Adrian Costea | Iulian Nastac

Three factors affecting the predictive performance of ANNs: pre-processing method, data distribution and training mechanism

Turku Centre *for* Computer Science

TUCS Technical Report
No 679, April 2005

Three factors affecting the predictive performance of ANNs: pre-processing method, data distribution and training mechanism

Adrian Costea
Turku Centre for Computer Science, Åbo Akademi University, Department of Information Systems
Lemminkäisenkatu 14 A, 20520 Turku, Finland
acostea@abo.fi
Iulian Nastac
Turku Centre for Computer Science
Lemminkäisenkatu 14 A, 20520 Turku, Finland
inastac@abo.fi

TUCS Technical Report
No 679, April 2005

Abstract

In this paper we analyze the implications of three different factors (preprocessing method, data distribution and training mechanism) on the classification performance of artificial neural networks (ANN). We use three preprocessing approaches: no preprocessing, normalization and division by the maximum absolute values. We study the implications of input data distributions by using five datasets with different distributions: the real data, uniform, normal, logistic and Laplace distributions. We test two training mechanisms: one belonging to the gradient-descent techniques, improved by a retraining procedure (RT), and the other is a genetic algorithm (GA), which is based on the principles of natural evolution. The results show statistically significant influences of all individual and combined factors on both training and testing performances. A major difference with other related studies is the fact that for both training mechanisms we train the network using as starting solution the one obtained when constructing the network architecture. In other words we use a hybrid approach by refining a previously obtained solution. We found that when the starting solution has relatively low accuracy rates (80-90%) GA clearly outperformed the retraining procedure, while the difference was smaller to zero when the starting solution had relatively high accuracy rates (95-98%). As reported in other studies we found little to no evidence of crossover operator influence on the GA performance.

Keywords: Artificial Neural Networks, Genetic Algorithms, preprocessing method, data distribution, training mechanism

TUCS Laboratory
Data Mining and Knowledge Management Laboratory

1 Introduction

Predictive data mining has two different aims: (1) the uncovering of hidden relationships and patterns in the data, and (2) the construction of usable prediction models (Zupan *et al.* 2001). One type of prediction model is represented by classification models or models for predicting the relative positions of newly observed cases against what is known. Financial performance classification problems concern many business players: from investors to decision makers, from creditors to auditors. They are all interested in the financial performance of the company, what are its strengths and weaknesses, and how the decision process can be influenced so that poor financial performance or, worse, bankruptcy, is avoided. Usually, the classification problem literature emphasizes binary classification, also known as two-group discriminant analysis problems, which is a simpler case of the classification problem. In the case of binary classifications everything is seen in black and white (e. g.: A model which implements a binary classifier would just show a bankruptcy or a non-bankruptcy situation giving no detailed information about the real problems of the company). Greater information would be obtained from the classification models if one particular business sector would be divided in more than two financial performance classes (and it would be easier to analyze the companies placed in these classes). This study introduces an artificial neural network trained using genetic algorithms (GA-based ANN) to help solve the multi-class classification problem. The predictive performance of the GA-based ANN will be compared to a retraining-based ANN which is a new way of training an ANN based on its past training experience and weights reduction. The two different training mechanisms have something in common, which is the ANN architecture. A new empirical method to determine the proper ANN architecture is introduced. Moreover, the study investigates the influence of data distributions and preprocessing approach on the predictive performances of the models.

The paper is organized as follows. In the second section we review the literature on classification models emphasizing ANN-based models for classification. Next, we introduce our model for assessing companies' financial performance. Research questions and derived hypotheses are formulated in section four. The datasets used are presented in section five. In the sixth section, we show our experiments' results, and finally, the conclusions and directions for future research are discussed.

2 Literature review

The problem of financial performance classification has been tackled in the literature for nearly 40 years. The taxonomy of classification models is based on the algorithm solution being used (Pendharkar, 2002). Firstly, **statistical techniques** have been deployed: univariate statistics for prediction of failures introduced by Beaver (1966), multivariate analysis in Altman (1968), linear discriminant analysis (LDA) introduced by Fisher (1936) who firstly applied it on Anderson's iris data set (Anderson, 1935), multivariate discriminant analysis (MDA) - Edmister (1972), Jones (1987), probit and logit models - Hamer (1983), Zavgren (1985),

Rudolpher *et al.* (1999) and recursive partitioning algorithm (RPA) in Frydman *et al.* (1985). The next step in solving the classification problem was the establishment of **induction techniques**. Some of the most popular such techniques are: CART (Breiman *et al.*, 1984), ID3-C4.5-C5.0 (Quinlan, 1993a; Quinlan, 1993b). In (Costea *et al*., 2002; Costea and Eklund, 2003) the authors applied and compared two of the above classifiers: multinomial logistic regression and Quinlan's C5.0 decision tree. The two classifiers performed similarly in terms of accuracy rates and outperformed SOM (Kohonen, 1997) classification. Among the financial application areas of **neural networks** in the early 80s, the financial performance classification problem was not an exception. ANNs were extensively used in financial applications, the emphasis being on bankruptcy prediction. A comprehensive study on ANNs for failure prediction can be found in O'Leary (1998). The author investigates fifteen related papers for a number of characteristics: what data was used, what types of ANN models, what software, what kind of network architecture, etc. Table 1 presents a sample of studies with their results which compared different classification techniques.

Table 1: Sample of pattern classification studies

Authors	Tasks	Techniques	Results
Marais *et al.* (1984)	Modelling commercial bank loan classifications	Probit, RPA	RPA is not significantly better, especially when data do not include nominal variables
Schtze *et al.* (1995)	Document routing problem	Relevance feedback, LDA, Logistic regression, ANN	Complex learning algorithms (LDA, logistic regression, ANN) outperformed weak learning algorithm (relevance feedback)
Jeng *et al.* (1997)	Prediction of bankruptcy, biomedical	Fuzzy Inductive Learning Algorithm (FILM), ID3, LDA	Induction systems achieve better results than LDA. FILM slightly outperforms ID3
Back *et al.* (1996, 1997)	Prediction of bankruptcy	LDA, Logit, ANN	ANN outperformed the other 2 methods in terms of accuracy

Koskivaara (2004) summarizes the ANN literature relevant to auditing problems. She concludes that the main auditing application areas of ANNs are as follows: material error, going concern, financial distress, control risk assessment, management fraud, and audit fee which are all, in our opinion, particular cases of classification problems. In other words, in these applications ANNs were used, mainly, as classifiers. Going concern and financial distress can be considered to be particular cases of bankruptcy prediction.

Costea and Eklund (2004) compared three classifiers for financial performance classification of telecom companies and found that the ANN performed similarly in terms of accuracy rates against statistical and induction techniques.

Coakley and Brown (2000) classified ANN applications in finance by the parametric model used, the output type of the model and the research questions.

Another technique to learn the connection weights for an ANN corresponds to the **evolutionary approach** and is represented by genetic algorithms. The literature in this area is relatively rich: Schaffer *et al.* (1992) listed 250 references that combined ANNs and genetic algorithms. GAs are used in the majority of these papers for solving the following problems: to find the proper architecture for the ANN, reduce the input space to the relevant variables, and as an alternative way of learning the connection weights. One paper that uses GAs to solve the last two forementioned problems is Sexton and Sikander (2001). The GA was found to be an appropriate alternative to gradient-descent-like algorithms for training neural networks and, at the same time, the GA could identify relevant input variables in the data set.

Many authors (e.g. Schaffer, 1994) found that GA-based ANNs are not as competitive as their gradient-descent-like counterparts. Sexton *et al.* (1998) argued that this difference has nothing to do with the GA's ability to perform the task, but rather with the way it is implemented. The candidate solutions (the ANN weights) were encoded as binary strings which is both unnecessary and unbeneficial (Davis, 1991 and Michalewicz, 1992) when the ANN has a complex structure. The tendency is toward using non-binary (real) values for encoding the weights. Pendharkar (2002) studied the application of a non-binary GA for learning the connection weights of an ANN under various structural design and data distributions, finding that additive noise, size and data distribution characteristics play an important role in the learning, reability and predictive ability of ANNs.

In this study we compare two different ANN training mechanisms for pattern classification: one based on the traditional gradient-descent training algorithms (RT-based ANN) and another one based on natural selection and evolution (GA-based ANN). We also propose an empirical procedure to determine the ANN architecture which is kept fix for both training mechanisms. We reveal classes of financial performance for the companies in the telecommunication sector based on profitability, liquidity, solvency and efficiency financial ratios. These ratios are suggested in Lehtinen's (1996) study of the reliability and validity of financial ratios in international comparisons.

3 Financial performance classification models

In this section we present our two approaches for financial performance classifications. Firstly, we describe the empirical procedure for determining the ANN architecture. Then, we present the two ANN training mechanisms. The generic classification model based on neural approaches is depicted in Figure 1.

