Hybrid Neuro-Genetic Systems as Effective Analysis schemes of Financial and Accounting statements

Loukeris Nikolaos[(a)], Matsatsinis Nikolaos[(b)]
Technical University of Crete
Department of Production Engineering and Management
Akrotiri Campus, 73100 Chania, Crete, Greece
(a)PhD student University of Essex, (b)Associate Professor Techn. Univ. of Crete

Abstract:-Financial markets and tax services require continuous information on the economic health of corporations. Data of financial indices and accounting statements include valuable elements that advanced methods of artificial intelligence can implement to provide analyses of high precision. Hybrid systems of neural networks with genetic algorithms optimization are able to support efficiently decisions on portfolio management, corporate management, and financial accounting.

Key-words: Financial Markets, Hybrid Systems, Neural Networks, Genetic Algorithms, Corporate Management

Introduction

Financial evaluation of corporations, detects their economic health enforcing users of financial data, such as investors, financial experts, the government. Corporate financial evaluation and bankruptcy prediction achieved significant results through advanced methods of artificial intelligence, data mining, finance, and operations research, **[1]** whilst new effective tools are always in demand. Data mining the past 20 years consists a new domain of Artificial Intelligence, and it is supported extensively. Data mining seeks to discover valid, hidden information in vast data sets, implementing i) Artificial Neural Networks, ii) Decision Trees, iii) Genetic Algorithms, iv) Methods of Nearest Neighbourhood.

Hybrid Algorithms

Hybrid algorithms compare genetic algorithms with the most successful optimization methods to a problem, ensuring their correct application. Genetic algorithms, despite their vigorous, are not always effective in optimization, **[2]**, **[3]** used non-linear coding, specializing operations of genetic algorithms to combinations with search models based on genetic search models. **[4]** describing a parallel genetic algorithm, concluded that, whilst an initial population is created, each individual each individual elaborates a local ascension and after each descendant is created it activates a local ascension as well. Researchers diverse their conclusions on hybridization matter. Addition of ascensions or hybridization with other optimisation methods, learning is added in the process of evolution. Coding of acquired information, in chromosomes indicates a form of Lamarck search. Optimized chromosomes by local ascension or other methods are put on the total population, and there are allowed to compete aiming to obtain reproduction opportunities.

Neural networks with genetic optimization

NeuroSolutions 4.3 environment provides a complete platform of Neural Networks (NN) that varies from simple Multi Layer Perceptrons-MLP, to very complex hybrid networks with time bias. Software of NeuroSolutiuons 4.3 simulates NN where overall dynamics of networks and training dynamics are divided to local interactive rules. This software follows the principle of

local interaction rules among simple neural components, which is followed on biological neural networks. The object–oriented form of NeuroSolutions specifies operations of elements and their interactions instead of implementing functions strictly, in the ways of ordinary programming offering a powerful environment which simulates non-linear diversification. An NN is described by a set of dynamic equations, and an additional set of dynamic equations for training to satisfy the adjustment ability. There are many different NN architectures with unique characteristics such as Hopfield networks **[5]** that differ significantly in topology from Multi Layer Perceptron–MLP, **[6]** since MLP implement feedforward technique and Hopfields have a recurrent, both use the additive model, **[7]**. Additive model of **[8]**, is described by the dynamics of sets of pairs in the first order non linear differential equations of the form:
$dx_i(t)/dt = G_i(x_i(t), e(t), w, x^*(t))$ (1) where $x_i(t) \epsilon R^n$ are vectors of state, $G_i: R^n \rightarrow R$ a dynamic chart, e(t) external input, w internal system parameters and $x^*(t) \epsilon R^n$ a desired trajectory of system condition. A neural model with distributed set of mapping equations G_i (x) is preferred for the conditional vector of system. In the additive model it is: $dx_i(t)/dt = -\tau_i x_i(t) + \sigma[\Sigma x_{ij} x_j(t)] + e_i(t)$ (2), where τ_i the constant time of i processing element, $\sigma: R \rightarrow R$ the transfer function of input-output, presented in the following figure 1:

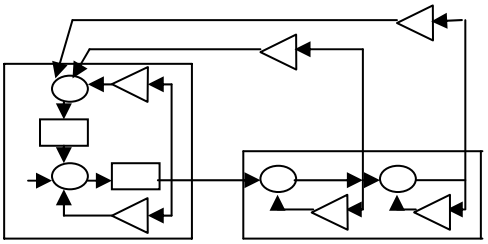

Figure 1. The processing in a neuron
Source: NeuroDimensions Inc.

