Recognising Visual Patterns to Communicate Gas Turbine Time-Series Data

Jin Yu, Jim Hunter, Ehud Reiter and Somayajulu Sripada
Department of Computing Science
University of Aberdeen, Aberdeen, AB24 3UE, UK
{jyu, jhunter, ereiter, ssripada}@csd.abdn.ac.uk
www.csd.abdn.ac.uk

Abstract

We have observed that visual patterns play an important part when domain experts interpret time series data. Such patterns change their appearance when displayed at different time scales and a systematic method is proposed to handle this problem. First, a rapid change detector combined with a dynamic limit checker (DRCD) is employed to detect primitive patterns at a basic time scale. Patterns obtained at that time scale are then transformed into patterns viewed at a higher time scale and the DRCD algorithm is reused at this time scale to identify new visual patterns that did not appear at lower time scales. An evaluation of the preliminary results is promising.

1. Introduction

Very large time series data sets are ubiquitous in a variety of on-line monitoring applications in engineering, medicine, business, finance, etc.. For example, more than 250 analogue channels are sampled once per second and archived by the *Tiger* system for monitoring gas turbines [1]. *Tiger* is very successful in performing diagnostic tasks, but its designers would like to explore its large and ever-expanding archive to detect 'interesting events' which *Tiger* does not pick up, in order to subject them to more detailed analysis. This is a common problem with archives of complex multi-channel time series data - they are too large to manually explore and investigate.

Currently, human examination of time series data is generally done either by direct inspection of the numerical values of the data (for small data sets), by graphical visualisation, or by statistical analyses. A further possibility is the generation of textual summaries. We are developing a knowledge-based system to summarise such data in the gas turbine domain [2]. In the knowledge engineering field, knowledge acquisition plays an important role in building such systems [3]. We are using a sophisticated tool - the Time Series Workbench (TSW) - to acquire think-aloud protocols as part of knowledge acquisition from domain experts. We discovered that the experts were focusing on particular patterns when they summarised the time series data. This means that it is very important to identify such patterns in any attempt at summarisation. In the gas turbine domain, there are

many complex patterns contained in the multiple channels - the term 'channel' refers to a series of samples from one variable. Among these patterns, certain primitive patterns such as spikes, oscillations, and steps are regarded as the most important since they are very common and have special meanings to the experts. It is therefore a fundamental task to find a suitable method for recognising these primitive patterns before we try to summarise temporal data in the domain.

The organisation of the remainder of this paper is as follows. Section 2 discusses some interesting questions which arise when we try to recognise visual patterns from time series data. A systematic method to identify such patterns will be described in Section 3. The approach is evaluated in Section 4. Section 5 compares our approach with those of others and some preliminary conclusions are presented in Section 6.

2. Interesting Questions when Recognising Visual Patterns from Time Series Data

Figure 1 displays an example temporal data set at time scale (or resolution) of five-seconds. This means that one screen pixel corresponds to five seconds worth of data. When we refer to a 'one second' time scale we mean that one pixel is used to display one second of data i.e. the data is more 'spread out'. In this example, there are six relevant channels used to monitor the gas fuel subsystem. Through knowledge acquisition activities, a summary of this scenario was obtained from our domain experts and is presented in Figure 2. From the summary we can see that certain primitive patterns (e.g. spikes) are basic words used to communicate the information contained in this scenario. Now, the question is how to recognise these primitive patterns in order to summarise these data?

