doi: 10.20532/cit.2019.1004635

# A Shapelet Transform Classification over Uncertain Time Series

Ruizhe Ma[1], Liangli Zuo[2] and Li Yan[2]

[1]Georgia State University, Atlanta, USA
[2]Nanjing University of Aeronautics and Astronautics, Nanjing, China

A shapelet is a time-series subsequence that can represent local, phase-independent similarity in shape. Time series classification with subsequences can save computing cost, improve computing speed and improve algorithm accuracy. The shapelet-based approaches for time series classification have an advantage of interpretability. Concentrating on uncertain time series, this paper tries to apply the shapelet-based method to classify uncertain time series. Due to the high dimensions of time series, the number of the generated candidate shapelets is generally huge. As a result, the calculation amount is large too. To deal with this problem, in this paper, we introduce a piecewise linear representation (PLR) method for uncertain time series based on key points so that the traditional shapelet discovery algorithm can be improved efficiently. We verify our approach with experiments. The experimental results show that the proposed shapelet algorithm can be used for uncertain time series and it can provide classification accuracy well while reducing time cost.

*ACM CCS (2012) Classification:* Mathematics of computing → Probability and statistics → Statistical paradigms → Time series analysis

Information systems → Data management systems → Database design and models → Data model extensions → Uncertainty

Computing methodologies → Machine learning → Learning paradigms → Supervised learning → Supervised learning by classification.

*Keywords*: uncertain time series, classification, shapelet, piecewise linear representation

## 1. Introduction

A time series is a sequence of ordered and equally spaced fixed values [1]. Time series data extensively exist in many real-world applications. It is necessary to provide accurate data analysis results and ensure the efficiency of the data representation and calculation process. Actually, one focus of time series studies is to process and analyze time series, which can provide foundational support for practical applications of time series data. Currently, time series analysis is receiving increasing attention, both from academy and industry. Common time series analysis mainly includes *forecasting and analyzing data trend* [2], *clustering data* [3], *classification data* [4], *outlier detection* [5] and so on.

Classification is an important research topic in the field of data mining. Time series data classification is also an important task of time series analysis. In recent years, time series classification has been widely used in devise applications. In network security monitoring, for example, the program cannot prevent attacks from malicious files that have never been encountered [7]. In this context, we can create an "entropy" time series to represent each file's content and then we apply a time series classification method to identify the malware. For example again, in meteorology domain, weather conditions are predicted according to satellite cloud image classification [8], in which households use electricity according to equipment to distinguish different household appliances [8].

In the context of traditional data classification, the classification criterion, which is determined by specific attribute value, plays a crucial role. Being a kind of sequence type data, however, time series data do not have obvious attribute characteristics because they appear in order according to the time point [6]. Here the order of data directly reflects the relationship between time series data. To classify time series data, many approaches have been proposed, including classifications based on the *nearest neighbor* and *shapelets*. The nearest neighbor (1-NN) classification is based on full-time domain features. Note that 1-NN classifier can lead to dimensional disaster problems because time series are typically high- dimensional data and data volume is very large. In addition, as a lazy classifier, 1-NN does not provide insights into time series data.

The shapelet-based classification uses local time series instead of full-time domain features in 1-NN classification to classify time series. This can effectively avoid phase offset and noise influence and can improve classification accuracy [10]. Moreover, this kind of classifier allows interpretation of the classification while maintaining its accuracy [11]. Due to its high efficiency and interpretability, it has become a common method for processing time series data in the field of data mining. Here we summarize the advantages of the shapelet-based classifiers as follows:

1. Shapelets can provide interpretable classification results for time series. Previous classifiers for time series do not have such interpretability.
2. Shapelet-based classification considers only local features of time series and this can avoid noise, phase shift errors *etc.* As a result, shapelet-based classification is more accurate and robust.

In addition, suppose that we classify a time series with length $m$ and each piece has length $l$. Let $k$ be the number of training sets. Then the time complexity of shapelet-based classification is $O(ml)$. But, for the classification method based on the nearest neighbor, its time complexity is $O(km^3)$ [12]. Note that time series are usually high-dimensional and the time complexity of calculating time series distance is very high. To this end, a piecewise linear representation (PLR) of time series is proposed for high-dimensional time series data. The PLR reflects the perception of sequence data understood by the human visual system well. It can implement data reduction and improve the operation speed. Nowadays, the PLR has been applied for time series analysis.

Note that, in time series, it is usually assumed that all values on time stamps are accurate and clear, and time series is a real-numbered sequence of fixed-point fixed values. However, this assumption is not always true. In many practical situations, time series data are generally collected by manual recording or physical devices (*e.g.*, wireless sensors). At this point, it is possible that the values of time series data contain uncertainty [6]. Actually, data uncertainty is common and inherent in most real applications. So, some efforts have been devoted to investigating uncertain time series with a special focus on similarity measures of uncertain time series. We argue that some classification approaches for general time series have been proposed, but the classification of uncertain time series is still scarce. To the best of our knowledge, this paper is the first effort to investigate the classification of uncertain time series.

In this paper, we concentrate on the classification of uncertain time series. Based on the idea of dimension reduction of time series data, we propose a shapelet filter pruning algorithm to remove similar shapelets in the candidate subsequences of shapelets. We reduce the number of shapelets and achieve a faster classification of uncertain time series. The main contributions of this paper are summarized as follows:

1. We propose a PLR for uncertain time series based on key points. We generate a low-dimensional time series, meanwhile ensuring that the data features do not disappear. This can help to improve the efficiency of the subsequent classification algorithm.
2. We propose a filtering pruning algorithm to remove similar shapelets in the candidate subsequence of shapelets and reduce the number of shapelet candidate sets.
3. We analyze the efficiency of the proposed algorithm and compare it with other shapelet-based classification algorithms.

The rest of this paper is organized as follows. We present related work in Section 2. Section

3 introduces the notations and definitions concerning uncertain time series and time series classification. In Section 4, we present the PLR for uncertain time series based on the key points and investigate binary tree construction. In Section 5, we propose the shapelet-based selection algorithm and the shapelet pruning algorithm. Section 6 shows the analysis and experimental results of our approach. Section 7 summarizes the work of this paper.

## 2. Related Work

There are some classification approaches for time series. In addition to 1-NN and decision tree classifiers, one can use other classification methods. Among them, shapelet-based classification for time series has attracted more attention due to its high efficiency and simplicity as well as its flexibility to take advantage of multiple classifiers.