Usually, when constructing classification models, the first step is to separate the data into training (*TR*) and test (*TS*) sets. We are concerned with decisions

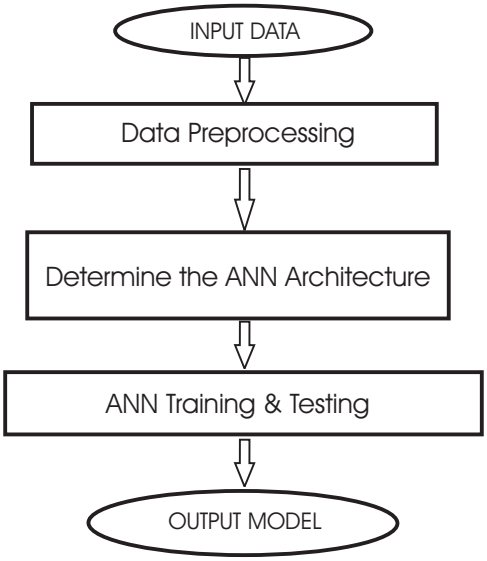

Figure 1: ANN generic classification model

regarding preprocessing of the data (what method of standardization, what method of removing the outliers, if any). In case the class variable is missing, as in our case, a clustering method could be applied to build this variable. The second step consists of selecting the proper ANN architecture. This step is concerned with determining the proper number of hidden layers, and the right number of neurons in each hidden layer. Also, here we decide how the class variable should be codified. In other words, how many neurons are necessary on the output layer to represent the class variable? The last step, ANN training, consists of specific tasks depending on the training mechanism used.

3.1 Data Preprocessing

Next, we present the steps undertaken to create the training and test sets for the classification models, which we generically call data preprocessing steps:

1. In order to ease the training and to avoid the algorithms placing too much emphasis on extreme values, we removed far outliers and outliers from the data. This has been done by calculating the quartiles for each variable. If we denote with *l* the lower quartile, with *m* the median and with *u* the upper quartile of variable *x*, then the far outliers (*fo*), outliers (*o*) and anomalies (*a*) for that variable belong to the following intervals:

$$
\begin{aligned}
&fo \in (-\infty, l - 3d) \cup (u + 3d, +\infty) \\
&o \in [l - 3d, l - 1.5d) \cup (u + 1.5d, u + 3d] \\
&a \in [l - 1.5d, l - d) \cup (u + d, u + 1.5d]
\end{aligned}
$$

   where *d* = *u* - *l* is the distance from the upper quartile to the lower. Once we have detected the far outliers and the outliers of each variable we have two

alternatives: to discard a sample that has at least one outlier value or to keep it by taking the peak(s) off. We chose the later alternative.

2. Regarding standardization three approaches were undertaken: one was to keep the data un-standardized ("no preprocessing"), the second was to normalize data to zero mean and unit standard deviation ("normalization") and the third was to divide the data by the maximum of absolute values ("maximum of absolute values").

3. A clustering technique was applied to build the class variable. We have used the fuzzy C-means clustering algorithm (Bezdek, 1984) to build the clusters and, consequently, the class variable. The number of clusters is a parameter of our models. It was set to 7 as this was the proper number of classes reported in our previous studies (Costea and Eklund, 2003).

4. In order to allow the ANN to equally learn the patterns within each cluster we chose an equal number of observations for each cluster.

5. Finally, we split the data into aproximately 90% for training and the remainder for testing.

As was described above, we reduce as much as possible the subjectivity in determining the class variable by applying directly FCM clustering algorithm. (Alcaraz and Costea, 2004) compared a modified version of FCM algorithm with normal FCM and SOM clustering. The modified FCM algorithm outperformed both the normal FCM and the SOM with respect to pattern classification. In this study, normal FCM was chosen for practical implementation reasons. We created for each financial ratio a linguistic variable that can help us in characterizing the clusters. Linguistic variables are quantitative fuzzy variables whose states are fuzzy numbers that represent linguistic terms (Klir and Yuan, 1995). Alcaraz and Costea (2004) model the seven financial ratios with the help of seven linguistic variables using five linguistic terms: *very low* (VL), *low* (L), *average* (A), *high* (H), *very high* (VH). Table 2 shows the characterization of the seven clusters for the real telecom dataset without preprocessing the data (first preprocessing approach).

We considered that one linguistic term characterizes one cluster if it represents more than 40 % out of total number of observations for that cluster. It seems that Receivables Turnover does not have any discriminatory power among data except for one cluster. By comparing the clusters we can easily label them as being good, bad, worst, etc. depending on their linguistic terms.

3.2 Empirical procedure for determining the ANN architecture

Once the data is ready to be trained, we need to perform one more preliminary step: to find a suitable architecture for the ANN. Choosing the number of hidden layers and the number of neurons in each hidden layer is not a straight-forward task. The choices of these numbers depend on input-output function complexity (Nastac and Matei, 2003). It is well known that neural networks are very sensitive regarding the dimensionality of the dataset. In general, a good model is obtained when we

Table 2: Characterization of Clusters[a]

	OM	ROTA	ROE	Current	E to C	IC	Rec. T.	Order
Cluster 1	VL	VL	VL&L	-	A&H	VL&L	-	Bad
Cluster 2	A	A	A&H	-	A	A	-	Average
Cluster 3	VL&L	VL	VL	-	VL&L	L	-	Worst
Cluster 4	H	H	VH	VL	A	A	A	Good
Cluster 5	A	A	A&H	H	H	VH	-	Good
Cluster 6	L	L	A	L	L	L	-	Bad
Cluster 7	VH	VH	H	VH	VH	VH	-	Best

[a] Alcaraz and Costea (2004)

have 10 times more training samples than the number of weights. We performed a number of experiments for ANN architectures with one and two hidden layers to see what the appropriate number of hidden layers is. Almost in every case, an ANN with two hidden layers performed better in terms of training mean square error. We did not take into consideration three hidden layer cases due to the number of cases per weights ratio-restriction.

We used the sigmoid and linear activation functions for the hidden and output layers, respectively. Regarding the training algorithms, they fall into two main categories: heuristic techniques (momentum, variable learning rate) and numerical optimization techniques (conjugate gradient, Levenberg-Marquardt). Various comparative studies, on different problems, were initiated in order to establish the optimal algorithm (Demuth and Beale, 2001; Nastac and Koskivaara, 2003). As a general conclusion, it is difficult to know which training algorithm will provide the best (fastest) result for a given problem. A smart choice depends on many parameters of the ANN involved, the data set, the error goal, and whether the network is being used for pattern recognition (classification) or function approximation. Statistically speaking, it seems that numerical optimization techniques present numerous advantages. Analyzing the algorithms that fall into this class, we observed that the Scaled Conjugate Gradient (SCG) algorithm (Moller, 1993) performs well over a wide variety of problems, including the experimental dataset presented in this paper. Even if SCG is not the fastest algorithm (as Levenberg-Marquardt in some situations), the great advantage is that this technique works very efficiently for networks with a large number of weights. The SCG is something of a compromise: it does not require large computational memory, and yet, it still has a good convergence and is very robust. Furthermore, we always apply the early stopping method (*validation stop*) during the training process, in order to avoid the overfitting phenomenon. And it is well known that for early stopping, one must be careful not to use an algorithm that converges too rapidly. The SCG is well suited for the validation stop method.

In our experiments we have kept all parameters of the ANN constant (the learning algorithm - Scale Conjugate Gradient, the performance goal of the classifier, the maximum number of epochs), except the numbers of neurons in the hidden layers (NH_1, NH_2).

The procedure used to determine the proper values for NH_1 and NH_2 consists of iteratively performing the following experiment:

- Randomly split the training set (*TR*) into two parts: one for the effective training (*TRe*) and the other for validation (*VAL*). In order to avoid the over-fitting phenomenon we have applied the early stopping method (*validation stop*) during the training process.

- Train the network for different values of NH_1 and NH_2. For each combination of NH_1 and NH_2, we performed 4 random initializations of the weights. NH_1 and NH_2 take values in the vicinity of the geometric mean (Basheer and Hajmeer, 2000) of the number of inputs (NI) and outputs (NO), respectively.

$$\sqrt{NI \cdot NO} - 2 \leq NH_i \leq \sqrt{NI \cdot NO} + 2$$

  E.g.: $NI = 7, NO = 7 \Rightarrow NH_1, NH_2 = \overline{5,9}$. In this case, in total 5*5*4 = 100 trainings are performed for each experiment.

- Save the best ANN architecture in terms of mean square error of the training set (MSE_{TRe}) with the supplementary condition: $MSE_{VAL} \leq (6/5) * MSE_{TRe}$. This supplementary condition was imposed so that the validation error is not too far from the training error, thus, avoiding over-fitting for the test set.

