World Wide Web (2020) 23:3055–3081
https://doi.org/10.1007/s11280-020-00820-z

Extracting diverse-shapelets for early classification on time series

Wenhe Yan[1] · Guiling Li[1,2] · Zongda Wu[3] · Senzhang Wang[4] · Philip S. Yu[5]

Received: 1 February 2019 / Revised: 26 December 2019 / Accepted: 22 April 2020 /
Published online: 21 May 2020

Abstract
In recent years, early classification on time series has become increasingly important in time-sensitive applications. Existing shapelet based methods still cannot work well on this problem. First, the effectiveness of traditional shapelet based methods would be influenced by the number of shapelet candidates. Second, it is difficult for previous methods to obtain diverse shapelets in shapelet selection. In this paper, we propose an Improved Early Distinctive Shapelet Classification method named IEDSC. We first present a new method to more precisely measure the similarity between time series, which takes into account of the relative trend of time series. Second, in shapelet extraction, we propose a pruning technique to reduce the number of shapelets by predicting the starting positions of shapelets with good quality. In addition, a new shapelet selection method is also proposed to remove the similar shapelets, so as to maintain the diversity of shapelets. Finally, the experimental results on 16 benchmark datasets show that the proposed method outperforms state-of-the-art for early classification on time series.

Keywords Time series · Early classification · Shapelet

1 Introduction

Time series is a sequence of data changing with time order, so it is high-dimensional, consecutive and infinitely increasing. In recent years, time series classification has attracted rising research interest, and has been widely used in many application domains, such as medical

✉ Guiling Li
guiling@cug.edu.cn

1 School of Computer Science, China University of Geosciences, Wuhan, China

2 Hubei Key Laboratory of Intelligent Geo-Information Processing, China University of Geosciences, Wuhan, China

3 Department of Computer Science and Engineering, Shaoxing University, Shaoxing, China

4 College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, China

5 Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

diagnosis [19, 27, 31], disaster prediction [38], industrial production control [39], financial market [32] and community discovery [7]. At present, time series classification has many new extensions, of which early classification on time series data is becoming increasingly important. Early classification on time series can be used in many time-sensitive fields, including but not limited to video surveillance, intrusion detection, earthquake warning, early diagnosis, and action recognition [23, 28]. For example, in the early diagnosis of heart disease, abnormal ECG signals may indicate a specific heart disease that needs immediate treatment. Early diagnosis is critical in applications such as intensive care. If a classification model that can make early diagnosis as soon as early of ECG time series is available, the patient with the heart disease can get early treatment.

The goal of early classification on time series is to make prediction as early as possible provided that the accuracy is comparable to the full length. Therefore, there are two requirements in early classification on time series. First, the algorithm should confirm at the earliest time of reliable classification, and thus the predictions could be used for next steps. Second, the classification accuracy of the classifier trained on partial data should achieve comparable performance to that of the classifier trained on the entire data.

It is challenging for early classification on time series to construct an effective classifier. For time series classification with complete data, it is feasible to extract the required features from the whole data for constructing an accurate classifier. However, this does not work for the early classification task as many features are unavailable due to the very limited observed time series data. Xing et al. [34, 35] proposed Early Classification on Time Series (ECTS) model to tackle the problem that conventional classifiers lack earliness. ECTS is a nearest neighbor based classifier, and uses minimum prediction length (MPL) to find a stable prefix subspace to make prediction, but the prediction result lacks interpretability. In [37], Ye et al proposed a new data mining primitive, called time series shapelets. Shapelets are time series subsequences which are in some extent maximally representative of a class. Shapelets could make the classification more accurate, interpretable, and efficient. Following this work, several works also tried to use shapelets for time series classification [11, 20, 22]. Xing et al. [36] constructed an early classifier by extracting the shapelets and the classification result of their method is more accurate and interpretable.

There are mainly two types of works that use shapelets for time series classification. One is to first extract all the shapelets and then select useful shapelets by certain feature selection strategy [36], while the other is to generate shapelets by optimization methods [11]. In this paper, we focus on improving the first type of work. A major limitation of exsting works is that they will produce a large number of redundant shapelets, and thus lead to low efficiency. Another issue of current works is that they do not consider the diversity of shapelets. If we keep the diversity of shapelets, we will make the shapelets more distinguishable and achieve more accurate classification.

In this paper, we propose an Improved Early Distinctive Shapelet Classification method named IEDSC for more accurate early classification on time series. Specifically, we first present a new trend-based Euclidean distance to measure the similarity between two time series. Next we propose to prune the shapelets based on the estimated starting positions. Because shapelet is a subsequence of time series, it has a starting position and end position. If we could obtain starting positions of high quality shapelets and extract shapelets near these starting positions, the number of shapelet candidates can be reduced greatly. Further, we define an extended self-similarity and propose a diverse shapelet selection method to make the extracted shapelets more diverse. To more clearly show the novelty of IEDSC method, we compare it with four candidate methods under five characteristics as shown in Table 1.

Table 1 Comparison of five classification methods

Method description	ECTS	EDSC	ECDIRE	RelClass	IEDSC
Early Classification	✓	✓	✓	✓	✓
Shapelet as Feature	×	✓	×	×	✓
Similarity with Trend	×	×	×	×	✓
Shapelet Pruning	×	×	×	×	✓
Diverse Shapelet Selection	×	×	×	×	✓

The main contributions of this paper are summarized as follows:

- A new similarity measure for time series is proposed by taking the relative trends of two time series into consideration.
- A shapelet pruning technique is proposed to effectively prune shapelet candidates. It first estimates the starting positions of the shapelets and then extracts shapelets from these starting positions, such that only the shapelets near the estimated starting position are extracted. Thus the number of shapelet candidates can be significantly reduced.
- A new shapelet selection method is proposed to effectively remove the similar shapelets so as to make the selected shapelets more diverse.
- Extensively experiments on a bunch of real datasets verify the effectiveness and efficiency of our proposal.

The rest of this paper is organized as follows. In Section 2, we review the related work. We introduce some definitions and preliminaries in Section 3. In Section 4, we describe the feature extraction method. We introduce the feature selection and early classification method in Section 5. We demonstrate our experimental results in Section 6. Finally, we conclude the paper in Section 7.

2 Related work

In recent years, early classification on time series has received increasing research attention in the community of machine learning [15, 26, 33]. However, early classification has its own uniqueness, such as the temporal correlation in the data [24, 30]. Xing et al. [34, 35] for the first time formulated early classification problem explicitly. They proposed the concept of minimum prediction length (MPL), and then presented an early classification algorithm called ECTS. During model training, ECTS needs to calculate the MPL for each sequence in the training set. To predict the class label of an unlabeled time series, at each time stamp ECTS searchs its current nearest neighbor under the condition that the MPL of nearest neighbor cannot be larger than the current time stamp. If the nearest neighbor meets the condition, the prediction can be made in advance, otherwise, it should wait for more incoming data.

Some early classification methods simply train a classification model with the time series data at the early stage, and design criteria to judge whether the prediction result is reliable. Lin et al. [21] proposed an early classification method that utilized the hidden markov models (HMM) to classify the time series. It was evaluated at each time stamp and could get the best results when the complete time series was obtained. This approach lacks flexibility, because the prediction is always done at time stamp which is consistent with the length of training time series, and thus limits the application of the model [9]. Ghalwash et al. [10]

proposed a specially designed hybrid model, which integrated HMM and support vector model (SVM). Although this model can make more reliable prediction by setting threshold value, its limitation is that plenty of samples are required for training.

Mori et al. [25] proposed an algorithm Early Classification of Time Series based on Discriminating Classes Over Time (ECDIRE). In the training phase, ECDIRE analyzes the discrimination of each class at each time stamp, and selects some certain time stamps that the classification accuracy of each class should exceed a predefined threshold. In the prediction phase, the prediction should be made at or after these time stamps. Ando and Suzuki [1] proposed an optimization-based learning method for timely prediction, which could directly minimize an empirical risk function and response time to achieve the minimal prediction risk without a user-defined trade-off between accuracy and earliness. Parrish et al. [26] proposed optimal and practical decision rules for classifying incomplete data. In [26], the class label of unclassified data is obtained only if the reliability threshold is met. The reliability threshold is used to ensure that the predicted class label of incomplete data would be the same as the label assigned for the complete data.

For early classification, the classification model should achieve a good trade-off between earliness and accuracy. Mori et al. [24] presented an approach for early classification of time series based on combining a set of probabilistic classifiers together with a stopping rule. The rule could decide when to make a prediction or when to wait for more data, which ensures the prediction accuracy and reliability. Romain et al. [27] proposed an early classification algorithm which can optimize classification accuracy and decision delay cost, and can point out the best time to make early classification. Hartvigsen et al. [12] proposed an early classification model, called EARLIEST. The model consists of the novel pairing of a recurrent discriminator network with a stochastic policy network, with the latter learning halting-policy as a reinforcement learning task. Schäfer and Leser [29] presented TEASER that modeled eTSC as a two-tier classification problem. The first tier periodically computes class probabilities and the second tier decides reliability of the predicted label. Once the reliability attains a threshold, the prediction is made.

Some early classification algorithms achieve good classification accuracy, but the drawback is the lack of interpretability. Ye et al. [37] proposed a new primitive, called time series shapelet. Shapelet can make classification faster, more accurate and interpretable. Therefore, there are some works that use shapelet for time series classification. Xing et al. [36] proposed the early distinctive shapelet classification (EDSC) algorithm. EDSC used shapelet with earliness to classify new time series, but its performance is undesirable when applied on larger dataset. Karlsson et al. [16] proposed a random forest algorithm based on shapelet for early classification, which can obtain high accuracy and earliness. As EDSC did not estimate the confidence of the prediction result, Ghalwash et al. [9] presented the modified EDSC with an uncertainty estimates (MEDSC-U) algorithm, which estimated the temporal uncertainty associated with the prediction. Wang et al. [32] proposed a new end-to-end deep learning framework Earliness-Aware Deep Convolutional Networks(EA-ConvNets) for early classification on time series. This framework can jointly perform feature learning and a dynamic truncation model learning to help deep feature learning architecture focusing on the early parts of each time series.

