IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 20, 2019, accepted September 4, 2019, date of publication September 24, 2019, date of current version October 7, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2943474

Learning Kullback-Leibler Divergence-Based Gaussian Model for Multivariate Time Series Classification

GONGQING WU[1,2], (Member, IEEE), HUICHENG ZHANG[1,2], YING HE[1,2], XIANYU BAO[3], LEI LI[1,2], (Senior Member, IEEE), AND XUEGANG HU[1,2]

[1]Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology, Ministry of Education, Hefei 230601, China
[2]School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230601, China
[3]Shenzhen Academy of Inspection and Quarantine, Shenzhen 518045, China

Corresponding author: Gongqing Wu (wugq@hfut.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFC1603601, in part by the Program for Innovative Research Team in University of the Ministry of Education under Grant IRT_17R32, and in part by the National Natural Science Foundation of China under Grant 61673152 and Grant 91746209.

ABSTRACT The multivariate time series (MTS) classification is an important classification problem in which data has the temporal attribute. Because relationships between many variables of the MTS are complex and time-varying, existing methods perform not well in MTS classification with many attribute variables. Thus, in this paper, we propose a novel model-based classification method, called Kullback-Leibler Divergence-based Gaussian Model Classification (KLD-GMC), which converts the original MTS data into two important parameters of the multivariate Gaussian model: the mean vector and the inverse covariance matrix. The inverse covariance is the most important parameter, which can obtain the information between the variables. So that the more variables, the more information could be obtained by the inverse covariance, KLD-GMC can deal with the relationship between variables well in the MTS. Then the sparse inverse covariance of each subsequence is solved by Graphical Lasso. Furthermore, the Kullback-Leibler divergence is used as the similarity measurement to implement the classification of unlabeled subsequences, because it can effectively measure the similarity between different distributions. Experimental results on classical MTS datasets demonstrate that our method can improve the performance of multivariate time series classification and outperform the state-of-the-art methods.

INDEX TERMS Kullback-Leibler divergence, Gaussian model, graphical lasso, multivariate time series, classification.

I. INTRODUCTION

With the development of the internet of things (IoT), big data and artificial intelligence technology, the number of time series data has increased explosively, which makes time series classification (TSC) become one of the most challenging problems in machine learning and data mining. Essentially, any classification problem can be converted to a TSC problem when the data has the temporal attribute. TSC is widely used in areas such as disease diagnosis [1], human activity recognition [2], acoustic scene classification [3], and network security [4]. Researchers focused on univariate time series (UTS) classification in the early stage, and there are at least hundreds of papers worked on this issue at present [5]–[8]. Since a UTS is data about a variable that describes only one aspect of the objects and may not satisfy most of the application domain, recently, researchers have paid more attention to the multivariate time series (MTS) classification. An MTS contains a set of ordered observations at discrete time for multiple variables, and it can be viewed as a collection of multiple UTS. However, if an MTS instance is broken into several UTSs, the correlations among the variables will be lost. This is why the MTS classification is more challenging: the relationships between the variables of the MTS instances are complex and time-varying.

MTS classification methods can be divided into four categories: the distance-based method [9], the feature-based method [10], the deep learning-based method [11], and the

The associate editor coordinating the review of this manuscript and approving it for publication was Tossapon Boongoen.

model-based method. The distance-based method predicts the test instances' categories based on the similarity between the test instances and the training instances. The feature-based method relies on extracting features from original MTS data, and then builds models based on those features. The deep learning-based method automatically learns the characteristics of the instances by constructing a neural network structure to achieve the aim of classification. The model-based method converts the original MTS instances into model parameters to build the corresponding model for classification. It is worth noting that the model-based method, which makes use of the statistical characteristics of data, is more informative and interpretable than the former three ones, and has been paid more and more attention by researchers. This paper mainly studies the model-based MTS classification method, aiming to propose an accurate MTS classification approach, which can handle multivariate sequences with variable length and phase.

In this paper, we propose a novel model-based MTS classification method, called Kullback-Leibler Divergence-based Gaussian Model Classification (KLD-GMC). KLD-GMC assumes that the MTS data obeys the Gaussian distribution, which is clearly defined as the model used for classification, and then solves the model parameters for MTS classification. Essentially, these model parameters are the features used to discriminate time series. The model parameters used by KLD-GMC to discriminate MTS features are mean and sparse inverse covariance, and they constitute a multivariate Gaussian model. In comparison, the sparse inverse covariance parameter is more important than the mean parameter. On the one hand, the inverse covariance parameter maps the MTS to a fixed-size vector space independent of the sequence length; and sparse graphical representation is a useful method to prevent overfitting [12]; In addition, the sparse inverse covariance represents the conditional independent structure between variables [13], which provides an interpretable insight of the classification results. On the other hand, a significant advantage of KLD-GMC is that it is very suitable for processing high-dimensional MTS data compared to other classification methods. Since the inverse covariance can measure the relationship between variables in the MTS well. The more variables in the MTS, the stronger the ability of the sparse inverse covariance to characterize the MTS, so that our method can achieve a better performance. Furthermore, the comparative experiments were conducted on several MTS datasets to show the performance of our proposed method by comparing with the state-of-the-art methods.

The contributions of this paper can be summarized as follows:

- As compared to existing MTS classification methods, our method can make full use of the information between variables by the covariance.
- We derive the calculation method of the Kullback-Leibler divergence between multivariate Gaussian models (as shown in Equation 7, Section III.*C*), and use the Kullback-Leibler divergence as a measure of similarity between two subsequences.
- In the experiments, we show that our method does improve the performance of MTS classification with many variables.

The rest of this paper is organized as follows. Section II introduces some related work. Section III describes the KLD-GMC algorithm in detail, including the introduction of Kullback-Leibler divergence, the solution of sparse inverse covariance and the Gaussian model classification process based on Kullback-Leibler divergence. Section IV gives experimental results on several MTS datasets to demonstrate the effectiveness of the proposed algorithm. Finally, the conclusion is given in Section V.

II. RELATED WORK

As mentioned above, the existing MTS classification methods can be roughly divided into four categories: the distance-based method, the feature-based method, the deep learning-based method, and the model-based method.

The distance-based method primarily studies the similarity measures between sequences, and then predicts a class label based on similarities between the instance to be predicted and training instances. There are various distance measurements can be used for MTS comparison, including the Euclidean distance [14], the short-term sequence distance [15], the probability-based distance function [16], the dynamic time warping (DTW) distance [17], and various variants [18]. In these works, the Euclidean distance and the short time series sequence distance are suitable for more uniformly sampled MTSs, since both require MTS to have the same phase. The probability-based distance function treats an MTS as a probability distribution, but after nonlinear transformation, there has a greater difference between the probability distributions of the two MTSs. The dynamic time warping (DTW) distance is the most famous method of distance similarity measures. It is good at finding the optimal alignment between two nonuniform time series. DTW and its variants have become standard benchmarks for distance-based methods. It is worth noting that the existing distance-based MTS classification methods often have low computational efficiency, because the similarity measurement between sequences and the classification process of KNN or SVM classifiers will cause expensive computational costs.

