Time Series Motif Discovery and Anomaly Detection Based on Subseries Join

Yi Lin * Michael D. McCool † Ali A. Ghorbani ‡

Abstract — **Time series are composed of sequences of data items measured at typically uniform intervals. Time series arise frequently in many scientific and engineering applications, including finance, medicine, digital audio, and motion capture.**

Time series motifs are repeated similar subseries in one or multiple time series data. Time series anomalies are unusual subseries in one or multiple time series data. Finding motifs and anomalies in time series data are closely related problems and are useful in many domains, including medicine, motion capture, meteorology, and finance.

This paper presents a novel approach for both the motif discovery problem and the anomaly detection problem. First, we use a subseries join operation to match similar subseries and to obtain similarity relationships among subseries of the time series data. The subseries join algorithm we use can efficiently and effectively tolerate noise, time-scaling, and phase shifts. Based on the similarity relationships found among subseries of the time series data, the motif discovery and anomaly detection problems can be converted to graph-theoretic problems solvable by known graph-theoretic algorithms. Experiments demonstrate the effectiveness of the proposed approach to discover motifs and anomalies in real-world time series data. Experiments also demonstrate that the proposed approach is efficient when applied to large time series datasets.

Keywords: ***pattern recognition, motif discovery, anomaly detection, time series, subseries join, graph-theoretic algorithm***

1 Introduction

A time series is composed of a sequence of values measured from a continuous signal. The term "time series"" is often used to refer to any such sampled data set with one independent variable, whether or not that independent variable is time. Examples of time series data include stock prices, scientific measurements, weather data, music, and motion capture data.

Motifs are approximately repeated subseries in a single time series data or a time series dataset. For example, in Figure 1(a), the underlined parts are repeated similar subseries. Anomalies are unusual subseries in a single time series data or a time series dataset. Anomalies can be outliers in a time series that contains approximately periodic patterns, or a part of a time series that deviates significantly from some typical behaviour. For example, in Figure 1(b), the underlined part is quite different from the other parts of the data that is sine-like.

Motif discovery and anomaly detection are fundamental tasks and are useful in many real-world applications. They are often used to analyse other kinds of data. For example, motifs in Web posts or comments consist of frequently repeated term-based patterns, and can be used for analyzing people's opinions or feedback. For an example based on time series, consider motifs in the context of stock prices. Such patterns may be used to predict trends. Likewise, anomaly detection applied to cardiogram data could help a doctor to diagnose heart disease. Anomaly detection in network traffic analysis could alert the network administrator of spam or malicious attacks. Given such wide applicability, motif and anomaly detection are attracting increasing attention from both academia and industry.

Existing approaches usually handle motif discovery and anomaly detection separately [1, 2]. They are also limited in that they can only find similar patterns of the same length or tolerate a limited amount of time-scaling and phase-shifting. Some of the existing approaches also have high computational complexities. Finally, in order

*Faculty of Computer Science, University of New Brunswick, 540 Windsor Street Fredericton, NB, Canada E3B 5A3. Tel: 1-506-458-7141. Fax: 1-506-453-3566. Email: ylin@unb.ca.

†Intel of Canada, Ltd. 180 King St S, Suite 500, Waterloo, ON, Canada N2J 1P8. Email: michael.mccool@intel.com.

‡Faculty of Computer Science, University of New Brunswick, 540 Windsor Street Fredericton, NB, Canada E3B 5A3. Email: ghorbani@unb.ca.

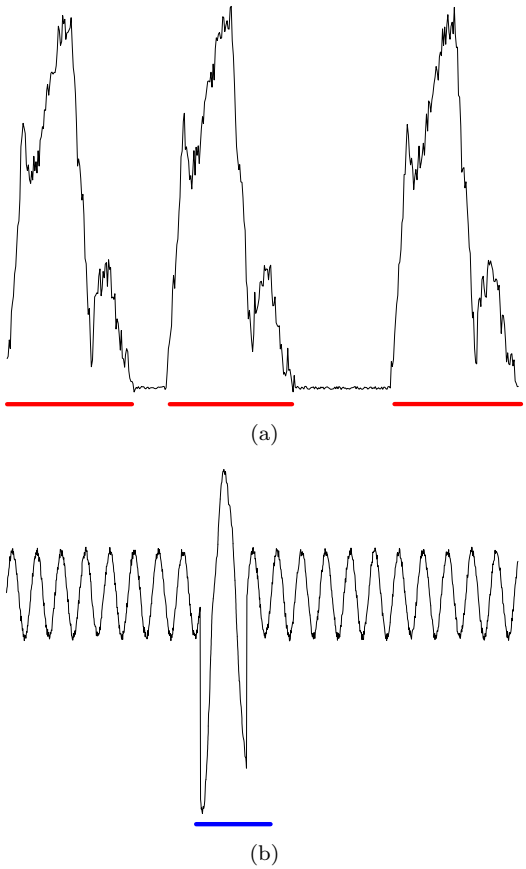


Figure 1: Simulated examples of time series motifs and anomalies. (a) The approximate repeated subseries that are underlined are a motif of this time series. (b) The unusual subseries that is underlined is an anomaly in this time series.

to solve the problem, we first have to define it well.

In this work, we propose an approach for both motif discovery and anomaly detection, based on a combination of subseries join and graph theoretic definitions. This approach is based on a definition of subseries join (and an efficient algorithm for solving it) that were proposed in our previous work [3, 4]. Based on subseries join, the motif discovery and anomaly detection problems can be well-defined as graph-theoretic problems, then both problems can be solved using known graph-theoretic algorithms. This paper also includes more details on the subseries join algorithm than could be included in our previous paper [5].

The rest of the paper is organized as follows: In Section 2, we review previous work and underlying technology for time series motif discovery and anomaly detection. In Section 3, we introduce the subseries join operation and our algorithm for solving it. Section 4 presents our motif discovery and anomaly detection definitions and solution methods. We perform empirical evaluations in Section 5 to evaluate the effectiveness and efficiency of the proposed approach. Finally Section 6 provides conclusions and suggestions for future work.

2 Related Work

The discovery of time series motifs is widely used in various medical applications, including examining the data from on-body monitoring sensors [6] and selecting maximally informative genes [7]. Time series motifs are also used in finding patterns in sports motion capture data [8] and in video surveillance applications [9].

To the best of our knowledge, the first formal definition of time series motifs was proposed in 2003 [10]. Based on this work, a fast motif discovery algorithm was recently presented [1]. However, this work is limited to discovering motifs whose subseries are all of the same length. Yankov et al. [11] proposed a motif discovery method that uses a uniform scaling Euclidean distance and a symbolic representation based on thresholding. Uniform scaling is important for indexing and matching time series for motion-capture data [12] and music. The thresholds they used to convert a time series to a symbolic sequence are heuristically determined. This method is also only semi-automatic because the user needs to specify the length of the motif subseries manually. This arbitrary segmentation may cause undesirable division of important features in the data into different segments. Overlapping sliding windows can be used [13] to avoid division of features but at the cost of a redundant representation. Generally, a better definition of a "motif" that does not depend on *a priori* knowledge of its shape or length is needed.

Wei et al. proposed an anomaly detection algorithm based on a symbolic representation of time series [14]. This representation was later used in finding unusual shapes in a large image database [2]. This symbolic representation divides a time series into segments of uniform length. However, the manual section of the segment length is not adaptable to the actual behavior of the data. Wavelet analysis techniques have been widely used for network intrusion detection [15, 16]. These techniques can adapt to features at different scales. However, wavelet analysis-based anomaly detection techniques often have high computation complexity.