There are many researches on early classification for multivariate time series, Galwash et al. [8] proposed an early classification method called multivariate shapelets detection(MSD) for multivariate time series. MSD extracts shapelets from all dimensions of the multivariate time series. The shapelet in each dimension starts from the same position and has the same length. Furthermore, MSD uses weighted information gain based utility score to evaluate the effectiveness of shapelets, which indicates the precision and earliness of shapelets. MSD is developed based on an assumption that all the multivariate shapelets have the same starting position and the same length, making many shapelet candidates lost. He et al. [13] proposed an early classification method on multivariate time series, called Mining Core Feature for Early Classification (MCFEC). In MCFEC, a new shapelet quality assessment method was proposed, so as to ensure the accuracy and the earliness. Furthermore, MCFEC designs two methods for building an early prediction of the class of unknown multivariate time series object. Further, He et al. [14] proposed an adaptive classification ensemble method to deal with early classification on inter-class and intra-class imbalanced MTS data. They first design an adaptive ensemble framework to learn an early classification model. Second, they introduce a cluster-based shapelet selection method to obtain optimal shapelets. Finally, they design an associate-pattern mining approach to learn base classifiers.

3 Definition and preliminaries

In this section, we will give some terminology definitions and the preliminaries.

3.1 Definition

Definition 1 Univariate Time Series. Univariate time series $T = \{t_1, t_2, t_3, \ldots, t_L\}$ is an ordered set of L real-valued variables. Data points t_1,t_2,t_3, …, t_L are typically arranged by temporal order, and t_i is the value at time stamp i.

Definition 2 Subsequence of time series. Given a time series T of length L, a subsequence s of T is a sampling from $l(l \leq L)$ continuous positions of T. That is, $s = \{t_p, \ldots, t_{p+l-1}\}$, $1 \leq p \leq L - l + 1$.

Definition 3 Distance between the time series. Given two time series T and R with the same length L, the distance between T and R can be denoted by Dist(T,R). By using Euclidean distance as measure, Dist(T,R) is calculated as follows.

$$Dist(T, R) = \sqrt{\frac{1}{L}\sum_{i=1}^{L}(t_i - r_i)^2} \tag{1}$$

Definition 4 Shapelet. Shapelet is a time series subsequence which is in some sense maximally representative of a class, and can be represented by a triple $f = (s, \delta, c)$, where s is a time series subsequence, δ is a distance threshold, and c is the class label.

Definition 5 Best Match Distance (BMD). The best match distance between shapelet $f = (s, \delta, c)$ and time series T is defined as:

$$BMD(f, T) = \min(Dist(s', s)), s' \in s_T^{|s|} \tag{2}$$

where, $s_T^{|s|}$ is a collection of subsequence with length $|s|$ in T.

Given a shapelet $f = (s, \delta, c)$ and a time series T, if $BMD(f, T) \leq \delta$, T is classified to class c.

Definition 6 BMD-list. Given a shapelet f and a training dataset D, which contains r time series, BMD-list is a list of the BMDs between f and the time series in D, sorted in a non-descending order, as shown in Eq. 3:

$$V_f = \{(d_1, c_1), (d_2, c_1), \ldots, (d_r, c_2)\} \tag{3}$$

where $d_i = BMD(f, T_i)$, $T_i \in D$, $d_i \leq d_j$ for $i < j$, and c_i is the class label of time series T_i.

Definition 7 Earliest Match Length (EML). Given a shapelet $f = (s, \delta, c)$ and a time series T, EML is defined as the minimal identifiable length of T. It means f could classify the time series T using its prefix from the beginning to the position EML(f,T).

$$EML(f, T) = \min_{len(s) \leq i \leq len(T)} (Dist(T[i - len(s) + 1, i], s) \leq \delta) \tag{4}$$

EML measures the earliness of f in correctly classifying T. If f cannot classify the time series T, we have $EML(f, T) = \infty$.

Definition 8 Weight Recall (WRecall). WRecall is defined to measure the earliness of shapelet f on a training dataset D.

$$WRecall(f) = \frac{1}{\|D_{\bar{c}}\|} \sum_{T \in D} \frac{1}{\sqrt[\alpha]{EML(f, T)}} \tag{5}$$

where α is used to control the importance of earliness, and $\|D_{\bar{c}}\|$ represents the number of non-target time series.

Definition 9 Precision. Given a shapelet $f = (s, \delta, c)$ and a training set D', precision is defined as the proportion of time series in D' that f can classify correctly.

$$Precision(f) = \frac{\|BMD(f, T) \leq \delta \wedge C(T) = c\|}{\|BMD(f, T) \leq \delta\|}, T \in D \tag{6}$$

Definition 10 Utility. Utility is defined as the quality score of shapelet $f = (s, \delta, c)$. A higher utility means a better quality of the shapelet.

$$Utility(f) = \frac{2 \times Precision(f) \times Wrecall(f)}{Precision(f) + Wrecall(f)} \tag{7}$$

3.2 Preliminaries

The concept of shapelet was first proposed in 2009 by Ye and Keogh [37]. Shapelet is a time series subsequence, which is in some sense maximally representative of a class. Ye and Keogh proposed a brute force search method, which can be used to find the optimal shapelet. However, early classification on time series needs to find multiple effective shapelets. Most shapelet based early classification algorithms first extract and estimate all the shapelets, and then select some good shapelets through a selection strategy. In shapelet extraction phase, two parameters $minL$ and $maxL$ need to be set first, and then extract all the possible shapelets from the training set whose lengths are between $minL$ and $maxL$. The procedure of generating the shapelet candidates is shown in Algorithm 1.

Algorithm 1 GenerateShapeletLibrary.

```
Input: $training\ set : D, minL, maxL$
Output: $a\ set\ of\ shapelet\ candidates : shapeletLibrary$
 1: $shapeletLibrary \leftarrow \varnothing$
 2: for $each\ time\ series\ T$ in $D$ do
 3:     for $l = minL$ to $maxL$ do
 4:         for $i = 1$ to $length(T) - l + 1$ do
 5:             $s_i^l \leftarrow \{t_i, t_{i+1}, \ldots, t_{i+l-1}\}$
 6:             $candidates \leftarrow GenerateShapelet(s_i^l, l, i, D)$
 7:             if $candidate\ satisfies\ conditions$ then
 8:                 $shapeletLibrary.add(candidate)$
 9:             else
10:                 $eliminate(candidate)$
11:             end if
12:         end for
13:     end for
14: end for
15: return $shapeletLibrary$
```

Given a training dataset D, Algorithm 1 extracts all the shapelet candidates (lines 2-14). In line 6, the distance threshold and utility of each candidate are obtained by $Generateshapelet()$ (shown in Algorithm 2). In lines 7-11, we remove part of shapelet candidates according to a precision threshold. At last, Algorithm 1 returns a shapelet candidates set (line 15).

Algorithm 2 GenerateShapelet.

```
Input: $subsequence : s_i^l, length\ of\ subsequence : l, start\ position : i, training\ set : D$
Output: $a\ shapelet\ candidate$ : shapelet
 1: $bmdlist \leftarrow \varnothing$
 2: for $each\ time\ series\ T$ in $D$ do
 3:     $bmd \leftarrow BMD(s_i^l, T)$
 4:     $bmdlist.add(bmd)$
 5: end for
 6: $delta \leftarrow computeDelta(bmdlist)$
 7: $precision \leftarrow computePrecision(bmdlist, delta)$
 8: $wRecall \leftarrow computeWrecall(s_i^l, D, delta)$
 9: $utility \leftarrow 2 \times precision \times wRecall/(precision + wRecall)$
10: return $shapelet(s_i^l, l, i, delta, precesion, utility)$
```

Algorithm 2 describes how to generate shapelet. It first computes the $bmdlist$ of each shapelet (lines 1-6), and obtains $delta$ according to $bmdlist$. Then, the $precision$ and $wRecall$ are computed by $computePrecision()$ and $computeWrecall()$ (lines 7-8). Finally, the $utility$ of each shapelet is obtained, which indicates the quality of shapelet (line 9).

4 Feature extraction

The workflow of IEDSC is shown as Figure 1. First, we generate shapelets from time series training data, by using the generation methods as shown in Algorithms 1 and 2 in Section 3. Algorithm 1 returns a shapelet candidates set, while Algorithm 2 describes how to generate shapelet. Second, we make feature extraction from the generated shapelets candidates which is presented in Section 4. Concretely, we propose the Trend-based Euclidean distance and shapelet pruning technique. Third, we select features from the extracted features and propose the diverse shapelet selection method in Section 5. Finally, we use the selected feature to make early classification on testing time series data.

In the feature extraction phase, in order to evaluate each shapelet, we need to compute the similarity between a shapelet and a time series. Most algorithms use Euclidean distance to measure the similarity. However, Euclidean distance simply calculates the point-to-point distance, without considering the trend of time series. In order to more precisely measure time series similarity we propose a trend-based Euclidean distance. In addition, some shapelet extraction methods extract all the shapelets directly, which can be time consuming. If the dataset is large, the shapelet candidate space will become extremely large. To solve this problem, we propose a shapelet pruning method, which can effectively reduce the number of shapelet candidates.

In this section, we first present the trend-based Euclidean distance, and then introduce the distance threshold calculation method. Finally, we introduce the shapelet pruning technique based on the prediction on the starting position.

4.1 The trend-based Euclidean distance

To measure the similarity of time series, we need compute the distance between two time series, which usually presents some trend changing over time. To take the changing trend of the time series data into consideration, we propose a trend-based Euclidean distance computation method. The idea is that if two time series present similar trend and the distance between them is small, we consider that the two time series are similar. The similarity can be calculated by:

$$TDist(T,R) = \sqrt{\frac{1}{L}\left(\sum_{i \in I_s}(t_i - r_i)^2 + \sum_{j \in I_{\bar{s}}}(t_j - r_j)^2 * \lambda\right)} \tag{8}$$

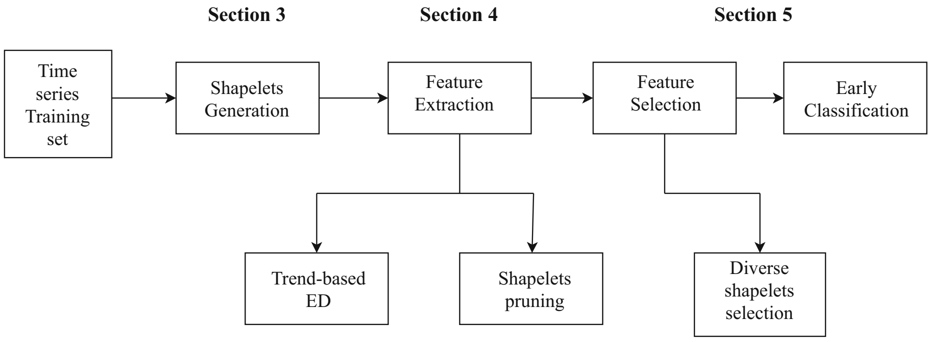


Figure 1 Workflow of the proposed algorithm IEDSC

where T and R are two time series with the same length L. t_i, r_i, t_j, and r_j represent the value of time series at time stamp i or j. All the time stamps in I_s indicate that the size relationship between T and R at time stamp i are the same as the last time stamp. $I_{\bar{s}}$ is the opposite of I_s.

