Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Singapore
Tokyo

Djamel A. Zighed   Jan Komorowski
Jan Żytkow (Eds.)

Principles of Data Mining and Knowledge Discovery

4th European Conference, PKDD 2000
Lyon, France, September 13-16, 2000
Proceedings

[illegible]

Springer

[illegible]

References

1. Michael J. A. Berry, Gordon Linoff. Data Mining Techniques For marketing, Sales and Customer Support. John Willey & Sons, Inc, 1996.
2. Rajesh N. Dave. "Validating fuzzy partitions obtained through c-shells clustering", *Pattern Recognition Letters*, Vol .17, pp613-623, 1996
3. J. C. Dunn. "Well separated clusters and optimal fuzzy partitions", *J. Cybern.* Vol.4, pp. 95-104, 1974.
4. Martin Ester, Hans-Peter Kriegel, Jorg Sander, Xiaowei Xu. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", *Proceedings of 2nd Int. Conf. On Knowledge Discovery and Data Mining*, Portland, OR, pp. 226-231, 1996.
5. Usama Fayyad, Ramasamy Uthurusamy. "Data Mining and Knowledge Discovery in Databases", *Communications of the ACM*. Vol.39, No11, November 1996.
6. Usama M. Fayyad, Gregory Piatesky-Shapiro, Padhraic Smuth and Ramasamy Uthurusamy. Advances in Knowledge Discovery and Data Mining. AAAI Press 1996
7. Gath, B. Geva. "Unsupervised Optimal Fuzzy Clustering". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 11, No7, July 1989.
8. Sudipto Guha, Rajeev Rastogi, Kyueseok Shim. "CURE: An Efficient Clustering Algorithm for Large Databases", *Published in the Proceedings of the ACM SIGMOD Conference*, 1998.
9. Alexander Hinneburg, Daniel Keim. "An Efficient Approach to Clustering in Large Multimedia Databases with Noise". *Proceeding of KDD '98*, 1998.
10. Zhexue Huang. "A Fast Clustering Algorithm to Cluster very Large Categorical Data Sets in Data Mining", *DMKD*, 1997
11. Ramze Rezaee, B.P.F. Lelieveldt, J.H.C Reiber. "A new cluster validity index for the fuzzy c-mean", *Pattern Recognition Letters*, 19, pp237-246, 1998.
12. Padhraic Smyth. "Clustering using Monte Carlo Cross-Validation". KDD 1996, 126-133.
13. C. Sheikholeslami, S. Chatterjee, A. Zhang. "WaveCluster: A-MultiResolution Clustering Approach for Very Large Spatial Database". *Proceedings of 24th VLDB Conference, New York*, USA, 1998.
14. S. Theodoridis, K. Koutroubas. *Pattern recognition*, Academic Press, 1999
15. M. Vazirgiannis, "A classification and relationship extraction scheme for relational databases based on fuzzy logic", in the proceedings of the *Pacific-Asian Knowledge Discovery & Data Mining '98 Conference*, Melbourne, Australia.
16. M. Vazirgiannis, M. Halkidi. "Uncertainty handling in the datamining process with fuzzy logic", in the proceedings of the IEEE-FUZZ conference, San Antonio, Texas, May, 2000.
17. Xunali Lisa Xie, Genardo Beni. "A Validity measure for Fuzzy Clustering", *IEEE Transactions on Pattern Analysis and machine Intelligence*, Vol13, No4, August 1991.
18. A.K Jain, M.N. Murty, P.J. Flyn. "Data Clustering: A Review", ACM Computing Surveys, Vol. 31, No3, September 1999.
19. Fisher,R.A. Machine readable .names file for MLC++ library. July, 1988

Algorithm for Matching Sets of Time Series

Iztok Savnik[1], Georg Lausen[1], Hans-Peter Kahle[2], Heinrich Spiecker[2], and Sebastian Hein[2]

[1] Freiburg University
Institute for Computer Science
D-79085 Freiburg im Breisgau, Germany
{savnik,lausen}@informatik.uni-freiburg.de
[2] Freiburg University
Institute for Forest Growth
D-79085 Freiburg im Breisgau, Germany
{kahle,spiecker,hein}@uni-freiburg.de

Abstract. Time series are time-stamped sequences of values which represent a parameter of the observed processes in subsequent time points. Given a set of time series describing a set of similar processes, the model of the behavior of processes is constructed as a range of classification trees which describe the characteristics of each particular time point in series. An algorithm for matching a sequence of values with the model is used for searching common patterns in the sets of time series, and for predicting the starting time points of undated time series. The algorithm was developed and analyzed in the frame of the study of tree-ring time series. The implementation and the empirical analysis of the algorithm on the tree-ring time series are presented.