Activation of node i depend only from present input, a characteristic which is undesired for a significant variety of problems such as: classification of time varied signals (speech, control, prediction). Additive model is altered if multiplications are replaced by convolutions of time as: $dx_i(t)/dt = -\tau_i x_i(t) + \sigma [\Sigma \int w_{ij}(\tau) x_j(\tau-t)d\tau] + e_i(t)$ (3), which is called **[9]** convolutions pattern, allowing activations of neural network to depend from pattern's past conditions and the input signal. Convolution is a linear operation, and neural activations are stored on a generic linear filter, with an undesired increasing number of coefficients. Convolutions model can be converted by constant forms of the first order, **[9]**, as on Gamma Neural Network: $dx_i(t)/dt = -\tau_i x_i(t) + \sigma [\Sigma \int w_{ij}(\tau) x_j(\tau-t)d\tau] + e_i(t)$ (4). A quite big K can approach the convolution pattern according to demands. Each topology is defined by a choice of specific forms on weights matrix. Training dynamics on n-dimensional dynamic system of (4) includes translation equations to network topologies, **[10]**. When the aim is a desired answer, a measure of error can be given by the difference between desired and real output. A typical measure is L2 rule (Mean Square Error) and gradient slope minimizes it, while network coefficients are updated in:

$\vec{w}_{k+1} = \vec{w}_k - \eta \, \partial E/\partial w$ (5) where E is the operational error, and n training proportion. Error is a function of networks production, which is directly connected to networks topology. Desired signal can be a coefficient (stationary point) or time variant (trajectory). Thus there are two potentials: either the network is a feedforward and stationary desired signal with inputs whilst slope calculations are time dependent, or the network is recurrent, or desired signal is time variant and slope calculation is time-dependent. The initial level in object-oriented formulation of NN is the characterization of a set in elements that deploy neural functions and regulate rules in local interaction. The elementary part in NN is the processing element PE, which is an abstraction of biological neuron, **[11]**. A PE receives input signals that are passed by other PEs' and then provides each signal with a weight w_i, accumulating all of them. The next step implements a transfer function, which is usually non-linear to produce the outcome of the PE which will move forth towards other PEs'. Hence a NN is a connected lattice of PEs. Local interaction in PE (neuron) is determined by the Axon category:

$$\vec{y} = f(\vec{x}, \vec{w}) \quad (6)$$

where $x \in R^n$ the input signal, $w \in R^n$ weights for Axon activity, $y \in R^n$ an output signal (mapping). Synapses receive Axons' activity implementing another mapping as a linear weighted accumulation transferring the outcome to other Axon. Thus a Synapse is the connecting element in the lattice of neurons, and it is represented with a labelled arrow in diagrams. Linear mapping which is necessary on the accumulation models during representation of regularized interaction is defined by the synapse category as:

$\vec{y} = f(\vec{x}, \vec{w})$ (7), where $y \in R^n$, $x \in R^n$ are the Axon's activity vectors, $w \in R^{n*m}$ a set of weights for the synapse category, $f: R^{2n} \rightarrow R^m$ a random mapping. All neural networks in NeuroSolutions belong either to Axon's or to Synapse categories.

Static Neural Networks

MLP is a feedforward network, which follows th additive pattern and the equation in discrete time that determines its topology for each neuron is:

$$x_i^l = \begin{cases} e_i & l = 0 \\ n_{l-1} & \\ \sigma\left[\Sigma\, w_{kj}^l x_j^{l-1}\right] & l \neq 0 \end{cases} \quad (8) \text{ for PE on level } l$$

This equation includes two mappings: a linear map between neighbour layers, represented by W matrix, and a non-linear map, represented by non linearity σ(.). These two mappings are between inserted activity, and activity stored in the layer. The form of maps fits exactly to those determined by Axon and Synapse, as: $f(x) = \sigma(x)$ (9) where $\sigma(x) = 1/(1+ e^{-x})$,

$\vec{f}(\vec{x}) = W\vec{x}$ (10), and $W \in R^{m \times n}$ are a fully colonized matrix of weights that provides all elementary dynamics required to create any MLP. Elementary dynamics given by the previous equation is necessary to imitate all topologies inside the additive neural pattern.

