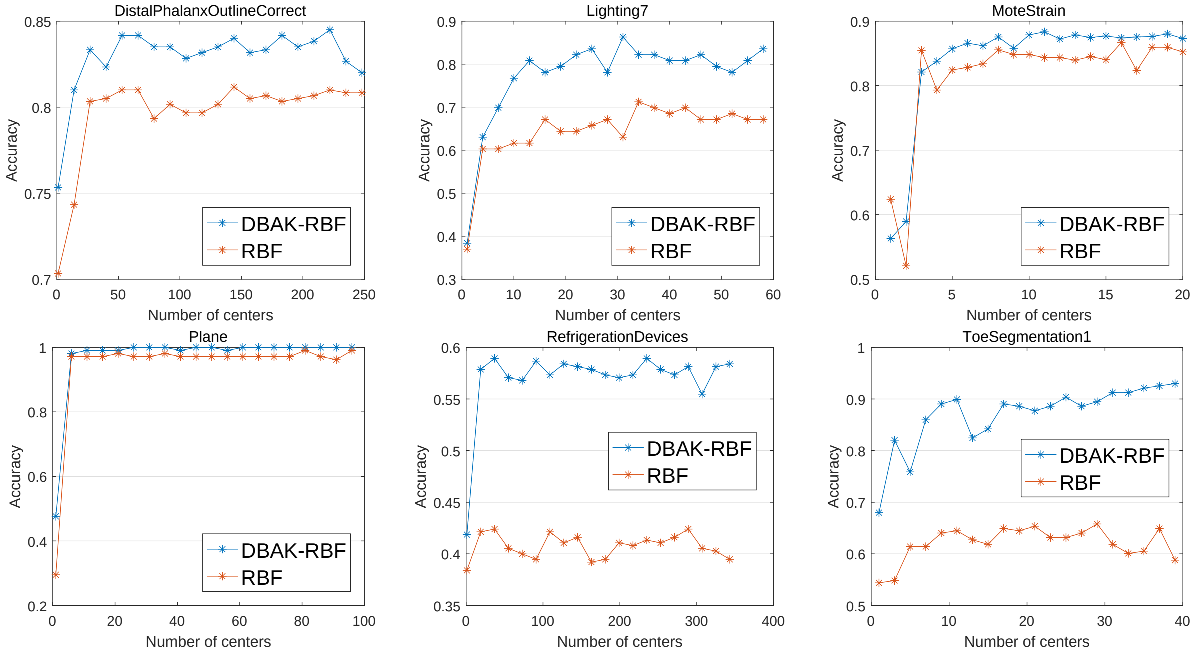


FIGURE 8: The performance of DBAK-RBF and RBF with different number of centers

REFERENCES

[1] F. Petitjean, J. Inglada, and P. Gançarski, “Satellite image time series analysis under time warping,” IEEE transactions on geoscience and remote sensing, vol. 50, no. 8, pp. 3081–3095, 2012.

[2] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, “Experimental comparison of representation methods and distance measures for time series data,” Data Mining and Knowledge Discovery, vol. 26, no. 2, pp. 275–309, 2013.

[3] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, “Time series classification using multi-channels deep convolutional neural networks,” in International Conference on Web-Age Information Management. Springer, 2014, pp. 298–310.

[4] L.-Y. Wei, “A hybrid anfis model based on empirical mode decomposition for stock time series forecasting,” Applied Soft Computing, vol. 42, pp. 368–376, 2016.

[5] D. J. Berndt, “Using dynamic time warping to find patterns in time series,” in Kdd Workshop, 1994, pp. 359–370.

[6] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, “Querying and mining of time series data: Experimental comparison of representations and distance measures,” Proc. VLDB Endow., vol. 1, no. 2, pp. 1542–1552, Aug. 2008.

[7] T. Górecki and M. Łuczak, “Using derivatives in time series classification,” Data Min. Knowl. Discov., vol. 26, no. 2, pp. 310–331, Mar. 2013.

