Maximum Likelihood Hebbian Learning Based Retrieval Method for CBR Systems

J. M. Corchado[1], E. S. Corchado[2], J. Aiken[3], C. Fyfe[4], F. Fdez-Riverola[5] and M. Glez-Bedia[1]

[1]Dept. de Informática y Automática, University of Salamanca, Plaza de la Merced s/n, 37008,
Salamanca, Spain
corchado@usal.es
[2]Dept. de Ingeniería Civil, University of Burgos, Esc. Politécnica Superior, Edificio C, C/
Francisco de Vitoria, Burgos, Spain
[3]Plymouth Marine Laboratory, Prospect Place, West Hoe, Plymouth, PL1 - 3DH, UK
[4]Computing and Information System Dept. University of Paisley, PA1-2BE, Paisley, UK
[5]Dept. Informática. University of Vigo, Campus As Lagoas, Ourense, Spain

Abstract: CBR systems are normally used to assist experts in the resolution of problems. During the last few years researchers have been working in the development of techniques to automate the reasoning stages identified in this methodology. This paper presents a Maximum Likelihood Hebbian Learning-based method that automates the organisation of cases and the retrieval stage of case-based reasoning systems. The proposed methodology has been derived as an extension of the Principal Component Analysis, and groups similar cases, identifying clusters automatically in a data set in an unsupervised mode. The method has been successfully used to completely automate the reasoning process of an oceanographic forecasting system and to improve its performance.

1 Introduction

Case based reasoning (CBR) systems have been successfully used in several domains such as diagnosis, prediction, control and planning [1], [10], [23], [26]. However, a major problem with these systems is the difficulty in case retrieval and case matching when the number of cases increases; large case bases are difficult to handle and require efficient indexing mechanisms and optimised retrieval algorithms, as explained later. Also there are very few standard techniques to automate their construction, since each problem may be represented by a different data set and requires a customised solution. This paper presents a method that can be used to alleviate these problems.

In the CBR cycle there is normally some human interaction. Whilst case retrieval and reuse may be automated, case revision and retention are often undertaken by human experts. This is a current weakness of CBR systems and one of their major challenges. For several years we have been working in the identification of techniques to automate the reasoning cycle of CBR systems [4], [8], [10], [19]. This paper presents a Maximum Likelihood Hebbian Learning (MLHL) based model to automate the process of case indexing and retrieval, that may be used in problems in which the cases are characterised predominantly by numerical information.

Maximum Likelihood Hebbian Learning (MLHL) based models were first developed as an extension of Principal Component Analysis [21], [22]. Maximum Likelihood Hebbian Learning Based Method attempts to identify a small number of data points, which are necessary to solve a particular problem to the required accuracy. These methods have been successfully used in the unsupervised investigation of structure in data sets [2], [3]. We have previously investigated the use of Artificial Neural Networks [8] and Kernel Principal Component Analysis (KPCA) [11], [12] to identify cases, which will be used in a case based reasoning system. In this paper, we present a novel hybrid technique.

Maximum Likelihood Hebbian Learning Based models can be used in case based reasoning systems when cases can be represented in the form of numerical feature vectors, examples of which would be temperature (ºC), distance (m), time (hh,mm,ss), dates (dd,mm,yy) etc. This is normally the case in most instance based reasoning systems (IBR) [19]. Large case/instance bases may have negative consequences for the performance of the CBR/IBR systems. This has been shown in several projects such as INRECA [27] and ORKA [5], [6]. When a CBR system is used in a real time problem, such as the oceanographic one presented latter in this paper, it may not be possible to manage a large case base and the necessary pre-processing algorithms with reasonable computational power. As has been shown in the ORKA project, new and updated cases should be included and maintained in the case base, and obsolete and redundant cases should be eliminated or transformed to maintain a case base with a stable size, in order to control the response time of the system and maintain its efficiency. The transformation of a number of cases into one representative case may help to reduce the volume of information stored in the case base without losing accuracy. The ability of the Maximum Likelihood Hebbian Learning-based methods presented in this paper to cluster cases/instances and to associate cases to clusters can be used to successfully prune the case-base without losing valuable information.

An instance based reasoning system developed for predicting oceanographic time series and to identify oceanographic fronts ahead of an ongoing vessel [4], [5], in real time, will be used to illustrate the efficiency of the solution here discussed. The identification of oceanographic fronts (areas in which two or more water masses converge) is very important for nuclear submarines [6]. This paper first presents the Maximum Likelihood Hebbian Learning Based Method and its theoretical background. The oceanographic problem in which this technique has been used is presented, and finally we show how this approach has been implemented to forecast oceanographic thermal time series in real time.

