Initializations, Back-Propagation and Generalization of Feed-Forward Classifiers

Wouter F. Schmidt[1], Sarunas Raudys[2], Martin A. Kraaijveld[1], Marina Skurikhina[2] and Robert P.W. Duin[1]

[1]Pattern Recognition Group Delft,
Faculty of Applied Physics,
Delft University of Technology,
P.O. Box 5046,
2600 GA Delft,
The Netherlands

[2]Department of Data Analysis
Institute of Mathematics and Informatics,
Akademijos 4,
Vilnius 2600,
Lithuania.

Abstract—The back propagation method is very sensitive to the initial weights. A commonly used heuristic is to train a large number of networks, using different initial weights for training. The network with the lowest mean squared error is selected from those networks as the optimal network. In this paper we will show that this simple heuristic, meant to improve network training, sometimes favors neural network classifiers with poor generalization capabilities. We will propose a measure to quantify this phenomenon and will study it as a function of the training time.

1. INTRODUCTION

The multi-layer feed forward neural networks has become one of the most widely used *classification* techniques during the recent years. The back propagation training algorithm (Rumelhart *et al.*[7]) is simple and elegant from a mathematical and implementation point of view. The training, although slow in practice, is easily mapped to parallel machines if the processing power of sequential machines is not enough. In recall mode (if the network is trained) the feed forward classifier is generally fast and special hardware exists to do real-time classification if needed. A chip set from Intel for example, implements a general feed forward network and the user can down load the weights on to the chip.

Besides that it is easy to implement the back propagation method, the user will notice that in practice it is not so easy at all to solve a particular application with this method. The back propagation method is known to be sensitive to the initial weights (see Kolen and Pollack[3]) and the training may be regarded to be an unstable system from traditional signals and systems viewpoint. A common heuristic is not to train only one feed forward network but to train a number of networks, that differ in the initial weights only. The network with the lowest value of the mean squared error is selected as the optimal network.

The objective function that is optimized to find a set of weights, is the mean squared error on a randomly selected learning set. In pattern recognition however the probability of error, or better the generalization of that for the whole population, is the main objective. The question is now, what is the effect of the *initializations* heuristic on the generalization capabilities of the obtained neural classifiers? Will this heuristic always favor the network with higher generalization capabilities or may this procedure be harmful? We will show experimentally and explain that the latter may be true.

In this paper we will advocate the idea that training a network using the back propagation method is a stochastic process and any change to the training algorithm will change the underlying probability distribution that controls the learning. A training session is no more than a random draw from this distribution. Important for the expected performance is the joint probability distribution of the mean square error that is optimized and the probability of error for the entire population.

This paper is organized as follows. In chapter 2 we will describe the basic definitions, the type of neural network, the training method and the artificially generated data set we have used for the experiments. In chapter 3 we will examine the stochastic nature of back propagation training and the consequences of selecting a network from a number of initializations, illustrated with an example. An important issue is the relation between the mean squared error and the generalization properties of the classifiers obtained. This will be the topic of chapter 4. In chapter 5 we will conclude with a discussion and some recommendations.

0-7803-0999-5/93/$03.00 ©1993 IEEE

2. Basic Definitions

2.1. The network architecture

The artificial neural networks considered here are standard feed-forward single hidden layer networks (see Rumelhart *et al.*[7]), which means that the processing units are fully connected to units in a previous layer but are not connected to units in the same layer. Only the outputs of the units in a layer are connected to units in a next layer. Therefore there is no feed back in the system. Furthermore only one hidden layer is used. As Funahashi[1] and Hornik[2] have shown, this is no fundamental restriction to the approximation capabilities of a network.

The units in the hidden layer of the feed forward network consist of processing elements that calculate the weighted sum of the inputs. Applied to this weighted sum is a so called squashing function F(x) that maps this value in the range between 0 and 1. The typical squashing function, used in all our experiments, is the standard sigmoid function introduced by Rumelhart *et al.*[7].

In chapter 3 and 4 we will show some statistics of network training to illustrate the problems that can arise when a network is trained for a specific task. In all examples we have used a network with one hidden layer, 3 units in that layer and one output unit (2-3-1).

