

# Mining an Optimal Prototype from a Periodic Time Series: an Evolutionary Computation-based Approach

Pekka Siirtola, Perttu Laurinen, Juha Rönning  
Intelligent Systems Group, P.O. BOX 4500, FIN-90014, University of Oulu, Oulu, Finland  
email: {pesiirto, perttu, jjr}@ee.oulu.fi

**Abstract**—The mining of meaningful shapes of time series is done widely in order to find shapes that can be used, for example, in classification problems or in summarizing signals. Normally, shapes that summarize periodic signals have to be mined visually, and in order to find a shape of high quality, several tests have to be made. This makes visual mining slow and sometimes even frustrating. A method for summarizing a periodic time series automatically is presented in this study. The method is based on evolutionary computation and the results show that by using it, shapes can be found that summarize a time series better than shapes found using visual mining.

## I. INTRODUCTION

Mining of time series data is done widely around the world, and these studies have affected many fields of science. One of the main attractions in this field is the search for meaningful shapes from signals. In the literature these shapes are called *frequent patterns*, *motifs* and *prototypes*: in this study the term prototype is used. The shape of the prototype can be meaningful, for example in the sense that it describes some unusual or frequent shape of the time series. Therefore, prototypes can be used, for example, to summarize massive time series databases, and they can also be used to cluster time series data sets and in classification problems [1]. In order to summarize a time series as well as possible, the prototype has to be as descriptive as possible. As a periodic time series is considered a time series which is mainly composed of one regularly existing shape. As an optimal prototype to summarize some periodic time series we here consider a prototype that is similar to the greatest number of frequent patterns of the time series. Now, by using this prototype as a template, these pattern can be found and calculated.

Though it is assumed in this article that the studied time series is periodic, it does not mean that all the periods in the time series are exactly similar to each other. For example, if the data are collected using wearable sensors, the difference in periods can be caused by white noise and other disturbances. When the time series is collected with high frequency from a periodic human activity using a wearable sensor, the periods look much alike but are not fully identical, because humans seldom do the same thing exactly the same way twice. Hence the shape caused by a periodic human action cannot be generalized. Therefore, if one randomly chosen pattern from a periodic human activity time series is used as a template to find other periods, most likely not all the periods will be found because of the differences in the patterns. In addition, because all humans are individuals

they do the same activities in their very own personal way, they cause different kinds of shapes in activity signals, see Figure 1. Hence if a prototype summarizes a certain person's activity signal perfectly, it does not necessarily summarize another person's activity signal as well. In order to maximize the number of found periods, the prototype must be chosen wisely.

Methods for mining the best subsignal of a periodic time series to summarize the whole signal have been proposed in the literature. Such an algorithm for finding optimal prototype is presented *e.g.* in [2]. Note that in [2] and also in most of the other studies too, a prototype, or set of prototypes, which is a subsequence of  $T$  is considered an optimal prototype for summarizing time series  $T$ .

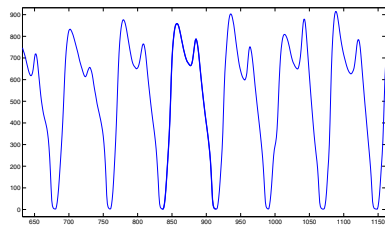
A subsequence of  $T$  is not always the best prototype for describing some time series  $T$ . It is possible that the prototype that summarizes a time series signal in an optimal way is a subsequence of some other time series or is not a part of any signal. This study suggest a method for mining an optimal prototype that describes a time series or group of time series in an optimal way, and this prototype is not necessarily a subsignal of any of these time series. The presented method is based on evolutionary computation and it is designed for finding prototypes from time series data where observations  $\mathbf{y}_i \in \mathbb{R}^d$  and  $d \in \mathbb{Z}_+$ , unlike other evolutionary computation based methods [3], [4], which are concerned with mining prototypes from discrete biological data, such as DNA sequences, where observations  $\mathbf{y}_i \in [a, b]^d$  and  $a, b \in \mathbb{Z}$ . So, in this study the data are continuous, while in other articles discrete data are studied.

## II. TERMINOLOGY AND NOTATIONS

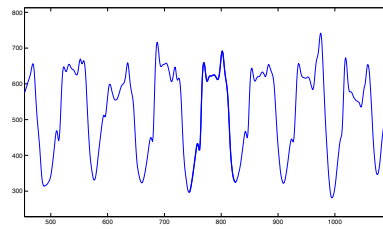
This section defines terms and notations used in this study.

