Expert Systems With Applications 55 (2016) 284–296

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Financial time series pattern matching with extended UCR Suite and Support Vector Machine

Xueyuan Gong, Yain-Whar Si*, Simon Fong, Robert P. Biuk-Aghai

Department of Computer and Information Science, University of Macau, China

ARTICLE INFO

Keywords:
Financial time series
Subsequence matching
Perceptually important points
UCR Suite
Support Vector Machine

ABSTRACT

Chart patterns are frequently used by financial analysts for predicting price trends in stock markets. Identifying chart patterns from historical price data can be regarded as a subsequence pattern-matching problem in financial time series data mining. A two-phase method is commonly used for subsequence pattern-matching, which includes segmentation of the time series and similarity calculation between subsequences and the template patterns. In this paper, we propose a novel approach for locating chart patterns in financial time series. In this approach, we extend the subsequence search algorithm UCR Suite with a Support Vector Machine (SVM) to train a classifier for chart pattern-matching. The experimental results show that our approach has achieved significant improvement over other methods in terms of speed and accuracy.

1. Introduction

A time series is a sequence of data points that are typically measured at successive, uniform time intervals. Time series are used in numerous areas such as the representation of the daily closing price of stocks, electrocardiograms, the gait of a walking person, and speech patterns. In general, time series pattern matching can be categorised into whole-sequence matching and subsequence matching (Agrawal, Faloutsos, & Arun, 1993). Whole-sequence matching focuses on the similarities between the time series (sequence) and the pattern, whereas subsequence matching addresses issues of similarity between the pattern and a subsequence within the sequence. Note that a subsequence of a sequence can be treated as a shorter sequence, and likewise, a pattern can also be considered as a kind of sequence that is associated with specific characteristics or meaning. Thus, the similarity calculation between a subsequence and a pattern can be formulated as the similarity calculation between two sequences.

Subsequence pattern matching has been applied to numerous areas, such as the location of technical patterns in financial time series in deciding when to buy or sell stocks, or the search for anomalies in time series in the health care domain for the detection of diseases. In the area of financial time series, patterns are also known as chart patterns and are widely used for technical analysis of the stock market. Specifically, a chart pattern is formed within a chart when prices are graphed. Some of the common chart patterns (Lo, Mamaysky, & Wang, 2000) that are widely used by traders and investors are depicted in Fig. 1.

Due to their high value in financial analysis, these chart patterns are commonly adopted as test patterns in many pattern-matching methods (Fu, Chung, Luk, & Ng, 2007; Zhang et al., 2010). Therefore, subsequence pattern matching in financial time series can be considered as the task of locating the subsequence of a time series with a shape similar to that of a query pattern. An example of a subsequence and the corresponding matched pattern is depicted in Fig. 2.

Many approaches in the financial time series area make use of perceptually important points (PIPs) (Chung, Fu, Luk, & Ng, 2001) to perform segmentation on subsequences as a pre-processing step, followed by pattern matching on the segmented subsequences with template-based (TB) (Fu et al., 2007), rule-based (RB) (Fu et al., 2007), and Hybrid (Chung et al., 2001) approaches. However, segmentation (with PIP or other methods) has three disadvantages:

- The quality of segmentation is important. The important features of the original subsequence should be retained and insignificant noise should be eliminated. If segmentation methods cannot ensure a good outcome, the pattern-matching step that follows can produce a poor result. An example of the selection of the wrong features by the segmentation method in the generation of a subsequence is depicted in Fig. 3. X in Fig. 3(a) represents the segmented subsequence of S by PIP; we can see that X is not at all similar to the pattern P. However, S should

* Corresponding author. Tel.: +853 8822 4454; fax: +853 8822 2426.
E-mail addresses: amoonfana@qq.com (X. Gong), fstasp@umac.mo (Y.-W. Si), ccfong@umac.mo (S. Fong), robertb@umac.mo (R.P. Biuk-Aghai).

http://dx.doi.org/10.1016/j.eswa.2016.02.017
0957-4174/

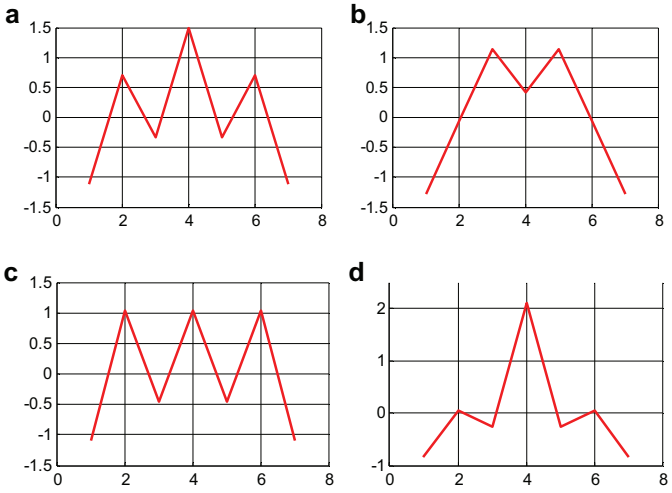

Fig. 1. Common chart patterns (a) Head&Shoulders, (b) DoubleTop, (c) TripleTop, and (d) SpikeTop.

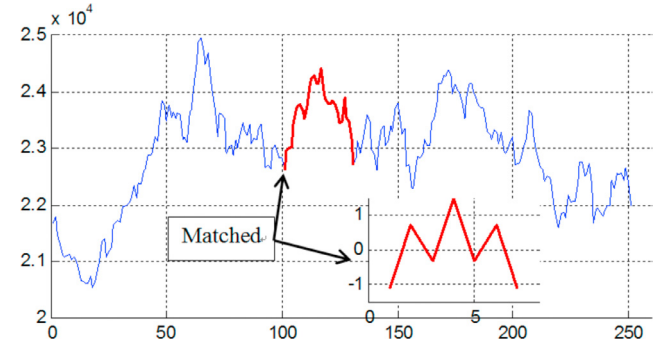

Fig. 2. Subsequence pattern matching in financial time series.

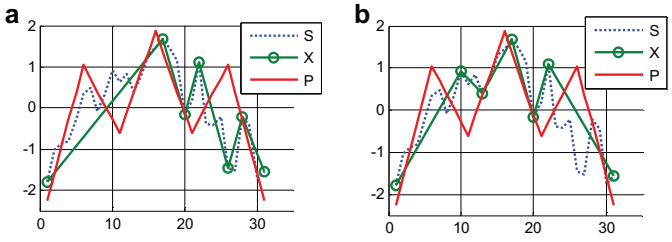

Fig. 3. Example for illustration of the possible effects of segmentation on the outcome of pattern matching, (a) result of PIP and (b) expected result of PIP.

be similar to P, because if S is segmented like X in Fig. 3(b), then S is similar to P. This shows that the performance of segmentation affects the accuracy of subsequence matching.

- Information loss can affect the pattern-matching result. Segmentation means sacrificing accuracy for reduced computation time. However, the use of an insufficient number of points can cause information loss and indivertibly affect the pattern-matching result. For instance, in previous financial time series pattern-matching analyses (Fu et al., 2007; Zhang et al., 2010), the patterns have been represented by only seven points. For a subsequence of length 31 and a pattern of length 7, the compression rate reaches $(31-7)/31 = 0.774$, which means that 77.4% of the original subsequence points are eliminated by segmentation. Therefore, the information retained by the remaining 22.6% of the points is fewer than the original subsequence.
- Time differences must be taken into account after the segmentation. Without segmentation, the subsequence and pattern are point-to-point, namely, the time values (x) of the corresponding price values (y) between the subsequence and the pattern are the same.

All of these discussions illustrate that segmentation as a pre-processing step can cause serious false-positive and false-negative results in the pattern-matching process. In this paper, we first compare the subsequence pattern-matching approaches from two categories, those that require segmentation as a pre-processing step and those that do not. Based on the results of the comparison, we propose a novel approach called the extended UCR Suite (EUCRS) that comprises two components: the UCR Suite (UCRS) and a Support Vector Machine (SVM). The EUCRS is an extended version of the UCRS (Keogh, 2002) developed by Rakthanmanon.

1.1. Definitions and notations

All definitions and notations used in this paper are defined in this section. For the purpose of illustration, we treat each time series as an ordered set. $T = \{(t_1, x_1), (t_2, x_2), \ldots, (t_n, x_n)\}$ represents a time series T that starts at time t_1 and ends at t_n. The length of T is $|T|$, namely, the number of elements n in T, written as $|T| = n$. Each element T_i of T, denoted as (t_i, x_i), is the value x_i at time t_i. Because the time value t_i in financial time series is sequential and often ignored, we can simplify the time series into $T = \{x_1, x_2, \ldots, x_n\}$. Accordingly, t_i can be simply replaced by the value of subscript i, namely $t_1 = 1$, $t_2 = 2$, ..., $t_n = n$. X is the segmentation of T. It usually contains fewer elements than T, and the segmentation is usually performed within the range of acceptable information loss.

A subsequence of T starting at t_i and ending at t_j is written as $T_{i,j} = \{(t_i, x_i), (t_{i+1}, x_{i+1}), \ldots, (t_j, x_j)\}$, or it can be simplified to $T_{i,j} = \{x_i, x_{i+1}, \ldots, x_j\}$, where $1 \le i < j \le n$. Note that like $T_{i,j}$, X_{ij} is also a subsequence of T, but $X_{i,j}$ is discrete on most occasions; we will introduce that in detail in Section 3.2.1. In addition, the pattern P is also a time series.

