Journal of Computers Vol. 30 No. 1, 2019, pp. 105-113
doi:10.3966/199115992019023001011

A Framework for Discovering Variable-length Motifs in Medical Data Streams

Le Sun[1*], Jinyuan He[2], Chen Wang[1], Jiangang Ma[2], Hai Dong[3], Yanchun Zhang[2]

[1] School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China
sunle2009@gmail.com; wangchennuist@126.com

[2] Centre for Applied Informatics, Victoria University, VIC, Australia
jinyuan.he@live.vu.edu.au; {jiangang.ma, yanchun.zhang}@vu.edu.au

[3] School of Science, RMIT University, VIC, Australia
hai.dong@rmit.edu.au

Received 9 January 2018; Revised 17 January 2018; Accepted 22 January 2018

Abstract. In this paper, we explore two key problems in time series motif discovery: releasing the constraints of trivial matching between subsequence with different lengths and improving the time and space efficiency. The purpose of avoiding trivial matching is to avoid too much repetition between subsequence in calculating their similarities. We describe a limited-length enhanced suffix array based framework (LiSAM) to resolve the two problems. We first convert the continuous time series to the discrete time series using the Symbolic Aggregate approXimation procedure, and then introduce two covering relations of the discrete subsequence: α-covering between the instances of LCP (Longest Common Prefix) intervals and β-covering between LCP intervals to support the motif discovery: if an LCP interval is βuncovered, its instances form a motif. The βUncover algorithm of LiSAM identifies the β-uncovered l-intervals, in which we introduce two LCP tabs: *presuf* and *nextsuf* to support the identification of the α-uncovered instances of an l-interval. Experimental results on Electrocardiogram signals indicate the accuracy of LiSAM on finding motifs with different lengths.

Keywords: motif discovery, suffix array, time series

1 Introduction

Discovering motifs for time series is an important and tough task. It has been proved that the subsequence clustering is meaningless in unsupervised data stream mining area, and the motif grouping in the discrete data stream mining has been applied as a replacement of the subsequence-clustering in the real-time series [1]. In this paper, we focus on two primary issues in the time series motif discovery: reducing the computational complexity and avoiding unexpected repetitions among different motifs and among instances of one motif.

The subsequence trivial matching [2] and the overlapping among different motifs [3] are two types of motif repetition issues in the literature. To avoid trivial matching, some methods assumed that the instances of a motif do not overlap with each other at all [4]. We believe that, however, a more flexible and user-manageable mechanism is necessary to control the numbers and styles of the discovered patterns.

Enhancing the time and space complexity, and at the same time, guarantying an expected accuracy is always one of the top topics in data processing. Some motif discovery researchers used approximate solutions to get an acceptable computational complexity [5]. In this work, we propose an unsupervised Limited-length suffix array based Motif Discovery algorithm (LiSAM) for continuous time series, which is time and space efficient, and supports approximately discovering motifs in different lengths. We first

[*] Corresponding Author

convert the continuous time series to the discrete time series by using the Symbolic Aggregate approXimation procedure (SAX) [6], and then identify the different-length motifs based on the discrete time series. Our illustration of discrete motif discovery is on the basis of an exact substring matching procedure, however, we can easily embed the existing approximate substring matching methods, such as [7-8], in LiSAM. The distinctive contribution of LiSAM is as below:

- *LiSAM* can discover motifs in different lengths (e.g., *maxLength* to *minLength* provided by users), avoid the unexpected trivial-matching by allowing user-defined overlapping degree (represented as α) between the instances of motifs, and support discovering motifs that overlap with each other in a specified degree (β). It can either be an automatic or semi-automatic algorithm by either manually setting all the parameters or by using default parameters (e.g., set $maxLength = \frac{1}{2}|T|$ is a time series), $minLength = 2$, $\alpha = 0$ and $\beta = 0$).
- We conduct extensive experiments based on both synthetic time series datasets to evaluate the performance of *LiSAM*. Experimental results show the high accuracy of *LiSAM* and its applicability in the pattern recognition of data streams such as ECG.