A shapelet is a subsequence of time series that can exhibit the features of the sequence class well [10]. The first shapelet discovery classification algorithm for time series was proposed in [13], where all subsequences that may become shapelets are extracted and then the shapelet with the largest information is recursively selected by means of information gain evaluation. Here a decision tree is established and the shapelets are applied as the splitting nodes of the established decision tree. This method can provide not only the classification results but also an interpretable classification process. Mueen, Keogh and Young in [10] thought that the decision tree formed by the original shapelet discovery algorithm cannot provide a complete decision tree for machine learning. To deal with the problem that only a single shapelet may not be able to distinguish different categories of time series, a more explanatory logic-based shapelet method was proposed to construct a decision tree in [10]. For the high time-consuming methods, various acceleration techniques have been employed to accelerate the construction of the classifier. Rakthanmanon *et al.* in [14] proposed a fast shapelet discovery algorithm based on Symbol Aggregate Approximation (SAX). Their approach can improve the efficiency of shapelet discovery while the classification accuracy is maintained.

It should be noted that, in the above methods, the shapelet classifiers are embedded in a decision tree. It means that the shapelet discovery process is performed recursively. As a result, the shapelet discovery algorithm must cost a lot of time. In addition, the feature extraction of time series and the classifier construction are tightly coupled together. This makes it difficult to adapt shapelets and further construct other classifiers, say SVM, Bayesian networks, and so on.

In [11], Bagnall *et al.* thought that it is necessary to separate the shapelet extraction from classification process. In their approach, the shapelet discovery process is used as a separate preprocessing step. After that, a new dataset is constructed by using shapelet transformation. Finally, based on the constructed new dataset, the classification of time series can be performed by using multiple classifiers. It is shown that their approach reduces the coupling of shapelet discovery and classifier construction. In [15] and [16], Lines *et al.* investigated time series-oriented shapelet transformation classification method. They realized the decoupling of the shapelet discovery process and classifier construction. By constructing new classification datasets before building the classifier, their method improves classification accuracy and maintains the interpretability of shapelet-based classification of time series. Converting a time series classification problem to an alternate data space prior to classification can provide a higher level of improvement than developing a classifier.

## 3. Background and Notations

### 3.1. Definitions and Notations

**Definition 1 (*Time series subsequence*).** A continuous sequence of $U$ that starts from time position $i$ and ends at time position $j$ is called a time series subsequence (time series for short), denoted as $S$ [17], where $U$ represents a universe of discourse.

$$S = [t_i, t_{i+1}, ..., t_j]$$

Here, $S$ represents a subsequence of length $j-i+1$ that is selected from $U$. A subsequence represents the subsequence that is selected by any sliding window.

In the classical time series, it is always assumed that the value at a given time position is precise and certain. Data from real-world applications, however, may be imperfect and it is rarely the case that such an assumption can fully be satisfied. To represent and process uncertain data, probability theory has been applied to enhance various data models. With the probability theory, an uncertain data can be modeled by a probability distribution, which is generally represented by a probability density function (pdf). To simplify the representation and processing of uncertain data, a pdf is also simply characterized by the mean and variance in some application scenarios. Such a representation of uncertain data has been applied in uncertain time series (*e.g.*, [1]). Following this step, in this paper, we adopt the pdf with a form of mean and variance as the underlying distributions of uncertain values in uncertain time series. Here the variances may differ for different time points. Note that there may be some complex distributions which need additional parameters. At this point, the complex distributions can be transformed into the underlying distributions.

**Definition 2 (*Uncertain time series*).** Uncertain time series (UTS) consists of observations at fixed, equally spaced time points, in which the value at each time point is uncertain [1].

$$T = [v_1, v_2, \ldots, v_n]$$

Here we have $v_i = (t_i, u_i, p_j)$, in which $t_i$ denotes the *i*th sampling time point, $u_i$ denotes the observation value at $t_i$, and $p_i$ represents the probability density at $t_i$ with the form of $f(\mu_j, \sigma_j)$ [1]. Then we have $u_j = r_j + e_j$, where $r_j$ is the real value and $e_j$ is the error value.

Figure 1 shows a continuous uncertain time series. The value of each time point is a random variable subject to a certain probability distribution with a mean of $\mu_j$ and variance of $\sigma_j$. The error function may be an arbitrary probability distribution.

**Definition 3 (*Uncertain time series distance*).** Given two uncertain time series $T_1$ and $T_2$, the distance between $T_1$ and $T_2$ is represented by $dist(T_1, T_2)$, which returns a non-negative value, indicating the distance between $T_1$ and $T_2$ [18].

$$dist(T_1, T_2) = (E(T_1) - E(T_2))^2 + Var(T_1) + Var(T_2)$$

It can be intuitively observed from the above formula that the expected distance can reflect data uncertainty well. First, $(E(T_1) - E(T_2))^2$ can become smaller along with decreasing the distance difference. Second, $Var(T_1) + Var(T_2)$ indicates that the distance between two time points can become larger along with increas-