Algorithm 3 distWithTrend.

```
Input: $series\ length : L,\ user\ defined\ coefficient : \lambda,\ two\ series : s, t$
Output: $dist$
 1: $dist \leftarrow (s_1 - t_1)^2$
 2: $flag \leftarrow \text{sgn}(s_1 - t_1)$
 3: for $i = 2 \rightarrow L$ do
 4:     if $sgn(s_i - t_i) == flag$ then
 5:         $dist \leftarrow dist + (s_i - t_i)^2$
 6:     else
 7:         $dist \leftarrow dist + (s_i - t_i)^2 * \lambda$
 8:     end if
 9:     $flag = sgn(s_i - t_i)$
10: end for
11: $dist \leftarrow sqrt(dist/L)$
12: return $dist$
```

The specific steps of trend-based Euclidean distance computation algorithm are shown as Algorithm 3. Given two time series, we first calculate the distance between the data points of the two sequences at the first time stamp and set it as the initial value of $dist$, and then calculate the size relationship between them (lines 1-2). We use $sgn()$ to represent the size relationship. $sgn()$ is a symbolic function that returns the positive or minus of a parameter. When calculating the distance between two points of two sequences at time stamp i , it is necessary to consider the size relationship between the two points at the previous time stamp $i - 1$. If the size relationship between two points at time stamp i of two sequences is the same as that at time stamp $i - 1$, the distance between points of the two sequences at time stamp i is added to $dist$ directly; otherwise the distance should be punished with the parameter $\lambda \in [1, 2]$, and then added to $dist$ (lines 3-9). The distance between two sequences will increase with the increase of λ, so the distance between two shapelets with similar trend will be small. Therefore, the similarity measure is more accurate. If the two sequences have similar trend, which means that the size relationship between two points at each time stamp of two sequences could be the same. If two points at any time stamp do not satisfy this condition, the extent of similarity is diminished. So we amplify the distance between the two points. Finally, the algorithm returns the final distance (line 12).

Example 1 Suppose there are 5 time series with length 10, $A = \{0.5, 0.9, 2, 2.5, 4.5, 3.9, 1.2, -0.5, -0.3, 0.6\}$, $B = \{1, 1.1, 2.4, 2.8, 5.5, 4.4, 1.6, 0, 0.2, 2.1\}$, $C = \{0, 0.7, 1.6, 2.2, 3.5, 3.4, 0.8, -0.8, -0.8, 0.2\}$, $D = \{0, 0.5, 2.4, 2.8, 5.5, 4.4, 0.8, -0.2, 0.2, 0.8\}$, $E = \{0.1, 1.4, 2.3, 2.9, 4.3, 4.2, 0.7, 0.5, 0.1, 0.1\}$.The trend relationship between A and the others is shown in Figure 2.

In Figure 2, if the Euclidean distance is used to calculate the distance between sequence A and the others, the distances are the same as the $\sqrt{0.245}$. But from Figure 2 one can see that the two curves in Figure 2 (1)(2) are closer than that in Figure 2 (3)(4). If we use

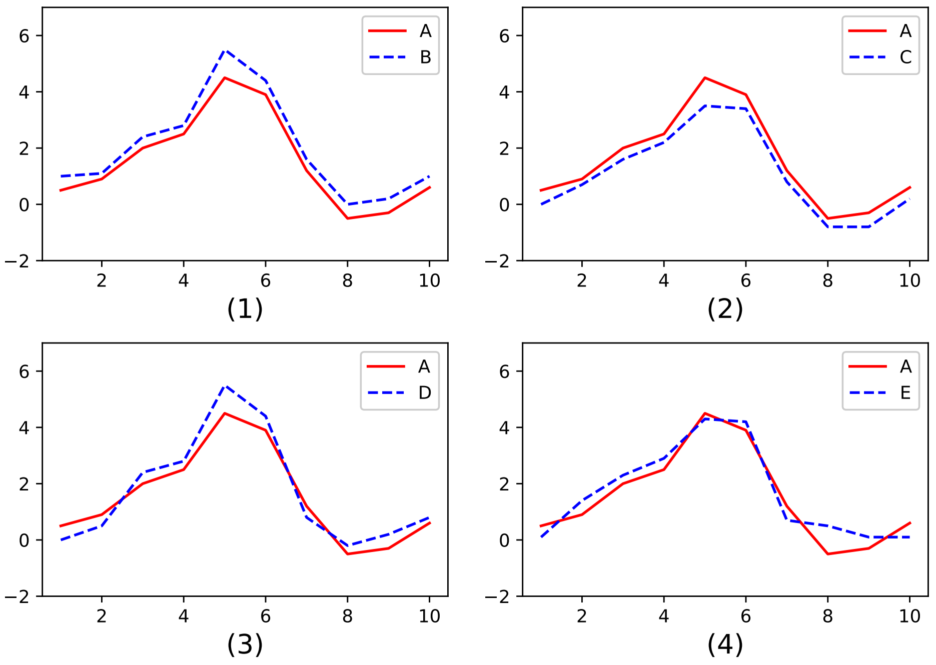


Figure 2 Trend relations between A and other four time series

Algorithm 3 to compute the trend-based Euclidean distance between the sequences, their distances are different. The largest distance is the distance between curves A and E, which is $\sqrt{0.247}$. The distance between curves A and B is the same as the distance between curves A and C, which is $\sqrt{0.245}$. The distance between curves A and D is $\sqrt{0.246}$. This example shows that our proposed trend-based Euclidean distance can better measure the similarity between time series compared with Euclidean distance.

4.2 Compute the distance threshold

The distance threshold is a very important attribute for shapelet, which is used to classify time series. Before calculating the distance threshold, we first get the BMD-list of shapelet, and then choose the best threshold according to the evaluation strategy.

Ye et al. [37] used information gain to evaluate the quality of the threshold, which divided the training dataset into two subsets. A higher purity of the two subsets leads to a higher information gain, indicating that the quality of this threshold is better. However, the limitation of this method is that it is prone to overfitting. Xing et al. [36] used kernel density estimation [6] and Chebyshev inequality to compute the distance threshold. Given a shapelet $f = (s, \delta, c)$ and a time series T, if $BMD(f, T) \leq \delta$, the kernel density estimation can guarantee the probability of time series T belonging to the target class is higher than a user-defined threshold. The Chebyshev inequality can guarantee the probability of the time series belonging to non-target class is lower than a user-defined threshold. He et al. [13] considered the accuracy and recall simultaneously, and obtained the best threshold when the harmonic mean of accuracy and recall was maximized. We use kernel density estimation to calculate the distance threshold of shapelet, because the distance threshold obtained by kernel density estimation is more accurate.

Given a sample $y = \{x_1, x_2, \ldots, x_n\}$, the kernal density estimation of $f(X = x)$ can be calculated as follows:

$$f(X = x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{9}$$

$$K\left(\frac{x - x_i}{h}\right) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-x_i)^2}{2h^2}} \tag{10}$$

$$h = 1.06\delta n^{-\frac{1}{5}} \tag{11}$$

where K is a Gaussian kernel function, h is a smoothing factor, and δ is the standard deviation of the sample.

If the training set contains m classes , $c \in \{1, \ldots, m\}$, for a threshold x, we can get the probability of x belonging to class i as follows.

$$Pr(c_x = i | X = x) = \frac{p_i f_i(x)}{\sum_{k=1}^{m} p_k f_k(x)} \tag{12}$$

where p_k is a prior probability of class k .

Given a shapelet $f = (s, ?, c)$ and the BMD-list of f, we can use kernel density estimation to obtain a distance threshold δ , satisfying that if $BMD(f, T) \leq \delta$, we have $Pr(c_x = c|X = x) \geq \beta$. Here, T is the time series in training set and β is a user-defined probability threshold.

4.3 Shapelet pruning based on the estimated starting position

In the conventional method of shapelet extraction, the number of shapelet candidates is extremely huge, and it discovers optimal shapelet by estimating the whole shapelet candidates. As it is time consuming to scan all the candidates, we argue that we do not need to estimate some shapelets with bad quality, and pruning them can significantly reduce the computational complexity. In this paper, we propose a shapelet pruning algorithm based on the prediction on the starting position. The basic idea is to estimate the starting positions of shapelets with good quality and then extract shapelets from the starting positions. Here we preset a user-defined parameter, $stepSize$, which is used to change the length of shapelets in shapelet extraction. In our method, we estimate the start positions according to part of shapelets, and the lengths of shapelets vary. In order to make the number of extracted shapelets appropriate, we extract part of shapelets with different lengths according to $stepSize$. The pruning process is as follows. First, we extract all the shapelets with length $minL$, and then extract all the shapelet candidates with length $minL + stepSize$. Next time we add $stepSize$ to the previous shapelet length until $maxL$. Second, we evaluate the extracted shapelets candidates and then select the better shapelets from these shapelet candidates. We take one of the extracted shapelets candidates as a feature denoted by f_1. All training examples covered by f_1 are marked. We consider the remaining unmarked shapelet candidates that can cover some training examples as covered, and iteratively select the shapelets candidates as features. The iteration continues until all the training examples are covered to obtain better shapelets. Finally, we extract the starting positions of these shapelets, and only keep the shapelets with lengths between $minL$ and $maxL$ from these starting positions. Note that these lengths do not include the length that has been previously extracted. Thereby we achieve the pruning of shapelets. The pruning method is described as Algorithm 4.