1.2. Problems of pattern matching

The problem of financial time series pattern matching can be divided into five steps,

1. *Define patterns*: to the best of the authors' knowledge, none of the technical patterns have an exact definition. Line charts are mostly used to represent technical patterns. Therefore, the criteria by which we should define a pattern and store it in a computer are ambiguous. For a TB approach, the pattern is defined as a time series. For an RB approach, the pattern is defined as a set of rules. We will discuss these approaches in more detail in Section 3.1.1.
2. *Subsequence search*: as mentioned in Fu et al. (2007), there is no need to find all subsequences $T_{i,j}$, because analysts and traders only care about a pattern with proper length, e.g., $28 < |T_{i,j}| < 32$. For the traders, monitoring the appearance of the latest $T_{i,j}$ that is similar to pattern P is more important than finding all of the subsequences. The aim of the subsequence search step is to find a $T_{i,n} = \{x_i, x_{i+1}, \ldots, x_n\}$ that is similar to pattern P, where $|T_{i,n}| = m$, and m is a user-specified size. The procedure that details the subsequence search is introduced in Section 3.1.2.
3. *Segmentation*: the volume of the time series data is too large for current methods to calculate; this is known as the "dimensionality curse" (Fu, Chung, Luk, & Ng, 2008). In addition, because a comparison of two time series with different lengths is not convenient, segmentation is commonly used for calculation of

Fig. 4. Overall procedure of subsequence pattern matching.

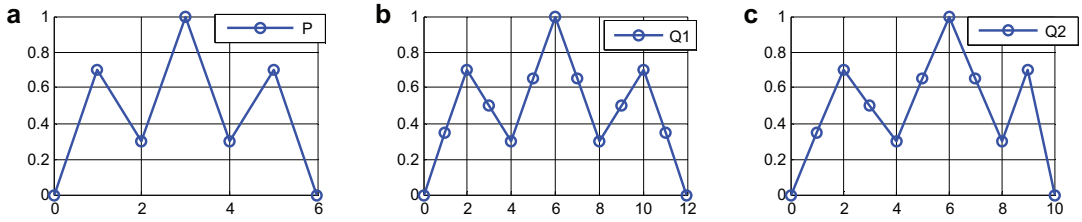

Fig. 5. Interpolation to stretch a pattern while keeping the shape balanced (a) original pattern, (b) the pattern after a new point is added between every two adjacent points, (c) an unbalanced pattern.

an X with a shorter length to represent T. X captures the main feature of T with fewer elements despite some loss of information. As a consequence, X has the same length as pattern P, and different similarity measurement methods can be adopted. This step is introduced in Section 3.2.1.

4. *Normalisation*: before calculating the similarity of T and P, they should be both normalised, as proposed by Keogh and Kasetty (2003). Without normalisation, the calculated similarity is meaningless, because the two are probably shifted or scaled even though their shape is similar. This step is introduced in detail in Section 3.2.2.
5. *Pattern matching*: the method used to measure the similarity of T and P is an essential problem because it affects efficiency and accuracy. Some methods measure similarity on the basis of the original time series, whereas others perform segmentation first to reduce the calculation time and extract features from it. Each type has its own advantages. This step is introduced in Section 3.3.

The remainder of this paper is organised as follows. In Section 2, related work about the five problems mentioned above is reviewed. The overall procedure of subsequence pattern matching and overview of the existing approaches is then introduced in Section 3. Section 4 describes the proposed approach, EUCRS. Finally, the experimental results and conclusions are presented in Sections 5 and 6.

2. Related work

The method used to represent time series is a fundamental problem in the context of time series subsequence pattern matching. A number of approaches have been proposed, including discrete Fourier transform (DFT) (Agrawal et al., 1993), discrete wavelet transform (DWT) (Chan, 1999), piecewise aggregate approximation (PAA) (Keogh & Pazzani, 2000), and adaptive piecewise constant approximation (APCA) (Keogh, 2002). Keogh, Chakrabarti, Pazzani, and Sharad (2001) compared the performance of singular value decomposition (SVD), DFT, DWT and PAA. However, to avoid smoothing or flattening the critical points, which is essential for the analysis and prediction of financial time series, researchers in the area of financial time series proposed their own approaches to represent time series. In financial time series, Fu et al. (2007) and Zhang et al. (2010) choose to leave time series in the time domain rather than changing them to other domains (e.g., the frequency domain for DFT).

Maintaining the same length between the subsequence and the pattern can significantly simplify the similarity calculation. Chung et al. (2001) proposed PIP as a segmentation approach for subsequence pattern matching. Because the number of points is reduced after segmentation, the calculation time is significantly reduced. Fu et al. (2007) and Zhang et al. (2010) adopted PIP as pre-processing step for their subsequence pattern-matching methods. Jiang, Zhang, and Wang (2007) increased the accuracy of PIP with the addition of a merge-and-split step inside PIP. In addition to subsequence pattern matching, PIP is also used for financial time series representation in different resolutions. In Fu et al., (2008a) and Fu, Chung, and Ng (2006), a specialised binary tree approach was developed to represent the critical points after segmentation by PIP. Moreover, there are various segmentation methods besides PIP in other areas in the financial domain. In the comprehensive survey prepared by Keogh, Chu, Hart, and Pazzani (2004), segmentation methods are classified into three categories: sliding window (SW), top-down and bottom-up. APCA (Keogh, Chakrabarti, Mehrotra, & Pazzani, 2001) is proposed to reduce the size of time series. It is more flexible than PAA because the interval between each set of two points can be different, whereas in PAA they cannot.

Before calculation of the similarity between a subsequence and a pattern, normalisation must be performed for both sequences, which was introduced by Keogh and Kasetty (2003). Batista, Wang, and Keogh (2011) described the problems in the calculation of similarity, including the amplitude invariance problem, the uniform scaling invariance problem, and the offset invariance problem, which are the reasons for normalisation. Still, many approaches do not consider normalisation, such as those in Chen, Chen, Chen, and Ooi (2009), Papapetrou, Athitsos, Potamias, Kollios, and Gunopulos (2011), Sakurai, Faloutsos, and Yamamuro (2007) and Zou, Su, Jia, Han, and Yang (2008).

After segmentation is performed, the similarity between the subsequence and the pattern can be calculated. In the area of financial time series, Fu et al. (2007) compared the RB approach with the TB approach; both need PIP as a pre-processing step. Zhang et al. (2010) proposed a hybrid approach that combines Spearman's rank correlation (SC) and an RB approach. The hybrid approach uses PIP as a pre-processing step.

A number of subsequence pattern-matching methods have been proposed in the general time series area. These methods include the Euclidean distance (ED) method and dynamic time warping (DTW). The ED method is sensitive to distortion in the time axis (Keogh, 2002). DTW (Berndt, 1994) has been used extensively in data mining and speech recognition. However, the complexity of DTW is poor compared to that of the ED method. The time complexity of DTW is $O(n^2)$; a number of improvements have been proposed to speed up the calculation. For instance, Keogh (2002) proposed a lower bounding (LB) approach, called LB_Keogh, to

accelerate DTW, which requires the interpolation of the subsequence or pattern so that the lengths of the subsequence and the pattern are equal. Keogh, Wei, Xi, Vlachos, Lee, and Protopapas (2009) introduced early abandoning of LB_Keogh for elimination of unnecessary calculations of DTW, and Rakthanmanon et al. (2012) proposed a suite of methods for acceleration of DTW, called UCRS, that has proven to be faster than the original ED method. In addition to these efforts to improve the speed of DTW, several attempts have also been made to improve its accuracy. Fu, Keogh, Lau, Ratanamahatana, and Wong (2008) combined DTW and uniform scaling to increase the accuracy. Jeong, Jeong, and Omitaomu (2011) proposed a method called weighted DTW for optimisation of the algorithm to increase the accuracy of DTW.

Subsequence pattern matching involves several problem-solving steps, such as subsequence search, segmentation (optional), normalisation, and similarity calculation. Due to the complex nature of the problem, some researchers have proposed methods for improving the time and space complexity. Sakurai et al. (2007) proposed SPRING for subsequence matching based on DTW distance for streaming time series with linear time and space complexity. However, despite its superiority in time and space complexity, normalisation was not considered in Sakurai et al. (2007). Without normalisation, SPRING would cause vast numbers of false-negative results. Ding, Yang, Kavs, and Li (2010) proposed a method for the analysis of the properties of financial time series to construct a new model for similarity measurement that adopts the radian, duration and time of time series to represent it; it is called radian-duration-time-based representation. Pratt (2001) performed a survey regarding the location of patterns and proposed a method to locate patterns by the line segments between every two points, called Legs.

3. Overview of existing approaches

The overall procedure of subsequence pattern matching can be divided into two steps, subsequence search and similarity calculation.

1. *Subsequence search*: read the newest point T_n and extract the subsequence $T_{i,\,n}$ from time series T.
2. *Similarity calculation*: calculate the similarity of $T_{i,\,n}$ and P, denoted SIM($T_{i,\,n}$, P). Before similarity calculation, segmentation and normalisation (pre-processing step) may be required. After pre-processing, the similarity of $T_{i,\,n}$ and pattern P can be measured by various methods and return the result SIM($T_{i,\,n}$, P) = *True* if $T_{i,\,n}$ and P are similar. Otherwise, they return the result SIM($T_{i,\,n}$, P) = *False* if $T_{i,\,n}$ and P are dissimilar.

The overall procedure is depicted in Fig. 4.

As shown in Fig. 4, the program first reads the new subsequence $T_{i,\,n}$, and then the similarity of $T_{i,\,n}$ and the pattern P, and calculates and returns the result (SIM($T_{i,\,n}$, P) = *True* or *False*). The program iterates the subsequence search step and calculates the similarity until all interested subsequences are calculated. The detailed steps for the different similarity calculation methods are listed in Table 1. These methods include TB, RB, Hybrid, DTW, UCRS, and EUCRS.

Because all similarity methods have the same segmentation and normalisation method (PIP and z-Normalisation), and they have already been introduced in Section 3.2, their algorithm is not included in this Section.

3.1. Subsequence search and pre-processing

The first step in subsequence pattern matching is to define the exact features of the patterns to be located in the time series. Next, the subsequence search algorithm determines the manner in which the program extracts subsequence candidates from the time series. A proper subsequence search method should be based on the needs of the user and can affect the type and quantity of subsequences extracted. After subsequences are extracted from the original time series, a pre-processing step (segmentation or normalisation) is required for similarity calculations in the TB, RB, and Hybrid methods.

