Proceedings ICNN'95, 27 Nov – 1 Dec 1995, Perth, WA, Vol 2, pp. 1028-1031, IEEE 1995

Short Term Prediction of Sales in Supermarkets

Frank M. Thiesing, Ulrich Middelberg, Oliver Vornberger
Department of Mathematics/Computer Science
University of Osnabrück
D-49069 Osnabrück, Germany
frank@informatik.uni-osnabrueck.de

ABSTRACT

In this paper artificial neural networks are applied to a short term forecast of the sale of articles in supermarkets. The times series of sales, prices and advertising campaigns are modelled to fit into feedforward multilayer perceptron networks that are trained by the back-propagation algorithm. Several net topologies and training parameters have been compaired. For enhancement the back-propagation algorithm has been parallelized in different manners. One batch and two on-line training algorithms are implemented on parallel systems with both the runtime environments PARIX and PVM. The research will lead to a practical forecasting system for supermarkets.

1. Introduction

Time series prediction for economic processes is a topic of increasing interest. In recent years artificial neural networks have been applied to this problem successfully [4], especially in the financial field. Neural networks can be used easier for the prediction of chaotic and noisy time series than statistical methods because they are able to learn the system dependencies on their own.

In order to reduce stock-keeping costs, a proper forecast of the demand in the future is necessary. In this paper we use feedforward multilayer perceptron networks for a short term forecast of the sale of articles in supermarkets. The nets are trained on the known sales volume of the past for a certain group of related products. Additional information like changing prices and advertising campaigns are also given to the net to improve the prediction quality. The net is trained by the back-propagation algorithm [2] on a window of inputs describing a fixed set of recent past states. For enhancement the algorithm is implemented on parallel systems.

2. Sale forecast by neural networks

In our project we use the sale information of 53 articles of a certain product group in a supermarket. The information about the number of sold articles and the sales revenues in DM (German currency unit) are given weekly starting September 1994. In addition there are advertising campaigns for articles often combined with temporary price reductions. Such a campaign lasts about two weeks and has a significant influence on the demand on this article. Sale, average price and advertising campaigns for a specific article are shown in figure 2.

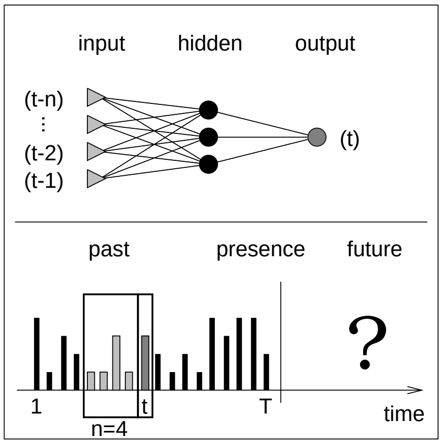


Fig. 1: feedforward MLP for time series prediction

We use feedforward multilayer perceptron (MLP) networks with one hidden layer together with the back-propagation training method [3]. In order to predict the future sale the past information of n recent weeks is given in the input layer. The only result in the output layer is the sale for the next week. So there is a window of n weeks in the past and one in the future (see figure 1).

