

Clustering of Bi-Dimensional and Heterogeneous Time Series: Application to Social Sciences Data

Rémi Gaudin, Sylvaine Barbier, Nicolas Nicoloyannis, Maks Banens

Abstract—We present an application of bi-dimensional and heterogeneous time series clustering in order to resolve a Social Sciences issue. The dataset is the result of a survey involving more than eight thousand more or less disabled persons. Sociologists need to know if there are in this dataset some homogeneous classes of disabled people according to two attributes: their life quality appreciation and their couple status (i.e. if they have a partner or not). These two attributes are time series so we had to adapt the k-Means clustering algorithm in order to be efficient with this kind of data. For this purpose, we use the Longest Common Subsequence time series distance for its efficiency to manage temporal gaps and we extend it to the bi-dimensional and heterogeneous case. The results of this unsupervised learning process give some pertinent clusters that are ready for sociological analysis.

I. INTRODUCTION

IN data mining research, time series represent an actual challenge due to the unique structure of this kind of data. Most classic data mining algorithms, which were initially conceived for classic (i.e. non temporal) data, do not work well for time series. The need to adapt data mining methods to time series has created a new field of research called temporal data mining [1], [2].

Temporal data mining includes association rules [3], [4], indexing (query by content) [5], [6], feature mining [7], [8], the discovery of recurrent or surprising motifs [9], [10], [11], [12], classification [13], [14], [15], [16] and clustering [17], [18], [19], [20], [21].

Time series clustering is a difficult field where numerous papers propose algorithms that work well with artificial data but they are not efficient in real-world dataset problems [22]. Time series clustering using Hidden Markov Models is proposed in [18], [20]. Some approaches perform clustering using k-Means algorithm with Euclidean distance measure [19], [21]. A time series clustering algorithm that uses k-Means with *Dynamic Time Warping* (DTW) distance measure is

proposed in [17]. Although it is efficient with several artificial datasets, it does not work with real heterogeneous dataset like the one we have to deal with.

Multivariate time series clustering is a more difficult issue where few methods and distances have been proposed. Traditional distances like Weighted Sum SVD [23], Principal Component Analysis similarity factor (S_{PCA}) [24], [25] or Eros [26] are only for numerical data with the same size. Moreover they are often too complex for large dataset and they are basically used for indexing process in databases (e.g. with k-Nearest Neighbors method). In another work, Lee et al propose a method to index sequences of multidimensional points [27]. They extend the ideas presented by Faloutsos et al. in [28] and they use the Euclidean distance.

Some works on indexing moving objects (i.e. bi-dimensional time series) are proposed in order to answer spatial proximity query [29], [30], [31], [32], [33]. Also in [34], [35], an efficient indexing of trajectories is performed by Vlachos *et al.* using *Longest Common Subsequence* (LCSS) distance. For reasons we explain in the next section, this method is relatively suitable for our issue. But Vlachos *et al.* are dealing with numerical data, so we have to make this distance work with heterogeneous data as well. Moreover, this technique, and all the others described above, has not been used for unsupervised clustering with algorithm such k-Means yet.

Our work is based on a survey carried out by the French National Institute for Statistics and Economic Studies [36]. This survey is trying to show how living in couples affects the view disabled people have on their lives. The dataset contains 8403 disabled persons. For each year all along their life, they noted a numeric estimation of the quality of their life (between 1 – the worst – and 3 – the best), and also if they had a partner or not (Fig.1).

Sociologists need to know if there exist in this dataset some homogeneous classes of people according to their “couple status / life-quality estimation” behaviors. So we have a set of 8403 bi-dimensional and heterogeneous time series that we try to classify in an unsupervised way. The aim is that sociologists can work with our

R. Gaudin, S. Barbier and N. Nicoloyannis are with the Laboratoire ERIC, Université Lumière Lyon 2, 5 Av. Pierre Mendès-France 69676 Bron, France (e-mail: {remi.gaudin; sylvaine.barbier; nicolas.nicoloyannis}@univ-lyon2.fr).

