

Exact Discovery of Length-Range Motifs^{*}

Yasser Mohammad¹ and Toyoaki Nishida²

¹ Assiut University, Egypt and Kyoto University, Japan
yasserm@aun.edu.eg, yasser@ii.ist.i.kyoto-u.ac.jp

² Kyoto University, Japan
nishida@i.kyoto-u.ac.jp

Abstract. Motif discovery is the problem of finding unknown patterns that appear frequently in real valued timeseries. Several approaches have been proposed to solve this problem with no a-priori knowledge of the timeseries or motif characteristics. MK algorithm is the de facto standard exact motif discovery algorithm but it can discover a single motif of a known length. In this paper, we argue that it is not trivial to extend this algorithm to handle multiple motifs of variable lengths when constraints of maximum overlap are to be satisfied which is the case in many real world applications. The paper proposes an extension of the MK algorithm called MK++ to handle these conditions. We compare this extensions with a recently proposed approximate solution and show that it is not only guaranteed to find the exact top pair-motifs but that it is also faster. The proposed algorithm is then applied to several real-world time series.

1 Introduction

Discovering recurrent unknown patterns in time series data is known in data mining literature as motif discovery problem. Since the definition of the problem in [6], motif discovery became an active area of research in Data Mining. A time series motif is a pattern that consists of two or more similar subsequences based on some distance threshold. Several algorithms have been proposed for solving the motif discovery problem [9], [7], [6], [3].

These algorithms can be divided into two broad categories. The first category includes algorithms that discretize the input stream then apply a discrete version of motif discovery to the discretized data and finally localize the recovered motifs in the original time series. Many algorithms in this category are based on the PROJECTIONS algorithm that reduces the computational power required to calculate distances between candidate subsequences [1]. Members of this category include the algorithms proposed in [3] and [7]. The second category discovers the motifs in the real valued time-series directly. Example algorithms in this category can be found in [9] and [2]. For example, Minnen et al. [7] used a discretizing motif discovery (MD) algorithm based on PROJECTIONS for basic action discovery of exercises in activity logs and reported an accuracy of 78.2% with a precision

* The work reported in this paper is partially supported by JSPS Post-Doc Fellowship and Grant-In-Aid Programs of the Japanese Society for Promotion of Science.

of 65% and recall of around 92%. The dataset used contained six exercises with roughly 144 examples of each exercises in a total of 20711 frames [7].

The second category tries to discover motifs in the original continuous time-series directly. Example algorithms in this category can be found in [2] and [9]. For example, Mohammad and Nishida [8] proposed an algorithm called MC-Full based on comparison of short subsequences of candidates sampled from the distribution defined by a change point discovery algorithm. Most of these algorithms are *approximate* in the sense that discovered motif occurrences may not have the shortest possible distance of all subsequences of the same length in the timeseries.

Mueen et al. [12] proposed the MK algorithm for solving the *exact* motif discovery problem. This algorithms can efficiently find the pair-motif with smallest Euclidean distance (or any other metric) and can be extended to find top k motifs or motifs within a predefined distance *range*. Mohammad and Nishida [10] proposed an extension of the MK algorithm (called MK+) to discover top k pair-motifs efficiently. The proposed algorithm in this paper, builds upon the MK+ algorithm and extends it further to discover top k pair-motifs at different time lengths.

The main contribution of the paper is a proof that mean-normalized Euclidean distance satisfies a simple inequality for different motif lengths which allows us to use a simple extension of MK+ to achieve higher speeds without sacrificing the exactness of discovered motifs.

2 Problem Statement

A time series $x(t)$ is an ordered set of T real values. A subsequence $x_{i,j} = [x(i) : x(j)]$ is a contiguous part of a time series x . In most cases, the distance between overlapping subsequences is considered to be infinitely high to avoid assuming that two sequences are matching just because they are shifted versions of each other (these are called trivial motifs [5]). There are many definitions in literature for motifs [13], [3] that are not always compatible. In this paper we utilize the following definitions:

Definition 1. Motif: Given a timeseries x of length T , a motif length L , a range R , and a distance function $D(.,.)$; a motif is a set M of n subsequences ($\{m_1, m_2, \dots, m_n\}$) of length L where $D(m_i, m_{j_k}) < R$, for any pairs $m_i, m_k \in M$. Each $m_i \in M$ is called an *occurrence*.

Definition 2. Pair-Motif: A pair-motif is a motif with exactly 2 occurrences. We call the distance between these two occurrences, the *motif distance*.