The feature-based method transforms time series into feature vectors, relying on extracting features from original MTS data, then constructing models on time features and classifying them by traditional classifiers. Typical feature-based methods are two-dimensional singular value decomposition (2dSVD) [19], unsupervised locality preserving projections (LPP) [20], symbolic representation for MTS (SMTS) [21], Shapelets [22] and its various variants [23]. 2dSVD captures eigenvectors of covariance matrices of MTS data as features, and calculates the distance between two MTSs by measuring the distance of these features. LPP projects the feature vectors extracted by 2dSVD into a lower-dimensional feature space,

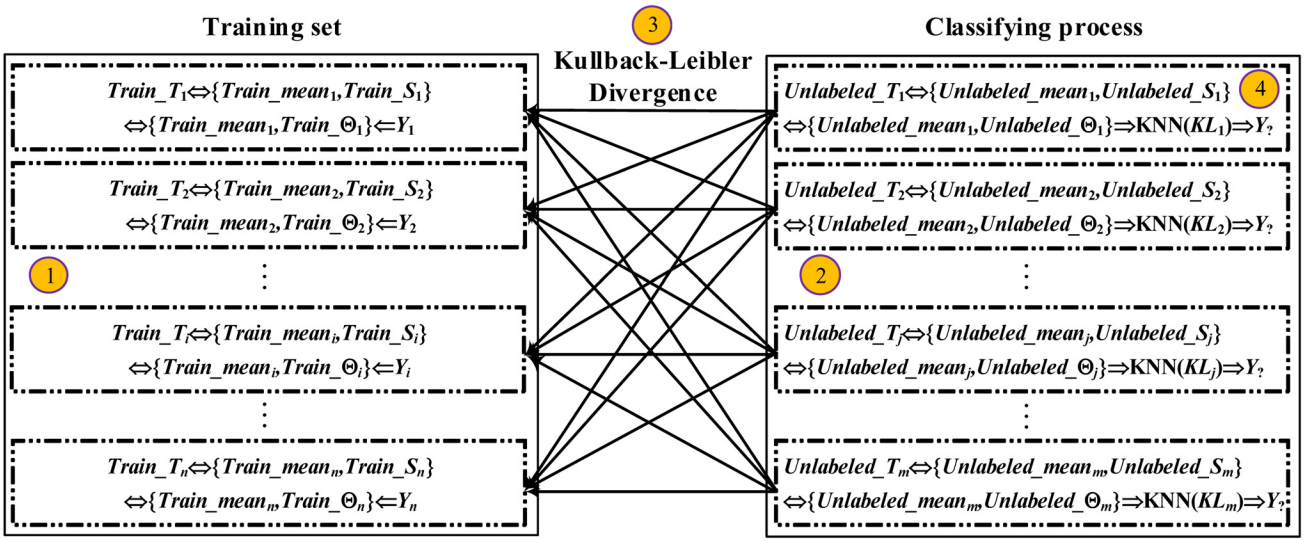

FIGURE 1. The framework of the KLD-GMC method.

in which the related MTS samples of the same class are close to each other. 2dSVD and LPP are not robust enough because they are sensitive to noise and outliers. In order to utilize the information of the sample class label, researchers have proposed a method with supervised learning characteristics. For example, symbolic representation for MTS (SMTS), SMTS considers all the attributes of the MTS simultaneously, rather than separately, to extract the information contained in the relationship; Ye and Keogh propose the time series original representation method called Shapelets [22], and other researchers propose the latest improved algorithm of Shapelets, but the discovery process of Shapelets and its variants is computationally expensive.

In recent years, with successful applying of deep neural networks (DNN) in various fields, more and more researchers applied deep learning-based methods to classify MTS. The three main DNN architectures for TSC tasks are: Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), and Echo State Network (ESN). MLP is the simplest and most traditional architecture of the deep learning model, including Multi-Layer Perceptron (MLP) [24]; CNN architectures include Fully Convolutional Neural Network (FCNN) [24], Residual Network (RN) [24], Encoder [11], Multi-scale Convolutional Neural Network (MCNN) [11], Time Le-Net (t-LeNet) [25], Multi-Channel Deep Convolutional Neural Network (MCDCNN) [26], Time Convolutional Neural Network (Time-CNN) [27]; ESN architectures include Time Warping Invariant Echo State Network (TWIESN) [28]. These deep learning-based methods have their own advantages and disadvantages for the MTS classification. For more details on time series classification based on deep learning methods, we guide interested readers to a recent empirical study [11].

The model-based method assumes that time series in a class are generated by the same model, and the same category of data can be characterized by the same model parameters. Typical model-based methods are the ARMA model [29], the Gaussian mixture model [30], and the hidden Markov model [31]. The model-based method makes use of statistical characteristics of data, which is more informative and interpretable than methods based on distance and deep learning, and the computational efficiency is significantly higher than the three previous ones.

III. CLASSIFICATION OF GAUSSIAN MODEL BASED ON KULLBACK-LEIBLER DIVERGENCE

Figure 1 shows the framework of our KLD-GMC method, and Table 1 illustrates the definitions of some symbols in Figure 1. The classification of the Gaussian model based on Kullback-Leibler divergence first converts MTS training samples into the parameters of the multivariate Gaussian model: the mean vector and the inverse covariance matrix. Secondly, applying Graphical Lasso to obtain the sparse inverse covariance of each subsequence. Then, the unlabeled samples of MTS are also converted into the parameters of the multivariate Gaussian model: the mean vector and the sparse inverse covariance matrix. The means and sparse inverse covariances can be used to calculate the Kullback-Leibler divergence between each unlabeled subsequence and each training subsequence. Finally, the Kullback-Leibler divergence is used as the similarity measurement, each unlabeled subsequence could be classfied by applying a KNN classifier.

A. INTRODUCTION OF KULLBACK-LEIBLER DIVERGENCE

The Kullback-Leibler (*KL*) divergence, also known as the relative entropy, is used to quantify the difference between two probability distributions and can effectively describe the similarity between different distributions. Using this similarity measurement, we can cluster or classify data.

TABLE 1. Symbol definitions of the KLD-GMC method.

Symbol	Definition
$Train\_T_i$	the i-th subsequence sample of the training set, $i \in [1,n]$, the training set has n subsequences
$Train\_mean_i$	the mean of $Train\_T_i$
$Train\_S_i$	the covariance of $Train\_T_i$
$Train\_\Theta_i$	the sparse inverse covariance of $Train\_T_i$, which is a $D*D$ dimensional matrix
$Train\_Y_i$	the class label of $Train\_T_i$
$Unlabeled\_T_j$	the j-th subsequence sample of the testing set, $j \in [1,m]$, the training set has m subsequences
$Unlabeled\_mean_j$	the mean of $Unlabeled\_T_j$
$Unlabeled\_S_j$	the covariance of $Unlabeled\_T_j$
$Unlabeled\_\Theta_j$	the sparse inverse covariance of $Unlabeled\_T_j$, which is a $D*D$ dimensional matrix
KL_j	the Kullback-Leibler divergence between $Unlabeled\_T_j$ and $Train\_T = \{Train\_T_1, \cdots, Train\_T_i, \cdots, Train\_T_n\}$
$Y_?$	an unknown class label
D	the number of time series variables (the dimension, *i.e.* the number of attributes)

Assuming that the statistical models P_1 and P_2 represent two N-dimensional probability distribution functions, respectively, the Kullback-Leibler divergence between those two models is defined as Equations 1 and 2 in the case of discrete and continuous random variables, respectively:

$$KL(P_1||P_2) = \sum_{x \in X} P_1(x) \log \frac{P_1(x)}{P_2(x)} \tag{1}$$

$$KL(P_1||P_2) = \int_{x \in X} P_1(x) \log \frac{P_1(x)}{P_2(x)} dx \tag{2}$$

The physical meaning of the above equations is to calculate the degree of difference between the statistical model and the given statistical model.