2 Maximum Likelihood Hebbian Learning Based Method

The use of Maximum Likelihood Hebbian Learning Based Method has been derived from the work of [3], [11], [12], [13], etc. in the field of pattern recognition as an extension of Principal Component Analysis (PCA) [21], [22]. We first review Principal Component Analysis (PCA) which has been the most frequently reported linear operation involving unsupervised learning for data compression, which aims to find that orthogonal basis which maximises the data's variance for a given dimensionality of basis. Then the Exploratory Projection Pursuit (EPP) theory is outlined. It is shown how Maximum Likelihood Hebbian Learning Based Method may be derived from PCA and it could be viewed as a method of performing EPP. Finally we show why Maximum Likelihood Hebbian Learning Based Method is appropriated for this type of problems.

2.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a standard statistical technique for compressing data; it can be shown to give the best linear compression of the data in terms of least mean square error. There are several artificial neural networks which have been shown to perform PCA e.g. [21], [22]. We will apply a negative feedback implementation [9].

The basic PCA network is described by equations (1)-(3). Let us have an N-dimensional input vector at time *t*, ***x***(*t*), and an M-dimensional output vector, ***y***, with W_{ij} being the weight linking input *j* to output *i*. η is a learning rate. Then the activation passing and learning is described by

Feedforward: $$y_i = \sum_{j=1}^{N} W_{ij} x_j \ , \forall i \qquad (1)$$

Feedback: $$e_j = x_j - \sum_{i=1}^{M} W_{ij} y_i \qquad (2)$$

Change weights: $$\Delta W_{ij} = \eta e_j y_i \qquad (3)$$

We can readily show that this algorithm is equivalent to Oja's Subspace Algorithm [21]:

$$\Delta W_{ij} = \eta e_j y_i = \eta (x_j - \sum_k W_{kj} y_k) y_i \qquad (4)$$

and so this network not only causes convergence of the weights but causes the weights to converge to span the subspace of the Principal Components of the input data. We might ask then why we should be interested in the negative feedback formulation rather than the formulation (4) in which the weight change directly uses negative feedback. The answer is that the explicit formation of residuals (2) allows us to consider probability density functions of the residuals in a way which would not be brought to mind if we use (4).

Exploratory Projection Pursuit (EPP) is a more recent statistical method aimed at solving the difficult problem of identifying structure in high dimensional data. It does this by projecting the data onto a low dimensional subspace in which we search for its structure by eye. However not all projections will reveal the data's structure equally well. We therefore define an index that measures how “interesting” a given projection is, and then represent the data in terms of projections that maximise that index.

The first step in our exploratory projection pursuit is to define which indices represent interesting directions. Now “interesting” structure is usually defined with respect to the fact that most projections of high-dimensional data onto arbitrary lines through most multi-dimensional data give almost Gaussian distributions [7]. Therefore if we wish to identify “interesting” features in data, we should look for those directions onto which the data-projections are as far from the Gaussian as possible.

It was shown in [16] that the use of a (non-linear) function creates an algorithm to find those values of W which maximise that function whose derivative is f() under the constraint that W is an orthonormal matrix. This was applied in [9] to the above network in the context of the network performing an Exploratory Projection Pursuit. Thus if we wish to find a direction which maximises the kurtosis of the distribution which is measured by s^4, we will use a function $f(s) \approx s^3$ in the algorithm. If we wish to find that direction with maximum skewness, we use a function $f(s) \approx s^2$ in the algorithm.

2.2 ε-Insensitive Hebbian Learning

It has been shown [28] that the nonlinear PCA rule

$$\Delta W_{ij} = \eta\left(x_j f(y_i) - f(y_i)\sum_k W_{kj} f(y_k) \right) \tag{5}$$

can be derived as an approximation to the best non-linear compression of the data. Thus we may start with a cost function

$$J(W) = 1^T E\left\{ \left(\mathbf{x} - Wf\left(W^T\mathbf{x}\right)\right)^2 \right\} \tag{6}$$

which we minimise to get the rule (5). [18] used the residual in the linear version of (6) to define a cost function of the residual

$$J = f_1(\mathbf{e}) = f_1(\mathbf{x} - W\mathbf{y}) \tag{7}$$

where $f_1 = \|.\|^2$ is the (squared) Euclidean norm in the standard linear or nonlinear PCA rule. With this choice of $f_1(\ )$, the cost function is minimised with respect to any set of samples from the data set on the assumption that the residuals are chosen independently and identically distributed from a standard Gaussian distribution. We may show that the minimisation of J is equivalent to minimising the negative log probability of the residual, $\mathbf{e}.$, if **e** is Gaussian.