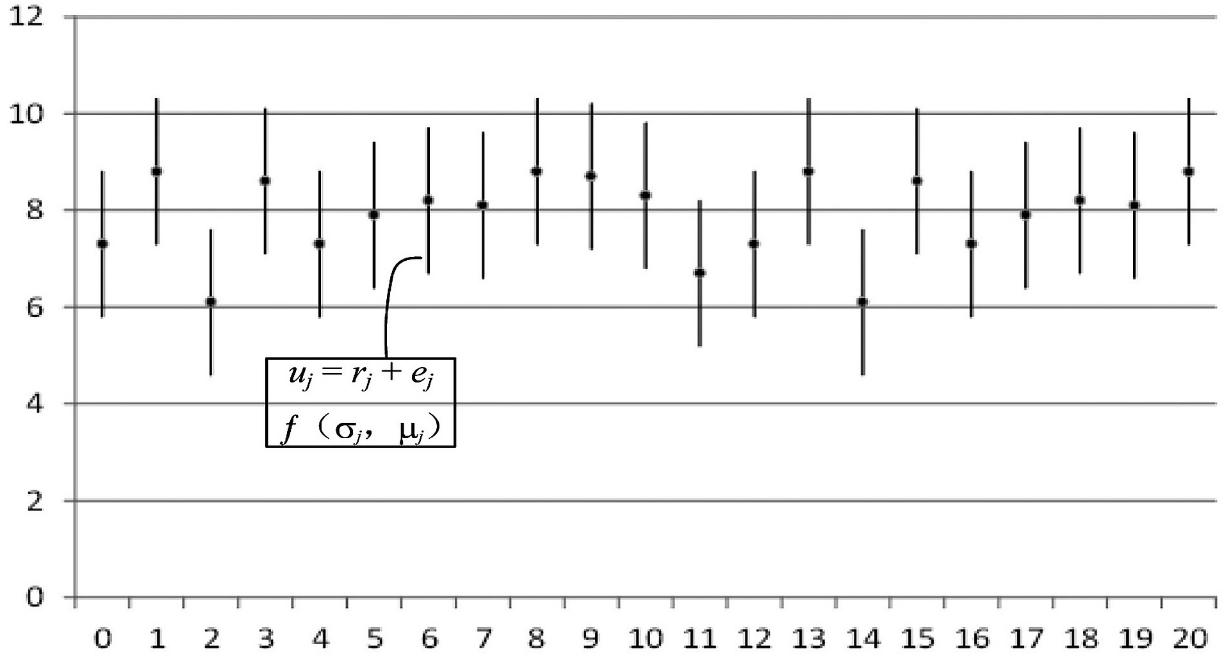

*Figure 1.* A continuous uncertain time series.

ing the errors in the uncertain time series. It is shown that the expected distance takes the mean and the variance of uncertain time series into account, which are two important parameters of the uncertain time series. So, it is feasible to describe uncertain time series data with the expectation and variance of uncertain data.

This paper focuses on uncertain time series classification with shapelets. For this purpose, we need to evaluate subsequence quality according to the distance from subsequence to uncertain time series. In Definition 4, we present a distance definition of subsequence to uncertain time series.

**Definition 4 (*Distance of subsequence to uncertain time series*).** Let $T$ be an uncertain time series and $S$ be a subsequence with length $l$ in $T$. Also, let $T_l$ represent all subsequences with length $l$ in $T$. Then, the distance between $S$ and $T$ is defined as the minimum of the distances of $S$ to $T_l$ and we have

$$subDist(S, T) = \min\{dist(S, T_1)\}.$$

**Definition 5 (*Information gain*).** A shapelet is determined by a subsequence $S$ in $DB$, where $DB$ is a set of subsequences. Then the shapelet divides $DB$ into two parts: $D_L$ and $D_R$. The information gain is calculated as follows.

$$InfoGain(S,\varepsilon) =$$

$$= E(DB) - \frac{|D_L|}{|DB|}E(D_L) - \frac{|D_R|}{|DB|}E(D_R)$$

Here

$$D_L = \{subdist(S, T_i) \le \varepsilon \mid T_i \in DB\} \text{ and}$$

$$D_R = \{subdist(S, T_i) > \varepsilon \mid T_i \in DB\}$$

We use $E()$ to represent the information entropy of the dataset. The definition of information gain is used to represent the quality of the time series of candidate shapelet segmentation [13]. The greater the information gain, the better the distinguishability of this shapelet. So, information gain can help to distinguish different sequence types.

### 3.2. Shapelet Transformation

In this paper, we classify time series based on shapelet transformation. In this method, the most important thing is to find the best set of shapelets, through which the raw data is mapped to new data space.

The classification algorithm based on shapelet transformation mainly consists of three steps. We first need to find all the time series subsequences as candidates, and then evaluate all candidates and choose the best shapelet set based on the chosen assessment. Here, each shapelet represents an attribute and its attribute value is the distance between it and the time series. We finally create a new data set, which uses shapelets as feature points, and then complete the classification by using general classification methods.

We briefly describe shapelet transformation processing in Algorithm 1. The input of this algorithm includes the pruned shapelet set and the time series set to be transformed. The output of this algorithm is the transformed data set. The algorithm calculates the distance of a time series to all shapelets in the shapelet collection and forms a series of new data in order (Steps 4–7). The data is stored in the transformed data set (Step 8). The above steps are iterated until all the data in the time series set is traversed (Steps 2–3).

*Algorithm 1.* Shapelet transformation.

```
Input: PrunedShapelet, time series set DB
Output: TD
1.  $TD \leftarrow \varnothing$;
2.  for $T_i$ in DB do  //each time series in DB
3.      transformed $\leftarrow \varnothing$;
4.      for $j = 0$ to |PrunedShapelet| do
5.          $S = PrunedShapelet.get(j)$
6.          $dist = subDist(S, T_i)$
7.          transformed.add(dist);
8.          TD.add(transformed);
9.      end for
10. end for
11. return TD
```

## 4. PLR for Uncertain Time Series

We can identify several dimensionality reduction representation methods of time series in the context of certain time series [17]: *symbol aggregate approximation* (SAX), *piecewise ag-*

*gregate approximate* (PAA), *piecewise linear representation* (PLR), *domain transformation representing*, *model representing*, and so on. Among them, the PLR method is simple and more in line with the visual reflection of the human visual system on the sequence data. In this paper, we adopt the PLR method to perform data dimensionality reduction of uncertain time series.

According to the partitioning strategy in the PLR, we identify two kinds of PLR. The first one is to segment the fitting error. This kind of PLR uses the straight line that connects the endpoint of the segment and the starting point to fit the original time series. Then the least squares between the fitting curve and the original time series can be guaranteed, and the error is minimal. Note that this method focuses only on the local minimum error of the sequence and does not take characteristic changes in the whole sequence into account. The second kind of PLR is the segment determined by the special points that can classify time series, say *local extremum points* and *boundary values points*. This method can avoid the global feature missing occurred in the first kind of PLR. The shortcoming of this method is that we need different methods to determine the feature points and these methods may produce large segmentation fitting errors.

By default, for the uncertain time series the elements on all timestamps obey a certain probability distribution function, where a mean value is used to calculate the fitting error of each time point. Then, the linear segmentation interpolation process can be performed. We apply a binary tree to store the key point error function. To achieve a more efficient selection process, we need a simple dimension reduction method of uncertain time series based on key points. In this way, we can improve the extraction efficiency process while we maintain computational accuracy to achieve stable classification accuracy. The existing dimensionality reduction work on time series is mainly for certain time series.

## 4.1. PLR of Uncertain Time Series Based on Key Points

For an uncertain time series, we propose a simple linear segmentation interpolation method based on key points to achieve data dimensionality reduction. In uncertain time series, the true values of all elements on timestamps are infinitely close to the mean of the probability distribution function, where the mean value is used to calculate the fitting error of each time point for linear segmentation processing.