Keywords: Data mining, knowledge discovery, time-series, matching, and pattern recognition.

1 Introduction

A set of processes can be observed by studying the values of a particular parameter of processes measured in the subsequent time points. The sequence of values of the parameter for a particular process is called a *time-series*. The behavior of the processes recorded in time series can be revealed by investigating the characteristic features of the time series.

A considerable research effort has been directed recently to the development of methods for matching characteristic patterns in time series databases [2,7, 1,8,5]. The methods vary in the representation techniques for time series, the algorithms for measuring similarity between the time series, and in the search mechanisms used for mining the patterns. The problem which is related to matching subsequences in time series databases is the discovery of common patterns from a set of time series [13,9]. This problem has to our knowledge received less attention by KDD community, although it is of importance in practice for identifying the common characteristics of similar processes in applications such as monitoring and diagnosing complex systems, stock market data analysis, and medical diagnosis.

In this paper we address the problem of finding the common characteristics of the time series which describe a set of similar processes. Our main contribution is the proposal of a new algorithm for matching sequences of values with a set of time series. The problem is seen as a machine learning problem [10]. Given a description of a domain in the form of a set of training instances (time series), a theory is constructed which describes the characteristic properties of the domain. The representation of the theory is based on the "local" properties of time points — the time points are described by the characteristics of values which occurred close to the described time point. The algorithm for matching a sequence of values with the sets of training time series is realized as the procedure for the characterization of new instances by means of the computed theory. We show in the paper that the algorithm can be employed for mining the common patterns in a set of time series, and for determining the matching time points (dating) of the undated sequence of values.

The rest of the paper is organized as follows. The following section presents the matching algorithm. First, the problem of matching a sequence of values with the set of time series is defined formally. Section 2.1 presents the procedure for the construction of domain theory from the sets of time series. Further, the algorithm for matching sequences of values against the theory is detailed in Section 2.2. The experiments with the matching algorithm on tree-ring time series are described in Section 3. The results of the experiments show that the algorithm can be effectively used for dating tree-ring time series, a method which is important for revealing the past climatic and other environmental conditions, and for identifying the characteristic patterns in the behavior of a set of trees. Section 4 gives an overview of the related work and its relations to the presented work. Finally, the concluding remarks are given in Section 5.

2 Algorithm

A set of time series representing the values of a parameter of similar processes is called a *domain*. The method for matching a sequence of values with a set of time series is divided into two parts. First, the theory of domain is constructed, and, second, the matching algorithm is used to determine the time points in the time period of domain where an input sequence of values matches the domain with a given precision. Let us now define the terminology used in the paper and formally define the problem.

A time series $s = (v_1, \dots, v_n)$ is a sequence of real numbers v_i which describe a property of entities from the domain in the subsequent time points. The time interval between the subsequent events is constant, that is, $t(v_{i+1}) = t(v_i) + \delta t$ where δt depends on the particular application domain. The application domain is described by a set of time series $\mathcal{D} = \{s_1, \dots, s_m\}$. Each time series $s_i \in \mathcal{D}$ and $s_i = (v_{i1}, \dots, v_{in})$ has n values with time points $t(v_{ij}) = t_j$.

The problem of matching sequences of values with the set of time series is defined as follows. Let $\mathcal{D}$ be a domain as defined above, and let $s_u = (u_1, \dots, u_r)$ be a sequence of values where the time points $t(u_i)$ are not known. The matching procedure must find the time point $t_j \in [t_1..t_n]$ such that the sequence s_u matches with the domain $\mathcal{D}$ starting at the time point t_j. Let us now detail the construction of the theory $\mathcal{T}$ describing the features of the domain, and the algorithm for matching a sequence of values s_u with $\mathcal{D}$ by the use of theory $\mathcal{T}$.