A *time series*  $T = \{(x_1^T, \mathbf{y}_1^T), \dots, (x_n^T, \mathbf{y}_n^T)\}$  is considered a sequence of observations measured over time, such that each observation is indexed by a time variable  $x_i$  and the measurements  $\mathbf{y}_i$  at each time  $x_i$  can be multivariate [5]. The length of time series  $|T|$  is equal to  $n$ . In this work the measurements of time series are one-dimensional so instead of using symbol  $\mathbf{y}_i$ , symbol  $y_i$  is used to define measurements.

A sequence  $S = \{(x_1^S, \mathbf{y}_1^S), \dots, (x_m^S, \mathbf{y}_m^S)\}$  is a *subsequence* of time series  $T$  if  $S \subseteq T$ . Note that if observations  $(x_i, \mathbf{y}_i)$  and  $(x_j, \mathbf{y}_j)$  are sequential observations in  $S$ , they must also be sequential in  $T$ .



(a) Subject a



(b) Subject b

Fig. 1. Running signals from two individuals collected using wrist-worn accelerometer and prototypes that summarizes them.

TABLE I  
SYMBOLS AND DEFINITIONS

Symbol	Definition
$T$	Time series $T$
$S$	Subsignal $S$
$P_i$	Prototype $P_i$
$x_i^T$	The time stamp of the $i$ -th point of time series $T$
$x_{last}^T$	The time stamp of the last point of time series $T$
$y_i^T$	The $y$ -coordinate of the $i$ -th point of time series $T$
$ T $	The number of points in time series $T$
$d(T_1, T_2)$	The distance between time series $T_1$ and $T_2$

In this study, *prototype*  $P = \{(x_1^P, y_1^P), \dots, (x_w^P, y_w^P)\}$  is defined as a time series that summarizes a time series  $T$  or describes a particular activity. Note that though prototype  $P$  summarizes time series  $T$ , it does not mean that  $P$  is a subsequence of  $T$ . If a prototype  $P$  summarizes time series  $T$  and  $P$  is not a subsequence of  $T$ , then  $P$  is just *similar* to  $T$  or subsequences of  $T$ .

The similarity of time series can be calculated using several distance measures: in this study a similarity measure presented in Section IV-C is used. Common to all similarity measures is that time series  $T_1$  and  $T_2$ ,  $|T_1 \cap T_2| = 0$ , are considered *similar* if  $d(T_1, T_2) < \delta$ , where  $d(\cdot, \circ)$  is the distance between  $\cdot$  and  $\circ$  and  $\delta$  is a predefined *similarity bound*. The similarity bound must be chosen wisely, as too big a bound produces false positive results and using too small a limit will not ensure that all patterns are found. A proper similarity bound can be chosen, for instance, by trying several values and choosing the most suitable value as the bound based on these tests. Note that different distance measures produce different values, and therefore the bound has to be changed when the distance measure is altered.

The symbols and definitions used herein are shown in Table I.

### III. FINDING AN OPTIMAL PROTOTYPE

#### A. Finding subsignals similar to the prototype

Set  $\mathcal{S}$  of subsignals of time series  $T$ , similar to prototype  $P$ , are found by *sliding*  $P$  through  $T$ , so if  $S_w \in \mathcal{S}$ , then  $d(P, S_w) < \delta$ . Here the term sliding means that first  $P$  is compared to subsignals of  $T$  that start from point  $(x_1^T, y_1^T)$  and whose end point is a point of interval  $[x_{minL}, x_{maxL}]$ , where  $minL$  and  $maxL$  are predefined values that tell how many points subsignal is allowed to

contain at the minimum and maximum. The next thing to do is to measure the distances  $d_i = d(S_i, P)$  between subsignals  $S_i = \{(x_1^T, y_1^T), \dots, (x_i^T, y_i^T)\}$ ,  $minL \leq i \leq maxL$ , and prototype  $P$ . Let us assume that  $d_j = \min_{minL \leq i \leq maxL} d_i$ . Sliding proceeds by doing one of the following steps:

- 1) If  $d_j$  is smaller than predefined similarity bound  $\delta$ , subsignal  $S_j$  is considered similar to prototype  $P$  and is added to set  $\mathcal{S}$ , which includes all the subsignals of  $T$  that are similar to  $P$ . Next, prototype  $P$  is compared to subsignals starting from time stamp  $x_j^T$ , which is the very same time stamp where subsignal  $S_j$  ended, and ending at interval  $[x_{j+minL}, x_{j+maxL}]$ , and so on, Figure 2(a).
- 2) If  $d_j > \delta$ , then none of the subsignals  $S_i$  were similar to prototype  $P$ , so the subsignals will not be added to set  $\mathcal{S}$ . Next, subsignals  $S = \{(x_2^T, y_2^T), \dots, (x_{i+1}^T, y_{i+1}^T)\}$ ,  $minL \leq i \leq maxL$ , are compared to  $P$ , and so on, Figure 2(b).