2 Related Work and Background Knowledge

2.1 Related Work

There has been a large amount of effort on exploring approximately accurate and fast motif discovery algorithms in continuous time series. The SAX [9] (Symbolic Aggregate approXimation) method was proposed to symbolize the continuous time series. The SAX method can lower bound the distance between the original time series based on the symbolized time series. Because of its time efficiency, it supports a streaming time series conversion. Based on the SAX, Moskovitch et al. [10] present a classification framework for clustering multivariate time series, which first transforms continuous time series to symbolic time series; then discovers frequent occurrence patterns based on data mining techniques; and at last designs classifiers based on the identified patterns. Floratou et al. [11] concentrated on improving the accuracy of motif discovery in continuous sequential data. They proposed a suffix tree based algorithm FLAME to find different motifs with high accuracy. As motif discovery is an unsupervised process, it is difficult to manually determine the lengths of the motifs in a time series. Against this problem, Yingchareonthawornchai [12] used a compression-based method to discover motifs with variable lengths. The proposed method also supports the motif evaluation and ranking in terms of their importance to the time series. Xie and Wang [13] developed an algorithm ADCMCST that supports the approximate construction of tree networks of wireless sensor networks, in order to balance the node payload and enhance the network lifetime. AdaBoost is a popular classification method in pattern recognition area. Wen et al. [14] introduced an advanced learning schema based on AdaBoost classification algorithm for vehicle detection, which reduces the complexity of time-consuming.

2.2 Background: Enhanced Suffix Array

We briefly introduce the frequently used symbols and the basic concept of the enhanced suffix array in this section. Readers can refer to [15-16] for more details. We first introduce and list the symbols and their definitions in this paper in Table 1.

A suffix array of S is an integer array (suftab) having values $k \in [0, n]$. An enhanced suffix array (ESA) is a suffix array with a number of additional supporting arrays, where two of them (lcptab and bwttab) will be used in this paper. We use an example of S_{examp} = *aceaceacece* to describe the ESA that is shown in Table.2. The *suftab* keeps the starting positions of suffixes of S in ascending lexicographic order. The definition of *lcptab* is in Table 1. From Table 2, *lcptab*[0] = 0 and *lcptab*[n] = 0.

To group the suffixes that have the longest common prefixes, the concept of LCP interval is proposed. We describe below the definition of an LCP interval.

Table 1. Symboles and definitions

Concepts	Definitions
T	a continuous time series
Σ	a finite ordered alphabet
Σ^*	strings over Σ
Σ^+	Σ^* without null
S	a discrete time series over Σ with length |S| = n
~	$\sim\in\Sigma$, $\sim>\sigma$, $\forall\sigma\in\Sigma$
S [i, j]	substring of S between positions i and j
suftab [suf]	suffix array table of S
presuf [pre]	the suffix index of the previous position of the current suffix in suftab
nextsuf [next]	the suffix index of the next position of the current suffix in suftab
$S_{suftab[i]}$	the i^{th} suffix of S, $i\in[0, n]$
lcptab [i]	Longest common prefix (LCP) of $S_{suf[i-1]}$ and $S_{suf[i]}$
bwttab [i] (bwt)	S [suftab[i]-1], if suf [i] > 0; null, if suf [i]=0
l - interval, l - [i, j]	an LCP interval from index i to index j with length
l - [l, l]	singleton interval (SI): $S_{suf[l]}$
NSI	non-singleton interval
m - [i, j]	m-interval: instances of l interval forming a motif

Table 2. An enhanced suffix array

index	suf	lcptab	bwt	$S_{suf[i]}$
0	0	0	null	aceaceacece~
1	3	6	e	aceacece~
2	6	3	e	acece~
3	1	0	a	ceaceacece~
4	4	5	a	ceacece~
5	7	2	a	cece~
6	9	2	e	ce~
7	2	0	c	eaceacece~
8	5	4	c	eacece~
9	8	0	c	cece~
10	10	0	c	e~
11	11	0	e	~

Definition 1. Given *S* and its Enhanced suffix array, an interval [*i, j*] of *index* (see Table 2), where *i, j* ∈ [0, *n*] and *i* < *j*, is a LCP interval with LCP length ℓ if the following conditions are satisfied: (1) *lcptab*[*i*] < ℓ; (2) *lcptab*[*k*] ≥ ℓ, $\forall k\in[i+1, j]$; (3) *lcptab*[*k*] = ℓ if $\exists k\in[i+1, j]$; (4) *lcptab*[*j* + 1] < ℓ. The LCP interval [*i, j*] with LCP length ℓ can be represented as l_ℓ - [*i, j*].

An LCP interval tree indicates the embedding and enclosing relations between LCP intervals. We describe an example of LCP tree of S_{examp} in Fig. 1. We can see that the root of the LCP tree covers all the suffixes of S_{examp}. The child intervals are the intervals embedded in their father intervals. The leaf intervals do not enclose any NSI. A fast traversing procedure for LCP trees is defined in [16]. Note that in this paper we use l_ℓ to represent an *l*-interval with LCP length ℓ, while use m_ℓ to represent a motif interval (Def.6) with LCP length ℓ. In addition, we refer the normal 'LCP intervals' to non-singleton intervals (NSIs).