2.1 Construction of Domain Theory

The application domain $\mathcal{D} = \{s_1, \dots, s_m\}$ can be seen through the sequence of time points $t_1, \dots, t_n$ for which the values of each time series s_i are specified. Given a time point $t_j \in [t_1..t_n]$, there exist one value $s_i[t_j]$ (or, v_{ij}) for each time series $s_i \in \mathcal{D}$ at time point t_j. The set of values of time series in a particular time point is denoted as $\mathcal{D}[t_j]$. Formally, $\mathcal{D}[t_j] = \{s_i[t_j] | \; i \in [1..m] \wedge s_i \in \mathcal{D}\}$. Further, the values of time series from $\mathcal{D}$ in subsequent time points $t_j, \dots, t_{j+w}$ is the projection of $\mathcal{D}$ on the given sequence of time points resulting the subsequences of time series from $\mathcal{D}$. The projection of $\mathcal{D}$ on the time points $t_j \dots t_{j+w}$ is defined as $\mathcal{D}[t_j..t_{j+w}] = \{(s_i[t_j], \dots, s_i[t_{j+w}]) | \; i \in [1..m] \wedge s_i \in \mathcal{D}\}$. The projection $\mathcal{D}[t_j..t_{j+w}]$ is called a *window* of $\mathcal{D}$ and w is the size of window.

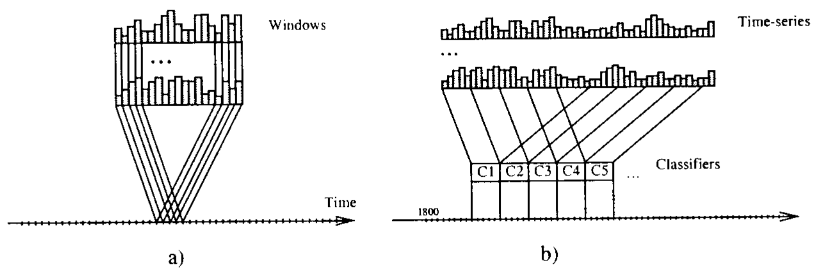

Fig. 1. Rules: a) learning cases, b) construction of classifiers.

The main idea used for the construction of the domain theory is to describe the characteristics of each particular time point in the domain by the properties of the values which are local with regards to the described time points. In other words, a time point t_j can be described by the characteristics of values which appeared some time before t_j, the values which appeared at t_j and some time after t_j. The values that are close to a time point t_j include the values from the window $\mathcal{D}[t_{j-\lfloor w/2 \rfloor}..t_{j+\lfloor w/2 \rfloor}]$ where w is the size of window. The windows of time series and the corresponding central time points are illustrated in Figure 1a.

Each window projected from the domain $\mathcal{D}$ is extended by adding to each subsequence from the window the time point of the central value of subsequence — such windows will be called *dated windows* $\mathcal{D}^*$. The dated windows are used as the basis for the construction of the classifiers for the time points of central values. More particularly, a range of dated windows is used as the input dataset of a program for the construction of classifiers. Formally, a *dataset* is defined as $\bigcup_{j\in[b..(b+d)]}\mathcal{D}^*[t_{j-\lfloor w/2\rfloor}..t_{j+\lfloor w/2\rfloor}]$ where w is the size of window, d is the number of windows in the range, and b is the index of the central time point of first window. The construction of the range of dated windows is illustrated by Figure 1a.

The time series from $\mathcal{D}$ can now be split into the overlapping partitions which are used for the definition of datasets. Splitting of the set of time series from the domain into partitions is illustrated by Figure 1b. Each dataset is used as the input of the program for the construction of a classifier. Note that the classifiers are in Figure 1b denoted by labels C_i. Each particular classifier describes the properties of a range of time points. The complete set of time points of the domain $\mathcal{D}$ is split into non-overlapping subsets such that each of them is covered by one of the classifiers.

2.2 Matching Algorithm

Let $\mathcal{D} = (s_1, \ldots, s_m)$ be a domain such that the time series s_i are defined for the time points $t_1, \ldots, t_n$. Further, let $C_1, \ldots, C_e$ be the set of classifiers constructed with windows of length w. The number of classifiers in the set is $e = \lfloor n/c \rfloor$ where c is the number of predicted classes by one classifier. Finally, let $s_u = (u_1, \ldots, u_r)$ be a sequence of values. The task of the matching algorithm is to find the time point $t_j \in [t_1..t_n]$ of the first value u_1 of the sequence s_u such that the sequence s_u matches with $\mathcal{D}$ starting at t_j. The precision of matching is specified by the set of parameters which will be presented shortly.