M. Banens is with the Centre d’Etudes Démographiques, Institut des Sciences de l’Homme, 16 avenue Berthelot, 69007 Lyon, France (e-mail: maks.banens@univ-lyon2.fr).

partitioning in order to bring out relevant categories of disabled people. We may sum up the difficulties of our issue as follows:

- Bi-dimensional and heterogeneous data: Each disabled person is represented by two time series; one numeric (the life quality estimation) and one symbolic (to have a partner or not).
- Temporal gap: Different persons may have the same bi-dimensional pattern that occurs at different moments in time axis. The process has to match two same patterns despite the potential time axis gap.
- Time series with very different size: Lengths of time series may vary between 2 and 80. The distance measure that we use must be able to manage these differences.
- Unsupervised clustering: The whole process must be automatic. This is more a data exploratory analysis than a machine learning problem.

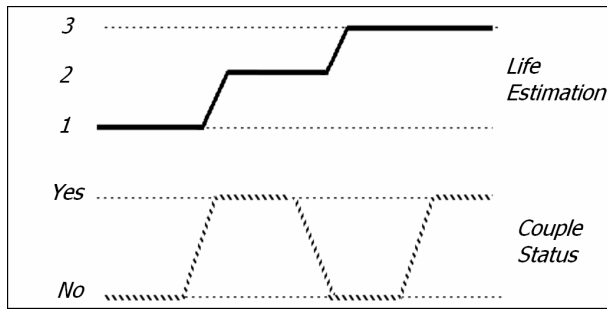


Fig. 1. Example of a bi-dimensional time series representing a disabled person. The first time series characterizes his/her life quality appreciation all along his/her life. This estimation takes a value between 1 (i.e. the worst appreciation) and 3 (i.e. the best appreciation). The second time series denotes the couple status of the disabled person (i.e. if he/she have a partner or not).

So first we have to choose an efficient distance measure. Then we have to develop an algorithm that uses the distance measure and performs the clustering in a satisfying way. In section 2, we present the LCSS distance and explain why it is efficient and how we adapt it for our bi-dimensional and heterogeneous issue. Then in section 3 we present our clustering algorithm, based on k-Means algorithm that we fit for LCSS distance. We present the results obtained with our algorithm on the disabled people dataset in section 4 and we conclude in the last section.

II. LONGEST COMMON SUBSEQUENCE DISTANCE

Euclidean distance is the most widely used distance measure, even for calculating distances between data such as time series, because it is easy to compute and

very fast. The operation consists in matching a given point from a time series with the point from another one that occurs at the same time. The main drawback of Euclidean distance is its inability to manage time axis gap. Two time series with the same shapes that do not occur concurrently on time axis may have a high Euclidean distance. This result is very illogical and it can significantly perturb the clustering [37].

For the particular case of our dataset, we need a distance measure that is able to match some shapes that don't occur at the same time. Indeed, some disabled persons may have similar "couple status / life-quality estimation" patterns at different periods of their life. Among all the measures able to perform time series distance in this way, the *Dynamic Time Warping* (DTW) distance is the most popular. The particularity of DTW is that it compare two time series together by allowing a given point from one time series to be matched with *one or several* points from the other [38], [39]. We choose not to use this distance for two reasons: First, DTW manages only numeric time series. Its adaptation to symbolic data is not obvious and it includes some additional parameters that are difficult to fix. Second, DTW forces all elements of each time series to be matched, even if these elements do not have any relevant meaning. Typically for our dataset is that we have a lot of non relevant periods (e.g. when neither the life-quality appreciation nor the couple status changes for a disabled person).

The *Longest Common Subsequence* (LCSS) distance, like DTW, is able to manage temporal gaps. It matches two time series by computing distance between points that not occur at the same time, without rearranging the sequence of the elements [3], [40], [41], [42]. Whereas in Euclidean and DTW distance *all* elements must be matched, LCSS can keep some elements unmatched by allowing one point of a time series to be matched with *one or zero* point of the other (Fig.2).

In order not to match too distant elements, we may add to LCSS a *warping window* (i.e. a constant δ) that controls how far in time we can go in order to match two points from two different time series. For example, if we set $\delta = 3$, a point that occurs at instant t must only be matched with points from the other time series that occurs at instants $t-3$, $t-2$, $t-1$, t , $t+1$, $t+2$ and $t+3$. In fact, δ is not inevitably a constant and may vary according to time [43], but for simplicity we consider δ to be a constant in this paper. Moreover, we have to set a *spatial window* (i.e. a constant ϵ) as a matching threshold that defines if two points from two different time series can be matched or not. LCSS distance gives

in result a value between 0 (i.e. the two time series are perfectly similar) and 1 (no common points between the two time series).