Algorithm 4 GenerateShapeletLibraryWithPruning.

```
Input: $training\ set : D,\ minL, maxL, stepSize$
Output: $a\ set\ of\ shapelet\ candidates : shapeletLibrary$
1: $shapeletLibrary \leftarrow \varnothing$
2: $lenArr \leftarrow \varnothing$
3: $lenArr.add(minL)$
4: $x \leftarrow minL + stepSize$
5: while $x < maxL$ do
6:     $lenArr.add(x)$
7:     $x \leftarrow minL + stepSize$
8: end while
9: $lenArr.add(maxL)$
10: $valShapelets \leftarrow FirstExtractShapelets(D, lenArr)$
11: $topShapelets \leftarrow SelectionShapelet(valShapelets)$
12: $startArr \leftarrow GetStartPos(topShapelets)$
13: $shapeletCandidates \leftarrow ExtractShaplets(D, minL, maxL, lenArr, startArr)$
14: $shapeletLibrary \leftarrow valShapelets \cup shapeletCandidates$
15: return $shapeletLibrary$
```

Given a training set, we first calculate the length array $lenArr$ of the shapelet according to the parameter $stepSize$(lines 3-9). The function $FirstExtractShapelets()$ only returns the shapelets with lengths in the length array $lenArr$ (line 10). $SelectionShapelet()$ and $GetStartPos()$ are used to select the better shapelets and obtain the starting positions of these better shapelets(lines 10-12). After that, according to the starting positions, we extract all the lengths of the shapelets, but those extracted shapelets are no longer extracted (line 13). Finally, lines 14-15 merge the two sets $valShapelets$ and $shapeletCandidates$, and return the final shapelet candidates .

Given a time series with length 80 as shown in Figure 3, suppose $minL = 5$, $maxL = L/2$, where L is the length of the complete time series. According to Algorithm 4, assuming $stepSize$ is 8, we first compute the length array [5,13,21,29,37,40]. We extract shapelets with lengths in the length array and evaluate these shapelets. Then we select the better shapelets (the red sequence in Figure 3) and obtain their starting positions (the red dots in Figure 3). Finally, we start to extract shapelets from these positions, for reducing the number of shapelets.

In the feature extraction step, we assume all the shapelet of length between $minL$ and $maxL$. Suppose we have N time series with length L, and a parameter $stepSize$ which is used to obtain $lenArr$. For a shapelet with length k, the computation cost is $O(kN(L-k+1))$. We need extract shapelets twice, the cost of first extraction is $O(\sum_{k \in lenArr} kN^2(L-k+1)^2)$. Supposing we obtain M starting positions from the first step, the cost of the second extraction is $O(M\sum_{k=minL}^{maxL} kN^2(L-k+1))$. The total cost of extraction is $O(\sum_{k \in lenArr} kN^2(L-k+1)^2 + M\sum_{k=minL}^{maxL} kN^2(L-k+1)) = O(N^2L^4)$.

5 Feature selection and early classification based on shapelet

Feature selection is one of the key steps of IEDSC, and the quality of shapelets determines the accuracy of classification. For early classification, good features should be frequent,

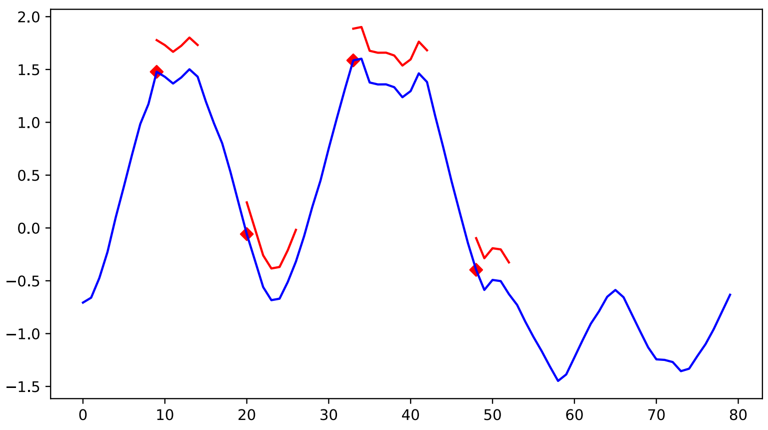


Figure 3 An illustration of the starting position estimation of shapelets

discriminative and of good earliness, and the selected features should be diverse. To this end, we propose a new feature selection method.

Xing et al. [36] uses a greedy method to select shapelets, and the steps are as follows. First, sort all the shapelets in non-ascending order according to the quality score. Second, select the shapelet $f = (s, \delta, c)$ with the highest score, and mark all the samples covered by f in the training set. Here, the covering means $BMD(f, T) \leq \delta \wedge c(T) = c$. Third, select the shapelet $f' = (s', \delta', c')$ with the second highest quality score, and repeat the covering operation on training examples that are not marked. The above steps are repeated until all training examples are marked.

Although the greedy method is simple, it has some disadvantages. The shapelets extracted by this method may be highly similar, and it might ignore some useful shapelets, leading to undesirable classification accuracy. In order to tackle this problem, we design a selection method according to the concept of non-self match [18] and trivial match [2, 3].

Definition 11 Non-Self Match. Given a time series T, containing a subsequence C of length n beginning at position p and a matching subsequence M beginning at q, we say that M is a non-self match to C at distance of $Dist(M, C)$ if $|p - q| \geq n$.

Definition 12 Trivial Match. Given a time series T, which contains a subsequence C with a beginning position p and a matching subsequence M with a beginning position q, we say that M is a trivial match to C if either $p = q$ or there does not exist a subsequenceM' beginning at q' such that $D(C, M') > \mathfrak{R}$, and either $q < q' < p$ or $p < q' < q$.

For two shapelets, if they derive from the same time series and their starting positions are nearby, they should be similar. The similarity between shapelets is shown in Figure 4. In Figure 4, there are five subsequences with the same length and different starting positions. The starting positions of subsequences a, b, c, d, e are 19, 20, 21, 22, 23 respectively. From Figure 4, we can see that two shapelets are very similar if their starting positions are 1 or 2 apart.

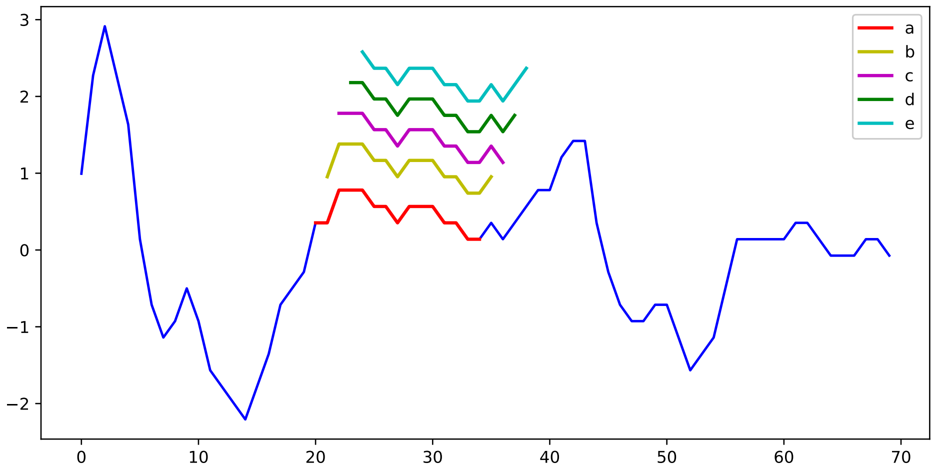


Figure 4 Shapelets with Similarity

5.1 Diverse shapelet selection

In order to remove similar shapelets and keep diverse shapelets, we define extended self-similarity below.

Definition 13 Extended Self-Similarity. Given two shapelets $f_1 = (s_1, \delta_1, c_1)$ and $f_2 = (s_2, \delta_2, c_2)$. We add three attributes Id, staPos and len to shapelets. Id represents the time series ID that we extract the shapelet from. staPos is the starting position in time series and len is the length of shapelet. We say $f1$ and $f2$ have extended self-similarity if the following conditions are satisfied.

1. $Id_1 = Id_2$;
2. $|staPos_1 - staPos_2| \leq \gamma$;
3. $|len_1 - len_2| \leq \eta$.

where, γ and η are user-defined threshold, γ denotes the allowed distance between the starting positions of two shapelets, and η represents the allowed difference of two shapelet lengths.

We illustrate extended self-similarity by the examples shown in Figure 5, assuming $\gamma = 1$ and $\eta = 1$. In Figure 5 (1), the starting position of c is the same as $c1$ and $c2$, but their lengths are different. c and $c1$, c and $c2$ both have a length difference of 1. Therefore, according to Definition 13, c and $c1$, c and $c2$ have extended self-similarity. In Figure 5 (2), the starting positions of c, $c3$, and $c4$ are different, and the starting position distance between c and $c3$, c and $c4$ is 1, but their lengths are identical. So from Definition 13 we know c and $c3$, c and $c4$ have extended self-similarity. In Figure 5 (3), c, $c5$ and $c6$ have different starting positions and lengths, but they satisfy the conditions in Definition 13. Thus c and $c5$, c and $c6$ have extended self-similarity. In Figure 5 (4), c, $c7$ and $c8$ also have various starting positions and lengths. c and $c7$ have a starting position distance of 1 and length difference of 2, and c and $c8$ are the same. Therefore, c and $c7$, c and $c8$ do not have extended self-similarity.

