# Mining Hierarchical Temporal Patterns in Multivariate Time Series

Fabian Mörchen and Alfred Ultsch

Data Bionics Research Group
University of Marburg, D-35032 Marburg, Germany

**Abstract.** The *Unification-based Temporal Grammar* is a temporal extension of static unification-based grammars. It defines a hierarchical temporal rule language to express complex patterns present in multivariate time series. The *Temporal Data Mining Method* is the accompanying framework to discover temporal knowledge based on this rule language. A semiotic hierarchy of temporal patterns, which are not a priori given, is build in a bottom up manner from static logical descriptions of multivariate time instants. We demonstrate the methods using music data, extracting typical parts of songs.

## 1 Introduction

Knowledge Discovery is the mining of previously unknown rules that are useful, understandable, interpretable, and can be validated and automatically evaluated [1]. There are few approaches for mining rules from time series, even less for multivariate time series. Some mining techniques are based on data models that can be obtained from time series by clustering [2], segmentation [3] or discretization [4]. Most methods concentrate on a single data model and their rules represent only one or very few temporal concepts, e.g. coincidence [5].

Our rule language, called *Unification-based Temporal Grammar* (UTG), is based on multiple data models. The problem is decomposed into the mining of single temporal concepts. The resulting rules have a hierarchical structure that opens up unique possibilities in relevance feedback during the knowledge discovery process and in the interpretation of the results. An expert can focus on particularly interesting rules and discard valid but known rules before the next level constructs are searched. After obtaining the final results, an expert can zoom into each rule to learn about how it is composed and what it's meaning and consequences might be. The decomposition is also of advantage for the mining algorithms, because the hypothesis space for a single mining step is smaller. We will give a detailed description of the UTG and explain the temporal concepts expressible with this language. The accompanying time series data mining framework, called *Temporal Data Mining Method*, will be briefly described and applied to music data.

In Section 2 we mention some alternative methods. Section 3 describes our rule language in detail and Section 4 outlines the mining framework. The use of the UTG and TDM is demonstrated in Section 5. The results of the application are discussed in Section 6. Section 7 summarizes the paper.

## 2 Related work

We know of two approaches that extract rules from multivariate time series. Both convert the time series to labeled intervals using segmentation and feature extraction. Höppner [6] mines temporal rules expressed with Allen's [7] interval logic and a sliding window to restrict the pattern length. The patterns are mined with an *a priori* algorithm using support and confidence and ranked by an interestingness measure afterwards. Last *et. al.* [3] mine association rules on adjacent intervals using the Info-Fuzzy Network (IFN). The rule set is reduced using Fuzzy theory.

The next two methods also work on interval sequences, that could be obtained from time series in analogy to the approaches above. Villafane *et. al.* [5] search for containments of intervals. A containment lattice is constructed from the intervals and rules are mined with the so called Growing Snake Traversal to reduce the storage space required by the naive algorithm. Kam and Fu [8] use Allen's interval operators to formulate patterns. The rules are restricted to so called A1 patterns, that only allow concatenation of operators on the right hand side. The patterns are mined with an *a priori* algorithm.

The remaining approaches work on symbol sequences possibly obtained from time series. Mannila *et. al.* discover frequent Episodes [9] using an *a priori* style algorithm. Das *et. al.* [2] use clustering of short time series segments extracted with a sliding window to produce a sequence of cluster labels. The symbol sequence is mined for association rules within a time window interpreted as *if-then* rules. An extensions to multivariate time series is proposed. Recently, Saetrom and Hetland [4] criticized the restrictive rule languages (e.g. for Episodes) needed to make many mining approaches feasible. A general rule language, that includes many others as special cases, is proposed. The patterns are mined using Genetic Programming [10]. The candidate patterns are evaluated using special hardware to speed up the search.

## 3 Unification-based Temporal Grammar

The *Unification-based Temporal Grammar* (UTG) is a rule language developed especially for the description of patterns in multivariate time series [11] [12].

Unification-based Grammars are an extension of context free grammars with side conditions. They are formulated with first order logic and use unification. The use of Definite Clause Grammars over Context Free Grammars makes it possible to formulate semantic side conditions. The conditions are checked while the rules are evaluated and they can also be used to interpret the semantics of the rules.

The UTG offers a hierarchical description of temporal concepts. With a hierarchy of semiotic levels complex patterns are successively built from lower level constructs. Starting with simple patterns, called *Primitive Patterns*, and using intermediate concepts called *Successions*, *Events*, and *Sequences*, the final rules, called *Temporal Patterns* are created (see Figure 1).

