A neural network for mining large volumes of time series data

Bojian Liang and James Austin
Advanced Computer Architectures Group
Department of Computer Science
University of York
York, YO10 5DD, UK
bojian,austin@cs.york.ac.uk

Abstract— **Efficiently mining large volumes of time series data is amongst the most challenging problems that are fundamental in many fields such as industrial process monitoring, medical data analysis and business forecasting. This paper discusses a high-performance neural network for mining large time series data set and some practical issues on time series data mining. Examples of how this technology is used to search the engine data within a major UK eScience Grid project (DAME) for supporting the maintenance of Rolls-Royce aero-engine are presented.**

I. Introduction

With the development of digital electronics and computer hardware/software, data acquisition and storage are becoming easy and inexpensive. Many databases today range in size up to terabytes level. The development of intelligent data analysis methodologies for efficiently discovering useful knowledge from large volume data sets has attracted more and more attention. Data mining is the process of efficiently extract unknown information from databases. It is a powerful tool for complex data analysis from large as well as small data set in a reasonable time scale. A good review of data mining can be found in [1] [6].

A time series signal is a set of observations of a variable at successive times. Mining time series data can reveal important knowledge for the subsequent processing such as indexing, clustering, regression and classification. They can be found in applications such as stock market price analysis/prediction, electrocardiogram (ECG) data analysis, weather forecasting, as well as many other industrial process-monitoring systems. One of the big challenges of mining time series data is their size and dimensionality. The performance of time series data mining algorithms becomes one of the key issues. Many papers have introduced new algorithms and applications on time series data mining. However, few of them are actually tested with large enough real world datasets. A recent survey on time series data mining can be found in [7]. As the author stated, many algorithms published have very little utility.

In this paper, we will present a neural network for fast time series similarity matching. The algorithm has been successfully implemented and used in a major UK eScience pilot project (DAME[1]) for analysis of Rolls-Royce aero-engine data.

The rest of the paper is organised as follows. In Section-II, we state some of the practical problems in mining time series. These problems are common in many applications as well as in the DAME project. In Section-III, we discuss a neural network based on a high–performance neural network called Correlation Matrix Memory (CMM) and how the time series is encoded to suit the operations of the network. In Section-IV we demonstrate the application of the technology using a tool called *Signal Data Explorer* (SDE) developed within the DAME project. In the last section, we summarise this work.

II. Background

In an engine health monitoring system, indicators of the health of an operating engine will be embedded in patterns of vibration and performance data which are represented as time series. An aero-engine on a civil airliner is capable of generating 1GB of vibration and performance data per engine per flight. A crucial part of engine health monitoring is the detection of earliest possible signs of the presence of features known to be associated with fault conditions and for deviation from a model of the known operation of the engine. The difficulties of such pattern search lie in the fact that the search is over terabytes of data and the limited knowledge of patterns representing known fault conditions. Many conventional techniques are no longer adequate for searching a non-precisely defined fault pattern within a huge collection of data in response time that meets the operational demands.

Grid technology allows the DAME system to be highly distributed and yet still appear a single coherent system. A high–performance search engine built within the DAME system provides a distributed, parallel and efficient pattern match service. In this paper, we will focus on a neural network deployed in the DAME system for time series pattern searching. The relevant issues which may be generic in other time series data mining problems are also discussed. The

[1] The details of DAME (Distributed Aircraft Maintenance Environment) can be found on web site: http://www.cs.york.ac.uk/dame/

0-7803-9484-4/05/$20.00 ©2005 IEEE

discussion of details of other techniques used in the DAME systems is out of the scope of this paper.

A. Similarity searching over time series

The most common operation of mining time series is searching for similar patterns from time series datasets (similarity searching). The basic form of similarity searching over time series is *subsequence matching* which assumes that the length of a main series is longer than that of the query and the query is used to scan through the whole length of the main series. Based on a pre-defined rule, the measure of each match (the score of matching) in a similarity searching provides essential information for the subsequent operation such as classification or prediction. Given the length of a main time series L and the length of a query series l, ($L \geq l$), the computational complexity of subsequence matching on one set of data is $O(l(L - l + 1))$. This is one of the key challenges facing the performance of subsequence matching on huge collection of time series datasets.