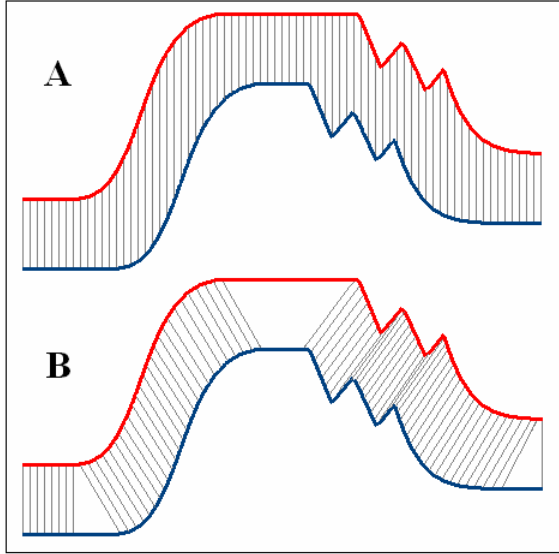


Fig. 2. Computation of the distance between two time series with Euclidean distance (fig. A) and LCSS distance (fig. B).

Originally, LCSS is a one-dimensional distance for numeric data. So we extended its definition to be able to manage time series with two heterogeneous dimensions (one numeric and one symbolic) as follow:

Let A and B be two bi-dimensional time series with size m and n respectively, where $A = \{(a_{x1}, a_{y1}), \dots, (a_{xm}, a_{ym})\}$ and $B = \{(b_{x1}, b_{y1}), \dots, (b_{xn}, b_{yn})\}$. a_{xi} and b_{xi} are the i^{th} value of the numeric time series of A and B respectively. a_{yi} and b_{yi} are the i^{th} value of the symbolic time series of A and B respectively.

Let $Equal(i, j)$ be the matching function between the i^{th} point of A and the j^{th} point of B , where:

$$Equal(i, j) = \begin{cases} \text{True} & \text{if } |a_{xi} - b_{yj}| \leq \epsilon \text{ and } a_{yi} = b_{yj} \\ \text{False} & \text{else} \end{cases} \quad (1)$$

Given the recursive function $Sim(i, j)$ that computes the similarity between the subsequence $A_i = \{(a_{x1}, a_{y1}), \dots, (a_{xi}, a_{yi})\}$ and the subsequence $B_j = \{(b_{x1}, b_{y1}), \dots, (b_{xj}, b_{yj})\}$ as follows:

$$Sim(i, j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ 1 + Sim(i-1, j-1) & \text{if } |i-j| \leq \delta \text{ and } Equal(i, j) = \text{True} \\ \max\{Sim(i-1, j), Sim(i, j-1)\} & \text{otherwise} \end{cases} \quad (2)$$

We can now define our LCSS distance as follows:

$$LCSS(A, B) = 1 - \frac{Sim(m, n)}{\min\{m, n\}} \quad (3)$$

It's important to notice that LCSS is not a metric distance. Therefore it does not necessary respect the triangular inequality $LCSS(A, B) \leq LCSS(A, C) + LCSS(C, B)$.

The recursive definition of our LCSS distance has a non computational complexity and needs a dynamic programming approach [44], [45]. Dynamic programming consists in creating a $m \times n$ matrix. Inside the cell (i, j) of the matrix we store 1 if $Equal(i, j) = \text{True}$, 0 otherwise. Once all the cells are filled we search for the best *warping path*. It is the path beginning in cell $(1, 1)$, finishing in cell (m, n) that maximize the sum of the cells it goes through (Fig. 3).