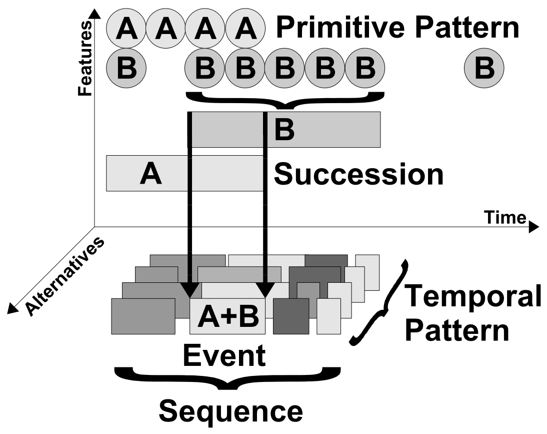

**Fig. 1.** Unification-based Temporal Grammar

At each hierarchical level the grammar consists of semiotic triples: a unique symbol (syntax), a grammatical rule (semantic), and a user defined label (pragmatic). The grammatical rule is produced by a mining algorithm for this hierarchy level (see Section 4). An expert is needed to interpret the rule and complete the triple with a meaningful label. Optionally, the symbol can also be set to a meaningful abbreviation. An example of such a semiotic triple is shown in Figure 2. The generated rule describes a Primitive Pattern (a state assignment for a time point) by two interval conditions. An expert could diagnose this as a beat from a *bass drum*, assign the appropriate label and choose e.g. *BD* as a symbol.

```
symbol: BD
rule: A PrimitivePattern is a ’BD’
if
        ’lt770Hz’    in      [-6908.37, 15860.27]
and
        ’lt920Hz’    in      [-2079.46, 17153.04]
.
label: bassdrum
```

**Fig. 2.** A semiotic triple

In absence of a domain expert, the unique symbols and labels can be generated automatically during the mining process, but they should be adjusted later for better interpretation. One semiotic triple describes a class of constructs, each class usually has many instances occuring at certain time points or during certain intervals.

On each level we allow a special blank symbol, called *Tacet*[1], to express the fact, that there is no UTG construct for a time point or interval. Short, not plausible interruptions of an otherwise persisting state are called *Transients*. The maximum length for Transients is application and level dependent. A group of related time series is called *Aspect*.

A *Primitive Pattern* describes a single point in time. It represents a temporal atom, because it has unit duration. Not every time point needs to represent a state, points can be labeled as Tacets. We allow all rules that map the data space of an Aspect to a symbol representing a class or to the complement of the class, i.e. classification rules, not temporal rules. This very general definition is informally constrained as follows: The classification rules should be given in first order logic (FOL), possibly extended by an estimation calculus, for example a conjunction of interval conditions. This ensures that the rules can be automatically evaluated by an expert system. The resulting data model for each Aspect on this hierarchy level is a sequence of discrete symbols including the Tacet label.

A *Succession* introduces the temporal concepts of duration and persistence. It represents a time interval where nearly all time points have the same Primitive Pattern label. The data model for each Aspect on this hierarchy level is a sequence of labeled intervals, including Tacet intervals.

An *Event* represents the temporal concepts of coincidence and synchronicity. It represents a time interval where several Successions overlap. If the distances between the first and last start point and between the first and the last end point of all Successions are below a threshold, the Event is called *synchronous*. Note, that only Events work on multivariate input data in form of several Succession series, the other hierarchy levels have univariate input data. The user interaction, in form of an expert choosing a label, is very important for Events. While a label can be generated from the Succession labels and durations involved, a short and precise label based on application insight will be easier to grasp in higher level constructs. The common data model for all Aspects on this hierarchy level is a single sequence of labeled intervals, including Tacet intervals, where no Events were found.

A *Sequence* introduces the temporal concept of order. A Sequence is composed of several Events occuring sequentially, but not necessarily with meeting end and start points. A Sequence is thus a typical subsequence of the Event sequence ignoring Event and Tacet durations. The common data model for all Aspects on this hierarchy level is a set of labeled intervals.

A *Temporal Pattern* is the abstraction of several Sequences based on alternatives. Several similar Sequences, differing in only a few Events, form a Temporal Pattern.

[1] A Tacet is the pausing of an instrument or voice in a musical piece

## 4 Temporal Data Mining Method

The time series Knowledge Discovery framework *Temporal Data Mining Method* (TDM) is described briefly.