We ran 3 experiments like the one described above (3*100 = 300 trainings) to determine the proper values for NH_1 and NH_2. See the flowchart of the procedure in the Appendix.

Regarding the number of output neurons, we have two alternatives when applying ANNs for pattern classification. The first alternative, which is the most commonly used, is to have as many output neurons as the number of classes. The second alternative is to have just one neuron in the output layer, which will take the different classes as values. We chose the first approach in order to allow the network to better disseminate the input space.

After we performed the 3 experiments we obtained the best ANN architecture and the set of final weights (the solution) that corresponds to this architecture. In the next two sections we will present two training mechanisms used to *refine* this solution.

3.3 RT-based ANN training

Once we determine the ANN architecture (with the corresponding set of weights), the next step is to train the network. The first training mechanism is a retraining-based ANN (Nastac and Costea, 2004), briefly described next:

- Start with a network with an initial set of weights from the previous step (Determining ANN architecture) as the reference network;
- Run a number of L experiments to improve the ANN classification accuracy. After each experiment we save the best set of weights (the solution) in terms of classification accuracy. Each experiment consists of:
  - Reduction of the weights of the current best network with successive values of scaling factor γ (γ = $0.1, 0.2, \ldots, 0.9$).
    - Retrain the ANN with the new weights and obtain 9 accuracy rates.
  - Choose the best network from the above 9 experiments in terms of classification accuracy.
  - Compare the accuracy rate of the current network with that obtained in the previous step and save the best one for the next experiment as current the best network.

Depending on the splitting of the training set (*TR*) in the effective training set (*TRe*) and validation set (*VAL*) we have 3 types of retraining mechanisms: one where *TRe* and *VAL* are common for all the experiments, another where *TRe* and *VAL* are different for each experiment, but the same for all 9 reduction weights trainings (second step of the experiment), and finally, where *TRe* and *VAL* are distinct for each training. We have 4 types of accuracy rates: training accuracy rate (ACR_{TRe}), validation accuracy rate (ACR_{VAL}), total training (effective training + validation) accuracy rate (ACR_{TR}) and test accuracy rate (ACR_{TS}). Correspondingly, we calculate 4 mean square errors: MSE_{TRe}, MSE_{VAL}, MSE_{TR}, and MSE_{TS}. In total 5 experiments were conducted resulting in 5*9 = 45 new trainings for each type of retraining mechanism.

3.4 GA-based ANN training

The second ANN training mechanism used to *refine* the solution is based on the principle of natural evolution. A population of solutions is provided, and by initialization, selection and reproduction mechanisms, near-optimal solutions are reached.

Unlike the traditional gradient-descent training mechanisms, GA-based ANN training starts with a population of solutions. A solution is the set of ANN weights after training represented as a vector. All solutions (chromosomes) compete with each other to enter the new population. They are evaluated based on the objective function.

3.4.1 Initialization and fitness evaluation

The population size is set to $PS = 20$. Several authors suggested that this value is good enough for any grade of problem complexity (Dorsey and Mayer, 1995). The first chromosome of the population is the set of weights obtained when determining the ANN architecture. The other 19 chromosomes are generated by training the ANN with the previously obtained architecture. Afterwards, the first generation of the algorithm may begin. The number of generations is related with the empirical

formula suggested in Ankenbrandt (1991). The number of generations for a non-binary GA without mutation is given by the formula: $N_{gen} = \ln[(n-1)^2]/\ln(r)$ where n is the population size and r is the average fitness of candidates with a particular gene value over the average fitness of all other candidates.

Each chromosome is evaluated using the accuracy rate for the training set (ACR_{TR}).

3.4.2 Selection

Firstly, the elitism technique is applied in the sense that the best N_{elite} chromosomes in terms of ACR_{TR} are inserted into the new population. The rest of the chromosomes ($20 - N_{elite}$) are selected based on the probability of selection (*roulette wheel* procedure) for each chromosome: $P_i = ACR_{TR_i} / \sum_{i=1}^{20} ACR_{TR_i}$

The higher the probability P_i for a chromosome is, the higher its chance of being drawn into the new population. This procedure tries to simulate the process of natural selection or survival of the fittest.

Next, 80% (probability of crossover: $P_c = 0.8$) of the chromosomes obtained previously are randomly selected for mating.

3.4.3 Reproduction

The selected chromosomes are randomly paired and recombined to produce new solutions. There are two reproduction operators: *crossover* and *mutation*. With the first the mates are recombined and new born solutions inherit information from both parents. With the second operator new parts of the search space are explored and, consequently, we expect that new information is introduced into the population. In this study we have applied four types of crossover: arithmetic, one-point, multi-point and uniform crossover. Let us denote with L the length of the chromosomes and with P_1 and P_2 two parent-chromosomes:

$$P_1 = g_{11}, g_{12}, \ldots, g_{1L}$$
$$P_2 = g_{21}, g_{22}, \ldots, g_{2L}$$

One-point crossover
For each pair of chromosomes we generate a random integer $X, X \in \{1, L\}$. The two new born children are constructed as follows:

$$C_1 = g_{11}, g_{12}, \ldots, g_{1X}, g_{2,X+1}, \ldots, g_{2L}$$
$$C_2 = g_{21}, g_{22}, \ldots, g_{2X}, g_{1,X+1}, \ldots, g_{1L}$$

Multi-point crossover
We split the chromosomes in n parts ($n \leq 5$). We generate randomly the number of splitting points n. Then, n distinct random numbers ($X_1, X_2, \ldots, X_n$) are generated with $X_i \in \{1, L\}$ and $X_1 < X_2 < \ldots < X_n$. The two children are:

$$C_1 = g_{11}, g_{12}, \ldots, g_{1X_1}, g_{2,X_1+1}, \ldots, g_{2X_2}, g_{1,X_2+1}, \ldots, g_{1X_3}, g_{2,X_3+1}, \cdots$$
$$C_2 = g_{21}, g_{22}, \ldots, g_{2X_1}, g_{1,X_1+1}, \ldots, g_{1X_2}, g_{2,x_2+1}, \ldots, g_{2X_3}, g_{1,x_3+1}, \cdots$$

Arithmetic crossover
Firstly, we split the parent-chromosomes in n parts as we did for multi-point crossover. The children' genes are convex combinations of the parents' genes.

$$C_1 = \begin{cases} \alpha * g_{1i} + (1-\alpha) * g_{2i}, & i = \overline{1, X_1} \\ (1-\alpha) * g_{1i} + \alpha * g_{2i}, & i = \overline{X_1+1, X_2} \\ \alpha * g_{1i} + (1-\alpha) * g_{2i}, & i = \overline{X_2+1, X_3} \\ \dots \end{cases}$$

$$C_2 = \begin{cases} (1-\alpha) * g_{1i} + \alpha * g_{2i}, & i = \overline{1, X_1} \\ \alpha * g_{1i} + (1-\alpha) * g_{2i}, & i = \overline{X_1+1, X_2} \\ (1-\alpha) * g_{1i} + \alpha * g_{2i}, & i = \overline{X_2+1, X_3} \\ \dots \end{cases}$$

where $\alpha \in [0, 1]$ is a random number and is generated for each chromosome-pair.

Uniform crossover
For each pair of genes of the parent-chromosomes we generate a random number $\alpha \in [0, 1]$. If $\alpha < 0.5$ the gene of the first parent goes to the first child and the gene of the second parent goes to the second child. Otherwise, the genes are inversed.

The children-chromosomes are *added* to the population. The size of the population becomes $PS' > PS$. Next we apply the mutation operator for all the chromosomes in PS'. We used only uniform mutation.

Uniform mutation
The probability of mutation is set to $P_m = 0.01$ which means that approximately 1% of the genes will mutate for each chromosome. An $\alpha \in [0, 1]$ is generated for each gene of each chromosome and if $\alpha \leq P_m$, the new gene is randomly generated within the variable domain. Otherwise, the gene remains the same. If at least one gene is changed then the new chromosome is added to the population, obtaining $PS'' > PS' > PS$.

The final step in constructing the new population is to reduce it in size to 20 chromosomes. We select from PS'' the best 20 chromosomes in terms of ACR_{TR} satisfying the condition that one chromosome can have no more than max_lim duplicates. We use the mutation operator to generate more chromosomes in the case that the number of best chromosomes which satisfy the above condition is less than 20.

As a summary, excluding the crossover, the parameters of our GA models are as follows: number of generations (N_{gen}), number of elite chromosomes (N_{elite}), maximum number of splitting points (max_split), probability of crossover (P_c), probability of mutation (P_m), and maximum number of duplicates for the chromosomes (max_lim).