The Kullback-Leibler divergence is non-negative and asymmetrical. Taking Equation 2 as an example, since the logarithm function is a convex function, according to the relative entropy and the Gibbs inequality [32], we can get Equation 3.

$$\begin{aligned} &KL(P_1||P_2) \\ &= \int_{x \in X} P_1(x) \log \frac{P_1(x)}{P_2(x)} dx = \int_{x \in X} -\log \frac{P_2(x)}{P_1(x)} P_1(x) dx \\ &\geq -\log \left(\int_{x \in X} \frac{P_2(x)}{P_1(x)} P_1(x) dx \right) = -\log \left(\int_{x \in X} P_2(x) dx \right) \\ &= -\log(1) = 0 \end{aligned} \tag{3}$$

Therefore, the Kullback-Leibler divergence is non-negative; at the same time, the Kullback-Leibler divergence has the asymmetry property, that is, the KL divergence is asymmetrical measure of the two probability distributions:

$$KL(P||Q) \neq KL(Q||P) \tag{4}$$

The Kullback-Leibler divergence is a method of describing the difference between the two probability distributions P and Q, if P represents the true distribution of random variables, Q represents the theoretical or fitting distribution, then $KL(P||Q)$ is called the forward Kullback-Leibler divergence, and $KL(Q||P)$ is called the backward Kullback-Leibler divergence. The fitting distribution in the forward Kullback-Leibler divergence is the denominator of the Kullback-Leibler divergence equation. If the value of the fitting distribution tends to 0 in a certain value range of the random variable, then the value of Kullback-Leibler divergence tends to be infinity. Therefore, when the forward Kullback-Leibler divergence is used to minimize the distance between fitting distribution and true distribution, the fitting distribution tends to cover all ranges of the theoretical distribution. The above properties of the forward Kullback-Leibler divergence are referred to as‘‘zero avoiding’’. Conversely, when using the backward Kullback-Leibler divergence to solve the fitting distribution, since the fitting distribution is a numerator, its zero value does not affect the integral of the Kullback-Leibler divergence, so the backward Kullback-Leibler divergence is ‘‘ero forcing’’. In this paper, the backward Kullback-Leibler divergence is used.

B. SOLVING SPARSE INVERSE COVARIANCE BY GRAPHICAL LASSO

Suppose Σ^{-1} is an inverse covariance matrix, the zeroes in Σ^{-1} correspond to the pairs of features that are

conditionally independent. Furthermore, the sparse inverse covariance can map original MTS instances to a fixed-size vector space independent of sequence length, and the sparse graphical representation is an effective method to prevent overfitting [12]. Therefore, the sparse inverse covariance not only provides an interpretable perspective for classification results, but also avoids overfitting.

Given an MTS training set, the inverse covariance is solved for each subsequence in the training set, assuming that a D-dimensional subsequence T_i ($T_i = (t_1, t_2, \cdots, t_k, \cdots, t_{N_i})$), $k \in [1, N_i]$ obeys a Gaussian distribution $N(U, \Sigma)$ which has two parameters: the mean U and the covariance Σ. Θ_i is the inverse covariance of T_i. Then the log likelihood can be calculated by the formula as shown in Equation 5.

$$\begin{aligned}
&\ell\ell\left(T_i, \Theta_i\right) \\
&\quad = \sum_{k=1}^{N_i} -\frac{1}{2}\left(t_k - U\right)\Theta\left(t_k - U\right)^T + \frac{1}{2}\log\det\Theta_i \\
&\qquad -\frac{D}{2}\log\left(2\pi\right) \\
&\quad = -\frac{D}{2}\log\left(2\pi\right) + \frac{D}{2}\log\det\Theta_i - \frac{1}{2}tr\left(S_i\Theta_i\right) \\
&\quad \propto -|N_i|\left(\log\det\Theta_i - tr\left(S_i\Theta_i\right)\right) + C
\end{aligned} \tag{5}$$

where, $|N_i|$ is the sample size of T_i, $S_i = \left(\sum_{k=1}^{N_i}\left(t_k - U\right)^T\left(t_k - U\right)\right)/D$ is the empirical covariance of T_i, and C is a constant that does not depend on Θ_i. tr represents the trace of matrix, and det is a function to calculate the determinant of matrix.

The classical maximum likelihood estimation method can't solve the sparse inverse covariance. In this paper, the idea of Lasso is used to solve the sparse inverse covariance. The least absolute shrinkage and selection operator (Lasso) [33] is a compression estimation method proposed by Tibshirani in 1996. Lasso gets a more refined model by constructing a penalty function. We uses the pattern of the Lasso algorithm and adds a penalty function, then Equation 5 can be written as:

$$\widehat{\Theta} = \underset{\Theta_i>0}{\arg\min}(-\log\det\Theta_i + tr(S_i\Theta_i) + \frac{1}{|N_i|}\left\|\lambda\circ\Theta_i\right\|_1) \tag{6}$$

Equation 6 describes a convex optimization problem called Graphical Lasso [34]. $\|\lambda\circ\Theta_i\|_1$ is a ℓ_1 norm penalty for the Hadamard product, used to excite the sparse inverse covariance. λ controls the weight of the penalty function, the larger the value of λ, the more sparse the parameter (inverse covariance) is solved, and vice versa.

C. CALCULATION OF MULTIVARIATE GAUSSIAN MODEL BASED ON KULLBACK-LEIBLER DIVERGENCE

Assume that the time series data for this paper is discrete, so we calculate the Kullback-Leibler divergence between the multivariate Gaussian models in the discrete case, as shown in Equation 1.

Given two subsequences T_1 and T_2, it is assumed that the probability distributions of T_1 and T_2 are P_1 and P_2, the corresponding Gaussian model parameters are $\{\mu_1, \Sigma_1\}$ and $\{\mu_2, \Sigma_2\}$ respectively, and the corresponding sparse inverse covariances are Θ_1 and Θ_2 respectively. Then the Kullback-Leibler divergence equation between multivariate Gaussian distributions is shown in Equation 7, the detailed proof is shown in Appendix.

$$\begin{aligned}
D_{KL}\left(P_1||P_2\right) = \frac{1}{2}\Bigg[\log\frac{\left|\Theta_2^{-1}\right|}{\left|\Theta_1^{-1}\right|} - n + tr\left(\Theta_1\Theta_2^{-1}\right) \\
+ \left(\mu_2 - \mu_1\right)^T\Theta_2\left(\mu_2 - \mu_1\right)\Bigg]
\end{aligned} \tag{7}$$

Using Equation 7, the Kullback-Leibler divergence can be calculated between each subsequence to predict the class label and each training subsequence, indicating the degree of difference between the two subsequences. According to the divergence, each subsequence to be predicted is classified.

D. DESCRIPTION OF KULLBACK-LEIBLER DIVERGENCE-BASED GAUSSIAN MODEL FOR MULTIVARIATE TIME SERIES CLASSIFICATION

Table 2 illustrates the definition of some symbols to facilitate the description of the KLD-GMC algorithm. Algorithm 1 is used to solve the mean vector and the sparse inverse covariance matrix of the multivariate time series. An inverse covariance is solved for a sample subsequence in the dataset, and the subsequence is represented by the multivariate Gaussian model corresponding to the mean and the inverse covariance.