3.1.1. Defining pattern

To compare the subsequence and the pattern, the exact features of the pattern must be defined in advance. Definition of the pattern is required in all situations except for the RB method. Some of the important points from the pattern are used to capture its prominent features. For instance, seven points are used to define the Head&Shoulders pattern in financial time series. However, a subsequence may not have the same number of points when compared with the pattern. Therefore, several methods (Fu et al., 2007; Zhang et al., 2010) adopt segmentation methods to reduce the number of data points in the subsequence. However, segmentation can cause the loss of information in the subsequence.

To alleviate these issues, an alternative method of defining a pattern is proposed in this paper. In this approach, a pattern is defined with a variable length. For example, a user may wish to find a pattern of length 7 in a subsequence of size 31. In that case, we can stretch the length of the pattern to 31 rather than using segmentation to reduce the length of the subsequence to 7. To maintain the original shape of the pattern, the same number of new points must be added between every two adjacent points. For example, in Fig. 5, for a given pattern $P = \{s_1, s_2, \ldots, s_7\}$, we can create two stretched subsequences, $Q1$ and $Q2$. The shape of $Q1$ is the same as that of P, whereas the shape of $Q2$ is different. In $Q2$, not every two adjacent points are added with the same number of points. Thus, $Q2$ is not a good definition of the pattern.

The time complexity of the proposed stretching method is $O(n)$, and it is faster than PIP, which has a time complexity of $O(n^2)$. However, the proposed approach has a drawback. The length of the resulting segment is restricted by the absolute value of Q which is defined by the following equation.

$$|Q| = n + (n-1) \times \mu \tag{1}$$

where n is the length of P, i.e. $|P| = n$. μ is a variable equal to any positive integer. The pseudo-code for stretching the length of a pattern is given in Algorithm 1.

In this paper, the definition of a pattern is required before the similarity measure for all approaches except for the RB method because the RB method only needs a set of rules rather than a pattern. See Section 3.3.2 for details.

3.1.2. Subsequence search methods

Jumping sliding window subsequence search approach: in Zhang et al. (2010), Zhang adopted an SW with a user-specified window size w as a search method. Given a time series T, where $|T| = n$, the SW starts at t_1 and ends at t_w to extract $T_{1,\,w}$. In fact, the SW extracts $T_{i-w+1,\,i}$ if the SW ends at t_i, where $i \in [w, n]$. There are two possibilities to update the SW,

- If $T_{1,\,w}$ is similar to P, namely SIM($T_{1,\,w}$, P) = *True*, then the SW jumps from $T_{1,\,w}$ to $T_{w,2w-1}$.
- Otherwise, the SW slides from $T_{1,\,w}$ to $T_{2,w+1}$.

These two possibilities are shown in Fig. 6.

In Fig. 6, the black dashed line rectangle represents the previous SW and the red solid line rectangle denotes the current SW. Suppose that $w = 7$, as shown in Fig. 6(a), if $T_{1,\,7}$ in the SW is not similar to P, then the SW moves from $T_{1,\,7}$ to $T_{2,\,8}$. However, in Fig. 6(b), if $T_{1,\,7}$ in the SW matches with P, the SW moves from $T_{1,\,7}$

Table 1
Procedures of different methods for calculating SIM($T_{i,n}$, P).

	SIM($T_{i,n}$, P)		
	Segmentation	Normalisation	Similarity measure
TB	Segment $T_{i,n}$ by PIP, get $X_{i,n}$	Normalise $X_{i,n}$ by z-normalisation, get $NX_{i,n}$	Measure similarity of $NX_{i,n}$ and P by TB
RB	Segment $T_{i,n}$ by PIP, get $X_{i,n}$	Normalise $X_{i,n}$ by z-normalisation, get $NX_{i,n}$	Check whether $NX_{i,n}$ can pass all predefined rules
Hybrid	Segment $T_{i,n}$ by PIP, get $X_{i,n}$	Normalise $X_{i,n}$ by z-normalisation, get $NX_{i,n}$	Calculate SC of $NX_{i,n}$ and P. Then check whether $NX_{i,n}$ can pass all predefined rules.
DTW		Normalise $T_{i,n}$ by z-normalisation, get $NT_{i,n}$	Calculate DTW distance to measure similarity of $T_{i,n}$ and P.
UCRS		Normalise $T_{i,n}$ by z-normalisation, get $NT_{i,n}$	Calculate DTW distance of $T_{i,n}$ and P by UCRS, which is faster than original calculation.
EUCRS		Normalise $T_{i,n}$ by z-normalisation, get $NT_{i,n}$	Calculate DTW path of $T_{i,n}$ and P by UCRS. Then input into SVM to get result.

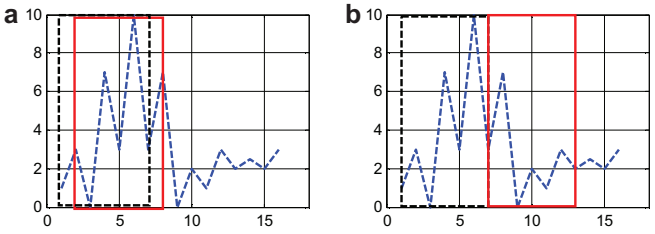

Fig. 6. Procedure of jumping window (a) SW slides from $T_{1,w}$ to $T_{2,w+1}$, (b) SW jumps from $T_{1,w}$ to $T_{w,2w-1}$.

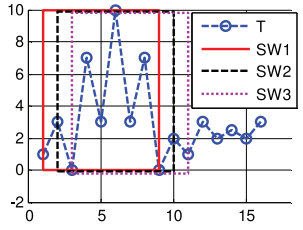

Fig. 7. Three similar adjacent subsequences.

to $T_{7,13}$. However, the approach proposed by Zhang cannot distinguish the best-matched sequence from the adjacent subsequences. In our testing, we find that adjacent subsequences are often similar because there is usually one point difference between every two adjacent subsequences. In addition, amongst these adjacent subsequences, there could be more than one best-matched sequence that is most similar to P. An example is given in Fig. 7 to illustrate this problem.

In Fig. 7, for a time series T, there are three SWs, namely, $SW1 = T_{1,9}$, $SW2 = T_{2,10}$ and $SW3 = T_{3,11}$. Suppose $T_{1,9}$ is similar to the Head&Shoulders pattern. As a result, the SW will jump to $T_{9,17}$ and both $T_{2,10}$ and $T_{3,11}$ will be skipped. However, we can observe that $T_{2,10}$ is the most similar to the Head&Shoulders pattern. Due to the existence of similar subsequences, the brute-force subsequence search approach is used in this paper.

Brute-force subsequence search approach: in this section, we describe the brute-force subsequence search approach for finding an optimal result from adjacent subsequences. In this approach, subsequences that are similar to P are searched exhaustively during the initial step. Next, we choose the optimal subsequence amongst the overlapped subsequences. Overlapped subsequences can be defined as follows.

Overlapped subsequences: for subsequences $T_{i,j}$ and $T_{a,b}$ from a time series T, where $|T| = n$, if $i - b > 0$ or $a - j > 0$, then $T_{i,j}$ and $T_{a,b}$ are not overlapped. Otherwise, they are overlapped subsequences.

Once all of the overlapped subsequences have been identified, the next step is to choose the optimal subsequence from those that overlap. Two options are available for this choice. We may choose the subsequence with the smallest DTW distance to P or we may choose the one with the smallest ED to P. The pseudo-code for the brute-force subsequence search method is presented in Algorithm 2. Note that in Algorithm 2, the SW slides from $T_{i-w+1,i}$ to $T_{i-w+2,i+1}$ regardless of whether it matches a pattern, because in stream time series, only one point arrives at each time.

Note that the above brute-force subsequence search method is used for all of the pattern-matching approaches presented in this paper.

3.2. Pre-processing

3.2.1. Segmentation

There are several studies on the segmentation of time series, such as DFT (Agrawal et al., 1993), DWT (Chan, 1999), PAA (Keogh & Pazzani, 2000), and APCA (Keogh et al., 2001). Segmentation in time series is often known as approximation or dimensionality reduction. The objectives of segmenting time series include a reduction in the number of points for representation quality and computing efficiency (Liu, Lin, & Wang, 2008). Although a number of segmentation methods are available, we focus on an approach called Perceptually Important Point (PIP) introduced by Chung et al. (2001) because PIP is widely used in financial time series (Chung et al., 2001; Fu et al., 2007, 2008a, b; Jiang et al., 2007; Liu et al., 2008; Zaib, Ahmed, & Ali, 2004; Zhang et al., 2010). PIP maintains critical points instead of smoothing them and retains the trends of time series. The PIP algorithm selects a group of critical points called PIPs by measurement and comparison of the distances amongst three points to represent the original time series. There are three criteria for the calculation of PIPs: PIP-ED, PIP-VD and PIP-PD. The PIP-ED equation is based on the ED, PIP-PD is based on the perpendicular distance and PIP-VD is based on the vertical distance (VD). According to Fu et al. (2007), PIP-VD has the best performance in terms of processing time and accuracy. In this paper, we adopt PIP as the segmentation method for the TB, RB and Hybrid methods.

3.2.2. Normalisation

Normalisation is required for the calculation of meaningful similarity between S and P (Keogh & Kasetty, 2003). The reason is that P and S may share a similar shape, but they could be in different resolutions. Therefore, it is essential that S and P are normalised by z-score normalisation (z-normalisation). The formula of

z-normalisation is given in Eq. (1), where $S = \{x_1, x_2, \ldots, x_n\}$ and $N = \{y_1, y_2, \ldots, y_n\}$ is the result of z-normalisation on S.

$$y_i = (x_i - \mu)/\sigma \tag{1a}$$

where

$$\mu = \frac{1}{n}\sum_{1}^{n} x_i \tag{2}$$

$$\sigma = \frac{1}{n}\sum_{1}^{n} x_i^2 - \mu^2 \tag{3}$$

In this paper, z-normalisation is used as the pre-processing step for all approaches because z-normalisation can eliminate the offset invariance problem (Zhou, Wong, & Chu, 2006), and a number of recent studies on time series (Batista et al., 2011; Keogh & Kasetty, 2003) are based on z-normalisation.