Definition 3. Exact Motif: An Exact Motif is a pair-motif with lowest range for which a motif exists. That is a pair of subsequences $x_{i,i+l-1}, x_{j,j+l-1}$ of a time series x that are most similar. More formally, $\forall_{a,b,i,j}$ the pair $\{x_{i,i+l-1}, x_{j,j+l-1}\}$ is the exact motif iff $D(x_{i,i+l-1}, x_{j,j+l-1}) \leq D(x_{a,a+l-1}, x_{b,b+l-1})$, $|i-j| \geq w$ and $|a-b| \geq w$ for $w > 0$. This definition of an exact motif is the same as the one used in [12].

Using these definitions, the problem statement of this paper can be stated as: Given a time series x , minimum and maximum motif lengths (L_{min} and L_{max}), a maximum allowed within-motif overlap (wMO), and a maximum allowed between-motifs overlap (bMO), find the top k pair-motifs with smallest motif distance among all possible pairs of subsequences with the following constraints:

1. The overlap between the two occurrences of any pair-motif is less than or equal to wMO . Formally, $\forall M_k \forall m_1 \in M_k, m_2 \in M_k : overlap(m_1, m_2) \leq wMO$ where M_k is one of the top k motifs at length l , and $overlap(.,.)$ is a function that returns the number of points common to two subsequences divided by their length (l).
2. The overlap between any two pair-motifs is less than or equal to bMO . Formally, $\forall 1 \leq i, j \leq K \forall 1 \leq l, n \leq 2 : min(overlap(m_i^l, m_n^j)) \leq bMO$, where m_y^x is the occurrence number y in motif number x and $overlap(.,.)$ is defined as in the previous point.

This paper will provide an *exact* algorithm for solving this problem in the sense that *there are no possible pair-motifs that can be found in the time series with motif distance lower than the motif distance of the K^{th} motif at every length*. Given this solution, it is easy to find motifs that satisfy Definition 1 and in the same time find a data-driven meaningful range value by simply combining pair-motifs that share a common (or sufficiently overlapping) occurrences. Our solution is based on the MK algorithm of Mueen et al. in [12] and for this reason we start by introducing this algorithm and its previous extension in the following section.

3 MK and MK+ Algorithms

The MK algorithm finds the top pair-motif in a time series. The main idea behind MK algorithm [12] is to use the triangular inequality to prune large distances without the need for calculating them. For metrics $D(.,.)$ (including the Euclidean distance), the triangular inequality can be stated as:

$$D(A, B) - D(C, B) \leq D(A, C) \tag{1}$$

Assume that we have an upper limit on the distance between the two occurrences of the motif we are after (th) and we have the distance between two subsequences A and C and some reference point B . If subtracting the two distances leads a value greater than th , we know that A and C cannot be the motif we are after without ever calculating their distance. By careful selection of the order of distance calculations, MK algorithm can prune away most of the distance calculations required by a brute-force quadratic motif discovery algorithm. The availability of the upper limit on motif distance (th), is also used to stop the calculation of any Euclidean distance once it exceeds this limit. Combining these two factors, 60 folds speedup was reported in [12] compared with the brute-force approach.

The inputs to the algorithm are the time series x , its total length T , motif length L , and the number of reference points N_r .

The algorithm starts by selecting a random set of N_r reference points. The algorithm works in two phases:

The first phase (called hereafter referencing phase) is used to calculate both the upper limit on best motif distance and a lower limit on distances of all possible pairs. During this phase, distances between the subsequences of length L starting at the N_r reference points and all other $T - L + 1$ points in the time series are calculated resulting in a distance matrix of dimensions $N_r \times (T - L + 1)$. The smallest distance encountered (D_{best}) and the corresponding subsequence locations are updated at every distance calculation.

The final phase of the algorithm (called scanning step hereafter) scans all pairs of subsequences in the order calculated in the referencing phase to ensure pruning most of the calculations. The scan progressed by comparing sequences that are k steps from each other in this ordered list and use the triangular inequality to calculate distances only if needed updating D_{best} . The value of k is increased from 1 to $T - L + 1$. Once a complete pass over the list is done with no update to d_{best} , it is safe to ignore all remaining pairs of subsequences and announce the pair corresponding to D_{best} to be the *exact* motif.