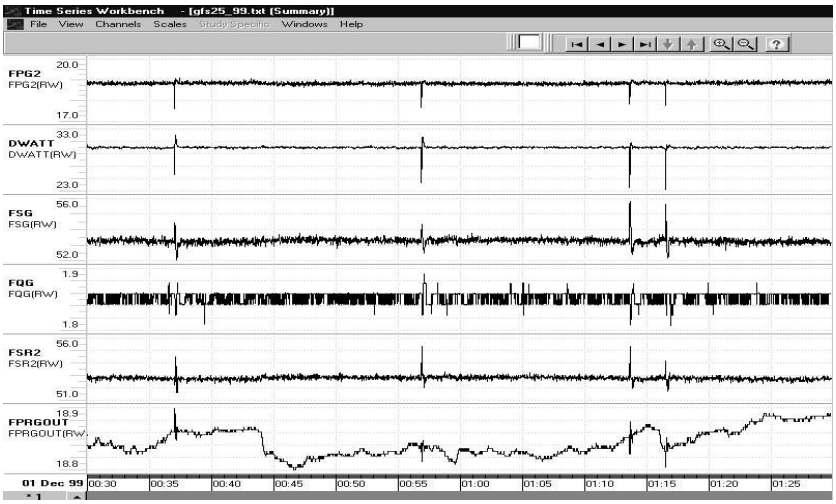


Figure 1: Visualisation of the sample temporal data set

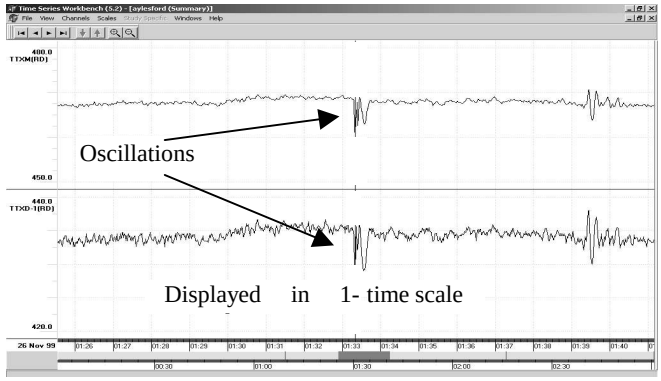
% FPG2 Intervalve gas fuel pressure input % DWATT Generator load watts

% FSR2 Gas fuel stroke reference from fuel splitter % FQG Gas fuel flow

% FPRGOUT Gas ratio valve servo command % FSG Fuel stroke reference

There are three major sets of spikes in this time. FPG2 pressure has huge drops at here, here, and here, when drops occur the power output of this case make a big drop, which is staggering 7 Mwatt, which is extremely unhealthy thing. Both the fuel valve moved and here we have got the almost same situation again the powers dropped, the pressures dropped, the fuel flow has dropped off a little bit here. The set point is come up to slightly delay there the set point coming up to recover the power output while put flow back. We have got two sets at the same time, if we look at timing of spikes of power output, when the power dropped the fuel valve is opened up to compensate. That brought the power back. There is a small amount of overshoot that's taken place in two channels. Here fuel flow drop of more or less most identical spikes, a drop in the power output the fuel valve has come right up compensate the pressure between valve. It's strange that there isn't a big drop in fuel flow at that point. In this period there were three of these spikes that affected the pressure, the power output and the fuel valve. And it is interesting that it isn't obvious that the fuel flow had a spike at the same time here.

Figure 2. Summary of the sample data set from domain experts



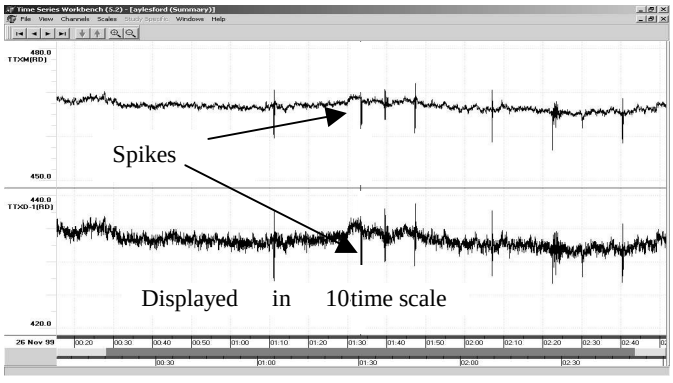


Figure 3: An 'oscillation' becomes a 'spike' when displayed at a different time scale

Data can be visualised at different time scales and/or different amplitude scales. We have observed that the visual shapes of patterns are significantly affected when displayed at different time scales. For example, Figure 3 shows a pattern which might be called an 'oscillation' when displayed at 1 second time scale, but be called a 'spike' when displayed at 10 second time scale. For some data sets, some primitive patterns can not be categorised when displayed at a lower time scale but can be when displayed at a higher time scale. For example, in Figure 4, a 'spike' is recognised when displayed at a 10 second time scale but is not recognised as such at one second.

