Artificial Intelligence in Knowledge Management for Time Series Forecasting

José Neves   Paulo Cortez
jneves@di-ia.uminho.pt   paulo@di-ia.uminho.pt
Departamento de Informática
Universidade do Minho
Largo do Paço, 4709 Braga Codex, PORTUGAL
Voice: +351(53)604466/70   Fax: +351(53)604471

Abstract

Knowledge Management (KM) is a keen topic for an organization, in particular to those that have to deal with knowledge acquired from different sources, either from its own experiences or from that of others, to decide on the effective use of that knowledge to fulfill the goals of the organization. As representative examples of *KM*, one may have the object-oriented data bases, hypermedia or concept maps. On the other hand, techniques developed in *Artificial Intelligence* for knowledge representation and discovery may be of great use in *KM*; in particular, it seems natural to explore the potential of the organization past data to deal with management decisions of the present.

One way is to use *Time Series Forecasting (TSF)*, where forecasts are based on pattern recognition of past observations ordered in time. Traditional *TSF* methods, such as the Holt-Winters and the Box-Jenkins ones, are based on particular characteristics of the *Time Series (TS)*, such as *trend* or *seasonal* effects. These methods work with well behaved *TS*, but present some drawbacks on *TS* with noise or some unknown nonlinear relations among the *TS* data.

An alternative approach is the use of *Artificial Neural Networks (ANNs)*, which present two main advantages: *ANNs* can extrapolate patterns from past data, even in *TS* with noise, and may adapt their behavior as new data comes in.

A problem with the use of this approach is the search time for the best *ANN* architecture, which involves a large searching space, demanding a huge computational effort. Other aspect of concern is that of *TS* data filtering. Not all lags of the *TS* have the same influence over the forecast. Feeding the *ANN* with a big time window will slow the *ANN* forecasting efficiency. To solve these pitfalls, one may use random search, hill climbing or genetic procedures. The last ones are known to work well on problems of combinatorial nature, obtaining good solutions where other methods seem to fail.

This paper presents an integrated approach for *TSF*: a set of rules will create the training cases, based on some lags of the *TS*; these rules and the *ANN* parameters will be encoded on the genetic chromosomes; finally, each *ANN* will be trained, leading to competition.

Keywords: Neural Networks, Genetic Algorithms, Data Filtering, Time Series.

1 Introduction

In recent years a strong emphasis has been placed on improving decision making in organizations. Indeed, in the old days, the management could run their businesses based on their own feelings and intuitions; however this kind of approach is disappearing, replaced by new management decision-making disciplines, such as operations research, statistics, principles of organizational design and computers [10].

Knowledge Management (KM) is a keen topic for any organization, in particular to those that have to deal with knowledge acquired from different sources, either from its own experiences or from that of others, to decide on the effective use of that knowledge to fulfill their goals [4][3]. As representative examples of *KM* one has the object-oriented data bases, hypermedia or concept maps. On the other hand, techniques developed in *Artificial Intelligence* for knowledge representation and discovery may be of great use in *KM* [14]. A key aspect of any resolution reached or given is the ability to predict the circumstances that surround it. Thus, it seems natural to explore the potential of the organization past data, to deal with management decisions of the present [10]. One way, that has gained ground, is to use *Time Series*

Forecasting (TSF), where forecasts are based on pattern recognition of observations ordered in time. Short term predictions, one or two predictions ahead, are used for current management decisions (eg. dealing with stocks); middle/long term forecasts are used for strategic decisions (eg. elaborating budgets). Traditional *TSF* methods, such as the Holt-Winters and the Box-Jenkins ones, are based on particular characteristics of the *Time Series (TS)*, such as *trend* or *seasonal* effects [10][2]. These methods return accurate forecasts on well behaved *TS*, but loose accuracy when the *TS* present noise or non linear relations among the *TS* data.

An alternative approach is the use of *Artificial Neural Networks (ANNs)*, which present some advantages: *ANNs* can extrapolate patterns from past data, even in *TS* with noise, and may adapt their behavior as new data comes in [13][11][6]. Comparative studies [4][11][3][5] suggested that *ANNs* can perform as well or even better that conventional methods; but the problem with the use of such an approach is the search time for the best *ANN* architecture. This involves a large searching space, demanding a huge computational effort. Another relevant point is concerned with *TS* data filtering. Not all lags of the *TS* have the same influence over the forecast. Feeding the *ANN* with a big time window can affect the *ANN* forecasting efficiency by increasing entropy, where entropy is to be understood as a statistical measure of the disorder of the system under study (a closed one), in terms of the amount of information that is output, expressed by $S = K \log(P) + C$, where P is the probability that a particular state of the system exists, K is the Boltzman constant and C is another constant; thus selecting the correct time lags to feed the *ANN* may improve the forecasts, specially for seasonal *TS* [5].