The starting point of the TDM is a multivariate time series, usually but not necessarily uniformly sampled. The knowledge discovery steps of the TDM are shown in Figure 3. First, preprocessing and feature extraction techniques should be applied where necessary. An expert should group the set of time series into possibly overlapping subsets, the Aspects. The series within an Aspect should be related w.r.t. the investigated problem domain. In the absence of such prior knowledge, one Aspect per time series can be used. The remaining steps correspond to the hierarchy levels of the UTG and are described below.

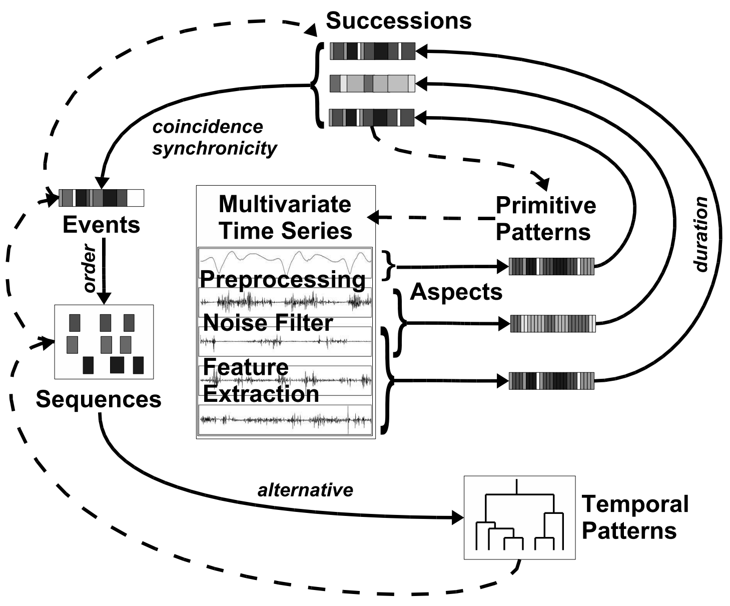

**Fig. 3.** Temporal Data Mining Method

Since the amount of data in multivariate time series is usually very large some abstraction mechanism is necessary to make the detection of regularities possible. Finding Primitive Patterns, i.e. reducing the large amount of distinct high dimensional values in a (multivariate) time series to a limited number of state labels provides such an abstraction. Each Aspect is treated individually to produce a discrete state sequence, possibly containing gaps in the form of Tacets.

For univariate Aspects many existing discretization techniques can be used, e.g. [13] [14] [15] [16] [17] [18]. It is important, that the symbols are accompanied by a rule and a linguistic description like *high* or *convexly increasing*.

For Aspects spanning several time series, it is often better to use clustering and rule generation on the spatial attributes. If the process alternates between several regimes or states, these regions should form clusters in the space spanned by the spatial attributes. In [19] we used Emergent Self-Organizing Maps [1] to identify clusters in the dataset. The rules for each class were generated using the Sig* Algorithm [20]. Other combinations of clustering and rule generation algorithms can be used as well.

A simple, straightforward algorithm is used to create Successions of Primitive Patterns. Transients are filtered out in a post-processing step. For this filter, the user needs to specify the maximum length of an interruption.

All Events maximal w.r.t the number of aspects involved and the length of the interval can be found with a linear time algorithm. Synchronous Events are identified, given a range threshold. The resulting Events are usually filtered by an interestingness measure, e.g. by length or frequency. Additionally, Transients can be filtered out.

A rather difficult step of the conversion process is the discovery of Sequences. We are using an algorithm based on Sequitur [21], that builds a hierarchical grammar from a sequence of discrete symbols in linear time. Alternatively, we have successfully used a simple genetic algorithm.

Since sequences can and often do overlap, the last step tries to find generalized Sequences, call Temporal Patterns. The distance between two Sequences can be calculated with a string distance metric and hierarchical clustering can be used to find groups of similar sequences.

## 5 Application

We applied the TDM to a multivariate time series extracted from audio data. Sound in audio CD quality is sampled at 44KHz and not multivariate at first sight. For stereo sound there are two channels, usually highly correlated. This essentially univariate time series does contain a lot of information, though. Many sounds, that can be modelled as combinations of sine waves, are overlayed. We therefore extract multiple channels from the univariate time series to describe different features.