4 Research questions and derived hypotheses

The main advantages of neural approaches for classification over the traditional ones are: ANNs are free of any distributional assumptions, are universal approx-

imators, no problems with intercorrelated data, and they provide a mapping function from the input to the outputs without any a priori knowledge about the function form (function approximation capability). The most popular ANN learning technique in the literature is back-propagation (BP), which is "an approximate steepest descendent algorithm"(Hagan *et al.*, 1996) for feedforward neural networks. BP has several limitations, the most important one being its scalability: as the size of the training problem increases, the training time increases non-linearly (Pendharkar and Roger, 2004). When the basic BP is applied to a practical problem, the training may take a relatively long time (Hagan *et al.*, 1996). Among other limitations: the difficulty of the training data itself, handling the outliers, and reduced power of generalization due to large solution space. The cause for the last limitation could be the fact that the BP algorithm is likely to quickly get stuck in a local optimum, which means that the algorithm depends strongly on the initial starting values. As we described in section 3.2 many techniques have been proposed to decrease the learning time of BP and to ignore shallow local minimum. SCG was used for ANN training throughout this study.

The difference between BP/BP-variants and GA-based ANN training techniques is that BPs start from one solution and try to improve it based on some error minimization technique, while GAs start with a population of solutions and through some initialization, reproduction and recombination methods tries to reach an optimal or near-optimal (heuristic) solution. GAs are known as hill climbing techniques, a capability that arises from the convex combination (*arithmetic crossover operator*) of two parents on the opposite sides of a hill. Moreover, the possible risk of reaching a local optimum is avoided by the GA since it creates new solutions by altering some elements of the existing ones (*mutation operator*), hence, widening the search space.

In this study we analyze the implications of three different factors (preprocessing method, data distribution and training mechanism) and their combinations on the classification performance of neural networks. We use three preprocessing approaches: no preprocessing, normalization and division with the maximum absolute values. We study the implications of input data distributions by using five datasets with different distributions: the real data, uniform, normal, logistic and Laplace distributions. We test two training mechanisms: one based on a traditional gradient-descent technique improved by a retraining procedure (RT), and the other on genetic algorithms (GA). Moreover, we analyze the influence of the crossover operator on the predictive performance of genetic algorithms.

We compared our research questions with what was previously reported in the literature (e.g.: Pendharkar and Rodger, 2004). However there are some important differences in the assumptions in our study compared with the others:

- The main difference is that here GA and gradient descent methods are used to *refine* the classification accuracy of an already obtained ANN-based solution for the classification problem. Both the GA and the RT-based ANNs start from a solution provided when determining the ANN architecture and they try to *refine* it. All other studies compared GA and gradient-descent methods starting from random solutions. We expect that the GA-based ANN will

outperform the RT-based ANN in refining what the ANN already learned due to the GA's better searching capabilities.

- The second main difference is the type of the classification problem itself. Here we are interested in separating the input space into more than 2 parts (e.g. 7 financial performance classes) providing more insights in the data.
- We are interested if the preprocessing approach and the distribution of the data have any impact on the classifiers' predictive performances.
- Here non-parametric statistical tests are used to validate the hypotheses. Only t-tests or ANOVA were used in the other studies, but no evidence of satisfaction of the assumptions was provided. We performed a 3-way ANOVA to strengthen the results of the non-parametric tests.
- Also four different crossover operators are used in order to find whether this operator has an influence on the GA's predictive performance. We introduce one crossover operator - multi-point crossover - that was not found in other similar studies.

The first difference has an impact on all the hypotheses that we formulate in this study since, here, there is a different problem. The GA and RT-based ANNs improve an already existing solution and do not construct it from scratch. Their behavior depends on how that solution was obtained (using what kind of method).

The main hypothesis of our paper is formulated as follows:

H1. The GA-based ANN will outperform the RT-based ANN both in training and testing on refining the solution obtained when determining the ANN architecture.

Additional hypotheses:

H2. The Crossover operator will have an influence on GA-based ANN training and testing performances.

H3. Data preprocessing will have an influence on both RT and GA-based ANN training and testing performances.

H4. Data distribution will have an influence on both RT and GA-based ANN training and testing performances.

5 Datasets

Telecommunications sector dataset. We used financial data about worldwide telecom companies. There are 88 companies structured in five groups: US (32), Europe except Scandinavian companies (20), Asia (20), Scandinavia (10), and Canada (6). The time span is 1995-2001. For each company and for each year seven financial ratios were collected with the Internet as the primary source. These ratios are suggested in Lehtinen's (1996) study of financial ratios' reliability and validity in international comparisons. The ratios measure four different aspects of companies' financial performance: profitability - 3 ratios (operating margin, return on total assets, and return on equity), liquidity - 1 ratio (current ratio = current assets / current liabilities), solvency - 2 ratios (equity to capital, interest coverage),

and efficiency - 1 ratio (receivables turnover) (Karlsson, 2002). In total the dataset consists of 651 rows taken from companies' financial statements in their annual reports: 88 companies * 7 years = 616 rows. 35 more rows were obtained with the averages for the five groups (5 groups * 7 years = 35 rows). Out of 651 rows 21 were discarded due to lack of data for calculating some ratios resulting in a final dataset of 630 rows.

Fictive datasets. In order to test the impact of data distribution on the predictive performances of the classifiers we generated four datasets with different distributions: uniform, normal, logistic and Laplace distributions. We estimated the distributions' parameters using the means and variances of the telecom dataset ratios.

6 Experiments

In all our experiments we applied the following methodological steps:

1. For the RT-based ANN we repeated the procedure (described in subsection 3.3) 30 times, obtaining 4 vectors (30 elements in size) of different accuracy rates for each retraining mechanism type: a vector of effective training accuracy rates ($RT\_VEC\_ACR_{TRe}$), a vector of validation accuracy rates ($RT\_VEC\_ACR_{VAL}$), a vector of total training (effective training + validation) accuracy rate ($RT\_VEC\_ACR_{TR}$) and a vector of test accuracy rates ($RT\_VEC\_ACR_{TS}$). Correspondingly, we obtained 4 vectors with the mean square errors: $RT\_VEC\_MSE_{TRe}$, $RT\_VEC\_MSE_{VAL}$, $RT\_VEC\_MSE_{TR}$, and $RT\_VEC\_MSE_{TS}$.

2. For the GA-based ANN we applied the procedure (described in subsection 3.4) 10 times for each type of crossover (one-point - GAO, multi-point - GAM, arithmetic - GAA, and uniform - GAU). The other GA parameters used were as follows: $N_{gen} = 1000$, $N_{elite} = 3$, $max\_split = 5$, $P_c = 0.8$, $P_m = 0.01$ and $max\_lim = 1$. We obtained 2 vectors (10 elements in size) for each type of crossover operator: a vector of training accuracy rates ($GA\_VEC\_ACR_{TR}$) and a vector of test accuracy rates ($GA\_VEC\_ACR_{TS}$) and, correspondingly, 2 vectors with mean square errors: $GA\_VEC\_MSE_{TR}$, and $GA\_VEC\_MSE_{TS}$.

3. We used statistical tests to compare the vectors of the two training mechanisms in order to validate our hypotheses.

The following experiments differ in two perspectives: the hypothesis that they try to validate and/or the type of statistical test used (non-parametric vs. parametric).

Experiment 1. In the first experiment we try to validate the first hypothesis using non-parametric tests (Siegel and Castellan, 1988). We used the real dataset (the original telecom data) without preprocessing the data (first preprocessing approach). After we separated the data in training (90%) and test (10%) sets, we generated the ANN architecture. Then, in order to refine our solution, we applied the two training mechanisms (RT-based ANN and GA-based ANN). We applied

the methodological steps described above and we compared statistically the results' vectors of both training mechanisms in order to validate our first hypothesis (Tables 3 and 4). We used Mann-Whitney-Wilcoxon and Kolmogorov-Smirnov non-parametric tests to avoid the assumptions of the parametric tests.