To solve these pitfalls, one may use random search, hill climbing or genetic procedures [14][9]. The last ones are known to work well on problems of combinatorial nature, obtaining good solutions where other methods seem to fail [7].

In this paper it is presented an integrated approach for *TSF* on seasonal *TS*: a set of rules will create the training cases, based on some lags of the *TS*; these rules and the *ANN* parameters will be encoded on the genetic chromosomes; finally, each *ANN* will be trained, leading to competition.

2 The Artificial Neural Network Architecture

One of the difficulties that arise when using *ANNs* is the selection of the best *ANN* architecture, the training algorithm and its parameters. This process depends on the characteristics of the problem, the data available, on empirical experiments and intuition [6][12]. There are different kinds of *ANNs* architectures being the most widely used and well known the feed-forward ones. Most of the research on the use of *ANNs* for *TSF* has focused on this architecture [4][11][3][5]. Based on these studies and in order to cut some of the searching space, it was decided to use fully connected feed-forward *ANNs* with bias, without shortcut connections and with one hidden layer. The *Resilient Backpropagation (RPROP)* algorithm [13] was used to perform the training. This is a fast backpropagation algorithm, defined as the extension of a two argument's function, taking values from a closed and well defined domain. The initial weights were randomly set within the range $[\frac{-2}{z}; \frac{2}{z}]$ for a node with z inputs [6]. Eight activation functions were used (Table 1)[1]. The *ANN* topology was represented in terms of productions of the form $L_i - L_h - L_o$, where L_i stands for the number of nodes of the input layer, L_h for the number of nodes of the hidden layer and L_o for the number of output nodes.

Table 1: Activation Functions

Name	Function $f(x)$	Codomain
linear	x	$]-\infty, +\infty[$
sigmoid	$\frac{1}{1+\exp(-x)}$	$[0, 1]$
sigmoid1	$\frac{2}{1+\exp(-x)} - 1$	$[-1, 1]$
sigmoid2	$\frac{x}{1+\|x\|}$	$[-1, 1]$
tanh	$\tanh(x)$	$[-1, 1]$
cos	$\sin(x \bmod 2\pi)$	$[-1, 1]$
sin	$\cos(x \bmod 2\pi)$	$[-1, 1]$
gaussian	$\exp(\frac{-x^2}{2})$	$[-1, 1]$

3 The Training Data

The normal process to do *TSF* with *ANNs* is to use a moving time-window of n lags, for an *ANN* of n inputs [15]. But not all time lags of the *TS* have the same influence on the forecast, specially for seasonal *TS*. For example the Arima model [2] suggests the $< 1, 12, 13 >$ or $< 1, 2, 12, 13 >$ lags for monthly *TS*, which are very common on organizations and will be the type of *TS* studied in this paper. Thus, twelve sets of lags will be tested (Table 2) by the genetic algorithm, working as a data filtering process. These sets were based on ones own experiments and on the results presented in [5]. Note that each set determines the L_i. The training cases are built according to each set of lags. The goal is to have one forecast at the output of the *ANN* when some previous lags of the *TS* feed the *ANNs* input nodes. Thus, L_o is defined always in terms of one node. As an example, for set 2 (Figure 1) the system will produce the following training data:

$$\begin{array}{lcl} x_1, x_{11}, x_{12} & \rightarrow & x_{13} \\ x_2, x_{12}, x_{13} & \rightarrow & x_{14} \\ \dots & \rightarrow & \dots \\ x_{k-12}, xk-2, x_{k-1} & \rightarrow & x_k \end{array}$$

Table 2: Sets of lags

Number	Lags	L_i
1	1,12	2
2	1,2,12	3
3	1,12,13	3
4	1,2,3,12	4
5	1,2,12,13	4
6	1,2,3,4,12	5
7	1,2,12,13,14	5
8	1,2,3,4,12,13	6
9	1,2,3,4,12,13,14	7
10	1,2,3,4,5,6,7,8,9,10,11,12	12
11	1,2,3,4,5,6,7,8,9,10,11,12,13	13
12	1,2,3,4,5,6,7,8,9,10,11,12,13,14	14

where $x_1, x_2, ..., x_k$ stands for the *TS*.