A straightforward way to obtain a multivariate series is to calculate the loudness in different frequency bands using the Short Time Fourier Transform (STFT). The frequency bands and some weighting factors were chosen according to psychoacoustic models as used by Pampalk *et. al.* [22]. Nine frequency bands were placed between 1.5KHz and 6.4KHz, because these frequencies are most relevant for the human auditory perception [22]. Two more bands cover the remaining low and high frequencies. We are also experimenting with more complex features describing the current pitch or beat [23]. For this analysis, only the loudness features were used.

One Aspect per frequency band was created. Alternatively, groups of correlated neighboring frequency bands could be merged into larger Aspects. The windows size for the STFT was about 0.1 seconds with 50% overlap. This produces a time series with about 5k samples for a typical 4 minute song. The series was smoothed with a weighted moving average using a window of width 10.

Each univariate aspect was discretized using a histogram with percentile based bins corresponding to the labels *low* (40%), *medium* (20%), and *high* (40%). See Figure 4 for a typical plot of the Pareto Density Estimation [24] with the histogram bins. Figure 5 shows one of the semiotic triples created.

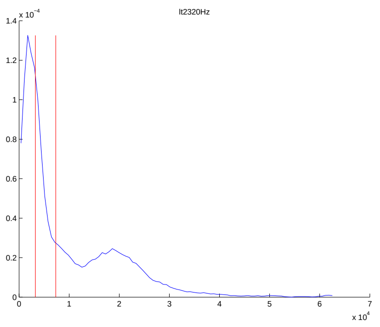

**Fig. 4.** Loudness probability density

```
symbol: M
rule: A PrimitivePattern is a 'M'
if
        'lt4400Hz'    in      [3244.18, 7328.63]
.
label: medium
```

**Fig. 5.** A Primitive Pattern triple

Only very short Successions were filtered out, to keep a high level of detail corresponding to elementary sounds. Figure 6 shows the Primitive Patterns and Successions of one frequency band over a time window of 45 seconds. A typical semiotic triple is listed in Figure 7. Note, that the label is inherited from the underlying Primitive Pattern. The minimum and maximum duration is annotated. There are several Transients removed around the sample indices 800 and 1600.

Most Events lasted less than a second, representing elementary sounds. The interpretation of Events is relatively easy, because the original audio data is available. By listening to all instances of an Event labels like *guitar riff*, *bass drum*, *scream* etc. can be assigned. The bottom row of Figure 8 shows some Event instances found in 37 seconds of the song *Island In The Sun* by *Weezer*. The high activity over all frequency bands from sample index 2000 to 2350 with

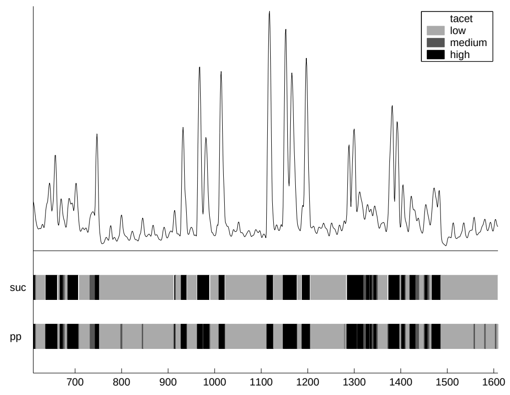

**Fig. 6.** Successions in 45 seconds audio data

```
symbol: L
A Succession is a 'L'
if
        'L' lasts [2, 15]
.
label: low
```

**Fig. 7.** A Succession triple

several events found corresponds to the chorus of this song. In Figure 9 you can see part of the triple for a typical scream present in the song *Bad Habit* by *The Offspring.*

The Sequences usually lasted between 5 and 10 seconds, representing typical parts of the song. Again, interpretation is fairly easy. Figure 10 shows the most frequent Sequences found in the song by Weezer. Sequence 9 in the second row corresponds to the chorus mentioned above. Figure 11 lists the semiotic triple.

The Sequences 2, 0, 11, and 3 occur partly during the same time intervals and partly during adjacent intervals at the beginning of the song. A closer look at a more compact representation in Figure 12 reveals a high similarity between them. These Sequences should thus be merged into a Temporal Pattern, the final form of the UTG representation.

## 6 Discussion

The results on audio data are promising. The TDM found typical parts in 11 songs of from different genres. These typical parts could be used for further analysis and feature extraction [23]. Determining the similarity of different songs based on a typical part of each should give better results than using an arbitrarily placed window. Compared to using the whole song, the storage and computational effort is tremendously reduced.