[8] J. Hills, J. Lines, E. Baranauskas, J. Mapp, and A. Bagnall, “Classification of time series by shapelet transformation,” Data Min. Knowl. Discov., vol. 28, no. 4, pp. 851–881, Jul. 2014.

[9] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, “Learning time-series shapelets,” in Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD ’14. New York, NY, USA: ACM, 2014, pp. 392–401.

[10] P. Schfer, “The boss is concerned with time series classification in the presence of noise,” Data Mining & Knowledge Discovery, vol. 29, no. 6, pp. 1505–1530, 2015.

[11] H. Deng, G. Runger, E. Tuv, and M. Vladimir, “A time series forest for classification and feature extraction,” Information Sciences, vol. 239, pp. 142–153, 2013.

[12] M. G. Baydogan, G. Runger, and E. Tuv, “A bag-of-features framework to classify time series,” IEEE transactions on pattern analysis and machine intelligence, vol. 35, no. 11, pp. 2796–2802, 2013.

[13] J. Lines and A. Bagnall, “Time series classification with ensembles of elastic distance measures,” Data Mining and Knowledge Discovery, vol. 29, no. 3, pp. 565–592, 2015.

[14] A. Bagnall, J. Lines, J. Hills, and A. Bostrom, “Time-series classification with cote: The collective of transformation-based ensembles,” IEEE Transactions on Knowledge and Data Engineering, vol. 27, no. 9, pp. 2522–2535, Sept 2015.

[15] J. Lines, S. Taylor, and A. Bagnall, “Hive-cote: The hierarchical vote collective of transformation-based ensembles for time series classification,” in IEEE International Conference on Data Mining, 2016.

[16] Z. Wang, W. Yan, and T. Oates, “Time series classification from scratch with deep neural networks: A strong baseline,” in 2017 International Joint Conference on Neural Networks (IJCNN). IEEE, 2017, pp. 1578–1585.

[17] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, “Deep learning for time series classification: a review,” arXiv preprint arXiv:1809.04356, 2018.

[18] D. S. Broomhead and D. Lowe, “Multivariable functional interpolation and adaptive networks,” Complex Systems, vol. 2, no. 3, pp. 321–355, 1988.

[19] J. Park and I. W. Sandberg, “Universal approximation using radial-basis-function networks,” Neural Computation, vol. 3, no. 2, pp. 246–257, June 1991.

[20] S. Chen, C. F. Cowan, and P. M. Grant, “Orthogonal least squares learning algorithm for radial basis function networks,” IEEE Transactions on neural networks, vol. 2, no. 2, pp. 302–309, 1991.

[21] H. Yu, T. Xie, S. Paszczynski, and B. M. Wilamowski, “Advantages of radial basis function networks for dynamic system design,” IEEE Transactions on Industrial Electronics, vol. 58, no. 12, pp. 5438–5450, 2011.

[22] H.-G. Han, Q.-l. Chen, and J.-F. Qiao, “An efficient self-organizing rbf neural network for water quality prediction,” Neural networks, vol. 24, no. 7, pp. 717–725, 2011.

[23] L. J. Herrera, H. Pomares, I. Rojas, A. Guillén, G. Rubio, and J. Urquiza, “Global and local modelling in rbf networks,” Neurocomputing, vol. 74, no. 16, pp. 2594–2602, 2011.

[24] J. Lu, H. Hu, and Y. Bai, “Generalized radial basis function neural network based on an improved dynamic particle swarm optimization and adaboost algorithm,” Neurocomputing, vol. 152, pp. 305–315, 2015.

[25] G. A. Montazer, H. Khoshniat, and V. Fathi, “Improvement of rbf neural networks using fuzzy-osd algorithm in an online radar pulse classification system,” Applied Soft Computing, vol. 13, no. 9, pp. 3831–3838, 2013.

[26] A. Alexandridis, E. Chondrodima, N. Giannopoulos, and H. Sarimveis, “A fast and efficient method for training categorical radial basis function networks,” IEEE transactions on neural networks and learning systems, vol. 28, no. 11, pp. 2831–2836, 2017.