3.3. Similarity measure calculation

3.3.1. Template-based (TB)

The TB method uses a straightforward means of calculating the similarity measure. Suppose that there is a query pattern (template), $P = \{(t_1, x_1), (t_2, x_2), \ldots, (t_n, x_n)\}$, and a subsequence $S_{1,m} = \{(a_1, y_1), (a_2, y_2), \ldots, (a_m, y_m)\}$, where $|P| < |S_{1,m}|$.

1. First, $S_{1,m}$ is segmented to $X_{1,m} = \{(a_1, y_1), (a_i, y_i), (a_j, y_j),\}$ $\{\ldots, (a_m, y_m)\}$, where $|P| = |X_{1,m}| = n$. Note that in $X_{1,m}$, x-coordinates $a_1, a_i, a_j, \ldots, a_m$ may not be consecutive numbers, e.g., $a_1 = 1$, $a_i = 2$, $a_j = 5$, ..., $a_m = m$.
2. Next, $AD(P, X_{1,m})$ and $TD(P, X_{1,m})$ are calculated for the similarity of P and $X_{1,m}$.

$$AD(P, X_{1,m}) = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(x_k - y_k)^2} \tag{4}$$

$$TD(P, X_{1,m}) = \sqrt{\frac{1}{n-1}\sum_{k=2}^{n}(t_k - a_k)^2} \tag{5}$$

where AD and TD represent the amplitude distance and time distance. Variables x_k and y_k are the y-coordinates of points in P and $X_{1,m}$, while t_k and a_k are the x-coordinates of points in them.

3. Finally, calculate the similarity of P and $X_{1,m}$, namely $D(P, X_{1,m})$ based on the following equation:

$$D(P, X_{1,m}) = w_1 \times AD(P, X_{1,m}) + (1 - w_1) \times TD(P, X_{1,m}) \tag{6}$$

where w is a user-defined parameter to determine the weight of $AD(P, X_{1,m})$ and $TD(P, X_{1,m})$. In Fu et al. (2007), w is set to 0.5. Variable x_k, y_k, t_k and a_k are normalised in the range [0, 1].

3.3.2. Rule-based

In the RB approach, instead of visual definition of a pattern and direct measurement of the similarity, rules are used to describe the features of the pattern P. Matching patterns based on the RB method is more rigorous than that with the TB method and is less likely to produce false-positive results. The RB method is intuitive and easy to understand and is able to find correct patterns. However, the RB method requires significant effort in the formulation of the appropriate rules because there is no uniform standard in defining the exact features of the technical patterns in the financial domain. Suppose there is a subsequence $S_{1,n} = \{(t_1, y_1), (t_2, y_2), \ldots, (t_n, y_n)\}$. Based on Fu et al. (2007), the rules for the Head&Shoulders pattern are defined in Table 2.where $diff(y_1, y_5)$ represents the difference of y_1 and $y_5 (|y_1 - y_5| < 0.15)$. S can be concluded as a pattern if it satisfies all of the rules.

Table 2
Rules for matching Head&Shoulders pattern.

Rule 1:	$y_3 > y_1$ and y_5
Rule 2:	$y_1 > y_0$ and y_2
Rule 3:	$y_5 > y_4$ and y_6
Rule 4:	$y_2 > y_0$
Rule 5:	$y_4 > y_6$
Rule 6:	$diff(y_1, y_5) < 15\%$
Rule 7:	$diff(y_2, y_4) < 15\%$

Table 3
Rules matching Head&Shoulders pattern.

Rule 1:	$\lvert y_1 - y_5\rvert < 15\%$
Rule 2:	$\lvert y_2 - y_4\rvert < 15\%$
Rule 3:	$XR[3] = 1$
Rule 4:	$XR[1]$ and $XR[5]$ must be 2 or 3
Rule 5:	$XR[0]$ and $XR[6]$ must be 5 or 6 or 7

3.3.3. Hybrid

The Hybrid method (Zhang et al., 2010) adopts two methods, SC and RB, to recognise a pattern. First, SC is adopted for elimination of subsequences that are obviously not similar to the patterns. The RB method is then used as a filter to determine which subsequence is similar to the pattern. The process of similarity measure calculation in the Hybrid method can be described as follows:

Suppose that there is a subsequence $S_{1,m} = \{(t_1, x_1), (t_2, x_2), \ldots, (t_m, x_m)\}$ and pattern $P = \{(t_1, y_1), (t_2, y_2), \ldots, (t_n, y_n)\}$, where $|S_{1,m}| > |P|$.

1. Segment $S_{1,m}$ into $X_{1,m}$, where $|X_{1,m}| = |P| = n$.
2. Compare the y-coordinates of every point in $X_{1,m}$ and P to get the rank (position of point in descending order) of each point, and store the rank into arrays XR and PR. For example, if $X_{1,5} = \{(1, 12), (2, 37), (4, 53), (6, 41), (7, 25)\}$, then $XR = [5, 3, 1, 2, 4]$. Likewise, PR stores the rank of each point P; we omit the example of PR here.
3. Calculate SC between XR and PR. The equation for calculating Spearman's rank correlation is given in Eq. (7).

$$SC = 1 - \frac{\alpha \sum_{i=1}^{n}(XR[i] - PR[i])^2}{n\left(n^2 - 1\right)} \tag{7}$$

where n is the size of XR and PR.α is the scalar value.

4. Compare SC with a user-defined threshold ε. The threshold is defined to filter less similar subsequences before they are checked with the RB method. If $SC > \varepsilon$, then go to the next step. Otherwise, conclude that $S_{1,m}$ is not similar to P.
5. Check whether $X_{1,m}$ satisfies all defined rules. If yes, $S_{1,m}$ is similar to P. Otherwise, $S_{1,m}$ is not similar to P. The rules for checking the Head&Shoulders pattern in Zhang et al. (2010) are given in Table 3.

3.3.4. Dynamic time warping

Unlike the TB, RB and Hybrid methods, DTW does not depend on the outcome of segmentation. In this section, we describe a DTW method that does not use segmentation as a pre-processing step to measure the similarity between the time series. The DTW method can be outlined as follows.

Suppose there is a pattern $P = \{(t_1, x_1), (t_2, x_2), \ldots, (t_n, x_n)\}$ and a subsequence $S_{1,m} = \{(t_1, y_1), (t_2, y_2), \ldots, (t_m, y_m)\}$,

1. Calculate a distance matrix that contains the distance of every point between P and $S_{1,m}$. Each entry of the matrix is denoted as $D(i, j) = (x_i - y_j)^2$, where $i \in [1, n]$, $j \in [1, m]$.

2. The DTW matrix is then calculated on the basis of the distance matrix.

$$\gamma(i,\ j) = D(i,j) + min\{\gamma(i-1,j), \gamma(i,j-1), \gamma(i-1,j-1)\} \tag{8}$$

where $i \in [1, n]$, $j \in [1, m]$. The $\gamma(i,j)$ is the entry of the DTW matrix. If $i \notin [1, n]$, or $j \notin [1, m]$, $\gamma(i,j)$ is equal to positive infinity.
3. Finally, the last entry of the DTW matrix is the result, namely $\gamma(n,m)$.

In the DTW matrix, we can find a path, denoted by $Path_k = \gamma(i,\ j)$, to obtain the final result $\gamma(n,m)$. This path is called the DTW warping path, which satisfies three important criteria:

1. *Boundary*: The start point and the end point of the warping path must be $\gamma(1,\ 1)$ and $\gamma(n,m)$, $Path_{start} = \gamma(1,\ 1)$ and $Path_{end} = \gamma(n,\ m)$.
2. *Monotonicity*: For the warping path $Path_k = \gamma(i,\ j)$, i and j are increasing monotonously from 0 to n and m. For example, if $Path_5 = \gamma(5,\ 6)$, then $Path_6$ could be $\gamma(5,7)$, $\gamma(6,6)$, or $\gamma(6,7)$, but $\gamma(4,6)$, $\gamma(5,5)$, or $\gamma(4,5)$ are impossible.
3. *Step size*: For the warping path, if $Path_k = \gamma(i,\ j)$, $Path_{k+1}$ could only be equal to $\gamma(i+1,\ j)$, $\gamma(i,\ j+1)$, or $\gamma(i+1,\ j+1)$. For instance, if $Path_5 = \gamma(5,\ 6)$, then $Path_6$ could be $\gamma(5,7)$, $\gamma(6,6)$, or $\gamma(6,7)$, but $\gamma(7,6)$, $\gamma(5,8)$, or $\gamma(7,8)$ are impossible.

DTW can be used to calculate the similarity measure of two time series with different lengths. In other words, the size (length) of the subsequence and the pattern do not need to be the same. However, DTW has its own disadvantages. First, the time complexity of DTW is $O(nm)$, which is rather slow for subsequence pattern matching. Second, DTW treats all points of the time series as being equally important and does not address the shape differences of the time series. To alleviate this issue, Jeong et al., 2011 proposed a Weighted DTW to take into account the phase difference of the points.

4. Extended UCR Suite

The UCRS (Rakthanmanon et al., 2012) was proposed for acceleration of the DTW algorithm. The UCRS combines several optimisation methods to improve the complexity of DTW. In this paper, we propose a pattern-matching method called the EUCRS. The proposed method is designed to improve the quality of the UCRS, whereas the original UCRS was proposed to improve the speed of DTW. In the EUCRS, an SVM is used for pattern matching. We chose an SVM rather than other machine learning approaches because it is one of the most widely used methods in decision support systems and financial modelling. For instance, SVM is proven to outperform other methods in application areas such as credit scoring (Huang, Chen, & Wang, 2007), bankruptcy prediction (Shin, Lee, & Kim, 2005), drug/nondrug classification (Byvatov, Fechner, Sadowski, & Schneider, 2003), credit rating analysis (Huang, Chen, Hsu, Chen, & Wu, 2004) and protein fold recognition (Ding & Dubchak, 2001). Compared to other approaches, SVM has more interpretable results and achieves higher accuracy.