Dynamic Neural Networks

A static neural network expands with short term memory mechanisms in vast applications, consisting the general additive pattern which replaces multiplications in a convolution operation. Aiming to apply memory elements Soma is given control of an Axon, adding a third dimension on the spatial lattice figure. Third dimension comes from temporary pairing in time levels. Order of calculations can be faced geometrically on a PEs' lattice according to the rules that the lattice can be in three dimensions with two spatial axes (x and y) lying within the plane of the paper, and one temporal axis going into the page. The present time is at the top of the stack of planes, and each following plane is delayed by one sample. The standard local interaction defined by the Axon class will remain the same with an added restriction that its nonlinear mapping be instantaneous. This can be represented for each processing element as the mapping: $y_i(t) = f(x_i(t), w_i)$ (11) where i index runs over the number of processing elements of the Axon. The Soma class may have temporally coupled the Axon to other PEs in the lattice, but the Axon itself will have no access to them. The Synapse class

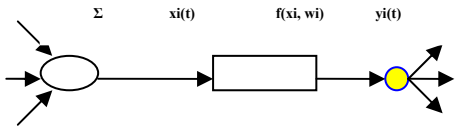

Figure 2. Mapping of Processing Element of Axon class, ND Inc.

will take activity presented by one of the coupled PEs for an Axon, apply its linear mapping and present the result to an Axon in the present temporal plane as, $y(t) = f(x((t-d), w)$ (12), d is a delay that represents which temporal plane the Synapse attaches to. Each Processing Element of the Synapse will produce the mapping $y_i = f(x_j(t-d), w_{ij})$ (13).

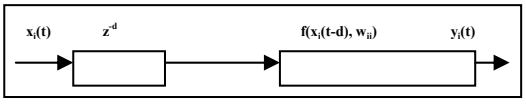

Figure 3. Mapping for PE of Synapse class, Source: NeuroDimensions Inc.

Axon and Synapse are both subclasses of Soma. Since Soma performed temporal coupling of the Axon, it can provide Synapse with access to coupled sites. Temporal coupling performed by Soma is inherent to all network elements and it is hidden from each one of them. Axon and

Synapse have no explicit understanding of time. Axon performs a mapping from a node on the lattice to that same node.

Description of Neural architectures

In this research 10 different architectures of neural networks were used: 1)Principal Component Analysis networks-PCA, 2) Recurrent networks, 3)Time Lag Recurrent Network – TLRN, 4) Support Vector Machine – SVM, 5) Kohonen Shelf Organizing Maps-SOFIM, 6) Jordan Elman networks, 7) Multi Layer Perceptrons – MLP, 8) Generalized Feed Forward – GFF, 9) Modular networks, 10) Radial Basis Function Network – RBFN. Network GRNN/probabilistic PNN did not operate, whilst RBFN was very slow and was rejected.

Data came by 1411 companies from the loan department of a Greek commercial bank, with the following 16 financial indices: 1) EBIT/ Total Assets**,** 2) Net Income/ Net Worth, 3) Sales/ Total Assets, 4) Gross Profit/Total Assets, 5) Net Income/ Working Capital, 6)Net Worth/ Total Liabilities 7)Total Liabilities/ Total assets, 8) Long Term Liabilities/ (Long Term Liabilities + Net Worth), 9)Quick Assets/ Current Liabilities, 10)(Quick Assets-Inventories)/ Current Liabilities, 11)Floating Assets/ Current Liabilities, 12)Current Liabilities/ Net Worth, 13) Cash Flow/ Total Assets, 14)Total Liabilities/ Working Capital, 15)Working Capital/ Total Assets, 16) Inventories/ Quick Assets. A 17th index had initial classification, by bank executives, test set was 50% of overall data, and training set 50% as well.