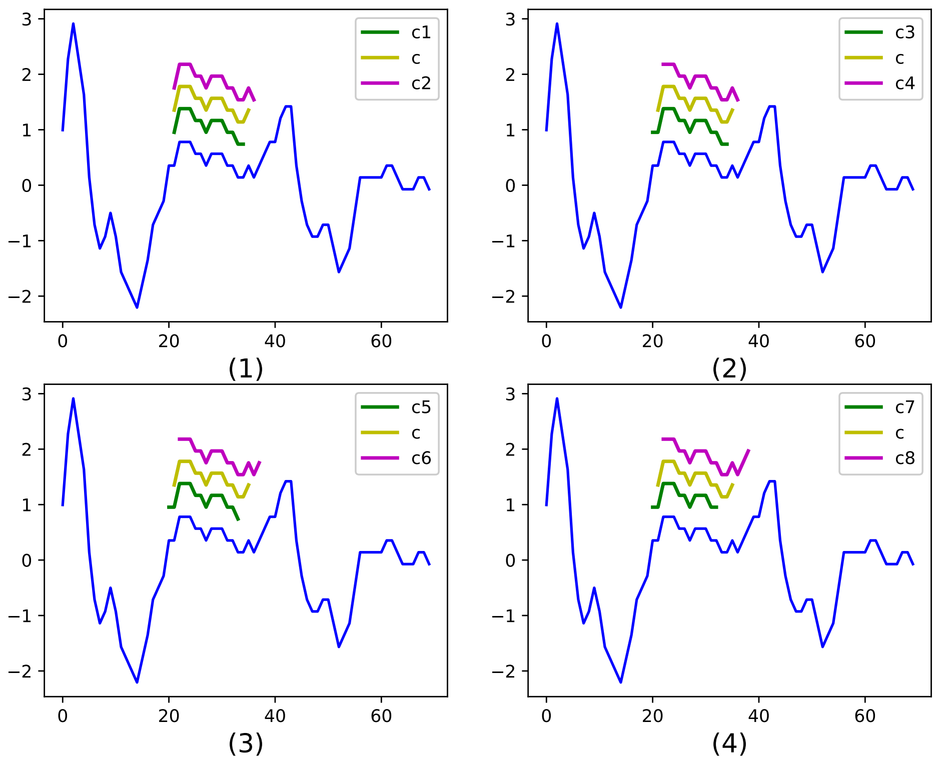


Figure 5 An Example of Extended Self-Similarity

The traditional feature selection algorithm does not consider the similarity between features, and the results might contain some similar features. In order to avoid this problem, we propose a new feature selection algorithm that selects features with larger diversity. The main idea is that when selecting a new feature, we first judge whether there is extended self-similarity between the current feature and the extracted features. If there exists, we omit it, otherwise, we use it to cover the samples.

The steps of the new feature selection algorithm are as follows:

1. Sort all the shapelets in non-ascending order according to the utility.
2. Take the shapelet $f = (s, \delta, c)$ with the highest utility, and then mark all the samples covered by f in the training set. The cover means $BMD(f, T) \leq \delta \wedge c(T) = c$.
3. Take the shapelet $f' = (s', \delta', c')$ with second highest utility, judge whether the current shapelet has extended self-similarity to the extracted shapelets. If there exists, omit this shapelet and select the next shapelet to repeat this step; if not, cover the unlabeled sample in the dataset, and then mark the samples covered by f' until most samples are marked.

5.2 Early classification method

The proposed early classification method is shown as Algorithm 5.

Algorithm 5 PredictClassLabel.

```
Input: $time\ series : T,\ shapelets$
Output: $the\ prediction\ class\ of\ T : c$
1: $shortlen \leftarrow shortestLengthInShapelets(shapelets)$
2: for $i = shortlen \rightarrow length(T)$ do
3:     for $f\ in\ shapelets$ do
4:         if $BMD(f, T) \leq \delta$ then
5:             $return\ c$
6:         end if
7:     end for
8: end for
9: return $noclass$
```

Given a sample in testing set D', classifier first obtains the minimum length of the extracted $shapelets$ (line 1), and then tries to match the time series with each shapelet in the $shapelets$. If there is a match, return the class label, otherwise get more data and repeat this operation (lines 2-8). If it is not possible to classify at the last time stamp, the sample is unclassified.

Given the selected m shapelets with length k, the number of the samples to be classified is $|D'|$ with length L, the online classification complexity is $(L - k + 1) * k * m * |D'|$.

6 Experimental evaluation

6.1 Experimental settings

We compare IEDSC with five baselines, 1NN-DTW(learned-w) [17], ECTS [35], EDSC [36], ECDIRE [25] and RelClass [26]. 1NN is a nearest neighbor based classification algorithm with complete time series, and we use Dynamic Time Warping(DTW) to measure the similarity of time series due to its good performance, where the warping windows are obtained after learning the best constraint from the training set. ECTS is the first early classification method, which uses partial time series for classification. EDSC is an early classification algorithm based on shapelet, which achieves good accuracy and interpretablity. ECDIRE uses probabilistic classifier to classify time series at time stamps that are discovered in learning phase. RelClass treats data as a random variable and makes decision depending on a reliability threshold.

The UCR time series data library [4] collects a number of standard time series datasets that are widely used in time series classification, each containing a training set and a test set. We select the following 16 datasets which are the representative datasets of different types from the UCR time series data library for evaluation and show the parameter setting of each dataset in Table 2.

We use accuracy and earliness as the performance evaluation metrics. The accuracy represents the proportion of samples that the classifier can correctly classify in the testing set.

$$Accuracy(EC) = \frac{NUM}{|D'|} \tag{13}$$

where, EC is the early classifier, D' is a testing dataset, $|D'|$ is the number of samples in the testing dataset, and NUM represents the number of samples that classifier EC correctly classifies.

Table 2 Datasets description

Dataset information					Parameter setting
Dataset	Training	Testing	Class	Length	λ
CBF	30	900	3	128	1.0319
Coffee	28	28	2	286	1.1
ECG200	100	100	2	96	1.0319
ECGfivedays	23	861	2	136	1.1
Facefour	24	88	4	350	1.1
FacesUCR	200	2050	14	131	1.3
Gun_Point	50	150	2	150	1.4
Motestrain	20	1252	2	84	1.2
Oliveoil	30	30	4	570	1.0319
SonyAIBOrobotsurface1	20	601	2	70	1.1
SonyAIBOrobotsurface2	27	953	2	65	1.1
Symbols	25	995	6	398	1.4
Synthetic_control	300	300	6	60	1.1
TwoleadECG	23	1139	2	82	1.0319
Two_Patterns	1000	4000	4	128	1.0319
Wafer	1000	6164	2	152	1.0319

Earliness is defined as the average minimum prediction length on time series set in which the time series are classified.

$$Earliness = \frac{1}{N}\sum_{i=1}^{N}\frac{l_i}{L} \tag{14}$$

where N represents the number of samples that could be classified, l_i is obtained by the EML(Eq. 4), which indicates the minimal indentifiable length of ith time series, and L represents the length of the complete ith sequence.

All the experimental results are obtained on the computer with Intel (R) Core (TM) I5-7400 , 3.0 GHZ and memory 8G. The algorithms are implemented by Java.

6.2 Performance comparison over accuracy and earliness

In the experiments, we set $minL = 5$, $maxL = L/2$, where L is the length of the complete time series. The two length parameters are set empirically, and all the datasets in experiment have the same $minL$ and $maxL$. In the distance calculation, we use the training set of each dataset to make cross validation and obtain the value of λ. The search space of λ is in [1,2] and we make 5-fold cross validation on the training set of each dataset to find the optimal value of λ shown in Table 2. When we evaluate the shapelets, we remove some shapelets whose precision obtained by a distance threshold is lower than 0.8. This is also an empirical value and is fixed for all datasets. When calculating the distance threshold, we set $\alpha = 3$, $\beta = 0.95$. The values of two parameters are obtained from EDSC [36], which could achieve higher accuracy. In feature extraction, the $stepSize$ is fixed to 8. In feature selection, we set $\gamma = 2$, the value of γ is fixed. The value $\eta \in [5, 6, 7, 8]$, we search the solution space and find there is no significant difference on the results with different η values, and thus we set $\eta = 5$.

We compare the proposed algorithm IEDSC with 1NN-DTW(learned-w), ECTS, EDSC , ECDIRE and RelClass, and the experimental results of accuracy and earliness are shown in Tables 3 and 5 respectively.

From Table 3, we can observe that 1NN-DTW obtains the best accuracy in 7 datasets and has the highest average rank. It is because 1NN-DTW uses the full time series data to make classification. EDSC is based on shapelet, but it only obtains the best result in *GunPoint*.The reason is that EDSC selects shapelet through a simple cover method, and thus the shapelets do not have enough diversity. The ECTS beats all the other methods in 2 datasets. ECTS is based on 1NN method, so the decision time stamp would affect the accuracy. The earlier decision time stamp usually leads to a lower accuracy, while the later decision time stamp leads to a higher accuracy. For ECDIRE, due to the unreliability of class label, some series could remain unclassified. RelClass obtains the best accuracy in two datasets. IEDSC obtains the best results in 5 datasets and achieves the second accuracy average rank, only worse than the full time series method, 1NN-DTW, which does not provide early detection. IEDSC classifies time series by shapelets, and the quality of shapelets decides the accuracy. IEDSC eliminates parts of shapelet candidates in the extraction process, selects more diverse shapelets, and thus it achieves a higher accuracy. Moreover, Figure 6 shows a critical difference diagram for Nemenyi test [5] over the accuracy average ranks of six classifiers. In the critical difference diagram, α is set to 0.05. The critical difference (CD) length is shown above the graph. From Figure 6, we can see that DTW is the best-performing classifier as it is based on full length time series, and IEDSC is ranked the second. There is no significant difference among DTW, IEDSC, RelClass, ECTS and ECDIRE because they form a clique of classifiers.

