Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'02), Vol. 5
Lipo Wang, Jagath C. Rajapakse, Kunihiko Fukushima,
Soo-Young Lee, and Xin Yao (Editors)

TRAINING ARTIFICIAL NEURAL NETWORKS FOR TIME SERIES PREDICTION USING ASYMMETRIC COST FUNCTIONS

Sven F. Crone

Institute of Business Information Systems, University of Hamburg, Germany

ABSTRACT

Artificial neural network theory generally minimises a standard statistical error, such as the sum of squared errors, to learn relationships from the presented data. However, applications in business have shown that real forecasting problems require alternative error measures. Errors, identical in magnitude, cause different costs. To reflect this, a set of asymmetric cost functions is proposed as novel error functions for neural network training. Consequently, a neural network minimizes an asymmetric cost function to derive forecasts considered preeminent regarding the original problem. Some experimental results in forecasting a stationary time series using a multilayer perceptron trained with a linear asymmetric cost function are computed, evaluating the performance in competition to basic forecast methods using various error measures.

1. INTRODUCTION

Artificial neural networks have found increasing consideration in forecasting theory, leading to successful applications in time series and explanatory sales forecasting. [21,23] In management, forecasts are a prerequisite for all decisions based upon planning. [2] Therefore, the quality of a forecast must be evaluated considering its ability to enhance the quality of this decision. Consequently, the final evaluation of a methods performance in management need to be measured by the monetary costs arising from decisions based on incorrect forecasts. [20] These costs from over- and underprediction are typically not quadratic in form and frequently non-symmetric. [11] For example, in medical inventory management the costs of over- or underpredicting the amount of needed blood-units of a specific blood group can result in highly asymmetric costs. Overprediction may cause inventory-holding costs while underprediction may be fatal. Modest research has been conducted in none-quadratic error functions for neural network training [6,18,22,14] or asymmetric costs for the ex post evaluation in prediction theory [1,11,24,9] However, neural network theory as traditional prediction theory focus on quadratic error functions and least-square predictors [1,16], implying a symmetric and quadratic cost relationship. In this paper, we propose a general asymmetric cost function to minimize the actual error of a forecast, training a multilayer perceptron of arbitrary topology directly with it as a novel error function to find a cost efficient forecast.

Following a brief introduction to the use of neural networks for time-series forecasting, section 3 assesses squared errors and alternative statistical error measures for neural network training. Section 4 introduces asymmetric cost functions to neural network training, providing a formal integration into the back-propagation learning algorithm. This is followed by an experimental evaluation of a neural network trained with an asymmetric cost function versus basic methods for time-series prediction in section 5. Conclusions are given in section 6.

2. NEURAL NETWORKS FOR TIME SERIES PREDICTION

Artificial neural networks (ANN) offer great flexibility in modelling quantitative forecasting methods. Although error measures play an equally important role in explanatory forecasting, modelling causal relationships of variables or between multiple time series [22], this first analysis is limited to time-series point predictions with neural networks. Therefore, a variable $\hat{y}_{t+h}$ is predicted using only previous observations of the same variable y_t, interpreting the time t as the only independent variable.[16]

Following, we consider a feed-forward multilayer perceptron (MLP) of an arbitrary topology; for the impact of alternative network architectures on time series prediction see Azoff [5] or Zimmerer [26]. At a point in time t $(t=1,...,T)$, a one-step ahead forecast $\hat{y}_{t+1}$ is computed using n observations $y_t, y_{t-1}, \mathrm{K}, y_{t-n+1}$ from n preceding points in time $t, t-1, t-2, ..., t-n+1$, with n $(n=1,...,N)$ denoting the number of input units. This models a time-series prediction in analogy to an non-linear autoregressive AR(n) model [16] of the form

$$\hat{y}_{t+1} = f(y_t, y_{t-1}, ..., y_{t-n+1}) \quad . \qquad (1)$$

A network architecture is displayed in figure 1. The task of the MLP is to model the underlying generator of the data during training, so that a valid forecast is made when the

trained network is subsequently presented with a new value for the input vector. [6]