Table 3: Technique influence on training

	GAO RT1	GAO RT2	GAO RT3	GAM RT1	GAM RT2	GAM RT3	GAA RT1	GAA RT2	GAA RT3	GAU RT1	GAU RT2	GAU RT3
Mann-Whitney U	10.000	30.000	20.000	10.000	30.000	20.000	10.000	29.500	20.000	10.000	28.500	20.000
Wilcoxon W	475.000	495.000	485.000	475.000	495.000	485.000	475.000	494.500	485.000	475.000	493.500	485.000
Z	(5.628)	(4.555)	(5.069)	(5.582)	(4.521)	(5.029)	(5.573)	(4.534)	(5.022)	(5.581)	(4.578)	(5.029)
Asymp. Sig. (2-tailed)	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000
Exact Sig. [2*(1-tailed Sig.)]	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000
Kolmogorov-Smirnov Z	2.647	2.465	2.556	2.647	2.465	2.556	2.647	2.465	2.556	2.647	2.465	2.556
Asymp. Sig. (2-tailed)	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000

Table 4: Technique influence on testing

	GAO RT1	GAO RT2	GAO RT3	GAM RT1	GAM RT2	GAM RT3	GAA RT1	GAA RT2	GAA RT3	GAU RT1	GAU RT2	GAU RT3
Mann-Whitney U	39.000	57.000	48.000	24.500	43.500	34.000	52.500	69.500	62.000	23.500	41.500	33.000
Wilcoxon W	504.000	522.000	513.000	489.500	508.500	499.000	517.500	534.500	527.000	488.500	506.500	498.000
Z	(4.777)	(3.716)	(4.219)	(5.205)	(4.139)	(4.648)	(4.369)	(3.313)	(3.766)	(5.231)	(4.207)	(4.676)
Asymp. Sig. (2-tailed)	.000	.000	.000	.000	.000	.000	.000	.001	.000	.000	.000	.000
Exact Sig. [2*(1-tailed Sig.)]	.000	.003	.001	.000	.000	.000	.001	.010	.005	.000	.000	.000
Kolmogorov-Smirnov Z	2.100	1.917	2.008	2.373	2.191	2.282	1.826	1.643	1.734	2.373	2.191	2.282
Asymp. Sig. (2-tailed)	.000	.001	.001	.000	.000	.000	.003	.009	.005	.000	.000	.000

As Table 4 shows (all significance coefficients = .000) all the pairs of accuracy rates vectors are statistically different. The direction of the difference is given by the statistics calculated. Mann-Whitney U statistic corresponds to the better group in the sense that it represents the smaller number of cases with higher ranks between groups. The Wilcoxon W statistic is simply the smaller of the two rank sums displayed for each group in the rank table. The Kolmogorov-Smirnov Z test statistic is a function of the combined sample size and the largest absolute differ-

ence between the two cumulative distribution functions of the two groups. Consequently, by analyzing both the calculated statistics and rank table we can determine the direction of the difference between the groups. For this particular experiment the rank table shows that the accuracy rates are always higher in the case of GA-based ANN training than for RT-based ANN, thus, validating first hypothesis.

As for training, GA-based ANN training models performed better than gradient-descent-like models in testing for all possible GA-RT technique-technique combinations.

Experiment 2. Here we try to validate our second hypothesis that crossover operator has influence on both training and testing performances using non-parametric tests (Table 5). As for the first experiment we used the real dataset (the original telecom data) without preprocessing the data (first preprocessing approach).

Table 5: The influence of crossover operator on training and testing

	GAO GAM TR	GAO GAA TR	GAO GAU TR	GAM GAA TR	GAM GAU TR	GAA GAU TR	GAO GAM TS	GAO GAA TS	GAO GAU TS	GAM GAA TS	GAM GAU TS	GAA GAU TS
Mann-Whitney U	35.000	40.000	45.000	45.000	32.000	36.500	45.000	47.000	41.000	49.000	45.500	44.500
Wilcoxon W	90.000	95.000	100.000	100.000	87.000	91.500	100.000	102.000	96.000	104.000	100.500	99.500
Z	(1.826)	(1.082)	(.608)	(.445)	(1.679)	(1.201)	(.610)	(.269)	(.976)	(.094)	(.548)	(.491)
Asymp. Sig. (2-tailed)	**.068**	.279	.543	.656	**.093**	.230	.542	.788	.329	.925	.584	.624
Exact Sig. [2*(1-tailed Sig.)]	.280	.481	.739	.739	.190	.315	.739	.853	.529	.971	.739	.684
Kolmogorov-Smirnov Z	.671	.671	.447	.224	.447	.447	.224	.447	.224	.447	.224	.447
Asymp. Sig. (2-tailed)	.759	.759	.998	1.000	.988	.988	1.000	.988	1.000	.988	1.000	.988

We found a very weak support: two pair-vectors differ significantly at a level of significance of 0.1: GAO vs. GAM and GAM vs. GAU, both in the case of training phase. Also, we found no evidence to differentiate between the three retraining mechanisms.

Experiment 3. Our third experiment validates third hypothesis using non-parametric tests. We preprocessed the real data using normalization and compared the results with those obtained for un-preprocessed data (Table 6). For each combination of the 2 preprocessing approaches and the 7 training techniques (4 GA-based ANN and 3 RT-based ANN) we calculated means for training and testing accuracy rates.

The preprocessing method had an impact on the both training mechanisms' performances. However, we found greater impact on the performance for training (U statistic = 0.000) than for testing (U = 6.000). Also, there is greater confidence on the results obtained for training (level of significance = 0.002) than for testing (level of significance = 0.02). Nevertheless, we obtained higher accuracy rates when we preprocessed the data using normalization than the case when we used no

Table 6: Preprocessing method influence

	PR1-PR2 (TR)	PR1-PR2 (TS)
Mann-Whitney U	.000	6.000
Wilcoxon W	28.000	34.000
Z	(3.130)	(2.380)
Asymp. Sig. (2-tailed)	.002	.017
Exact Sig. [2*(1-tailed Sig.)]	.001	.017
Kolmogorov-Smirnov Z	1.871	1.604
Asymp. Sig. (2-tailed)	.002	.012

PR1 – "no preprocessing" PR2 – "normalization"

preprocessing for both training and testing.

Experiment 4. To test our fourth hypothesis we applied the methodology on the fictive datasets and compare the results with those for the real data. In Table 7 we present the accuracy rates for training and testing samples. For this experiment we used no preprocessing of data. We calculated the means of accuracy rates vectors for each technique-distribution combination.

Table 7: Accuracy rates for distribution pairs' comparison (no preprocessing)

	REAL TR	UNIF TR	NORM TR	LOG TR	LAP TR	REAL TS	UNIF TS	NORM TS	LOG TS	LAP TS
GAO	93.02	95.84	96.39	94.82	90.09	85.24	88.57	89.46	81.19	80.41
GAM	92.86	95.84	96.60	94.99	90.00	85.48	87.86	89.64	81.43	81.02
GAA	92.92	95.78	96.49	94.80	90.16	85.48	88.93	90.00	81.43	81.22
GAU	93.30	95.76	96.43	94.82	90.19	85.95	88.75	89.82	81.19	81.02
RT1	92.22	94.92	95.48	92.70	88.12	83.49	89.11	89.46	79.92	78.10
RT2	92.43	95.04	95.45	92.70	88.06	83.97	88.57	89.52	79.92	77.76
RT3	92.41	94.84	95.52	92.53	88.06	83.81	89.05	89.52	79.29	77.89

We applied the non-parametric tests to check the validity of our fourth hypothesis (Tables 8 and 9). The hypothesis is strongly supported both for training and testing cases. There is a statistical difference in performance between all distribution pairs, except three: real-logistic and uniform-normal pairs in the case of training and logistic-Laplace pair in the case of testing. The performance order of the distributions fit our expectations; the best accuracy rates were obtained for normally distributed data, followed by data distributed uniformly. The third best performances were achieved for the real dataset which overcame logistic and Laplace distributions in this order.

Table 8: Distribution influence on training

	REAL UNIF	REAL NORM	REAL LOG	REAL LAP	UNIF NORM	UNIF LOG	UNIF LAP	NORM LOG	NORM LAP	LOG LAP
Mann-Whitney U	.000	.000	12.000	.000	12.000	2.000	.000	.000	.000	.000
Wilcoxon W	28.000	28.000	40.000	28.000	40.000	30.000	28.000	28.000	28.000	28.000
Z	(3.134)	(3.130)	(1.599)	(3.134)	(1.599)	(2.881)	(3.137)	(3.134)	(3.134)	(3.137)
Asymp. Sig. (2-tailed)	.002	.002	.110	.002	.110	.004	.002	.002	.002	.002
Exact Sig. [2*(1-tailed Sig.)]	.001	.001	.128	.001	.128	.002	.001	.001	.001	.001
Kolmogorov-Smirnov Z	1.871	1.871	1.069	1.871	1.069	1.604	1.871	1.871	1.871	1.871
Asymp. Sig. (2-tailed)	.002	.002	.203	.002	.203	.012	.002	.002	.002	.002

Table 9: Distribution influence on testing

	REAL UNIF	REAL NORM	REAL LOG	REAL LAP	UNIF NORM	UNIF LOG	UNIF LAP	NORM LOG	NORM LAP	LOG LAP
Mann-Whitney U	.000	.000	.000	.000	.000	.000	.000	.000	.000	14.000
Wilcoxon W	28.000	28.000	28.000	28.000	28.000	28.000	28.000	28.000	28.000	42.000
Z	(3.134)	(3.137)	(3.144)	(3.134)	(3.134)	(3.141)	(3.130)	(3.144)	(3.134)	(1.346)
Asymp. Sig. (2-tailed)	.002	.002	.002	.002	.002	.002	.002	.002	.002	.178
Exact Sig. [2*(1-tailed Sig.)]	.001	.001	.001	.001	.001	.001	.001	.001	.001	.209
Kolmogorov-Smirnov Z	1.871	1.871	1.871	1.871	1.871	1.871	1.871	1.871	1.871	.802
Asymp. Sig. (2-tailed)	.002	.002	.002	.002	.002	.002	.002	.002	.002	.541

Experiment 5. In the first 4 experiments, when we validate our hypotheses, we relied exclusively on non-parametric tests. We argued that the parametric tests (like t-test, univariate $ANOVA$ etc.) require the analyzed vectors to satisfy different assumptions. For instance when applying $ANOVA$ analysis one should check the following assumptions: observations are independent, the sample data have a normal distribution, and scores in different groups have homogeneous variances. The first assumption is satisfied since all other factors besides preprocessing, distribution and training mechanism that could influence the classifiers' performances are fixed. For the second assumption we argue that $ANOVA$ is robust against normality assumptions if the sample size is large. Regarding the third assumption, $SPSS$ (the software that we used) incorporates the case when the variances between groups are assumed to be non-equal.