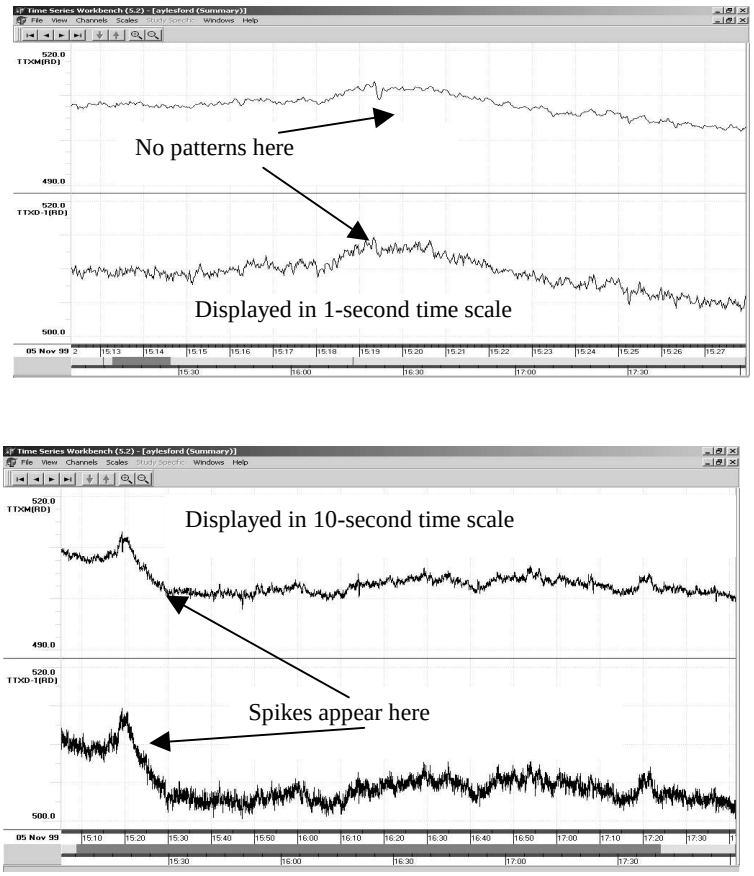


Figure 4: Same data set exhibits different visual shapes
when displayed at different time scales

3. Systematic Method for Recognising Primitive Patterns Displayed at Different Time Scales

In this section we propose a systematic method for recognising primitive patterns in time series data sets when displayed at different time scales. First, a rapid change detector combined with a dynamic limit checker (DRCD) is established to detect primitive patterns at the basic time scale. These patterns are then transformed into patterns at a higher time scale and the DRCD algorithm is reused to identify new visual patterns that did not appear at lower time scales.

3.1 Recognition of Primitive Patterns from Time Series Data Displayed at the Basic Time Scale

Since data samples were collected and archived once per second in the gas turbine domain, we regard one second as the basic time scale when displaying the data during knowledge acquisition. We concentrate on recognising three primitive patterns: spikes, oscillations, and steps i.e. determining what the pattern is and when it starts and ends. The main procedures of DRCD algorithm are given in Figure 5.

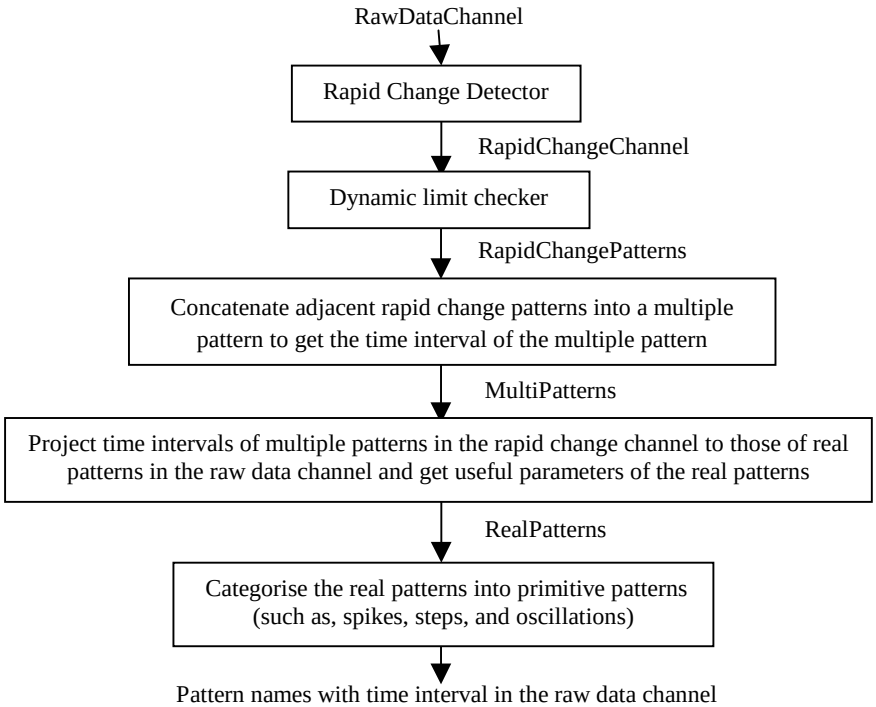


Figure 5: The main procedures of the DRCD algorithm to recognise primitive patterns at the basic time scale

The various steps are explained below.

(1) Rapid change detector (RCD). Our analysis shows that these primitive patterns are composed from one basic element - a ‘rapid change’ (from the point of view of the observer). We chose to measure change by running a fixed sized window over the raw data and measuring the difference between the maximum and minimum values within that window. All our algorithms are implemented within the Time Series Workbench (TSW) - an experimental software environment for manipulating and displaying time series data; the TSW offers a number of predefined 'filters'. The rapid change detector, implemented within the TSW, is composed of a max window filter, a min widow filter, and a difference filter (Figure 6). The two window filters measure max and min value of the raw data within a fixed size window. For example, with a window size of 5 seconds and the window centred on a sample at time 4 seconds (i.e. the window covers the interval from 2 to 6 seconds), there are five samples with values: 2, 4, 6, 3, 9. The output of the RCD at 6 seconds will be 7 (9-2). The window is advanced along the time axis by 1 second increments. A sample raw temporal data channel and the corresponding rapid change channel (the output of the RCD), are given in Figure 7.