Let $p(\mathbf{e}) = \frac{1}{Z}\exp(-\mathbf{e}^2)$ (8)

Then we can denote a general cost function associated with this network as

$$J = -\log p(\mathbf{e}) = (\mathbf{e})^2 + K \tag{9}$$

where *K* is a constant. Therefore performing gradient descent on *J* we have

$$\Delta W \propto -\frac{\partial J}{\partial W} = -\frac{\partial J}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial W} \approx \mathbf{y}(2\mathbf{e})^T \tag{10}$$

where we have discarded a less important term. See [16] for details.

In general [25], the minimisation of such a cost function may be thought to make the probability of the residuals greater dependent on the probability density function (pdf) of the residuals. Thus if the probability density function of the residuals is known, this knowledge could be used to determine the optimal cost function. [11] investigated this with the (one dimensional) function:

$$p(\mathbf{e}) = \frac{1}{2+\varepsilon}\exp\left(-|\mathbf{e}|_\varepsilon\right) \tag{11}$$

where

$$|e|_\varepsilon = \begin{cases} 0 & \forall|\mathbf{e}| < \varepsilon \\ |\mathbf{e}| - \varepsilon & \text{otherwise} \end{cases} \tag{12}$$

with ε being a small scalar ≥ 0.

[11] described this in terms of noise in the data set. However we feel that it is more appropriate to state that, with this model of the pdf of the residual, the optimal $f_1(\ )$ function is the ε-insensitive cost function:

$$f_1(\mathbf{e}) = |\mathbf{e}|_\varepsilon \tag{13}$$

In the case of the negative feedback network, the learning rule is

$$\Delta W \propto -\frac{\partial J}{\partial W} = -\frac{\partial f_1(\mathbf{e})}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial W} \qquad (14)$$

which gives:

$$\Delta W_{ij} = \begin{cases} 0 & \text{if} \left| e_j \right| < \varepsilon \\ \text{otherwise} & \eta y(sign(\mathbf{e})) \end{cases} \qquad (15)$$

The difference with the common Hebb learning rule is that the sign of the residual is used instead the value of the residual. Because this learning rule is insensitive to the magnitude of the input vectors ***x***, the rule is less sensitive to outliers than the usual rule based on mean squared error. This change from viewing the difference after feedback as simply a residual rather than an error permits us to consider a family of cost functions each member of which is optimal for a particular probability density function associated with the residual.

2.3 Applying Maximum Likelihood Hebbian Learning

The Maximum Likelihood Hebbian Learning algorithm is constructed now on the bases of the previously presented concepts as outlined here. Now the ε-insensitive learning rule is clearly only one of a possible family of learning rules which are suggested by the family of exponential distributions. This family was called an exponential family in [15] though statisticians use this term for a somewhat different family. Let the residual after feedback have probability density function

$$p(\mathbf{e}) = \frac{1}{Z} \exp(-|\,\mathbf{e}\,|^p)\,. \qquad (16)$$

Then we can denote a general cost function associated with this network as

$$J = E(-\log p(\mathbf{e})) = E(|\,\mathbf{e}\,|^p + K) \qquad (17)$$

where *K* is a constant independent of *W* and the expectation is taken over the input data set. Therefore performing gradient descent on *J* we have

$$\Delta W \propto -\frac{\partial J}{\partial W}\Big|_{W(t-1)} = -\frac{\partial J}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial W}\Big|_{W(t-1)} \approx E\{\mathbf{y}(p|\mathbf{e}|^{p-1}\, sign(\mathbf{e}))^T\,|_{W(t-1)}\} \qquad (18)$$

where T denotes the transpose of a vector and the operation of taking powers of the norm of ***e*** is on an elementwise basis as it is derived from a derivative of a scalar with respect to a vector.