3 Problem Definition

In this section, we introduce the basic concepts to be used in LiSAM. A continuous time series *T* is a sequence of real values that have temporal properties. To identify the motifs of a time series, previous work has given different forms of motif definitions [17]. We summarize these definitions and present a comprehensive motif concept in Definition 2.

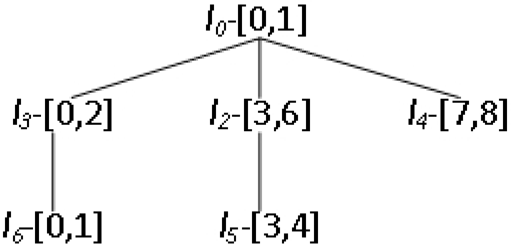


Fig. 1. LCP tree of S_{examp}

Definition 2. A motif M of a time series T is a set of similar subsequence $SQ = \{sq_0, ..., sq_{n-1}\}$ such that $n \geq 2$, and $\forall i, j \in [0, n-1]$, the length of $|sq_i| \geq 2$, $|sq_i \cap sq_j| \leq o$, and $Dis(sq_i, sq_j) \leq d$, where o is an overlapping threshold to constraint the overlapping length between two subsequence of M, Dis is a distance measure, and $d \geq 0$ is a small value to guarantee a certain similarity among subsequence. We call a subsequence of M as an instance of this motif.

Definition 3. Given two l-intervals l_{ℓ_1} - $[i_1, j_1]$ and l_{ℓ_2} - $[i_2, j_2]$, $s_{k1}(k_1 \in [i_1, j_1])$ is an instance of l_{ℓ_1}, $s_{k2}(k_2 \in [i_2, j_2])$ is an instance of l_{ℓ_2}, $sz_1 = |j_1 - i_1 + 1|$: (1) instance s_{k1} is α-covered by s_{k2} if $\ell_1 < \ell_2$, s_{k1} overlaps with s_{k2} at sub-string s'', where $s'' \subset s_{k2}$ and $s'' \subset s_{k1}$, and $|s''| > \alpha$, $|s_{k_1}| \geq \alpha \geq \frac{1}{2} * |s_{k_1}|$. Or else, s_{k1} is α-uncovered by s_{k2}; (2) Interval l_{ℓ_1} is β-covered by l_{ℓ_2}, if h instances of l_{ℓ_1} are covered by the instances of l_{ℓ_2}, where $(sz_1 - \beta) < h \leq sz_1$, and h is a pre-defined threshold. Or else, l_{ℓ_1} is β-uncovered (or uncovered) by l_{ℓ_2}.

From the definition of l-interval, an l_ℓ -interval is composed of at least two suffixes that have the LCP of length ℓ. Therefore, an l-interval can be seen as a pattern of S, and the LCPs of the l-interval correspond to the occurrences of the pattern. A pattern of S is defined as:

Definition 4. Given an alphabet set Σ and an approximate time series $S \in \Sigma^*$, a pattern of S is a time series pt that $1 \leq |pt| \leq |S|, pt \subset S$, and occurs k ($k \geq 2$) times in S at positions $\{p_1, ..., p_k\}$, $p_1 \neq ... \neq p_k$, where a position is the start point of an occurrence of pt in S.

In the above definition, we define that a pattern should occur at least twice in a time series. From the definition of l-interval, an l_ℓ -interval is composed of at least two suffixes that have the LCP of length ℓ. Therefore, an l-interval can be seen as a pattern of S, and the LCPs of the l-interval correspond to the occurrences of the pattern. However, the requirement on the minimum occurrence times of a pattern varies in different situations. For example, in a very long S (e.g., $\geq$ 10 thousands), the element that repeats a small number of times (e.g., $<$ 10 times) is meaningless for the time series analysis. Therefore, we define a general concept of an approximate motif of discrete time series as below.

Definition 5. Assume $u = S[a, b]$ $(a \leq b)$ is an instance of an l-interval l_ℓ -$[i, j]$ of S. Given a lower bound $mint$ ($minT \geq 2$) of the pattern occurrences, if $\varepsilon = j - i + 1 \geq minT$, and l_ℓ is uncovered by any other l-intervals of S, it is an approximate motif of S, represented as $mf = <\ell; P = \{p_1, \cdots, p_\varepsilon\}>$, where $\ell = b - a + 1$ $(l \geq 1)$ is the length of mf, p_i is the start index of the occurrences of u in S, and ε is the size of the motif mf.