Our method firstly finds the key points in uncertain time series according to the weight of fragments, then splits uncertain time series into several time segments with the key points, and finally completes the piecewise linear representation of the sequence. Here, all key points are put into a binary tree. Considering the uncertainty in time series and avoiding excessive noise reduction, we build a binary tree according to the index of the selected point.

In the built tree, each node stores the key point and the error function that corresponds to the key point. By obtaining key points in the binary tree, the number of segments directly achieves a fast PLR. Compared with the ordinary time series piecewise linear representation method, our method comprehensively considers the global error, segmentation error, and single index point error, and to the great extent retains the global and local features of time series in the dimension reduction process. In the following, we present the definition of key points in the uncertain time series.

**Definition 6 (*Key point in uncertain time series*).** In the PLR process of uncertain time series, the key points must satisfy the following two conditions:

- they should be located in the uncertainty time series fragment with the largest weight, and
- the fitting error in the uncertain time series segment should be the largest.

In Definition 6, the weight of the indeterminate time series segment is related to all single point fitting errors in the segment.

Let $U = [u_1, u_2, ..., u_n]$ be a given uncertain time series, where $u_i$ represents the random variable at time point $i$ and obeys the probability distribution $f(\mu_i, \sigma_i)$. Then we can get the vertical distance from the point to the fitted curve (*i.e.* the fitting error). The calculation formula is shown as follows.

$$d_i = \left| \mu_1 + \frac{(\mu_n - \mu_1) * (t_i - t_1)}{t_n - t_1} - \mu_i \right|$$

The weight of an indeterminate time series segment is represented as follows.

$$weight = \max\{d_{sum}, 2 \times d_{max}\}$$

Here, $d_{sum}$ represents the sum of the fitting errors of the points and $d_{max}$ represents the single point maximum fitting error. The maximum fitting error multiplied by 2 is applied in order to stress the overall error caused by a single erroneous point.

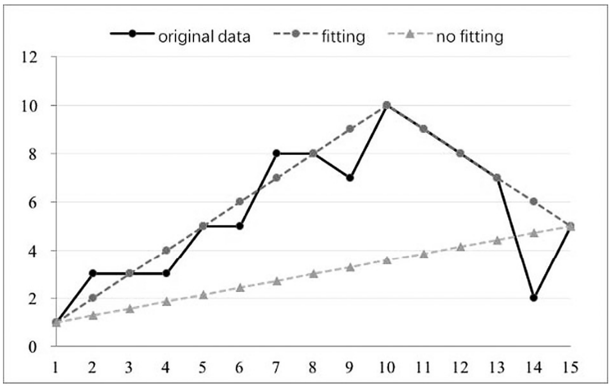

*Figure 2.* Example of a single point error.

Figure 2 shows the case when the overall error is small, but the single point error is large.

By default, the true values of elements on all timestamps are infinitely close to the mean of the probability distribution function, where the mean value is used to represent the value at the time point. Then, the fitting error of each point is calculated and the linear segmentation processing is performed.

In Figure 2, the solid black line indicates the original sequence before segmentation. The line segmentation error threshold is set to 6, and the segmentation result is represented with the grey dotted line in Figure 2. A point with an index of 10 is considered to be a split point. Then the fit line is divided into two segments with a fitting error of 6 on the left and a fitting error of 4 on the right, respectively. Among them, the fitting error on the left is caused by 5 points, the single point error is 1 or 2, and the fitting effect is close to the original data. The fitting error on the right is only caused by a single point. The fitting error of index point 14 in Figure 2 is 4 and the fitting effect at this time is not ideal. A simple method that is based on segmentation of the overall error segmentation may result in an unsatisfactory segmentation result because such a method neglects the case when there is a large single point error in the segmentation.

In response to this deficiency, we apply a key point-based segmentation method. Compared to the general segmentation points, the key points contain more features of the sequence data and they can result in a good segmentation fit line. By adding a constant factor to the weight function, the observation point with a large fitting error is taken into account within the function and this can avoid the situation shown in Figure 3. The purpose why we use segmentation to represent uncertain time series is to reduce the dimensionality of uncertain time series. For this purpose, we need to maintain the original features of data as much as possible in the dimension reduction process. As shown in the weight function definition, when the weight of the segment is large, the fitting effect is poor at this time point. The reason may be that the overall error of the segment is large or the single point error is large. At this time point, the time series segment with the highest weight should be continuously segmented. When the piecewise fitting error is less than the threshold and the single point error distance is greater than 1/2 threshold, the segmentation for the time point continues.

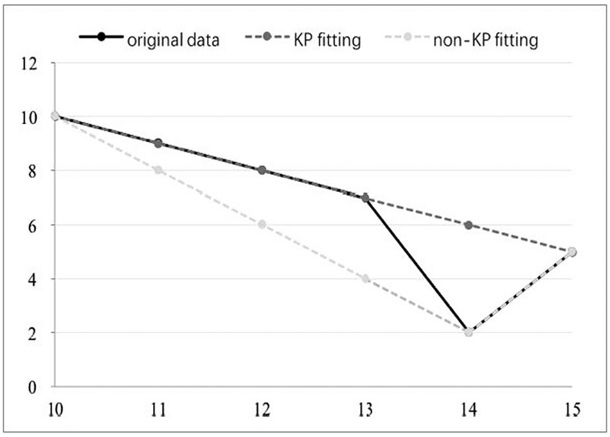

*Figure 3.* Time series fragment segmentation point selection.

Figure 3 shows a detailed fragmentation diagram. As shown in Figure 3, the black line shows the raw data (the first half of line is covered by the grey line and the last half of the line is covered by the light-grey line). The light-grey dashed line represents a non-KP piecewise fitting line, and the fitting distance of the segment is smaller than the threshold 6. Note that the single point fitting error of the index point 14 is 4, which is greater than 3. At this moment, the segmentation is performed. The final fitted line is indicated by the grey dashed line in Figure 3.

In the linear segmentation process of uncertain time series, the initial situation is that the entire sequence is regarded as the first segment by default. Depending on the segment weight and on whether a segmentation will continue or not, each segment for the time series segmentation should preferably retain the overall or local shape characteristics of the sequence. In order to segment the time series segmentation, we need to compare the fitting distance of each time point in the segment, mark the time point with the largest single point fitting distance as the key point, and then use the key point as the dividing point to preserve the global feature of uncertain time series to the greatest extent. At the same time, for the case that the segmentation fitting error is small and the deviation of a certain time point in the segment is large, the segmentation is performed again, until the local features of uncertain time series are well reflected.