To reduce the computational complexity, many dimensionality reduction algorithms have been reported [8][10]. Most of these algorithms are *information non-preserving* approaches. They are highly application specific and not applicable in many other cases. Re-sampling the over-sampled time series is one way to reduce the dimensionality of the series. It is an information preserving approach. Unfortunately, this is not always applicable. The arbitrary length of query is another problem to be tackled in the DAME pattern matching system since the lengths of fault patterns vary and may not be defined in advanced. There are only a few authors who have considered this issue [8][9]. In the DAME project, due to the limited knowledge and the uncertainty of the engine fault conditions, there is no way to pre-construct a dimensionality reduced dataset for general pattern searching.

A system may "hit" a similarity in a data set. The hit is the local maximum of the matches which is within a given tolerance threshold. The output of a similarity searching may include all of hits or may generate only one hit (the best match) per dataset. Together with the scores is the position of the match that represents the time instance where the event (the fault pattern) happened.

B. The measures

Measures are crucial for pattern matching and can be highly application specific. It is important to choose a proper measure for mining time series according to the application. Many researchers do not use a correct measure when mining time series. For example an attempt [7] was made to compare the "usefulness" of measures that belonged to different classes and designed to be used under different circumstances. This comparison is not meaningful in general. For instance, Euclidean distance may work well on searching for an identical shape (there is no amplitude and time scaling on the query and main series). Correlation is a well known measure for matching signal with scaling on amplitude. For signal with time variation/scaling, dynamic time warping may be a better choice. Hence, there is no single measure that fits all. The use of a measure is the balance of accuracy, performance and many other conditions applied by an application.

In the DAME system, both the amplitude and shape of the engine data are important. The system is currently designed to support both correlation and Euclidean distance measures which can be selected by a domain export before the searching is invoked.

A time series with length n can be represented as a n–dimensional vector. Given two sequences $\mathbf{Q} = \{q_1, q_2, ..., q_n\}$ and $\mathbf{S} = \{s_1, s_2, ..., s_n\}$, correlation coefficient of the two time series is defined as:

$$corr(\mathbf{Q}, \mathbf{S}) = \frac{\sum_{i=1}^{n} s_i q_i}{\sqrt{\sum_{i=1}^{n} q_i^2 \sum_{i=1}^{n} s_i^2}}, \quad (1)$$

$$= \frac{\mathbf{Q} \bullet \mathbf{S}}{\|\mathbf{Q}\| \|\mathbf{S}\|} \quad (2)$$

where $\bullet$ denotes the inner product of vectors $\mathbf{Q}$ and $\mathbf{S}$. An important physical meaning of (2) is that **the correlation coefficient of two vectors is the cosine of the angle between the vectors**.

The correlation coefficient of two vectors is independent of both origin and scale since it is only defined by the angle between the vectors. By interpreting the correlation coefficient in this way, the estimation of measurement error introduced by the quantisation noise becomes much easier. We will discuss this in the next section. Eq.(2) involves the normalisation operation for both the query ($\mathbf{Q}$) and the main series ($\mathbf{S}$). For subsequence matching, the query can be normalised in advanced of the search in order to simplify the computation. Part of the square sum of the main series segment can be re-used for the next subsequence. This can further optimise the computation.

The Euclidean distance of two vectors is defined as:

$$D(\mathbf{Q}, \mathbf{S}) = \|\mathbf{Q} - \mathbf{S}\| \quad (3)$$

Using the same components of (2), Euclidean distance measure can be re-written as:

$$D(\mathbf{Q}, \mathbf{S}) = \sqrt{\|\mathbf{Q}\|^2 + \|\mathbf{S}\|^2 - 2\mathbf{Q} \bullet \mathbf{S}}. \quad (4)$$

This provides means to compute the two measures at the same time without significantly increasing the total searching time.

C. Multiple searching

The status of many industrial processes is often characterised by more than one variable over time. For example, in aero-engine diagnosis, several parameters over time may be required to describe one condition. Searching for multiple patterns from time series data is a more complicated problem than searching for a single pattern. This is because it is not just a simple combination of separate single pattern searching procedures but requires efficient collaboration of all searching procedures, intelligent management of the searching constraints applied and a clear, meaningful output is generated.

