Structure Optimization of Neural Networks in Relation to Underlying Data

Marijana Zekic
University of Osijek
Faculty of Economics
tel: +385 31 224-400
e-mail: marijana@oliver.efos.hr

Abstract. Optimization of neural network topology has been one of the most important problems since neural network came in front as a method for prediction, classification, and association. Number of heuristics formulas for determining the number of hidden units were developed (Masters,T., 1993, Marcek, D., 1997), and some algorithms for structure optimization were suggested, such as cascading, pruning, A* algorithm, and others. The connection between optimization techniques and underlying data in the model is not investigated enough. The paper deals with the influence of variable selection and statistics of input and output variables to several algorithms for structure optimization. Principal component analysis and analysis of variance among other statistical tests are conducted in stock return prediction models. The predictive power of neural networks is captured, and also the sensitivity of the dependent variables to changes in the inputs.

Keywords: structure optimization of neural networks, cascading, pruning, variable selection, principal component analysis

1. Introduction

Although latest research in NNs introduces various new techniques for NN optimization, it still suffers from undefined paradigms about the connection between NN model and underlying data (Zahedi, 1996). The paper tests some structure optimization techniques and observes their performance in relation to variable selection, range and type of data (prediction vs. classification), and transfer functions used. First section describes the process of variable selection for total return prediction on S&P stock market using principal component analysis, general linear regression, and multiple linear regression. Several models are extracted and tested by NNs using optimizing techniques such as: cascade correlation, pruning, and Masters' heuristic formula. Finally, sensitivity analysis is conducted to perform "what-if" analysis of the best NN models.

2. Variable selection

Due to a large number of financial ratios that could be used in financial analysis, variable selection is very important phase in designing NN model and has been approached in different ways. Some authors use traditional normative approach that selects variables according to their relevance to various aspects of decision maker interests (Rees, 1990), while statistical approach aims to find the evidence for the rejection of ratios that do not provide additional information to the remaining ratios. Latter approach evolved from correlation analysis towards factor analysis or principal component analysis. Since principal component analysis generates a linear combination of input variables in the way that composite factors contain as much of information or variability as possible, its disadvantage is in linearity. However, it can provide us with useful information for modeling.

2.1. Description of variables and data sample

Total return on stock is observed as rate of return that reflects price appreciation plus reinvestment of monthly dividends and the compounding effect on dividends paid on reinvested dividends (Compustat, 1997). 4 financial ratios emphasized as important factors for stock return in previous research (Yoon, Guimaraes, Swales, 1994) are used as the input variables:

- current ratio (CR),
- return on equity (ROE),
- price/equity (P/E),
- price/sales (P/S),

with additional 2 variables:

- total assets (TA), and
- net sales (NS).

Last two variables are used to extend already used model, for two purposes: (1) examining the difference in the influence of data type (ratios vs. absolute values) to optimizing techniques, and (2) improving the performance of NN. All the variables reflect quarterly data of 100 US companies listed on Standard & Poor 100 index, ranging from March 1989 till June 1997, available in Compustat database. Sample size was 2326 cases, divided into three separate samples: train sample (60% or 1396 cases), test sample (20% or 465 cases), and validation sample (20% or 465 cases). In order to achieve generalization capability of a NN, the test sample is used to obtain the best structure with optimization techniques, while the validation sample is used to validate the best architecture for each model on out-of-sample data.

Descriptive statistics conducted on the total sample shows that there is a small standard deviation in input variables: CR (0.17) and P/S(1.27), and a large standard deviation in input variables: ROE (18.17), P/E (81.14), TA (15611.78), and NS (4550.315). The output variable has also large deviation: 31.71. Lowest total return rate in the sample is -72.5, and highest is 222.93. Such large variance is partially caused by the cross-sectional nature of data (100 companies with different size and successfulness), and therefore makes prediction accuracy difficult to achieve. Total return, as output variable has the following distribution: 23% of cases are with negative return, 0.04% of cases with zero return, and 76.96% of cases with positive total return rate.