These steps are repeated until the end of the time series  $T$  is reached.

#### B. Evolutionary computation-based approach

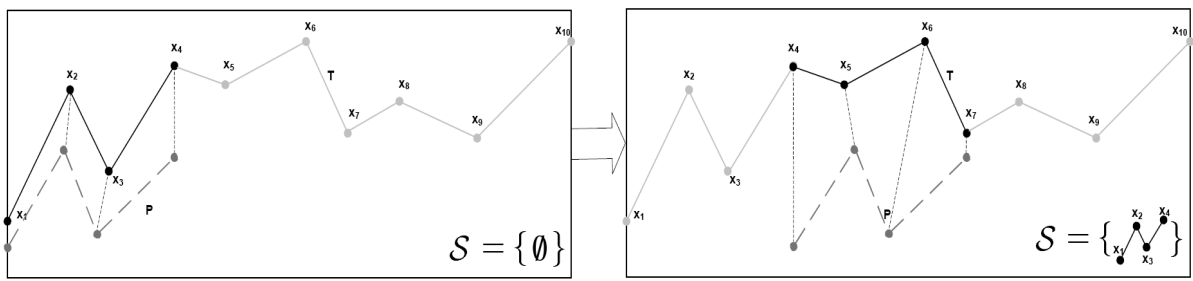
A method for evolutionary computation-based prototype mining is presented in Algorithm 1. As an input the algorithm gets a candidate collection  $\mathcal{P}_0$  (also called a candidate population or generation zero) of prototypes and the total number of iterations that tells how many populations the algorithm generates. The items in the candidate collection can be chosen by the user, for example by visually mining.

At every iteration, the algorithm transforms the population using genetic operations: crossover and mutation. Finally, new populations are formed by using a weighted lottery. In this article time series is periodic, thereby it can be assumed that by sliding optimal prototype  $P_{opt}$  through  $T$ , set  $\mathcal{S} = \{S_1, \dots, S_m\}$  can be found, where  $d(P_{opt}, S_i) < \delta$  for every  $i \leq m$ , of subsignals of time series  $T$ , so that

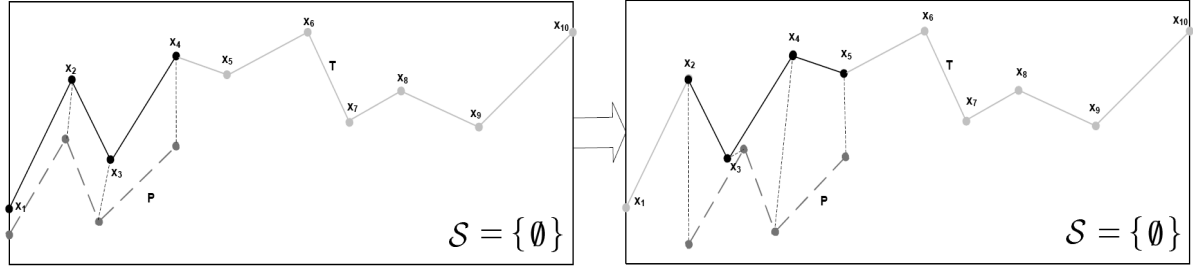
$$\frac{\sum_{i=1}^m (x_{last}^{S_i} - x_1^{S_i})}{x_{last}^T - x_1^T} \approx 1. \quad (1)$$

Because subsignals are not allowed to be located one on another, in the optimal case

$$S_1 \cup S_2 \cup \dots \cup S_m = T.$$



(a) Case 1: A subsignal of the time series is similar to  $P$  and is added to set  $S$ , which contains all the subsignals of the time series that are similar to  $P$ .



(b) Case 2: A subsignal of the time series is not similar to  $P$ , so it is not added to set  $S$ .

Fig. 2. Sliding consists of two cases. In case 2(a) prototype  $P$  is similar to the subsignal and in case 2(b) the similarity limit is chosen so that the prototype is not considered similar to the subsignal.

According to Equation (1), the fitness of prototype  $P$  can be defined using the function

$$f_P(S_{found}) = \left( \frac{\sum_{i=1}^k (x_{last}^{S_i} - x_1^{S_i})}{x_{last}^T - x_1^T} \right)^\alpha, \quad (2)$$

where  $S_{found}$  is a set of subsignals of time series  $T$  similar to candidate prototype  $P$  and  $\alpha \in \mathbb{R}_+$ .