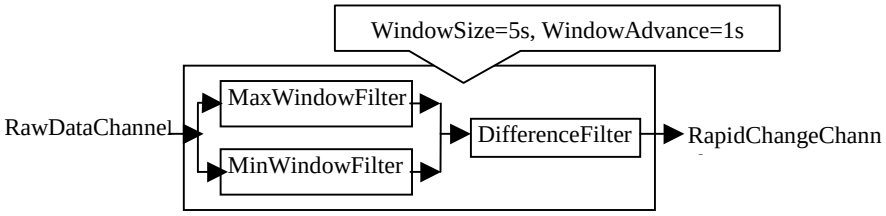


Figure 6: Main components of the rapid change detector

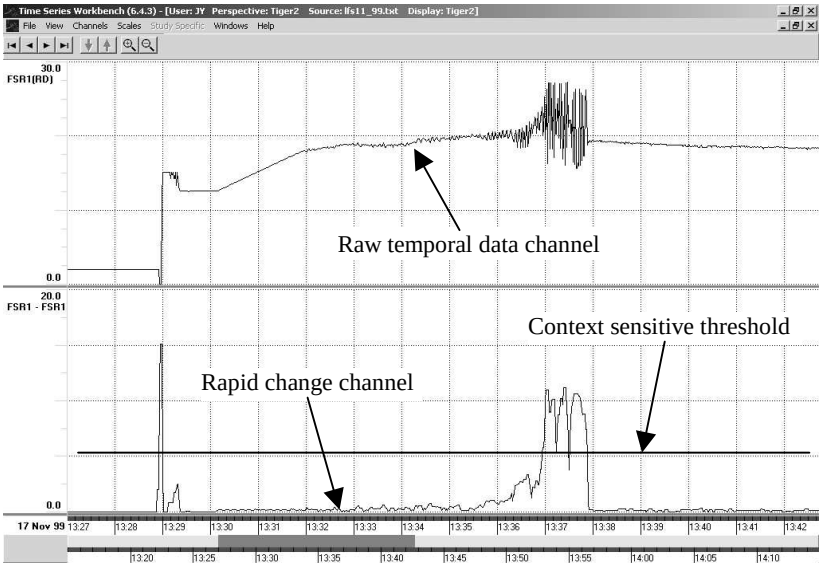


Figure 7: A sample raw temporal data channel and the RCD output

From Figure 7, we can see that there are some useful relationships between the raw temporal data channel and the rapid change channel. Firstly the rapid change channel is simpler than the raw temporal data channel, and is therefore easier to process. Secondly, features of patterns in the rapid change channel correspond with features of patterns in the raw temporal data channel. In the following steps, we will take advantages of these relationships.

(2) Use the dynamic limit checker to select candidates for interesting patterns in the rapid change channels. The standard deviation (σ) is used to measure the distribution of values. For each rapid change channel, σ can be easily calculated:

$$\sigma^2 = \frac{\Sigma(y_i - \bar{y})^2}{N} = \frac{N\Sigma y_i^2 - (\Sigma y_i)^2}{N^2}$$

N = total number of data points in a rapid change channel.

$y_i = f(t_i)$: amplitude value of a data point ($1<=i<=N$) in this channel.

Values between 2σ and 4σ are chosen as a context sensitive threshold in the dynamic limit checker to select candidates for interesting patterns (see Figure 7). The selected candidate patterns are called rapid change patterns and are defined over the interval where the RCD output exceeds the threshold.

(3) If the intervals identified in step (2) satisfy criteria of proximity then they are merged into one interval; this method is similar to temporal interpolation mechanism within Shahar's KBTA framework [4]. The resulting interval is further extended at one or both ends according to further criteria based on the shape of the rapid change channel to yield an interval which we refer to as the 'concatenated pattern' - see Figure 8.