### 4.2. Binary Tree Construction

In order to quickly perform linear segmentation of uncertain time series, the mean value of the random variable at a time point is used to represent uncertain data value. For uncertain value on the timestamp in an uncertain time series, we apply a binary tree to store the error function, which corresponds to the selected key point. The choice of key points ensures that the characteristics of data fluctuations are reflected. This can avoid the missing and excessive noise removal of data feature points. As a result, we improve the extraction efficiency in the classification algorithm of uncertain time series and, meanwhile, we maintain the calculation accuracy to obtain stable classification accuracy.

For a node that is inserted into the binary tree, we have to consider the index position, the observation value, the probability density function, the selection order, the left and right weights of the corresponding key points, and the left and right child nodes. Let *TreeNode* be an insertion node and its formal definition is given as follows.

$$\text{TreeNode} = \{index, value, rank, f(\mu, \sigma), \\ weight_l, weight_r, child_l, child_r\}$$

In the segmentation process, the key points with the largest fitting error are used for segmentation, and two new slice objects segment ($slice.p_b$, $slice.p_{max}$) and segment ($slice.p_{max}$, $slice.p_e$) are generated after segmentation. The key points and related information are stored in the binary tree. The segmenting process is iteratively handled until there is not any segment that needs to be segmented.

By constructing a binary tree, the information of uncertain data in time series is not lost and the required key points can be obtained from the tree according to the given number of time series fragments or fragment weights. Constructing a binary tree cannot only help the subsequent classification on uncertain time series but also improve the extraction efficiency by maintaining the calculation accuracy and obtaining stable classification accuracy. The classification algorithm has an advantage of being explanatory. Note that high dimensionality of time series can lead to a high time cost in the distance calculation, which makes the efficiency of the entire classification process lower. Therefore, a dimensionality reduction in data can effectively alleviate this problem.

For the uncertain time series after PLR, we can use the shapelet selection algorithm and pruning strategy to obtain a small number of sets of optimal feature subsequences of retained time series. We further use the shapelet transformation algorithm to get a new data set. This new data set is small and meanwhile it retains the ability to represent data characteristics. For new data sets, we can flexibly use different classifiers to achieve uncertain time series classification.

## 5. Shapelet-Based Classification for Uncertain Time Series

We apply the shapelet transformation method to perform the classification of uncertain time series. First, all subsequences of uncertain time series constitute a set of candidate shapelets.

Second, all candidate subsequences are evaluated according to the information gain assessment in order to select the best shapelets. Let each shapelet represent an attribute and then its value is the distance between the attribute to the time series. This creates a new data set. Finally, for the new data set, the general classification methods (*e.g.*, Naive Bayes, decision trees and SVMs) can be used for classification.

### 5.1. Shapelet-Based Selection Algorithm

In [16], a caching algorithm was introduced for storing $k$ best shapelets from a dataset in a single traversal. This algorithm traverses all candidate shapelets at a time, calculates the maximum information gain of all sub-sequences, stores them in the shapelet set, sorts them by information entropy size, removes self-similar sequences, and finally collates and extracts the first $k$ shapelets. Obviously, the number of shapelets generated with this algorithm is very large and there must be redundancy. The situation becomes worse in the context of an uncertain time series. So, it is necessary to optimize this algorithm and eliminate redundant shapelets so that it can participate in the shapelet conversion classification well.

*Algorithm 2.* All shapelet selection.

```
Input: DB, min, max
Output: CandShapelet
CandShapelet = ∅;
for $T_i$ in DB do
  shapelets = ∅
    for l = min: max do
      cand = generateCandidates($T_i$, l)
      for all subsequence S in cand do
        ds = subDist(S, DB)
        quality = assessCandidate(S, ds) //max
        information gain
      shapelets.add(S, quality);
      end for
    end for
  sortByQuality(shapelets) //sort by information
  gain
  removeSelfSimilar(shapelets) //remove
  selt-similar subsequences
  CandShapelet.add(shapelets)
end for
return CandShapelet
```

Following the step of the algorithm developed in [16], we propose Algorithm 2 for a shapelet selection. Algorithm 2 iterates through all subsequences, calculates their distance to all uncertain time series, evaluates the maximum information gain, adds them to the candidate shapelets, and then arranges them in descending order of their information gain. Finally, we can get all shapelet collections after removing self-similar shapelets in the same time series.

### 5.2. Shapelet Pruning Algorithm

This section describes a filtering pruning method to remove similar features. This allows feature subsequences with discriminative advantages to participate in data conversion. For this purpose, we iterate through all time series, perform pruning filtering on all the generated sub-sequences, remove similar sub-sequences, and then improve the quality of elements in the shapelet.

Given two shapelets $S_1$ and $S_2$ as well as their corresponding distance thresholds $\varepsilon_1$ and $\varepsilon_2$, let $S_1$ and $S_2$ be in the same time series class label and $subDist(S_1, S_2) < \varepsilon_1$. At this point, it can be determined that $S_1$ and $S_2$ are similar. Here $\varepsilon_1$ is a distance threshold that is capable of obtaining the maximum information gain, and $subDist(S_1, S_2) < \varepsilon_1$ means that $S_1$ and $S_2$ have similar shapes. In other words, $S_1$ can replace $S_2$ in most cases.

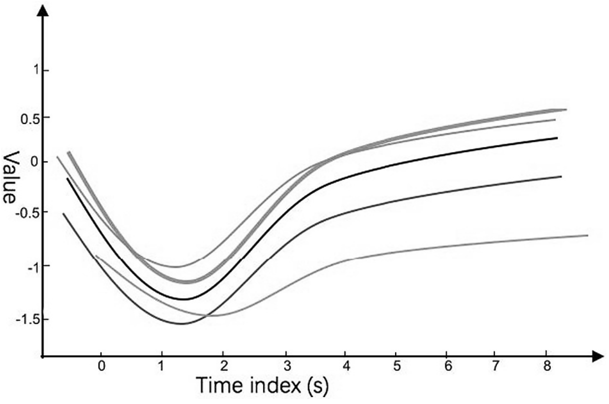

*Figure 4.* Five shapelets and the degree of matching when overlapping each other.