After the desired number of populations is generated, the algorithm returns the most recent generation of prototypes.

a) *Crossover*: The crossover operation combines the features of two prototypes  $P_1$  and  $P_2$  in order to produce a new and better prototype  $P_{avg}$ , see Figure 3. The crossover algorithm scales two input prototypes to the same size: the values of each coordinate are scaled. Prototypes  $P_1$  and  $P_2$ , whose average prototype is calculated, can be chosen using, for instance fitness proportional selection, ranking selection, implementing selection probabilities or tournament selection [6]. In this study fitness proportional selection is used. The crossover algorithm combines the features of two input prototypes by searching for the closest point  $x_{P_2}^i$  of prototype  $P_2$  to every point  $x_{P_1}^j$  of prototype  $P_1$ . When the closest points are found, a combination of prototypes  $P_1$  and  $P_2$  is considered the time series  $P_{avg}$ , whose points are located in the middle of lines drawn between points  $x_{P_1}^j$  and  $x_{P_2}^i$ . The purpose of the crossover is to augment the portion of the good results in the candidate prototype set by creating average prototypes from prototypes with good fitness.

b) *Mutation*: Mutation changes the shape of the prototype a little and adds this modified prototype to the candidate prototype collection. A prototype is modified by moving a random number of points of the prototype into a different

position, Figure 4. Both the  $x$ - and the  $y$ -coordinate are moved and the new position is defined by adding a random value from a Gaussian distribution to the  $x$ - and the  $y$ -coordinates of the selected points. Different random values are added to different points. The purpose of mutation is to bring new and surprising features into the candidate set and to produce new and better prototypes.

c) *Selecting a new population*:: The new population is selected using a weighted lottery. The probability that an individual prototype  $P_k, 1 < k < m$  of the candidate set  $\mathcal{P}_i = \{P_1, \dots, P_m\}$  is selected to new population  $\mathcal{P}_{i+1}$  is

$$\frac{f_{P_k}}{\sum_{i=1}^m f_{P_i}}, \quad (3)$$

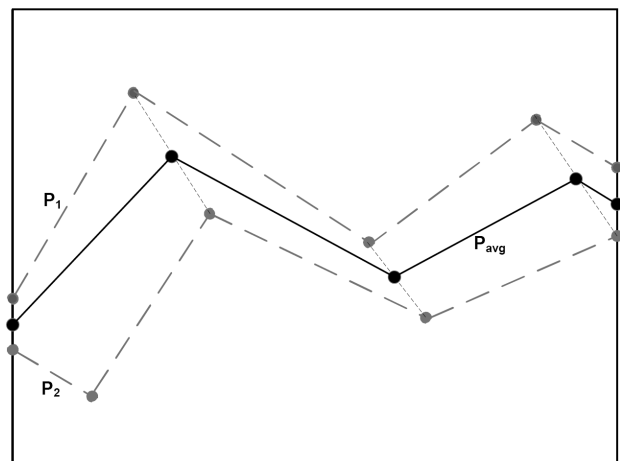


Fig. 3. Calculating average trajectory.

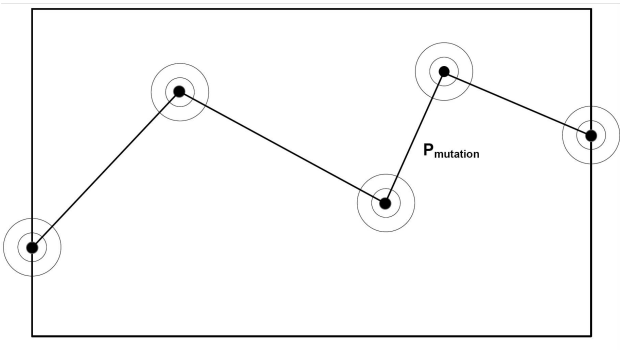


Fig. 4. Mutation changes the shape of the prototype by moving points to different positions. The new positions of the points lie within the circles.

---

**Algorithm 1:** Algorithm for mining prototypes.

---

**input** : Candidate collection  $\mathcal{P}_0 = \{P_1, \dots, P_n\}$ , Total number of iterations  $i$

**output**: Collection of prototypes  $\mathcal{P}_i$

**while** iteration number  $k < i$  **do**

Apply crossover to random number of prototypes pairs  $P_n$  and  $P_m$ ,  
add gained average prototypes to collection  $|\mathcal{P}_{k-1}|$ .