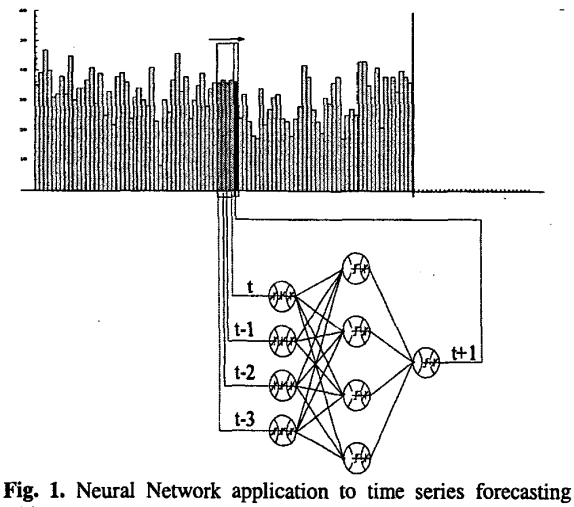

Fig. 1. Neural Network application to time series forecasting with a (4-4-1) MLP, using *n=4* input neurons for observations in *t, t-1, t-2, t-3*, four hidden units, one output neuron for time period *t+1* and two layers of 20 trainable weights. [23]

Unless a network is perfectly trained, these network outputs differ from the desired outputs. The network learns the underlying relationship through minimization of this difference on the training data. The real-world significance of these deviations depends on the application and is measured by an objective function, also called error function, whose output rates the quality of the networks response. [18] As the specification and estimation of an objective function calls for an actual application, we consider alternative error functions for sales predictions in inventory management.

In forecasting with ANNs, as in prediction in general, error functions are applied in all phases of modelling, network selection and application. During modelling, errors measures are computed and minimised to estimate parameters fitting a forecasting method to the data, in neural network terminology referred to as training. After finding valid parameters, error measures are calculated to verify the ex post quality of a single method, to compare forecast results of different architectures of a method or to compare results of competing methods. During the application of a chosen method, error measures may act as tracking-signals for constant evaluation, model-adaptation or retraining. [20] Although error functions play a predominant role in neural network forecasting, standard statistical error measures are routinely used instead of the actual objective function. As the training of a network determines its validity and reliability in forecasting, we focus the following analysis on the use of error measures as objective functions in ANN training.

3. NEURAL NETWORK TRAINING USING STATISTICAL ERROR MEASURES

3.1. Neural Network Training using Squared Errors

Supervised online-training of a MLP may be achieved using various training methods. One commonly used class of algorithms consists of the back-propagation algorithm with its extensions using dynamic learning rates, momentum terms, weight decay, batch calculation etc. for enhanced performance regarding learning time and accuracy. [25,18]

Training a MLP with a back-propagation derivative is the task of adjusting the weights of the links w_{ij} between units j $(j=1,...,J)$ and adjusting their thresholds to achieve a desired system behaviour. [25,18] Generally, a pattern p $(p=1,...,P)$, consisting of n observations $y_t, y_{t-1},..., y_{t-n+1}$ from n points in time t, $t-1$, $t-2$,..., $t-n+1$, is presented to the input layer of n units and propagated forward. An error δ_j of the network behaviour is calculated using a specific error function, measuring the difference between the desired output t_j and the actual output o_j of the network. [18,19] Using derivatives Δw_{ij} of the error, the weights w_{ij} are adjusted to minimize the error δ_j, using gradient descent on a single-pattern error. [18] The error δ_j of units in hidden layers is computed using the error of following layers, thus propagating the errors backwards through the network unto the input layer. This process is repeated for each pattern of a dataset and several epochs. [25] In time series point prediction, the single network output o_p corresponds to the forecast $\hat{y}_t$ of a network, while the teaching input t_p represents the actual value of the observation y_t for each point in time t. The function determining the size of the error δ_j should reflect the significance of this difference depending on the underlying learning problem. [18] Back-propagation algorithms traditionally minimize a modified sum of squared errors (*SSE*), ever since the popular description of the original algorithm by Rumelhart, Hinton and Williams [19]:

$$E_p = \frac{1}{2}\sum_j (t_{pj} - o_{pj})^2 \quad . \tag{2}$$

The factor ½, amending the standard *SSE*, is chosen for convenience in deviation. [18] Using this modified *SSE*, the calculation of the errors δ_j for the weight updates is computed using the equation

$$\delta_{pj} = \begin{cases} (t_{pj} - o_{pj}) f_j'(net_{pj}) & \forall\ j \text{ in the output layer} \\ f_j'(net_{pj}) \sum_k \delta_{pk} w_{pjk} & \forall\ j \text{ in a hidden layer} \end{cases}, \tag{3}$$