The ***mean*** variable in Step 1 and Step 2 is a $1*D$ matrix, and *mean*() is a function to calculate the mean matrix. $\boldsymbol{S}$ in Step 3 is the empirical covariance matrix, and *cov*() is a function to calculate the covariance matrix. In Step 4, λ' is a hyperparameter that controls the weight of penalty function, $Num(\boldsymbol{T})$ is the number of subsequence' attributes. Because the inverse covariance is related with the number of the variables, the datasets with the different numbers of the variables require different λ' to control the weight of penalty function, we propose a fine tuning Equation 8 to adjust the weight of λ' to get λ as a new parameter to control the weight of penalty function.

$$\lambda(\lambda', Num(\boldsymbol{T})) = \lambda' * \frac{Num(\boldsymbol{T})}{\log_2 Num(\boldsymbol{T})} \tag{8}$$

Because more variables a sequence has, more complex relationships of variables in the sequence are. We need to make a large adjustment to make sure the parameters representing features with small influence are zeroes, so that the covariance can characterize the sample well. Therefore we use $Num(\boldsymbol{T})/\log_2 Num(\boldsymbol{T})$ to adjust λ' smoothly, make λ bigger enough, but not too big as illustrated in Equation 8.

Furthermore, the sparse solution is obtained to better characterize the sample, and the *Graphical Lasso* ($\boldsymbol{S}$, λ) function in Step 5 is used to solve the sparse inverse covariance.

The solution algorithm for the MTS classification is shown in Algorithm 2. In Step 2 and Step 4 of Algorithm 2, the sparse

TABLE 2. Symbol definitions used in the KLD-GMC algorithm.

Symbol	Definition
$\boldsymbol{T}$	a subsequence of MTS
$\boldsymbol{mean}$	the empirical mean of $\boldsymbol{T}$
$\boldsymbol{S}$	the empirical covariance of $\boldsymbol{T}$
$\boldsymbol{\Theta}$	the sparse inverse covariance of $\boldsymbol{T}$
$Train\_T$	the training set of MTS, $Train\_T = \{Train\_T_1, \cdots, Train\_T_i, \cdots, Train\_T_n\}$
$Train\_mean$	the empirical mean of each subsequence in the training set, $Train\_mean = \{Train\_mean_1, \cdots, Train\_mean_i, \cdots, Train\_mean_n\}$
$Train\_\Theta$	the sparse inverse covariance of each subsequence in the training set, $Train\_\Theta = \{Train\_\Theta_1, \cdots, Train\_\Theta_i, \cdots, Train\_\Theta_n\}$
$Train\_Y$	the labels corresponding to $Train\_T$, $Train\_Y = \{Train\_Y_1, \cdots, Train\_Y_i, \cdots, Train\_Y_n\}$
$Unlabeled\_T$	an unlabeled MTS
$Unlabeled\_mean$	the empirical mean of $Unlabeled\_T$
$Unlabeled\_\Theta$	the sparse inverse covariance of $Unlabeled\_T$
$result$	the predicted classification result of $Unlabeled\_T$
k	the hyperparameter of the KNN classifier

Algorithm 1 Solving_Sparse_Inverse_Covariance_Matrix

Input: The multivariate time series $\boldsymbol{T}$ and sparseness parameter λ'

Output: $\boldsymbol{\Theta}$ and $\boldsymbol{mean}$

1. Initializing $\boldsymbol{mean}$ be a $1 * D$ zero matrix
2. $\boldsymbol{mean} \leftarrow mean(\boldsymbol{T})$
3. $\boldsymbol{S} \leftarrow cov(\boldsymbol{T})$
4. $\lambda = \lambda(\lambda', Num(\boldsymbol{T}))$
5. $\boldsymbol{\Theta} \leftarrow Graphical\ Lasso\ (\boldsymbol{S}, \lambda)$

Algorithm 2 KNN Classification

Input:
$Train\_T = \{Train\_T_1, \cdots, Train\_T_i, \cdots, Train\_T_n\}$, $Train\_Y = \{Train\_Y_1, \cdots, Train\_Y_i, \cdots, Train\_Y_n\}$, $Unlabeled\_T$, and k.

Output: $result$

1. **for** $i = 1$ to n **do**
2. $\quad (Train\_\Theta_i, Train\_mean_i) \leftarrow Solving\_sparse\_inverse\_covariance\_matrix\ (Train\_T_i)$ by using Algorithm 1
3. **end**
4. $(Unlabeled\_\Theta, Unlabeled\_mean) \leftarrow Solving\_sparse\_inverse\_covariance\_matrix(Unlabeled\_T)$ by using Algorithm 1
5. init KL as a 1^*n vector
6. **for** $i = 1$ to n **do**
7. $\quad KL_i \leftarrow KLD(Unlabeled\_\Theta, Unlabeled\_mean, Train\_\Theta_i, Train\_mean_i)$ by using Equation 7
8. **end**
9. $result \leftarrow KNN(KL, k)$ by using the KNN classifier

inverse covariance matrix and the mean vector of each sample in the training set and the unlabeled sample are solved to construct a multivariate Gaussian model. The Kullback-Leibler divergence is used as the similarity measurement in Step 7, and the KNN classifier is used to classify an unlabeled sample in Step 9.

The idea of the KNN classifier is that the training samples are represented as vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the classification phase, k is a user-defined constant, and an unlabeled vector is classified

TABLE 3. Multivariate time series classification datasets and their characteristics.

Name	attributes	classes	length	instances	training set	test set
JapaneseVowels	12	9	7~29	640	270	370
CMUsubject16	62	2	127~580	58	29	29
KickvsPunch	62	2	274~841	26	16	10
WalkvsRun	62	2	128~1918	44	28	16

TABLE 4. Classification accuracy compared with the state-of-the-art methods.

Name	KLD-GMC	MLP	Encoder	MCNN	t-LeNet	MCDCNN	Time-CNN	TWIESN
JapaneseVowels	**0.981**	0.972	0.976	0.092	0.238	0.944	0.956	0.965
CMUsubject16	0.966	0.6	**0.983**	0.531	0.51	0.514	0.976	0.893
KickvsPunch	**0.7**	0.61	0.61	0.54	0.5	0.56	0.62	0.67
WalkvsRun	**1**	0.7	**1**	0.75	0.6	0.45	**1**	0.944
Average Accuracy	**0.912**	0.721	0.892	0.478	0.462	0.617	0.888	0.868

by assigning the label which is most frequent among the k training samples nearest to that unlabeled point. Since the KNN algorithm mainly relies on the surrounding limited samples, rather than relying on the method of discriminating the class field to determine the category, therefore, the KNN method is more suitable than other methods for samples with more overlaps or intersections in class fields. By using Equation 7, the degree of difference between an unlabeled sample and each training sample can be solved, and then the KNN classifier is used to select the category to an unlabeled sample.

IV. EXPERIMENTAL RESULTS

In this section, we conduct experiments on four datasets to analyze the performance of our method, compare the performance of the proposed algorithm with the state-of-the-art methods, and illustrate the application scenarios that our method is suitable for. Our algorithm is implemented in Python 3.6.7, and all experiments are performed on a computer with Intel(R) Xeon(R) CPU E5-2620 0 @ 2.00 GHz CPU, 64 GB RAM, Ubuntu 16.04.

A. DESCRIPTION OF DATASETS

In the experiments, four real datasets were selected from the University of California Irvine (UCI) Machine Learning Repository [35] and the CMU Graphics Lab Motion Capture Database [36]. All selected datasets are listed in Table 3.