Computing the mean of a function of a data set (or even the sample averages) can be tedious, and we also wish to cater for the situation in which samples keep arriving as we investigate the data set and so we derive an online learning algorithm. If the conditions of stochastic approximation [17] are satisfied, we may approximate this with a difference equation. The function to be approximated is clearly sufficiently smooth and the learning rate can be made to satisfy $\eta_k \geq 0, \sum_k \eta_k = \infty, \sum_k \eta_k^2 < \infty$ and so we have the rule:

$$\Delta W_{ij} = \eta . y_i . sign\left(e_j\right) | e_j |^{p-1} \tag{19}$$

We would expect that for leptokurtotic residuals (more kurtotic than a Gaussian distribution), values of p<2 would be appropriate, while for platykurtotic residuals (less kurtotic than a Gaussian), values of p>2 would be appropriate. Researchers from the community investigating Independent Component Analysis [14], [15] have shown that it is less important to get exactly the correct distribution when searching for a specific source than it is to get an approximately correct distribution i.e. all supergaussian signals can be retrieved using a generic leptokurtotic distribution and all subgaussian signals can be retrieved using a generic platykutotic distribution. Our experiments will tend to support this to some extent but we often find accuracy and speed of convergence are improved when we are accurate in our choice of p. Therefore the network operation is:

Feedforward: $$y_i = \sum_{j=1}^{N} W_{ij} x_j , \forall_i \tag{20}$$

Feedback: $$e_j = x_j - \sum_{i=1}^{M} W_{ij} y_i \tag{21}$$

Weight change: $$\Delta W_{ij} = \eta . y_i . sign\left(e_j\right) | e_j |^{p-1} \tag{22}$$

[11] described their rule as performing a type of PCA, but this is not strictly true since only the original (Oja) ordinary Hebbian rule actually performs PCA. It might be more appropriate to link this family of learning rules to Principal Factor Analysis since PFA makes an assumption about the noise in a data set and then removes the assumed noise from the covariance structure of the data before performing a PCA. We are doing something similar here in that we are basing our PCA-type rule on the assumed distribution of the residual. By maximising the likelihood of the residual with respect to the actual distribution, we are matching the learning rule to the probability density function of the residual.

More importantly, we may also link the method to the standard statistical method of Exploratory Projection Pursuit: now the nature and quantification of the interestingness is in terms of how likely the residuals are under a particular model of the probability density function of the residuals. In the results reported later, we also sphere the data before applying the learning method to the sphered data and show that with this method we may also find interesting structure in the data.

2.4 Sphering of the Data

Because a Gaussian distribution with mean a and variance *x* is no more or less interesting than a Gaussian distribution with mean *b* and variance *y* - indeed this second order structure can obscure higher order and more interesting structure - we remove such information from the data. This is known as “sphering". That is, the raw data is translated till its mean is zero, projected onto the principal component directions and multiplied by the inverse of the square root of its eigenvalue to give data which has mean zero and is of unit variance in all directions. So for input data *X* we find the covariance matrix.

Where *U* is the eigenvector matrix, *D* the diagonal matrix of eigenvalues, *T* denotes the transpose of the matrix and the angled brackets indicate the ensemble average. New samples, drawn from the distribution are transformed to the principal component axes to

$$\sum = \left\langle \left(X - \langle X \rangle\right)\left(X - \langle X \rangle\right)^T \right\rangle = UDU^T \tag{23}$$

give y where

$$y_i = \frac{1}{\sqrt{D_i}} \sum_{j=1}^{n} U_{ij}\left(X_i - \langle X_i \rangle\right), for 1 \le i \le m \tag{24}$$

Where *n* is the dimensionality of the input space and *m* is the dimensionality of the sphered data.

3 IBR for oceanographic real-time forecasting

A forecasting system capable of predicting the temperature of the water ahead of an ongoing vessel in real time has been developed using a IBR system [4], [6]. An IBR system was selected for its capacity of handling huge amounts of data, of adapting to the changes in the environment and to provide real time forecast.

In Figure 1, shadowed words represent the four steps of a typical IBR life cycle [1], the arrows represent data coming in or out of the instance-base (situated in the centre of the diagram) and the text boxes represent the result obtained by each of the four stages of the IBR life-cycle. Data are recorded in real time by sensors in the vessels and satellite pictures are received weekly. A *Knowledge Acquisition module* is in charge of collecting, handling and *indexing* the data in the instance-base. Once the real-time system is activated on an ongoing vessel, a *new instance* is generated every 2 km using the temperatures recorded by the vessel during the last 40km. This new instance is used to retrieve *m cases* from a collection of previous cases using Kernel methods [10]. The *m-*

retrieved instances are adapted by an Unsupervised Kernel method during the reuse phase to obtain an initial *(proposed) forecast* [10]. Though the revision process, the proposed solution is adjusted to generate the *final forecast* using the confidence limits from the knowledge base [5]. *Learning* (retaining) is achieved by updating the Kernels. A complete description of this system can be obtained in [10]. This IBR system has been successfully tested and it is presently operative in several oceanographic vessels. Improving this system has been our challenge and this section will outline the modifications that has been done to it with the intention of demonstrating that the Maximum Likelihood Hebbian Learning algorithm can provide successful results and automate the instances indexing and the instance retrieval process. The following tables shows the changes that have been done in the IBR system for real time oceanographic forecasting.