Figure 4 shows the top five best shapelets extracted by the method in [16]. It is shown in Figure 4 that these data have great similarity when they ate matched. Using a pruning strategy can effectively reduce similar shapelets while preserving the shapelets that represent time series features very well.

Similar shapelets in the same time series are regarded to be self-similarity, which is removed in Algorithm 2 (Step 13). Then, two shapelets to be compared generally exist in different time series. In this paper, we propose Algorithm 3 as our pruning algorithm.

*Algorithm 3.* Shapelet pruning.

```
Input: CandShapelet
Output: PrunedCandidate
1.  PrunedCandidate ← ∅;
2.  for i = 1:|CandShapelet| do
3.     shapelet = CandShapelet[i]
4.     PrunedCandidate.add(shapelet)
5.     ts = shapelet.splitThreshold
6.     for u = (i + 1) : |CandShapelet| do
7.        prun = CandShapelet[u]
8.        dist = Dist(prun, shapelet)
9.           if (dist < ts) //If the distance is less than ts
10.             if (prun.label! = shapelet)// the class is
                different, keep
11.                PrunedCandidate.add(prun)
12.          else   //If the distance is greater than the ts,
             keep it
13.             PrunedCandidate.add(prun)
14.    end for
15. end for
16. return PrunedCandidate
```

For each shapelet in the candidate shapelets, Algorithm 3 compares it with the shapelet that is better than it (Steps 2–6). When the distance between these two shapelets is less than the corresponding threshold, Algorithm 3 compares their class labels. When their class labels are the same, we do not need to make a place in the shapelet collection (Steps 6–11); otherwise, we put them in the PrunedShapelet (Steps 12–13). Finally, we get a collection of dissimilar shapelets for shapelet transformation classification.

Figure 5 shows the top ten best shapelets generated for the Gun_Point dataset in [16]. It is observed that there are significant similarity and difference between these data (left), which can be regarded as two different types of feature subsequences. These two clusters can interpret the time series in an explanatory manner. Using the shapelet pruning algorithm and the expected distance for distance calculation, we remove the similar shaplelets in the same time series. We leave only a single shapelet to form a new shapelet data set (right) for subsequent data conversion and achieve a shapelet-based classification for uncertain time series.

## 6. Experimental Results and Analysis

In this section, we illustrate with experiments the feasibility and advantages of our approach for uncertain time series classification. First, we compare the proposed shapelet transformation classification algorithm with the algorithms proposed in [14], [16] and [19] in order to illustrate the usability of shapelets in classifying uncertain time series. Second, with different classifiers to classify test dataset, we illustrate that the pruning algorithm and PLR method proposed in this paper can provide better accuracy.

We used Java language to implement the algorithm proposed in the paper. Our experiments run in a laptop with AMD processor, Win10 operating system, and 8GB system memory.

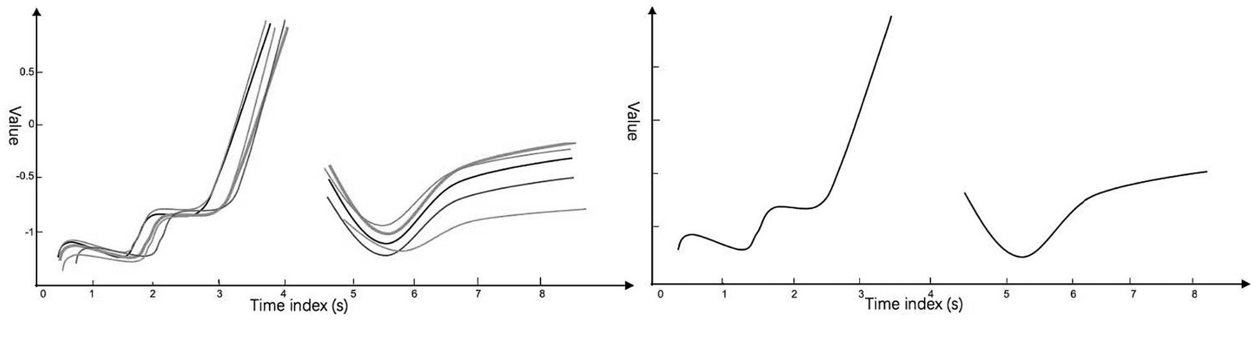

*Figure 5.* Top 10 shapelets of the Gun_Point dataset.

## 6.1. Experimental Datasets

In the experiments, 17 datasets in UCR time series data [20] were used as the test targets. All datasets consist of a training set and a testing set. Among them, the training set is used to perform shapelet selection discovery and classifier construction, and the test set is used to evaluate the classification accuracy of the classifier.

We artificially add interference to the UCR data to obtain uncertain data. Following the underlying distributions of uncertain time series that are discussed in Section 3.1 (that is, a form of mean and variance), we here apply values in (0, 1) as noises, which are randomly generated and may be positive or negative. The same processing is performed for all sequences. The function of arbitrary distribution with the expectation of 0 and the variance of $\sigma$ are used as the error function at each time point. Then the data is processed by the phased error function according to the distribution trend of the data itself, $0.1\sigma \sim 2\sigma$ is used for the data of different time periods as the variance of the error function. Here $\sigma$ represents the standard deviation of the sample itself.

## 6.2. Classification Effect Comparison

We compare our method with three shapelet-based classification methods, which are FS [14], ST [16] and LS [19], respectively. The FS represents the SAX-based fast shapelet algorithm proposed in [14], the ST represents the shapelet transformation algorithm proposed in [16], and the LS is the machine learning-based shapelet algorithm proposed in [19]. In this paper, we use decision tree classifier in the classification experiments and our method is referred to as the STU. Note that, in the experiments, the methods of FS, ST and LS were proposed for certain time series and our method of STU is for uncertain time series.

We want to demonstrate that the shapelet transformations of uncertain time series can maintain the similar accuracy of classification results with respect to the shapelet transformations of certain time series. For each original dataset in UCR (certain time series), say GunPoin, we respectively applied the methods of FS, ST and LS and obtained their accuracy of classification. Then, for the selected original dataset we created the corresponding uncertain dataset (uncertain time series), which was used by the STU, and obtained its classification accuracy. Table 1 presents the classification accuracy obtained with the above-mentioned four different methods.