2.2. Principal Components Analysis and Multiple Regression

In our experiment, 5 of 6 variables are selected using principal component analysis. Variables TA and NS showed very high correlation (0.88), and they also form the first extracted factor that explains 35.07% of variance. Next three significant factors consist of different combinations of other 4 variables, but explain much smaller portion of variance (below 20% each). Therefore, we can replace last two input variables (TA and NS) by one variable. Because of its higher influence to the factor, NS is selected for the model, which now consists of CR, ROE, P/E, P/S, and NS as input variables. Correlation among other 4 variables is very low, which allows us to leave other 4 variables in the model.

To extract models with regression techniques, we conducted generalized linear model (GLM) which assumes that the response y(i) has distribution from one of exponential family (normal, inverse Gaussian, gamma, Poisson, binomial). The GLM shows high F value, but variables ROE and P/E are not significant for the model.

Forward multiple linear regression (MLR) excluded P/E and NS variable from the model. The same result is obtained by stepwise linear regression. Backward regression removed variables ROE and P/S from the model.

In order to see the connection between underlying data and optimization procedures of NN model, we are going to use the model with all 6 available variables, then statistically generated models, and finally Yoon, Guimaraes, and Swales' model (1994) to compare optimizing techniques.

Table 1. Models tested

Model	Number of input variables	Input variables
1	6	CR, ROE, P/E, P/S, TA, NS
2	5	CR, ROE, P/E, P/S, TA
3	4	CR, P/S, TA, NS
4	4	CR, ROE, P/S, TA
5	4	CR, ROE, P/E, P/S

Second model is obtained by principal component analysis and contains only one variable with absolute value, others are ratios. Models 3 contains two ratios and two absolute value variables, since model 4 uses three ratios and one absolute value variable. Last, 5th model consists of all 4 ratio variables.

3. Optimizing NN structure

All NN learning algorithms are actually optimizing the objective function in the network. Learning algorithms, such as backpropagation and others, optimize the network weights on predefined structure. The problem is how to define the best structure that can lead to an optimal NN result. It is proved (by Hornik, Kurt, Stinchombe, Maxwell and White, Halbert, 1989, described in Masters, 1993) that "three-layer networks having arbitrary squashing activation functions are capable to approximate any Borel measurable function from one finite dimensional space to another to any desired degree of accuracy". In many cases, more layers slow down the training time because the gradient of the error is more unstable and there is a higher risk of local minima. Masters (1993) also claims that more than one hidden layer is needed only with discontinuing functions, i.e. the functions that are mostly continuous, but have few sudden jumps that cannot be easily predicted by one hidden layer. To obtain the best NN structure, most of the authors have just gradually added new neurons, comparing the test results. One of the first techniques for structure optimization was skeletonisation, developed by Mozer, 1989 (in Quinlan, 1998) who showed that processing units (nodes) vary in their functional importance for solving problem. Therefore, it is possible to determine the relevance of a unit by comparing performance of the NN when the unit is included to the performance of the NN when the unit is removed.

Besides some heuristic formulas, there are several common techniques used for NN structure optimization, such as Cascade-correlation (or cascading), pruning, genetic algorithms, and some others. Quinlan (1998) synthesizes a number of optimization techniques and he groups them in the following way:

a) Backpropagation derivatives

1) removal of units techniques:
- skeletonisation
- artificial programmed cell death
- gauging the optimal size of network in terms of generalization capability introduced by Sietsma, 1991

2) pruning connections techniques
- Thodberg's technique of pruning the connections with small weights
- Optimal brain damage, introduced by Le Cun at al, 1990

3) adding units and connections
- Meiosis networks - weights are observed according to the distribution of values rather than by a single value, Hanson, 1990
- Dynamic node creation, Ash, 1989

4) both adding and pruning units and connections
- Bartlett's technique
- Hirose et al., 1991 – network is allowed to overcome the problem of local minimum.

b) algorithms for discrete unit networks
- Tiling algorithm, Mezard et al.
- Pointing and tower algorithms, Smieja, 1993
- Neural tree approach, Sirat et al.
- Upstart algorithm, Frean, 1990

Most of those techniques are based on two common principles: gradual adding new nodes or gradually excluding the nodes, using different criteria. Another shared characteristic of those techniques is that they make the network dynamic in a sense of changing its structure.