4.1. Data distribution

The pattern matching of subsequences based on an SVM can be defined as a two-class classification problem. The data distribution is essential for this kind of problem because it determines the type of classifier for the classification. For instance, if the data are linearly separable, a simple perception-based network can be sufficient. However, if the data are linearly inseparable, a more complicated classifier should be used. In general, the distribution of data for subsequence pattern matching of financial time series can be visualised as a 'circle'. Suppose there is a subsequence S and a pattern P, the easiest way to calculate the similarity between S and P is the ED.

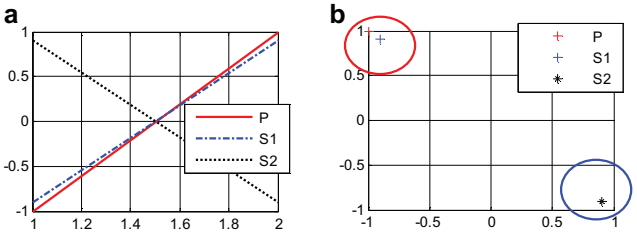

Fig. 8. Illustration that data distribution is similar to 'circle' (a) when ED is used to classify two subsequences $S1$ and $S2$ (b) mapping the original time series from the original space to the SVM space.

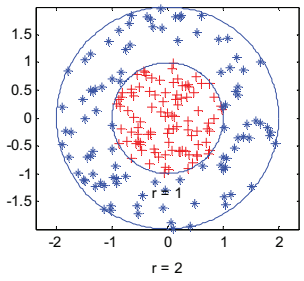

Fig. 9. Data distribution of pattern and subsequence.

$$ED = \sqrt{\sum_{i=0}^{n} (p_i - y_i)^2} \le d \tag{9}$$

Eq. (9) can be rewritten as follows:

$$ED^2 = \sum_{i=0}^{n} (p_i - y_i)^2 \le d^2 \tag{10}$$

For a two-dimensional case, the equation becomes $(p_1 - y_1)^2 + (p_2 - y_2)^2 \le d^2$, which represents a circle with the centre (y_1, y_2). Suppose there is a pattern $P = \{x_0, x_1,\ \ldots, x_n\}$ and a subsequence $S = \{x_i, x_{i+1},\ \ldots, x_j\}$, where $|P| = |S|$. We can say that P and S are similar only when their shapes are similar. Specifically, P and S are considered to be similar only when the ED is approximately equal to zero, $ED(S,P) \approx 0$. For instance, for a two-dimensional case with $P = \{x_0, x_1\}$ and $S = \{x_i, x_{i+1}\}$, if $P = \{-1,\ 1\}$, $S1 = \{-0.9,\ 0.9\}$, $S2 = \{0.9,\ -0.9\}$, $ED(S1,\ P) \approx 0.14$, and $ED(S2, P) = 2.69$, we can conclude that $S2$ is not similar to P. For illustration, we show P, $S1$, and $S2$ in Fig. 8(a).

However, if we map the original time series from the original space to the SVM space (mapping the first element as the x coordinate and the second element as the y coordinate), the data distribution of P, $S1$, and $S2$ can be represented as points in Fig. 8(b). We can say that S and P are more similar when they are closer to each other. As a result, a circle is formed just like the circle shown in Fig. 9, in which red points represent positive cases and blue points represent negative cases. For the time series pattern-matching problems with three, four or more dimensions, the distribution can be represented in a similar manner.

4.2. Training SVM

In the proposed EUCRS, the DTW warping path is used as the training data set for SVM instead of the original time series points. There are two reasons for this. First, the use of the DTW warping path handles the time distortion problem, which causes

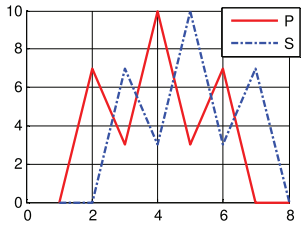

Fig. 10. Example to show time distortion that DTW takes into account.

Table 4
An example of calculated warping path.

n	$\gamma(n,\ 1)$	$\gamma(n,\ 2)$	–	$\gamma(n,\ m-1)$	$\gamma(n,\ m)$
$n-1$	$\gamma(n-1,\ 1)$	$\gamma(n-1,\ 2)$	–	$\gamma(n-1,\ m-1)$	$\gamma(n-1,\ m)$
–	–	–	–	–	–
2	$\gamma(2,\ 1)$	$\gamma(2,\ 2)$	–	$\gamma(2,\ m-1)$	$\gamma(2,\ m)$
1	$\gamma(1,\ 1)$	$\gamma(1,\ 2)$	–	$\gamma(1,\ m-1)$	$\gamma(1,\ m)$
P/S	1	2	–	$m-1$	m

false-negative results (false dismissals). In concrete terms, time distortion is the position shift of points between the subsequence and the pattern. An example of patterns with time distortion is depicted in Fig. 10. Second, the use of the DTW warping path supports the similarity measure between two time series with different sizes. In addition, the UCRS is used as the algorithm to calculate the DTW warping path because the original DTW calculation algorithm is too slow.

In Fig. 10, the ED between the pattern $P = \{0, 7, 3, 10, 3, 7, 0, 0\}$ and the subsequence $S = \{0, 0, 7, 3, 10, 3, 7, 0\}$ is 15. Although their shapes are similar, the ED of these two patterns is large. This kind of problem is mostly referred to as the time-distortion problem or the local time-scaling problem. In this case, the UCRS and SVM can be used to solve this problem. The steps of the training SVM are shown below. Suppose that there is a pattern P and a list of subsequences:

1. Normalise all subsequences and P by the z-Normalise method.
2. Calculate the DTW warping path between subsequence S and P and store the distance, $D(i,\ j) = (x_i - y_j)^2$, along the path into T as training data. The pseudo-code of calculating T for a subsequence S is given in Algorithm 3.

   Note that despite the random size of the warping path, the SVM requires the inputs to be the same size. Therefore, the size of T is modified to be same as that of pattern P. In detail, there are three possible path movements, from $\gamma(i,\ j)$ to $\gamma(i,\ j-1)$, $\gamma(i-1,\ j)$ and $\gamma(i-1,\ j-1)$ respectively. Suppose the warping path $T = [T_1, T_2,\ \ldots,\ T_n]$. When the path moves from $\gamma(i,\ j)$ to $\gamma(i-1,\ j)$ or $\gamma(i-1,\ j-1)$, then T adds a new value $D(i-1,\ j)$ or $D(i-1,\ j-1)$ accordingly. When the path moves from $\gamma(i,\ j)$ to $\gamma(i,\ j-1)$, the last value of T is revised to $T_n + D(i,\ j-1)$. For example, suppose the DTW matrix γ and the warping path as shown in Table 4, and then $T = [D(n,\ m),\ D(n-1,\ m-1),\ \ldots,\ D(2,\ 2),\ D(1,\ 1) + D(1,\ 2)]$.

3. Train SVM for classification.

4.3. Classification

After the SVM is trained, it can be used for classification. Suppose that we are given an unseen subsequence S. To perform the classification, we can use Algorithm 3 to calculate the warping path of subsequence S. We then input the path into the SVM to check whether S is a pattern. LIBSVM (Chang, 2011) is used in the proposed algorithm, and the C-support vector classification (C-SVC) in LIBSVM is adopted as the classification method. C-SVC can classify a subsequence S in two classes, namely, a pattern (1) or not a pattern (−1). Suppose there is a vector $x_i \in S$ and a vector $y_i \in \{-1,\ 1\}$, and then the C-SVC for solving the prime optimisation problem can be described as follows:

$$\min_{\omega,b,\varepsilon} \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C\sum_{i=1}^{l}\varepsilon_i$$
$$\text{subject to} y_i(\boldsymbol{\omega}\hat{}T\emptyset(x\_i) + b) \geq 1 - \varepsilon_i \quad (11)$$

where $\phi(x_i)$ represents the kernel function that maps x_i into higher-dimensional space and $\varepsilon_i \geq 0, i = 1, \ldots, l$. C is a regularisation parameter that subjects to $C > 0$. $\boldsymbol{\omega}$ is the vector variable that will be determined after training, and $\boldsymbol{\omega}^T$ is the transposed matrix of $\boldsymbol{\omega}$. The kernel function used is a Gaussian radial basis function, and gamma is set to $1/n$ and n is the length of P. Interested readers can refer to Cortes (1995) for more information about SVM.

5. Experimental results

All experiments in this paper were carried out on a 2.5 GHz personal computer with 4 GB RAM. The compiler used was NetBeans IDE Ver 7.2, and the language used was JAVA. The operating system was Microsoft Windows 7.

5.1. Experiment on synthetic data set

5.1.1. Creating data set

Based on the pattern of Fig. 1, we generated 800 synthetic subsequences as a data set to evaluate the performance of each pattern-matching method. Amongst these synthetic subsequences, four technical patterns were tested: Head&Shoulders, DoubleTop, TripleTop and SpikeTop. The synthetic data generation algorithm is extended from Fu et al. (2007) and Zhang et al. (2010). The algorithm comprises three parts: Time Scaling, Time Warping and Noise Adding. The purpose of Time Scaling is to increase the length of the pattern so that subsequences of various lengths can be generated. Time Warping changes the position of important points so the subsequences appear 'warped' when compared to the pattern. Noise Adding alters the value of each point and generates a subsequence with additional small fluctuations. The pseudo-code of the algorithm for generation of a synthetic data set is given in Algorithm 4.

5.1.2. Accuracy

The accuracies of the seven pattern-matching approaches when applied to the synthetic data are shown in Fig. 11. Note that the accuracies of the UCRS and DTW are the same because UCRS is the optimized version of DTW search in terms of time complexity (Rakthanmanon et al., 2012). Therefore, the accuracy of the UCRS is not shown in Fig. 11.