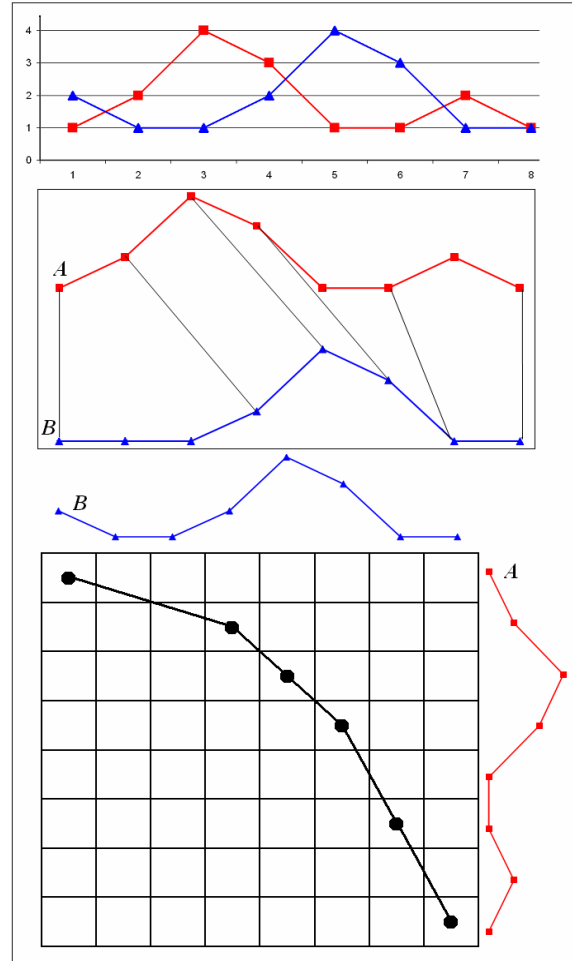


Fig. 3. Example of best warping path between two time series $A = \{1, 2, 4, 3, 1, 1, 2, 1\}$ and $B = \{1, 1, 1, 2, 4, 3, 1, 1\}$. The sequence of cells crossing by the best warping path is $(1, 1), (2, 4), (3, 4), (4, 6), (6, 7), (8, 8)$.

Actually, the non recursive algorithm we use to compute the LCSS distance in an optimal and efficient way does not search directly for the best warping path. It simply obtains the LCSS distance value using a cumulative similarity matrix π built as shown in Table 1 (here for simplicity we only show the algorithm without warping window, but it is trivial to add it with the insertion of the δ condition in each *if* statement).

TABLE 1
NON RECURSIVE ALGORITHM TO COMPUTES THE LCSS DISTANCE
BETWEEN TWO BI-DIMENSIONAL AND HETEROGENEOUS TIME SERIES A
AND B .

```

for  $i = 1$  to  $m$ , do:
{
  for  $j = 1$  to  $n$ , do:
  {
    if ( $(i = 1 \text{ or } j = 1)$  and  $Equal(i, j) = \text{True}$ )
      then  $\pi_{ij} = 1$ 
    if ( $(i = 1 \text{ or } j = 1)$  and  $Equal(i, j) = \text{False}$ )
      then  $\pi_{ij} = 0$ 
    if ( $i > 1$  and  $j > 1$  and  $Equal(i, j) = \text{True}$ )
      then  $\pi_{ij} = \pi_{i-1, j-1} + 1$ 
    if ( $i > 1$  and  $j > 1$  and  $Equal(i, j) = \text{False}$ )
      then  $\pi_{ij} = \max\{\pi_{i-1, j}, \pi_{i, j-1}\}$ 
  }
}
LCSS( $A, B$ ) =  $1 - \frac{\pi_{mn}}{\min\{m, n\}}$ 

```

So the longer step of this algorithm is the completion of the cumulative similarity matrix. This step has a complexity of $O(m \times n)$ ($O(m^2)$ for two time series with the same length). If we add a warping window with size δ , this algorithm allows to compute LCSS in $O(m \times \delta)$ time (with $\delta \ll n$).

III. THE CLUSTERING PROCESS

A. The k -Means algorithm

A classic way to perform clustering is the use of the k -Means algorithm [46]. This approach is very interesting for us because it generates “spherical” clusters (i.e. each cluster can be considered as a hypersphere inside the multidimensional data space. The center of the hypersphere is the fictive mean between all the objects owned by this cluster. The radius is the distance between the fictive mean and the furthest object in the cluster. Because of admitting relocation after each iteration, using k -means clustering allows poor initial partitions to be corrected at a later stage. So when the fictive mean moves, the sphere-shaped structure of the cluster is conserved and it keeps its homogeneity. This characteristic is empirically observed for non metric

distances like LCSS. It permits the creation of homogeneous and proportional clusters that are, for our study, less sensitive to outliers than Hierarchical Clustering clusters.