showing the derivation of the modified *SSE* being directly embedded within the back-propagation algorithm. Consequently, most neural networks simulators offer only the modified *SSE* as an error function for back-propagation algorithms, although other error measures are feasible and

also occasionally calculated for the ex-post evaluation of network architectures. The consistent use of the modified *SSE* is motivated primarily by analytical simplicity [6], e.g. the advantages of easy differentiability and the independence of single errors, [18,25] and the similarity of hetero associative problems to statistical regression problems, modelling the conditional distribution of the output variables. [6] Therefore, the implications of using the *SSE* or alternative error measures for training require further discussion.

3.2. Alternative Error Measures for Neural Network Training

Each error measure should be selected in accordance to the underlying problem structure, representing approximations of the objective function. Consequently, a variety of alternative statistical error measures have been developed for dissimilar purposes, all derived from a simple forecast error. The basic forecast error e_{t+h} for a time period $t+h$ $(t=1,...,T)$, with h $(h=1,...,H)$ denoting the number of time periods forecasted into the future, is calculated as the difference between the actual observation y_{t+h} and the forecast $\hat{y}_{t+h}$ [11,16], using

$$e_{t+h} = y_{t+h} - \hat{y}_{t+h} \quad . \tag{4}$$

From this basic error, alternative errors are derived, such as the absolute error (*AE*), with

$$AE_{t+h} = \left| y_{t+h} - \hat{y}_{t+h} \right| \quad , \tag{5}$$

or the squared error (*SE*), with

$$SE_{t+h} = \left(y_{t+h} - \hat{y}_{t+h} \right)^2 \quad . \tag{6}$$

If time series of observations exists within a time period l $(l,l+1,...,T)$, with $T-l$ observations and corresponding forecasts, the calculation of statistical error measures such as summed or mean errors is required, using the single errors of each observation. These are also applicable if more than one value is forecasted or errors are cumulated over numerous patterns, e.g. in batch back-propagation. Otherwise single errors apply to ANN learning.

Although all single errors and resulting error measures produce a value of 0 for an optimal forecast and are symmetric about $e_{t+h} = 0$, each error measure implies a different weight for a deviation of the forecast value from the real value. Quadratic error measures, as the *SSE*, penalize a forecast more for extreme deviations than for small ones [16], while absolute error measures give identical weight to every error regardless of scale. [3,4] This effect may be seen in the difference of the *SE* and the *AE* displayed in Fig. 3.

Choosing the *SSE* as a forecast error criterion implies the preference of many small deviations versus fewer large deviations. In the case of noisy data, this may lead to the selection of a method that forecasts the underlying time-series pattern less valid but is more robust against outliers. Consequently, neural networks might be trained to anticipate outliers instead of generalizing from the actual patterns. In addition to overestimating large errors, with $e>1$, squared errors underestimate small errors, with $e<1$. As data is often rescaled to intervals of [0;1] or [-1;1] to allow faster training, the errors are reduced in magnitude as well, underestimating the actual error and altering the error surface further. Applying the modified *SSE* (2) alters the objective function even further, implying a halved quadratic cost relationship.The error surface for a given dataset takes on a distinct form in dependence of the used error. Therefore the choice of error is crucial in parameter optimisation in ANN training with a local search algorithm such as gradient descent, as the performance of the algorithm depends on the slope and distribution of the multidimensional error surface. [14] Steepening or flattening the error surface may result in different learning behaviour, converging to different local or global minima. The use of the modified *SSE* flattens the slope of the error through a monotonous transformation of the surface but without alteration of the local and global minima [18].

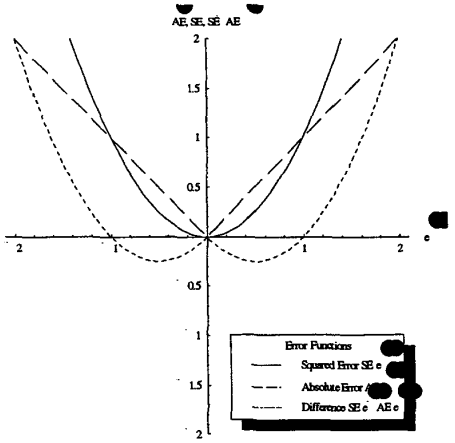

Fig. 2. Error functions *AE*, *SE* and the Difference of *SE-AE*

Due to these properties, the modified *SSE* may not be a valid and reliable error measure, even for business applications with a symmetric, quadratic cost relationship. In inventory control, the errors arising from over- and underprediction are usually considered to be non quadratic but linear in form, implying the use of absolute error measures. Therefore, selected authors have proposed the use of alternative error-measures to neural network training. Thiesing proposes the ln(cosh)-function, similar to the symmetric *AE* error function in form but allowing full differentiation [22]. Hanson et al. [13] as Bishop [6] propose the Minowski-R power metrics as an error function for a generalised training, allowing the use of the city block metric equalling the *AE*, the Euclidian metric