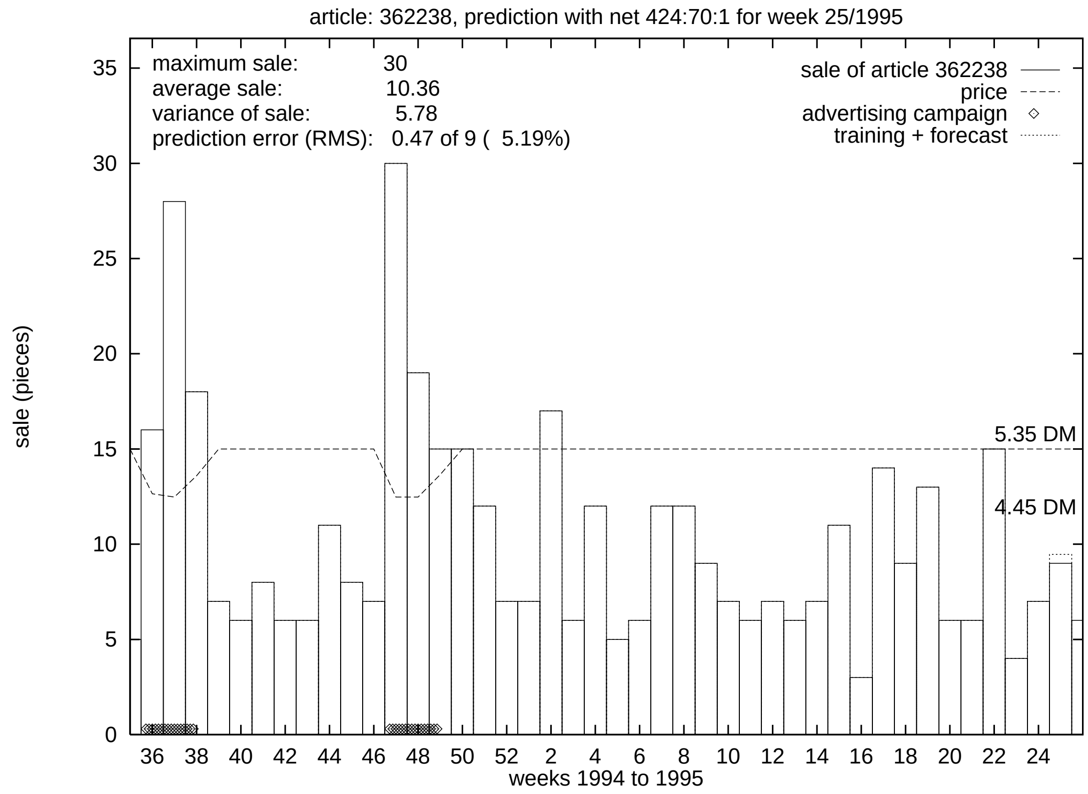


Fig. 2: sale and prediction for an article with advertising

3. Preprocessing the input data

An efficient preprocessing of the data is necessary to input it into the net. Due to our implementation of the back-propagation algorithm all information must be scaled to $[0, 1]$. We assume that the necessary information is given for T weeks in the past. With the following definitions

$$\begin{aligned} ADV_i^t &:= \text{number of advertising days} \\ & \quad\ \text{for article } i \text{ within week } t \\ SAL_i^t &:= \text{sale of article } i \text{ within week } t \\ MAXSAL_i &:= \max_{1 \le t \le T} \left\{ SAL_i^t \right\} \end{aligned}$$

we have decided to use the following inputs for each article i and week t:

$$adv_i^t := \frac{ADV_i^t}{6} \quad \text{(normalized number of advertising days within week } t \text{)}$$

$$pri_i^t := \left\{ \begin{array}{lcl} 1.0 & : & \text{price increases} \\ 0.5 & : & \text{price keeps equal} \\ 0.0 & : & \text{price decreases} \end{array} \right\} \begin{array}{l} \text{within} \\ \text{week } t \end{array}$$

$$sal_i^t := \frac{SAL_i^t}{MAXSAL_i} \cdot 0.8$$

For each article i and recent week t we use the three-dimensional vector:

$$vec_i^t := \left(adv_i^t, pri_i^t, sal_i^t \right)$$

For a week t in the future the vector is reduced by the unknown sale:

$$\widehat{vec}_i^t := \left(adv_i^t, pri_i^t \right)$$

To predict the sale for one article within a week t, we use a window of the last n weeks. So we have the following input vector for each article i:

$$input_i^t := \left(vec_i^{t-n}, vec_i^{t-n+1}, ..., vec_i^{t-1}, \widehat{vec}_i^t \right)$$

We have quite a constant sales volume of all products, because all the considered articles belong to one product group. An increasing sale of one article in general leads to a decrease of other sales. Due to this, we concatenate the *input* vectors of all p articles to get the vector given to the input layer:

$$INPUT^t := \left(input_1^t, input_2^t, ..., input_p^t \right)$$

The sale of article i within week t (sal_i^t) is the requested nominal value in the output layer that has to be learned by one net for this $INPUT^t$ vector.