The University of California Irvine (UCI) Machine Learning Repository provides a dataset, namely the Japanese Vowels dataset. The Japanese Vowels dataset collected nine male speakers for two consecutive Japanese vowels /ae/. For each utterance, a 12-degree linear prediction analysis was applied to it to obtain a discrete-time series with 12 LPC cepstrum coefficients. This means that a speaker's utterance forms a time series with a length in the range of 7-29, and each point of the time series has 12 features (12 coefficients). The total number of time series is 640. Among them, 270 time series are used as training sets and 370 time series are used as test sets. CMU created a Graphics Lab Motion Capture Database, from which we selected the WalkvsRun dataset, the KickvsPunch dataset, and the CMUsubject16 dataset for our experiments.

B. COMPARATIVE EXPERIMENTS

For the above four datasets, we compare the proposed algorithm with the state-of-the-art MTS classification algorithms, including MLP [24], Encoder [11], MCNN [11], t-LeNet [25], MCDCNN [26], Time-CNN [27], TWIESN [28]. Evaluation metric is accuracy, which is shown as Equation 9.

$$\text{Accuracy} = \frac{\text{the number of samples classified correctly}}{\text{the number of all samples}} \tag{9}$$

Table 4 presents the experimental results for different methods. The experimental results of MLP, Encoder, MCNN, t-LeNet, MCDCNN, Time-CNN, TWIESN are reported by [11]. Comparing the performance of the state-of-the-art methods, we can see that KLD-GMC achieves better performance on all datasets.

Through Table 4, we can get some interesting conclusions. First of all, comparing with other methods, KLD-GMC achieves best performance on most datasets which have many variables. The reason for this phenomenon is obvious: our method converts the MTS sample into the parameters of the multivariate Gaussian model by calculating the mean vector and the inverse covariance matrix of the MTS samples, so that the multivariate Gaussian model can be constructed by using those two parameters. The most important parameter is the inverse covariance which can measure the relationship between variables in the sample well. Therefore, the more variables in the sample, the more information can be obtained by the inverse covariance, and the constructed multivariate Gaussian model can also represent the sample better, so that our method can achieve better performance.

The second conclusion is that KLD-GMC is not very good at dealing with anomalies, and cannot achieve best results on

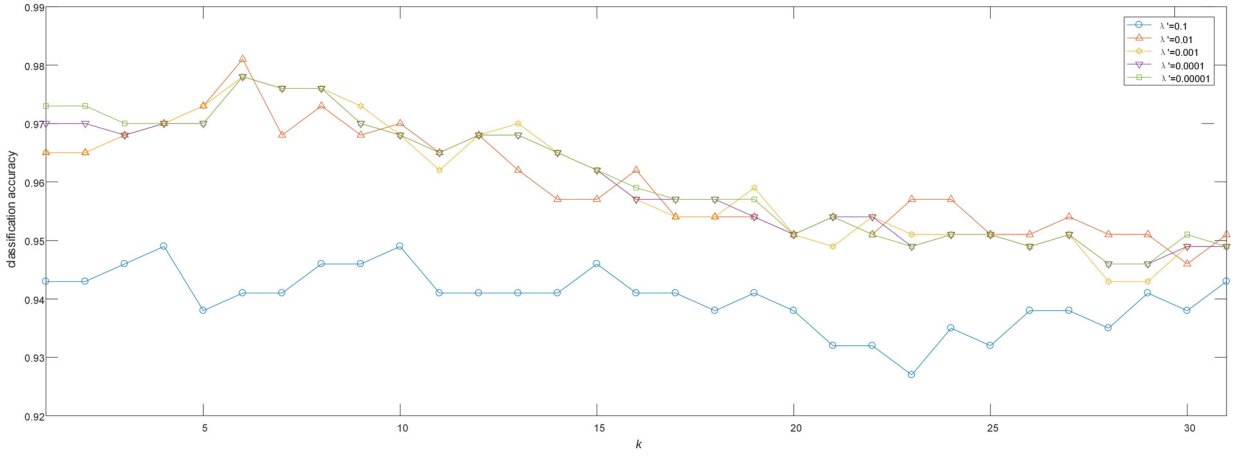

FIGURE 2. Classification accuracy varies with k at different values of λ' for JapaneseVowels.

some datasets. In general, the distribution of data points in a class is relatively concentrated, but in the real world, there is a situation where the exception point of one class may be closer to another. And then, by calculating the Kullback-Leibler divergence between the unlabeled sample and each training sample to determine the degree of difference between them, if the 1NN classifier is used, the exception point will greatly affect the result. Hence, we apply the KNN classifier to improve the effect, the KNN classifier makes decision based on the dominant category of k objects rather than a single object's category, and experimental results show that the KNN classifier does improve the effect and achieve the best performance.

The third conclusion is that our method is superior compared with the current deep learning-based method, which not only can exceed most methods in effect, but our method is more interpretable. In addition, in the case of small sample datasets, deep learning-based methods may not train the model well with small amount of data. Our method mainly considers how to better represent the data, it refers to the characteristics of the data: the relationship between multiple variables, for this reason our method does not have a process of training the model. So that there is no over-fitting condition, our method does not sensitive to the size of the training dataset. Therefore, the proposed method is very suitable for multivariable situations, such as car networking with many sensors [37], health monitoring equipment [38] and so on.

C. COMPUTATIONAL COMPLEXITY ANALYSIS

As can be seen from Section III, the proposed approach consists of two main parts. The first part is to use Graphical Lasso to solve the sparse inverse covariance of all subsequences of the training set and the unlabeled sample. The second part is to calculate the Kullback-Leibler divergence between the unlabeled sample and each training sample, and classify the unlabeled sample by using the Kullback-Leibler divergence as the similarity measurement, where the KNN classifier is used. Similarly, the complexity of the algorithm in this paper is mainly composed of these two parts. Therefore, the computational complexity of the proposed algorithm can be obtained by analyzing the two parts.

The first part uses Graphical Lasso to solve the sparse inverse covariance, which has been studied by many researchers. As articles [34], [39] proved, computational complexity for solving the sparse inverse covariance is $O(K^*D^3)$, where K is the maximum number of iterations, and D is the number of variables in MTS. Therefore, the computational complexity of the first part is $O(N^*K^*D^3)$, where N is the number of samples in the training set.

The second part is to calculate the Kullback-Leibler divergence between the unlabeled sample and each training sample. The Kullback-Leibler divergence is used as the similarity measurement, and the KNN classifier is used to classify test samples. The computational complexity of calculating the Kullback-Leibler divergence between the unlabeled sample and a training sample is $O(D^3)$ by analyzing the KLD equation item by item, where D is the number of variables in MTS. Therefore, the computational complexity of the second part is $O(N^*D^3)$.

In conclusion, the overall computational complexity of the proposed algorithm is $O((K+1)^*N^*D^3)$.

D. IMPACT OF PARAMETERS ON PERFORMANCE

Among the proposed approach, there are two parameters that may have an impact on the performance of MTS classification. The first hyperparameter is λ' which controls the weight of the penalty function. The second is the choice of k in the KNN classifier. Figure 2 uses the JapaneseVowels dataset as an example to describe the accuracy of the classification results as a function of k under different λ'. Figure 3 describes the accuracy of the classification results as a function of λ' in the case of $k = 6$.