To search for multiple variables, records of the same group in databases must be associated by some kind of identification parameters such as group-name or index. All the records in the same group are synchronised by time. In this way, the search engine can locate the records and compute the final results correctly. An *event window*, T_w, is a time interval including a set of hits across the variables. T_w is an important search constraint to define a set of valid hits for multiple searches. Properly used, the event window constraint can greatly reduce the searching space. Each set of valid hits of a multiple search procedure with n variables defines an n–dimensional output vector. Conventional processing such as classification or indexing can then be applied to the output vector to obtain further information. In a particular case, multiple search can be done by a simple neural network. However, in the DAME system, this is done by a *pattern matching controller* (**PMC**) on top of the neural networks and other search methods.

III. Mining time series using neural networks

A common methodology for mining large volumes of data is by deploying an imprecise but fast approximate search in order to quickly obtain a small set of candidates. Then the conventional methods may be applied to compute the final output from the candidates (post-processing). Neural networks can be used to efficiently extract similar patterns from time series. One of the advantages of the technique used here is that it can be easily implemented in hardware.

Advanced Uncertain Reasoning Architecture (AURA) is a parallel pattern matching technology developed as a set of general–purpose methods for searching large unstructured datasets. The technology is based on a high–performance neural network called Correlation Matrix Memory (CMM) [11][12]. AURA provides fast approximate search and match operations on large unstructured datasets. The latest developments of AURA include the new AURA C++ library, the hardware system (PERSENCE-II) and a Grid enabled AURA pattern match service. There are a growing number of applications for AURA. These include a postal address matching, high-speed rule-matching systems [13], fraudulent benefit claims detection system (FEDAURA), structure–matching (e.g 3D molecular structures) [14], human face recognition and trademark–database searching [15].

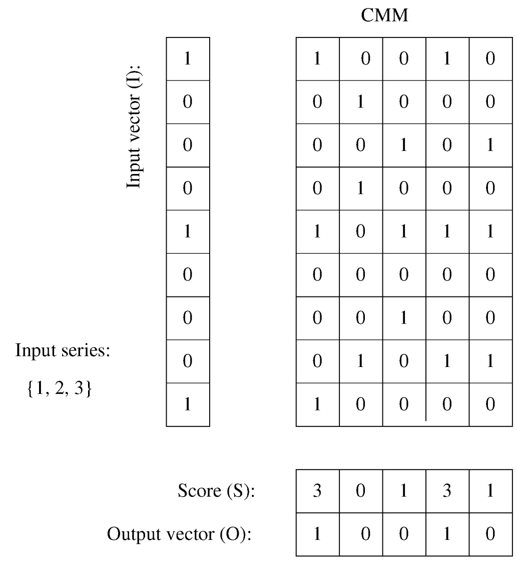
A. Correlation matrix memories

A Correlation Matrix Memory (CMM) is a type of binary associative neural network. A CMM with m columns and n rows can be represented by a $n \times m$ matrix with binary weights, $\mathbf{M}$. Given a n–dimensional binary input pattern $\mathbf{I}_k$ and a m–dimensional binary associated output pattern $\mathbf{O}_k$, the CMM learning can be written as:

$$\mathbf{M}_k = \mathbf{M}_{k-1} \cup \mathbf{I}_k^T \mathbf{O}_k. \tag{5}$$

Where, $\mathbf{M}_{k-1}$ and $\mathbf{M}_k$ are the CMM pattern before and after the k^{th} learning operation. $\cup$ denotes a logical OR operation. A CMM recall $\mathbf{S}_i$ referring to input pattern $\mathbf{I}_i$ is given by:

$$\mathbf{S}_i = \mathbf{I}_i \mathbf{M}. \tag{6}$$



Input vector (I)	CMM				
1	1	0	0	1	0
0	0	1	0	0	0
0	0	0	1	0	1
0	0	1	0	0	0
1	1	0	1	1	1
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	1	1
1	1	0	0	0	0
Score (S):	3	0	1	3	1
Output vector (O):	1	0	0	1	0

Fig. 1. An example of CMM recall

The CMM recall $\mathbf{S}_i$ is a m–dimensional integer vector. Each element of $\mathbf{S}_i$ is called **the score of a CMM matching**. $\mathbf{S}_i$ can be thresholded to from a binary vector, $\mathbf{O}_i$, which is the possible matches. $\mathbf{S}_i$ can also be used as integer vector to represent the measure of the match. Fig.1 shows an simplified example of a CMM recall. The CMM is first trained as shown in the figure. An input vector is then applied to the trained CMM. Each row of the CMM is activated if the relevant element of the input vector is set. $\mathbf{S}$ is the sum of all activated rows of the CMM. $\mathbf{O}$ is a binary vector by applying threshold 2 to each element of vector $\mathbf{S}$. Column vectors of the CMM is said approximately similar to the input vector if the relevant element of $\mathbf{O}$ is 1.