So we have p nets where the i-th net adapts the sale behaviour of article i. Therefore we have a training set with the following pairs (see figure 1):

$$(INPUT^t, sal_i^t) \text{ with } n \leq t \leq T$$

To forecast the unknown sale sal_i^{T+1} for any article i within a future week $T+1$ we give the following input vector to the trained i-th net:

$$INPUT^{T+1}$$

The output value of this net is expected to be the value sal_i^{T+1}, which has to be re-scaled to the value for the sale of article i within week $T+1$:

$$SAL_i^{T+1} = \frac{sal_i^{T+1} \cdot MAXSAL_i}{0.8}$$

4. Empirical results

To determine the appropriate configuration of the feedforward MLP network several parameters have been varied:

1. time window: $n = 2$ resp. $n = 3$
2. the number of hidden neurons: $\frac{1}{12}$ resp. $\frac{1}{6}$ of the number of input neurons
3. training rate and momentum

We are using the information of 53 articles in the input layer. The topology of the net is described by the syntax: `(input neurons:hidden neurons:output neurons)`.

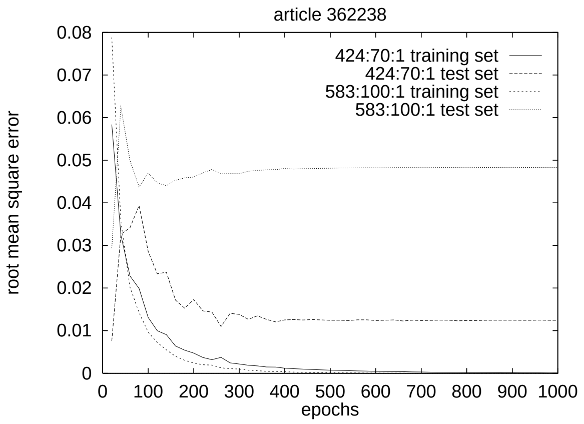


Fig. 3: error during training

The given data is split into a training set (week 36/1994 to week 24/1995) and a test set (week 25/1995). The test set is not trained and only considered to check whether the net has generalized the behaviour of the time series. With $n = 2$ we have 39 pairs in the training set and one in the test set, with $n = 3$ we have 38 pairs in the training set and one in the test set.

Several experiments have led to a training rate of 0.25 and a momentum of zero that are best for training and prediction.

Figure 3 shows the root mean square error on the training and test set for $n = 2$ resp. $n = 3$, while learning 1000 epochs of the time series for the article in figure 2 with this parameter settings. The error is going down immediately on the training set, especially for the larger nets.

More important is the error on the test set — the prediction error. This is better for the net with $n = 2$. It needs more epochs to learn the rule of the time series, but can generalize its behaviour better.

The prediction error of the net 424:70:1 in means of sales can be seen from figures 2, too. For the week 25/1995 the forecasted sale is drawn dotted: the error is smaller than one piece.

The time for training the nets on a sequential SUN SPARC 20 can be seen in table 1.

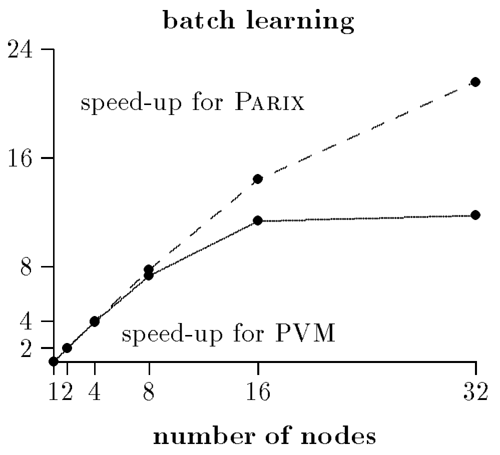
Table 1: training times on SPARC 20-50MHz

net topology	n	# training pairs	time for 1000 epochs
424:35:1	2	39	489 sec
424:70:1	2	39	1018 sec
583:50:1	3	38	907 sec
583:100:1	3	38	1815 sec

5. Parallelization

For enhancement the back-propagation algorithm has been parallelized in different manners: First the training set can be partitioned for the batch learning implementation. The neural network is duplicated on every processor of the parallel machine, and each processor works with a subset of the training set. After each epoch the calculated weight corrections are broadcasted and merged.