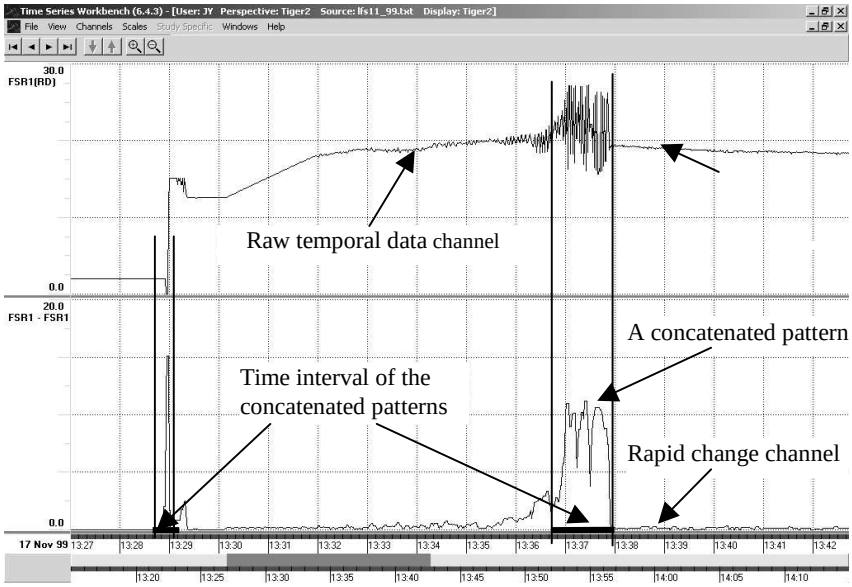


Figure 8: The relations between the two channels

(4) Having established the interval over which we have established that a significant event is taking place, we now return to the raw data and extract a number of statistical values. These are used to categorise the event as one of the following 'primitive' patterns (primitive as viewed by the domain expert) according to the following criteria:

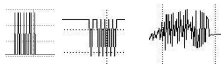
Names	Typical shapes	Typical features (rules to categorise events into primitive patterns)
Steps		Time interval of steps is small The difference between start and end value of steps is large and is near to the difference between the max and min value of steps during the interval
Spikes		Time interval of spikes is small The difference between start and end value of steps is almost same and is much smaller than the difference between the max and min value of spikes during the interval
Oscillations		Time interval of oscillations is big The difference between start and end value of steps is almost same and is much smaller than the difference between the max and min value of oscillations during the interval

Table 1: Typical shapes and features of primitive patterns

3.2 Recognise Primitive Patterns from Time Series Data Displayed at Different Time Scales

We have discussed the DRCD algorithm operating at the basic time scale in detail. In order to get all patterns from when displayed at different time scales, the concept of the *'layer'* is introduced. Suppose a time series data set is displayed at different time scales: *TimeScale1*, *TimeScale2*, and *TimeScale3* (where *TimeScale13* < *TimeScale2* < *TimeScale3*); there are three corresponding layers: *Layer1*, *Layer2*, and *Layer3*; each layer only includes new patterns detected at its corresponding time scale but does not include the patterns that have been detected at other time scales. For example, *Layer1* includes all patterns detected at the basic time scale (1 second). *Layer2* only includes new patterns (if new patterns appear) detected from the same data when displayed at the 5 second time scale but excludes the patterns detected when displayed at the 1 second time scale. Figure 9 presents a systematic algorithm to recognise primitive patterns from time series data displayed in different time scales.

To make it clear, let us see how to identify visual patterns at *TimeScale2*:

(1) Use the DRCD algorithm to automatically detect primitive patterns at the basic time scale (*TimeScale1*). To detect basic patterns means categorising the pattern as

a spike, oscillation, or step and determining when it starts and ends. We have.

Layer1={Pattern11, Pattern12, ----, Pattern1N} *TimeScale1*

Pattern1i=(namei, start_timei, end_timei, max_valuei, min_valuei) (1<=i<=N) (for more details see Figure 11)

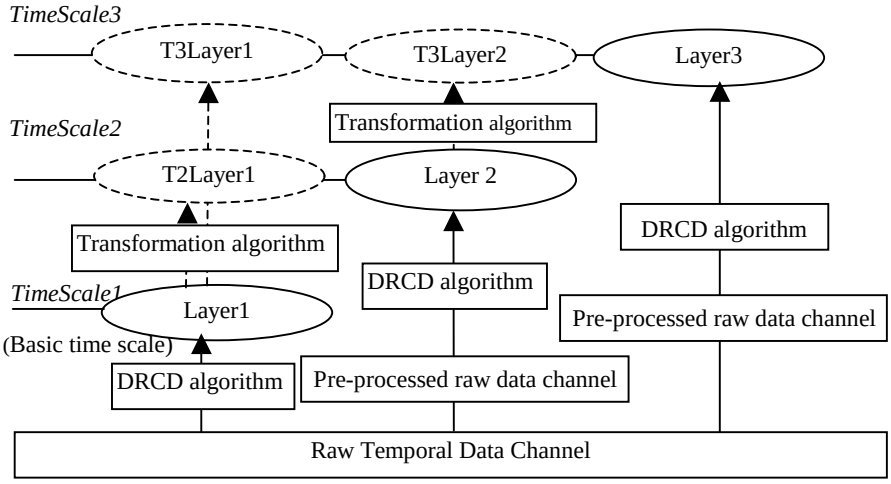


Figure 9: A systematic algorithm to identify all patterns displayed at different time scales

(2) Transform the results obtained at *TimeScale1* into results at *TimeScale2*. We recognise that some patterns will change their visual shapes at different time scales, while others will not. Generally there is a transformation algorithm between patterns displayed at *TimeScale1* and patterns displayed at *TimeScale2*.