Apply mutation to random number of prototypes, add these prototypes to collection  $|\mathcal{P}_{k-1}|$

**for**  $j \leftarrow 1$  **to**  $|\mathcal{P}_{k-1}|$  **do**  
| set Fitness of prototype  $P_j$  to  $w_j$

**end**

Randomly choose  $n$  prototypes from  $\mathcal{P}_{k-1}$  weighted by  $w_j$  to generate new candidate collection  $\mathcal{P}_k$

**end**

**return**  $\mathcal{P}_i$ ;

---

where  $f_{P_i}$  is the weight of the prototype  $f_{P_i}$  calculated using Equation (2). Note that population  $\mathcal{P}_{i+1}$  can include several copies of a prototype  $P_i \in \mathcal{P}_i$ .

## IV. EXPERIMENTS

### A. Data set

The data were collected from nine volunteers. Each participant ran for ten minutes at an independently chosen speed, but the route was the same for everyone. This running data were used to test the presented method. Wearing sensors has to be easy, not time consuming and they should not disturb the user. For this reason only wrist-worn accelerometer was used. The measuring device was a one-dimensional capacitive accelerometer which was positioned to measure acceleration perpendicular to the direction of the forearm. The sampling frequency of the accelerometer was set to 100 Hz.

In order to speed up calculations, the data were compressed so that they contained only such points of the original data where the derivative is equal to zero, see Figure 5.

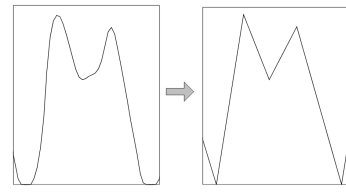


Fig. 5. Running prototype and a compressed version of it.

Therefore, the time series constituting the data set

$$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$$

was reduced to time series

$$T_{zero} = \{(x_k, y_k)\} \quad (4)$$

$$(y_{k-1} < y_k \wedge y_{k+1} < y_k) \vee (y_{k-1} > y_k \wedge y_{k+1} > y_k),$$

$$y_{k-1}, y_k, y_{k+1} \in T\}.$$

### B. About the experiment

In the experiments data described in Section IV-A were used. The results of the proposed algorithm were compared to the results of the visually mined prototypes. One prototype was selected from each person's running data as a feature that summarizes the signal. This prototype was extracted by choosing several prototype candidates by visually mining signals. The quality of these prototypes was tested using the fitness function (Equation 2), and based on this information, the best of these prototypes was selected as the prototype that represents the signal. This prototype was considered an optimal visually mined prototype.

When the similarity between prototype  $P$  and subsignal  $S$  of time series  $T$  was calculated, both  $P$  and  $S$  were scaled to the same size in order to eliminate the effect of different running speeds. This caused a problem: if all possible subsignals are scaled to the same size as the prototype, a huge number of false positive similarities are found, e.g. from white noise. To avoid this, before the similarity between  $P$  and  $S$  was calculated, we checked that the original amplitude  $a_S$  of  $S$  is between  $[(1-x) \cdot a_P, (1+x) \cdot a_P]$ , where  $a_P$  is the amplitude of prototype  $P$  and  $x \in [0, 1]$ . In this study the value  $x = 0.50$  was used. Only subsignals whose amplitude was between these limits were accepted as similar to prototype  $P$ .

Because of different running speeds, for example, the pattern that describes one running step can be longer for some persons than for others, therefore not only the similarities between subsignals of length  $|S| = |P|$  and prototype were calculated. Instead, all subsignals of  $T$  of length  $[\frac{|P|}{2}, 2 \cdot |P|]$  were compared to  $P$ . This means that  $|S|$  was not necessarily equal to  $|P|$ , and thus the similarity between  $P$  and  $S$  had to be calculated using a similarity measure that enables similarity calculation of two trajectories of different lengths.

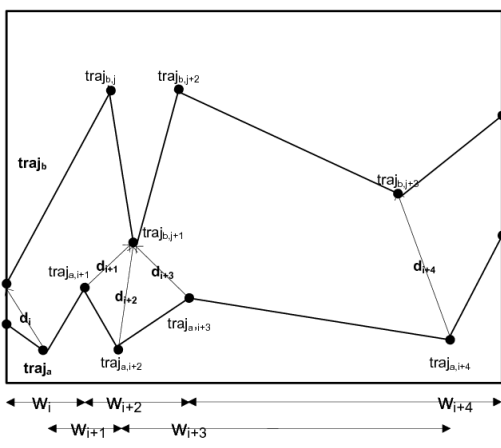


Fig. 6. Employed similarity measure gives different weights to the different points: points that are close to each other get smaller weights than do points that have no neighbors nearby.