The computational efficiency of a CMM shown in Fig.1 lies in the fact that only the necessary nodes in the CMM pattern are processed. In the above example, there are only three rows with 2, 4 and 1 nodes on each of the rows respectively set as 1. To compute all of the 5 outputs, we need only to go through $2 + 4 + 1 = 7$ rather than $5 \times 9 = 45$ nodes. To effectively match two time series with a CMM, an efficient encoding method must be applied in order to optimise the searching.

B. The hardware implementation of AURA

The binary neural network based architecture used in AURA can be efficiently implemented in hardware. PRESENCE II (**P**a**R**all**E**l **S**tructur**E**d **N**eural **C**omputing **E**ngine) is the latest generation of AURA hardware implementation. At the heart of the card is a large Field Programmable Gate Array. The FPGA interfaces to a fast Digital Signal Processor, up to 4GBs of SDRAM memory, two independent fast Zero-Bus-Turnaround memories, dual high-speed data channels, Sundance digital I/O header and a mezzanine expansion card connector. With each on-board resource given an independent interface to the FPGA, the designer is able to implement bus structures of choice, rather than the board itself imposing a fixed wiring scheme.

Additional host system resources (system memory, I/O devices etc) are accessible via the PCI bus as bus master. The details of the AURA hardware implementation are out of the scope of this paper.

C. Encoding the time series for CMM matching

Most of the current dimensional reduction techniques for time series pattern match are the information non-preserving approaches. Due to missing information, a lower bound applied does not guarantee a small enough number of approximate matching results that are within the boundary, i.e. the lower bound is not tight enough. The worst case is that the approximate search returns the whole set of original data. Hence the post-processing will work on the full set of data and the effort for approximate search becomes nugatory. As a result, the total searching time may not be improved. The second problem of such techniques is that a particular lower bound (as well as the dimensional reduction algorithm) is usually valid only for a specific measure, e.g. Euclidean distance measure. To work on a different measure, different dimensional reduction algorithms are required. Such algorithms are not always available and the use of multiple sets of dimensional reduced data simply degrades the system performance and increases the requirement for the system storages. These techniques do not support either arbitrary query length or the desired measures. To the best of our knowledge, there are no existing approaches for time series pattern matching that meet well the DAME system requirements.

In the DAME system, a differential encoding technique is used to encode the time series data. A relevant low quantisation precision is used in order to further reduce the repeated sample values on the data. This will also reduce the computational complexity on the CMM matching. The differential code is then loaded to a CMM and the pattern match is done by applying the AURA technique. Instead of applying a measure which lower bounds the measures on the original data [7], we prove that correlation measurement errors are predictable and it is the function of the quantisation precision. Hence, one can control the quantisation precision to ensure that a small enough set of data returns from the approximate matching to the post-processing. In many cases, the post-processing can be ignored by using a proper quantisation precision on encoding the data.

1) Differential encoding: The differential encoding is widely used for data compression. Differential encoding does not directly compress the data but reduce the repeated number of sample values. A reasonably low quantisation precision differential code can further reduce the number of repeated values in the serial hence reduce the computation load for the search. If the measurement error boundary of a search is predictable, balance between the processing time of approximate search and post-processing can be made to optimise the overall system performance.

Given a series of real value data $\mathbf{X} = \{x_1, x_2, \cdots, x_n\}$, the differential code is defined as:

$$d_i = ADC(x_i - x_{i-1}), \tag{7}$$

where $ADC()$ denotes a analogue–to–digit conversion. The shortcoming of using (7) is the uncontrollable accumulation encoding error. If the quantisation error is δ and the length of data is n, then the maximum encoding error will be $n\delta$. This will cause serious distortion on the signal and lead to mismatching. To overcome the problem of (7), feedback encoding technique must be used.