equalling *SE* and higher metrics through parameterisation. Reed proposes a piecewise linear error function with upper and lower tolerance limits with an error of zero within the limits [18]. Nonetheless, all error measures proposed apply symmetric error functions as an approximation of the true cost relationship. However, the cost arising in management forecasts are often not only non quadratic, but also non symmetric in form. Therefore a new set of error measures is introduced and applied directly to neural network training: asymmetric cost functions.

4. NEURAL NETWORK TRAINING USING ASYMMETRIC COST FUNCTIONS

4.1. Asymmetric Cost Functions in Forecasting

In business management, all forecast are generated as a prerequisite of business decisions. Through decisions based on sub optimal forecasts costs arise to the decision maker in choosing a wrong alternative. Although the amount of costs will generally increase with the numerical magnitude of the errors, the sign of the error plays a significant role. Regarding business forecasts, the costs arising from over- and underprediction are frequently non symmetric and typically non quadratic in form [11,10].

Granger develops an asymmetric linear cost function for the ex-post evaluation of a forecast in inventory management problems.[12] The future sales $\hat{y}_{t+h}$ of a stock keeping unit (SKU) at a point in time $t+h$ is predicted. The inventory is restocked exactly to this amount $\hat{y}_{t+h}$, adding additional SKUs or taking SKUs out of stock in order to satisfy demand while minimising the costs of stock-keeping. [15] Evaluating the inventory decision after the forecast period $t+h+1$, three scenarios are feasible. If the forecast was exact, no costs arise. If the forecast was higher than actual sales, the units that are overstocked lead to inventory holding costs per SKU, denoted by a. If the forecast was lower than the actual sales, a stockout arises, resulting in costs of lost sales revenue per SKU, denoted by b. The resulting *LINLIN* cost function yields:

$$LLC(y_{t+h}, \hat{y}_{t+h}) = \begin{cases} a|y_{t+h} - \hat{y}_{t+h}| & \text{for } y_{t+h} < \hat{y}_{t+h} \\ 0 & \text{for } y_{t+h} = \hat{y}_{t+h} \\ b|y_{t+h} - \hat{y}_{t+h}| & \text{for } y_{t+h} > \hat{y}_{t+h} \end{cases} \quad (7)$$

The *LINLIN* cost function (*LLC*) is linear to the left and right of *0*. The parameters a and b give the slopes of the branches for each cost function and measure the costs of error for each SKU difference between the forecast $\hat{y}_{t+h}$ and the actual value y_{t+h}. For $a \neq b$ these cost functions are non-symmetric about *0* and are therefore called asymmetric cost functions or asymmetric loss functions. The degree of asymmetry depends on the ratio of a to b. [9] The names reflect the functional form of the branches, in analogy to Varian's *LINEX* loss function. [9] For $a = b$ the *LLC* equals a linear symmetric cost function using a symmetric cost parameter. For $a = b = 1$ the *LLC* equals that the function of the statistical error measure *AE*. Therefore, symmetric cost functions as well as statistical error measures may be interpreted as special cases of the corresponding asymmetric cost functions. The shape of one particular asymmetric *LINLIN* cost function as a linear approximation of a cost function taken from a current project in inventory management, and used in the subsequent experiment, is given with the *AE* in Fig. 3.

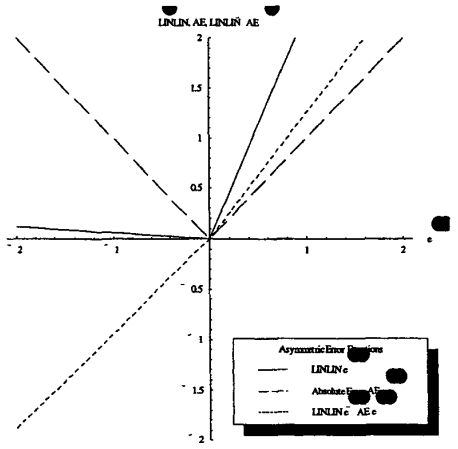

Fig. 3. Empirical Asymmetric Cost Function showing cost arsing for over- and under-prediction of pack sales at cigarette vending machines in the UK, using a=£0,014 and b=£2,25.