Table 3 Accuracy

Dataset	1NN-DTW	ECTS	EDSC	ECDIRE	RelClass	IEDSC
CBF	**1(1)**	0.85(4)	0.84(5)	0.89(3)	0.64(6)	0.92(2)
Coffee	**1(1)**	0.75(5.5)	0.75(5.5)	0.96(2)	0.89(4)	0.93(3)
ECG200	0.88(4)	0.89(2.5)	0.85(5)	**0.91(1)**	0.89(2.5)	0.84(6)
ECGfivedays	0.8(2)	0.62(4)	0.74(3)	0.6(5)	0.52(6)	**0.97(1)**
FaceFour	0.89(2)	0.82(4)	0.75(5)	0.61(6)	0.83(3)	**0.92(1)**
FacesUCR	**0.91(1)**	0.71(5)	0.63(6)	0.74(4)	0.77(2)	0.76(3)
GunPoint	0.91(3.5)	0.87(5.5)	**0.94(1.5)**	0.87(5.5)	0.91(3.5)	**0.94(1.5)**
Motestrain	0.87(2)	**0.88(1)**	0.78(4)	0.8(3)	0.58(6)	0.76(5)
Oliveoil	0.87(2)	**0.9(1)**	0.6(5)	0.4(6)	0.77(3.5)	0.77(3.5)
SonyAIBOrobotsurface1	0.7(5)	0.69(6)	0.8(3)	0.83(2)	0.79(4)	**0.84(1)**
SonyAIBOrobotsurface2	0.86(2)	0.85(3)	0.81(5)	0.74(6)	**0.88(1)**	0.82(4)
Symbols	**0.94(1)**	0.81(2.5)	0.51(6)	0.81(2.5)	0.71(4)	0.59(5)
SyntheticControl	**0.98(1.5)**	0.88(6)	0.89(5)	0.96(3)	**0.98(1.5)**	0.92(4)
TwoLeadECG	0.87(3)	0.73(5)	0.88(2)	0.81(4)	0.72(6)	**0.95(1)**
TwoPatterns	**1(1)**	0.86(5)	0.8(6)	0.87(4)	0.93(2)	0.9(3)
Wafer	**1(1)**	0.99(3)	0.99(3)	0.97(5)	0.99(3)	0.9(6)
Average rank	2.0625	3.9375	4.375	3.875	3.625	3.125

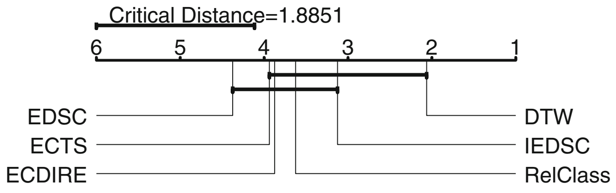


Figure 6 Critical difference diagram of accuracy for six classifiers

In order to further investigate whether the five early classification methods present similar performance, we conduct Wilcoxon rank sum test to evaluate the significance level of ranks on accuracy and earliness. Table 4 shows the results of rank sum test of significance level of the ranks on accuracy, wherein, $\overline{\alpha}$ is the significance level which is set to 0.05 for test. The p-value is the significant probability that the pairwise is consistent. The last column h denotes the test result. If h is equal to 1, it indicates the pairwise has significance difference. From Table 4, one can observe that the pairwise EDSC vs. IEDSC are significantly different, while the others do not present significant difference.

Table 5 shows the result for earliness. It shows that IEDSC obtains the lowest values in 5 datasets out of 16, followed by EDSC, which obtains the best results in 4 datasets. ECDIRE and RelClass obtain the best resluts in only 3 datasets. ECTS does not obtain the best result in any dataset. Table 6 shows the results of rank sum test of significance level of the ranks on earliness. One can see that ECTS vs. IEDSC has significance difference, and the pairwise RelClass vs. IEDSC also has significance difference.

From the above analysis, we can conclude that IEDSC could not only make prediction as early as possible, but also use partial data to achieve comparable accuracy with the classifier using complete time series.

6.3 Method effectiveness analysis

IEDSC contains three key designed features, trend-based Euclidean distance, shapelet pruning, and diverse shapelet selection. In order to evaluate how these three parts affect the classification performance, we conduct the experiments by changing one of these three parts and keeping the others unchanging.

We conduct the experiments on four datasets, i.e., $ECG200$, SonyAIBORobotSurface2, $ECGfivedays$ and $FaceFour$. We set $\lambda = 1.1$ for $ECG200$ and SonyAIBORobotSurface2 datasets, and $\lambda = 1.2$ for $ECGfivedays$ and $FaceFour$ datasets.

Figure 7 shows the accuracy of four datasets with Euclidean distance and Trend-based Euclidean distance. We can observe that, the accuracy of $ECG200$ and SonyAIBORobotSurface2 datasets has improved, while the accuracy of $ECGfivedays$ and $FaceFour$ datasets does not change much.

Table 4 Wilcoxon Rank Sum Test of Significance Level of Average Ranks on Accuracy

No.	Hypothesis	α	p-value	h
1	ECTS vs IEDSC	0.05	0.2161	0
2	EDSC vs IEDSC	0.05	0.0468	1
3	ECDIRE vs IEDSC	0.05	0.2379	0
4	RelClass vs IEDSC	0.05	0.4025	0

Table 5 Earlinesss

Dataset	ECTS	EDSC	ECDIRE	RelClass	IEDSC
CBF	71.5(5)	31.85(4)	28.55(3)	**23.08(1)**	27.41(2)
Coffee	83.94(5)	54.23(3)	82.14(4)	38.44(2)	**35.67(1)**
ECG200	60.11(3)	**23.24(1)**	90.1(5)	68.81(4)	26.91(2)
ECGfivedays	63.82(5)	53.6(3)	21.07(2)	**15.84(1)**	61.96(4)
Facefour	72.26(5)	47.98(4)	**22.31(1)**	34.22(2)	38.14(3)
FacesUCR	87.21(4)	**51.58(1)**	59.15(2)	92.71(5)	63.55(3)
GunPoint	46.92(4)	45.58(2)	32.37(1)	71.33(5)	46.56(3)
Motestrain	79.06(4)	38.08(2)	**12.1(1)**	90.94(5)	43.83(3)
Oliveoil	87.34(5)	38.82(4)	30(2)	**18.76(1)**	36.3(3)
SonyAIBOrobotsurface1	68.49(5)	47.03(2)	62.26(4)	57.7(3)	**26.87(1)**
SonyAIBOrobotsurface2	54.54(4)	35.51(3)	**17.66(1)**	70.86(5)	34.83(2)
Symbols	51.3(4)	60.25(5)	45.33(2)	45.82(3)	**31.27(1)**
SyntheticControl	87.88(5)	50.81(2)	61.92(3)	71.54(4)	**46.8(1)**
TwoleadECG	64.43(3)	**46.85(1)**	69.38(4)	83.63(5)	52.01(2)
TwoPatterns	86.52(3)	**64.04(1)**	98.76(5)	91.82(4)	67.8(2)
Wafer	44.38(5)	27.99(3)	10.87(2)	30.75(4)	**10.8(1)**
Average rank	4.313	2.563	2.625	3.375	2.125

Figure 8 shows the accuracy of four datasets without and with pruning methods. The accuracy of $ECG200$ increased by 5%. For the $SonyAIBORobotSurface2$, the accuracy increases from 0.81 to 0.82. The accuracy of $ECGfivedays$ and $FaceFour$ datasets with pruning also increases. Figure 9 shows the extraction time and the number of shapelet candidates without and with pruning method for four datasets. From Figure 9 (1)(2)(3)(4), we can observe that the extraction time reduces significantly after pruning, and the number of shapelet candidates is also signficantly reduced.

Figure 10 clearly shows that the proposed shapelet selection method improves the classification accuracy. The accuracy of $ECG200$ rises to 0.83, and the accuracy of $SonyAIBORobotSurface2$ increases by 6% with diverse shapelet selection method. The accuracy of $ECGfivedays$ and $FaceFour$ with diverse selection method also has improvement. The reason is that diverse selection method removes some of similar shapelets, and thus makes the shapelets more diverse and makes classifier more robust.

Table 6 Wilcoxon Rank Sum Test of Significance Level of Average Ranks on Earliness

No.	Hypothesis	α	p-value	h
1	ECTS vs IEDSC	0.05	0.0101	1
2	EDSC vs IEDSC	0.05	0.3488	0
3	ECDIRE vs IEDSC	0.05	0.3806	0
4	RelClass vs IEDSC	0.05	0.0188	1

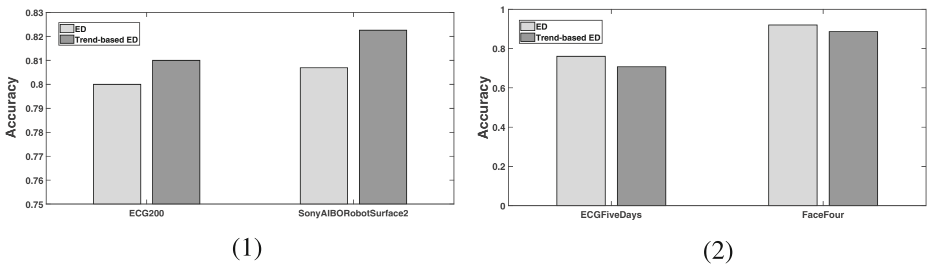


Figure 7 Accuracy with different distance methods

6.4 Interpretability of features

In this section, we use $ECG200$ as an example to demonstrate the selected shapelets by our feature selection method. $ECG200$ dataset consists of the two classes of time series, and the lengh of each time series is 96, as shown in Figure 11. The class 1 represents a normal heartbeat, and the class -1 is an abnormal class which represents a myocardial infarction. In a normal human ECG, there are five waves in a heartbeat cycle, including the PQRST waves shown as Figure 12(1). If a person has a myocardial infarction, it is usually observed from the ECG that the ST wave is changed and elevated.

From Figure 12 (2), one can observe that the shapelet shown as red line is corresponding to the ST wave, and it has a gentle trend which indicates a normal heartbeat. The shapelet shown as blue line in Figure 12 (3) also contains the ST wave, and the elevated ST wave indicates that a person has a myocardial infarction.

Using 1NN-DTW on this dataset with complete data, the accuracy is 0.88. However, IEDSC uses only 26.91% data to obtain a comparable accuracy 0.84. From the above analysis, one can see the selected shapelets could represent the difference between two class time series and achieve a good accuracy.

6.5 Case study: Early classification of CBF

The CBF dataset is one of the most commonly used datasets in time series classification. In order to show the performance of IEDSC, we select the CBF dataset as a case study, and in the training phase we use standard-divided training sets to train the classifier.