Results

NeuroSolutions 4.3 was deployed with many different network topologies. Initially Multi Layer Perceptrons – MLP were implemented because of their capacity to resolve linear and non-linear problems, a characteristic that enforces their popularity. Genetic Algorithms were chosen to be used in each intermediate step of solutions that neural networks produced, because they offered the optimal choice of solution genes in neural network, consuming significant time periods. The method of data representation on each genetic session of offspring was on-line, because it provides always the optimal solution in each generation, although sometimes it is exposed to the higher risk of falling in local minima/maxima. Batch representation was rejected because confusion matrix changes on each new chromosome, without adjusting instantly, whilst it has lower risk of getting trapped in local minima/maxima. Radial Basis Function Networks-RBFN, with 1 hidden layer, 500 epochs and genetic optimization was very slow, while Generalized Feed Forward-GFF, networks performed in a very fast convergence. CANFIS networks couldnot be used, as they implemented fuzzy logic, whilst RBFN with 0 hidden layers were extremely fast. RBFN network with 1 hidden layer, 400 epochs and genetic optimization had a slower convergence, compared to Recurrent network. On the other hand RBFN was more reliable in use, while Recurrent collapsed 7 times. Support Vector Machine-SVM networks with 300 epochs produced the confusion matrix fastly from the very beginning. In all neural networks 500 epochs were chosen for each generation of solutions that Genetic Algorithms used to, aiming to give the necessary time for convergence in the optimal offspring set, without wasting surplus time when the optimal set of solutions was found in each repeat. In the following table 1, are presented the optimal results of each different neural network. Neural networks are presented in groups of the same architecture, but with different topologies, where in each topology hidden layers. The 10 different neural network architectures used, were examined thoroughly in different topologies, to evaluate their performance with the same data set of 1411 companies and the training set 50% of overall data. In each architecture of neural networks we implied initially 1 hidden layer and after convergence and results, we increased the number of hidden layers, noticing that while a change in the number of hidden layers the results such as confusion matrix, MSE varied significantly. This is expected since neural networks' performance is a 'black box' to users and neural nets cannot produce the same output, given the same input. We

noticed that as the number of hidden layers was different in the same architecture, results in the confusion matrix were different for each topology in the same architecture. Results provided by two different architectures, Multi Layer Perceptron – MLP and Radial Basis Function Network – RBFN in equivalent confusion matrix in all the topologies, with the apostrophe of non-convergence, since the confusion matrix classified all companies of mark 0 to 0 but also classified all companies of mark 1 to 0, which is a failure to verify initial classifications by bank executives. Analytically the performance of each optimal neural network's architecture, is in the table1.

At first Recurrent neural network produced successful results. Optimal Recurrent network were deployed with 3 hidden layers, with an MSE were very low: 0.116 with 3 layers, providing adequate classification results for companies of mark 1 (in distress) 66.66%in one case (3 layers). Convergence time vas 5 hours 55' (3 layers) and. Although their performance was of adequate convergence, acceptably low in speed, and with very low statistical error (MSE), Recurrent networks could not approach the effectiveness of SVM networks.

Table 1. Overall results in Neural Networks per architecture

Neural Network	Active Confusion Matrix				Performance						Time
	Layers	0->0	0->1	1->0	1->1	MSE	NMSE	r	%error	AIC	MDL	
PCA	1	100	0	33,33	66,66	0,153	0,598	0,664	17102534	95,48	69,78	2 h 37'39
Recurent	3	100	0	33,34	66,66	0,116	0,454	0,771	11092527	940,05	764,36	5 h 25'00
TLRN	4	95,83	4,16	33,33	66,66	0,19	0,739	0,625	41161888	3109,21	2553,98	10 h 58'00
SVM 1000 epoc.		100	0	0	100	0,849	2,672	0,677	444452228	427,58	351,53	4 h 13'00
SOFM	1	100	0	0	100	0,010	0,042	0,979	4880354	1035,8	832,003	5 h 01'00
JordanElman	1	100	0	0	100	0,029	0,113	0,960	6954795,5	12,618	-6,39	2 h 01'00
MLP	1	100	0	100	0	0,375	1,457	0,542	9402632	407,53	331,13	6 h 11'00
GFF	6	100	0	33,33	66,66	0,150	0,583	0,664	1641632	7228,79	5947,21	32 h 33'00
Modular	3	100	0	0	100	0,013	0,054	0,972	9733604	654,63	519,083	12 h 09'00
RBFN	0	100	0	100	0	0,372	1,449	0,58	9400172	135,37	106,85	3 h 11'00

Conclusions-Future Research.