The intuition behind k -Means approach may be summarized in four steps: First, initialize the k cluster centers (randomly, if necessary). Second, decide the class memberships of the N objects by assigning them to the nearest cluster center. Third, re-estimate the k cluster centers, by assuming the memberships found above are correct. Fourth, if none of the N objects changed membership in the last iteration, exit. Otherwise return to step 2.

To resolve our catalysis clustering problem, this approach has one major drawback: At step 3, we have to re-estimate the k cluster centers. This means computing the average of all the time series for each cluster in the multidimensional data space. This is straightforward with non temporal data (we just have to compute the Euclidean average) but illogical for temporal data like time series. We will resolve this difficulty with by using a variant of this algorithm proposed by Didey in [47].

B. Using k -Means with LCSS distance

Didey proposed in [47] a variant based on the Forgy algorithm [48] that resumes the basic intuition behind all partitional clustering algorithms like k -Means. It creates clusters with only two parameters: the number of clusters noted k and the size of seeds noted c . For the reasons explained in the previous section, the distance measure used by our method is LCSS, so we have adapted this algorithm to make time series clustering work with this distance (Table 2).

Each cluster is characterized by a seed of c time series. Seeds are used to compute distances between time series and clusters as well as distances between each cluster. The c times series of a cluster are those that minimize the *Inertia* function. This function is also used to compute the intra-class variance between all clusters that evaluates the quality of the clustering. The exact complexity of this algorithm can not be determined because it depends on the relative size of each cluster during the iterations. However, for a dataset of N time series, the complexity of step 4 is equal to $O(NkL)$, where k is the number of clusters specified by the user, and L is the duration of the c LCSS calculations at step 3 (i.e. $O(c \times m \times \delta)$). The complexity of steps 7 and 8 depends on clusters relative sizes but is inferior or equal to $O(N^2kL)$. So if we have P iterations until convergence, this algorithm has a complexity inferior or equal to $O(P \times (NkL + N^2kL)) = O(PkL \times (N + N^2))$.

TABLE 2
DIDEY'S K-MEANS GENERALIZATION WITH LCSS DISTANCE

1	Let X be a set of N time series that we aim to split in k clusters, where $X = \{x_1, \dots, x_n\}$.
2	Let S be a set of k seeds, where $S = \{S_1, \dots, S_k\}$. Each seed S_j is composed of c time series chosen among the initial set X (randomly if necessary). One time series can not belong to more than one seed.
3	Given $L(c, S_j)$, the distance between the time series x_i and the seed S_j as follows: $L(x_i, S_j) = \frac{1}{c} \sum_{y \in S_j} LCSS(x_i, y)$
4	For $i = 1, \dots, N$ do: For $j = 1, \dots, k$ do: Compute $L(x_i, S_j)$
5	Assign to each time series x_i its nearest seed (i.e. the seed S_j that minimize $L(x_i, S_j)$).
6	Let C be a set of k clusters, where $C = \{C_1, \dots, C_k\}$. Each cluster C_j is made of all time series that have S_j as nearest seed.
7	Redefine a new set of seeds $S' = \{S'_1, \dots, S'_k\}$. Each new seed S'_j is made of the c time series x_i from C_j that minimize: $Inertia(x_i, C_j) = \sum_{y \in C_j} LCSS(x_i, y)$
8	To estimate the clustering quality, calculate the intra-class variance $Var(C)$ as follows: $Var(C) = \frac{1}{n} \sum_{j=1}^k \sum_{x_i \in C_j} Inertia(x_i, C_j)$ If the value of $Var(C)$ does not decrease between the iteration p and the iteration $p+1$ (or decrease less than an arbitrary threshold), then stop the process. Else restart a new iteration at step 4 with $S = S'$.

It is obvious that the final time series clustering depends on the seeds initialization. Ideally, each seed should be initialized only with the time series that belong to the same cluster, but if we do not have any *a priori* knowledge about the dataset (as it is usually the case in unsupervised knowledge data discovery), we have to initialize the seeds at random. For a dataset with k time series clusters (with the same number of time series for each cluster), the probability to have a perfect initialization at random is approximately equal to $k! / k^k$ (for example if $k = 6$ then the probability is equal to 0.015, i.e. 1 in 65 to have a perfect initialization). Here we use intra-class variance to evaluate the quality of the seeds initialization: if a seed initialization does not give an intra-class variance inferior to a threshold, then another initialization is tried.