3 Subseries Join

Efficiently searching similarities in a large time series dataset is still a challenging problem, especially when partial or subseries matches are needed. Much of previous work focused on either whole match or subseries match problems. Whole match finds one or more time series in a dataset that are similar to a given query time series. Whole match is a one-to-one match. One example of whole match is shown in Figure 2(a). Subseries match finds similar segments of time series in a time series dataset given a single query time series. Subseries match is a one-to-many match, since a single query may match at different places in a single target time series. One example of subseries match is shown in Figure 2(b). However, whole match and subseries match may not work in many scenarios. For example, when time series are very long and contain complex features, whole match may find many meaningless matches. These forms of match can also do nothing unless a query time series is available. However, in many cases, it may not be easy to provide a query series. For example, in anomaly detection, we don't know in advance what the anomaly looks like. The same is true for motif detection: we don't know in advance what the motif looks like.

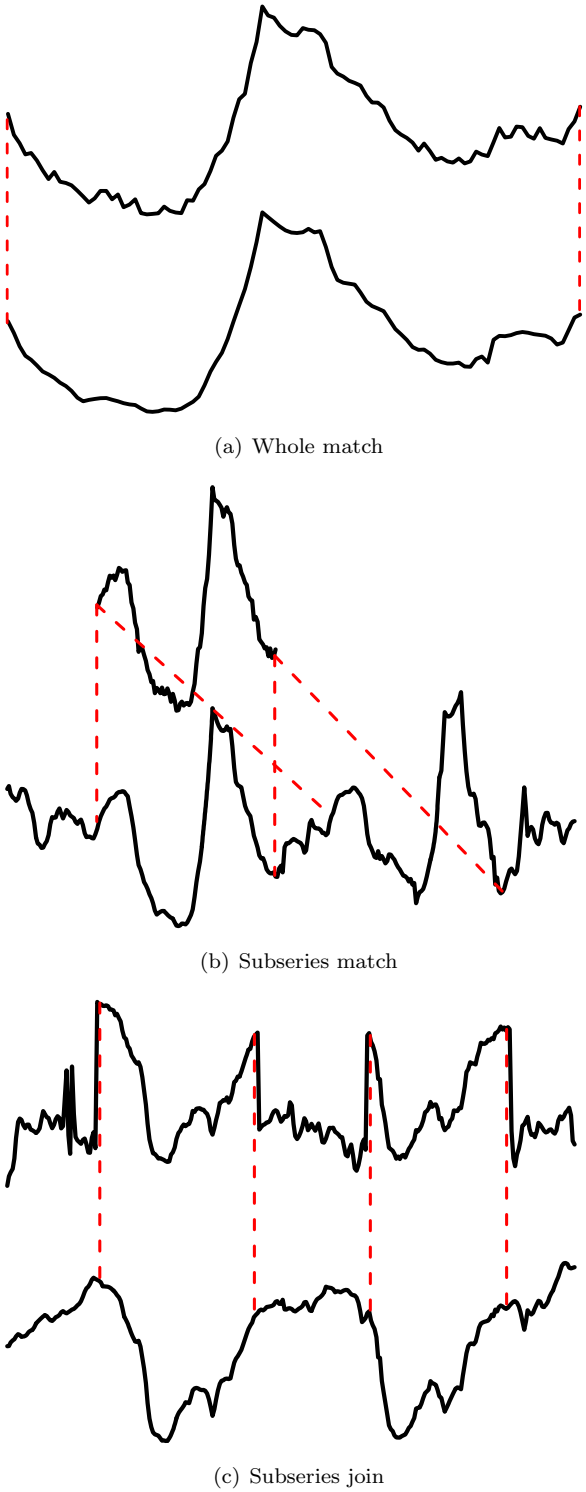
To deal with the above issues, we defined a *subseries join* operation [4]. A subseries join finds pairs of similar segments of time series in two time series datasets.

Subseries join produces many-to-many matches and is a generalization of both whole match and subseries match. One example of subseries join is shown in Figure 2(c).

Our formal definition of subseries join is a generalization of subseries match. Subseries join is a symmetric operation that returns all pairs of subseries drawn from two datasets that satisfy a given similarity threshold ϵ relative to some metric d and are also of maximal length:

Definition 1 ***Subseries join:*** *Given two sets of time series $\mathcal{X}$ and $\mathcal{Y}$, the subseries join is the set of all pairs $(X_{i,k}, Y_{j,\ell})$ of subseries $X_{i,k} \subseteq X_k$ for $X_k \in \mathcal{X}$ and $Y_{j,\ell} \subseteq Y_\ell$ for $Y_\ell \in \mathcal{Y}$ such that $d(X_{i,k}, Y_{j,\ell}) \leq \epsilon$, and for which there does not exist any $X'_{i,k} \supset X_{i,k}$ and $Y'_{j,\ell} \supset Y_{j,\ell}$ where $X'_{i,k}$ is longer than $X_{i,k}$ and contains $X_{i,k}$ as a proper subset and where $Y'_{j,\ell}$ is longer than $Y_{j,\ell}$ and contains $Y_{j,\ell}$ as a proper subset for which $d(X'_{i,k}, Y'_{j,\ell}) < \epsilon$ or for which $d(X_{i,k}, Y'_{j,\ell}) < \epsilon$ or for which $d(X'_{i,k}, Y_{j,\ell}) < \epsilon$.*

Note that a subseries join computes a subseries match if one of the input datasets $\mathcal{X}$ is a singleton set $\{X\}$. In other words, subseries join does not require a "query"



(a) Whole match

(b) Subseries match

(c) Subseries join

Figure 2: Whole match, subseries match, and subseries join.

but can support query-like operations as a special case. As another special case, which we will in fact use for motif and anomaly detection, a dataset can be joined with itself.

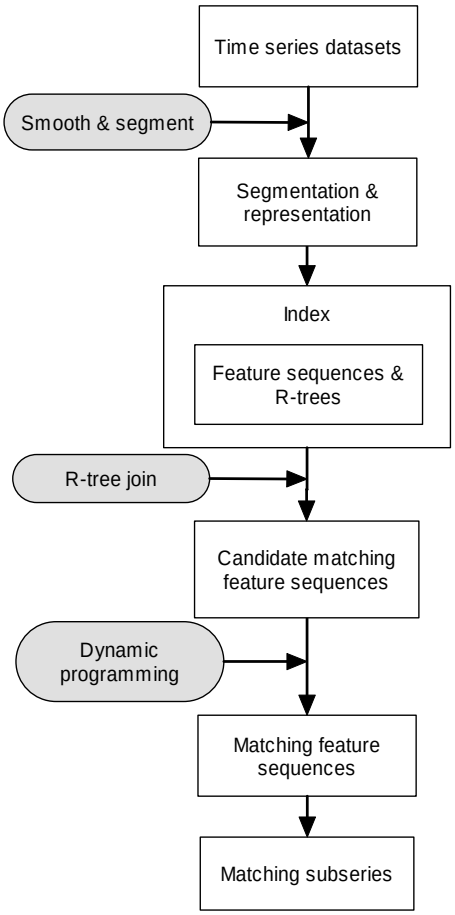


Figure 3: Overview of the proposed approach.