The matching algorithm extracts all possible windows of length w from the input sequence s_u. Each of the extracted windows is classified by each of the available classifiers. If the accuracy of the prediction is better than the predefined constant T_g then the prediction is called a *guess*. After a guess is obtained, it is verified using all other windows from s_u. The number of predictions which match the initial guess and the collective probability of matching is computed. A prediction which confirms with the initial guess will be in the sequel called a *hit*. In the case that the number of hits is higher than the predefined constant T_h, and at the same time, the collective probability of the match is higher than the predefined constant T_m, then we say that the sequence s_u matches the domain $\mathcal{D}$ in the predicted time point. The algorithm which implements the above sketched procedure is presented as follows.

Algorithm 1.
```
Input: Sequence of values $s_u = (u_1, \ldots, u_r)$ and a range of
       classifiers $R = (C_1, \ldots, C_e)$.
Output: Predictions $(g, p_g)$ for the starting time points of $s_u$.
Method:
1. foreach ( window $o \subset s_u$ ) do
2.     foreach ( classifier $C_i \in R$ ) do
3.         $(g, p_g)$ = predict($o, C_i$);
4.         if ( $p_g > T_g$ ) then
5.             $(h, p_h)$ = compute_hits($R, s_u, g$);
6.             if ( $h > T_h \wedge p_h > T_m$ ) then
7.                 print_match($g, p_g$);
8.         fi;
9. od;
```

We will now comment the above algorithm. The first **foreach** loop at line 1 iterates through all windows o extracted from s_u. Each of the windows o is classified by each of the classifiers from R using the function `predict`(o, C_i) in line 3. The result of a prediction is a pair (g, p_g) where g is predicted time point of the first value of s_u and p_g is the probability that g is correct. In the case that p_g is greater than the threshold T_g (line 4) than g is a guess and the number of confirmations of the guess (hits) is computed in the line 5. This is done by classifying all the remaining windows of s_u. The procedure for the computation of the confirmations of guess is detailed by Algorithm 2. The threshold T_h defines the required number of confirmations of guess g and the threshold T_m defines the required collective probability for matching.

Algorithm 2.
```
Input: A set of classifiers $R = (C_1, \ldots, C_e)$, a sequence of values
       $s_u = (u_1, \ldots, u_r)$, and a guess $g$.
Output: Number of hits $h$ and collective probability $p_h$.
Method:
1. function compute_hits($R, s_u, g$);
2. begin
3.     hits = 0; sump = 0;
4.     foreach ( window $o \subset s_u$ ) do
5.         $(p, p_p)$ = classify($o, R, g$);
6.         if ( $g = p$ ) then
7.             hits = hits + 1;
8.             sump = sump + $p_p$;
9.         fi;
10.    od;
11.    return( hits, sump/$(r - w + 1)$ );
12.end;
```

The algorithm 2 verifies the guess g using all possible windows from s_u (line 4). In line 5 each window is classified using the function `classify`(o, R, g). The classifiers which are used for the computation of the prediction are selected with respect to the guess g — we suppose that g represents the correct matching point and fix the position of s_u within the time interval of the domain $\mathcal{D}$ with respect to the time point of the guess g. The result of the function `classify` is the time point p of the first value of s_u and the probability p_p that p is correct. Finally, the number of hits and the sum of matching probabilities are augmented in lines 6-9 if prediction p agrees with the guess g. The collective probability returned by the function is the average matching probability, that is, **sump** divided with the number of classifications $(r - w + 1)$.

3 Analysis

The algorithm for matching sets of time series has been developed and analyzed in the frame of the study of tree-ring time series. The algorithm was implemented using the C5.0 knowledge discovery tools [15] and the `Perl` programming language. In this section we present the application domain, the implementation of matching algorithm, and the analysis of matching algorithm on the tree-ring time series. Two applications of the matching algorithm are analyzed: dating of sequences, and pattern discovery. Let us first present the tree-ring application domain, and the procedure for the acquisition of tree-ring data.

3.1 Tree-Ring Domain

The tree-ring time series are the sequences of values which represent the annual increments of the tree trunks measured on cross sections obtained after the tree is cut. The cross sections are taken from a stem height of 1.3 m. The annual radial increments are measured along eight equidistant measurement radii with a computer assisted image analysis system. The series used are the quadratic means of the eight measurements.

The tree-ring time series used in the experiments presented in the following sections are obtained from 50 European beech (Fagus sylvatica) sample trees from two geographical areas in Germany. The initial tree-ring measurement series have been detrended in order to eliminate individual tree specific variations in the time series. The result of this standardization procedure are normalized time series which are stationary in their mean and variance.