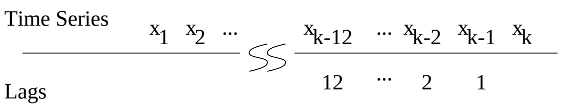

Figure 1: Time Series Lags

Some of the activation functions (Table 1) require that the data has to be in a certain range, so that all data was normalized to the range $[0.2, 0.8]$. This range is suggested in [15] and allows some "space" for *TS* with *trend* (general tendency or direction). To avoid problems of overfitting (loose of the *ANN* generalization capacity) early stopping was implemented [12], being the training data divided into two categories: a training set (to assimilate the patterns) and a validation set (to test the *ANN* generalization accuracy). The system uses a validation set built on 10% of the available training examples.

One-step ahead forecasts are easily done by feeding the trained *ANN* with the present time lags for the *TS*. Multi-step ahead forecasts are done using feedback data from the forecasts, namely:

$$\begin{aligned}
f_{1,k} &= output(x_{k-11}, x_{k-1}, x_k) \\
f_{2,k} &= output(x_{k-10}, x_k, f_{1,k}) \\
&\dots \\
f_{n,k} &= output(f_{n-12,k}, f_{n-2,k}, f_{n-1,k})
\end{aligned}$$

where $output$ stands for the output function of the *ANN* and $f_{i,j}$ for the forecast in the j period to i periods ahead of j.

4 The Genetic Algorithm

A *Genetic Algorithm (GA)* procedure will be used as an optimization tool for the selection of the best set of lags and the ANNs parameters. After experimenting, one concluded that there were five factors that affected the forecasting: the set of lags used (lag), the RPROP algorithm parameters (Δ_0 and Δ_{max}), the activation function (f), the number of hidden nodes (L_h), and the random weights initialization seed (s). This can be easily explained: the first factor (lag) sets which data to feed to the *ANN* while the remaining ones set how the *ANN* learns.

The *ANN's* parameters were encoded using base 2 gray codes, being the most influent ones at the left (Figure 2). To reduce some of the searching space L_h was set to the range $[3, 14]$ [3]. Thus, $lags$ and L_h were encoded with 4 bits. Δ_{max} was set to 50 (value advised in [13]), and Δ_0 was encoded in only 3 bits using discrete values in the range $[0.1, 0.8]$. The 8 activation functions where also encoded into 3 bits. Finally, the random initialization seed was not encoded since the influence of the seed in the forecast is random by definition. For example, the encoded string 00111101111101 states that one is using $1, 12, 13$ ($lags = 3$) set of lags on a $3 - 12 - 1$ *ANN* to be trained with the *tanh* activation function, being $\Delta_0 = 0.6$.

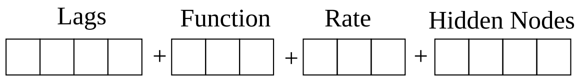

Figure 2: String Encoding

As with ANNs, when using *GAs* one has to set some parameters and operators according to the peculiarity of the problem. However, since in this case the *GA* works as a high order optimization process, the choice of these parameters and operators may not be so crucial. After some empirical experiences it was decided to use a population of 30 individuals, rank-based selection [9], one point crossover with a crossover rate of 1 and a mutation rate of 0.02. As the fitness function the system uses the *Mean Squared Error (MSE)* calculated for the validation cases as:

$$fitness = \frac{1}{MSE}$$

Figure 3 shows how the process works. At the beginning a set of of individuals (or chromosomes) are randomly generated. Next, the chromosomes are evaluated, which means that the training cases and the *ANN* are created according to the genome information. The fitness value is determined after training the *ANN* with the RPROP algorithm. After evaluation, all individuals are ranked. *Crossover* and *mutation* operations will create a new population of individuals, which will be also evaluated. Finally the *rank-based selection* operation will select the best individuals of both populations, leading to a new generation. This process goes on until some stopping criterion is fulfilled (in one's case after g generations).

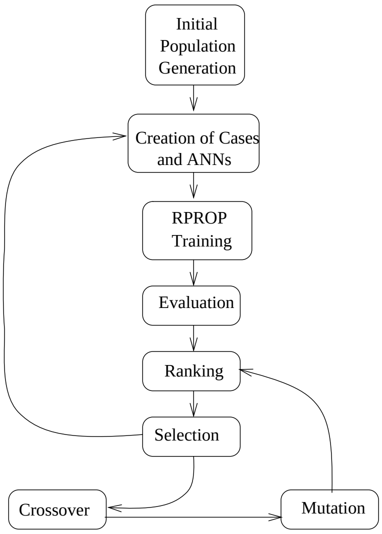

Figure 3: The structure of the GANN system

Table 3: The best *ANN* for each series

Series	$lags$	Topology	Function	Δ_0
1	$1, 12, 13$	$3-8-1$	*linear*	0.7
2	$1, 2, 3, 12$	$4-3-1$	*gaussian*	0.5
3	$1, 2, 12, 13, 14$	$5-5-1$	*tanh*	0.1
4	$1, 2, 3, 4, 12, 13$	$6-6-1$	*linear*	0.2