The subseries join results include similar subseries but exclude dissimilar subseries. The returned similar subseries consist of motifs, while the excluded dissimilar subseries without matches may contain anomalies.

In the following, we will briefly describe the subseries join algorithm that are presented in our previous work [4]. This subseries join algorithm will be applied to efficiently solve the joint motif discovery and anomaly detection problem.

The overview of our approach to solving the subseries join problem is shown in Figure 3. First, each time series in the dataset is smoothed by an anisotropic diffusion analysis operation [17] before being broken into segments by a Canny edge detector [18]. A minimal polynomial envelope is then used to represent each segment in a reduced-dimensionality space. The tuple of parameters used to represent each segment is called a *feature*. Note that a feature is represented by a finite number of parameters, no matter how long the corresponding segment is. This feature representation automatically segments time series at its own intrinsic discontinuities, which is not possible in most of previous work. This avoids inappropriately breaking up subseries with arbitrary segmentation.

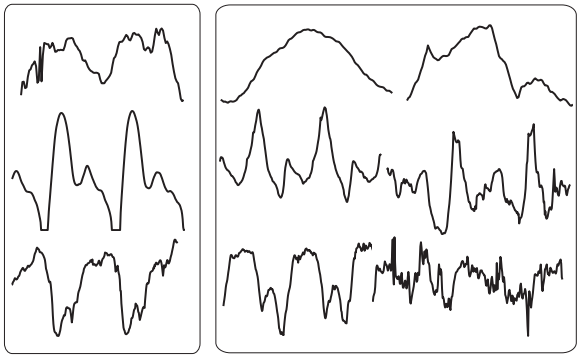
We then convert the problem of computing pairs of approximately matching features into an equivalent spatial database search. All features in the dataset are indexed by an R-tree, and then pairs of matching features are found by an efficient R-tree join operation [19]. We use an encoding of features so the Euclidean distance used by the R-tree corresponds exactly to a computation of a suitable distance between the time series shapes encoded by the features. Matching features returned by the R-tree search provide candidates for generating matching feature sequences. We use a dynamic programming algorithm to compare and align candidate matching feature sequences. This dynamic programming algorithm can tolerate small misalignments, as well as insertions and deletions of features, while the encoding of the features themselves can tolerate time scaling.

Figure 4 illustrates the important steps of our subseries join algorithm. Given two time series datasets shown in Figure 4(a), time series in the datasets are non-uniformly segmented and pairs of matching subseries are found using dynamic programming, which is shown in Figure 4(b). Figure 4(c) shows a set of pairs of matching subseries that satisfy a similarity threshold and are maximal-length.

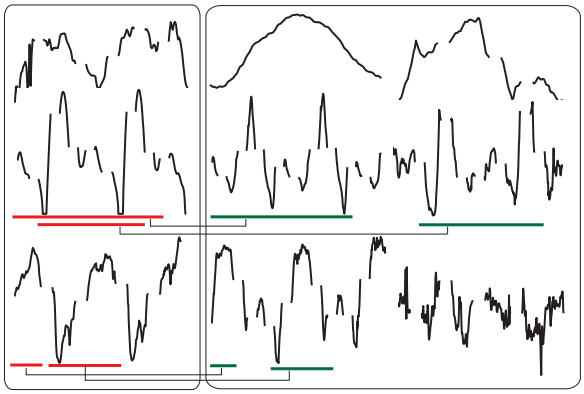
The main stages of our subseries join algorithm include smoothing and segmentation, feature representation, indexing, and feature sequence alignment. In the following, we will introduce and discuss each part separately.

3.1 Smoothing and Segmentation

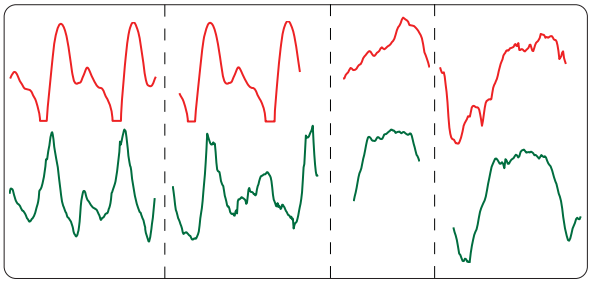
Perona and Malik [17] proposed a noise reduction method for images using anisotropic diffusion, which can be seen as a generalization of Gaussian smoothing but without the discontinuity migration problem. The Perona-Malik method uses a filter defined as a modified diffusion process that encourages intra-region smoothing while inhibiting inter-region smoothing. Their smoothing process can avoid the blurring and localization problems of filters based on convolution. Although alternatives are possible, including bilateral filters [20], anisotropic diffusion filters are preferred because they automatically generate a scale space that maintains the positions of discontinu-



(a) Two time series datasets.



(b) Segmenting and matching subseries.



(c) Subseries join results.

Figure 4: Middle results of subseries join.

ities. The term "anisotropic" generally means that the smoothing (diffusion) process is different in different directions. In 2D, this refers to different radial directions around a point. In 1D, it simply means that the smoothing to the left may be different from the smoothing to the right at each point.

Given a continuous signal $X(x)$, the continuous form of the 1D anisotropic diffusion filter is given by the solution to

$$\frac{\partial}{\partial s}X(x,s) = \frac{\partial}{\partial x}\cdot\left(c(x,s)\frac{\partial}{\partial x}X(x,s)\right) \qquad (1)$$

where s is scale. The function c is a conductance function that returns a value in the range of $[0,1]$. It is a function of the gradient of X and should be a monotonically decreasing function of the gradient's magnitude. One of the following definitions can be chosen:

$$c(x,s) = \exp\left(-\left(\kappa^{-1}\frac{\partial}{\partial x}X(x,s)\right)^2\right) \qquad (2)$$

$$c(x,s) = \frac{1}{1+(\left|\frac{\partial}{\partial x}X(x,s)\right|/\kappa)^2} \qquad (3)$$

If we discretize Equation 1 using the sequence $X = (x_1,\ldots,x_n)$ and replace the scale s with the number of iterations $\sigma = \lambda^{-1}s$, we get the following implementation:

$$x_i^{\sigma+1} = x_i^{\sigma}+\lambda c(i+1,\sigma)(x_{i+1}^{\sigma}-x_i^{\sigma})-\lambda c(i-1,\sigma)(x_i^{\sigma}-x_{i-1}^{\sigma}) \qquad (4)$$

The boundary conditions are $x_i^0 = x_i$, $x_1^{\sigma} = x_1$, and $x_n^{\sigma} = x_n$. For stability, we must have $0 \le \lambda \le 1/4$. The function c is called the *conductance function*. The conductance values are conceptually interdigitated with the smoothed signal with c_i between x_i and x_{i+1}. It should be computed using $c(i,\sigma) = g(x_{i+1}^{\sigma} - x_i^{\sigma})$. The function g takes the form given by Equation 2 or Equation 2, but with the finite difference given replacing the gradient.