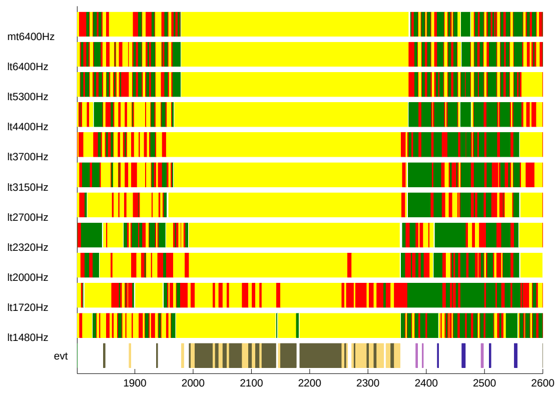

**Fig. 8.** Events in 90 seconds audio data

```
symbol: S
An Event is a 'S'
if
        'lt1480Hz high'
coincides with
        'lt1720Hz low'
...
coincides with
        'lt6400Hz high'
coincides with
        'mt6400Hz high'
lasts
        [3,26]
.
label: scream
```

**Fig. 9.** An Event triple

A validation of the results other than listening to the extracted samples is difficult, but for the above mentioned application it should be enough to have some typical part, not the most typical.

The discretization of the loudness into Primitive Patterns leaves room for improvements. Instead of percentile based bin boundaries, the modes of the energy distribution should be taken into account, e.g. by using Gaussian mixture models.

Note, that Sequences are very robust with respect to the single Event durations and the length of the gaps between consecutive Events. Grounded on the underlying Event instances, a Sequence bridges the gaps by including the Tacet intervals in the final Sequence interval. This is especially helpful for audio data, because longer consecutive audio samples are extracted. While we see this robustness and completion as an advantage of our method, for some applications

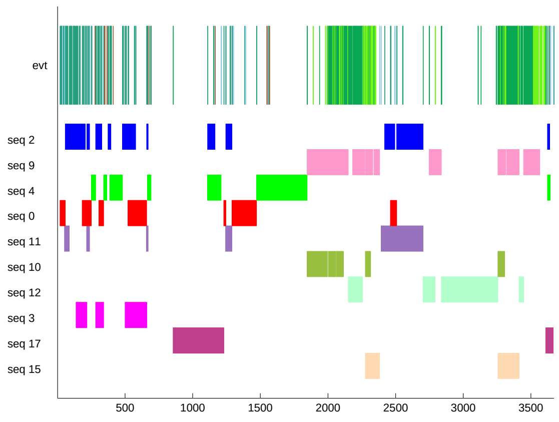

**Fig. 10.** Sequences in a song

```
symbol: C
A Sequence is a 'C'
if
        'almost all high energy' lasts [5, 10]
followed after [2, 4] by
        'all high energy' lasts [4, 12]
...
lasts
        [53, 200]
.
label: chorus
```

**Fig. 11.** A Sequence triple

constraints on the length of the gaps might be needed. The instance of Sequence 2 in Figure 11 at the sample index 2500 seems to span too large gaps.

One could criticize the manual interaction needed for our mining method. Many parameters need to be adjusted. Based on the intermediate results, decisions have to be revised and previous mining steps have to be rerun. But this is only needed during the Knowledge Discovery phase. Automating this step is not advisable, because (partial) results should always be carefully validated. The hierarchical structure of the Unification-based Temporal Grammar offers unique possibilities for the expert to interpret, investigate and validate the discovered rules at different abstraction levels and change parameters based on this analysis. The application of the temporal rules to new data can be automated using an expert system interpreting the logical UTG rules. We are working on finding a robust set of parameters for audio data, to automate the location of typical song elements.

```
seq  2: 3 -> 3 -> 3 -> 3 -> 3 -> 3 -> 3 -> 3
seq  0: 3 -> 4 -> 3 -> 4
seq 11: 4 -> 3 -> 3
seq  3: 3 -> 3 -> 3 -> 3 -> 3 -> 4
```

**Fig. 12.** Similar Sequences

A detailed analysis of Successions indicated, that allowing the concept of order on the Succession level, might improve the results. This way Events would be more robust towards little time shifts and varying durations of typical Succession patterns. While the data model for Events is currently univariate, we are experimenting with algorithms allowing overlapping Events. However, this increases the number of Events found and makes mining Sequences more problematic.

## 7 Summary

We have described our temporal rule language and the accompanying time series knowledge extraction framework. Many methods and algorithms can and need to be combined to mine rules in UTG notation. The UTG builds up a hierarchy of concepts that introduces the temporal concepts *duration*, *coincidence*, *synchronicity* and *order* at successive levels. Rules from each level are accompanied by linguistic descriptions, thus partial results can be interpreted and filtered by experts. All mining steps were demonstrated using audio data. We were able to find typical parts in songs from different genres.