Tree rings reflect various influences of the environment on the growth of trees such as, for example, the climatic changes, the characteristics of the ground, or the events such as floods. A field of science which uses tree rings to analyze temporal and spatial patterns of processes is *Dendrochronology* [4]. The most widely used method for the study of tree-rings in Dendrochronology is called cross-dating [11]. The method has been effectively used to reconstruct the past climate for more thousands years.

3.2 Construction of Domain Theory

The domain which includes 50 tree-ring time series was split into two parts: the training time series (80%) and the test time series (20%). The analysis presented in this section is done on the set of training time series. The domain theory was constructed from the training time series as presented in Section 2.1. The test data was used for the experiments which are presented in the following two sections.

The decision trees were constructed using the C5.0 data mining tool. The rules extracted from the decision trees are used as the domain theory. The classifiers were built using the boosting technique [15]. Further, the cross-validation method was used for the estimation of the classification accuracy of the domain

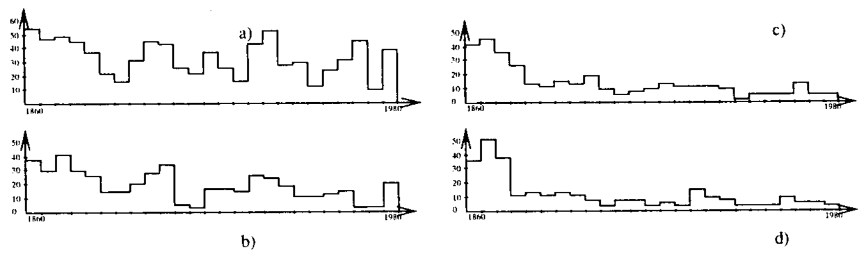

Fig. 2. Error rates of classifiers (in percents) constructed using the windows of size: a) 5, b) 10, c) 20, and d) 30.

theory. The results of the experiments with the construction of the domain theories are presented in Figure 2. In most cases 1-4 rules were computed for each of the time points. A typical example of a set of rules describing a time point 1900 is as follows.

$irp7 > 3.086$, $irp9 > 2.693$, $irp10 > 3.071$, $irp14 > 3.526$ `-> class 1900 [0.965]`
$irm10 > 2.861$, $irp10 <= 3.071$, $irp13 <= 3.267$ `-> class 1900 [0.916]`
$irm5 <= 2.835$, $irm3 <= 2.979$, $irp10 > 3.071$ `-> class 1900 [0.748]`

The attributes in rules are named as follows. An attribute named $irmX$ represents the value of an instance in a time point which occurred X time points before the central time point. Similarly, an attribute named $irpX$ represents the value of an instance in a time point which occurred X time points after the central time point. The classification accuracy of a rule is specified in the square brackets at the end of rule. A rule is typically based on 2 to 4 attributes.

3.3 Dating

The problem of dating sequences of values is defined as follows. Given a set of time series and an undated sequence of values, the time point t of the first value of undated sequence has to be found such that the undated sequence matches the set of time series at time point t with a given precision.

The experiments with dating were done by using 40 time series for the construction of the theories, and 10 time series as the basis for the generation of test sequences. The length of time series from the domain is from 100 to 250. For the construction of the theories we used 110 subsequent time points (years) which were defined for all time series. Further, four different theories were built for the experiments, each of them with a different size of the window used for the construction of the classifiers — the window sizes used were 5, 11, 21, and 31. The test sequences of lengths 21, 31, 51, and 71 were extracted from 10 test time series. The starting points of test sequences where chosen at random. In the experiments we dated each test sequence from the above described four groups of 10 sequences using different theories. For each run of the algorithm we collected the actual and predicted year of first value in the sequence, the