In the following description, a motif of S refers to an approximate motif. The relation between an l-interval and a motif of S is defined as an m-interval.

Definition 6. For an l-interval l_ℓ -$[i, j]$ of S, if the instances of l_ℓ is one-to-one matched to the occurrences of a motif $mf = <\ell; suftab[i], \cdots, suftab[j]>$, then l_ℓ is an m-interval, represented as m_ℓ -$[i, j]$.

In the following sections, we refer an m-interval to a motif.

4 Limited-length Suffix-array-based Motif Discovery

4.1 Identify β-uncovered l-intervals for Discrete Time Series

In this section, we first discuss the determination of the β-uncovered intervals with an assumption that $\alpha = 1$ for the α-covering relation between instances. In section 4.2, we introduce the α-covered algorithm and illustrate how to interactively perform the β- and α-uncovered algorithms to identify the motifs.

In ESA, identifying LCP intervals is a bottom-up traversing process. When an LCP interval is being processed, its child intervals have been identified, so the child intervals can support the determination of β-covering of the LCP interval. We distinguish the case of an LCP interval having a single character (the singleChar interval) with the case that the interval is comprised of more than one character (the multiChar interval). We give Lemma 1 to identify the β-uncovered multiChar intervals.

Lemma 1. Given an multiChar LCP interval l_ℓ - $[i, j]$, its child intervals Θ, and the lower bound of the occurrence times of motifs $minT \geq 2$, let $\lambda = j - i + 1$, l_ℓ is β-uncovered by other l-intervals if any of the following conditions is satisfied:

(1) $|\Theta| = 0$, $\lambda = minT$ and $bwttab[i, j]$ are pair-wise different, i.e., $bwttab[i] \neq \ldots \neq bwttab[j]$;

(2) $|\Theta| = 0$, and $\exists \sigma_1 \neq \ldots \neq \sigma_\gamma$, $\sigma_{1, \ldots, \gamma} \in$ bwttab[i ... j], $\mathrm{minT} + 1 \leq \gamma \leq \lambda$;

(3) $|\Theta| > 0$, $\exists\, l_{\ell_1}$ - $[w_1, z_1]$, $l_{\ell_1} \in \Theta$ and $\lambda_\theta = z_1 - w_1 + 1 \geq minT$, and $\exists\, r_1 \ldots r_k \in [w_1, z_1]$ and $h_1 \ldots h_k \in [i, j]$ but $\notin$ $[w_1, z_1]$ that $bwttab[r_1] \neq bwttab[h_1]$, ..., $bwttab[r_k] \neq bwttab[h_k]$, $k \geq minT$.

(4) $|\Theta| > 1$, $\exists\, m_{\ell_1}$ - $[w_1, z_1]$, ..., m_{ℓ_k} - $[w_k, z_k] \in \Theta$, $k \geq minT$, and m_{ℓ_1} ,..., m_{ℓ_k} are β-uncovered.

Proof of Lemma 1.

(1) $|\Theta| = 0$, so the characters after the LCP subsequences of l_ℓ are pair-wise different, i.e., $S[suftab[i] + \ell\,] \neq S[suftab[j] + \ell\,]$. Meanwhile, $\lambda = minT$ and $bwttab[i] \neq \ldots \neq bwttab[j]$. So the instances of l_ℓ are not covered by any longer repeated sequences in S. Hence, l_ℓ is β-uncovered.

(2) if $\gamma > minT$, then at least $minT + 1$ characters in $bwttab[i, j]$ are different (assume $bwttab[k_1] \neq bwttab[k_2]$); and as $\Theta = 0$, the $k_1 th$ and $k_2 th$ LCP subsequences are not covered by any longer subsequences of its child intervals. So l_ℓ is β-uncovered.

(3) assume l_ℓ have one child interval c_θ, where $\lambda_\theta \geq minT$, $i \leq w_\theta \leq z_\theta \leq j$ and $\lambda > minT$. (a)Assume $\lambda - \lambda_\theta = 0$, then $l_\ell = c_\theta$, c_θ is not a child interval of l_ℓ. Assumption (a) is not true. (b) Assume $\lambda - \lambda_\theta < minT$, then there are $\lambda - minT$ instances of l_ℓ covered by the instances of c_θ, so interval l_ℓ is covered by interval c_θ, and l_ℓ is not a motif. Assumption (b) is not true. (c) as $\lambda - \lambda_\theta \geq minT$, then there are at least $minT$ instances of l_ℓ that are not covered by the instances of c_θ. In addition, $\exists \sigma_1 \neq \ldots \neq \sigma_\gamma$, $\sigma_{1, \ldots, \gamma} \in bwttab[i \ldots j]$, $minT < \gamma \leq \lambda$, based on the proof of (3), l_ℓ is β-uncovered.