Patterns {Pattern11, Pattern12, ----, Pattern1N}

displayed at *TimeScale1* (Layer1) can be transformed into

Patterns {T2Pattern11, T2Pattern12, ----, T2Pattern1N}

displayed at *TimeScale2* (T2Layer1)

In the gas turbine domain, for example, we propose the following transformation algorithm at *TimeScale2*.

	TimeScale1		*TimeScale2*	Transformation condition
Name:	Oscillation	----->	Spike	when (Duration/TimeScale2) <=2

where Duration= End_time - Start_time

(3) Repeat (1) in *TimeScale2* to detect new primitive patterns that did not appear in *TimeScale1*.

Patterns that appear at a lower time scale (*TimeScale1*) will also appear at a higher time scale (*TimeScale2*) possibly with a different visual shape, so we need to avoid detecting primitive patterns that were detected in the lower time scale (*TimeScale1*). The key question is how to prevent such patterns from being re-detected in the higher time scale (*TimeScale2*). Our solution is to pre-process the

raw temporal data channels to use suitable substitute intervals of patterns to replace the intervals of patterns detected in *TimeScale1* so that pattern detected in *TimeScale1* will not be re-detected in *TimeScale2*. This substitution can be performed in a number of different ways. For example, if we represent a primitive pattern such as a spike or an oscillation as:

Pattern = {StartTime, Interval, Y(t)}

the pattern can be replaced by

SubstitutePattern={StartTime, Interval, Y'(t)}

where Y'(t) is a linear interpolation over the interval between the start and end values. When the DRCD algorithm is applied to the pre-processed raw temporal data channels, the only thing we need to do is choose a suitable window size in the rapid change detector according to the time scale at which the raw temporal data channels will be displayed.

In this way, we can obtain new primitive patterns at *TimeScale2*.

Layer2={Pattern21, Pattern22, ---, Pattern2M} *TimeScale2*

(4) Add the transformed patterns obtained in (2) to the new primitive patterns obtained in (3) to get all patterns at *TimeScale2*.

Using the same method, we can detect all patterns in *TimeScale3*. Figure 10 displays the results from the sample data set given in Figure 1 displayed at the 5 second time scale. A bar '-' in Figure 10 means the detected pattern is a spike; it also indicates the temporal interval occupied by the pattern.

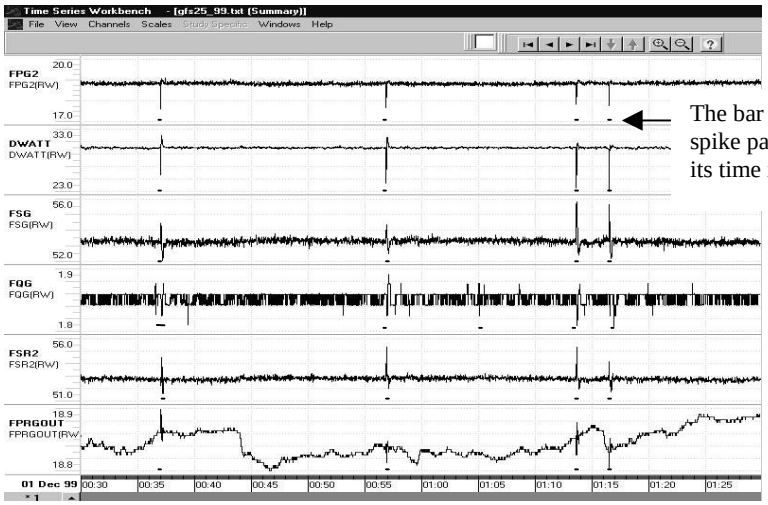


Figure 10: Output of the systematic pattern recognition in a sample data set

(5) Output detected primitive patterns from a temporal data channel in a suitable data structure. Figure 11 gives such data structure representing the patterns detected from one data channel in the sample temporal data set.