Table 1. Evaluation of matching algorithm for tree-ring dating

SeqL	WinL	NumM	AvgH	MinH	MaxH	AvgGP	AvgMP
21	5	9	75.82	52.94	94.12	0.88	0.64
31	5	7	73	59.26	92.59	0.88	0.58
51	5	8	71.53	53.19	91.49	0.89	0.56
71	5	8	72.39	55.22	89.55	0.87	0.57
21	11	5	87.27	72.72	100	0.94	0.77
31	11	8	82.14	61.90	100	0.92	0.64
51	11	10	78.78	53.66	97.56	0.92	0.61
71	11	10	79.34	59.02	98.36	0.86	0.62
31	21	6	97	90.91	100	0.90	0.81
51	21	9	87.45	64.52	100	0.92	0.68
71	21	10	84.51	58.82	100	0.93	0.67
51	31	9	93.14	76.19	100	0.90	0.71
71	31	10	90.97	65.85	100	0.92	0.70

number of hits which confirmed the match, the probability of the first guess and the collective probability of the match. Table 1 presents the results compressed in a single line for each group of ten test sequences of the same length. For each group we present: the length of sequences (SeqL); the size of window used for the construction of theory (WinL); the number of correct matches (NumM); the average, minimum and maximum number of hits (AvgH, MinH, and MaxH) expressed in percentages; the average probability of a guess (AvgGP) that leaded to matching; and the average collective probability of a correct match (AvgMP).

The parameters of Algorithm 1 which define matching of a sequence with the theory were as follows. The required percent of classifiers which agreed on the prediction T_h were 50%, the required probability of the initial guess T_g was 80%, and the required collective probability of the match T_m was 40%. Note that the thresholds T_h and T_m were set relatively low in order to allow the matchings based on the partial agreement between the subsequence and the theory; the values of AvgH in Table 1 denote the average proportion of the agreement by matchings. Finally, the prediction supported with the largest number of hits is chosen among the results of Algorithm 1.

The percentage of correctly dated test sequences in the complete set of experiments is 83.8%. However, the available set of time series describe the trees from 2 different geographical areas which may be one of the reasons for 16.2% incorrectly dated cases. This claim is supported by the fact that the incorrectly dated subsequences are, in many cases, extracted from the distinctive subset of 10 test time series.

3.4 Pattern Recognition

In this section we present the application of matching algorithm for pattern recognition. The patterns are represented as short sequences of values which capture the characteristics behavior of the observed parameter. The patterns which are used in the experiments are drawn in Figure 3. While the sequences are very specific representations of patterns, the rules which describe the properties of the time points are general — there are only a few rules which describe a particular time point. For this reason, the matching algorithm can find general patterns which are supported by a subset of time series from the domain.

The patterns which are used in the experiments on the tree-ring data are of length 5, 7 and 9 (years). We consider that in this domain the selected length of patterns (sequences) can capture the characteristic episodes in the growth of trees which may be the effects of the conditions in the environment. The first line of patterns represents temporarily defeated growth of the trees. The second and the third lines represent simple steps. Finally, the last line presents pattern where the growth rate of trees increases temporarily and than again decreases in the second part of the interval to the initial rate.

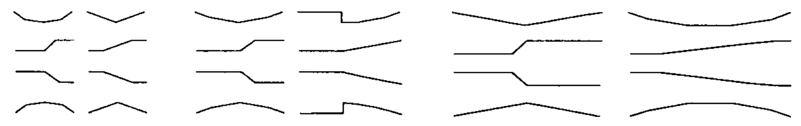
Fig. 3. Examples of patterns.

The theory which is constructed using the window of size 5 is used in all experiments. Because of the low classification accuracy of the complete theories constructed with the window of length 5 (Section 3.2), we used considerable high threshold for the collective probability of match (70%) as well as the high threshold for the required number of hits for a match (90%). Notice that the high threshold for the collective probability of match implies high accuracy of rules triggered for each particular time point in the pattern.

Let us now describe the results of the experiments presented in Table 2. The columns of the table represent: the pattern number (PatN), the length of the pattern (PatL), the number of matchings of the pattern with the theory (NumM), the average probability of guesses which leaded to matching (AvgGP), the average collective probability of the match (AvgMP), and the list of time points (TPts) where the pattern appeared. The pattern number stated in Table 2 represents the ordering number of patterns from Figure 3. The ordering of patterns is from left to right and then from the first to the fourth line (pattern number 1 is in the upper left corner). Finally, note that the number of hits obtained in matchings is not presented in the table since all possible hits were achieved for each matching.