A naive way to extend the MK algorithm to discover top k pair-motifs rather than only the first is to apply the algorithm K times making sure in each step (i) to ignore the $(i - 1)$ pair-motifs discovered in the previous calls. To resist trivial motif matches, a maximum allowable overlap between any two motifs can be provided and all candidates overlapping with an already discovered motif with more than this allowed maximum are also ignored. A major problem with this approach is that even though we require $K * (l_{max} - l_{min} + 1)$ pair-motifs, the number of times MK must be ran can be much higher because it is likely that the motif discovered in run $i + 1$ will be a trivial match of the motif discovered in run i . This is more likely to happen the longer and smoother motifs are. To extend the algorithm to discover motifs of multiple lengths, we simply apply the algorithm for every length sequentially. This algorithm will be called *NaiveMK* in the rest of this paper.

3.1 MK+

Mohammad and Nishida proposed an extension to the MK algorithm for discovering top k pair-motifs of the same length efficiently (MK+) [10] that keeps an ordered list of K candidate motifs (L_{bests}) and their corresponding motif distances (D_{bests}) rather than a single candidate motif during both the referencing and scanning phases. This allows MK+ to discover the top k motifs exactly in a single run. This algorithm was reported to provide an order of magnitude improvement in speed compared to rerunning MK [10].

MK+ though works with a single length. A possible extension of this algorithm to multiple lengths is by simply re-running it for each length. This is called *NMK+* in the rest of this paper. As will be shown in section 5, this approach is suboptimal in terms of speed and a faster approach is proposed in this paper.

4 Proposed Extension

The main idea behind the proposed extension is to use the fact that distance calculations at one length provide an upper bound on possible distances of lower lengths and a lower limit on distances of longer lengths. This fact can be utilized (in a similar way to the triangular inequality in the original MK algorithm) to speed up the calculations by removing the need to compare pairs that are *obviously* not candidates for being pair-motifs according to these bounds.

For this approach to work the distance function used to compare subsequences must be a metric (as is the case with the original MK algorithm). We also need to be sure that for any positive number s the following property is true:

$$D_l(i, j)D_{l+1}(i, j) \forall l > 0 \wedge 1 \leq i; j \leq T - l + 1 \quad (2)$$

where $x_{a:b}$ is the subsequence of the time series x starting at index a and ending at index b and all indices do not exceed the length of the time series and $D_l(i, j) = D(x_{i:i+l-1}, x_{j:j+l-1})$. If the condition in Eq. 2 is not satisfied then distances at different lengths cannot be used to infer any bounds about distances of either larger or lower lengths and the best that could be done is *NMK+*. This puts a limitation on the types of distance functions that can be used. For example, Mueen et al. normalized all subsequences by subtracting the mean and dividing by the standard deviation prior to distance calculation [12]. This cannot be done in our case because this distance function does not respect the condition in Eq. 2. For this reason we normalize all subsequences only by subtracting the mean.

The effect of division by the standard deviation before distance calculation is to reduce the effect of variability in scale. In some applications; this is not needed or even desirable. For example, one of our primary goal domains is discovery of recurring motions in human motion and gesture datasets. A small change in scale between two occurrences of the same action can be handled with the Euclidean distance while a large scale difference corresponds in many cases to a quantitative difference in the action.

If subsequences are not normalized and the distance measure used is the Euclidean is used for $D(., .)$, it is trivial to show that equation 2 holds. Subtracting the mean of the whole timeseries does not change this result. It is not immediately obvious that normalizing each subsequences by subtracting its own mean will result in a distance function that respects Equation 2.

Theorem 1. *Given that the distance function $D(., .)$ in Definition 1 is defined for subsequences of length l as:*

$$D_l(i, j) = \sum_{k=0}^{l-1} [(x_{i+k} - \mu_i^l) - (x_{j+k} - \mu_j^l)]^2 \quad (3)$$

and $\mu_i^n = \frac{1}{n} \sum_{k=0}^{n-1} x_{i+k}$; then

$$D_l(i, j)D_{l+1}(i, j) \forall l > 0 \wedge 1 \leq i; j \leq T - l + 1$$

A sketch of the proof for Theorem 1 is given below: Let $\Delta_k \equiv \Delta_{j+l}^{i+k} \equiv x_{i+k} - x_{j+k}$ and $\mu_{ij}^l \equiv \mu_j^l - \mu_i^l$, then the definition of $D_l(i, j)$ can be written as:

$$D_l(i, j) = \sum_{k=0}^{l-1} (\Delta_k - \mu_{ij}^k)^2 = \sum_{k=0}^{l-1} (\Delta_k)^2 - 2 \sum_{k=0}^{l-1} (\Delta_k \mu_{ij}^k) + \sum_{k=0}^{l-1} (\mu_{ij}^k)^2$$

$$D_l(i, j) = -l(\mu_{ij}^l)^2 + \sum_{k=0}^{l-1} (\Delta_k)^2$$

We used the definition of μ_{ij}^l in arriving at the last result. Using the same steps with $D_{l+1}(i, j)$, subtracting and with few manipulations we arrive at:

$$D_{l+1}(i, j) - D_l(i, j) = \Delta_l^2 + l(\mu_{ij}^l)^2 - (l+1)(\mu_{ij}^{l+1})^2 \quad (4)$$

Using the definition of μ_{ij}^l , it is straight forward to show that:

$$(l+1)\mu_{ij}^{l+1} = l\mu_{ij}^l + \Delta_l \quad (5)$$

Substituting in Equation 4 and rearranging terms we get:

$$D_{l+1}(i, j) - D_l(i, j) = \Delta_l^2 + (\mu_{ij}^l)^2 - 2\mu_{ij}^l \Delta_l = (\Delta_l - \mu_{ij}^l)^2 \geq 0$$

This proves Theorem 1.

4.1 MK++

The following listing gives an overview of the proposed algorithm:

```

Function MK++(x, {L}, K, R, wMO, bMO)
1. Find motifs at shortest length
Lbests ← ∅, Dbests ← ∅
For (r ∈ Rand(R) ∧ i = 1 : |x|)
    IF (|i - r| ≥ wMO × min(L))
        UpdateBests (min(L), K, bMO, r, i)
    EndIf
EndFor
Zr(i) ← ordering of reference points by their distance variance
ref ← reference point with max. variance
Z(i) ← ordering of subsequences by distance to ref
Dist ← {D(ref, Z(i))}
ScanningPhase()
2. Find motifs at higher lengths
For (l ∈ {L})
    Update Dbests then re-sort
    IF (R)
        recalculate ref

```

```

    EndIf
    ScanningPhase()
EndFor

```

The MK++ algorithm starts by detecting 2-motifs at the shortest length (L_{min}) and progressively finds 2-motifs at higher lengths. The algorithm keeps three lists: D_{bests} representing a sorted list of K best distances encountered so far and L_{bests} representing the 2-occurrence motif corresponding to each member of D_{bests} , and μ_{bests} keeping track of the means of the subsequences in L_{bests} . The *best-so-far* variable of MK is always assigned to the maximum value in D_{bests} . During the referencing phase, the distance between the current reference subsequences and all other subsequences of length L_{min} that do not overlap it with more than $wMO \times L_{min}$ points are calculated. For each of these distances (d) we apply the following rules in order:

- Rule1.** If the new pair is overlapping the corresponding $L_{bests}(i)$ pair with more than $wMO \times L$ points, then this i is the index in D_{bests} to be considered
- Rule2.** If *Rule1* applies and $D < D_{bests}(i)$, then replace $L_{bests}(i)$ with P .
- Rule3.** If *Rule1* does not apply but $D < D_{bests}(i)$, then we search L_{bests} for all pairs $L_{bests}(i)$ for which *Rule1* applies and remove them from the list. After that the new pair P is inserted in the current location of L_{bests} and D in the corresponding location of D_{bests}

It is trivial to show that following these rules D_{bests} will always be a sorted list of *best-so-far* distances encountered that satisfy both between and within motif overlap constraints (wMO and bMO). The details of this calculation are given in the program listing below:

```

Function updateBests(l,K,bMO,r,i)
done ← false, d ← D(xr:r+l-1, xi:i+l-1)
IF (d > max(Dbests) ∧ |Dbests| < K)
    Lbests ← Lbests ∪ ⟨r,i⟩, Dbests ← Dbests ∪ d, done ← true
EndIf
For(j = 1 : K ∧ ¬done)
    ζ ← OverlapBoth(Lbests[j], ⟨r,i⟩)
    IF(ζ > bMO)
        IF(d < max(Dbests))
            Lbests[j] ← ⟨r,i⟩, Dbests[j] ← d, done ← true
        EndIf
    Else
        IF(d < max(Dbests))
            o ← w : w ∈ Lbests ∧ OverlapBoth(w, ⟨r,i⟩) > bMO
            IF(o = ∅) o ← K EndIf
            remove o from Lbests, Dbests, Lbests ← Lbests ∪ ⟨r,i⟩
            Dbests ← Dbests ∪ d
        EndIf
    EndIf
EndIf
EndFor