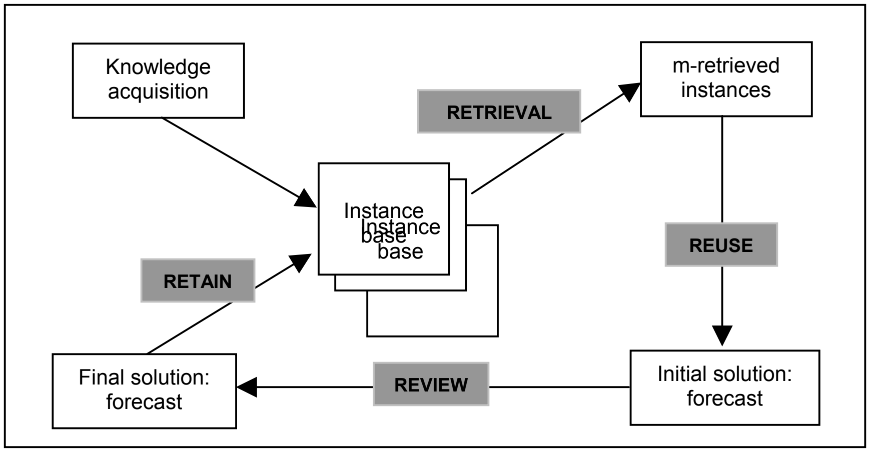


Fig. 1. IBR system architecture.

Table 1: Changes in the IBR system for real time oceanographic forecasting

STEP	Operating IBR system	Modifications and improvements
Indexing	Rule based system	Maximum Likelihood Hebbian Learning algorithm
Retrieval	Kernel methods	Maximum Likelihood Hebbian Learning algorithm
Reuse	Unsupervised Kernel methods	Unsupervised Kernel methods
Retrain	Kernel methods	Kernel methods / Maximum Likelihood Hebbian Learning algorithm

Table 1 outlines the changes made to the original system. The first column of the table indicates in which parts of the IBR system the changes have been made, the second column indicates the method originally used (and now eliminated) and column three indicates which methods have been included in the system. The changes indicated in table 1 have been introduced with the intention of developing a robust model, based on a technology easy to implement and that can automate the process of defining the retrieval step of the IBR system, facilitating the indexing of cases and helping in the learning and adaptation stage. The Maximum Likelihood Hebbian Learning algorithm automates these processes, clustering the instances and facilitating the retrieval of the most similar cases to a problem case. In this particular application the adaptation stage is carried out by an unsupervised kernel network, which structure need to be identified in advance, and tuned manually. We now present the structure of a case and indicate how the Maximum Likelihood Hebbian Learning algorithm has been used in the mentioned IBR parts.

3.1 The Instance

Each stored instance contains information relating to a specific situation and consists of an *input profile* (i.e. a vector of temperature values) together with the various fields shown in Table 2. A 40 km data profile has been found to give sufficient resolution to characterise the problem instance [6].

Table 2. Instance structure.

Instance Field	Explanation
Identification	unique identification: a positive integer in the range 0 to 64000
Input Profile, I	A 40 km temperature input vector of values Ij, (where j = 1, 2, … 40) Representing the structure of the water between the present position of the vessel and its position 40 km back.
Output Value, F	A temperature value representing the water temperature 5 km ahead of the present location
Time	Time when recorded (although redundant, this information helps to ensure fast retrieval)
Date	Date when the data were recorded (included for the same reasons as for the previous field).
Location	Geographical co-ordinates of the location where the value I_{40} (of the input profile) was recorded.
Orientation	Approximate direction of the data track, represented by an integer x, $(1 \leq x \leq 12)$.
Retrieval Time	Time when the instance was last retrieved.
Retrieval Date	Date when the instance was last retrieved.
Retrieval Location	Geographical co-ordinates of the location at which the instance was last retrieved.
Average Error	Average error over all forecasts for which the instance has been used during the adaptation step.

The parametric features of the different water masses that comprise the various oceans vary substantially, not only geographically, but also seasonally. Because of these variations it is therefore inappropriate to attempt to maintain an instance base representing patterns of ocean characteristics on a global scale; such patterns, to a large extent, are dependent on the particular water mass in which the vessel may currently be located. Furthermore, there is no necessity to refer to instances representative of all the possible orientations that a vessel can take in a given water mass. Vessels normally proceed in a given predefined direction. So, only instances corresponding to the current orientation of the vessel are normally required at any one time.