The algorithm found from 3 to 8 instances of each pattern of length 5 in the domain. We can observe from the values of the column `TPts` that in some cases the neighboring patterns (horizontally in Figure 3) appear on the same time point which means that the occurrence of the pattern is more general. An

Table 2. Evaluation of matching algorithm for pattern recognition

PatN	PatL	NumM	AvgGP	AvgMP	TPts
1	5	8	0.89	0.89	1880,1888,1896,1912,1918,1929,1937,1963
2	5	7	0.86	0.86	1880,1888,1896,1929,1937,1948,1956
3	7	2	0.97	0.87	1896,1929
4	7	3	0.92	0.78	1900,1911,1921
6	9	2	0.94	0.81	1901,1912
7	5	5	0.91	0.91	1881,1889,1905,1913,1930
8	5	3	0.87	0.87	1889,1905,1913
9	7	3	0.88	0.85	1881,1930,1939
11	9	2	0.91	0.78	1898,1931
13	5	7	0.89	0.89	1892,1900,1911,1921,1928,1936,1953
14	5	4	0.89	0.89	1900,1928,1945,1955
15	7	5	0.93	0.78	1892,1900,1921,1936,1962
19	5	5	0.83	0.83	1884,1890,1899,1903,1926
20	5	4	0.85	0.85	1873,1884,1916,1926
21	7	2	0.93	0.79	1916,1932
22	7	3	0.92	0.86	1926,1931,1940

example when the similar patterns of the length 5, 7 and 9 are matched is the step represented by patterns 7, 9 and 11. The steps of length 5 and 7 appear in the year 1930 while the step of length 9 appears in the year 1931. The shift of one year in the central points of patterns appears because the steps are on the right side of the central time point in the patterns 7 and 9 while the step is on the left side of the center in the pattern 11. The patterns of length 7 and 9 are less frequent than the patterns of length 5. This reflects the nature of the processes — stable patterns in in the growth rate of trees are expected to appear only in the period of few years.

The study of the occurrences of the patterns in the data showed the correspondence between the collective probability of the match, and the support of the pattern in the data. The higher the collective probability is more the pattern is supported by the domain. Still, the support of the patterns in the data varies. The computation of more accurate estimation for the support of patterns in data would require the information about the support in data for each particular rule triggered for the time points of patterns.

4 Related Work

We distinguish between two types of methods for matching subsequences of time series. Firstly, a considerable research interest has been directed recently to the development of methods for matching subsequences in time series databases. These methods are defined to identify subsequences in particular time series. Secondly, methods for discovering characteristic subsequences of sets of time series

have been recently proposed. These methods deal with the common features of sets of time series and not with the characteristics of the individual time series.

Let us first overview the methods for matching subsequences in time series databases and their relations to the proposed algorithm. As pointed out in [8], the methods differ in the representation of the time series, the comparison of time series, and in the search mechanisms. The representations of the time series used by the methods are: the normalized series [2], the representation based on the feature space obtained by Discrete Fourier Transformation [7], the piecewise linear models [8], and the piecewise linear models augmented with weights [9]. The methods for the identification of similar time series are: a simple distance function [2], the probabilistic method for the computation of the similarity measure [8], Euclidean distance between the time series [7], and the longest common subsequence of two series [6]. The search methods which were recently proposed for matching subsequences in time series databases are based on: the dynamic programming technique [2], the sequential scan for pattern identification [8], and the index data structures based on feature representation of time series [7,8].

Let us now present some of the approaches to the discovery of the common characteristic of the set of time series. The most widely known approach to the identification of the common subsequences of the set of time series is the algorithm for finding the longest common subsequences [13]. This algorithm has been used recently for the computing the similarity of sequences [6,12]. The main disadvantage of the algorithm is its exponential space and time complexity. Finally, a method for the identification of distinctive time series from the set of time series by using clustering has been introduced recently in [12].

5 Conclusions

An algorithm for matching sequences with the set of time series was presented in the paper. The algorithm can be employed for locating the starting time points of the undated sequences of values, and for locating the pattern templates which are common to the set of sequences. The algorithm was successfully used for the problems of dating and pattern recognition in tree-ring time series. We consider that the matching algorithm is general enough to be effectively used for other domains including stock market data, sensor data in engineering environments, and medical measurements.

Acknowledgments

We are thankful to Luc De Raedt and Stefan Kramer for their helpful comments and suggestions.