In order to give more strength to our results we finally performed a 3-way $ANOVA$ analysis having as grouping variables: the technique used (GAO, GAM, GAA, GAU, RT_1, RT_2, and RT_3), the preprocessing method (PR_1 - "no preprocessing", PR_2 - "normalization", PR_3 - "dividing the variables by the maximum absolute values"), the data distribution ($REAL$, $UNIF$, $NORM$, LOG, and LAP). With the third preprocessing method we obtained values between -1 and +1. We used the vectors' means to fill in our accuracy rates data. Tables 10 and 11 include the data we used to perform 3-way $ANOVA$.

Table 10: Accuracy rates for training

PREPROC	DISTRIB	TECHNIQUE						
		GA				RT		
		GAO	GAM	GAA	GAU	RT1	RT2	RT3
Un-preprocessed	REAL	93.02	92.86	92.92	93.30	92.22	92.43	92.41
	UNIF	95.84	95.84	95.78	95.76	94.92	95.04	94.84
	NORM	96.39	96.60	96.49	96.43	95.48	95.45	95.52
	LOG	94.82	94.99	94.80	94.82	92.70	92.70	92.53
	LAP	90.09	90.00	90.16	90.19	88.12	88.06	88.06
Normalization	REAL	99.43	99.49	99.46	99.33	99.11	99.10	99.08
	UNIF	99.79	99.81	99.79	99.79	99.80	99.80	99.80
	NORM	97.90	97.90	97.90	97.90	98.07	98.03	97.97
	LOG	99.11	99.06	98.98	98.98	98.95	98.98	98.96
	LAP	98.08	98.13	98.01	98.10	98.02	98.01	98.06
Max of Absolute Values	REAL	99.68	99.68	99.68	99.68	99.69	99.69	99.69
	UNIF	97.79	97.77	97.73	97.84	97.77	97.77	97.89
	NORM	96.91	97.00	97.02	97.02	96.93	96.90	96.91
	LOG	96.50	96.52	96.52	96.52	96.68	96.59	96.60
	LAP	95.64	95.83	95.76	95.81	95.26	95.47	95.23

Next, the results of 3-way $ANOVA$ for both training and test accuracy rates are shown in Tables 12 and 13.

Table 11: Accuracy rates for testing

PREPROC	DISTRIB	TECHNIQUE						
		GA				RT		
		GAO	GAM	GAA	GAU	RT1	RT2	RT3
Un-preprocessed	REAL	85.24	85.48	85.48	85.95	83.49	83.97	93.81
	UNIF	88.57	87.86	88.93	88.75	89.11	88.57	89.05
	NORM	89.46	89.64	90.00	89.82	89.46	89.52	89.52
	LOG	81.19	81.43	81.43	81.19	79.92	79.92	79.29
	LAP	80.41	81.02	81.22	81.02	78.10	77.76	77.89
Normalization	REAL	85.71	86.19	85.71	85.71	85.79	85.63	85.79
	UNIF	92.86	93.04	92.86	92.86	92.86	92.86	92.92
	NORM	96.43	96.43	96.43	96.43	96.49	96.19	96.43
	LOG	88.10	88.10	88.10	88.10	88.10	88.25	88.25
	LAP	92.25	92.45	91.84	92.25	91.36	91.50	91.56
Max of Absolute Values	REAL	97.62	97.62	97.62	97.62	97.54	97.70	97.62
	UNIF	95.00	95.36	95.71	95.36	96.31	96.43	96.07
	NORM	93.21	93.57	93.93	93.57	92.86	93.15	93.27
	LOG	88.10	88.10	87.86	88.10	88.25	88.10	88.25
	LAP	88.37	89.18	88.98	89.59	89.86	89.93	89.86

Table 12: 3-way ANOVA for training

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Model	979340.512	30	32644.684	6322228.263	.000	1.000
PREPROC	540.706	2	270.353	52358.680	.000	.999
DISTRIB	148.280	4	37.070	7179.276	.000	.997
TECHNIQ	6.396	1	6.396	1238.708	.000	.943
PREPROC * DISTRIB	138.900	8	17.362	3362.559	.000	.997
PREPROC * TECHNIQ	9.486	2	4.743	918.554	.000	.961
DISTRIB * TECHNIQ	1.426	4	.356	69.036	.000	.786
PREPROC * DISTRIB * TECHNIQ	2.574	8	.322	62.310	.000	.869
Error	.387	75	.005			
Total	979340.899	105				

Table 13: 3-way ANOVA for testing

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Model	844172.942	30	28139.09	506683.910	.000	1.000
PREPROC	1296.821	2	648.411	11675.541	.000	.997
DISTRIB	904.441	4	226.110	4071.431	.000	.995
TECHNIQ	3.537	1	3.537	63.695	.000	.459
PREPROC * DISTRIB	605.016	8	75.627	1361.771	.000	.993
PREPROC * TECHNIQ	10.714	2	5.357	96.461	.000	.720
DISTRIB * TECHNIQ	5.432	4	1.358	24.454	.000	.566
PREPROC * DISTRIB * TECHNIQ	10.002	8	1.250	22.511	.000	.706
Error	4.165	75	.056			
Total	844177.107	105				

As the tables show all the factors are statistically significant. In other words they have an individual and combined influence on both training and testing performances. The last column (*partial eta squared*) reports the "practical" significance of each term, based upon the ratio of the variation (sum of squares) accounted for by the term, to the sum of the variation accounted for by the term and the variation left to error. Larger values of *partial eta squared* indicate a greater amount of variation accounted for by the model term, to a maximum of 1. Here the individual factors and their combinations, while statistically significant, have great effect on classifier accuracy.

In the next 3 tables we present the pairs' comparison for the training performances. The third hypothesis (*H3*) is validated (Table 14) and "normalization" is the best preprocessing approach, followed by "maximum absolute values" and "no preprocessing" in this order. Concerning the fourth hypothesis (*H4*) the best performance was obtained when data were *normally* distributed (Table 15). The next best distribution was that of the real data, followed by uniform, logistic and Laplace. Our main hypothesis (**H1**) is satisfied (Table 16), GA performing better than RT in *refining* the solution. However, the difference between accuracy rates is not as obvious as it was for the "real" data from experiment 1. This is explainable since in later case (only "real" data) the starting solution has relatively low accuracy rates (80-90%) and it could have been easily improved while in this experiment (centralized data) we have some starting solutions with high accuracy rates (95-98%) that would be hard to improve whatever would be the training mechanism used to *refine* them. We find no evidence for our second hypothesis (*H2*), all crossover operators achieving comparable results.

In the case of pairs' comparisons for testing performances we encounter a similar result. All the mean differences are statistically significant. Only the order of best performers has slightly changed: "maximum of absolute values" - "normalization" - "no preprocessing" for the "preprocessing" factor, uniform - normal - real -

Table 14: Pairs' comparison for "preprocessing" factor

(I) PREPROC	(J) PREPROC	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval for Difference	
					Lower Bound	Upper Bound
1	2	$(5.438)(^*)$	.017	.000	(5.473)	(5.404)
	3	$(3.933)(^*)$	.017	.000	(3.967)	(3.898)
2	1	$5.438(^*)$	.017	.000	5.404	5.473
	3	$1.506(^*)$	.017	.000	1.471	1.540
3	1	$3.933(^*)$	.017	.000	3.898	3.967
	2	$(1.506)(^*)$	.017	.000	(1.540)	(1.471)

1-"no preprocessing", 2-"normalization", 3-"maximum of absolute values"
$(^*)$ The mean difference is significant at the .05 level.

Laplace - logistic for the "distribution" factor. Once again, our main hypothesis is satisfied, GA performing better on test data as well.