## Acknowledgements

We thank Niko Efthymiou, Ingo Löhken, and Michael Thies for programming the audio feature extraction.

## References

1. Ultsch, A.: Data mining and knowledge discovery with emergent self-organizing feature maps for multivariate time series. In: Oja, E., Kaski, S. (Eds.): Kohonen Maps. (1999) 33–46
2. Das, G., Lin, K.I., Mannila, H., Renganathan, G., Smyth, P.: Rule discovery from time series. In: Knowledge Discovery and Data Mining. (1998) 16–22
3. Last, M., Klein, Y., Kandel, A.: Knowledge discovery in time series databases. IEEE Transactions on Systems, Man, and Cybernetics 31B(2001). (2001)
4. Saetrom, P., Hetland, M.L.: Unsupervised temporal rule mining with genetic programming and specialized hardware (2003)
5. Villafane, R., Hua, K.A., Tran, D., Maulik, B.: Mining interval time series. In: Data Warehousing and Knowledge Discovery. (1999) 318–330

6. Höppner, F.: Learning dependencies in multivariate time series. Proc. of the ECAI'02 Workshop on Knowledge Discovery in (Spatio-) Temporal Data, Lyon, France (2002) 25–31
7. Allen, J.F.: Maintaing knowledge about temporal intervals. Comm. ACM, 26(11) (1983) 832–843
8. Kam, P.S., Fu, A.W.C.: Discovering temporal patterns for interval-based events. In Kambayashi, Y., Mohania, M.K., Tjoa, A.M., eds.: Second International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2000). Volume 1874., London, UK, Springer (2000) 317–326
9. Mannila, H., Toivonen, H., Verkamo, A.I.: Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery **1** (1997) 259–289
10. Koza, J.R.: Genetic programming. In Williams, J.G., Kent, A., eds.: Encyclopedia of Computer Science and Technology. Volume 39., Marcel-Dekker (1998) 29–43
11. Ultsch, A.: A unification-based grammar for the description of complex patterns in multivariate time series (german) (1996)
12. Ultsch, A.: Unification-based temporal grammar. In: Technical Report No. 37, Philipps-University Marburg, Germany. (2004)
13. Geurts, P.: Pattern extraction for time series classification. Lecture Notes in Computer Science **2168** (2001) 115–127
14. Lin, J., Keogh, E., Lonardi, S., Chiu, B.: A symbolic representation of time series, with implications for streaming algorithms (2003)
15. Agrawal, R., Psaila, G., Wimmers, E.L., Zaot, M.: Querying shapes of histories. In Dayal, U., Gray, P.M.D., Nishio, S., eds.: Twenty-first International Conference on Very Large Databases (VLDB '95), Zurich, Switzerland, Morgan Kaufmann Publishers, Inc. San Francisco, USA (1995) 502–514
16. Rodriguez, J.J., Alonso, C.J., Bostroem, H.: Learning first order logic time series classifiers (2000)
17. Kadous, M.W.: Learning comprehensible descriptions of multivariate time series. In: Proc. 16th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA (1999) 454–463
18. Höppner, F.: Discovery of temporal patterns – learning rules about the qualitative behaviour of time series. In: Proc. of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases, Lecture Notes in Artificial Intelligence 2168, Springer. (2001)
19. Guimaraes, G., Ultsch, A.: A method for temporal knowledge conversion. In: D. J. Hand, J. N. Kok, and M. R. Berthold, editors, Advances in Intelligent Data Analysis, Proc. of the 3rd Int. Symp., Amsterdam, The Netherlands, Sprginer, Berlin. (1999) 369–380
20. Ultsch, A.: Connectionistic models and their integration in knowledge-based systems (german) (1991)
21. Nevill-Manning, C., Witten, I.: Identifying hierarchical structure in sequences: A linear-time algorithm. Journal of Artificial Intelligence Research, 7 (1997) 67–82
22. Pampalk, E., Rauber, A., Merkl, D.: Content-based Organization and Visualization of Music Archives. In: Proceedings of the ACM Multimedia, Juan les Pins, France, ACM (2002) 570–579
23. Tzanetakis, G., Essl, G., Cook, P.: Automatic musical genre classification of audio signals (2001)
24. Ultsch, A.: Pareto density estimation. In: Proc. GfKl 2003, Cottbus, Germany. (2003)