One of the most important ways to determine the best number of hidden neurons is cascading or Cascade-correlation algorithm proposed by Fahlman (described in Doering, Galicki, Witte, 1997), that starts with a small number of hidden neurons and adds a new neuron in each additional training and testing phase. Cascading can be performed by keeping the learned weights for the next test, or by retraining all hidden neurons again. Masters suggests to preserve the learned weights, but it is necessary to try the same architecture with random starting points because the network, which is non-linear, has totally different approach when a new neuron is added. There is a high risk of local minimum if the network is not initialised before each new training phase. In the NeuralWare software used in experiment, cascading procedure stops when the maximum number of epochs set by the user is reached or when the stability condition is obtained. The stability condition can be described as:

$$\left|E_c - E_p\right| < T$$

where E*c* is the RMS error over the current epoch, E*p* is the RMS error in previous epoch, and T is the specified error tolerance.

In our experiment, we have used cascading with preserved weights, max. number of hidden neurons was set to 20, candidate pool size 5, vector dimension 3, and error tolerance of 0.05. Maximum number of learning iterations was set to 1000 epochs, where epoch size was 16 cases.

The idea of self-pruning is in simultaneous minimization of output error and minimization of the number of hidden neurons. It is performed by "optimizing composite function of the output error with the hidden-neuron economisation criterion". Although Masters' does not recommend this method because it does not directly deal with the problem of overfitting, many authors use it as effective method for optimizing number of hidden nodes, or optimizing the weights in the NN.
This method is used in the paper with max. number of 20 hidden neurons, in 5 test intervals of each 5000 of learn iterations (max. number of 100 000 iterations), with error tolerance 0.975.

The main disadvantage of both cascading and pruning procedures is that upper limit of number of hidden nodes should be determined in advance, while there is no proof of how to determine it. One way to overcome this limitation is to use genetic algorithms that start with different, randomly chosen topologies.

Genetic algorithms are effectively used for optimizing NN structure by several authors (Wallrafen, Protzel, Popp, 1996). They use the principle of guidance and random search algorithm to determine the structures, test and compare the results in order to find the best one. This technique can be used to optimize weights, or the number of neurons and layers in the network.

It is important to experiment other, not so common, optimizing techniques, such as analyzing the variance of output variables using the principal components analysis, proposed by Johnston and Wichern, 1982 (in Marcek, 1997). It can be used when there are more than one output variables in the model, since it determines the number of hidden layers on the basis of the eigenvalues and eigenvectors of output variables. If the l
additional output variable (s=1,2,…,j), where j is the number of outputs in the model,

$$^{c}l_1 \geq {}^{c}l_2 \geq \ldots \geq {}^{c}l_j$$

that explains at least 95% of variance ($^{c}l_s \geq 95$). The first *s* that satisfy that criteria can be the number of hidden layer units.

A*- algorithm, proposed by Nilson, 1980 (in Doering, Galicki, and Witte, 1997) applies graph theory to find the optimal path that presents the best NN structure. This heuristic method is based on the construction of a graph of network structures and an evaluation value, which is assigned to each of them, and it would not be applied in this experiment. We will only mention that the advantage of this method is that it does not imply any restrictions on generated structure (no limited no. of hidden neurons

Besides dynamic optimizing techniques, there are some static heuristic rules for determining the NN structure, such as Masters' formula. Masters' heuristic formula

hidd) on the basis on

m) and output units ( ):

$$= \sqrt{m \cdot n}$$

when a NN consists of one hidden layer. By this static method the structure of a NN is determined in advance and does not change during the learning process. Formulas for two hidden layered networks are also developed, but will not be used in this experiment since tests using two hidden layered NNs did not give us better result on this data sample.

In our experiment, following optimizing techniques were used:

a) cascading
b) pruning
c) heuristic Masters' formula

Above techniques were tested on each one of five generated data models.

4. Results

4. 1. Test results of NN models for total return rate prediction

Cascading and pruning are used on max. number of 100 000 learning iterations, with testing after each learning phase, according to the parameters noted in section 3. Architectures that use Masters' formula are trained on 100 000 iterations, then tested on the test sample. Test results are shown in tables 2 and 3.