An important parameter is the value of k. It can be observed from Figure 2 that in the case of $\lambda' \leq 0.01$, the value of k is the

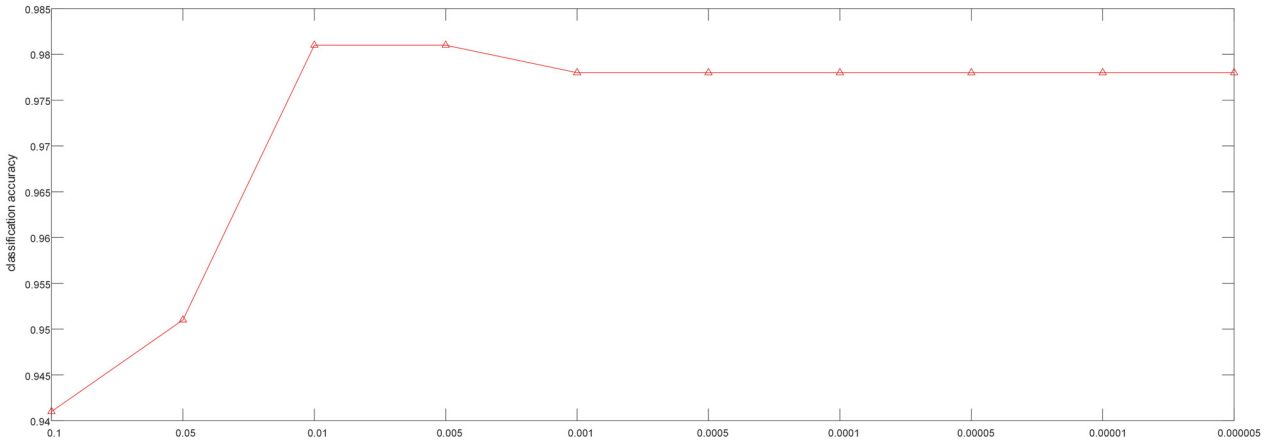

FIGURE 3. Classification accuracy varies with λ' in the case of $k = 6$.

same when the effect is the best. So we can get an experience that the value of k is universal for a given dataset. In the JapaneseVowels dataset, $k = 6$ is a suitable choice.

The value of λ' is a more important parameter. According to Figure 3, in the case of $k = 6$, we can see the performance achieved the best result when $0.005 \leq \lambda' \leq 0.01$. On the one hand, if λ' is too large, the penalty function will have an excessive influence on the solution of the sparse inverse covariance, so that the data cannot be accurately characterized. On the other hand, if it is very small, the penalty function will have little or no effect on the solution of the sparse inverse covariance, and the correction of the solution process will not be enough, which will lead to the deterioration of the characterization data. Therefore, we need to choose a suitable value, as shown in Figure 2, in the JapaneseVowels dataset, the best results will be achieved when $\lambda' = 0.01$, $K = 6$.

Through the experiments on the JapaneseVowels dataset, we applied $\lambda' = 0.01$, $K = 6$ as the empirical parameters to other datasets, and the results verified that it can achieve a good performance.

V. CONCLUSION

In this paper, we consider the problem of MTS classification, which is one of the foundations of pattern recognition applications, and propose a novel model-based MTS classification method KLD-GMC. KLD-GMC first converts the MTS data into the parameters of the multivariate Gaussian model: the mean vector and the inverse covariance matrix; then uses the Graphical Lasso to sparsely solve each inverse covariance to obtain the sparse inverse covariance. After that, we consider the Kullback-Leibler divergence as the similarity measurement between different samples. The Kullback-Leibler divergence between each unlabeled subsequence and each training subsequence can be calculated by using the mean and the sparse inverse covariance. Finally, each test subsequence is classified by using the KNN classifier. Experimental results show that the method has higher accuracy.

A valuable enhancement for KLD-GMC is to improve the computational efficiency, on account of the Kullback-Leibler divergence between every unlabeled sample and every training sample has to be calculated which cause the high computational cost. Thus, further research should be carried out on the computational optimization of the proposed method. In addition, the choice of parameters is a difficult problem. Since the different datasets have the different numbers of the variables, the choice of λ' and k will change slightly. Although the empirical values $\lambda' = 0.01$, $k = 6$ have already achieved a very good performance, we need to consider how to design the algorithm to set the parameters adaptively in the future.

APPENDIX

In this section, we will detail the proof of Equation 7 and introduce some basic theorems used in the proof process.

Assuming that there is an n-dimensional matrix A, then the trace of matrix A (represented by $tr\ (A)$) is equal to the sum of the eigenvalues of A, that is, the sum of the properties of the main diagonal elements of the matrix A, with the following properties:

$$tr\ (\alpha A + \beta B) = \alpha tr\ (A) + \beta tr\ (B) \tag{10}$$

$$tr\ (AB) = tr\ (BA) \tag{11}$$

Equation 12 can be derived from Equation 11:

$$tr\ (ABC) = tr\ (CAB) = tr\ (BCA) \tag{12}$$

The result of the trace operation is constant under the transposition operation, as shown in Equation 13:

$$tr\ (A) = tr\left(A^T\right) \tag{13}$$

For the column vector λ, the result of the formula $\lambda^T A\lambda$ is a scalar, so there is:

$$\lambda A\lambda^T = tr\left(\lambda^T A\lambda\right) = tr\left(A\lambda\lambda^T\right) \tag{14}$$

The property of expectation E and covariance $\sum$ in a multivariate distribution is shown in Equation 15:

$$E\left[xx^T\right] = \sum + \mu\mu^T \tag{15}$$

Equation 15 is proved as follows:

$$\begin{aligned}
&\sum \\
&= E\left[(x-\mu)(x-\mu)^T\right] = E\left[xx^T - x\mu^T - \mu x^T + \mu\mu^T\right] \\
&= E\left[xx^T\right] - \mu\mu^T - \mu\mu^T + \mu\mu^T = E\left[xx^T\right] - \mu\mu^T
\end{aligned} \tag{16}$$

Assuming x is a column vector, Equation 17 can be obtained by using Equation 14, and the proof process is as shown in Equation 18:

$$\begin{aligned}
&E\left(x^TAx\right) \\
&= tr\left(A\sum\right) + \mu^TA\mu
\end{aligned} \tag{17}$$

$$\begin{aligned}
&E\left(x^TAx\right) \\
&= E\left[tr\left(x^TAx\right)\right] = E\left[tr\left(Axx^T\right)\right] \\
&= tr\left[E\left(Axx^T\right)\right] = tr\left[AE\left(xx^T\right)\right] = tr\left[A\left(\sum + \mu\mu^T\right)\right] \\
&= tr\left(A\sum\right) + tr\left(A\mu\mu^T\right) = tr\left(A\sum\right) + tr\left(\mu^TA\mu\right) \\
&= tr\left(A\sum\right) + \mu^TA\mu
\end{aligned} \tag{18}$$

Using the above properties and derivations, we can get Equation 19, the proof process as follows:

$$\begin{aligned}
&D_{KL}(P_1||P_2) \\
&= \sum_{x\in X} P_1(x)\, log\frac{P_1(x)}{P_2(x)} \\
&= E_{P_1}[logP_1 - logP_2] \\
&= \frac{1}{2}E_{P_1}\left[-log\left|\sum_1\right| - (x-\mu_1)^T\sum_1^{-1}(x-\mu_1)\right. \\
&\quad \left. + \log\left|\sum_2\right| + (x-\mu_2)^T\sum_2^{-1}(x-\mu_2)\right] \\
&= \frac{1}{2}\log\frac{|\sum_2|}{|\sum_1|} + \frac{1}{2}E_{P_1}\left[-(x-\mu_1)^T\sum_1^{-1}(x-\mu_1)\right. \\
&\quad \left. + (x-\mu_2)^T\sum_2^{-1}(x-\mu_2)\right] \\
&= \frac{1}{2}\log\frac{|\sum_2|}{|\sum_1|} + \frac{1}{2}E_{P_1}\left\{-tr\left[\sum_1^{-1}(x-\mu_1)(x-\mu_1)^T\right]\right. \\
&\quad \left. + tr\left[\sum_2^{-1}(x-\mu_2)(x-\mu_2)^T\right]\right\} \\
&= \frac{1}{2}\log\frac{|\sum_2|}{|\sum_1|} + \frac{1}{2}E_{P_1}\left\{-tr\left[\sum_1^{-1}(x-\mu_1)(x-\mu_1)^T\right]\right\} \\
&\quad + \frac{1}{2}E_{P_1}\left\{tr\left[\sum_2^{-1}(x-\mu_2)(x-\mu_2)^T\right]\right\} \\
&= \frac{1}{2}\log\frac{|\sum_2|}{|\sum_1|} - \frac{1}{2}tr\left\{E_{P_1}\left[\sum_1^{-1}(x-\mu_1)(x-\mu_1)^T\right]\right\} \\
&\quad + \frac{1}{2}tr\left\{E_{P_1}\left[\sum_2^{-1}(x-\mu_2)(x-\mu_2)^T\right]\right\} \\
&= \frac{1}{2}\log\frac{|\sum_2|}{|\sum_1|} - \frac{1}{2}tr\left\{\sum_1^{-1}E_{P_1}\left[(x-\mu_1)(x-\mu_1)^T\right]\right\} \\
&\quad + \frac{1}{2}tr\left\{E_{P_1}\left[\sum_2^{-1}\left(xx^T - \mu_2x^T - x\mu_2^T + \mu_2\mu_2^T\right)\right]\right\} \\
&= \frac{1}{2}\log\frac{|\sum_2|}{|\sum_1|} - \frac{1}{2}tr\left\{\sum_1^{-1}\sum_1\right\} \\
&\quad + \frac{1}{2}tr\left\{\sum_2^{-1}E_{P_1}\left(xx^T - \mu_2x^T - x\mu_2^T + \mu_2\mu_2^T\right)\right\} \\
&= \frac{1}{2}\log\frac{|\sum_2|}{|\sum_1|} - \frac{1}{2}n + \frac{1}{2}tr\left\{\sum_2^{-1}\left(\sum_1 + \mu_1\mu_1^T - \mu_2\mu_1^T\right.\right. \\
&\quad \left.\left. - \mu_1\mu_2^T + \mu_2\mu_2^T\right)\right\} \\
&= \frac{1}{2}\left[\log\frac{|\sum_2|}{|\sum_1|} - n + tr\left(\sum_2^{-1}\sum_1\right) + tr\right. \\
&\quad \left.\times\left(\sum_2^{-1}\mu_1\mu_1^T - \sum_2^{-1}\mu_2\mu_1^T - \sum_2^{-1}\mu_1\mu_2^T + \sum_2^{-1}\mu_2\mu_2^T\right)\right] \\
&= \frac{1}{2}\left[\log\frac{|\sum_2|}{|\sum_1|} - n + tr\left(\sum_2^{-1}\sum_1\right)\right. \\
&\quad \left. + tr\left(\mu_1\sum_2^{-1}\mu_1^T - 2\mu_1^T\sum_2^{-1}\mu_2 + \mu_2\sum_2^{-1}\mu_2^T\right)\right] \\
&= \frac{1}{2}\left[\log\frac{|\sum_2|}{|\sum_1|} - n\right. \\
&\quad \left. + tr\left(\sum_2^{-1}\sum_1\right) + (\mu_2-\mu_1)^T\sum_2^{-1}(\mu_2-\mu_1)\right]
\end{aligned} \tag{19}$$

We put the sparse inverse covariances Θ_1 and Θ_2 into Equation 19 to take the place of the covariances $\sum_1$ and $\sum_2$, finally we can get Equation 7:

$$\begin{aligned}
D_{KL}(P_1||P_2) = \frac{1}{2}\left[\log\frac{\left|\Theta_2^{-1}\right|}{\left|\Theta_1^{-1}\right|} - n + tr\left(\Theta_1\Theta_2^{-1}\right)\right. \\
\left. + (\mu_2-\mu_1)^T\Theta_2(\mu_2-\mu_1)\right]
\end{aligned} \tag{7}$$

REFERENCES

[1] G. Paragliola and A. Coronato, "Gait anomaly detection of subjects with Parkinson's disease using a deep time series-based approach," *IEEE Access*, vol. 6, pp. 73280–73292, 2018. doi: 10.1109/ACCESS.2018.2882245.

[2] D. P. Barrett and J. M. Siskind, "Action recognition by time series of retinotopic appearance and motion features," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 12, pp. 2250–2263, Dec. 2016. doi: 10.1109/TCSVT.2015.2502839.

[3] P. M. Baggenstoss and B. F. Harrison, "Class-specific model mixtures for the classification of acoustic time series," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 52, no. 4, pp. 1937–1952, Aug. 2016. doi: 10.1109/TAES.2016.150285.

[4] M. Cheng, Y. Ling, and W. B. Wu, "Time series analysis for jamming attack detection in wireless networks," in *Proc. GLOBECOM-IEEE Global Commun. Conf.*, Singapore, Dec. 2017, pp. 1–7. doi: 10.1109/GLOCOM.2017.8254000.

[5] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "DeepAnT: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2019. doi: 10.1109/ACCESS.2018.2886457.

[6] E. S. García-Treviño and J. A. Barria, "Structural generative descriptions for time series classification," *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1978–1991, Oct. 2014. doi: 10.1109/TCYB.2014.2322310.

[7] K. Shi, H. Qin, C. Sima, S. Li, L. Shen, and Q. Ma, "Dynamic barycenter averaging kernel in RBF networks for time series classification," *IEEE Access*, vol. 7, pp. 47564–47576, 2019. doi: 10.1109/ACCESS.2019.2910017.

[8] H. Li and C. Wang, "Similarity measure based on incremental warping window for time series data mining," *IEEE Access*, vol. 7, pp. 3909–3917, 2019. doi: 10.1109/ACCESS.2018.2889792.

[9] A. Abanda, U. Mori, and J. A. Lozano, "A review on distance based time series classification," in *Data Mining Knowl. Discovery*, vol. 33, no. 2, pp. 378–412, 2019.

[10] J. Wu, L. Yao, and B. Liu, "An overview on feature-based classification algorithms for multivariate time series," in *Proc. IEEE 3rd Int. Conf. Cloud Comput. Big Data Anal. (ICCCBDA)*, Chengdu, China, Apr. 2018, pp. 32–38. doi: 10.1109/ICCCBDA.2018.8386483.

[11] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Müller, "Deep learning for time series classification: A review," *Data Mining Knowl. Discovery*, vol. 33, no. 4, pp. 917–963, Mar. 2019.

[12] D. Koller, N. Friedman, and F. Bach, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA, USA: MIT Press, 2009.

[13] S. L. Lauritzen, *Graphical Models*. Gloucestershire, U.K.: Clarendon Press, 1996.

[14] S.-W. Kim, D. H. Park, and H. G. Lee, "Efficient processing of subsequence matching with the Euclidean metric in time-series databases," *Inf. Process. Lett.*, vol. 90, no. 5, pp. 253–260, 2004.

[15] C. S. Möller-Levet, F. Klawonn, K.-H. Cho, and O. Wolkenhauer, "Fuzzy clustering of short time-series and unevenly distributed sampling points," in *Proc. Int. Symp. Intell. Data Anal.*, in Lecture Notes in Computer Science, vol. 2810. Berlin, Germany: Springer, 2003, pp. 330–340.