The value κ is referred to as the *diffusion constant* and controls the rate of conduction. If κ is low, then small intensity gradients are able to block conduction and hence inhibit diffusion across step edges. A large value, in contrast, reduces the influence of intensity gradients on conduction. The constant κ can be selected either by hand or using the "noise estimator" proposed by Canny [18]. An overly small value of the constant κ may cause staircases in smoothing and also greatly slows convergence; however, the Canny noise estimator generally avoids this problem. The Canny noise estimator computes a histogram of the absolute values of the gradient and sets κ to the 90% value of its integral at each iteration. However, the Canny noise estimator is relatively slow, especially when the number of iterations is large. Since $|\frac{\partial}{\partial x}X(x,s)| \approx 2\kappa$ is approximately where the value of the conductance function (Equation 2) drops to zero, an alternative is to set the value of κ to

$$\kappa = \frac{|\max(X)-\min(X)|}{2} \qquad (5)$$

The values $\max(X)$ and $\min(X)$ are the maximum value and minimum values of X, respectively. This value is reset after every iteration. If $\kappa = 0$, which means all values in X are equal, then the anisotropic diffusion process (Given by Equation 4) stops.

In Figure 5(a), the curve at the bottom is the original time series. The curves above are the smoothed time series. The higher the level of the curve, the smoother the curve is. This smoothing process gives us a scale-space of smoothed time series.

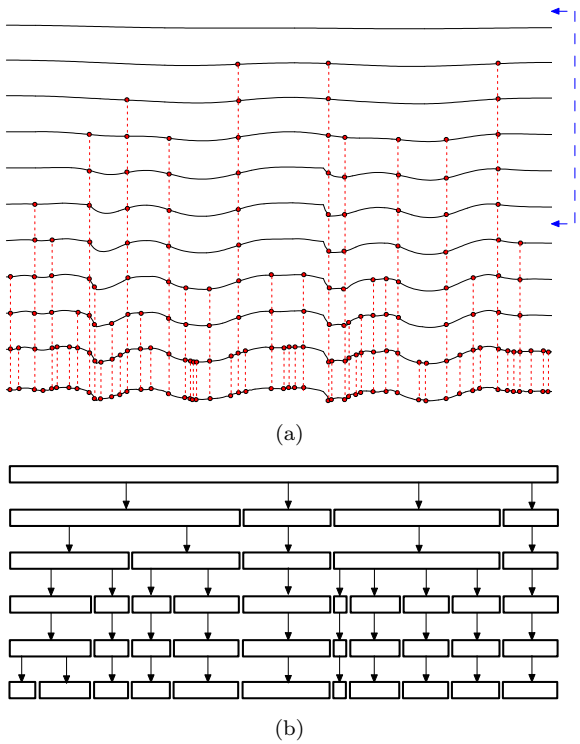


Figure 5: (a) The intrinsic discontinuities are lined up by red dotted lines at the same positions in different scales. (b) The hierarchical structure of the scales is marked by long dashed lines and arrows on the right side of (a).

Anisotropic diffusion smoothes the curve over a scale space. To extract the boundary points of a series at a particular scale, the 1D Canny edge detector [18] can be applied to the smoothed curve. The theory of edge detection was introduced by Marr and Hildreth in their early paper [21]. The Canny edge detector detects boundaries at the zero-crossings of the second derivative of data and the gradient magnitude is also above some threshold $\epsilon_b > 0$, i.e.,

$$\frac{\partial^2}{\partial x^2} X(x,s) = 0 \quad (6)$$

$$\left| \frac{\partial}{\partial x} X(x,s) \right| \geq \epsilon_b \quad (7)$$

The positions of the zero crossings of the second derivative are invariant under anisotropic diffusion and so can be aligned across scales. Coarser scales simply eliminate weaker discontinuities.

In Figure 5(a), all the time series in the scale space are segmented using the Canny edge detector [18]. The boundary points between pairs of segments are lined up at the same positions of different scales, which are are shown Figure 5(a). The segments at different scales give a hierarchical structure, an interval tree, which is shown Figure 5(b).

There is a slight paradox here: smoothing across edges identified by large values of the second derivative is inhibited but edges with zero values of the magnitude of the second derivative are identified as edges by the edge detectors. This paradox can be resolved by realizing that step edges are associated with both a large positive spike and a large negative spike in the second derivative. It should be noted this definition of boundary points also segments the curve into regions of positive and negative acceleration, which is consistent with a categorization of segments into concave and convex regions.

In summary, the anisotropic diffusion process generates a scale-space analysis of a signal and segmentation of this scale space produces a hierarchical representation. Moving from fine to coarse, two or more segments may be merged into a single segment at each iteration because of the erosion of boundary points. As shown in Figure 5(b), this structure can be represented as a tree.

3.2 Feature Representation

Given a time series X, we denote a segment of X by $A = X[i : j]$, where i and j are the starting and ending element indices respectively. Inclusive indices for i and j are used. Then when interpreted in the continuous domain, this is the same as if the segments are split halfway between samples, and consistent with the interpretation of time series as samples of a continuous function. Then we use the parameters of length and shape that are based on a fitting polynomial to characterize it. Such a representation is called a *feature*. The length parameter gives the number of elements in this segment, specifically, $|A| = j - i + 1$.

A polynomial $P(A, t)$ is then used to approximate the shape of each segment A, with t being a real value varying over $[i - 1/2, j + 1/2]$. The linear mapping $t_{i:j} = \tau(j - i + 1) + (i + 1/2)$ reparameterizes the polynomial over $[0, 1]$. We use $P(A, \tau) = P(A, t_{i:j}(\tau))$ to represent this reparameterized polynomial. To derive minimal envelopes, the constant part of this polynomial can be replaced with an interval to bound the original fine-scale

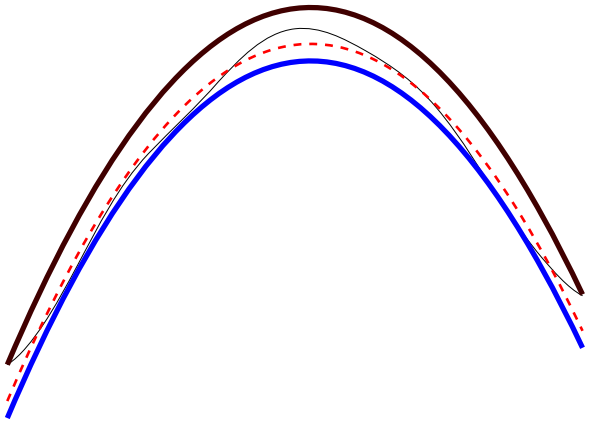


Figure 6: The thin solid curve is the original time series. The dashed curves are a quadratic polynomial that fits the original time series. The thick curves are the minimal quadratic polynomial envelope.

data. A polynomial approximation and the minimal envelope of quadratic order are shown in Figure 6. Every segment in scale space is represented as an nth order polynomial envelope that is mapped to an $(n+1)$-dimensional line segments in an abstract "feature space". Note that features actually bound segments, instead of merely approximating them. The spatial join is then over axis-aligned line segments rather than just points.

Now we will show how distances between polynomial approximations can be computed and distances between functions enclosed with minimal polynomial envelopes can be bounded. Given two polynomials A and B, the distance between them can be defined as

$$\begin{aligned} d^2(A,B) &= \gamma \int_0^1 (P(A,\tau) - P(B,\tau))^2 d\tau \\ &\quad +(1-\gamma)\left(\frac{|A|-|B|}{\max\{|A|,|B|\}}\right)^2 \\ d(A,B) &= \sqrt{d^2(A,B)} \end{aligned} \tag{8}$$

where $P(A,\tau)$ and $P(B,\tau)$ are the rescaled polynomials fitting segments A and B respectively. A rescaled polynomial represents the shape of a segment. and γ is a weight and $0 \le \gamma \le 1$.