3.2 Indexing and Retrieving Instances the Maximum Likelihood Hebbian Learning algorithm

To explore the structure of a data set we are using Maximum Likelihood Hebbian Learning. Applying equations 20 to 22 to the Case-base, the MLHL algorithm groups the cases in clusters automatically. The proposed indexing mechanism classifies the cases/instances automatically, clustering together those of similar structure. This technique attempts to find interesting low dimensional projections of the data so that humans can investigate the structure of the data even by eye. One of the great advantages of this technique is that it is an unsupervised method so we do not need to have any information about of the data before hand. When a new case is presented to the IBR system, it is identified as belonging to a particular type by applying also equations 20 to 22 to it. This mechanism may be used as an universal retrieval and indexing mechanism to be applied to any problem similar to the presented here.

3.3 Forecasting with the Instance-base Reasoning System

Several experiments have been carried out to illustrate the effectiveness of the IBR system, which incorporates the MLHL algorithm. Experiments have been carried out using data from the Atlantic Meridian Transept (AMT) Cruise 4 [6]. We show in Figure 2 the errors on a data set of 500 instances randomly taken from the AMT 2000 data set (composed of more than 150.000 instances) using the Kernel based CBR system. Figure 3 shows the results obtained with the new MLHL proposed modification. The mean absolute error, when forecasting the temperature of the water 5 Km ahead of an ongoing vessel, along 10.000 km (form the UK to the Falkland Island) was 0.0205 ºC [10]. With the new proposal, the average error have been reduced to 0.0167 ºC, for the whole data set, which compares very favourably with the initial Instance based reasoning system and other previous methods [6]. Using the MLHL algorithm the number of predictions with an error higher than 0,1 has been reduced in more than 30%. The reason may be that the data selection carried out with the MLHL algorithm facilitate the creation of more consistent models during the adaptation stage than with the Unsupervised Kernel algorithm. We have compared the proposal presented in this paper with several classification algorithms that may be used for the indexing and retrieval of cases: Principal Components Analysis (PCA), Kernel Methods and Growing Cells Structures (GCS) [20]. The Maximum Likelihood Hebbian Learning method outperforms the others techniques improving the final results clustering the instances adequately for a future adaptation using and Unsupervised Kernel algorithm. The average forecasting error obtained with the PCA method was of 0.0310 ºC, with the Kernel Method was of 0.0205 ºC and with the GCS was of 0.0231ºC.

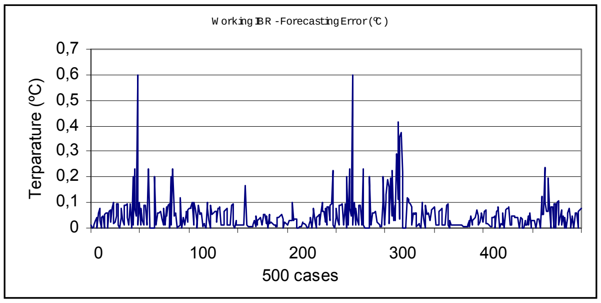


Fig. 2. Average error of the working IBR system in 500 forecasts carried out during the AMT 2000 cruise from the UK to the Falkland Island.

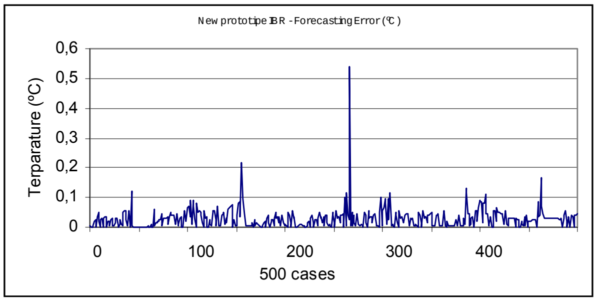
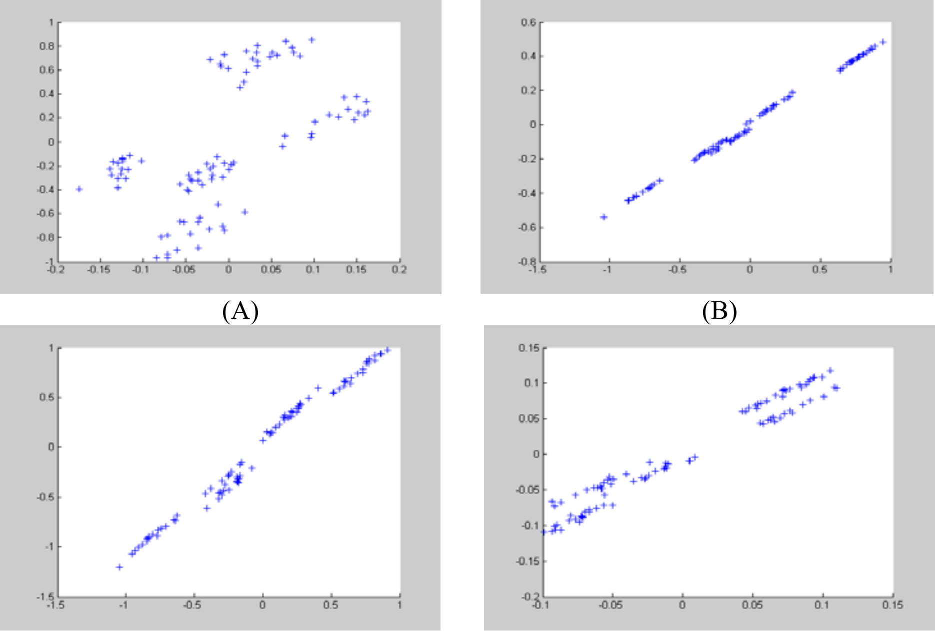