(4) if $k = minT$, as m_{ℓ_1}, ..., m_{ℓ_k} are k motifs, the subsequences in all of the $minT$ intervals are pairwise different, so the interval l_ℓ, where $\ell < \ell_1$, ..., ℓ_{minT}, cannot be covered by any of $\{m_{\ell_1}$ (as $\forall |m_{\ell_t}| \geq minT$, $t \in [1, k]$, $t \neq 1$), ..., $m_{\ell_{minT}}\}$, that is, the interval l_ℓ cannot be individually covered by any of its k child motifs. So l_ℓ is β-uncovered.

For singleChar intervals, the problem of determining their motif property is to avoid finding a shorter singleChar motif covered by a longer singleChar motif. Lemma 2 shows how to determine if a singleChar interval is β-uncovered.

Lemma 2. Given a singleChar interval l_ℓ - $[i, j]$ that its LCP subsequence, i.e., $S[suftab[i], suftab[i] + \ell - 1]$, is only comprised of one character (assume σ),

(1) if l_ℓ does not have child intervals, i.e., $|\Theta| = 0$ and $\exists \sigma_1 \neq \ldots \neq \sigma_\gamma$, $\sigma_{1, \ldots, \gamma} \in bwttab[i \ldots j]$, $minT + 1 \leq \gamma \leq \lambda$, then l_ℓ is β-uncovered;

(2) if $|\Theta| > 0$ and θ - $[w, z] \in \Theta$, that $\exists \sigma'_1 \neq \ldots \neq \sigma'_\lambda \neq \sigma$ and $\sigma'_{1 \ldots \lambda} \in bwttab[w' \ldots z']$, where$z'$ - $\mathrm{w}' + 1 \geq 2$, $\lambda > 0$, $[w' .. z'\,] \subset [i \ldots j]$ and $[w' \ldots z'\,]$ is β-uncovered by $[w \ldots z]$;

Proof of Lemma 2.

(1) As l_ℓ does not have child intervals, l_ℓ cannot be covered by an interval comprising LCP subsequences of $u^{'} = S[suftab[k_1], ..., suftab[k_1] + \ell^{''} - 1]$, where $k_1 \in [i, j]$, $\ell^{'} > \ell$. In addition, as $\exists \sigma_1 \neq ... \neq \sigma_\gamma$, $\sigma_{1, ..., \gamma} \in bwttab[i ... j]$, $minT+1 \leq \gamma \leq \lambda$, l_ℓ cannot be covered by an interval comprising LCP subsequences of $u^{''} = S[suftab[k_2] - 1, ..., suftab[k_2] - 1 + \ell^{''} - 1]$, where $k_2 \in [i, j]$, $\ell^{''} > \ell$. So l_ℓ is a β-uncovered.

(2) Assume $u = S[suftab[i] ... suftab[j] + \theta - 1]$ is the prefix of l_ℓ, and $u^{'} = S[suftab[w] ... suftab[w] + \theta - 1]$ is the prefix of l_θ, and assume $\exists \sigma_1 \in bwttab[w ... z]$ and $\exists \sigma_2 \in bwttab[w^{'}, z^{'}]$ that $\sigma_1 \neq \sigma$ and $\sigma_2 \neq \sigma$, then (1) anychild interval l_θ cannot cover l_ℓ, since $z^{'} - w^{'} + 1 \geq 2$; (2) we prove that under condition 2 in Lemma 2, if l_ℓ is a singleChar interval with LCPs like $\mu = x_1, ..., x_\ell$, then not $\exists l_\theta$ (the strings of its *singleChar* LCP $\mu = x_1, ..., x_\theta (\theta > \ell)$ that cover l_ℓ. Assume exist such l_θ, then the strings of the LCP of l_θ include all the stings whose prefixes with length θ are $u^{'}$, i.e., $\exists k (= z - w + 1)$ subsequences $u \subset S$, and there must be $\eta (= k * \ (\theta - 1))$ *bwttabs* that $bwttab[r_1] = ... = bwttab[r_\eta] = \sigma$, $\eta = z^{'} - w^{'} + 1$ and $k + \eta = j - i + 1$; which means there must not exist $\sigma^{'}_{1, ..., \lambda} \neq \sigma$, $\lambda > 0$ in bwttab$[w^{'}, z^{'}]$. This is contradicting with condition 2 of Lemma2, so the second statement (2) is correct. Combining statements (1) and (2), the singleChar interval l_ℓ is β-uncovered given condition 2 of Lemma 2.