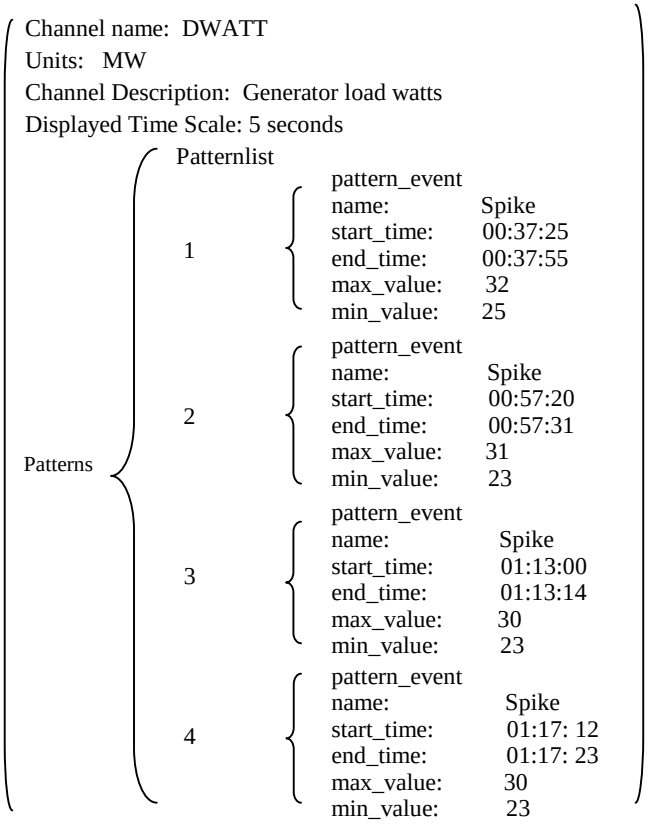


Figure 11: Data structure representing detected patterns from one data channel

4. Evaluation of the Systematic Pattern Recognition Method

We can compare patterns identified by domain experts with patterns that are identified by the above algorithm from the same temporal data channel when displayed at the same time resolution. Suppose the expert identifies a set of patterns Pe={Pe1, Pe2, ---, Pem}. And the algorithm identifies another set of patterns Pc={Pc1, Pc2, ---, Pcn}. We use the usual definitions:

True positives:

$$TPi = \begin{cases} 1 & \text{when } Pci \in Pe \quad (1<=i<=n) \\ 0 & \text{otherwise} \end{cases}$$

False negatives:

$$FNi = \begin{cases} 1 & \text{when } Pei \notin Pc \quad (1<=i<=m) \\ 0 & \text{otherwise} \end{cases}$$

False positives:

$$FPi = \begin{cases} 1 & \text{when } Pci \notin Pe \quad (1<=i<=n) \\ 0 & \text{otherwise} \end{cases}$$

We then have the following derived metrics, where the Robust Correctness is applied at the 90% confidence level:

$$Correctness = \frac{\sum_{i=1}^{n} TPi}{\sum_{i=1}^{n} TPi + \sum_{j=1}^{n} FPj} \times 100\% \qquad Correctness = \frac{\sum_{i=1}^{n} TPi}{\sum_{i=1}^{n} TPi + \sum_{j=1}^{n} FPj} \times 100\%$$

$$RobustCorrectness = \frac{\sum_{i=1}^{n} TPi - 9\sum_{j=1}^{n} FPj}{\sum_{i=1}^{n} TPi + \sum_{j=1}^{m} FNj} \times 100\%$$

In the gas turbine domain, spikes, oscillations, and steps can collectively be categorised as 'transitory deviations'. It is therefore possible to carry out our evaluation at the level of the super-class 'transitory deviations' as well as for each of the three sub-classes.

Here we only demonstrate how we will use these measurements to evaluate the systematic pattern recognition method on spikes. In the scenario in Figure 1, the expert marked up the following spikes:

Se={ SE^1_{FPG2}, SE^2_{FPG2}, SE^3_{FPG2} , SE^4_{FPG2}, SE^1_{DWATT}, SE^2_{DWATT}, SE^3_{DWATT}, SE^4_{DWATT}, SE^1_{FSG}, SE^2_{FSG}, SE^3_{FSG}, SE^4_{FSG}, SE^1_{FQG}, SE^2_{FG}, SE^3_{FQG}, SE^4_{FQG}, SE^5_{FQG}, SE^1_{FSR2}, SE^2_{FSR2}, SE^3_{FSR2}, SE^4_{FSR2}, $SE^1_{FPRGOUT}$, $SE^2_{FPRGOUT}$, $SE^3_{FPRGOUT}$, $SE^4_{FPRGOUT}$}

In the same scenario the algorithm derived the results shown in Figure 10:

Sc={ SC^1_{FPG2}, SC^2_{FPG2}, SC^3_{FPG2}, SC^4_{FPG2}, SC^1_{DWATT}, SC^2_{DWATT}, SC^3_{DWATT}, SC^4_{DWATT}, SC^1_{FSG}, SC^2_{FSG}, SC^3_{FSG}, SC^4_{FSG}, SC^1_{FQG}, SC^2_{FQG}, SC^3_{FQG}, SC^4_{FQG}, SC^5_{FQG}, SC^1_{FSR2}, SC^2_{FSR2}, SC^3_{FSR2}, SC^4_{FSR2}, $SC^1_{FPRGOUT}$, $SC^2_{FPRGOUT}$, $SC^3_{FPRGOUT}$, $SC^4_{FPRGOUT}$}

From the two metrics, we have:

TP={1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1}

FN={0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}

FP={0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0}

So we have the derived metrics.