Because of the nature of our data, we do not use a temporal window because similar patterns that we search in two disabled persons can occur anywhere in

time axis. So it leaves us only with three parameters: the number of clusters k , the size of seeds c and the spatial window ε for LCSS. Only the numeric variable of our bi-dimensional time series needs the ε parameter. This parameter must be fixed by user according to the maximum difference that he considers that two persons have the same life estimation. We notice that the value of c does not influence the intra-class variance defined in our algorithm (contrary to δ and k). So we can consider that the optimal value of c minimizes this variance. We can not use the same method to find the optimal value of k because this parameter influences the intra-class variance final value (the more k increases, the more the intra-class variance decreases). This limitation can be minimized by attempting all values of k within a large range.

IV. CLUSTERING OF THE DISABLED PEOPLE DATASET

We apply our algorithm on the disabled people dataset. We split the data into two parts according to the sex (i.e. one part with 4617 women and one part with 3786 men). Sociologists need to know if the “direct link” hypothesis (i.e. to have a partner implies an increase of life-quality estimation for disabled people) is valid or not. Results of our process give, for each sex, surprising clusters that contradict this expectation. In spite of the very large multiplicity of patterns, we are able to bring out some homogeneous classes, in particular if we cluster the dataset with $k = 4$ (Figure. 4).

The first surprising fact we notice is the quasi-perfect symmetry between the two sexes. The difference between the proportional cluster sizes for the two sexes is always inferior or equal to 1 %. That means that living in couples affects in the same way the view disabled men and women have on their lives.

The other surprising fact is the frequency of disabled people that have no variation of their life-quality estimation in spite of couple-status changes (72 % for men and 73 % for women). The second largest cluster is composed of people without a couple-status change. People that directly relate life quality with couple status (i.e. to find a partner implies a life quality increase) make up a third cluster. With a frequency of 6 % for both sexes, this cluster is surprisingly small. In the same way, the cluster with people having an opposite relation between the two variables is relatively insignificant (3 % for each sex).

V. CONCLUSION

In this paper we presented an exploratory analysis on a survey on 8403 disabled persons. The dataset needs a methodology that is able to manage bi-dimensional,

heterogeneous data, with different sizes and temporal gaps.

As a conclusion and on the contrary of what sociologists expected, results permit to consider that living in couples (or not) is not a determinant variable to explain increases (or decreases) in the life quality estimation of disabled people. This conclusion may be an important decision making factor for future assistance programs towards disabled people.

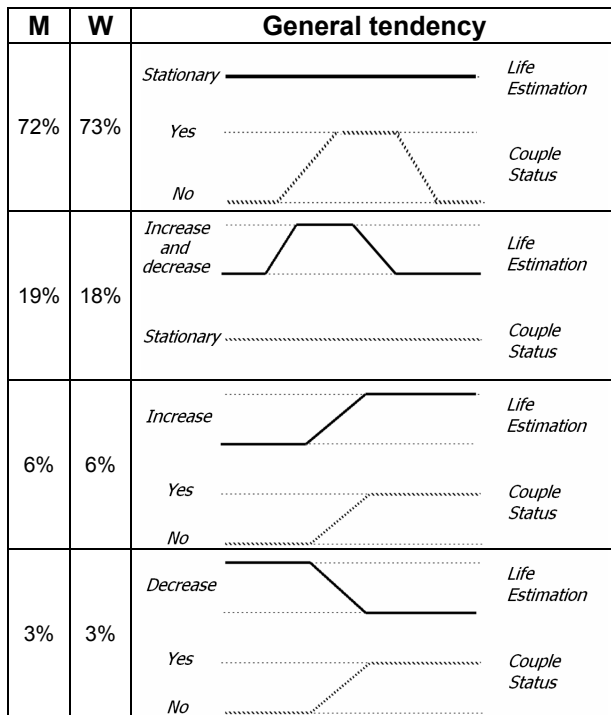


Fig. 4. The clustering result of the disabled people dataset (with $k = 4$). The first and the second columns give the proportional size of each cluster for disabled men (M) and women (W) respectively. Because of the very large multiplicity of patterns, we only show here for each cluster the main pattern that is the most representative of the general tendency of the cluster.

REFERENCES

[1] W. Lin, M. A. Orgun, and G. J. Williams, "An overview of temporal data mining," presented at The Australasian Data Mining Workshop, Macquarie University and CSIRO Data Mining, 2002.