References

1. Agrawal, R., Lin, K.I., Sawhney, H.S., Shim, K., 'Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases', In *Proc. of VLDB Conference*, pp.490-501, 1995.
2. Berndt, D.J., Clifford, J., 'Finding Patterns in Time Series: A Dynamic Programming Approach', In *Advances in Knowledge Discovery and Data Mining*, U.M. Fayyad, Shapiro, G.P., Smith, P., and Uthurusamy, R., Eds., AAAI Press / The MIT Press, 1996.
3. Bollobas, B., Das, G., Gunopulos, D., Mannila, H., 'Time-Series Similarity Problems and Well-Separated Geometric Sets', *Proc. of 13th Annual ACM Symposium on Computational Geometry*, To appear, 1998.
4. Cook, E.R., Kairiukstis, L.A., *Methods of Dendrochronology: Applications in the Environmental Sciences*, Kluwer Academic Publishers, 1990.
5. Das, G., Gunopulos, D., Mannila, H., 'Finding similar time series', In *Proc. of PKDD'97*, LNCS, 1997.
6. Das, G., Gunopulos, D., Mannila, H., 'Rule discovery from time series', In *Proc. of Conf. on Knowledge Discovery and Data Mining*, AAAI Press, 1998.
7. Faloutsos, C., Ranganathan, M., Manolopoulos, Y., 'Fast Subsequence Matching in Time-Series Databases', In *Proc. of SIGMOD Conference*, 1994.
8. Keogh, E., Smyth, P., 'A Probabilistic Approach to Fast Pattern Matching in Time Series Databases', In *Proc. of Conf. on Knowledge Discovery and Data Mining*, AAAI Press, 1997.
9. Keogh, E.J., Pazzani, M.J, 'An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback', In *Proc. of Fourth Conf. on Knowledge Discovery and Data Mining*, 1998.
10. Mitchell, T.M., *Machine Learning*, McGraw-Hill Series in Computer Science, 1997.
11. Monserud, R.A., 'Time-series analyses of tree-ring chronologies', *Forest Science* 32, pp.349-372, 1986.
12. Oates, T., 'Identifying Distinctive Subsequences in Multivariate Time Series by Clustering', In *Proc. of Conf. on Knowledge Discovery and Data Mining*, 1999.
13. Sankoff, D., Kruskall, J.B., *Time Wraps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*, Addison-Wesley, 1983.
14. Savnik, I., Lausen, G., Kahle, H.P., Spiecker, H., Hein, S., 'Algorithm for Matching Sets of Time Series', Technical Report 134, Institut für Informatik, Universität Freiburg, Mar 2000.
15. Quinlan, J.R., *C4.5: Programs for Machine Learning*, Morgan Kauffman, 1993.

MSTS: A System for Mining Sets of Time Series

Georg Lausen, Iztok Savnik, and Aldar Dougarjapov

University of Freiburg, Institute for Computer Science
Georges-Köhler-Allee, 79110 Freiburg, Germany
`{lausen,savnik,dougarja}@informatik.uni-freiburg.de`

Abstract. A system to support the mining task of sets of time series is presented. A model of a set of time series is constructed by a series of classifiers each defining certain consecutive time points based on the characteristics of particular time points in the series. Matching a previously unknown series with respect to a model is discussed. The architecture of the *MSTS*-System (*M*ining of *S*ets of *T*ime *S*eries) is described. As a distinctive feature the system is implemented as a database application: time series and the models, i.e. series of classifiers, are database objects. As a consequence of this integration, advanced functionality as the manipulation of models and various forms of meta learning can be easily build on top of MSTS.

1 Introduction

Data in form of time series represents the behaviour of some time-dependent processes and can be used to exhibit the characteristics of the observed processes. In the literature there exists much work treating various aspects of time series. Matching characteristic patterns in time series databases is discussed in [2, 8, 1, 12, 5, 14]. These methods vary in the representation techniques for time series, the algorithms for measuring similarity between the time series, and in the search mechanisms used for mining patterns. Methods for the discovery of rules [6, 11] and frequent episodes [15] in time series have been proposed, as well. Rules and episodes describe the interrelations among the local shapes of time series.

The discovery of common patterns from a *set* of time series has a long tradition in statistics and one of the most widely used techniques is cross-correlation [3]. In the machine-learning framework the problem has previously been studied only by [13], to the best of our knowledge. Similar to cross-correlation, [13] base their approach on time series as a whole and classify them according to a distance measure. We take a different approach and classify concrete time points based on their local properties, i.e. properties of time points within a certain window on the whole time series. In contrast to the more mean-value oriented known technique, the resulting pointwise classification of a time series gives direct handles to explanations of the behaviour of the time series. This approach originated from a practical project to date tree trunks according to the yearly radial increase [17]; in the current paper we present a simplified and more efficient version of the algorithm and discuss its usuage as part of a data mining system.