This distance can be computed analytically from the coefficients of the polynomials and the lengths of the segments. In fact, the polynomial coefficients and the segment lengths can be placed in a single vector. This vector can be linearly transformed so that the ordinary Euclidean distances on the transformed coefficients can be used [4]. We do not apply the Euclidean distance directly into the original time series data, instead we use the Euclidean distance over our transformed feature representations. We can also compute the min-max bounds of this distance between line segments, which actually represent bounds, not just approximations. Thus, our distance function can deal with time series of different lengths and noise.

Consider the specific case of quadratic polynomials. We assume the transformed vector is $\mathbf{a} = [a_x, a_y, a_z, a_w]$, where a_x, a_y, and a_z are transformed coefficients of $P(A)$ and a_w are the transformed length. Now consider a minimal polynomial envelope where $\mathbf{a}^I = [a_x^I, a_y, a_z, a_w]$ and $a_x^I = [\underline{a}_x, \overline{a}_x] = [a_x - h/2, a_x + h/2]$, where we will call h the *radius* of the interval. This minimal polynomial envelope of a feature can be mapped to 4D axis-aligned line segments [4]. It should be obvious how to extend this analysis to polynomials of any order.

As shown in Figure 7(a), we can compute the maximum distance between two features $\mathbf{a}$ and $\mathbf{b}$ by taking the maximum of the distance between end points of two 4D axis-aligned line segments

$$d_M(\mathbf{a},\mathbf{b}) = \max\{d_E(\mathbf{a}_\ell, \mathbf{b}_u), d_E(\mathbf{a}_u, \mathbf{b}_\ell)\} \tag{9}$$

where

$$\begin{aligned} \mathbf{a}_\ell &= [\underline{a}_x, a_y, a_z, a_w] \\ \mathbf{a}_u &= [\overline{a}_x, a_y, a_z, a_w] \\ \mathbf{b}_\ell &= [\underline{b}_x, b_y, b_z, b_w] \\ \mathbf{b}_u &= [\overline{b}_x, b_y, b_z, b_w] \end{aligned}$$

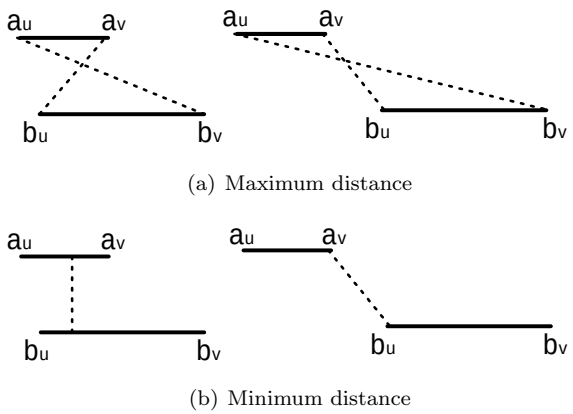
The function d_E^2 is the Euclidean distance, i.e.,

$$\begin{aligned} d_E^2(\mathbf{a}_\ell, \mathbf{b}_u) &= (\underline{a}_x - \overline{b}_x)^2 + (a_y - b_y)^2 + (a_z - b_z)^2 \\ &\quad +(a_w - b_w)^2 \\ d_E(\mathbf{a}_\ell, \mathbf{b}_u) &= \sqrt{d_E^2(\mathbf{a}_\ell, \mathbf{b}_u)} \end{aligned} \tag{10}$$

As shown in Figure 7(b), the minimum distance between two feature points $\mathbf{a}$ and $\mathbf{b}$ can be computed as follows:

$$\begin{aligned} &d_m^2(\mathbf{a},\mathbf{b}) = (a_y - b_y)^2 + (a_z - b_z)^2 + (a_w - b_w)^2 \\ &d_m(\mathbf{a},\mathbf{b}) = \\ &\begin{cases} \sqrt{d_m^2(\mathbf{a},\mathbf{b})}, & \text{if } [\underline{a}_x, \overline{a}_x] \cap [\underline{b}_x, \overline{b}_x] \neq \emptyset \\ \min\{d_E(\mathbf{a}_\ell, \mathbf{b}_u), d_E(\mathbf{a}_u, \mathbf{b}_\ell)\}, & \text{otherwise} \end{cases} \end{aligned} \tag{11}$$

The minimum distance is used in the R-tree join operation as a lower bound to guarantee no false dismissals. The intent here is that the minimal polynomial envelope bounds the actual data and that we can compute a distance function between features that is a lower bound on



(a) Maximum distance

(b) Minimum distance

Figure 7: Compute maximum and minimum distances between two features represented as line segments.

the actual distance between the data. R-tree join operation will generate pairs of candidate matching subseries. Both the maximum distance and the minimum distance are ultimately used in comparing the candidate matching subseries.

3.3 Indexing and Search

To build the index, we insert all features at all scales of the dataset into an R-tree. Every feature is represented as an nth order polynomial that can be mapped to an $(n+1)$-dimensional line segment in an abstract "feature space". Every time series in the dataset is represented as a feature sequence. Both the feature sequences of all time series and the R-tree of the dataset are saved in secondary storage devices. We can select the scale of features in the scale space so that the corresponding sub-R-tree fits into a specific amount of memory space. We also associate the R-tree leaves with the features in the feature sequences instead of saving feature representations redundantly. We use an R-tree join operation [19] to obtain pairs of features whose minimum distance (11) is less than a predefined threshold. Based on the associated leaves of the R-trees, pairs of feature sequences can be found by counting the number of pairs of matching features from each sequence. If this number is greater than a predefined threshold, these two feature sequences are taken as a pair of candidate matching feature sequences.

3.4 Feature Sequence Alignment

We use Smith-Waterman (SW) algorithm [22] to align two candidate matching feature sequences. SW computes local alignments of two discrete sequences. Given two feature sequences $F = (f_1, f_2, \cdots, f_m)$ and $G = (g_1, g_2, \cdots, g_n)$, where m and n are the number of features in each sequence, respectively, SW computes the similarity $H(i,j)$ of two sequences ending at position i and j, where $f_i \in F$ and $g_j \in G$. The computation of $H(i,j)$, for $1 \le i \le m$, $1 \le j \le n$, is given by the following recurrences:

$$\begin{aligned} H(i,j) &= \max\{0, E(i,j), F(i,j), H(i-1,j-1) + \\ &\quad \text{score}(i,j)\} \\ E(i,j) &= \max\{H(i,j-1) - \alpha, E(i,j-1) - \beta\} \\ F(i,j) &= \max\{H(i-1,j) - \alpha, F(i-1,j) - \beta\} \end{aligned} \quad (12)$$

The function score is

$$\text{score}(i,j) = \begin{cases} 1, & \text{if } d_M(f_i, g_j) \le \theta_1 \text{ and } d_m(f_i, g_j) \le \theta_2 \\ 0, & \text{otherwise} \end{cases} \quad (13)$$

where θ_1 and θ_2 are predefined thresholds and $\theta_1 \ge \theta_2$. Initialization of $H(i,j)$ is given by $H(i,1) = \text{score}(i,1)$ and $H(1,j) = \text{score}(1,j)$. The values α and β are the gap penalties. A trace-back procedure starts from the element having the highest score. The alignment path is generated as follows: if the current position is the element (i,j), then the next position is $\max\{H(i,j), H(i-1,j), H(i,j-1)\}$, until it reaches zero. The resulting aligned feature sequences locate the pairs matching subseries.