*Table 1.* Classification accuracy with different shapelet optimization methods.

| **Dataset** | **ST** | **FS** | **LS** | **STU** |
|---|---|---|---|---|
| Syn_Con | 0.897 | 0.777 | 0.783 | 0.833 |
| GunPoin | 0.925 | 0.866 | 0.980 | 0.930 |
| CBF | 0.864 | 0.865 | 0.934 | 0.876 |
| Face (all) | 0.607 | 0.658 | 0.592 | 0.632 |
| OSULeaf | 0.583 | 0.593 | 0.516 | 0.600 |
| Swe_Leaf | 0.664 | 0.670 | 0.721 | 0.675 |
| 50Words | 0.769 | 0.633 | 0.657 | 0.777 |
| Trace | 1.000 | 1.000 | 1.000 | 1.000 |
| Two_Patt | 0.695 | 0.644 | 0.652 | 0.670 |
| Wafer | 0.960 | 0.948 | 0.978 | 0.935 |
| Light-2 | 0.777 | 0.783 | 0.815 | 0.773 |
| Light-7 | 0.726 | 0.644 | 0.795 | 0.719 |
| ECG | 0.997 | 0.924 | 0.996 | 0.989 |
| Adiac | 0.783 | 0.593 | 0.522 | 0.784 |
| Yoga | 0.670 | 0.664 | 0.721 | 0.629 |
| Beef | 0.900 | 0.567 | 0.867 | 0.916 |
| Coffee | 0.964 | 0.929 | 1.000 | 0.983 |
| ***Average*** | **0.811** | **0.750** | **0.796** | **0.807** |

In Table 1, for a given dataset with certain numbers of classes, we extract its feature subsequence fragments and then calculate the distances between all shapelets and the time series to be measured. The class of the shapelet with the shortest distance to the time series to be measured will be the class of the time series to be measured. Generally, we have a number of the extracted shapelets due to a dataset with numerous classes. The final accuracy measured is determined by the approaches for extracting the shapelets and calculating the distances.

As we know, the C4.5 and 1NN classifiers are commonly used in time series for classification. In addition to the methods of FS, ST and LS, we also compared the classification accuracy of the C4.5, 1NN and STU. Similarly, we used the original datasets (certain time series) in the C4.5 and 1NN and the corresponding uncertain datasets (uncertain time series) in the STU. Table 2 presents the classification accuracy obtained with the methods of C4.5, 1NN and STU. Note that the C4.5 and the ST are identical for time series classification and, as a result, they have the same classification accuracy on the same datasets.

*Table 2.* Classification accuracy with different classifiers.

| **Dataset** | **C4.5** | **1NN** | **STU** |
|---|---|---|---|
| Syn_Con | 0.897 | 0.983(6) | 0.833 |
| GunPoin | 0.925 | 0.913(0) | 0.930 |
| CBF | 0.864 | 0.96(11) | 0.876 |
| Face (all) | 0.607 | 0.808(3) | 0.632 |
| OSULeaf | 0.583 | 0.662(7) | 0.600 |
| Swe_Leaf | 0.664 | 0.846(2) | 0.675 |
| 50Words | 0.769 | 0.758(6) | 0.777 |
| Trace | 1.000 | 0.99(3) | 1.000 |
| Two_Patt | 0.695 | 0.998 (4) | 0.670 |
| Wafer | 0.960 | 0.995 (1) | 0.935 |
| Light-2 | 0.777 | 0.869 (6) | 0.773 |
| Light-7 | 0.726 | 0.712 (5) | 0.719 |
| ECG | 0.997 | 0.88(0) | 0.989 |
| Adiac | 0.783 | 0.609 (3) | 0.784 |
| Yoga | 0.670 | 0.845 (2) | 0.629 |
| Beef | 0.900 | 0.667 (0) | 0.916 |
| Coffee | 0.964 | 1.000 (0) | 0.983 |
| ***Average*** | **0.811** | **0.853** | **0.807** |

Note that the ST method in Table 1 and the C4.5 method in Table 2 are identical because they apply the same method in [16]. Therefore, the shapelet optimization method ST produces the same exact results in Table 1 as the classifier C4.5. in Table 2. It is shown in Table 1 and Table 2 that our method can provide the similar classification accuracy with other five methods. For the given 17 datasets, totally speaking, the STU method provides a better classification accuracy than the methods of ST, C4.5 and 1NN, and almost the same classification accuracy as the LS method. Note that the classification accuracy provided by the STU method is not better than the classification accuracy provided by the FS method. But the STU method can deal with the classification of uncertain time series and other five methods can classify only certain time series.

## 6.3. Algorithm Efficiency Analysis

As we know, the number of time series is usually very large and the training process has high time complexity. Many efforts have been devoted to reducing the complexity of the training process [13], [10], [15]. Suppose that the number of time series objects in the dataset is $k$ and the average length of each time series is $m$. In [13], Ye and Keogh used SDEA (Subsequence Distance Early Abandon) and AEP (Admissible Entropy Pruning) to reduce running time and their method has the time complexity of $O(m^4k^2)$. Mueen *et al.* in [10] tried to reduce this complexity by caching distance calculations for future use. They introduced a pruning strategy and achieved an order of magnitude faster than the method in [13]. In [15], Hill *et al.* used hierarchical clustering to obtain different shapelets. The worst-case time complexity of hierarchical clustering is $O(k^3)$ and after optimization, the complexity of hierarchical clustering is still $O(k^2 \log k)$. The 1-NN classifier focuses on classifying time series with optimal accuracy, which provides the best classification accuracy. However, the use of DTW-based 1-NN classifiers is less suitable for time series classification because it has the $O(k^4m^4)$ complexity of each instance in the training set.

The time complexity of the shapelet filtering pruning method proposed in this paper is $O(k^2m^2)$. It is shown that the shapelet classification algorithm used in this paper has a low algorithm complexity, and good interpretability. So, the method proposed in the paper is a classification method with better comprehensive performance. The main advantage of the 1-NN classification method is only in its classification accuracy. In some research fields that classification is completed according to feature types (say image recognition and gesture recognition), and it is very necessary to provide an interpretable process for classification. Classification metrics in these areas are not limited to classification accuracy. Classification algorithm based on shapelets has such an advantage in interpretability.

## 7. Conclusion

This paper presents a transformation for uncertain time series classification. We provide a new algorithm that just scans the datasets once and uses the pruning strategy to remove similar shapelets. Experimental results show that the proposed classifier can perform uncertain time series classification better on most datasets compared with the other classifiers. Compared with other algorithms, the algorithm proposed in the paper can provide similar accuracy and less time complexity. More importantly, the proposed algorithm can provide good interpretability in uncertain time series classification.