There are several standard series signal encoding techniques in communication systems. The popular types of these are PCM (pulse code modulation), DPCM (differential pulse code modulation) and DM (delta modulation)[4], [2], [3]. PCM directly encodes the sample values of a signal while DPCM and DM encode the difference of the values. A start value z_0 is set before the encoding, a predictor (usually a feedback DAC) generates the estimation signal z_i and the difference of current input signal x_i and z_{i-1}, $\delta_i = x_i - z_{i-1}$, is encoded. After a short start interval, the differential code will trace the input signal within a given tolerance (e.g. half quantisation level). The differential code series c_i is given by:

$$c_i = ADC(x_i - z_{i-1}). \tag{8}$$

where z_{i-1} is the estimation of input signal x_i. Let Δ be the quantisation step. z_i $(i > 0)$ is given by

$$z_i = z_0 + \Delta \sum_{n=0}^{i-1} c_n. \tag{9}$$

DM is a special case of DPCM where there are only two quantisation levels.

The quantisation error is the difference between the original input data and the encoding data,

$$e_i = x_i - z_i. \tag{10}$$

The time series $\mathbf{e}_q = \{e_i\}$ is called the quantisation noise. It is common to assume that the quantisation noise is additive. Details of quantisation noise analysis can be found on [4], [2], [5] and elsewhere. The power of the quantisation noise is represented by variance,

$$\sigma^2 = E(\|\mathbf{e}_q\|^2). \tag{11}$$

For a uniform quantiser, the quantisation noise level is given by,

$$\sigma^2 = \frac{\Delta^2}{12} \tag{12}$$

If the noise from the signal channel can be neglected, quantisation is the only error source. The signal–to–noise ratio (power) is then given by,

$$SNR = 2^{2m}, \tag{13}$$

where m is the quantisation precision[4]. (13) is useful for the estimation of boundary of the measurement errors.

2) Estimation of the error boundary : In Section-II-B, correlation coefficient is interpreted as the cosine of an angle between the two vectors. In this section, we prove that the boundary of the measurement errors for correlation is defined by the quantisation precision.

Assuming α_q is the angle between the two quantised vectors and α is the angle of the two input vectors, the measurement error boundary of estimate α by α_q is,

$$\|\alpha_q - \alpha\| \le \varepsilon_q + \varepsilon_s, \tag{14}$$

where ε_q and ε_s are the maximum measurement errors on the angle introduced by the quantisation errors. As shown in Fig.2, the error angle is defined by,

$$\sin\varepsilon = \frac{1}{\sqrt{SNR}}, \tag{15}$$

Substitute (13) to the above equation, we have

$$\varepsilon = \arcsin 2^{-m}. \tag{16}$$

For example, encoding the data by 4-bit quantisation precision, the error angle is about $3.58°$. If we want to search for similar patterns with similarity measure (correlation coefficient) larger than 0.9 ($\alpha \le 25.8°$), the boundary for approximate match is $\cos(25.8 + 2 \times 3.58) = 0.84$. All matches with measure larger than 0.84 will be passed to the post-processing module to obtain the final results.

It is clear that the error angle becomes larger when the signal–to–noise ratio decreases. In practice, (16) can be replaced by a minimum ratio SNR_{min}. Notice (13) does not involve the quantisation step Δ. One can choose a proper quantisation step for each input signal so that the SNRs are higher than a given value, SNR_{min}, for all data.

For Euclidean distance measure, measurement errors introduced by quantisation noise is defined as:

$$e_E \le \|\mathbf{e}_q\| + \|\mathbf{e}_s\|. \tag{17}$$

The proof of (17) is straightforward by using the triangle inequality. Assuming the maximum quantisation error is $\frac{\Delta}{2}$, the boundary of measurement errors of Euclidean distance measure is estimated by,

$$e_E \le \sqrt{n}\,\Delta, \tag{18}$$

where n is the length of the vector.