Although the computational complexity of SW is quadratic, in our prototype system implementation, we constrained the alignment along the diagonal region, which improves the computational complexity to be linear.

Note that a coarse scale contains fewer features than a fine scale. Therefore, using a coarse scale of features for the R-tree join operations will produce fewer candidate matching feature sequences, and then fewer matching subseries than using a fine scale of features. The average length of matching subseries using a coarse scale of features is longer than that using a fine level features. It means that using a coarse level of features produces less precise matching subseries that tolerate more differences than using a fine level of features. Many real-world applications do not require precise value matches, but require general shape matches that can tolerate noise and scaling. Our approach allows for flexible balance between these by selecting different parameters according to specific application requirement. This flexibility is hard to achieve for much of the previous work.

4 Motif Discovery and Anomaly Detection

In this section, we will discuss how to use the results of the subseries join algorithm to solve the problems of motif discovery and anomaly detection. The results of the subseries join of one or multiple time series can be converted a graph to find motifs and anomalies, as shown in Figure 8. Subseries A matches E. Subseries A', E', and G are found to be similar. The same is true for subseries B, D, and F. Subseries C has no matches. The relationships between subseries can be mapped to an undirected graph as shown in Figure 9(a). Every vertex represents a subseries. Every edge between two vertices represents a matching relationship between a pair of subseries.

We then remove self-loops from the graph. There may also exist overlapped subseries, for example, A and B. If the overlap percentage is above a certain threshold (our prototype used 70% overlap) the overlapped subseries, such as A and A', E and E', are merged. After this preprocessing, the example shown in Figure 9(a) turns into Figure 9(b).

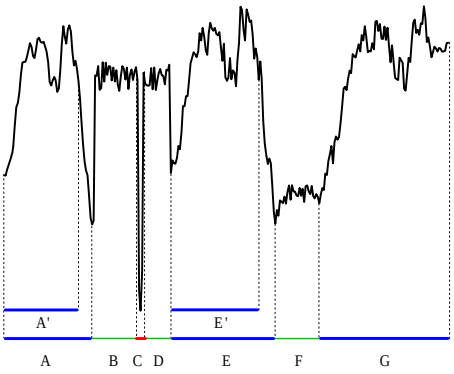
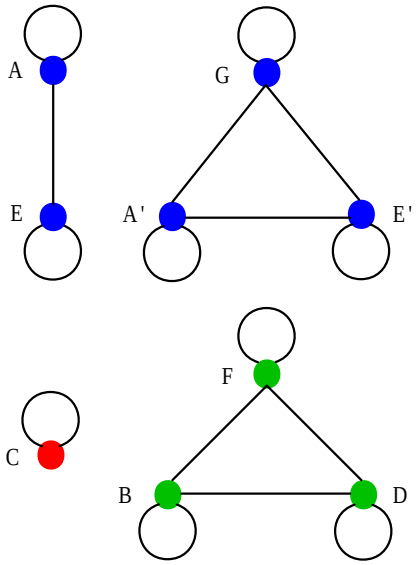
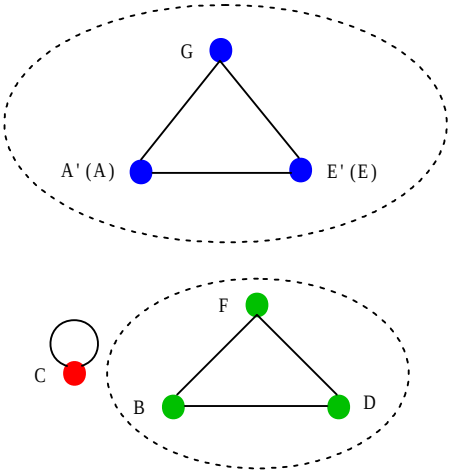


Figure 8: Subseries join results of a time series.



(a) The graph of subseries join results.



(b) Remove self-loops and merge overlapped vertices A and A', as well as E and E'.

Figure 9: Convert the subseries join results into a graph. The maximal cliques that are framed by dashed circles give the motifs. The isolated vertex C is an anomaly.

Now motif discovery can be defined in graph-theoretic terms. Several alternative graph-theoretic definitions of a motif can be considered. One possible definition is based on *maximal cliques*. A *clique* in an undirected graph is a subgraph in which every vertex is connected to every other vertex in the subgraph. A maximal clique is a complete subgraph that is not contained in any other complete subgraph. Unfortunately, the maximal clique problem is an NP-complete problem, and the clique enumeration problem is NP-hard. However, there exist many heuristic algorithms to approximately solve the clique enumeration problem [23, 24], including parallel algorithms [25, 26] and polynomial-time approximation algorithms [27]. Another possible definition of motif is based on the *k-connected subgraph*. A k-connected subgraph in an undirected graph is a subgraph in which every vertex is connected to at least k other vertices in the subgraph. Although the k-connected subgraph problem is also NP-complete, many existing approximation algorithms can efficiently solve this problem [28].

Anomaly detection can be defined as finding the isolated vertices in the graph. For example, in Figure 9, the vertex C is an isolated vertex. Finding isolated vertices is

simpler than finding maximal cliques or k-connected subgraphs in a graph, because it requires only linear computational time.

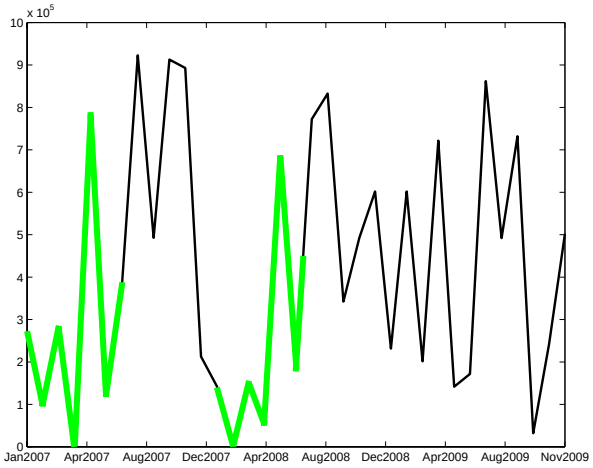
Based on the above discussion, we propose graph-theoretic definitions of *motif* and *anomaly* based on k-connectivity as follows.

Definition 2 ***Motif****: Given a time series $\mathcal{X}$ construct a graph $\mathcal{G}$ from the results of a subseries join with itself. A edge connects every pair of subseries $(X_{i,j}, X_{k,\ell}$ in the join and all subseries in the join are vertices of the graph. We call the set of vertices (subseries) that belong to any k-connected subgraph of $\mathcal{G}$ a motif (or k-motif).*

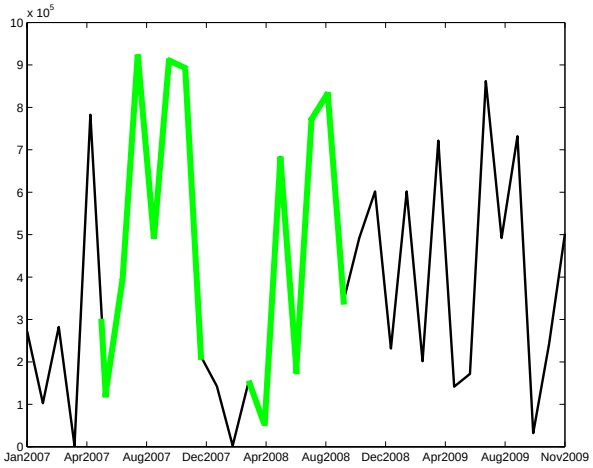
Definition 3 ***Anomaly****: Given a time series $\mathcal{X}$, we call $X_{i,j}$ an anomaly if $X_{i,j}$ does not join with any other subseries of $\mathcal{X}$. In other words, an anomaly is a 1-motif.*

In our experiments, we selected k as 20% of the total number of subseries in the time series $\mathcal{X}$. Since k is usually small, the computational time can be considered linear. We also used the Boost Graph Library [29] in our prototype system implementation.