Fig. 3. Average error with IBR system using MLHL algorithms in the same 500 predictions as the ones showed in Figure 2.

For pedagogical purposes, we illustrate the method on a small sample of cases. 150 instances which characterise the oceanographic problem has been selected from five different areas of the Atlantic ocean (five water masses with different properties). This has been done because it is useless to represent the 150000 instances in one Figure. Figure 4 shows the classification ability of the MLHM proposed, which gives a rather better separation of the individual five oceanographic areas found in the data set. We are using an unsupervised learning technique in the field of artificial neural networks so generally we do not need any information about the data. Of course, the data must have some kind of structure (correlation, redundancy, etc). The only condition that we need is to set the dimension of the output vector (M), i.e. the number of output neurons, to be greater than the number of patterns or factors (number of water masses, in this case), otherwise some outputs will have found two or more patters simultaneously. We have used 10 output neurons in this experiment.



(A) (B)

(C) (D)

Fig. 4. Figure A shows the results of the Maximum Likelihood Hebbian learning network using *p*=0. The Maximum Likelihood Method identifies a projection, which spread the data out identifying the five different clusters. Figure B presents the result obtained with the PCA method, Figure C shows the clusters obtained with the Kernel Method and Figure D presents the results obtained with the GCS methods.

4. Conclusion

We have demonstrated a new technique for case/instance indexing and retrieval, which could be used to construct instance based reasoning systems. The basis of the method is a Maximum Likelihood Hebbian Learning algorithm. This method provides us with a very robust model for indexing the data and retrieving instances without any need of information about the structure of the data set in advance. It has been shown to give extremely accurate results on an exemplar-forecasting task: our results of 0.0167 ºC error are among the best we have ever achieved on this data set. This is very important in the identification of fronts in these large bodies of water particularly since such fronts have an extremely adverse effect on underwater communications. The retrieval of the best matching instance is a very simple operation using the proposed method and presents no major computational obstacles. The whole system may be used with any number-based data set; an area of ongoing research is the application of algorithms that combine MLHL and kernel methods. We believe that this method may be improved with Competence-coverage techniques [24], and several experiments are in progress. This method is also more stable and accurate than previous implementation carried out with Kernel Methods, K-nearest neighbour algorithms, and growing cells structures.