### C. Weighted Distance Measure

Because the running signals were compressed using Equation 4, the signals are sparse, meaning that the data points of the signals are not distributed at equal-length intervals. Normally distance measures do not take this into consideration, so an interval including several points has significantly more weight on the distance than does an interval that does not include any points at all. A solution to this problem is presented in [7]. The article presents a similarity measure that gives different weights to different points: smaller weights to points that have neighbors nearby and bigger weight to points that are in areas where not many points exist, Figure 6. This ensures that areas with several points do not have a much greater impact on distance than do other areas. The weight of the point  $p_i = (x_i, y_i)$  of time series  $T$  can be calculated using equation:

$$w_i = \log(c_i + 1), \text{ where} \quad (5)$$

$$c_i = \begin{cases} \frac{x_2^T - x_1^T}{denominator}, i = 1, \\ \frac{x_{i+1}^T - x_{i-1}^T}{2 \cdot denominator}, 1 < i < |T|, \\ \frac{x_{last}^T - x_{last-1}^T}{denominator}, i = |T|, \text{ where} \end{cases} \quad (6)$$

$$denominator = \frac{x_{last}^T - x_1^T}{|T| - 1}. \quad (7)$$

Coefficients  $c_i$  tell how far away the neighbors of point  $p_i$  of  $T$  are located compared with a case where the points of the time series are located at equidistant intervals. In the case of  $1 < i < |T|$ , the variable *denominator* presented in Equation (7) is multiplied by two, because in these cases point  $P_i$  has two neighbors, one on each side, but in the case of  $i = 1$  and  $i = |T|$ , point  $p_i$  has only one neighbor and the coefficient of the *denominator* is one. The variable *denominator* tells the average length of an interval of time series  $T$ .

The distance between time series  $T_1$  and  $T_2$  is then calculated by searching for a point  $p_j^{T_2}$  from time series  $T_2$

for each point  $p_i^{T_1}$  of  $T_1$  for which

$$d(p_i^{T_1}, p_j^{T_2}) < d(p_i^{T_1}, p_k^{T_2}),$$

where  $1 \leq j, k \leq |T_2|, j \neq k$ . This point-to-point distance is then multiplied by the weight of the point  $p_i^{T_1}$  and the average of these distances is considered the distance between the time series  $T_1$  and  $T_2$ .

Because weighted distance measure is not symmetric, the distance of patterns  $T_1$  and  $T_2$  was considered as

$$\max(d(T_1, T_2), d(T_2, T_1)) \quad (8)$$

which clearly is symmetric. This way similarity measure satisfies three out of four requirements of metric space (non-negativity, identity of indiscernibles and symmetry). Only triangle inequality does not hold.

In this study, the similarity limit of the distance measure was set to 80.32. The value of the limit was chosen by testing several values and choosing the most suitable from the tested values.

### D. Results

Optimal prototype mined using an evolutionary calculation-based approach was compared with an optimal visually mined prototype. The comparison was made using the data set presented in Section IV-A.

One running signal from nine collected running signals (one from each person) was used as test data in turn, and visually selected prototypes from other signals were considered as population zero for Algorithm 1. Thus, Algorithm 1 gets prototypes that are from the same activity as time series  $T$ , which it is trying to summarize, as *a priori* information, but these prototypes are not performed by the same person as  $T$ . This means the presented algorithm has to find a prototype that describes persons running style without knowing anything about it *a priori*. The purpose of the test was to show that by using the presented evolutionary algorithm it is possible to find a prototype that summarizes the test signal without mining it visually. The quality of the found prototype was compared with that of optimal visually mined prototype.

The test was done by running Algorithm 1 for 50 iteration rounds. This seemed to be an adequate number, because in most cases the results did not seem to get much better during the last iteration rounds. The results presented in Figure 7 and Table II show how the fitness of the best prototype found using Algorithm 1 improves iteration by iteration. The fitness of the optimal visually mined prototype is marked with scatter line. The results show that by using presented evolutionary calculation-based method it is possible to find a prototype that summarizes a time series without visualizing it. Note that in eight cases out of nine the optimal prototype found using the evolutionary calculation-based approach was better than the optimal visually mined prototype, and in all of these cases the found prototype was not a subsignal of the time series that it tried to summarize. This means that likely it is not even possible to find better prototypes by visually mining a signal. In addition, the employed method