Note that we assume that an anomaly pattern does not repeat in a time series, because if an anomaly pattern repeats, it may be considered as a motif. In real applications, determining whether a pattern is a motif, anomaly, or something else is often subjective, related to different data, domains, and tasks. However, our approach can help the user to efficiently find candidate targets of interest. The continuum between motifs and anomalies can also be defined as the number of times a motif repeats (its k-connectivity or the size of the clique it belongs to).



(a) One motif is marked by thick green lines.



(b) Another motif is marked by thick blue lines.

Figure 10: Motifs discovered in stock prices: NASDAQ Monthly High, Jan 1, 2007∼Nov 30, 2009.

5 Experimental Results

We evaluate the effectiveness of our proposed approach for discovering motifs and anomaly in real-world data including stock prices [30] and heart rate data [31]. Figure 10 shows motifs discovered by our prototype system in a time series of stock prices. Figure 11 shows an anomaly discovered by our prototype system in a time series of heart rate data. The found motifs often have similar shapes of different lengths. Parts of the found motifs are also overlapped. The properties of the discovered anomalies, such as height and length, can be controlled by parameters in the prototype system. The user can choose different parameters for different data, domains, and tasks.

To evaluate the performance of our approach, the prototype system was also tested on a 3.15GB motion capture dataset [32] containing 3,962,581 frames. A frame is an element in a motion time series. The dataset contains various kinds of motion time series data. Each motion segment consists of time series data ranging in length from about 300 to 23000 frames. All experiments were run on Linux Kernel 2.6 PC with 500MB RAM and Pentium IV 3.0GHz CPU. The prototype generated 101,397 segments of the dataset. The prototype system needs about 3.5 hours for representation and index construction for the whole dataset. We conduct the proposed subseries join operation on the whole dataset to discover motifs, i.e., motion subseries of different styles. The to-

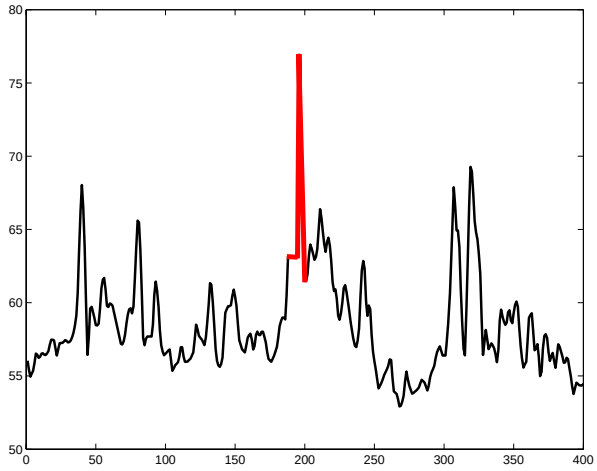
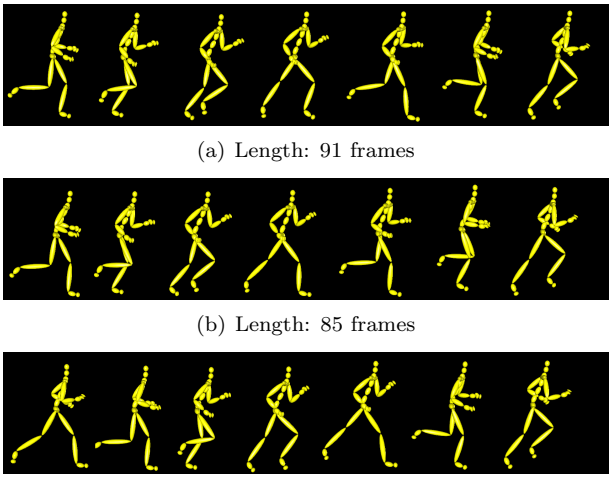


Figure 11: An anomaly marked by thick red lines discovered in heart rate data.

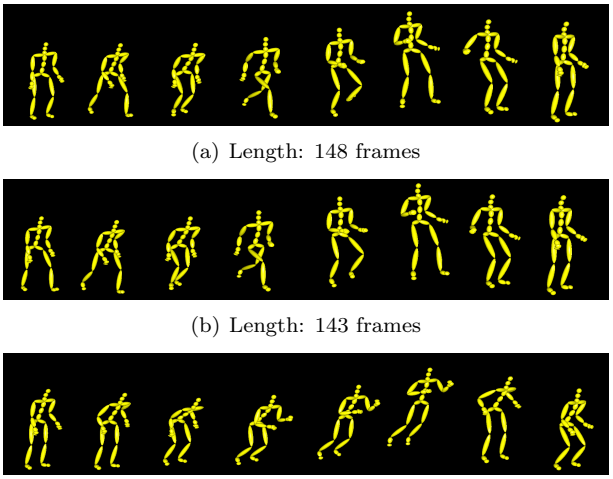


(a) Length: 91 frames

(b) Length: 85 frames

(c) Length: 109 frames

Figure 12: A motif found in the dataset that consists of running motions.

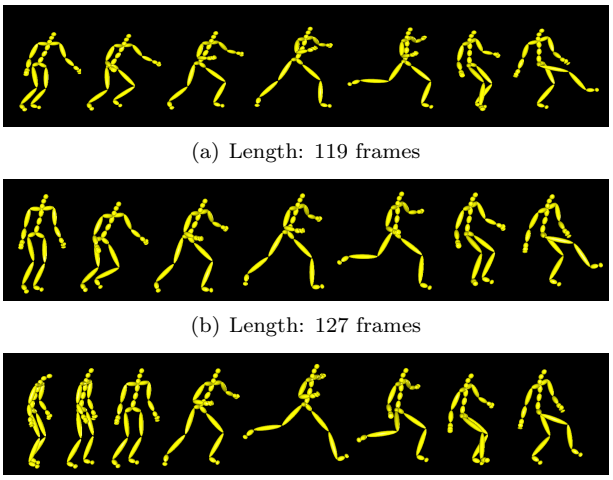


(a) Length: 148 frames

(b) Length: 143 frames

(c) Length: 126 frames

Figure 13: A motif found in the dataset that consists of jumping motions.



(a) Length: 119 frames

(b) Length: 127 frames

(c) Length: 153 frames

Figure 14: A motif found in the dataset that consists of kicking motions.

tal computational time was 35.6 minutes, while the average computational time per time series was 0.86 seconds. Figures 12-14 show some motifs that are similar motions found in the whole dataset. From the figures, we can see that the motif motions have similar but different poses, e.g., the second elements in the running motions. The motif motions also have different lengths, i.e., different speeds of motions. This demonstrates that our approach can tolerate different lengths and scaling.