7 Conclusions

In this study, we applied two different training mechanisms of an ANN to *refine* an initial solution for a classification problem. The initial solution (the ANN set of weights) was obtained when determining the ANN architecture, which was kept fixed in the refining process for both training mechanisms. An empirical procedure to determine the proper ANN architecture was introduced. The two training mechanisms are: a gradient-descent-like mechanism improved by a retraining procedure (RT) and a natural-evolution-based mechanism known as genetic algorithm (GA). Depending on where the training and validation sets are generated we have three RT-based training mechanisms and depending on the crossover operator used we have four GA-based training mechanisms. Our main hypothesis states that GA performs better than RT-based mechanism in refining the solution and is strongly supported by our experiments.

The other three hypotheses concern different factors that can have an influence on the performance of the two training mechanisms: the crossover operator, the preprocessing method of the data, and the distribution of the dataset. We found a weak support for the crossover influence, and very strong support for the other two factors in both training and testing cases. As we have shown, this study is different from the other studies that use genetic algorithms to train neural networks from at least four points of view. The most important difference is that here both training mechanisms are used to *refine* the solution obtained previously when constructing the ANN architecture.

We found that when the starting solution has relatively low accuracy rates (80-

Table 15: Pairs' comparison for "distribution" factor

(I) DISTRIB	(J) DISTRIB	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval for Difference	
					Lower Bound	Upper Bound
1	2	$(.439)(*)$	.022	.000	(.483)	(.394)
	3	$.251(*)$	.022	.000	.206	.296
	4	$.775(*)$	.022	.000	.730	.819
	5	$2.984(*)$	.022	.000	2.939	3.3028
2	1	$.439(*)$	.022	.000	.394	.483
	3	$.690(*)$	.022	.000	.645	.734
	4	$1.213(*)$	.022	.000	1.169	1.258
	5	$3.422(*)$	.022	.000	3.378	3.467
3	1	$(.251)(*)$	.022	.000	(.296)	(.206)
	2	$(.690)(*)$	.022	.000	(.734)	(.645)
	4	$.524(*)$	.022	.000	.479	.568
	5	$2.733(*)$	.022	.000	2.688	2.777
4	1	$(.775)(*)$	.022	.000	(.819)	(.730)
	2	$(1.213)(*)$	.022	.000	(1.258)	(1.169)
	3	$(.524)(*)$	.022	.000	(.568)	(.479)
	5	$2.209(*)$	.022	.000	2.164	2.254
5	1	$(2.984)(*)$	.022	.000	(3.028)	(2.939)
	2	$(3.422)(*)$	.022	.000	(3.467)	(3.378)
	3	$(2.733)(*)$	.022	.000	(2.777)	(2.688)
	4	$(2.209)(*)$	.022	.000	(2.254)	(2.164)

1-REAL, 2-NORM, 3-UNIF, 4-LOG, 5-LAP

$(*)$ The mean difference is significant at the .05 level.

Table 16: Pairs' comparison for "technique" factor

(I) TECHNIQ	(J) TECHNIQ	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval for Difference	
					Lower Bound	Upper Bound
1	2	$.499(*)$	.014	.000	.471	.527
2	1	$(.499)(*)$	.014	.000	(.527)	(.471)

1-GA, 2-RT

$(*)$ The mean difference is significant at the .05 level.

90%) GA outperformed the RT mechanism, while the difference was smaller to zero when the starting solution had relatively high accuracy rates (95-98%). This can be considered a normal result since we do not expect great improvements starting from an already very good solution. It is interesting to check in the future studies whether these hybrid approaches overcome the classical ones (the ones were the weights of the ANN are randomly initialized).

In our experiments RT was 10 times faster than GA. Therefore, when the time is a critical factor, RT can be taken into consideration as long as there is no major difference between the performances of these two approaches.

In our prediction models the number of financial performance classes is set to 7. We can easily change this parameter to simulate the binary classification problem allowing us precise and detailed comparisons with other related studies.

Acknowledgements

We wish to thank Jonas Karlsson for providing us the telecom dataset and Barbro Back for her valuable comments.

A Appendix

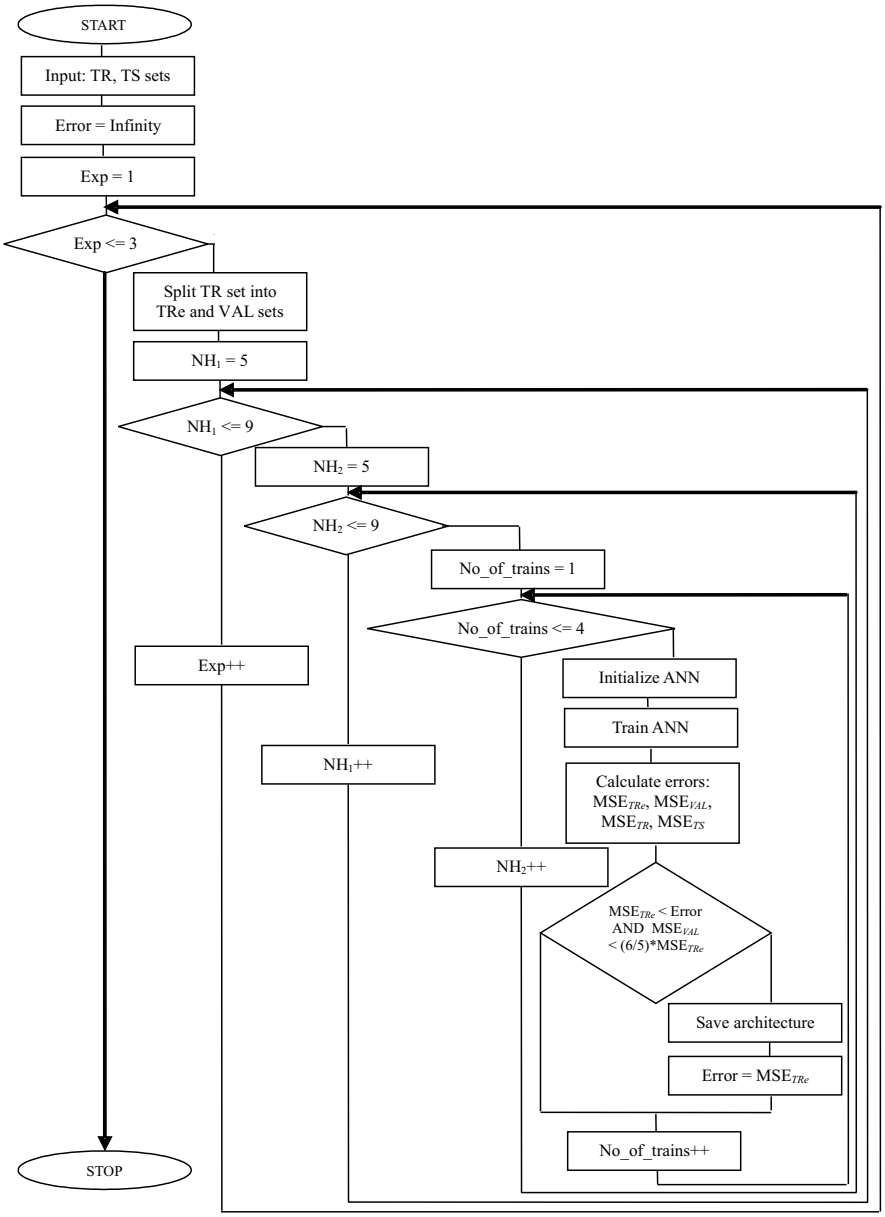

Figure 2: Flowchart of the empirical procedure to determine ANN architecture

References

[1] Alcaraz-Garcia AF, Costea A. 2004. A Weighting FCM Algorithm for Clusterization of Companies as to their Financial Performances. In *Proceedings of The IEEE 4th International Conference on Intelligent Systems Design and Applications*, Rudas I. (ed.), Budapest, Hungary. Sponsor: IEEE Systems, Man, and Cybernetics Society.

[2] Altman EI. 1968. Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy. In *The Journal of Finance* **23**: 589-609.

[3] Anderson E. 1935. The irises of the Gaspe' peninsula. In *Bull Am Iris Soc* **59**: 2-5.

[4] Ankenbrandt CA. 1991. An extension to the theory of convergence and a proof of the time complexity of genetic algorithms. In *Proceedings of 4th International Conference on Genetic Algorithm*; 53-68.

[5] Back B, Laitinen T, Sere K, van Wezel, M. 1996. Choosing Bankruptcy Predictors Using Discriminant Analysis, Logit Analysis, and Genetic Algorithms. In *TUCS Technical Report* **40**, September 1996.

[6] Back B, Laitinen T, Hekanaho J, Sere K. 1997. The Effect of Sample Size on Different Failure Prediction Methods. In *TUCS Technical Report* **155**, December 1997.