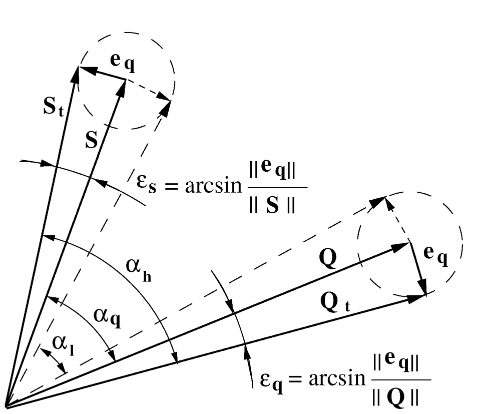

e_q
S_t
S
ε_s = arcsin ||e_q|| / || S ||
α_h
Q
e_q
Q_t
α_q
α_l
ε_q = arcsin ||e_q|| / || Q ||

Fig. 2. The boundary of correlation measurement error

IV. Evaluation and experimental results

A. Data reduction

Firstly we show how the differential encoding reduces the non–zero data size. We use the term "saturation" to measure this efficiency. Saturation is the percentage of the non–zero data points referring to the whole data set. If a time series has 1,000,000 sample points and after encoding there are 400,000 points are zero, we call that the saturation of the data set is 60 percent. Given the saturation of the query vector p_q and the main series p_m, the computational complexity, $O(p)$, of a CMM is estimated as:

$$O(p) \le Min(p_q, p_m). \tag{19}$$

In fact, it is the probability when both a node in the CMM and the relevant entry of the query vector take non–zero values.

Table-I lists the saturation values of 10 variables from the DAME system. The saturations of original data before encoding are 100 percent. Each set of data contains 24 records with 1285200 sample points. All data were encoded to differential code with 8-bit quantisation precision. The saturations of data passed a Gaussian filter are also listed in the table for comparison. It is easy to see that the saturation of the code is greatly reduced after the encoding.

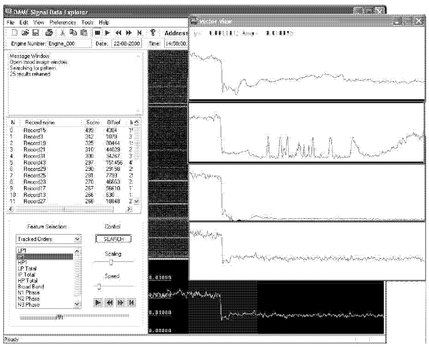
TABLE I
Data saturation comparison

Data name	S_1	S_2	S_3	S_4	S_5
Without filter	0.64	0.24	0.48	0.46	0.6
With filter	0.75	0.15	0.49	0.23	0.69
Data name	S_6	S_7	S_8	S_9	S_{10}
Without filter	0.21	0.50	0.33	0.28	0.18
With filter	0.15	0.58	0.19	0.16	0.06

B. Performance and application example

The technology has been successfully used for engine data analysis in the DAME system. A user interface and other tools for signal processing are integrated into a tool called *Signal Data Explorer* (SDE). The search engines in the DAME system are distributed and can be invoked locally or remotely (over the Grid) by the SDE. At the moment we have not fully tested the hardware implementation. For a software implementation, the system can search 10GB of raw ZMOD engine data in less than 1 second on a standard PC (1.8GHz P4 512MB, using subsequence matching. data rate 5Hz and the duration of the query series is 100 seconds and the average length of the records is 2.3 hours)

The following screen-shots show particular examples of the operation of the SDE. The first example shows searching for similar pattern with the input template from large volume of history data store. The pattern segment that may belong to a fault is selected by domain export and is highlighted as the search template. The first four best matching are displayed in the pop up windows. Search results are listed in the results window, as shown in Fig.3. The second example shows a search for a particular operation pattern using the multiple search functionality of the system. The two query patterns are shown in Fig.4. The search constraint is to find the top pattern on Fig.4 and then find the pattern on bottom of the figure within the same record. The event window is set as 150 seconds (time between the two events). The four best matches are shown in Fig.5. The search process collects all operation patterns from the history data so that the values of variable is firstly decreased and then increased within 150 seconds. A conventional approach can then be applied to the results to obtain a more precise information of the pattern.



Engine Number: Engine_000 Date: 22-08-2000 Time: 14:50:00

Message Window
Open zmod image window.
Searching for pattern.
25 results returned.