Matching the form of the cost function to the actual costs arising from the decisions, which are often linear but may take on various forms, is essential for valid decisions in stock control. Although other linear [1], quadratic [9] and mixed [24,1,17,9,8] asymmetric cost functions have been specified, we limit our analysis to a simple *LLC* function as a valid approximation of the cost function in our inventory problem. The actual function and the parameters a and b are easily estimated through standard accounting procedures.

In prediction theory, these asymmetric cost functions have been applied to estimate the ex post performance of a forecasting method as well as an ex post adjustment of the predictor. Granger [11] analyses *LINLIN* cost functions regarding optimal prediction under conditionally Gaussian processes. These results have been extended to unconditionally non-Gaussian processes and general loss functions. [9] In the following section, their use is extended towards a direct training of a neural network. Asymmetric transformations of the error function alter the error surface significantly, resulting in changes of slope and creating different local and global minima. Therefore different solutions may be found minimizing these cost functions with gradient descent, finding a cost minimum prediction for the underlying problem.

4.2. Asymmetric Cost Functions in Back-Propagation Training

In order to allow training of a neural network with an asymmetric cost function, the standard back-propagation algorithm is modified. Drawing upon equation (3) we derive a generalisation of the back-propagation rule, which is independent of a specific error function. Instead of (2) we allow the use of a general cost of error function

$$E_p = C(t_{pj}, o_{pj}) \quad , \tag{8}$$

which permits the use of arbitrary cost functions. As we have seen above, symmetric cost functions and standard statistical error functions are special cases of an asymmetric cost function. This allows the selection of a problem specific performance measure for training through modification of the parameters of the general cost of error function. Modifying the subsequent equations in accordance with Rumelhart et al. [19] we receive

$$\delta_{pj} = \begin{cases} \dfrac{\partial C(t_{pj}, o_{pj})}{\partial o_{pj}} f_j'(net_{pj}) & \forall\, j \text{ in the output layer} \\ f_j'(net_{pj}) \sum_k \delta_{pk} w_{pjk} & \forall\, j \text{ in a hidden layer} \end{cases} , \tag{9}$$

which is a simple generalisation of the error term of the back-propagation rule. The importance of the cost function in the output layer for the weight adaptation within the network becomes evident. It is sufficient to amend the cost function calculated for the units in the output layer, as it is consequently propagated backwards through the net, leaving the equations for units in hidden layers unchanged.

The choice of a semi linear activation function remains independent from the parameterisation of the cost function. Therefore, different activation functions may be chosen in order to enhance neural network performance.

However, the cost function used in the output layer needs to be a fully differentiable function to allow the calculation of the error's derivatives for training with gradient descent algorithms. This poses a potential problem, as the *LINLIN* cost functions are not fully differentiable. If no differentiable approximation of the actual cost function is known, Reed proposes the use of gradient estimates or the conjugate-directions method [18], calling for a modification within the ANN software simulator for computation of results. [22] Alternatively, global search methods as simulated annealing, threshold acceptance or genetic algorithms may be required. [18]

5. SIMULATION OF NEURAL NETWORKS USING ASYMMETRIC COST FUNCTIONS

5.1. Experimental Design of Forecasting Methods

We conduct an experiment to evaluate the ability of a MLP to evolve a set of weights to minimize an asymmetric cost function with back-propagation. As the experiment design plays an important rule for the reliability of the results it is described in detail.

The experiment is computed using a noisy time series from a current project, forecasting weekly cigarette pack sales of one stock-keeping unit (SKU) at a vending machine. A sample of $n=89$ observations is split into consecutive datasets, using 64 observations for the training-, 15 for the validation- and 14 for the test-set, resulting in 60, 15 and 10 patterns in each set.