```

These steps are similar to MK+ [10] except that we carry along the means of subsequences in the list μ_{bests} . Once the top K 2-motifs of the first length are found, the algorithm progressively calculates the top K 2-motifs of longer lengths. Since our distance function defined in Equation 3 satisfies the condition of Equation 2 as Theorem 1 proves, we know that the distances in D_{best} from the previous length provide a lower bound on the distances at this length. This allows us to directly update the D_{best} array by appending the new distances after using Equation 5 to calculate the new mean and distance.

Updating the distances in D_{bests} may take them out of order. We sort L_{bests} (with corresponding sorting of D_{bests} and μ_{bests}) in ascending order and update the *best-so-far* accordingly. This gives us a –usually– tight upper bound on the possible distances for the top K 2-motifs at the current length. This is specially true if the increment in motif length is small and the timeseries is smooth. This is the source of the speedup achieved by MK++ over NMK+ which have to find these lists from scratch at every length.

The referencing step can either be recalculated for the new length or the older reference values can be used as lower bounds (again because of Theorem 1). Nevertheless, recalculating the reference distances can speedup the scanning phase because it provides a tighter bound. Because this depends on the data, we keep the choice for the algorithm user by providing a parameter R that controls whether this recalculation is carried out. It is important to notice that either choice will not affect the accuracy of the final results and in our experiments it had a negligible effect on the speed as well. The scanning step is the same as in MK and MK+ and for lack of space will not be elaborated upon further. Please refer to [12] for its details.

5 Evaluation

The first evaluation considered comparing MK++ with NMK+. A set of 40 time series of lengths between 1000 and 10000 points each were generated using a random walk. This kind of random timeseries is the most challenging input for any *exact* motif discovery algorithm because distances between subsequences are similar which reduces the advantage of using lower and upper bounds. Both MK++ and NMK+ were used to find 10 pair-motifs of each length in the range 10 to 100. The average execution time for NMK+ was 3.8ms/point while the average execution of MK++ was 0.77ms/point achieving a speedup of more than 500%. Both algorithms returned the same motifs at all lengths. For completion we also applied NaiveMK to the same dataset which required an average execution time of 73.5ms/point (more than 60 times the proposed MK++ algorithm). Again both algorithms returned exactly the same motifs at all lengths.

5.1 Real World Data

To test MK++ on a real world dataset we used CMU Motion Capture Dataset available online from [4]. We used the time series corresponding to basketball



Fig. 1. Forward dribbling motion: discovered by the proposed algorithm

category. The occurrences of each recurring motion pattern in the time series of the fifteen available in this collection (21 different motions in total) were marked by hand from the videos using ANVIL [14] to get ground-truth motif locations.

In this paper, we report the results using AMC files only. The total number of frames in the fifteen time series was 8155. There were seven different motion patterns in the time series with an average of 8 occurrences each after removing motions that appeared only once or twice. Fig. 1 shows an example pattern.

The data from all sessions were concatenated to form a single timeseries and random data was added to it to make its length 10000 points. Before applying motif discovery algorithms, we reduced the dimensionality of the aggregated time series using Principal Component Analysis (PCA).

We applied MK++, shift-density based motif discovery (sdCMD) [15], two stem extension algorithms (GSteXS, GSteXB) [11], MCFull [9] and the recently proposed MOEN algorithm [13] with a motif length between 50 and 250 to the concatenated timeseries and calculated the fractions of discovered motifs that cover ground-truth patterns completely, and the fractions that cover partial patterns or multiple patterns. We also calculated the fraction of true patterns that were discovered by every algorithm (covered-fraction) and the extra parts appended to it (extra-motif). This data is summarized in Table 1.