N	Record name	Score	Offset
0	Record15	499	4384
1	Record3	342	1078
2	Record19	325	80444
3	Record21	310	44029
4	Record31	300	34267
5	Record43	297	151456
6	Record29	290	29158
7	Record25	281	7799
8	Record23	270	46653
9	Record17	267	56610
10	Record13	266	630
11	Record27	266	18848

Tracked Orders: LP1, IP1, HP1, LP Total, IP Total, HP Total, Broad Band, N1 Phase, N2 Phase, N3 Phase

Ready

Fig. 3. Searching for similar patterns

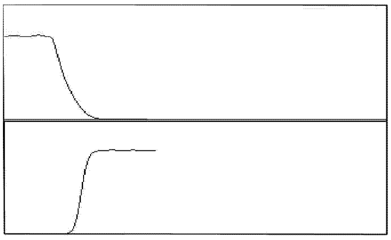
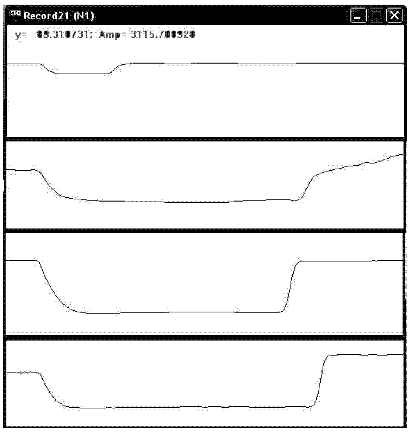


Fig. 4. Application examples 2: Templates for multiple searching



Record21 (N1)

y= 85.310731; Amp= 3115.700528

Fig. 5. Application examples 2: Four best match patterns

V. Conclusions

We have presented a neural network for efficiently searching for similar patterns from large volumes of time series datasets. This work has tackled several problems that are important in mining time series data but have not been undertaken by many researchers. The technique has been successfully used in the DAME system for aero-engine data analysis.

Acknowledgement

This research was supported under the EPSRC e-Science pilot project grant (DAME), GR/R67668/01. DAME was a collaboration between the Universities of York, Leeds, Sheffield and Oxford, Rolls-Royce plc, Data Systems and Solutions LLC and Cybula Ltd.

References

[1] Fayyad, U., Piatetsky-Shapiro, G. Smyth, P. and Uthurasamy, R. : Advances in Knowledge Discovery and Data Mining, AAAI Press/The MIT Press 1996.

[2] Ferrel G. Stremler: Introduction to communication systems, Addison-Wesley Publishing Company, Inc. 1990.

[3] Sinnema, W. and McGovern, T. : Digital, analog, and data communication, Prentice-Hall International, Inc. 1986.

[4] Ziemer, R.E. and Tranter, W.H. : Principles of communications: Systems, Modulation and Noise Houghton MiffLin Company, Boston. 1995.

[5] Gersho, A. : Asymptotically optimal block quantisation, IEEE Transactions on information theory, Vol.IT-25, (1979) 373-380.

[6] Sushmita M, Ewnkw5 K.P and Pabitra M. : Data Mining in Soft Computing Framework: A Survey, IEEE Transactions on neural networks, Vols.13 No.1 (2002), 3–14.

[7] Keogh, E and Kasetty, S. : On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstrating, SIGKDD, Edmonton, Alberta, Canada, (2002) 102-111.

[8] Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. : Locally adaptive dimensionality reduction for index large time series databases. In proceedings of ACM SIGMOD Conference on Management of Data, (2001) 151-162.

[9] Yi, B. and Faloutsos, C. : Efficient retrieval of similar time sequences under time warping, In proceedings of the 14^{th} international Conference on Data Engineering. (1998) 201–220.

[10] Yi, B. and Faloutsos, C. : Fast Time Sequence Indexing for Arbitrary lp Norms, In proceedings of the 26^{th} international Conference on Very Large Databases. (2000) 385–394.

[11] Austin, J.: Distributed associative memories for high speed symbolic reasoning. In, IJCAI Working Notes of workshop on connectionist-Symbolic Integration, ¿From Unified to Hybrid Approaches.(1995) 87-93.

[12] Austin, J., Kennedy, J., and Lees, K. : The advanced uncertain reasoning architecture. In Weightless Neural Network Workshop, 1995.

[13] Austin, J., Kennedy, J., and Lees, K. : A neural architecture for fast rule matching. In Artificial Neural Networks and Expert Systems Conference (ANNES 95), Dunedin, New Zealand, December 1995.

[14] Turner, A., and Austin, J. : Performance evaluation of a fast chemical structure matching method using distributed neural relaxation. In fourth International Conference on Knowledge-based Intelligent Engineering Systems, August 2000.

[15] Alwis, S.,and Austin, J. : A novel architecture for trade-mark image retrieval systems. In Electronic Workshops in Computing. Springer, 1998.