Table 2. Optimizing the structure of backpropagation NN with sigmoid transfer function and delta learning rule (RMS as evaluation measure)

NN model	Cascading		Pruning		Master's formula	
	NN structure	RMS test error	NN structure	RMS test error	NN structure	RMS test error
1	6-15-1	0.0502	6-1-1	0.0501	6-2-1	0.0505
2	5-3-1	0.0504	5-1-1	0.0502	5-2-1	0.0506
3	4-1-1	0.0503	5-1-1	0.0504	4-2-1	0.0381
4	4-20-1	0.0503	4-1-1	0.0504	4-2-1	0.0505
5	4-1-1	0.0506	4-1-1	0.0505	4-2-1	0.0507

Although the above table presents the NNs results on test sample, which is used in structure optimization process, it is possible to remark that structure variations according to data models are obtained only with cascading technique. Pruning suggests only 1 hidden unit in all 5 models, and Masters' formula also gives constant structure of the hidden layer. The obvious best performance is obtained by model 3, using Masters' formula (RMS=0.0381), while other models and techniques perform very similar (RMS is around 0.05). In order to make generalization conclusions, these results will be tested on validation sample later.

Table 3. Optimizing the structure of backpropagation NN with hyperbolic tangent transfer function and normative cumulative delta learning rule (RMS as evaluation measure)

NN model	Cascading		Pruning		Master's formula	
	NN structure	RMS test error	NN structure	RMS test error	NN structure	RMS test error
1	6-19-1	0.1335	6-1-1	0.1340	6-2-1	0.1341
2	5-11-1	0.1347	5-1-1	0.1335	5-2-1	0.0945
3	4-19-1	0.1335	4-1-1	0.1348	4-2-1	0.1339
4	4-8-1	0.1346	4-1-1	0.1336	4-2-1	0.1348
5	4-9-1	0.1350	4-1-1	0.1342	4-2-1	0.1352

Table 3. shows that hyperbolic tangent function and normative cumulative delta learning rule give higher RMS error than previously tested sigmoid transfer function and delta rule. Another observation can be made: no matter which transfer function and learning rule is used, cascading is model sensitive, while other 2 techniques are not. Best model obtained by this NN algorithm is model 2, with structure given by Masters' formula again.

4.2. Validation of selected models

For each prediction model, best optimization method is selected and tested on the validation sample. Results are presented in Table 4.

Table 4. Validation of best selected models for prediction

Model	Optimization method	NN structure	RMS on test sample	RMS on validation sample
1	pruning, sigmoid transfer function, delta learn rule	6-1-1	0.0501	0.0426
2	pruning, sigmoid transfer function, delta learn rule	5-1-1	0.0502	0.0609
3	Master’s rule sigmoid transfer function, delta learn rule	4-2-1	0.0381	0.0427
4	Cascading, sigmoid transfer function, delta learn rule	4-20-1	0.0503	0.0426
5	Pruning, sigmoid transfer function, delta learn rule	4-1-1	0.0505	0.0424

Validation process introduced some changes in the optimization methods performance. While Masters’ formula performed best on test sample (model 3), it gives much worse RMS on out-of-sample data (RMS=0.0427). The overall best performance is obtained by pruning on model 5, where RMS on validation sample was 0.0424, which proves that already tested model (Yoon, Guimaraes, Swales, 1994) performs better than combinations obtained by adding or extracting the variables. It can be seen from the table that pruning and cascading generalize well, except in model 2.

4.3. Sensitivity analysis of the overall best model

In order to observe the change of the output depending the change of the input variables, sensitivity analysis is conducted, with following results:

Table 8. Sensitivity analysis of overall best model

Input variable	Contribution to output
Current Ratio	-3.7775
Return on Equity	+0.3664
Price/Earnings	+0.7983
Price/Sales	+17.8111

Price/Sales is the variable with the significantly strongest influence to the stock return rate (+17.81). The contribution of other variables is much smaller, therefore emphasize the need for investigating additional models. Negative influence of the current ratio is not logical (-3.78), and it can be one of the variables that caused errors in prediction of stock return rate. This sensitivity analysis suggests us a detailed investigation of Price/Sales ratio, since it has so high influence to the output variable (it would be informative to include the variables related to price and sales into the model).