Table 1. Evaluation of MD for CMU MoCap Dataset with the best and worst performance highlighted (worst in italics and best in bold font)

Algorithm	Correct	Partial	Multiple	Covered-fraction	Extra-motif	Accuracy
Proposed	0.3169	0.3761	0.1089	0.2543	3.7496	0.6931
sdCMD [15]	0.1594	<i>0.4783</i>	0.2899	0.1847	0.5264	0.6377
GSteXS [11]	0.1111	0.1296	0.6852	0.9796	8.8143	0.2407
GSteXB [11]	0.0323	0.2742	0.6129	0.8562	2.5713	0.3065
MCFull [9]	<i>0</i>	0	<i>1.0000</i>	1.0000	<i>115.8188</i>	<i>0.0</i>
MOEN [13]	0.3333	0	0.3333	<i>0.1299</i>	0.2001	0.3333

As Table 1 shows, the proposed algorithm provides a good balance between specificity (extra-motif=3.74) and sensitivity (covered-fraction=0.25) and provides the second best correct discovery rate (0.3169 compared with 0.3333 for MOEN) with best boundary separation between discovered motifs (multiple=0.1089). To get a sense of the accuracy of each algorithm we defined the total discovery rate as the summation of correct discovery rate and partial discoveries as it is always easy to use a motif-extension algorithm to extend partially discovered motifs. The proposed algorithm provides the highest total discovery rate of 69.31% compared with 63.77% for sdCMD, 33.33% for MOEN, 30.65%

for GSteXB, 24.07% for GSteXS and 0% for MCFull. Even though this result is based on a limited dataset, it provides a proof-of-applicability of the proposed algorithm for solving real world motif discovery problems. In the future, more extensive analysis with larger datasets will be conducted.

6 Conclusions

This paper presents an extension of the MK exact motif discovery algorithm called MK++ that can discover multiple motifs at multiple motif lengths simultaneously achieving a speed up of over 500% compared with repeated applications of the MK+ algorithm. The paper provides a proof that MK++ is still an exact algorithm for the mean-shifted Euclidean distance function. In the future, the proposed approach will be extended to zscore normalization.

References

1. Buhler, J., Tompa, M.: Finding motifs using random projections. In: 5th International Conference on Computational Biology, pp. 69–76 (2001)
2. Catalano, J., Armstrong, T., Oates, T.: Discovering patterns in real-valued time series. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 462–469. Springer, Heidelberg (2006)
3. Chiu, B., Keogh, E., Lonardi, S.: Probabilistic discovery of time series motifs. In: ACM SIGKDD KDD, pp. 493–498 (2003)
4. CMU: Cmu motion capture dataset, <http://mocap.cs.cmu.edu>
5. Keogh, E., Lin, J., Fu, A.: Hot sax: efficiently finding the most unusual time series subsequence. In: Fifth IEEE ICDM, pp. 8–17 (2005)
6. Lin, J., Keogh, E., Lonardi, S., Patel, P.: Finding motifs in time series. In: The 2nd Workshop on Temporal Data Mining, at the 8th ACM SIGKDD International, pp. 53–68 (2002)
7. Minnen, D., Starner, T., Essa, I.A., Isbell Jr., C.L.: Improving activity discovery with automatic neighborhood estimation. In: IJCAI, vol. 7, pp. 2814–2819 (2007)
8. Mohammad, Y., Nishida, T.: Learning interaction protocols using augmented bayesian networks applied to guided navigation. In: IEEE IROS, pp. 4119–4126 (2010)
9. Mohammad, Y., Nishida, T.: Constrained motif discovery in time series. *New Generation Computing* 27(4), 319–346 (2009)
10. Mohammad, Y., Nishida, T.: Unsupervised discovery of basic human actions from activity recording datasets. In: IEEE/SICE SII, pp. 402–409 (2012)
11. Mohammad, Y., Ohmoto, Y., Nishida, T.: G-SteX: Greedy stem extension for free-length constrained motif discovery. In: Jiang, H., Ding, W., Ali, M., Wu, X. (eds.) IEA/AIE 2012. LNCS, vol. 7345, pp. 417–426. Springer, Heidelberg (2012)
12. Mueen, A., Keogh, E.J., Zhu, Q., Cash, S., Westover, M.B.: Exact discovery of time series motifs. In: SDM, pp. 473–484 (2009)
13. Mueen, A.: Enumeration of Time Series Motifs of All Lengths. In: IEEE ICDM (2013)
14. Kipp, M.: Anvil – A Generic Annotation Tool for Multimodal Dialogue. In: Eurospeech, pp. 1367–1370 (2001)
15. Mohammad, Y., Nishida, T.: Approximately Recurring Motif Discovery Using Shift Density Estimation. In: Ali, M., Bosse, T., Hindriks, K.V., Hoogendoorn, M., Jonker, C.M., Treur, J. (eds.) IEA/AIE 2013. LNCS, vol. 7906, pp. 141–150. Springer, Heidelberg (2013)