The second approach is a parallel calculation of the matrix products that are used in the learning algorithm. The neurons on each layer are partitioned into p disjoint sets and each set is mapped on one of the p processors. After the calculation of the new activations of the neurons in one layer they are broadcasted. We have implemented this on-line training in two variants: For the first parallelization of Morgan et al.[1] one matrix product is not determined on one processor, but it is calculated



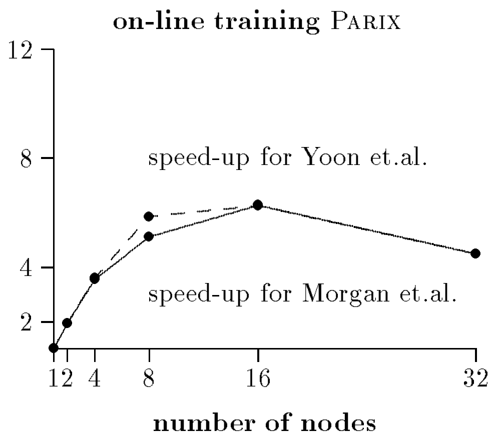


Fig. 4: Speed-ups for GC/PP with PowerPC CPUs

while the subsums are sent around on a processor cycle. The second method of Yoon et al.[5] tries to reduce the communication time. This leads to an overhead in both storage and number of computational operations.

All parallel algorithms are implemented on Parsytec multiprocessor systems based on Transputers and PowerPC processors using the message passing environments Parix and PVM. The measurements took place on a GC/PP at the University of Paderborn, Germany. The speed-ups for parallel training are shown in figure 4.

One can see that the parallelization of the batch learning scales very good. Concerning the on-line training the parallelization of Yoon outperforms Morgan's parallelization a little bit. For 32 processors these parallelizations do not scale anymore because of their enormous communication demands.

6. Conclusions and future research

It has been shown that feedforward multilayer perceptron networks can learn to approximate the time series of sales in supermarkets. For a special group of articles neural networks have been trained to forecast future demands on the basis of the past data. To improve the prediction quality we use additional price and advertising information. Thereby the prediction accuracy is sufficient.

The time consuming back-propagation learning has been parallelized and so accelerated significantly. The necessary training for prediction has been reduced to an acceptable value.

Another approach in order to reduce training time is to minimize the number of input neurons. By correlation analysis we want to find out only the relevant time series that have to be taken into consideration.

For the future the modelling of the input vectors should be improved in order to minimize the prediction error: especially season and holiday information have to be given to the net; the value of changing prices can be modelled quantitatively.

The aim of our research is to develop a forecasting system for supermarkets. This system will reduce stock-keeping costs by flexible adaptability to changing circumstances.

References

[1] N. Morgan, J. Beck, P. Kohn, J. Bilmes, E. Allman, J. Beer. *The Ring Array Processor: A Multiprocessor Peripheral for Connectionist Applications.* Journal of Parallel and Distributed Computing 14, pp. 248-259, Academic Press Inc., 1992.

[2] D.E. Rumelhart, G.E. Hinton, R.J. Williams. *Learning internal representations by error propagation.* In D.E. Rumelhart and J.L. McClelland (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1, pp. 318-362, MIT 1987.

[3] Z. Tang, P.A. Fishwick. *Feed-forward Neural Nets as Model for Time Series Forecasting.* TR91-008, University of Florida, 1991.

[4] V.R. Vemuri, R.D. Rogers. *Artificial Neural Networks – Forecasting Time Series.* IEEE Computer Society Press 5120-05, 1994.

[5] H. Yoon, J.H. Nang, S.R. Maeng. *A distributed backpropagation algorithm of neural networks on distributed-memory multiprocessors.* Proceedings of the 3rd symposium on the Frontiers of Massively Parallel Computation, pp. 358-363, IEEE 1990.