Although the dimensionality reduction process can reduce the partial complexity, our method still inevitably produces some errors. Based on the characteristics of uncertainty in time series data, one of the directions for our future work is to achieve better shapelet extraction.

## References

[1] L. L. Zuo and L. Yan, "A Weighted DTW Approach for Similarity Matching over Uncertain Time Series", *Journal of Computing and Information Technology*, vol. 26, no. 3, pp. 179–190, 2018.
https://dx.doi.org/10.20532/cit.2018.1004217

[2] R. Z. Ma *et al.*, "A Data-Driven Analysis of Interplanetary Coronal Mass Eject and Magnetic Flux Ropes", in *Proceedings of the 2016 IEEE International Conference on Big Data*, 2016, pp. 3177–3186.
http://dx.doi.org/10.1109/BigData.2016.7840973

[3] R. Z. Ma and R. A. Angryk, "Distance and Density Clustering for Time Series Data", in *Proceedings of the 2017 IEEE International Conference on Data Mining Workshops*, 2017, pp. 25–32.
http://dx.doi.org/10.1109/ICDMW.2017.11

[4] J. Lines *et al.*, "Time Series Classification with HIVE-COTE: The Hierarchical Vote Collective of Transformation-Based Ensembles", *ACM Transactions on Knowledge Discovery from Data*, vol. 12, no. 5, p. 52, 2018.
http://dx.doi.org/10.1145/3182382

[5] T. Kieu *et al.*, "Outlier Detection for Multidimensional Time Series Using Deep Neural Networks", in *Proceedings of the 19th IEEE International Conference on Mobile Data Management*, 2018, pp. 125–134.
http://dx.doi.org/10.1109/MDM.2018.00029

[6] J. Aßfalg *et al.*, "Probabilistic Similarity Search for Uncertain Time Series", in *Proceedings of the 2009 International Conference on Scientific and Statistical Database Management*, 2009, pp. 435–443.
https://doi.org/10.1007/978-3-642-02279-1_31

[7] O. Patri *et al.*, "Discovering Malware with Time Series Shapelets", in *Proceedings of the 50th Hawaii International Conference on System Sciences*, 2017, pp. 1–10.
http://dx.doi.org/10.24251/HICSS.2017.734

[8] A. McGovern *et al.*, "Identifying Predictive Multi-Dimensional Time Series Motifs: an Application to Severe Weather Prediction", *Data Mining and Knowledge Discovery*, vol. 22, pp. 232–258, 2011.
https://doi.org/10.1007/s10618-010-0193-7

[9] J. Lines *et al.*, "Classification of Household Devices by Electricity Usage Profiles", in *Proceedings of the 2011 International Conference on Intelligent Data Engineering and Automated Learning*, 2011, pp. 403–412.
https://doi.org/10.1007/978-3-642-23878-9_48

[10] A. Mueen *et al.*, "Logical-Shapelets: an Expressive Primitive for Time Series Classification", in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2011, pp. 1154–1162.
http://dx.doi.org/10.1145/2020408.2020587M

[11] A. Bagnall *et al.*, "The Great Time Series Classification Bake off: A Review and Experimental Evaluation of Recent Algorithmic Advances", *Data Mining and Knowledge Discovery*, vol. 31, no. 3, pp. 606–660, 2017.
http://dx.doi.org/10.1007/s10618-016-0483-9

[12] L. Ye and E. J. Keogh, "Time Series Shapelets: A Novel Technique that Allows Accurate, Interpretable and Fast Classification", *Data Mining and Knowledge Discovery*, vol. 22, pp. 149–182, 2011.
http://dx.doi.org/10.1007/s10618-010-0179-5

[13] L. Ye and E. J. Keogh, "Time Series Shapelets: A New Primitive for Data Mining", in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 947–956.
http://dx.doi.org/10.1145/1557019.1557122

[14] E. J. Keogh and T. Rakthanmanon, "Fast Shapelets: A Scalable Algorithm for Discovering Time Series Shapelets", in *Proceedings of the 2013 SIAM International Conference on Data Mining*, 2013, pp. 668–676.
http://dx.doi.org/10.1137/1.9781611972832.74

[15] J. Hills *et al.*, "Classification of Time Series by Shapelet Transformation", *Data Mining and Knowledge Discovery*, vol. 28, no. 4, pp. 851–881, 2014.
http://dx.doi.org/10.1007/s10618-013-0322-1

[16] J. Lines *et al.*, "A Shapelet Transform for Time Series Classification", in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 289–297.
http://dx.doi.org/10.1145/2339530.2339579

[17] C. Ji *et al.*, "A Piecewise Linear Representation Method Based on Importance Data Points for Time Series Data", in *Proceedings of the 20th IEEE International Conference on Computer Supported Cooperative Work in Design*, 2016, pp. 111–116.
http://dx.doi.org/10.1109/CSCWD.2016.7565973

[18] W. K. Ngai *et al.*, "Efficient Clustering of Uncertain Data", in *Proceedings of the 16th International Conference on Data Mining*, 2006, pp. 436–445.
http://dx.doi.org/10.1109/ICDM.2006.63

[19] J. Grabocka *et al.*, "Learning Time-Series Shapelets", in *Proceedings of the 2014 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 392–401.
http://dx.doi.org/10.1145/2623330.2623613

[20] Y. Chen *et al.*, "The UCR Time Series Classification Archive", 2015.
www.cs.ucr.edu/~eamonn/time_series_data/

*Received:* February 2019
*Revised:* November 2019
*Accepted:* November 2019

*Contact addresses*:
Ruizhe Ma
Georgia State University
Atlanta
USA
e-mail: cstnuaa@163.com

Liangli Zuo
Nanjing University of Aeronautics and Astronautics
Nanjing
China
e-mail: llzuo@163.com

Li Yan*
Nanjing University of Aeronautics and Astronautics
Nanjing
China
e-mail: yanli@nuaa.edu.cn.
*Corresponding author

Ruizhe Ma received her PhD degree from the Department of Computer Science at the Georgia State University, USA. Her research interests include time series analysis, Big Data processing and data mining.

Liangli Zuo received her Master degree from the College of Computer Science and Technology at the Nanjing University of Aeronautics and Astronautics, China. Her research interests include time series analysis and uncertain data management.

Li Yan is a full professor at the College of Computer Science and Technology at the Nanjing University of Aeronautics and Astronautics, China. Her current research interests include uncertain data and knowledge engineering.