6 Conclusions

In this paper, we have discussed motif discovery and anomaly detection for time series data. We propose a novel approach for both problems. This approach is based on a new definition of subseries join and an algorithm to compute subseries join efficiently. Experiments have demonstrated the effectiveness of our proposed approach for discovering motifs and anomalies in real-world time series data. Experiments also show that our approach can efficiently find motifs and anomalies in a large dataset.

Our work provides a general solution motif discovery and anomaly detection on any time series data. Therefore, our prototype systems need some user-defined parameters that are tuned for data in a specific domain. At present, the parameters are selected empirically without any domain knowledge involved. In the future, we will apply this work in some specific domains, such as Web mining and network security. Domain knowledge can then be used to automatically tune the system parameters.

References

[1] A. Mueen, E. Keogh, Q. Zhu, S. Cash, and B. Westover, "Exact Discovery of Time Series Motifs," *SIAM International Conference on Data Mining (SDM'09)*, 2009.

[2] L. Wei, E. Keogh, X. Yi, and M. Yoder, "Efficiently Finding Unusual Shapes in Large Image Databases," *Data Mining and Knowledge Discovery*, Vol. 17, No. 3, pp. 343 - 376, 2008.

[3] Y. Lin and M. D. McCool, "Nonuniform Segment-Based Compression of Motion Capture Data," *Proceedings of the 3rd International Symposium on Visual Computing (ISVC'07)*, pp. 56-65, 2007.

[4] Y. Lin, "Subseries Join and Compression of Time Series Data Based on Non-uniform Segmentation," Ph.D. Dissertation, *School of Computer Science, University of Waterloo*, 2008.

[5] Y. Lin, M. D. McCool, and Ali A. Ghorbani "Motif and Anomaly Discovery of Time Series Based on Subseries Join," *Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2010*, IMECS 2010, 17-19 March, 2010, Hong Kong, pp. 481-486.

[6] D. Minnen, T. Starner, I. Essa, and C. Isbell, "Discovering Characteristic Actions from On-body Sensor Data," *The 10th International Symposium on Wearable Computers (ISWC'06)*, pp. 11-18, 2006.

[7] I. Androulakis, J. Wu, J. Vitolo, and C. Roth, "Selecting Maximally Informative Genes to Enable Temporal Expression Profiling Analysis," *Proceedings of Foundations of Systems Biology in Engineering*, 2005.

[8] Y. Tanaka, K. Iwamoto, and K. Uehara, "Discovery of Time-series Motif from Multi-dimensional Data Based on MDL Principle," *Machine Learning*, Vol. 58, No. 2-3, pp. 269-300, 2005.

[9] R. Hamid, S. Maddi, A. Johnson, A. Bobick, I. Essa, and C. Isbell, "Unsupervised Activity Discovery and Characterization from Event-streams," *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI'05)*, Lecture Notes in Computer Science, Springer, Vol. 3669/2005, pp. 119-130, 2005.

[10] B. Chiu, E. Keogh, and S. Lonardi, "Probabilistic Discovery of Time Series Motifs," *Proceedings of the 9th International Conference on Knowledge Discovery and Data mining (KDD'03)*, pp. 493-498, 2003.

[11] D. Yankov, E. Keogh, J. Medina, B. Chiu, and V. Zordan, "Detecting Time Series Motifs under Uniform Scaling," *Proceedings of the 13rd International Conference on Knowledge Discovery and Data Mining (SIGKDD'07)*, pp. 844-853, 2007.

[12] E. Keogh, T. Palpanas, V. Zordan, D. Gunopulos, and M. Cardle, "Indexing large human-motion databases," *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04)*, pp. 780-791, 2004.

[13] Y-S. Moon and K-Y. Whang and W-S. Han, "General Match: A Subsequence Matching Method in Time-series Databases Based on Generalized Windows," *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, pp. 382-393, 2002.

[14] L. Wei, N. Kumar, V. Lolla, E. Keogh, S. Lonardi, and C. A. Ratanamahatana, "Efficiently Finding Unusual Shapes in Large Image Databases," *Proceedings of the 17th International Conference on Scientific and Statistical Database Management*, pp. 337 - 340, 2005.

[15] C-T. Huang, S. Thareja, and Y-J. Shin, "Wavelet-based Real Time Detection of Network Traffic Anomalies," *Proceedings of Workshop on Enterprise Network Security and the 2nd International Conference on Security and Privacy in Communication Networks*, pp. 1-7, 2006.

[16] W. Lu and A. A. Ghorbani, "Network Anomaly Detection Based on Wavelet Analysis," *Journal on Advances in Signal Processing*, Vol. 2009, 2009.

[17] P. Perona and J. Malik, "Scale-space and Edge Detection using Anisotropic Diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 7, pp. 629-639, 1990.

[18] J. Canny "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, pp. 679-698, 1986.

[19] S. Shekar and S. Chawla, "Spatial Databases: a Tour," *Prentice Hall*, 1st Edition, 2003.

[20] C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images", *Proceedings of the 6th International Conference on Computer Vision*, pp. 836-846, 1998.

[21] D. Marr and E. Hildreth, "Theory of Edge Detection," *Proceedings of the Royal Society*, pp. 287-217, 1980.

[22] T. F. Smith and M. S. Waterman, "Identification of Common Molecular Subsequences," *Journal of Molecular Biology*, Vol. 147, pp. 195-197, 1981.

[23] E. A. Akkoyunlu, "The Enumeration of Maximal Cliques of Large Graphs," *SIAM Journal on Computing*, Vol. 2, pp. 1-6, 1973.

[24] J. M. Byskov, "Algorithms for k-colouring and Finding Maximal Independent Sets," *Proceedings of the 14th ACM and SIAM Symposium on Discrete Algorithms*, pp. 456-457, 2003.

[25] E. Dahlhaus and M. Karpinski, "A Fast Parallel Algorithm for Computing All Maximal Cliques in a Graph and the Related Problems," *The 1st Scandinavian Workshop on Algorithm Theory (SWAT'88)*, Springer-Verlag, London, UK, No. 318, pp. 139-144, ISBN:3-540-19487-8, 1988.

[26] N. Du, B. Wu, L. Xu, B. Wang, and X. Pei, "A Parallel Algorithm for Enumerating all Maximal Cliques in Complex Network," *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM'06)*, pp. 320-324, 2006.

[27] H. Ito, K. Iwama, and T. Osumi, "Linear-Time Enumeration of Isolated Cliques," *Proceedings of 13rd Annual European Symposium on ALgorithms (ESA'05)*, pp. 119-130, 2005.

[28] Z. Nutov, "An almost $O(\log k)$-approximation for k-connected subgraphs," *Proceedings of the Nineteenth Annual ACM -SIAM Symposium on Discrete Algorithms*, pp. 912-921, 2009.

[29] J. Siek, L.-Q. Lee, and A. Lumsdaine, The Boost Graph Library (BGL), *Indiana University*, `http://www.boost.org/doc/libs/1_43_0/libs/graph/doc/index.html`.

[30] Economagic.com, Stock Price Indicies, `http://www.economagic.com/sp.htm`.

[31] G. B. Moody and A. L. Goldberger, Heart Rate Time Series, `http://ecg.mit.edu/time-series/index.html`.

[32] Carnegie-Mellon MoCap Database, *Graphics Lab, Carnegie-Mellon University*, `http://mocap.cs.cmu.edu`.