Completeness = (24/(24+1)) * 100% = 96%

Correctness = (24/(24+1)) * 100% = 96%

RobustCorrectness = ((24-9)/(24+1)) * 100% = 60%

We have used this evaluation method on other time series data sets selected randomly from the *Tiger* data archive and have obtained the results given in Table 2. We consider that these preliminary results are promising and are extending the evaluation to more data sets.

Data Source Turbine name: Aylesford (All data sets below are from TIGER3 provided by IA Ltd.)	Evaluation Results (when temporal data sets are displayed at 5s time scale)		
	Completeness	Correctness	Robust Correctness
Date: 25/11/1999. Start time: 12:00 Duration: 3 hours (TTXD1-5)	94.7% (spikes)	100% (spikes)	94.7% (spikes)
Date: 25/11/1999. Start time: 15:00 Duration: 3 hours (TTXD1-5)	94.1% (oscillations)	100% (oscillations)	94.1% (oscillations)
Date: 25/11/1999. Start time: 15:00 Duration: 3 hours (TTXD1-5)	100% (spikes)	97.7% (spikes)	78.6 (spikes)
Date: 17/11/1999. Start time: 12:00 Duration: 3 hours (TTXD1-5)	100% (steps)	100% (steps)	100% (steps)

Table 2: Evaluation results on some randomly selected temporal data sets

5. Related Work

A variety of methods for spike detection exist. A standard spike detection procedure compares the signal value (or a function thereof) with a preset threshold. When the threshold is crossed, a spike is detected. The threshold value is usually based on the estimate of the signal variance and the threshold value is often adjusted by the user. Setting the threshold too high leads to missed spikes, while setting it too low leads to detection of many small-amplitude spikes that cannot be classified. Many authors have discussed the selection or automatic determination of optimal threshold values. For example, an optimal setting of the threshold was discussed to improve spike-sorting in tetrode recordings [5]. A system for on-line spike detection and analysis was introduced in [6]. In this system, spikes are detected by an adaptive threshold which varies as a function of signal mean and its variability. Since threshold value in this system is determined automatically by the signal-to-noise ratio analysis, the user is not actively involved in controlling its level. But when we try to communicate time series data in terms of interesting patterns, we face a quite new problem. Since patterns in temporal data channels are very complicated, including not only spikes, but also oscillations and steps, a standard spike detection procedure can not be easily applied in such complicated situations. Also, the visual shapes of patterns will have different appearances appear when displayed at different time scales, which makes the task pattern recognition very challenging.

6. Conclusions

In this paper, we have described a systematic method of recognising primitive patterns from time series data sets when displayed at different time scales in order to summarise these temporal data sets precisely and concisely. Raw temporal data channels are processed into rapid change channels through a rapid change detector and context sensitive thresholds are used in a dynamic limit checker to identify interesting candidates. Through concatination, extension and projection, real patterns in the raw temporal data channels are determined. The concept of 'layers' has been introduced and its benefits are that different layers are transparent and DRCD algorithm can be reused in different layers.

The evaluation method is currently being applied to other time series data sets in the gas turbine domain and the preliminary results show that this systematic pattern recognition method seems promising.

Acknowledgements

We are grateful to our collaborators at IA (Intelligent Applications), especially Dr. Rob Milne and Dr. John Aylett, for their contributions to knowledge acquisition. This project is supported by the UK EPSRC under grant GR/M76881. Jin Yu is also funded by the China Scholarship Council (CSC).

References

1. Milne R, Trave-Massuyes L. Model based aspects of the TIGER gas turbine condition monitoring system. LAAS Report, 1997
2. Yu J, Hunter J, Reiter E, and Sripada S. An approach to generating summaries of time series data in the gas turbine domain. In Proceedings of ICII2001, Beijing, 2001, pp 44-51
3. Karen L, Harbison-Briggs M K. Knowledge acquisition: principles and guidelines. Prentice Hall, 1989
4. Shahar Y. A framework for knowledge-based temporal abstraction. Artificial Intelligence, 1997; 90(1-2): pp 79-133
5. Rebrik SP, Wright BD, AA Emondi, and KD Miller. Cross channel correlations in tetrode recordings: implications for spike-sorting. In Proceedings of Computation and Neural Systems Meeting, Big Sky Montana, 1999
6. Soto E, Manjarrez E and Vega R. A microcomputer program for automated neuronal spike detection and analysis. International Journal of Medical Informatics, 1997; 44: pp 203-212