References

1. Aamodt A. and Plaza E. (1994) Case-Based Reasoning: foundational Issues, Methodological Variations, and System Approaches. AICOM. Vol. 7. No 1. March 1994.
2. Corchado E. and Fyfe C. (2002) Maximum and Minimum Likelihood Hebbian Rules as a Exploratory Method. 9th International Conference on Neural Information Processing. 18 - 22 November 2002. Singapore.
3. Corchado E., MacDonald D. and Fyfe C. (2002) Optimal Projections of High Dimensional Data. ICDM '02. The 2002 IEEE International Conference on Data Mining, IEEE Computer Society; Maebashi TERRSA, Maebashi City, Japan December 9 - 12, 2002
4. Corchado J. M. and Aiken J. (2003) Hybrid Artificial Intelligence Methods in Oceanographic Forecasting Models. IEEE SMC Transactions Part C. Vol.32, No.4, November 2002: 307-313.
5. Corchado J. M. and Lees B. (2000) Adaptation of cases for case-based forecasting with neural network support., S.K Pal, T.S. Dillon and D.S. Yeung (Eds.), Soft Computing in Case Based Reasoning", Springer Verlag, London.
6. Corchado J. M., Aiken J. and Rees N. (2001) Artificial Intelligence Models for Oceanographic Forecasting. Plymouth Marine Laboratory, U.K, 2001. ISBN: 0-9519618-4-5.
7. Diaconis P. and Freedman D. (1984) Asymptotics of Graphical Projections. The Annals of Statistics. 12(3): 793-815.
8. Fdez-Riverola F. and Corchado J. M. (2003) FSfRT: Forecasting System for Red Tides. Applied Intelligence. Soft Computing in Case-Based Reasoning. In press. ISSN: 0924-669X.
9. Fyfe C. and Baddeley R. (1995) Non-linear data structure extraction using simple Hebbian networks, Biological Cybernetics 72(6), p533-541.
10. Fyfe C. and Corchado J. M. (2001) Automating the construction of CBR Systems using Kernel Methods. International Journal of Intelligent Systems. Vol 16, No. 4, April 2001. ISSN 0884-8173.
11. Fyfe C. and MacDonald D. (2001) ε-Insensitive Hebbian learning, Neuro Computing.
12. Fyfe C. and Corchado E. (2002a). Maximum Likelihood Hebbian Rules. 10th European Symposium on Artificial Neural Networks, ESANN"2002, Bruges, April 24-25-26, 2002.
13. Fyfe C. and Corchado E., (2002b) A New Neural Implementation of Exploratory Projection Pursuit. IDEAL2002 Third International Conference on Intelligent Data Engineering and Automated Learning. Manchester, UK . 12-14 August, 2002.
14. Hyvärinen A. (2001) Complexity Pursuit: Separating interesting components from time series. Neural Computation, 13: 883-898.
15. Hyvärinen A. Karhunen J. and Oja E. (2002) Independent Component Analysis, Wiley, ISBN 0-471-40540-X.

16. Karhunen J. and Joutsensalo J. (1994.) Representation and Separation of Signals Using Non-linear PCA Type Learning, Neural Networks, 7:113-127.
17. Kashyap R. L., Blaydon C. C., and Fu K. S. (1994.) Stochastic Approximation. in A Prelude to Neural Networks: Adaptive and Learning Systems, Ed Jerry M. Mendel, Prentice Hall, ISBN 0-13-147448-0.
18. Lai P. L., Charles D., and Fyfe C., (2000.) Seeking Independence using Biologically Inspired Artificial Neural Networks, in Developments in Artificial Neural Network Theory : Independent Component Analysis and Blind Source Separation, Editor M. A. Girolami, Springer Verlag.
19. Lees B. and Corchado J. M. (1999) Integrated case-based approach to problem solving. In: Lecture Notes in Artificial Intelligence 1570, XPS-99: Knowledge-Based Systems - Survey and Future Directions, edited by Frank Puppe, Springer, Berlin, pp. 157-166.
20. MacDonald D. and Fyfe C. (2000) The Kernel self-organising map. In R.J Howlett and L.C. Jain, editors, Fourth International Conference on Knowledge-based Intelligent Engineering Sustems and Allied Technologies, KES 20000.
21. Oja E. (1989) Neural Networks, Principal Components and Subspaces, International Journal of Neural Systems, 1:61-68..
22. Oja E., Ogawa H. and Wangviwattana J. (1992)Principal Components Analysis by Homogeneous Neural Networks, part 1, The Weighted Subspace Criterion, IEICE Transaction on Information and Systems, E75D: 366-375. 1992.
23. Pal S.K. , Dillon T.S. and Yeung D.S. (2000) Soft Computing in Case Based Reasoning. Springer Verlag, 2000, London, U.K. ISBN 1-85233262-X.
24. Smyth, B. and McKenna, E. (2001). Competence Models and the Maintenance Problem. Special Issue on Maintaining Case-Based Reasoning Systems, Journal of Computational Intelligence: 17(2) 235-249
25. Smola A.J. and Scholkopf B.(1998) A Tutorial on Support Vector Regression. Technical Report NC2-TR-1998-030, NeuroCOLT2 Technical Report Series.
26. Watson I. (1997) Applying Case-Based Reasoning: Techniques for Enterprise Systems. Morgan Kaufmann.
27. Wilke W. and Bergmann R. (1996). Incremental Adaptation with the INRECA- System. ECAI 1996 Workshop on Adaptation in Case-Based Reasoning.
28. Xu L. (1993) Least Mean Square Error Reconstruction for Self-Organizing Nets", Neural Networks, Vol. 6, pp. 627-648.