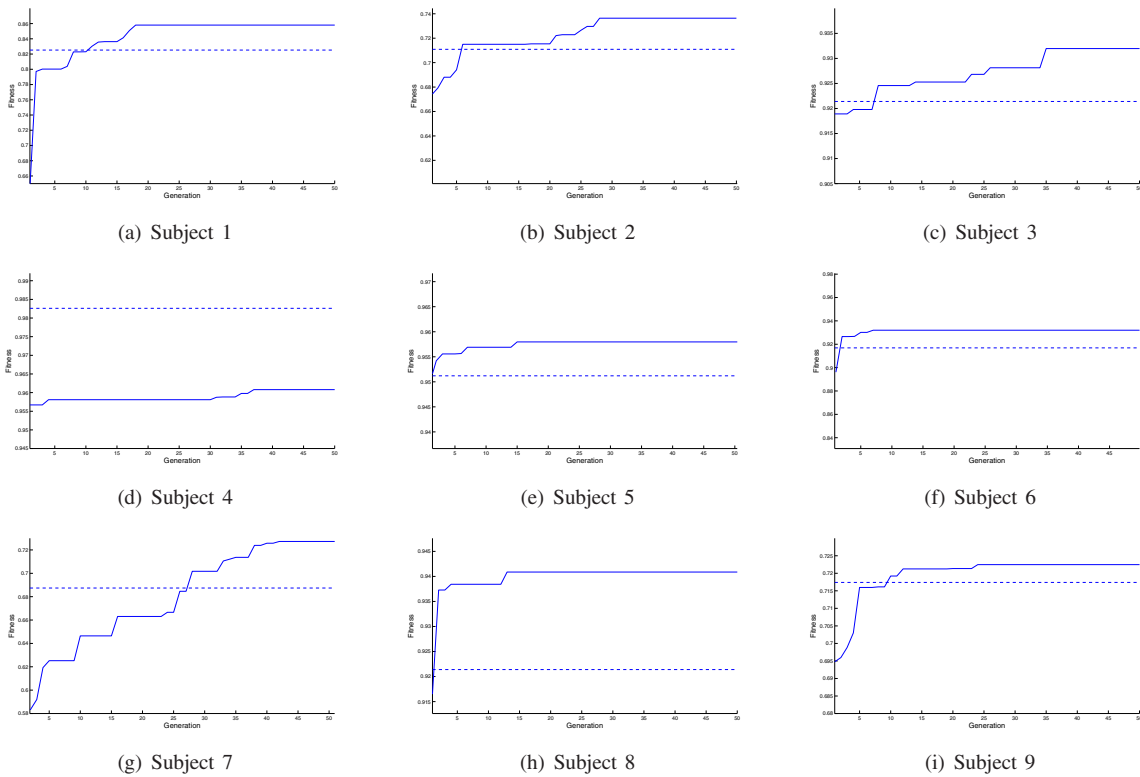


Fig. 7. Fitness of the best found prototype on each iteration round for each person tested. The fitness of the visually found prototype is marked with scatter line.

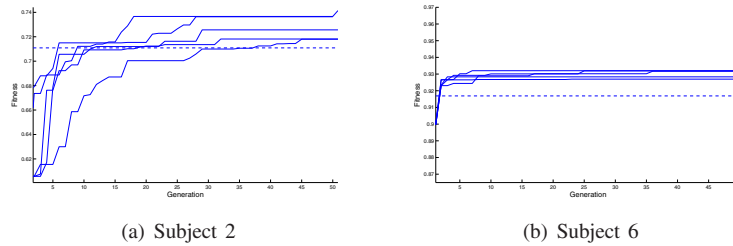


Fig. 8. The proposed algorithm produces prototypes of high quality on each time it is applied.

is also easier and painless to apply, because it does the same thing as visual mining, but automatically. In only one case, see Figure 7(d), the visually mined prototype was better than the prototype found using evolutionary approach. In this particular case the fitness of the visually found prototype was so good that it is probably almost impossible to find a prototype that summarizes a time series better. Possibly, prototype that describes time series better could have been found by choosing generation zero differently or by increasing the number of iteration rounds.

Figure 7 shows how fast the fitness of the best found prototype improves when Algorithm 1 is applied. In most of the cases the results do not improve significantly after 20 iteration rounds. The quality of the results seems to improve especially fast during the first few iteration rounds. Therefore Algorithm 1 is very efficient and by using it descriptive results can be found very quickly.

Figure 8 shows how the presented method succeeded when

it was applied several times. Note that each run has produced a prototype whose fitness is better than the optimal visually mined prototype. This proves that the presented method is reliable and it is not necessary to run it several times in order to find a prototype that summarizes a time series, because on each run it produces a prototype of high quality. Of course, as Figure 8 shows, when the presented algorithm is applied many times it is possible to find better prototypes compared with situation where the algorithm is applied only once.

Table II shows the ratio of the fitnesses of the visually mined and evolutionary calculation approach-based prototypes. The values presented in the table show that almost every time the proposed method manages to find patterns whose fitness is better than the fitness of the visually mined prototype. In many cases the difference is not big, but on the other hand it cannot be much bigger because the superiority of methods is measured in percentages, and in many cases

TABLE II

FITNESS VALUES OF OPTIMAL VISUALLY MINED PROTOTYPE AND OPTIMAL PROTOTYPE MINED USING EVOLUTIONARY CALCULATION FOR EACH TESTED PERSON AND RATIO OF FITNESSES.