[16] M. Kumar, N. R. Patel, and J. Woo, "Clustering seasonality patterns in the presence of errors," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Edmonton, AB, Canada, 2002, pp. 557–563.

[17] M. Okawa, "Template matching using time-series averaging and DTW with dependent warping for online signature verification," *IEEE Access*, vol. 7, pp. 81010–81019, 2019. doi: 10.1109/ACCESS.2019.2923093.

[18] A. Sharabiani, H. Darabi, A. Rezaei, S. Harford, H. Johnson, and F. Karim, "Efficient classification of long time series by 3-d dynamic time warping," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 10, pp. 2688–2703, Oct. 2017. doi: 10.1109/TSMC.2017.2699333.

[19] X. Weng and J. Shen, "Classification of multivariate time series using two-dimensional singular value decomposition," *Knowl.-Based Syst.*, vol. 21, no. 7, pp. 535–539, Oct. 2008. doi: 10.1016/j.knosys.2008.03.014.

[20] X. Weng and J. Shen, "Classification of multivariate time series using locality preserving projections," *Knowl.-Based Syst.*, vol. 21, no. 7, pp. 581–587, 2008.

[21] M. G. Baydogan and G. Runger, "Learning a symbolic representation for multivariate time series classification," *Data Mining Knowl. Discovery*, vol. 29, no. 2, pp. 400–422, Mar. 2015. doi: 10.1007/s10618-014-0349-y.

[22] L. Ye and E. Keogh, "Time series shapelets: A new primitive for data mining," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Paris, France, 2009, pp. 947–956. doi: 10.1145/1557019.1557122.

[23] J. Zakaria, A. Mueen, E. Keogh, and N. Young, "Accelerating the discovery of unsupervised-shapelets," *Data Mining Knowl. Discovery*, vol. 30, no. 1, pp. 243–281, 2016.

[24] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Anchorage, AK, USA, May 2017, pp. 1578–1585. doi: 10.1109/IJCNN.2017.7966039.

[25] A. L. Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time series classification using convolutional neural networks," in *Proc. ECML/PKDD Workshop Adv. Anal. Learn., Temporal Data*, Riva Del Garda, Italy, Sep. 2016 pp. 1–9. [Online]. Available: https://halshs.archives-ouvertes.fr/halshs-01357973/

[26] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *Proc. Int. Conf. Web-Age Inf. Manage. (WAIM)*, Macau, China, 2014, pp. 298–310.

[27] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *J. Syst. Eng. Electron.*, vol. 28, no. 1, pp. 162–169, Feb. 2017. doi: 10.21629/JSEE.2017.01.18.

[28] P. Tanisaro and G. Heidemann, "Time series classification using time warping invariant echo state networks," in *Proc. 15th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Anaheim, CA, USA, Dec. 2016, pp. 831–836. doi: 10.1109/ICMLA.2016.0149.

[29] S. Gouveia, T. A. Möller, C. H. Weiß, and M. G. Scotto, "A full ARMA model for counts with bounded support and its application to rainy-days time series," *Stochastic Environ. Res. Risk Assessment*, vol. 32, no. 9, pp. 2495–2514, 2018.

[30] H. Cao, V. Y. F. Tan, and J. Z. F. Pang, "A parsimonious mixture of Gaussian trees model for oversampling in imbalanced and multimodal time-series classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2226–2239, Dec. 2014. doi: 10.1109/TNNLS.2014.2308321.

[31] H. Guo, W. Pedrycz, and X. Liu, "Hidden Markov models based approaches to long-term prediction for granular time series," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 5, pp. 2807–2817, Oct. 2018. doi: 10.1109/TFUZZ.2018.2802924.

[32] J. D. Weeks, "External fields, density functionals, and the Gibbs inequality," *J. Stat. Phys.*, vol. 110, nos. 3–6, pp. 1209–1218, 2003.

[33] R. Tibshirani, "Regression shrinkage and selection via the lasso: A retrospective," *J. Roy. Stat. Soc., B (Stat. Methodol.)*, vol. 73, no. 3, pp. 273–282, 2011. doi: 10.1111/j.1467-9868.2011.00771.x.

[34] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, Jul. 2008. doi: 10.1093/biostatistics/kxm045.

[35] A. Frank and A. Asuncion. *UCI Machine Learning Repository*. Accessed: Aug. 20, 2019. [Online]. Available: http://archive.ics.uci.edu/ml

[36] CMU. (2012). *Graphics Lab Motion Capture Database*. [Online]. Available: http://mocap.cs.cmu.edu

[37] M. L. Bernardi, M. Cimitile, F. Martinelli, and F. Mercaldo, "Driver and path detection through time-series classification," *J. Adv. Transp.*, vol. 2018, pp. 1–20, Mar. 2018.

[38] F. Hussain, F. Hussain, M. Ehatisham-Ul-Haq, and M. A. Azam, "Activity-aware fall detection and recognition based on wearable sensors," *IEEE Sensor J.*, vol. 19, no. 12, pp. 4528–4536, Jun. 2019. doi: 10.1109/JSEN.2019.2898891.

[39] J. Honorio, D. Samaras, N. Paragios, R. Goldstein, and L. E. Ortiz, "Sparse and locally constant Gaussian graphical models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 745–753.

GONGQING WU received the bachelor's degree from Anhui Normal University, China, the master's degree from the University of Science and Technology of China (USTC), and the Ph.D. degree from the Hefei University of Technology, China, all in computer science.

He has authored or coauthored over 30 research papers. He is currently an Associate Professor of computer science with the Hefei University of Technology, China. His current research interests include data mining and web intelligence.

Dr. Wu received the Best Paper Award at the 2011 IEEE International Conference on Tools with Artificial Intelligence (ICTAI) and the Best Paper Award at the 2012 IEEE/WIC/ACM International Conference on Web Intelligence (WI).

HUICHENG ZHANG received the bachelor's degree in computer science and technology from the Hefei University of Technology, China, in 2017, where he is currently pursuing the master's degree with the Key Laboratory of Knowledge Engineering with Big Data, School of Computer Science and Information Engineering.

His research interests are in the field of data mining and pattern recognition.

YING HE received the M.S. degree in computer science and information engineering from the Hefei University of Technology, China, in 2018, where she is currently pursuing the Ph.D. degree with the Key Laboratory of Knowledge Engineering with Big Data, School of Computer Science and Information Engineering.

Her research interests are in the field of data mining and web intelligence.

XIANYU BAO received the B.S. and Ph.D. degrees in electronic information engineering from the Hefei University of Technology (HFUT), Hefei, China.

He is currently a Research Fellow with the High-tech Department of Shenzhen Academy of Inspection and Quarantine. His current research interests include big data technology, and monitoring and early warning.

LEI LI received the bachelor's degree in information and computational science from Jilin University, China, in 2004, the master's degree in applied mathematics from the Memorial University of Newfoundland, Canada, in 2006, and the Ph.D. degree in computing from Macquarie University, Australia, in 2012.

He is currently an Associate Professor of computer science with the Hefei University of Technology, China. He has published over 70 peer-reviewed articles in prestigious journals and top international conferences. His research interests include graph computing and data mining.

XUEGANG HU received the B.S. degree from the Department of Mathematics, Shandong University, Shandong, China, and the M.S. and Ph.D. degrees in computer science from the Hefei University of Technology (HFUT), China.

He is currently a Professor of computer science with the Hefei University of Technology, China. He has published over 80 peer-reviewed articles in prestigious journals and top international conferences. His current research interests include data mining and knowledge engineering.