5. Conclusions

Although the conducted analysis represents only the introduction for more complex research about NN performance in financial models, it indicates that pruning algorithm for structure optimization is efficient in most of the models tested. Some other conclusions (limited on this data sample) could be made from above results:

- model that includes all available variables does not perform much worse than other models extracted (the differences among their performance were very small both in test and validation results),
- model that principal component analysis selected shows very good results in test phase, since the validation of the model shows worst performance, which could indicate that models suggested by principal component analysis do not have to necessarily perform good with NNs,
- Masters' rule, which is a static method for determining NN structure, performs good on the models generated by multiple regression and general linear model, but those NNs do not generalize good, i.e. the RMS is lower on validation sample,
- Cascading technique performs good on data with small variance (i.e. when CR, ROE, and P/S are in the model),
- pruning performs good on most of the models, and generalization capability of such models is acceptable (RMS error in validation sample is lower or similar to the RMS in test sample)

6. Guidelines for further research

In this paper we tried to draw the connection between optimizing technique and data model used, as the preliminary research for further more complex analyses. However, it helped us to define some different approaches to this problem, such as how to design the data model, which optimizing techniques to use, how to evaluate the performance, and others. In further research, we would like to determine more characteristics of distribution before designing the models, and to:

- conduct the comparison on more samples, with different distribution and characteristics (such as small companies vs. large companies, S&P index vs. NASDAQ, or Nikkei, etc.)
- use more variables in the models, with different data types,
- test more NN architectures (algorithms other than backpropagation, more transfer functions and learning rules), by varying more parameters,
- look for a less bounded optimization method by testing others than pruning and cascading.

Such complex and time consuming comparisons are needed to extract at least some rules that could be used in the absence of defined paradigms, or could help to define those paradigms that will make the NN methodology less uncertain for various problem domains.

References:

1. Doering, A., Galicki, M., Witte, H., Structure Optimization of Neural Networks with the A*-algorithm, IEEE Transactions on Neural Networks, Vol. 8., No. 6., November 1997, pp. 1434-1445.
2. Hwang, J.N., The Cascade-Correlation Learning: A Projection Pursuit Learning Perspective, IEEE Transactions on Neural Networks, Vol. 7., No. 2., March 1996, pp. 278-289.
3. Kwok, T.Y., Yeung, D.Y., Objective Functions for Training New Hidden Units in Constructive Neural Network, IEEE Transactions on Neural Networks, Vol. 8., No. 5., September, 1997, pp. 1131-1148.
4. Marcek, D., Neural Network Model for Stock Prices Forecasting, Neural Networks World, No. 3, 1997, p.347-352.
5. Masters, T., Just what are we optimizing, anyway?, International Journal of Forecasting, No. 14., 1998., pp. 277-290.
6. Masters, T., Practical NN Recipes in C++, Academic Press, 1993.
7. Parisi, R., Di Claudio, E.D., Orlandi, G., Rao, B., A Generalized Learning Paradigm Exploiting the Structure of Feedforward Neural Networks, IEEE Transactions on Neural Networks, Vol. 7., No. 6., Novemeber 1997, pp. 1450-1459.
8. Quinlan, P.T., Structural Change and Development in Real and Artificial Neural Networks, Neural Networks, No. 11, 1998, pp. 577-599.
9. Refenes, A.N., Burges, A.N., Bentz, Y., Neural Networks in Financial Engineering: A Study in Methodology, IEEE Transactions on Neural Networks, Vol. 8., No. 6., Novemeber 1997, pp. 1222-1267.
10. Shi, S., Xu, L.D., Liu, B., Application of Artificial Neural Networks to the Nonlinear Combination of Forecast, Expert Systems, August, 1996, Vol. 13, No. 3, pp.195-201.
11. Wallrafen, J., Protzel, P., Popp, H., Genetically Optimized Neural Network Classifiers for Bankruptcy Prediction, Proceedings of the 29th Annual Hawaii International Conference on System Sciences, 1996, pp.419-426.
12. Yoon, Y., Guimaraes, T., Swales, G., Integrating Artificial Neural Networks With Rule-Based Expert Systems, Decision Support Systems, vol. 11, 1994, pp. 497-507.
13. Zahedi, F., A Meta-Analysis of Financial Applications of Neural Networks, International Journal of Computational Intelligence and Organizations, Vol. 1, No. 3, 1996, pp. 164-178.
14. Compustat database, McGraw-Hill Co., Inc., 1997.
15. Quinlan, P., Structural Change and Development in Real and Artificial Neural Networks, Neural Networks, No. 11, 1998., pp. 577-599.