[7] Basheer IA, Hajmeer M. 2000. Artificial neural networks: fundamentals, computing, design, and application. In *Journal of Microbiological Methods* **43**: 3-31.

[8] Beaver WH. 1966. Financial Ratios as Predictors of Failure, Empirical Research in Accounting: Selected Studies. In *Supplement to Journal of Accounting Research* **4**: 71-111.

[9] Bezdek JC. 1981. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York.

[10] Breiman L, Friedman JH, Olshen R, Stone C. 1984. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA.

[11] Coakley JR, Brown CE. 2000. Artificial Neural Networks in Accounting and Finance: Modeling Issues. In *International Journal of Intelligent Systems in Accounting, Finance & Management* **9**: 119-144.

[12] Costea A, Eklund T. 2003. A two-level approach to classifying countries/companies economic/financial performance. In *Proceedings of The Thirty-sixth Annual Hawai'i International Conference on System Sciences* Decision Technologies for Management Track, Sprague RH (ed). IEEE Computer Society Press: Los Alamitos, CA.

[13] Costea A, Eklund T. 2004. Combining Clustering And Classification Techniques For Financial Performance Analysis. In *Proceedings of 8th World Multi-Conference on Systemics, Cybernetics and Informatics*. Published by IIIS, Orlando, Florida, USA, July 18-21.

[14] Costea A, Eklund T, Carlsson J. 2002. A framework for predictive data mining in the telecommunication sector. In *Proceedings of the IADIS International Conference - WWW/Internet*, Isaías P. (ed). IADIS Press, Lisbon, Portugal.

[15] Davis L (ed.). 1991. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold: New York.

[16] Demuth H, Beale M. 2001. *Neural Network Toolbox*. The MathWorks Inc, Natick Press: MA, USA.

[17] Dorsey RE, Mayer WJ. 1995. Genetic Algorithms for Estimation Problems with Multiple Optima, Non-differentiability, and other Irregular Features. In *Journal of Business and Economic Statistics* **13**(1): 53-66.

[18] Edmister RO. 1972. An Empirical Test Of Financial Ratio Analysis For Small Business Failure Prediction. In *Journal of Financial and Quantitative Analysis* **7**: 1477-1493.

[19] Fisher RA. 1936. The use of multiple measurements in taxonomic problems. In *Ann. Eugenics* **7**: 179-188.

[20] Frydman H, Altman EI, Kao DL. 1985. Introducing Recursive Partitioning for Financial Classification: The Case of Financial Distress. In *The Journal of Finance* **40**: 269-291.

[21] Hagan MT, Demuth HB, Beale M. 1996. *Neural Networks Design*. MA: PWS Publishing, Boston.

[22] Hamer M. 1983. Failure Prediction: Sensitivity of classification accuracy to alternative statistical method and variable sets. In *Journal of Accounting and Public Policy* **2**(Winter): 289-307.

[23] Jeng B, Jeng YM, Liang TP. 1997. FILM: A Fuzzy Inductive Learning Method for Automatic Knowledge Acquisition. In *Decision Support Systems* **21**(2): 61-73.

[24] Jones F. 1987. Current techniques in bankruptcy prediction. In *Journal of Accounting Literature* **6**: 131-164.

[25] Karlsson J. 2002. *Data-Mining, Benchmarking and Analysing Telecommunications Companies*. Licentiate Thesis at the Department of Information Systems at bo Akademi University, Turku.

[26] Klir GJ, Yuan B. (1995). *Fuzzy Sets and Fuzzy Logic. Theory and Applications*. Prentice Hall PTR, Upper Saddle River, New Jersey.

[27] Kohonen T. 1997. *Self-Organizing Maps*. 2nd edition, Springer-Verlag, Heidelberg.

[28] Koskivaara E. 2004. Artificial Neural Networks in Analytical Review Procedures. In *Managerial Auditing Journal* **19**(2): 191-223.

[29] Lehtinen J. 1996. *Financial Ratios in an International Comparison*. Acta Wasaensia. 49, Vasa.

[30] Marais ML, Patell JM, Wolfson MA. 1984. The experimental design of classification models: An application of recursive partitioning and bootstrapping to commercial bank loan classification. In *Journal of Accounting Research* **22**: 87-114.

[31] Michalewicz Z. 1992. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer: Berlin.

[32] Moller MF. 1993. A scaled conjugate gradient algorithm for fast supervised learning. In *Neural Networks* **6**: 525-533.

[33] Nastac DI, Costea A. 2004. A Retraining Neural Network Technique for Glass Manufacturing Data Forecasting. In *Proceedings of IJCNN 2004*, Budapest, Hungary.

[34] Nastac DI, Koskivaara E. 2003. A Neural Network Model for Prediction: Architecture and Training Analysis. In *TUCS Technical Report* **521**, April 2003. (also available on http://www.tucs.fi/publications/insight.php?id=tNaKo03a&table=techreport). [12 June 2004]

[35] Nastac DI, Matei R. 2003. Fast retraining of artificial neural networks. In *Rough Sets, Fuzzy Sets, Data Mining and Granular Computing*, Wang *et al.* (eds). Springer-Verlag in the series of Lecture Notes in Artificial Intelligence (LNAI 2639): 458-462.

[36] O'Leary DE. 1998. Using Neural Networks to Predict Corporate Failure. In *International Journal of Intelligent Systems in Accounting, Finance & Management* **7**: 187-197.

[37] Pendharkar PC. 2002. A computational study on the performance of artificial neural networks under changing structural design and data distribution. In *European Journal of Operational Research* **138**: 155-177.

[38] Pendharkar PC, Rodger JA. 2004. An empirical study of impact of crossover operators on the performance of non-binary genetic algorithm based neural approaches for classification. In *Computers & Operations Research* **31**: 481-498.

[39] Quinlan JR. 1993. A Case Study in Machine Learning. In *Proceedings of ACSC-16 Sixteenth Australian Computer Science Conference*, January, Brisbane; 731-737.

[40] Quinlan JR. 1993. *C4.5 Programs for Machine Learning*. Morgan Kaufmann Series in Machine Learning. Morgan Kaufmann Publishers, San Mateo.

[41] Rudolfer S, Paliouras G, Peers I. 1999. A Comparison of Logistic Regression to Decision Tree Induction in the Diagnosis of Carpal Tunnel Syndrome. In *Computers and Biomedical Research* **32**: 391-414.

[42] Schaffer JD, Whitley D, Eshelman LJ. 1992. Combinations of Genetic Algorithms and Neural Networks: A survey of the state of the art. In *COGANN-92 Combinations of Genetic Algorithms and Neural Networks*. IEEE Computer Society Press: Los Alamitos, CA; 1-37.

[43] Schaffer JD. 1994. Combinations of genetic algorithms with neural networks or fuzzy systems. In *Computational Intelligence: Imitating Life*, Zurada JM, Marks RJ, Robinson CJ (eds). IEEE Press: New York; 371-382.

[44] Schütze H, Hull D, Pedersen J. 1995. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, Seattle, Washington, United States. ACM Press: New York, NY, USA; 229-237.

[45] Sexton RS, Dorsey Re, Johnson JD. 1998. Toward a global optimum for neural networks: A comparison of the genetic algorithm and backpropagation. In *Decision Support Systems* **22**(2): 171-186.

[46] Sexton RS, Sikander NA. 2001. Data Mining Using a Genetic Algorithm-Trained Neural Network. In *International Journal of Intelligent Systems in Accounting, Finance & Management* **10**: 201-210.

[47] Siegel S, Castellan-Jr. NJ. 1988. *Nonparametric Statistics for the Behavioral Sciences*, 2nd edition. McGraw-Hill International Editions.

[48] Zavgren C. 1985. Assessing the vulnerability to failure of American industrial firms: A logistics analysis. In *Journal of Business Finance and Accounting* (Spring): 19-45.

[49] Zupan B, Demšar J, Kattan MW, Ohori M, Graefen M, Bohanec M, Beck JR. 2001. Orange and Decisions-At-Hand: Bridging predictive data mining and decision support. In *IDDM-2001: ECML/PKDD-2001 Workshop Integrating Aspects of Data Mining, Decision Support and Meta-Learning*, Freiburg, Giraud-Carrier C, Lavrac N, Moyle S, Kavšek B (eds); 151-162.

Turku
Centre *for*
Computer
Science

Lemminkäisenkatu 14 A, 20520 Turku, Finland | www.tucs.fi

University of Turku
- Department of Information Technology
- Department of Mathematics

Åbo Akademi University
- Department of Computer Science
- Institute for Advanced Management Systems Research

Turku School of Economics and Business Administration
- Institute of Information Systems Sciences

ISBN 952-12-1530-5
ISSN 1239-1891