Secondly PCA networks produced very low MSE: 0.153 for 1 layer net. Thirdly GFF networks with 6 hidden layers required large amounts of time periods, while their classifications converged with low accuracy. Optimal TLRN networks with 4 levels provided average level of MSE, but with significant misclassification in confusion matrix. SOFM networks with 1 layer performed in long time periods 5 hours 1 minute, with very low MSE. Modular networks had long times of processing 12 hours 9 minutes, three out of six networks had a relative convergence, whilst their statistical error was moderate. Jordan Elman networks except the best network of 1 layer, all the others lacked to converge, since only one topology out of the rest five had a relative convergence, their MSE was moderate. Finally RBFN networks and MLP failed completely to provide converged classifications, and they demanded significant time periods.

Excellent performance was achieved by Jordan/Elman net with 1 hidden layer. Only Support Vector Machine – SVM networks converged in the same confusion matrix, obtaining acceptable processing time periods to convergence with high MSE. Hybrids of Neural Networks with Genetic Algorithms for optimization on genes solutions produced 5 independent confusion matrices with correct classification at a level 100% that resulted in the same form. Hybrids of Neural Networks with Genetic Algorithms for optimization on genes solutions produced 5 independent confusion matrices with correct classification at a level 100% that resulted in the same form. The most excellent hybrid Neural Network optimized by Genetic Algorithms was Jordan/Elman with 1 hidden layer, table 2, with a very low MSE, the second lower of 43 networks that were deployed in this research, its NMSE was very low at 0.042 and correlation coefficient r was very high

at 0.960, whilst the time to converge was the fastest of all at 2 hours and 1 minute. The second better network was SOFM with 1 hidden layer converged slower in 5 hours and 1' whilst it had the lowest MSE of all: 0.010, and the highest correlation coefficient at 0.979. Another hybrid neural network that had a quick convergence was Modular network with 3 hidden layers concluded its convergence in 2 hours 9 minutes, the MSE was 0.013, the lowest of nets, and correlation coefficient r 0.972. Finally SVM – 1000 epochs that converged in 4 hours 13 minutes with a very high cost function at MSE of 0.849 and the lowest correlation r at 0.677.

Table 2. Networks with excellent performance

Network	Layers	epochs	MSE	NMSE	r	Time
Jordan Elman	1	500	0.029	0.113	0.96	2 h 01'
SVM		1000	0.849	2.672	0.677	4 h 13'
SOFM	1	500	0.042	0.042	0.979	5 h 01'
Modular	3	500	0.013	0.054	0.972	12 h 09'

Bibliography

[1] Loukeris N., Matsatsinis N., "Corporate Financial Evaluation and Bankruptcy Prediction implementing Artificial Intelligence methods", Journal of Business and Economics, Issue 4, Volume 3, April 2006, WSEAS Transactions

[2] Davis. L. "Handbook of Genetic Algorithms"- Van Nostrand Reinhold, New York, 1991

[3] Michalewitz Z. "Genetic algorithms + data structures = Evolution programs", Springer Verlag, 1992

[4] Mühlenbein H. (1991)-Evolution in Time and Space. The Parallel Genetic Algorithm- Foundtions of Genetic Algorithms, G. Raawline ed. Morgan-Kaufmann pp 316-337

[5] Hopfield J. "Neural networks and physical systems with emergent collective computational abilities." Proc. Natl. Acad. Sci. (USA) 79, 2554-2558, 1982.

[6] Lippman R. "An introduction to computing with neural nets." IEEE Trans. ASSP Magazine 4, 4-22, 1987.

[7] Grossberg S. and Cohen M. "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks." IEEE Trans. on Syst. Man Cybern. SMC-13, 815-826, 1983.

[8] Amari S. "Characteristics of random nets of analog neuron-like elements." IEEE Trans. Syst. Man Cybern. SMC-2,5, 643-657, 1972

[9] deVries B. and Principe J. "The gamma model - A new neural model for temporal processing." Neural Networks 5(4), 565-576, 1992.

[10] Thrun S. and Smieja F. "A general feedforward algorithm for gradient descent learning in connectionist networks." Int. Rep. German National Res. Centre Comp. Sci., 1991.

[11] McCulloch W. and Pitts W. "A logical calculus of the ideas imminent in the nervous activity." Bulletin of Mathematical Biophysics 5, 115-133, 1943