[27] I. Aljarah, H. Faris, S. Mirjalili, and N. Al-Madi, “Training radial basis

function networks using biogeography-based optimizer," Neural Computing and Applications, vol. 29, no. 7, pp. 529–553, 2018.

[28] N. Pitelis, C. Russell, and L. Agapito, "Semi-supervised learning using an unsupervised atlas," in Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML). Springer, 2014, pp. 565–580.

[29] I. Ilievski, T. Akhtar, J. Feng, and C. A. Shoemaker, "Efficient hyperparameter optimization for deep learning algorithms using deterministic rbf surrogates." in AAAI, 2017, pp. 822–829.

[30] T. C. Fu, "A review on time series data mining," Engineering Applications of Artificial Intelligence, vol. 24, no. 1, pp. 164–181, 2011.

[31] H. Shimodaira, K.-i. Noma, M. Nakai, and S. Sagayama, "Dynamic time-alignment kernel in support vector machine," in Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, ser. NIPS'01. Cambridge, MA, USA: MIT Press, 2001, pp. 921–928.

[32] C. Bahlmann, B. Haasdonk, and H. Burkhardt, "Online handwriting recognition with support vector machines - a kernel approach," in Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition, 2002, pp. 49–54.

[33] Y. Xue, L. Zhang, Z. Tao, B. Wang, and F. Li, "An altered kernel transformation for time series classification," in Neural Information Processing, D. Liu, S. Xie, Y. Li, D. Zhao, and E.-S. M. El-Alfy, Eds. Cham: Springer International Publishing, 2017, pp. 455–465.

[34] M. Cuturi, "Fast global alignment kernels," in Proceedings of the 28th international conference on machine learning (ICML-11), 2011, pp. 929–936.

[35] F. Petitjean, A. Ketterlin, and P. Ganarski, "A global averaging method for dynamic time warping, with applications to clustering," Pattern Recognition, vol. 44, no. 3, pp. 678 – 693, 2011.

[36] H. Qin, L. Shen, C. Sima, and Q. Ma, "Rbf networks with dynamic barycenter averaging kernel for time series classification," in International CCF Conference on Artificial Intelligence, 2018, pp. 139–152.

[37] Petitjean, "Source code for averaging for dynamic time warping," 2017, https://github.com/fpetitjean/DBA.

[38] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista, "The ucr time series classification archive," July 2015.

[39] C. A. Ratanamahatana and E. Keogh, "Everything you know about dynamic time warping is wrong," in Workshop on Mining Temporal and Sequential Data, 2004.

[40] F. Petitjean, G. Forestier, G. I. Webb, A. E. Nicholson, Y. Chen, and E. Keogh, "Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm," Knowledge and Information Systems, vol. 47, no. 1, pp. 1–26, 2016.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," CoRR, vol. abs/1412.6980, 2014.

[42] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," Journal of computational science, vol. 2, no. 1, pp. 1–8, 2011.

[43] J. Bollen, H. Mao, and A. Pepe, "Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena." Icwsm, vol. 11, pp. 450–453, 2011.

[44] J. Ma, W. Gao, Z. Wei, Y. Lu, and K.-F. Wong, "Detect rumors using time series of social context information on microblogging websites," in Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. ACM, 2015, pp. 1751–1754.

[45] A. Benavoli, G. Corani, and F. Mangili, "Should we really use post-hoc tests based on mean-ranks?" The Journal of Machine Learning Research, vol. 17, no. 1, pp. 152–161, 2016.

[46] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[47] A. Bagnall, A. Bostrom, J. Large, and J. Lines, "The great time series classification bake off: An experimental evaluation of recently proposed algorithms. extended version," arXiv preprint arXiv:1602.01711, 2016.

[48] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," vol. 7, pp. 1–30, 01 2006.

[49] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Data augmentation using synthetic data for time series classification with deep residual networks," arXiv preprint arXiv:1808.02455, 2018.

• • •