We consider a fully connected MLP without shortcut connections as displayed in Fig. 1, with a topology of 4 input, 4 hidden, 1 output and 1 bias unit to model the thresholds for all units in the hidden and output layer. All processing units use a summation as an input-function, the tanh as a semilinear activation function and the identity function as an output function. Two sets of networks were trained. One set was trained on minimizing the symmetric *SSE*, the other was trained minimizing a *LINLIN* asymmetric cost function. The parameters of the *LLC* were computed from the same project, resulting in a=£0,0144 and b=£2,25 for equation (7). Each MLP was initialised and trained for five times to account for [-1;1] randomised starting weights. Training consisted of 2000 epochs with a validation after every epoch, using a hold out method for cross-validation. After training, the results for the best network, chosen on its performance on the validation set, as well as the average of all five networks are computed for the test-set-data to measure generalisation.

For comparison, the basic forecasting methods Naïve 1a, using last periods sales as a forecast, Naïve 1x with a simple forecasting-strategy of always restocking to the maximum capacity of 60 SKUs, and variations of 1st order exponential smoothing are computed. [7] The α-parameter for exponential smoothing was computed minimizing the *SSE*, *AE* and *LINLIN* on the training set respectively. Supplementary, we calculate exponential smoothing forecasts including safety stocks, for z=2,0 and z=2,326 standard deviations of the forecast errors, aiming at 97.8% and 99.0% service levels assuming a Gaussian distribution. No parameter was estimated using data from the test-set. The experiment was implemented using NeuralWorks Professional II with an additional error function table to bias the calculated standard error.

5.2. Experimental Results on Asymmetric Costs

Table 1 displays the results using mean error measures computed on the test-set. Descriptive performance measures of the mean number of stockout units (*MSU*), giving the amount of suppressed sales per observation, and the number of overstocked units (*MOU*) are calculated. An asymmetric ex-post performance measure is calculated, denoting the ex post mean *LINLIN* costs (*MLLC*) resulting from a given forecast method. The methods are ranked by performance for the mean *SE (MSE)* and the *MLLC*.

Table 1. Experimental Results for Statistical Forecasting Methods and ANNs trained on Asymmetric Costs and Squared Error Measures

Performance Measure / Simple Forecast Method	Statistical Measures		Descriptive Measures		Cost Measures	Rank	
	MAE	*MSE*	*MOU*	*MSU*	*MLLC*	*MSE*	*MLLC*
Naïve 1a	8.90	112.90	4.40	4.50	10.19	8	12
Exp. Smoothing minimizing *SAE*	5.60	43.80	1.60	4.00	9.02	3	9
Exp. Smoothing minimizing *SSE*	5.30	40.30	1.90	3.40	7.68	2	8
Exp. Smoothing minimizing *LINLIN*	4.90	36.90	2.80	2.10	4.77	1	7
Artificial Neural Network Forecasts							
ANN trained on *SSE*, best network	5.70	46.10	1.70	4.00	9.02	4	9
ANN trained on *SSE*, average of 5 nets	5.98	48.58	1.82	4.16	9.39	5	11
ANN trained on *LINLIN*, best network	8.70	97.50	8.70	0.00	0.13	7	1
ANN trained on *LINLIN*,- average of 5 n.	6.98	72.50	6.62	0.36	0.99	6	6
Inventory Levels from Forecast Methods							
Exp. Smoothing minim. *SSE*, sl *97.8%*	9.60	130.20	9.60	0.00	0.14	9	2
Exp. Smoothing minim. *SSE*, sl *99.0%*	11.60	172.60	11.60	0.00	0.17	11	4
ANN trained on *SSE, service level 97.8%*	9.70	134.90	9.70	0.00	0.14	10	2
ANN trained on *SSE, service level 99.0%*	11.70	177.70	11.70	0.00	0.17	12	4

Various results may be drawn from the experiment. The ANN trained with the asymmetric *LINLIN* cost function gives a superior forecast, achieving the lowest mean costs on the test-data with 0.13. It exceeds all methods using safety stocks and clearly outperforms single forecasts of ANNs trained with the *SSE* criteria and exponential smoothing on any error measure. But, as the average of the five networks trained with *LINLIN* costs shows an inferior performance compared to exponential smoothing with safety stocks, these findings may not be generalised, However, all five networks minimize the asymmetric cost function robustly. ANNs trained with *LINLIN* costs show a significantly high *MSE* and vice versa. Therefore, a MLP may be used successfully to minimize an arbitrary cost function distinct from the squared error paradigm.