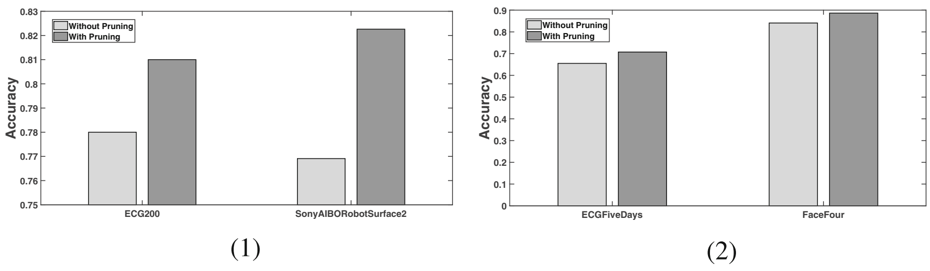


Figure 8 Accuracy of without pruning and with pruning methods

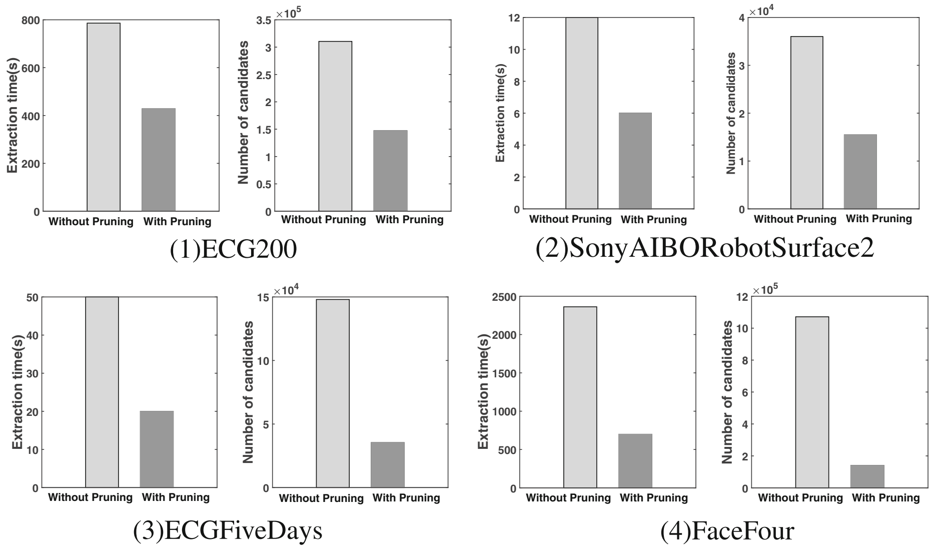


Figure 9 Comparison of without pruning and with pruning methods

CBF contains three classes of time series, and we draw the contours of the time series for these three classes in Figure 13. The accuracy of EDSC on CBF is 84% and the earliness is 31.85%, while the accuracy of IEDSC on CBF is 92% and the earliness is 27.41%. Obviously, our algorithm achieves higher accuracy than EDSC, and also gets the result more earlier. In feature selection phase, EDSC only gets three shapelets from CBF dataset, and in this paper we obtain 19 shapelets, which are displayed and marked out in red in Figures 14, 15 and 16 respectively.

IEDSC selects 10 shapelets from four different time series belonging to class 1 shown as Figure 14 . Shapelets from the same time series have partial differences, but shapelets from disparate time series are distinct. For class 2, there are only one shapelet selected in

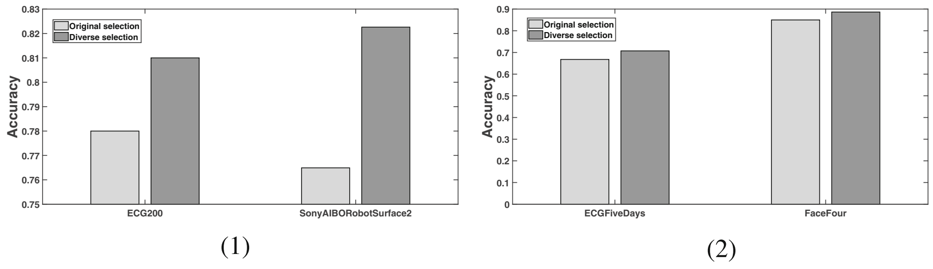


Figure 10 Accuracy of different selection methods

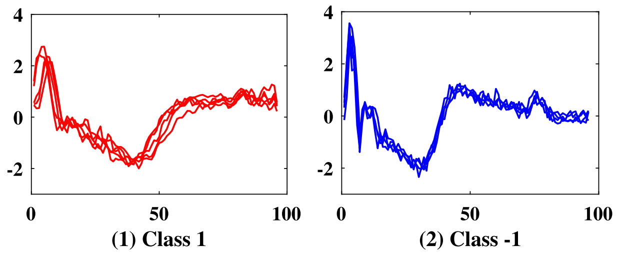


Figure 11 ECG time series

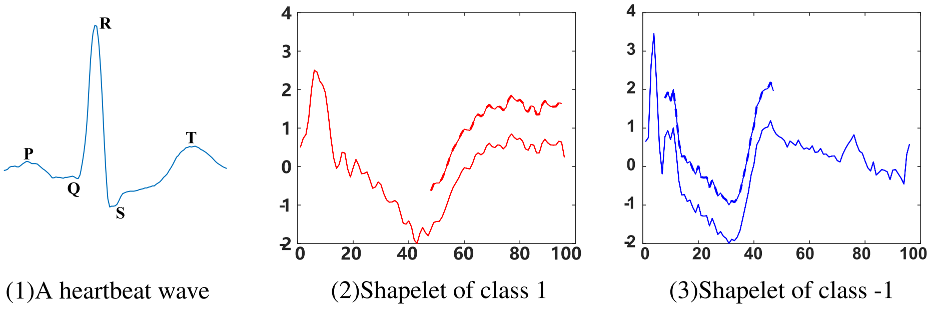


Figure 12 Interpretability of shapelets for classification

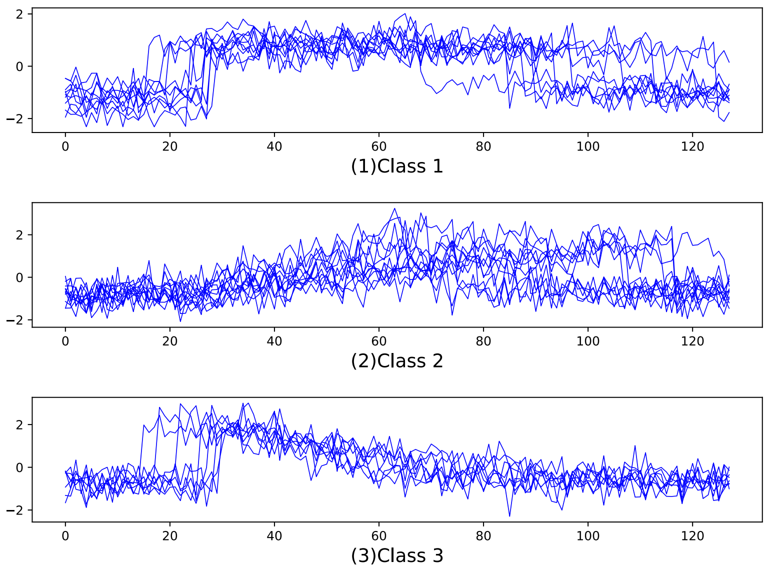


Figure 13 CBF time series

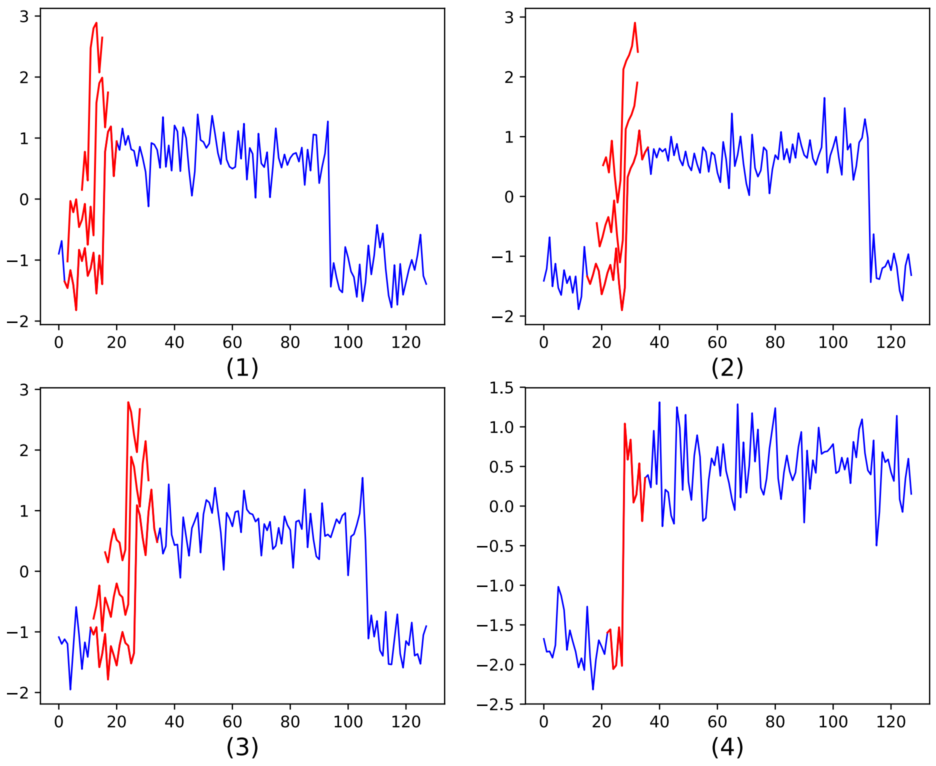
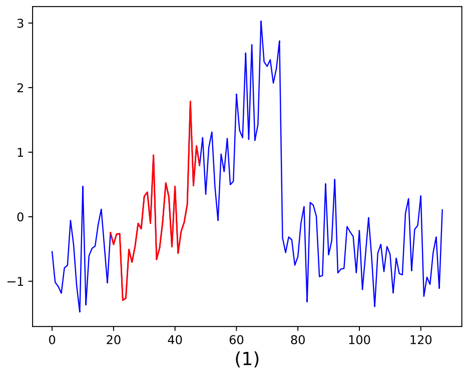


Figure 14 Shapelet of class 1

Figure 15 . Moreover, IEDSC extracts 8 shapelets from class 3, and from Figure 16 (1)(2) one can see that the shapelets from the same time series are diverse. The feature selection results show that, IEDSC can get more diverse shapelets, which may be the reason that the accuracy increases.