[2] C. M. Antunes and A. L. Oliveira, "Temporal data mining: an overview," presented at the Workshop on Temporal Data Mining, at the 7th International Conference on Knowledge Discovery and Data Mining (KDD'01), San Francisco, CA, 2001.

[3] G. Das, D. Gunopulos, and H. Mannila, "Finding similar time series," presented at Principles of Data Mining and Knowledge Discovery, 1st European Symposium, Trondheim, Norway, 1997.

[4] G. Das, K. Lin, H. Mannila, G. Renganathan, and P. Smyth, "Rule discovery from time series," presented at the 4th International Conference on Knowledge Discovery and Data Mining, New York, NY, 1998.

[5] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowledge and Information Systems: An International Journal (KAIS)*, 2004.

[6] B. K. Yi, H. Jagadish, and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping," presented at IEEE International Conference on Data Engineering, 1998.

[7] D. Kudenko and H. Hirsh, "Feature generation for sequence categorization," presented at the 15th National Conference in Artificial Intelligence (AAAI'98), Menlo Park, California, USA, 1998.

[8] N. Lesh, M. J. Zaki, and M. Ogihara, "Scalable feature mining for sequential data," *IEEE Intelligent Systems*, vol. 15, pp. 48-56, 2000.

[9] J. Buhler and M. Tompa, "Finding motifs using random projections," *Journal of Computational Biology*, vol. 9, pp. 225-242, 2002.

[10] B. Chiu, E. Keogh, and S. Lonardi, "Probabilistic discovery of time series motifs," presented at the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 2003.

[11] E. Keogh, S. Lonardi, and W. Chiu, "Finding surprising patterns in a time series database in linear time and space," presented at the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, 2002.

[12] J. Lin, E. Keogh, P. Patel, and S. Lonardi, "Finding motifs in time series," presented at the 2nd Workshop on Temporal Data Mining, at the 8th International Conference on Knowledge Discovery and Data Mining (KDD'02), Edmonton, Alberta, Canada, 2002.

[13] J. C. Chappelier, M. Gori, and A. Grumbach, "Time in connexionist models," in *Sequence Learning: Paradigms, Algorithms and Applications*, R. S. a. G. L. Giles, Ed.: Springer-Verlag, 2000, pp. 105-134.

[14] D. L. James and R. Miikkulainen, "A self-organizing feature map for sequences," *Advances in Neural Processing Systems*, vol. 7, pp. 577-584, 1995.

[15] E. Keogh and M. Pazzani, "An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback," presented at the 4th International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 1998.

[16] P. Somervuo and T. Kohonen, "Self-organizing maps and learning vector quantization for feature sequences," *Neural Processing Letters*, vol. 10, pp. 151-159, 1999.

[17] R. Gaudin and N. Nicoloyannis, "Apprentissage non supervisé de séries temporelles à l'aide des k-means et d'une nouvelle méthode d'agrégation de séries," presented at 5èmes Journées d'Extraction et de Gestion des Connaissances (EGC'05), Paris, France, 2005.

[18] F. R. Lin, L. S. Hsieh, and S. M. Pan, "Learning clinical pathway patterns by hidden markov model," presented at the 38th Annual Hawaii International Conference on System Sciences (HICSS'05), 2005.

[19] J. Lin, M. Vlachos, E. Keogh, and D. Gunopulos, "Iterative incremental clustering of time series," presented at the IX Conference on Extending Database Technology (EDBT 2004)², Crete, Greece, 2004.

[20] T. Oates, L. Firoiu, and P. R. Cohen, "Clustering time series with hidden markov models and dynamic time warping," presented at IJCAI-99 Workshop on Sequence Learning, 1999.

[21] M. Vlachos, J. Lin, E. Keogh, and D. Gunopulos, "A Wavelet-Based Anytime Algorithm for K-Means Clustering of Time-Series," presented at Workshop on Clustering High-Dimensionality Data and its Applications, SIAM Datamining, San Francisco, CA, USA, 2003.

[22] E. Keogh and S. Kasetty, "On the need for time series data mining benchmarks: A survey and empirical demonstration," the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002.