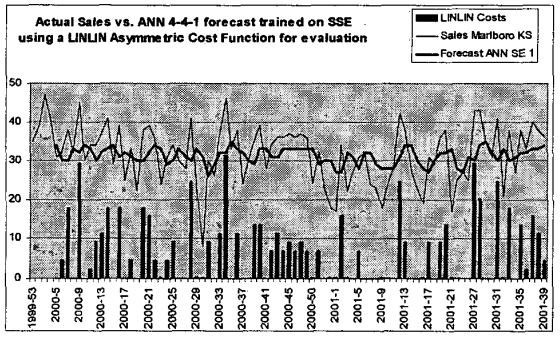

Fig. 4. ANN trained to forecast cigarette pack sales minimizing a *SSE*, showing the difference between actual sales and the ANN forecast measured by the asymmetric linear cost function.

In regard to the asymmetry of the costs function, the neural network raises its predictions to achieve a cost efficient forecast and a cost efficient inventory level, avoiding costly stockouts. This may be seen in comparison of figures 4 and 5. In addition, the ANN trained with an asymmetric cost function outperforms ANNs and exponential smoothing inventory procedures, finding a cost efficient inventory level without the separate calculation of safety stocks. From a business management perspective, this reduces the complexity of the overall management process of stock control, calculating a cost efficient stock level directly from the forecasting method.

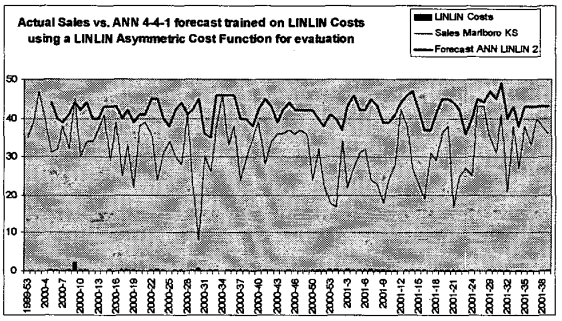

Fig. 5. ANN trained to minimize a *LINLIN cost function.* The difference between actual sales and the forecast measured by the asymmetric cost function, barely noticeable near the x-axis.

Those ANNs trained to minimize the *SSE* perform less well on the test-set than exponential smoothing methods. This, as some other inconsistencies in the results of the exponential smoothing methods, indicates a possibly inhomogeneous initial distribution of the data within sets, requiring additional testing. The ANNs trained on the *SSE* with additional safety stocks perform equally well as exponential smoothing methods with safety stocks. A general superiority of ANNs over statistical methods cannot and shall not be derived from this experiment.

The experiment further validates, that choosing a network or an alternative forecasting method through an ex-post comparison of the *MSE* or the *MAE* may lead to the selection of an inferior method, leading to sub optimal solutions regarding the actual cost situation. Therefore, cost of error measures should be applied to the ex post evaluation of real-world forecasting applications.

6. CONCLUSION

We have examined symmetric and asymmetric error functions as performance measures for neural network training. The restriction on using squared error measures in neural network training may be motivated by analytical simplicity, but it leads to biased results regarding the final performance of forecasting methods. Asymmetric cost functions can capture the actual problem structure and allow the minimization of relevant costs using standard methods, similar to minimizing an arbitrary statistical error function. Standard multilayer perceptrons can minimize these asymmetric cost functions robustly, using a modified back-propagation training algorithm. Our approach to train neural networks with asymmetric cost functions has a number of advantages. Minimizing an asymmetric cost function allows the neural network to learn directly from actual cost functions, taking the model building process closer towards business reality. For instance, considerations of optimum service levels in inventory management may be incorporated within the forecasting process, leading directly to the forecast of a cost minimum stock level without further computations.

However, the limitations and promises of using asymmetric cost functions with neural networks require systematic analysis. Future research may incorporate the modelling of dynamic carry-over-, spill-over-, threshold- and saturation-effects for exact asymmetric cost functions where applicable. In particular, verification on multiple time-series, other network topologies and architectures is required, in order to evaluate current research results.