In Fig. 11, the methods with the best accuracy are DTW and EUCRS. The ED method does not consider the time distortion and is therefore less accurate than DTW. The experimental results also show that the accuracies of the RB and Hybrid methods are the lowest amongst all of the methods. The poor accuracy of the RB and Hybrid approaches could be caused by the quality of the rules defined for the patterns. In concrete terms, the rules may be well defined for some patterns (e.g., Head&Shoulders) but not for others (e.g., DoubleTop). The quality of the RB and Hybrid methods is highly dependent on the quality of the rules, but it is not feasible for the rules for different patterns to be defined equally well. Thus, the results of the RB and Hybrid methods fluctuate greatly.

5.1.3. Analysis of specific cases

To investigate the performance of each pattern-matching approach, we select some specific cases from the experiment for further discussion. Fig. 12 depicts a subsequence S and two patterns,

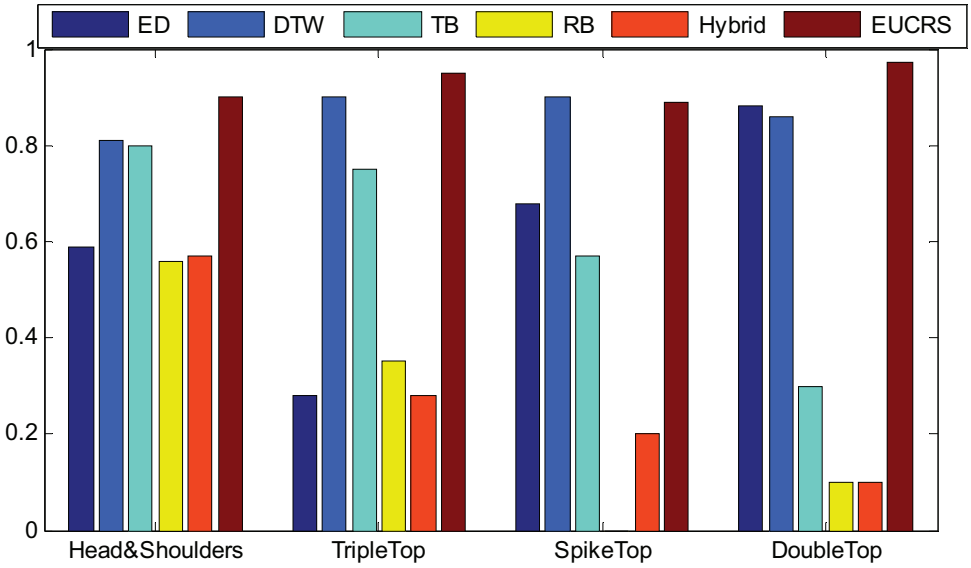

Fig. 11. Accuracy of pattern-matching methods for different patterns.

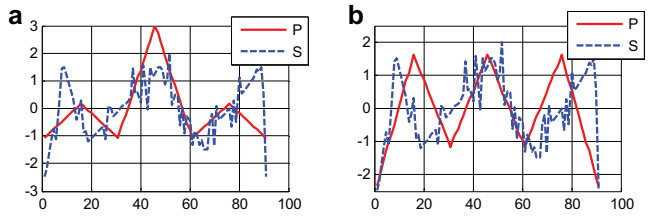

Fig. 12. (a) A synthetic subsequence recognised by ED method as a SpikeTop pattern (b) A TripleTop pattern overlaid onto the same synthetic subsequence.

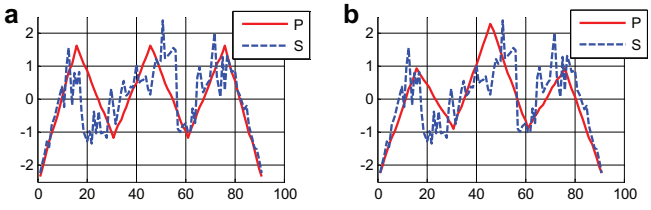

Fig. 13. (a) A TripleTop pattern overlaid onto a synthetic subsequence (b) The same subsequence misclassified by the ED method as a Head&Shoulders pattern.

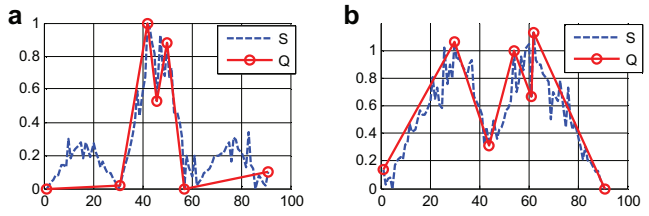

Fig. 14. Examples in which PIP chooses incorrect important points (a) missing two 'spikes' after PIP process in *Q*, (b) distorted *Q* produced by PIP for a DoubleTop pattern.

SpikeTop in Fig. 12(a) and TripleTop in Fig. 12(b), overlaid onto *S*. In our experiment, the ED method recognises *S* as a SpikeTop pattern even though *S* is synthetically generated from a TripleTop pattern. This false-positive result occurs because the ED method does not take time distortion into account. In this case, an 'out of phase' problem causes the misclassification.

In addition, the performance of the ED method is unstable and it performs poorly on TripleTop patterns when compared with the other three because TripleTop patterns can easily be misclassified as Head&Shoulders patterns. Fig. 13 depicts a subsequence *S* and two patterns, TripleTop in Fig. 13(a) and Head&Shoulders in Fig. 13(b), overlaid onto *S*. In our experiment, the ED method recognises *S* as a Head&Shoulders pattern.

From the experimental results, we also find that the performance of DTW is better than most of the methods that make use of segmentation as a pre-processing step because DTW relies on the similarity between the original subsequence and the query pattern. Therefore, there is no information loss. However, the time complexity is greater than that of the other methods. In addition, DTW takes into account time distortion, and as a result it is found to be superior to the ED method.

Although the TB method takes time distortion into account like DTW, there is no obvious performance gain compared to the ED method because the accuracy of the TB method is affected by the performance of PIP, whereas the ED method is affected by a failure to consider time distortion.

Fig. 14(a) illustrates a situation in which the main feature of an original subsequence is missed by PIP. *S* is the original subsequence, and *Q* is the segmented subsequence. In Fig. 14(a), two 'spikes' are missing for *Q*, and as a result the TB method or any other pattern-matching method cannot recognise it as a SpikeTop pattern. Additionally, in Fig. 11, the performance on DoubleTop patterns is worse than those of the other three patterns for the TB method because the performance of PIP is worse than in the other three. The DoubleTop pattern can usually be represented by five important points. However, for the sake of consistency and simplicity, we use seven points to represent the DoubleTop pattern. The segmented subsequence based on seven points is depicted in Fig. 14(b). On most occasions, PIP segmentation produces a subsequence similar to *Q* in Fig. 14(b) rather than the subsequence with two tops.

The RB and Hybrid methods are affected by the performance of PIP. However, finding a set of appropriate rules to represent a pattern is difficult because there is no reference model or industry standard for defining what constitutes a technical pattern. In general, the relative position (coordinates) and the number of important points is the key features used to define the shape of a pattern. In other words, if rules are to be used to represent a pattern,

Algorithm 1
Pseudo-code of stretching the length of a pattern.

```
Main function: time scaling
Input: Pattern P and parameter μ
Output: Stretched pattern Q
FOR i = 1 to (n − 1)
    FOR EACH p_i and p_{i+1}   //For every two adjacent points
      FOR j = 1 to μ   //Insert μ points
        The equation of calculating each point is: p_{i−1} + ((p_i − p_{i−1}) * j)/μ
      END FOR
    END FOR EACH
END FOR
```

Algorithm 2
Pseudo-code of brute-force subsequence search.

```
Main function: Brute-force subsequence search
Input: time series T, pattern P and window size w
Output: Matched pattern
Initialise a temporary subsequence TS = null
FOR i = w to n//where n = |T|
    Apply SW to T_{i−w+1,i}
    Calculate SIM(T_{i−w+1,i}, P) //Details in Section 5
//The following part is for choosing optimal one from overlapped subsequences
    IF T_{i−w+1,i} is similar to pattern P
      IF TS == null
        TS = T_{i−w+1,i}
      ELSE
        IF T_{i−w+1,i} is overlapped with TS
IF T_{i−w+1,i} is more similar to P compared with TS
          TS = T_{i−w+1,i}
END IF
      ELSE
        Return that TS is a pattern
      END IF
    END IF
  END IF
END FOR
Return that TS is a pattern
```

Algorithm 3
Pseudo-code of calculating the warping path.

```
Main function: calculating the warping path
Input: DTW matrix γ calculated by subsequence S and pattern P
Output: T
Initialisation: i = n, j = m, and k = 0
T_k = D(n, m)
WHILE i > 0 AND j > 0
    min{γ(i − 1, j), γ(i, j − 1), γ(i − 1, j − 1)}        //Get the minimum one
    IF γ(i − 1, j) is minimum
      k++
      T_k = D(i − 1, j)
      i−−
    END IF
    IF γ(i, j − 1) is minimum
      T_k = T_k + D(i, j − 1)
      j−−
    END IF
    IF γ(i − 1, j − 1) is minimum
      k++
      T_k = D(i − 1, j − 1)
      i−−, j−−
    END IF
END WHILE
```

the features of the pattern must be well understood. However, expert knowledge is not always available, and patterns are often represented in various scales. From our experiment, we find that the accuracy of the RB method for the SpikeTop pattern in Fig. 11 is zero because all of the subsequences that should be recognised as a SpikeTop pattern are wrongly segmented by PIP into the shape of Q in Fig. 14(a) and (b). In addition, the rules defined in the RB method are more strict (i.e., all are crisp rules), and therefore not a single subsequence is recognised as a SpikeTop pattern.

The EUCRS and DTW methods achieve similar results, and the accuracy of EUCRS is slightly better than that of DTW. We further elaborate on the advantages of EUCRS in the experiment for the real data set.