5 Performance Evaluation

In this section, we present the experimental results to show the efficiency of LiSAM. Our experiments are conducted on a windows 64-bit system with 3.2GHz CPU and 4 GB RAM, and is implemented by *Java*.

We extract patterns from six different ECG data streams, repeat each pattern 30 times and insert the repeated patterns to Gaussian white noise data streams separately. The information of the extracted patterns and the parameter settings is shown in the top part of Table 3. The first three datasets are from the UCR Time Series Classification Archive [18], and the other three are from the Physionet [19]. Particularly, the *nL* is the length of a piece of noise subsequence between two pieces of a pattern. We use the fixed-length intervals (i.e., length of noise subsequence) between two pattern subsequences to make the annotation of the pattern instances easy. Column *sL* sets the parameters of the SAX-based symbol conversion, representing the length of a subsequence that corresponds to a symbol. Columns *maxM* set the upper bounds of the lengths of the discovered patterns. The lower bounds of the lengths of the discovered patterns for all datasets are set as 10.

Table 3. Dataset settings & old and InDis performance

Datasets	nL	sL	maxM	old	inDis
ECG200	50	2	100	0.9892	0.0076
ECGfivedays	50	2	140	0.9924	0.0076
ECGtorse	100	10	1640	0.9947	0.0068
ECGtwa01	150	3	300	0.9933	0.0086
ECGsvdb800	150	2	170	0.9939	0.0112
ECGmitdb100	150	2	150	0.9966	0.006
LTDB14134	-	2	150	-	-
SVDB800	-	2	150	-	-
AHADB0001	-	2	120	-	-
CARTI01	-	2	100	-	-

We use *old* (see equation 1.) to measure the accuracy of the discovered motifs, which represents the overlapping degree between the inserted pattern (p_i) and the discovered pattern (d_j):

$$old = \frac{\sum_i \sum_j overlap(p_i, d_j)}{length(plantedPattern)} \quad \textbf{(1)}$$

The *old* values for each of the simulated ECG time series are shown in Table 4. We can see that the proposed motif discovery algorithm can identify the inserted patterns with very high accuracy (all over 0.9). We compare the shapes of the planted patterns and the discovered motifs in each of the six time series in Fig. 2. In addition, we use the average pair-wise distances among instances (represented as *inDis*) of a motif to measure the dis-similarity degree of the instances of one discovered motif (e.g., motif m), which is calculated as:

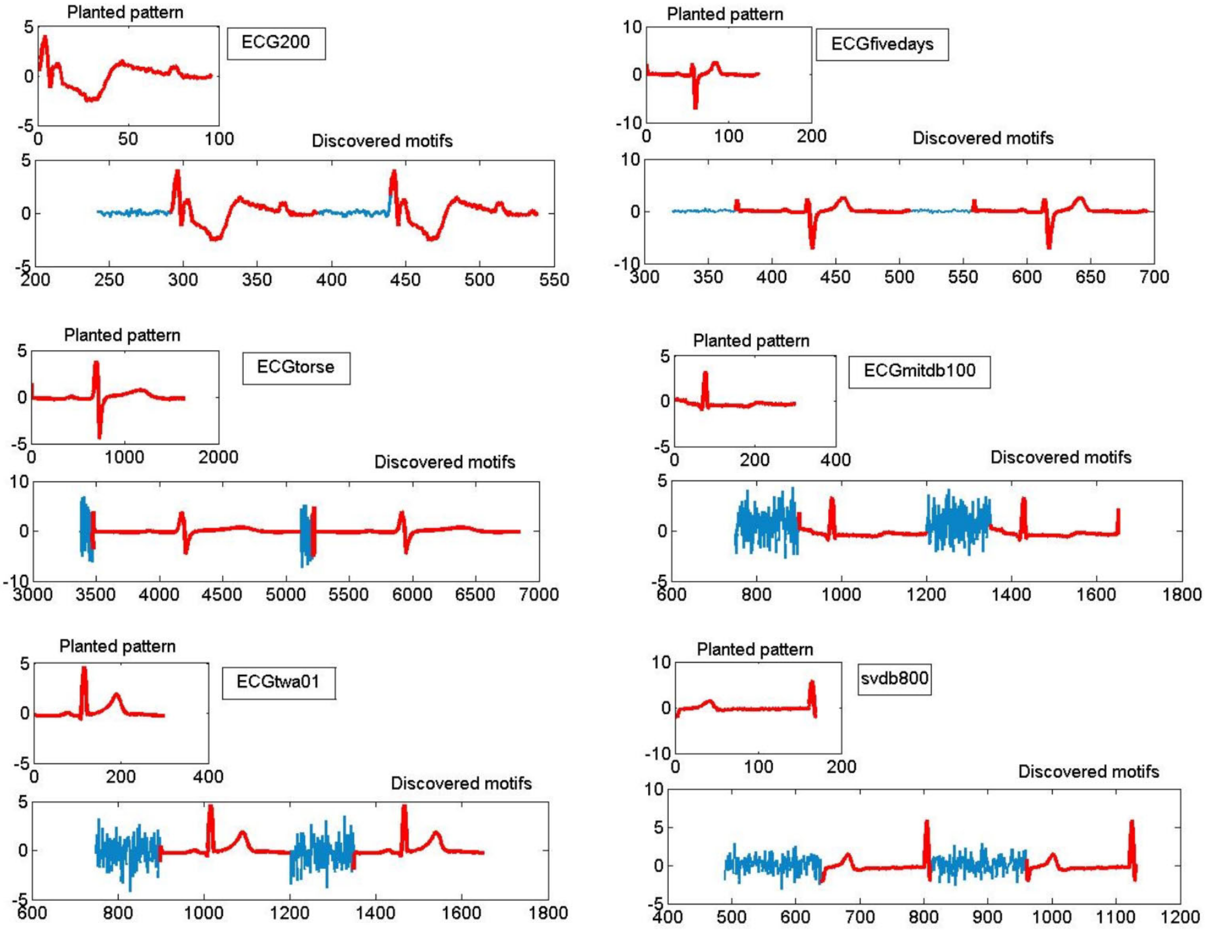


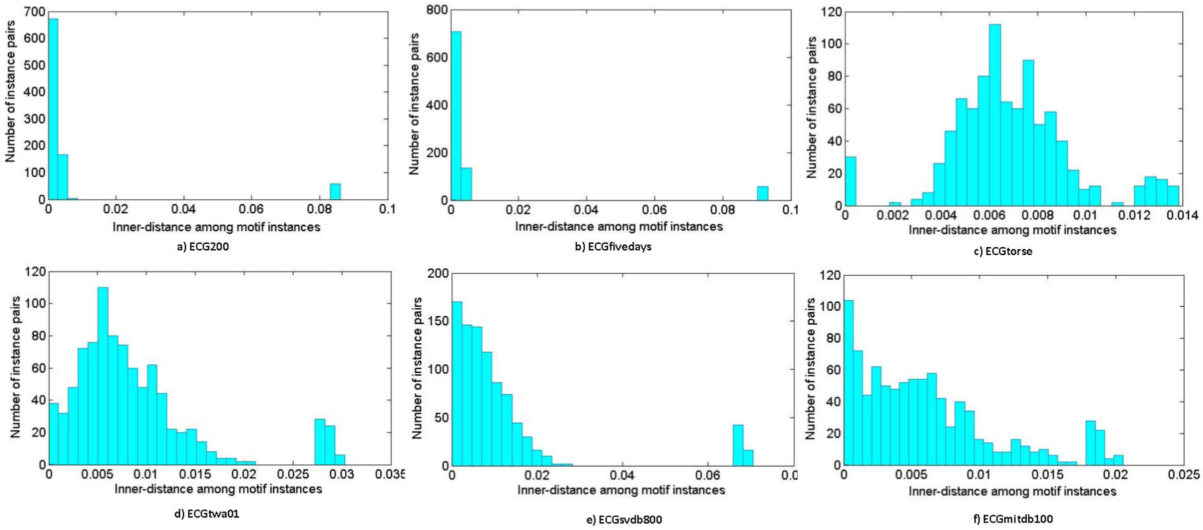
Fig. 2. Planted patterns and discovered motifs

$$inDis(m) = \frac{\sum_{i,j} dis(m_i, m_j)}{m.len * m.size} \tag{2}$$

where m_i and m_j represent the *ith* and *jth* instances of *m*; and *m.len* is the length of this motif; *m.size* is the number of its instances, and *dis* is the Euclidean distance function. The average *inDis* value of each time series is shown in Table 4, and the distance distribution of each instance pair of the most frequent motif for each dataset is shown in Fig. 3. We can see that the instances of one motif for each datasets are very close to each other, all of which have less than 0.1 average instance dissimilarities.