Figure 15 Shapelet of class 2



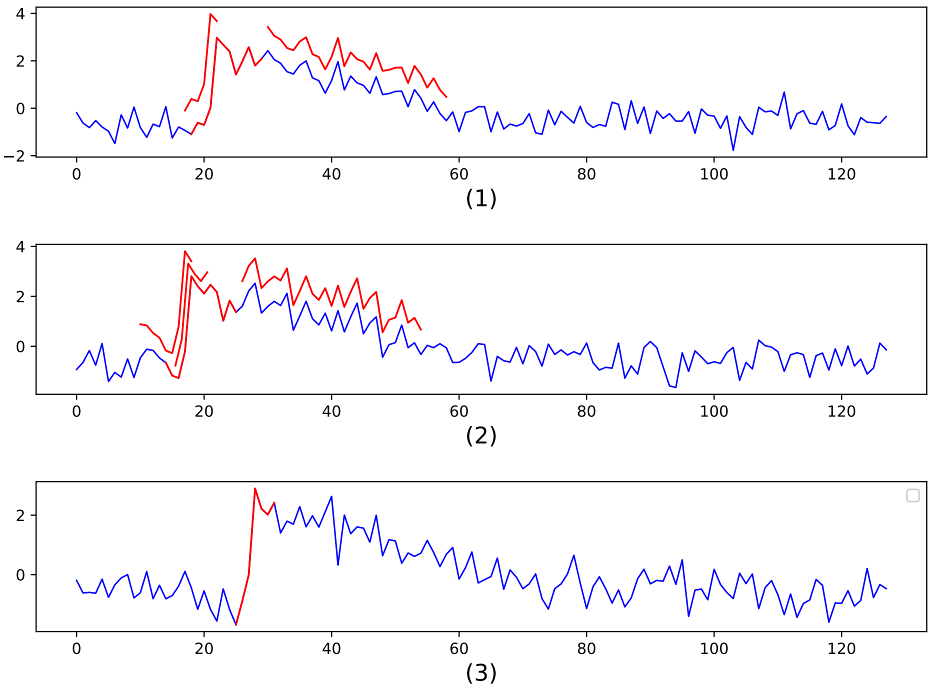


Figure 16 Shapelet of class 3

7 Conclusions

In this paper, we propose a new algorithm extracting diverse-shapelet for early classification on time series, called IEDSC. First, the IEDSC algorithm uses a new trend-based Euclidean distance to compute the similarity of time series, which can better measure the similarity of time series. Second, in order to reduce the shapelet candidates space, the IEDSC algorithm presents a shapelet pruning method based on the predictive starting positions to reduce the number of candidates. Third, in the shapelet selection phase, IEDSC makes the selected shapelets more diverse by a new feature selection method. Finally, we conduct the experiments on the real datasets. The experimental results demonstrate that IEDSC can make prediction earlier, obtain the comparable classification accuracy with that using full time series, and provide more interpretable results. In future work, we plan to improve the efficiency and effectiveness of similarity measure and feature selection, and also further study how to apply IEDSC to multivariate time series.

Acknowledgment The authors would like to thank Prof. Eamonn Keogh and all the people who have contributed to the UCR time series classification archive for their selfless work. We also thank the anonymous reviewers for their valuable advice.

The work is supported by the National Natural Science Foundation of China (No. 61702468), Open Research Project of The Hubei Key Laboratory of Intelligent Geo-Information Processing (no. KLIGIP-2018B03) and the Zhejiang Provincial Natural Science Foundation of China (No. LZ18F020001).

References

1. Ando, S., Suzuki, E.: Minimizing response time in time series classification. Knowl. Inf. Syst. **46**(2), 449–476 (2016)
2. Bentley, J.L., Sedgewick, R.: Fast algorithms for sorting and searching strings. In: 8th Acm-Siam symposium on discrete algorithms, pp. 360–369 (1997)
3. Chiu, B., Keogh, E., Lonardi, S.: Probabilistic discovery of time series motifs. In: Proc.acm Sigkdd int.conf.on knowledge discovery & data mining, pp. 493–498 (2003)
4. Dau, H.A., Keogh, E., Kamgar, K., Yeh, C.C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Yanping, H.B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The ucr time series classification archive. https://www.cs.ucr.edu/eamonn/time_series_data_2018/ (2018)
5. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**(Jan), 1–30 (2006)
6. Di Marzio, M., Taylor, C.C.: Kernel density classification and boosting: an l2 analysis. Stat. Comput. **15**(2), 113–123 (2005)
7. Fulcher, B.D.: Feature-based time-series analysis. arXiv:1709.08055 (2017)
8. Ghalwash, M.F., Obradovic, Z.: Early classification of multivariate temporal observations by extraction of interpretable shapelets. BMC Bioinforma. **13**(1), 195 (2012). https://doi.org/10.1186/1471-2105-13-195
9. Ghalwash, M.F., Radosavljevic, V., Obradovic, Z.: Utilizing temporal patterns for estimating uncertainty in interpretable early decision making. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, pp. 402–411 (2014)
10. Ghalwash, M.F., Ramljak, D., Obradovic, Z.: Early classification of multivariate time series using a hybrid hmm/svm model. In: Proceedings of the 2012 IEEE international conference on bioinformatics and biomedicine (BIBM), pp. 1–6 (2012)
11. Grabocka, J., Schilling, N., Wistuba, M., Schmidt-Thieme, L.: Learning time-series shapelets. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, pp. 392–401 (2014)
12. Hartvigsen, T., Sen, C., Kong, X., Rundensteiner, E.: Adaptive-halting policy network for early classification. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp. 101–110 (2019)
13. He, G., Duan, Y., Peng, R., Jing, X., Qian, T., Wang, L.: Early classification on multivariate time series. Neurocomputing **149**, 777–787 (2015)
14. He, G., Zhao, W., Xia, X., Peng, R., Wu, X.: An ensemble of shapelet-based classifiers on inter-class and intra-class imbalanced multivariate time series at the early stage. Soft Computing (2018)
15. Jiang, L., Li, C., Cai, Z.: Learning decision tree for ranking. Knowl. Inf. Syst. **20**(1), 123–135 (2009)
16. Karlsson, I., Papapetrou, P., Boström, H.: Early random shapelet forest. In: Calders, T., Ceci, M., Malerba, D. (eds.) Discovery science, pp. 261-276. Springer International Publishing, Cham (2016)
17. Keller, J.M., Gray, M.R., Givens, J.A.: A fuzzy k-nearest neighbor algorithm. IEEE Trans. Syst. Man Cybern. **4**, 580–585 (1985)
18. Keogh, E., Jessica, L., Ada, F.: Hot sax: Finding the most unusual time series subsequence: Algorithms and applications. In: International conference on data mining, pp. 1–27 (2008)
19. Li, G., Bräysy, O., Jiang, L., Wu, Z., Wang, Y.: Finding time series discord based on bit representation clustering. Knowl.-Based Syst. **54**, 243–254 (2013)
20. Li, G., Yan, W., Wu, Z.: Discovering shapelets with key points in time series classification. Expert Syst. Appl. **132**, 76–86 (2019)
21. Lin, T.H., Kaminski, N., Bar-Joseph, Z.: Alignment and classification of time series gene expression in clinical studies. Bioinformatics **24**(13), 147–155 (2008)
22. Lines, J., Davis, L.M., Hills, J., Bagnall, A.: A shapelet transform for time series classification. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '12, pp. 289–297. ACM (2012)
23. Ma, C., Weng, X., Shan, Z.: Early classification of multivariate time series based on piecewise aggregate approximation. In: Health information science, pp. 81–88 (2017)
24. Mori, U., Mendiburu, A., Dasgupta, S., Lozano, J.A.: Early classification of time series by simultaneously optimizing the accuracy and earliness. IEEE Transactions on Neural Networks and Learning Systems (2017)
25. Mori, U., Mendiburu, A., Keogh, E., Lozano, J.A.: Reliable early classification of time series based on discriminating the classes over time. Data Min. Knowl. Disc. **31**(1), 233–263 (2017)

26. Parrish, N., Anderson, H.S., Gupta, M.R., Hsiao, D.Y.: Classifying with confidence from incomplete information. J. Mach. Learn. Res. **14**(1), 3561–3589 (2013)
27. Romain, T., Simon, M.: Cost-aware early classification of time series. In: Machine learning and knowledge discovery in databases, pp. 632–647 (2016)
28. Sangnier, M., Gauthier, J., Rakotomamonjy, A.: Early and reliable event detection using proximity space representation. In: Proceedings of the 33rd international conference on international conference on machine learning - vol. 48, ICML'16, pp. 2310–2319 (2016)
29. Schäfer, P., Leser, U.: Teaser: Early and accurate time series classification. arXiv:1908.03405 (2019)
30. Song, W., Wang, L., Xiang, Y., Zomaya, A.Y.: Geographic spatiotemporal big data correlation analysis via the hilbert-huang transformation. J. Comput. Syst. Sci. **89**, 130–141 (2017)
31. Wang, S., Cao, J., Yu, P.S.: Deep learning for spatio-temporal data mining: A survey. arXiv:1906.04928 (2019)
32. Wang, W., Chen, C., Wang, W., Rai, P., Carin, L.: Earliness-aware deep convolutional networks for early time series classification. arXiv:1611.04578 (2016)
33. Wu, J., Pan, S., Zhu, X., Cai, Z.: Boosting for multi-graph classification. IEEE Trans. Cybern. **45**(3), 430–443 (2015)
34. Xing, Z., Pei, J., Yu, P.S.: Early prediction on time series: A nearest neighbor approach. In: International jont conference on artifical intelligence, pp. 1297–1302 (2009)
35. Xing, Z., Pei, J., Yu, P.S.: Early classification on time series. Knowl. Inf. Syst. **31**(1), 105–127 (2012)
36. Xing, Z., Pei, J., Yu, P.S., Wang, K.: Extracting interpretable features for early classification on time series. In: 11th Siam international conference on data mining, SDM 2011, April 28-30, 2011, Mesa, Arizona, USA, pp. 247–258 (2011)
37. Ye, L., Keogh, E.: Time series shapelets:a new primitive for data mining. In: ACM SIGKDD international conference on knowledge discovery and data mining, Paris, France, June 28 - July, pp. 947–956 (2009)
38. Yeh, C.C.M., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, H.A., Zimmerman, Z., Silva, D.F., Mueen, A., Keogh, E.: Time series joins, motifs, discords and shapelets: A unifying view that exploits the matrix profile. Data Mining & Knowledge Discovery **32**(1), 83–123 (2018)
39. Zalewski, W., Silva, F., Maletzke, A.G., Ferrero, C.A.: Exploring shapelet transformation for time series classification in decision trees. Knowl.-Based Syst. **112**, 80–91 (2016)