1 = FITNESS OF OPTIMAL VISUALLY MINED PROTOTYPE,

2 = FITNESS OF OPTIMAL PROTOTYPE MINED USING EVOLUTIONARY CALCULATION BASED APPROACH.

Name	1	2	Ratio (2/1)
Subject 1	0.8252	0.8580	1.040
Subject 2	0.7110	0.7273	1.023
Subject 3	0.9143	0.9320	1.019
Subject 4	0.9826	0.9611	0.978
Subject 5	0.9512	0.9580	1.007
Subject 6	0.9170	0.9312	1.015
Subject 7	0.6874	0.7190	1.046
Subject 8	0.9214	0.9409	1.021
Subject 9	0.7174	0.7225	1.007

the fitness of visually found prototype is already very good.

Often the main point of using the evolutionary approach is actually not that the fitness of prototype mined using it is better than the fitness of visually found prototype. The main benefit is that patterns that summarize time series can be found without spending time on visual data mining, which can be slow and frustrating. By using evolutionary calculation-based approach, good results, and in most cases better results, can be achieved automatically without even seeing signal.

## V. DISCUSSION

This study presented a method for finding a prototype that describes periodic signal. The results of this study show the supremacy of the method compared with visual mining. By using the presented method it is possible to automatically find prototypes whose fitness is better than the fitness of a visually mined prototype. It is even possible to find prototypes whose fitness is better than the fitness of any subsignal of time series it is trying to summarize, because the found optimal prototype does not have to be a subsignal of the summarized time series.

In order to automate the proposed method more, it is needed to develop a method for automatically selecting candidate population zero.

All signals collected from human activities using wearable sensors contain lots of interference caused by white noise and external factors. Especially external factors, such as disturbance of traffic and drying sweat from the forehead, cause shapes in the acceleration signal that are not typical for running. Also, in this study, small parts at the beginning and end of the collected time series were mainly from null activity and did not include information about running. Therefore, it was not possible to find prototypes whose fitness is exactly 1.0. From the results in Table II it is easy to say which signals contain more disturbances than others. For example the signal of subject 4 is much more free from defects than the running signal of subject 9, because the signal of subject 4 can be summarized more perfectly. The

same can be seen from Figure 1, where parts of the running signals of subjects 4 (Figure 1(a)) and 9 (Figure 1(b)) are presented. It shows that the running patterns of subject 9 differ much more between each other than do those of subject 4. The figure also shows how the different running styles of different persons can be, which is the reason why a prototype that summarizes one persons running signal perfectly does not necessarily summarize another person's activity signal as well.

## ACKNOWLEDGMENT

The authors would like to thank the Finnish Funding Agency for Technology and Innovation and Infotech Oulu for funding this work. Pekka Siirtola would like to thank GETA (The Graduate School in Electronics, Telecommunications and Automation) for financial support.

## REFERENCES

- [1] E. Keogh and M. Pazzani, "An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback," in *Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98)*, R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, Eds. New York City, NY: ACM Press, 1998, pp. 239–241. [Online]. Available: [citeseer.ist.psu.edu/keogh98enhanced.html](http://citeseer.ist.psu.edu/keogh98enhanced.html)
- [2] P. G. Ferreira, P. J. Azevedo, C. G. Silva, and R. M. M. Brito, "Mining approximate motifs in time series," in *Discovery Science*, 2006, pp. 89–101.
- [3] U. Baloglu and M. Kaya, "Top-down motif discovery in biological sequence datasets by genetic algorithm," *Hybrid Information Technology, 2006. ICHIT'06. Vol 2. International Conference on*, vol. 2, pp. 103–107, Nov. 2006.
- [4] F. Liu, J. Tsai, R. Chen, S. Chen, and S. Shih, "Fmga: finding motifs by genetic algorithm," *Bioinformatics and Bioengineering, 2004. BIBE 2004. Proceedings. Fourth IEEE Symposium on*, pp. 459–466, May 2004.
- [5] D. J. Hand, H. Mannila, and P. Smyth, *Principles of data mining*. Cambridge, MA, USA: MIT Press, 2001.
- [6] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Springer, 2003.
- [7] P. Siirtola, P. Laurinen, and J. Röning, "A weighted distance measure for calculating the similarity of sparsely distributed trajectories," in *ICMLA'08: Proceedings of the Seventh International Conference on Machine Learning and Applications*, 2008.