5.2. Experiment on real data set

In this experiment, the real data set was used to evaluate seven pattern-matching methods: TB, RB, Hybrid, ED, DTW, UCRS and EUCRS. These pattern-matching methods were tested on four patterns: Head&Shoulders, TripleTop, DoubleTop and SpikeTop. The training data used for EUCRS is the synthetic data generated by the algorithm in Section 5.1.1. The training data contains 800 subsequences in total with 400 positive cases and 400 negative cases. The number of patterns found and the execution times of the different pattern-matching methods are listed in Table 5. We do not compare the accuracy because there are no benchmark data (collections of patterns) in the real data. Note that the UCRS and DTW identify the same number of patterns because the UCRS is the 'sped up' version of DTW.

All methods are tested with data from the Heng-Seng Index (HSI) (YahooFinance, Last Accessed on 11 Nov. 2014) from 1 January 2003 to 31 December 2013. The window size is set to 31, 61 and 91 respectively. Table 5 shows the total number of patterns found by different approaches and the execution time of each method. Note that when a pattern-matching method identifies several adjacent subsequences as a pattern in our experiment, we treat them as one pattern.

5.2.1. Analysis of TB, RB, Hybrid methods

In this section, we analyse the segmentation effect of PIP on the real data set for the TB, RB and Hybrid methods. The time dimension is a crucial factor, and time-distorted subsequences generated by the segmentation process are often recognised as patterns in our experiment. To illustrate this problem, one of the false-positive TripleTop patterns found by the RB and Hybrid methods is depicted in Fig. 15.

In Fig. 15(a), we can see that segmented X is not similar to any pattern. However, if we zoom in on the right part of subsequence S (see Fig. 15(b)), it is similar to a TripleTop pattern. We find that the TB method takes the time dimension into account, and therefore X is not recognised as a TripleTop pattern. However, the RB and Hybrid methods are based on rules and neither method considers the time dimension; therefore, X is identified as a TripleTop pattern. Interestingly, this kind of 'time distortion' characteristic is similar to that seen in DTW, as DTW was proposed for tolerating time distortion.

In addition, all three of these approaches (TB, RB and Hybrid) depend on the performance of segmentation (PIP method). Thus, their performance is highly dependent on the performance of PIP. Moreover, because the RB and Hybrid methods are based on rules, their accuracy for different patterns fluctuates, which we also introduced in Section 5.1.2. Through our analysis, we find that segmentation affects the results of the pattern-matching methods (TB, RB and Hybrid) at different levels. We also find that methods that do not use segmentation (UCRS, DTW, ED and EUCRS) appear to have better accuracy and are simpler in calculation. For instance, in Fig. 16, the TB method recognises the segmented result X as a DoubleTop pattern. However, X is not similar to pattern P, and therefore it is a misclassified case of the TB method.

The performance of PIP is poor for DoubleTop patterns. One reason for this poor performance is that a DoubleTop pattern can be represented by five points, but in Fu et al. (2007) and Zhang et al. (2010) it is defined by seven points for the purposes of simplicity and consistency. The DoubleTop pattern in our experiment is

Table 5
Number of patterns found and execution time of different pattern-matching methods.

Window size		TB	RB	Hybrid	ED	DTW	UCRS	EUCRS
31	Patterns found	40	8	11	47	22	22	31
	Wall Clock time(ms)	530	531	575	468	733	624	280
61	Patterns found	19	9	9	21	15	15	17
	Wall Clock time(ms)	562	583	612	523	1357	635	343
91	Patterns found	11	5	7	12	13	13	10
	Wall Clock time(ms)	593	655	683	592	2184	749	387

Algorithm 4
Pseudo-code of algorithm for generation of synthetic data.

```
Time scaling
Input a pattern P (length of P is n) and a number m (m = 49, 91 and 133 in this paper)
FOR each two adjacent P_i and P_{i+1}
    X = (m − n)/(n − 1)
Insert X points
END FOR
Time Warping
Input a pattern P
FOR each critical point P_i
Change the position of P_i between P_{i−1} and P_{i+1} randomly
END FOR
Noise Adding
Input a pattern P
FOR each point P_i
    Generate a probability R randomly
    IF R < threshold (0.5 in this paper)
      Generate a value A randomly (A ∈ [0,0.3])
      P_i = P_i + A(P_{i+1} − P_i)
    END IF
END FOR
```

Fig. 15. (a) A false-positive TripleTop pattern found by RB and Hybrid methods (HSI during 2011/12/22–2012/05/08) (b) The enlarged right part of subsequence *S*.

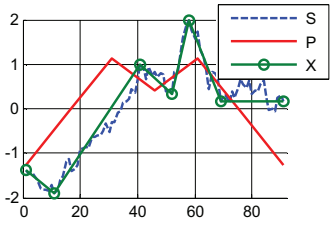

Fig. 16. TB misclassified *S* as a DoubleTop pattern (HSI during 2010/08/25–2010/12/31).

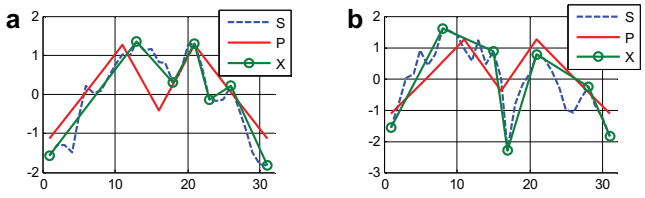

Fig. 17. Two cases showing the effect of PIP segmentation on the matching results (a) TB, RB and Hybrid do not recognise *S* (HSI during 2004/02/05–2004/03/18) as a Double Top pattern (b) RB and Hybrid methods do not recognise *S* (HSI during 2009/11/15–2009/12/17) as a Double Top pattern.

defined with seven points. However, the decision to use seven points influenced the inaccuracy of PIP. Fig. 17 shows two cases to illustrate how PIP affects the matching result.

As shown in Fig. 17(a) and (b), *S* is similar to *P*. But for Fig. 17(a), the approaches that make use of PIP (TB, RB and Hybrid) do not recognise it as a DoubleTop pattern because the segmented subsequence *X* has 'three tops'. For Fig. 17(b), *X* is similar to *P*. However, the RB and Hybrid methods do not recognise it as a DoubleTop pattern because these two methods use rules for pattern matching and one of the rules requires that the third point of *X* must be higher than the second point. The firing of this rule returns false when *X* is input for pattern matching.

5.2.2. Analysis of UCRS, DTW and ED methods

The UCRS, DTW and ED methods do not use PIP for preprocessing. Amongst these approaches, UCRS and DTW achieve the same result because UCRS is simply the 'sped up' version of DTW. Therefore, we focus our analysis on the comparison of the DTW and ED methods. In general, DTW has the ability to find hidden patterns inside a subsequence because DTW can recognise

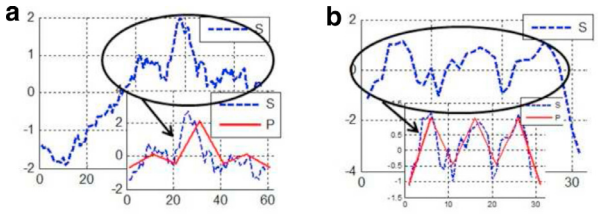

Fig. 18. Two cases that illustrate the differences between DTW and ED methods (a) the right part of S is similar to a SpikeTop pattern (HSI during 2010/08/25–2010/12/31) (b) the left part of S is similar to a TripleTop pattern (HSI during 2007/01/17–2007/03/01).

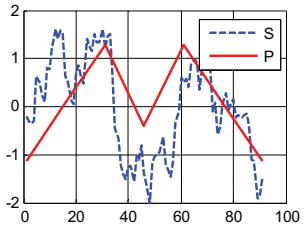

Fig. 19. EUCRS recognises a DoubleTop pattern (HSI during 2004/11/17–2005/03/31).

time-distorted subsequences. Two cases that show the differences between the DTW and ED methods are given in Fig. 18.

In Fig. 18(a) and (b), S is not similar to any pattern, but if the right part of S in Fig. 18(a) and the left part of S in Fig. 18(b) are zoomed in, they are similar to SpikeTop and TripleTop patterns, respectively, which are recognised by DTW. Hence, DTW is more effective when the length of the pattern of interest is not known in advance. In contrast, if the length of the subsequence is known or predefined, then the ED method will perform better.

5.2.3. Analysis of EUCRS

Recall that in Fig. 11, the accuracy of DTW and the EUCRS are similar. However, EUCRS does not support a spike in the time distortion. This situation is illustrated in Fig. 18. DTW recognises Fig. 18(a) and (b) as SpikeTop and TripleTop patterns, respectively, but EUCRS does not. Meanwhile, EUCRS is not as strict as the ED method. As illustrated in Fig. 19, EUCRS recognises the subsequence as a DoubleTop pattern, but the ED method does not recognise this.

6. Conclusions

A novel subsequence search and pattern matching method called EUCRS is proposed in this paper. In EUCRS, we extend the subsequence search algorithm UCR Suite (UCRS) with a Support Vector Machine (SVM) to train a classifier for chart pattern-matching in financial time series. Our approach not only calculates the distance between two time series as in DTW or the UCR Suite, but also adopts the advantage of an SVM in the recognition of the patterns. SVM is more sensitive to important points and is well suited for our purposes. In the experiments with synthetic and real data sets, we compare the effectiveness of EUCRS with 6 existing subsequence-matching approaches (TB, RB, Hybrid, ED, DTW, UCRS). The experimental results on synthetic and real data show that EUCRS performs better than other approaches. In addition, the execution speed of EUCRS is also faster than that of other approaches.