REFERENCES

[1] G. Arminger, N. Götz, Asymmetric Loss Functions for Evaluating the Quality of Forecasts in Time Series for Goods Management Systems - Technical Report 22/1999, Dortmund, 1999

[2] J.S. Armstrong, Strategic Planning and Forecasting Fundamentals, K.Albert (ed.), Strategic Management Handbook, McGraw-Hill, New York, 1983, 2, 2-1 - 2-32, 1983

[3] J.S. Armstrong, Error Measures for Generalizing About Forecasting Methods -Empirical Comparisons, International Journal of Forecasting, 1992, 8, 69-80, 1992

[4] J.S. Armstrong, Evaluating Forecasting Methods, in J. S. Armstrong (ed.), Principles of Forecasting - A Handbook for Researchers and Practitioners, Kluwer Academic, Boston, 2001, 443-472, 2001

[5] E.M. Azoff, Neural Network Time Series Forecasting of Financial Markets, John Wiley & Sons, New York, 1994

[6] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, 1995

[7] R.G. Brown, Statistical Forecasting for Inventory Control, McGraw-Hill, New York, 1959

[8] P.F. Christoffersen, F.X. Diebold, Further Results on Forecasting and Model Selection under Asymmetric Loss, Journal of applied econometrics, 11, 1996, 5, 561-572, 1996

[9] P.F. Christoffersen, F.X. Diebold, Optimal prediction under general asymmetric loss, Econometric Theory, 13, 808-817, 1997

[10] M.P. Clements, D.F. Hendry (eds.), Forecasting economic time series, Cambridge University Press, Cambridge, 1998

[11] C.W.J.Granger, Prediction with a Generalized Cost of Error Function, Operational Research Society (eds.), Operational Research Quarterly, 20, S. 199-207, 1969

[12] C.W.J.Granger, Forecasting in Business and Economics, Academic Press, New York, 1980

[13] S.J.Hanson, D.J.Burr, Minowski-r Back-Propagation Learning in Connectionist Models with non Euclidian Error Signals, in D.Anderson (ed.), Neural Information Processing Systems, American Institute of Physics, New York, 1987, 348-357, 1987

[14] J. Hertz, A. Krogh, R.G. Palmer, Introduction to the Theory of Neural Computation, Addison Wesley, Redwood City, 1991

[15] S.D. Krane, The Distinction between Inventory Holding and Stockout Costs - Implications for Target Inventories, Asymmetric adjustment, and the effect of Aggregation on Productions Smoothing, International Economic Review, 35, 1, 117-136, 1994

[16] S.Makridakis, S.C.Wheelwright, R.J.Hyndman, Forecasting Methods and Applications, Wiley, New York, 1998

[17] A. Parsian, N.S. Farsipour, Estimation of the mean of the selected population under asymmetric loss function, O. Anderson, et al. (eds.), Metrika, International Journal for theoretical and applied statistics, 50, 2, 89 - 107, 1999

[18] R.D. Reed, R.J. Marks II, Neural Smithing - Supervised Learning in Feedforward Artificial Neural Networks, MIT Press, Camebridge, MA, 1998

[19] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors - (reprint), in J.A. Anderson, E. Rosenfeld (eds.), Neurocomputing – Foundations of Research, MIT Press, Camebridge, MA, 1, 696-699, 1988

[20] J. Schwarze, Statistische Kengrößen zur Ex-Post-Beurteilung von Prognosen - (Prognosefehlermaße), in J. Schwarze (ed.), Angewandte Prognoseverfahren, Neue Wirtschafts-Briefe, Herne, 1980, 317-344, 1980

[21] Z. Tang, P.A. Fishwick, Feed-forward Neural Nets as Models for Time Series Forecasting - Technical Report TR91-008 Computer and Information Sciences, University of Florida, 1991

[22] F.M. Thiesing, Analyse und Prognose von Zeitreihen mit Neuronalen Netzen, Shaker, Aachen, 1998

[23] F.M. Thiesing, O. Vornberger, Sales Forecasting using Neural Networks, IEEE (eds.), Proceedings ICNN'97 - Houston, Texas, 9-12 June 1997, 4, 2125-2128, 1997

[24] H.R. Varian, A Bayesian approach to real estate assessment in S. E. Fienberg, A. Zellner (eds.), Studies in Bayesian econometric and statistics in Honor of Leonard J. Savage, North-Holland Publishing, Amsterdam, 195-208, 1975

[25] A. Zell, Simulation neuronaler Netze, Oldenbourg, München, 1997

[26] T. Zimmerer, Künstliche Neuronale Netze versus ökonometrische und zeitreihenanalytische Verfahren zur Prognose ökonomischer Zeitreihen, Peter Lang, Frankfurt a.M., 1997