6 Conclusion and Future Work

In this paper, we proposed an algorithm LiSAM to resolve two important problems in discovering approximate time series motif: releasing the constraints of trivial matching between subsequences with different lengths and improving the time and space efficiency. We proposed two covering relations: *α*-covering between instances of *l*-intervals and *β*-covering between *l*-intervals to support the motif discovery. We use the LiSAM algorithm to identify the *β*-uncovered *l*-intervals, and we introduced two LCP tabs: *presuf* and *nextsuf* to support the identification of the *α*-uncovered instances of an *l*-interval.



a) ECG200

b) ECGfivedays

c) ECGtorse

d) ECGtwa01

e) ECGsvdb800

f) ECGmitdb100

Fig. 3. Distance distribution of instance pairs of the most frequent motif for six datasets

Acknowledgements

This paper is supported by the National Natural Science Foundation of China (Grants No. 61702274) and the Natural Science Foundation of Jiangsu Province (Grants No. BK20170958).

References

[1] S. Aghabozorgi, A.S. Shirkhorshidi, T.Y. Wah, Time-series clustering a decade review, Information Systems 53(2015) 16-38.

[2] R. Anirudh, P. Turaga, Geometry-based symbolic approximation for fast sequence matching on manifolds, International Journal of Computer Vision 116(2)(2016) 161-173.

[3] J. Grabocka, N. Schilling, L. Schmidt-Thieme, Latent time-series motifs, ACM Transactions on Knowledge Discovery from Data 11(1)(2016) 6:1-6:20.

[4] D. Minnen, C.L. Isbell, I. Essa, T. Starner, Discovering multivariate motifs using subsequence density estimation and greedy mixture learning, in: Proc. the 22nd National Conference on Artificial Intelligence, 2007.

[5] T. Sun, H. Liu, H. Yu, C.L.P. Chen, Degree-pruning dynamic programming approaches to central time series minimizing dynamic time warping distance, IEEE Transactions on Cybernetics 47(7)(2017) 1719-1729.

[6] P. Nickerson, R. Baharloo, A.A. Wanigatunga, T.D. Manini, P.J. Tighe, P. Rashidi, Transition icons for time series visualization and exploratory analysis, IEEE Journal of Biomedical and Health Informatics PP(99)(2017) 1-1.

[7] A. Bottrighi, G. Leonardi, S. Montani, L. Portinale, P. Terenziani, A time series retrieval tool for sub-series matching, Applied Intelligence 43(1)(2015) 132-149.

[8] M.G. Baydogan, G. Runger, Time series representation and similarity based on local auto-patterns, Data Mining and Knowledge Discovery 30(2)(2016) 476-509.

[9] M. Gupta, J. Gao, C.C. Aggarwal, J. Han, Outlier detection for temporal data: a survey, IEEE Transactions on Knowledge and Data Engineering 26(9)(2014) 2250-2267.

[10] R. Moskovitch, Y. Shahar, Classification-driven temporal discretization of multivariate time series, Data Mining and Knowledge Discovery 29(4)(2015) 871-913.

[11] A. Floratou, S. Tata, J. Patel, Efficient and accurate discovery of patterns in sequence data sets, IEEE Transactions on Knowledge and Data Engineering 23(8)(2011) 1154-1168.

[12] S. Yingchareonthawornchai, H. Sivaraks, T. Rakthanmanon, C. Ratanamahatana, Efficient proper length time series motif discovery, in: Proc. IEEE 13th International Conference on in Data Mining (ICDM), 2013.

[13] S. Xie, Y. Wang, Construction of tree network with limited delivery latency in homogeneous wireless sensor networks, Wireless Personal Communications 78(1)(2014) 231-246.

[14] X. Wen, L. Shao, Y. Xue, W. Fang, A rapid learning algorithm for vehicle classification, Information Sciences 295(2015) 395-406.

[15] T.D. Wu, Bitpacking techniques for indexing genomes: enhanced suffix arrays, Algorithms for Molecular Biology 11(1)(2016) 9.

[16] M.I. Abouelhoda, S. Kurtz, E. Ohlebusch, Replacing suffix trees with enhanced suffix arrays, Journal of Discrete Algorithms 2(1)(2004) 53-86.

[17] A. Mueen, E. Keogh, Q. Zhu, S. Cash, M. Westover, N. BigdelyShamlo, A disk-aware algorithm for time series motif discovery, Data Mining and Knowledge Discovery 22(1-2)(2011) 73-105.

[18] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, The UCR time series classification archive. <www.cs.ucr.edu/~eamonn/time_series_data/>, 2015 (accessed September, 2017)

[19] A.L. Goldberger, L.A. Amaral, L. Glass, J.M. Hausdorff, P.C. Ivanov, R.G. Mark, J.E. Mietus, G.B. Moody, C.-K. Peng, H.E. Stanley, Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals, Circulation 101(23)(2000) e215-e220.