5 Forecasting Results

In this paper were used monthly series of real data from different types of sources (airline passengers, industrial sales of paper, restaurant sales and loads of pollution equipment) [2][10][8]. The system's results were compared with those obtained using two well known conventional methods: Holt-Winters [10] and Arima [2]. Table 3 shows the *ANNs* modeled by the system for each series, while Tables 4 and 5 compare the system's results with the ones obtained by conventional methods for short and long term forecasting.

The Holt-Winters method seems to work very well on short term forecasts. In fact this method outperforms the other methods for seasonal series. One's system managed to outperform the ARIMA model on the first series. This is not surprising since the Holt-Winter method was specially developed for this kind of *TS*. The situation changes somehow on long term forecasting, where the system's results outperforms both methods on series 1 and outperformes the Holt-Winters one on series 4.

Table 4: Comparing the system's one-ahead forecasts for 12 periods with other methods (using the MSE as the metric for the forecasts)

Series	System	Arima	Holt-Winters
1	368.3	452	271
2	4500	2581	1885
3	32090	15290	11435
4	759581	-	530654

Table 5: Comparing the system's long term forecasts ($f_{i,k}$, i=1,...,12) with other methods (using the MSE as the metric for the forecasts)

Series	System	Arima	Holt-Winters
1	308	521	621
2	4792	2707	3046
3	31279	20289	16954
4	657445	-	2927255

6 Conclusions

The results obtained so far suggest that the *ANNs* can be used as an alternative method for seasonal *TSF* , specially on long term forecasting. The system presented has the disadvantage of being much more demanding on computational power than the ones potentiate by the Holt-Winters and the Arima methods. However, Arima requires the use of an expert analyst, while the proposed system works automatically with a minimum of human intervention. The data filtering process revealed to improve forecasts but, the bad results on short term forecasting indicate that further research is necessary, namely: the use of more elaborated data filtering techniques, the use of different kinds of *ANNs* (eg. with shortcut connections) and the use of a better early stopping criteria (since a part of the available data is not used on the training).

References

[1] E. Azoff. *Neural Network Time Series Forecasting of Financial Markets*. John Wiley & Sons, 1994.

[2] G. Box and G. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden Day, San Francisco, USA, 1976.

[3] P. Cortez, J. Machado, and J. Neves. An Evolutionary Artificial Neural Network Time Series Forecasting System. In *IASTED International Conference on Artificial Intelligence, Expert Systems and Neural Networks*, Honolulu, Havai, August 1996.

[4] P. Cortez, M. Rocha, J. Machado, and J. Neves. A Neural Network Based Forecasting System. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, Perth, Western Australia, November 1995.

[5] J. Faraday and C. Chatfield. Times Series Forecasting with Neural Networks: A Case Study. Research

Report, Statistics Group, University of Bath, UK, 1995.

[6] S. Gallant. *Neural Network Learning and Expert Systems*. MIT Press, Cambridge, USA, 1993.

[7] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Inc., NY, USA, 1989.

[8] J. Hanke and A. Reitsch. *A Business Forecasting*. Allyn and Bancon Publishing Company Inc., Massachussetts, USA, 1989.

[9] P. Koenh. Combining Genetic Algorithms and Neural Networks. Thesis for Master Science Degree, University of Tennessee, Knoxville, USA., 1994.

[10] S. Makridakis, S. Weelwright, and V. McGee. *Forecasting: Methods and Applications*. John Wiley & Sons, New York, USA, 1978.

[11] G. Papadourakis, G. Spanoudakis, and A. Gotsias. Application of Neural Networks in Short-Term Stock Price Forecasting. In *First International Workshop on Neural Networks in the Capital Markets*, London, UK, November 1993.

[12] L. Prechelt. PROBEN1 - A Set of Neural Network Benchmark Problems and Benchmarking Rules. Research Report, Fakultất fűr Informatik, Universitất Karlsruhe Germany, 1994.

[13] M. Riedmiller and H. Braun. A Direct Adaptative Method for Faster Backpropagation Learning: The RPROP Algorithm. In *Proceeedings of the IEEE International Conference on Neural Networks*, San Francisco, CA, USA, 1993.

[14] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, New Jersey, USA, 1995.

[15] Tang Z. and Fishwick F. Feed-forward Neural Nets as Models for Time Series Forecasting. Technical Report, University of Florida, 1991.