Due to its high accuracy and fast speed in searching, EUCRS can be used by financial analysts to automate the tasks of locating and identifying chart patterns in historical price data. EUCRS can also be adopted for matching chart patterns in real-time streaming stock data. In this case, the right-hand border of a sliding window can be mapped into the recently received data point to search the subsequences in the reverse order of the time dimension (x axis). For instance, EUCRS can be integrated to technical analysis trading software provided by brokerage firms. Some of the functions provided by these software include functions for displaying quotes and real-time market news, calculating technical indicators, trading, research, and analysis. Specifically, the proposed method can be used as a component of an expert system to assist traders in forecasting market trends. For instance, chart patterns were used for prediction of stock market timing (Lee & Jo, 1999), discovery of trading rules (Leigh, Modani, Purvis, & Roberts, 2002), identification of abrupt increases in volume in the New York Stock Exchange Composite Index (NYSE) (Leigh, Modani, & Hightower, 2004), and forecasting of the NYSE composite index (Leigh, Purvis, & Ragusa, 2002) and the DJIA index (Cervelló-Royo, Guijarro, & Michniuk, 2015). Our approach can be considered complementary to the above systems.

From the experimental results, we also uncover several performance issues of other pattern matching methods. The TB method considers many criteria, including the coordinates and ratio, but it relies on a segmentation method and could lead to a loss of information from the original subsequence. Moreover, it does not take into account the relative positions of important points. The important points are the points selected to remain in subsequence X after it is segmented. They are not eliminated because the segmentation approach treats them as important points. The RB method is easy and intuitive. However, the design of rules requires expert knowledge and experience from the financial domain. Designing rules can also be time consuming and tedious. The Hybrid method makes use of the SC plus the RB method as the criteria for pattern matching, and because segmentation is used as a pre-processing step, it suffers drawbacks similar to those of the TB approach. Naïve DTW and UCRS are both DTW-based measures of distance; therefore, their results are also similar. These two approaches do not depend on a segmentation method, but they do require longer calculation times. UCRS accelerates the original DTW algorithm and is faster than the ED method. Therefore, UCRS is efficient in terms of execution speed. However, similar to the TB method, it does not take into account the relative positions of important points, which are very important in the analysis of financial time series.

As for future work, we plan to extend the subsequence-matching problem to allow the flexible matching of a subsequence whereby the size of the subsequence is not necessarily equal to that of the pattern, as mentioned in Fu et al. (2007).

Acknowledgments

This research is funded by the University of Macau under grants MYRG041(Y1-L1)-FST13-SYW and MYRG2015-00054-FST.

References

Agrawal, R., Faloutsos, C., & Arun, S. (1993). Efficient similarity search in sequence databases. In *Proceedings of the fourth international conference on foundations of data organization and algorithms* (pp. 69–84).

Batista, G. E. A. P. A., Wang, X., & Keogh, E. J. (2011). A complexity-invariant distance measure for time series. In *Proceedings of SIAM international conference on data mining* (pp. 699–710).

Berndt, D., & Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *Proceedings of the AAAI workshop on knowledge discovery in databases* (pp. 229–248).

Byvatov, E., Fechner, U., Sadowski, J., & Schneider, G. (2003). Comparison of support vector machine and artificial neural network systems for drug/nondrug classification. *Journal of Chemical Information and Computer Sciences, 43*, 1882–1889.

Cervelló-Royo, R., Guijarro, F., & Michniuk, K. (2015). Stock market trading rule based on pattern recognition and technical analysis: Forecasting the DJIA index with intraday data. *Expert systems with Applications, 42*, 5963–5975.

Chan, K. P., & Fu, A. C. (1999). Efficient time series matching by wavelets. In *Proceedings of the 15th IEEE international conference on data engineering* (pp. 126–133).
Chang, C. C., & Lin, C. J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST), 2*, 27.
Chen, Y., Chen, G., Chen, K., & Ooi, B. C. (2009). Efficient processing of warping time series join of motion capture data. In *Proceedings of the IEEE international conference on data engineering* (pp. 1048–1059).
Chung, F. L., Fu, T. C., Luk, R., & Ng, V. (2001). Flexible time series pattern matching based on perceptually important points. In *Proceedings of the international joint conference on artificial intelligence (IJCAI) workshop on learning from temporal and spatial data* (pp. 1–7).
Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*, 273.
Ding, C. H., & Dubchak, I. (2001). Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics, 17*, 349–358.
Ding, Y. W., Yang, X. H., Kavs, A. J., & Li, J. F. (2010). Financial data representation and similarity model. *International Journal of Trade, Economics and Finance, 1*, 320–324.
Fu, T. C., Chung, F. L., & Ng, C. M. (2006). Financial time series segmentation based on specialized binary tree representation. In *Proceedings of international conference on data mining* (pp. 1–7).
Fu, T. C., Chung, F. L., Luk, R., & Ng, C. M. (2007). Stock time series pattern matching, template-based vs. rule-based approaches. *Journal of Engineering Applications of Artificial Intelligence, 20*, 347–364.
Fu, T. C., Chung, F. L., Luk, R., & Ng, C. M. (2008). Representing financial time series based on data point importance. *Journal of Engineering Applications of Artificial Intelligence, 21*, 277–300.
Fu, W. C., Keogh, E., Lau, Y. H., Ratanamahatana, C. A., & Wong, C. W. (2008). Scaling and time warping in time series querying. *Journal the VLDB*, 899–921.
Huang, C.-L., Chen, M.-C., & Wang, C.-J. (2007). Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications, 33*, 847–856.
Huang, Z., Chen, H., Hsu, C.-J., Chen, W.-H., & Wu, S. (2004). Credit rating analysis with support vector machines and neural networks: A market comparative study. *Decision Support Systems, 37*, 543–558.
Jeong, Y. S., Jeong, M. K., & Omitaomu, O. A. (2011). Weighted dynamic time warping for time series classification. *Journal Pattern Recognition, 44*, 2231–2240.
Jiang, J., Zhang, Z., & Wang, H. (2007). A new segmentation algorithm to stock time series based on PIP approach. In *Proceedings of the international conference on wireless communications* (pp. 5609–5612).
Keogh, E. (2002). Exact indexing of dynamic time warping. In *Proceedings of the 28th international conference on very large data bases* (pp. 406–417).
Keogh, E., & Kasetty, S. (2003). On the need for time series data mining benchmarks: A survey and empirical demonstration. *Data Mining and Knowledge Discovery, 7*, 349–371.
Keogh, E., Chakrabarti, K., Mehrotra, S., & Pazzani, M. (2001). Locally adaptive dimensionality reduction for indexing large time series databases. In *Proceedings of the ACM SIGMOD international conference on management of data* (pp. 151–163).
Keogh, E., Chakrabarti, K., Pazzani, M., & Sharad, M. (2001). Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems, 3*, 263–286.
Keogh, E., Chu, S., Hart, D., & Pazzani, M. (2004). Segmenting time series: A survey and novel approach. *Data Mining in Time Series Databases, 57*, 1–22.
Keogh, E., & Pazzani, M. (2000). A simple dimensionality reduction technique for fast similarity search in large time series databases. In *Proceedings of the Fourth Pacific-Asia conference on knowledge discovery and data mining (PAKDD)* (pp. 122–133).
Keogh, E., Wei, L., Xi, X., Vlachos, M., Lee, S. H., & Protopapas, P. (2009). Supporting exact indexing of arbitrarily rotated shapes and periodic time series under Euclidean and warping distance measures. *The International Journal on Very Large Data Bases, 18*, 611–630.
Lee, K. H., & Jo, G. S. (1999). Expert system for predicting stock market timing using a candlestick chart. *Expert Systems with Applications, 16*, 357–364.
Leigh, W., Modani, N., & Hightower, R. (2004). A computational implementation of stock charting: Abrupt volume increase as signal for movement in New York Stock Exchange Composite Index. *Decision Support Systems, 37*, 515–530.
Leigh, W., Modani, N., Purvis, R., & Roberts, T. (2002). Stock market trading rule discovery using technical charting heuristics. *Expert Systems with Applications, 23*, 155–159.
Leigh, W., Purvis, R., & Ragusa, J. M. (2002). Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: A case study in romantic decision support. *Decision Support Systems, 32*, 361–377.
Liu, X., Lin, Z., & Wang, H. (2008). Novel online methods for time series segmentation. *IEEE Transactions on Knowledge and Data Engineering, 20*, 1616–1626.
Lo, A. W., Mamaysky, H., & Wang, J. (2000). Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *Journal of Finance, 55*, 1705–1765.
Papapetrou, P., Athitsos, V., Potamias, M., Kollios, G., & Gunopulos, D. (2011). Embedding-based subsequence matching in time-series databases. *ACM Transactions on Database Systems, 36*(3), 17:1–17:39.
Pratt, K. B. (2001). *Locating patterns in discrete time-series*. Florida: University of South Florida.
Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., & Keogh, E. (2012). Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the conference on knowledge discovery and data mining* (pp. 262–270).
Sakurai, Y., Faloutsos, C., & Yamamuro, M. (2007). Stream monitoring under the time warping distance. In *Proceedings of ICDE* (pp. 1046–1055).
Shin, K.-S., Lee, T. S., & Kim, H.-J. (2005). An application of support vector machines in bankruptcy prediction model. *Expert Systems with Applications, 28*, 127–135.
YahooFinance University of South Florida. Historical stock price. In. http://finance.yahoo.com/q/hp?s=%5EHSI+Historical+Prices Accessed 14.08.15.
Zaib, G., Ahmed, U., & Ali, A. (2004). Pattern recognition through perceptually important points in financial time series. In *Proceedings of the international conference on fuzzy sets and soft computing in economics and finance* (p. 89).
Zhang, Z., Jiang, J., Liu, X., Lau, R., Wang, H., & Zhang, R. (2010). A real time hybrid pattern matching scheme for stock time series. In *Proceedings of the 21st Australasian conference on database technologies* (pp. 161–170).
Zhou, M., Wong, M. H., & Chu, K. W. (2006). A geometrical solution to time series searching invariant to shifting and scaling. *Knowledge and Information Systems, 9*, 202–229.
Zou, P., Su, L., Jia, Y., Han, W., & Yang, S. (2008). Fast similarity matching on data stream with noise. In *Proceedings of the ICDEW* (pp. 194–199).