- [23] C. Shahabi and D. Yan, "Real-time pattern isolation and recognition over immersive sensor data streams," presented at the 9th International Conference On Multi-Media Modeling, 2003.
- [24] W. Krzanowski, "Between-groups comparison of principal components," *JASA*, vol. 74, 1979.
- [25] D. Singhal and A. Seborg, "Clustering of multivariate time-series data," presented at the American Control Conference, 2002.
- [26] K. Yang and C. Shahabi, "A PCA-based similarity measure for multivariate time series," presented at the Second ACM International Workshop on Multimedia Databases, 2004.
- [27] S.-L. Lee, S.-J. Chun, D.-H. Kim, J.-H. Lee, and C.-W. Chung, "Similarity Search for Multidimensional Data Sequences," presented at ICDE, 2000.
- [28] C. Faloutsos, M. Ranganathan, and I. Manolopoulos, "Fast Subsequence Matching in Time Series Databases," presented at ACM SIGMOD, 1994.
- [29] P. K. Agarwal, L. Arge, and J. Erickson, "Indexing moving points," presented at the 19th ACM Symposium on Principles of Database Systems (PODS), 2000.
- [30] G. Kollios, D. Gunopulos, and V. Tsotras, "On Indexing Mobile Objects," presented at the 18th ACM Symposium on Principles of Database Systems (PODS), 1999.
- [31] D. Pfoser, C. Jensen, and Y. Theodoridis, "Novel Approaches in Query Processing for Moving Objects," presented at VLDB, Cairo Egypt, 2000.
- [32] Y. Qu, C. Wang, and X. Wang, "Supporting Fast Search in Time Series for Movement Patterns in Multiple Scales," presented at the ACM CIKM, 1998.
- [33] S. Saltenis, C. Jensen, S. Leutenegger, and M. A. Lopez, "Indexing the Positions of Continuously Moving Objects," presented at the ACM SIGMOD, 2000.
- [34] M. Vlachos, D. Gunopulos, and G. Kollios, "Robust similarity measures for mobile object trajectories," presented at 13th Database and Expert Systems Applications (DEXA), 5th International Workshop on Mobility in Databases and Distributed Systems (MDDS), Aix-en-Provence, France, 2002.
- [35] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," presented at the 18th International Conference on Data Engineering (ICDE'02), San Jose, CA, USA, 2002.
- [36] C. Goillot and P. Morniche, "Société n°22, Enquêtes Handicaps-Incapacités-Dépendance," INSEE 2003.
- [37] E. Keogh and M. Pazzani, "Scaling Up Dynamic Time Warping for Data Mining Applications," presented at the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, USA, 2000.
- [38] D. J. Berndt and J. Clifford, "Using Dynamic Time Warping to Find Patterns in Time Series," presented at KDD Workshop, 1994.
- [39] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimisation for spoken word recognition," *IEEE Trans. Acoustics, Speech and Signal Processing, ASSP*, vol. 26, pp. 43-49, 1978.
- [40] R. Agrawal, K. Lin, H. S. Sawhney, and K. Shim, "Fast Similarity Search in the Presence of Noise, Scaling and Translation in Time-Series Databases," presented at VLDB, 1995.
- [41] B. Bollobas, G. Das, D. Gunopulos, and H. Mannila, "Time-Series Similarity Problems and Well-Separated Geometric Sets," presented at the 13th SCG, Nice, France, 1997.
- [42] T. Bozkaya, N. Yazdani, and M. Ozsoyoglu, "Matching and Indexing Sequences of Different Lengths," presented at the CIKM, Las Vegas, USA, 1997.
- [43] C. A. Ratanamahatana and E. Keogh, "Making Time-series Classification More Accurate Using Learned Constraints," presented at SIAM International Conference on Data Mining (SDM '04), Lake Buena Vista, Florida, USA, 2004.
- [44] R. Bellman, *Dynamic Programming*. New Jersey: Princeton University Press, 1957.
- [45] D. J. Berndt and J. Clifford, "Finding Patterns in Time Series: A Dynamic Programming Approach," presented at Advances in Knowledge Discovery and Data Mining, 1996.
- [46] J. McQueen, "Some Methods for Classification and Analysis of Multivariate Observation," presented at the 5th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, CA, USA, 1967.
- [47] E. Diday, "The dynamic clusters method in non hierarchical clustering," *International Journal of Computer Sciences*, vol. 2, 1973.
- [48] Forgy, "Cluster analysis of multivariate data: efficiency versus interpretability of classification," *Biometrics*, vol. 21, pp. 768-769, 1965.