2.2. The Loss function

The network weights are chosen to minimize an objective error function, called a loss function. This loss function is a function of the difference of the network output and the desired output. A general form of a loss function may be expressed as:

$$Loss = \frac{\sum_{i}^{patterns} \psi(t_i - O(\mathbf{X}_i))}{\#\ patterns} \tag{1}$$

Two common pattern loss functions are $\psi 1(d) = d^2$ and the classification error function, $\psi 2(d)=0$ if $\mathbf{X}_i$ is classified correctly and $\psi 2(d)=1$ if $\mathbf{X}_i$ is classified incorrectly. A sample is regarded to be classified incorrectly if the difference between the target and output is larger than a specific threshold. For target coding (t_i) 0.9 and 0.1 for two classes A en B respectively, this threshold is set to 0.4.

Both these loss functions will be used. The loss function $\psi(d) = d^2$ is the well-known mean squared error (Mse) criterion that will be used to optimize the network weights:

$$Mse = \frac{1}{\#\ patterns} \sum_{i}^{patterns} (t_i - O(\mathbf{X}_i))^2 \tag{2}$$

The mean squared error can be measured on the training set (notation Mse^{train} or shortly Mse in this paper), or on an independent test set (notation Mse^{test}). The second loss function $\psi 2$ is be called the probability of error and can be measured on the training set or on a the independent test set (P^{train} and P^{test} respectively).

The introduction of two error measures seems to be confusing, but the learning rule (back-propagation) for the optimization of the network weights, requires a loss function with a continuous first order derivative. If a network is used for classification purposes a low value of the Mse is not of main concern, but instead the probability of error should be as low as possible. The first derivative of this function is not continuous, and can therefore not be used together with the back-propagation learning rule.

2.3. The training algorithm

The optimization method used to train the feed forward classifier is the well-known *vanilla* back-propagation method as introduced by Rumelhart *et al.*[7]. This method is a gradient descent method which is a modified steepest descent. The basic update rule can be written as:

$$\theta_i(t+1) = \theta_i(t) + \Delta\theta_i(t) + \alpha \cdot \Delta\theta_i(t-1) = \theta_i(t) - \eta \frac{\partial Mse}{\partial \theta_i} + \alpha \cdot \Delta\theta_i(t-1) \tag{3}$$

In this formula $\theta_i(t)$ is weight i at time t. The parameter's η and α are called the learning rate and momentum respectively. The details of this update rule can be found in Rumelhart *et al.*[7] and are omitted here.

2.4. The data set used

The data set, used in experiments described in chapter 3 and 4, consists of two banana-shaped classes in a two dimensional space. The statistics of class A and B are given below:

$$\mathbf{X}_a = \begin{pmatrix} \rho_a \cos(\gamma_1) + \zeta_1 \\ \rho_a \sin(\gamma_1) + \zeta_2 \end{pmatrix}, \quad \mathbf{X}_b = \begin{pmatrix} \rho_b \cos(\gamma_2) + \zeta_3 \\ \rho_b \sin(\gamma_2) + \zeta_4 \end{pmatrix} \tag{4}$$

In this formula ζ_i is a Gaussian random sample with zero mean and unit variance (N(0,1)) and γ_i is a uniform distributed random sample from the interval $[-\pi/3, \pi/3]$. The radii's are ρ_a=6.2, ρ_b=10.0 for class A and B respectively. The Bayes optimal solution is has a probability of error (P^{test}) of approximately 2.9%.

3. Training a Classifier in Practice

The error surface is complex and the learning process can be trapped in many local minimums or area's were the mean squared error is only slowly varying. A serious problem is

the sensitivity of the learning method for the initial weights (Kolen and Pollack[3]). Different starting positions in weight space lead to different solutions, and the final classifier obtained depends on the initial weights.

Because the error surface is so complex and the final classifier is sensitive to the initial weights, it is almost impossible to predict the performance of the classifier that will be obtained. This leads us to the idea that the learning process is a stochastic process and one learning session is just a realization of that process. The probability distribution of the learning process, describing the probability that a certain classifier will be found, is important in this context.

A commonly used heuristic in feed forward classifier design is the idea to train m network classifiers, using different starting points in weight space, but using the same learning data. The network with the lowest Mse is chosen from those m initializations. This heuristic is meant to improve the classifier and to make the learning less sensitive to the initial weights. It is obvious that the probability distribution of the best out of m networks differs from the plain probability distribution. It is important to know how this heuristic changes the probability to obtain a good classifier.

In paragraph 3.1 we will first investigate the influence of the number of training epochs on the probability distribution of the Mse. In the next paragraph we will focus on the problem of the selection of the best network out of m initializations. In the training process a direct objective is to minimize the Mse. In application of the neural network classifier however the unknown generalization error P^{test} of the classification rule is of primarily interest. We will see that this is an important issue, that must not be neglected on the selection process. This will be investigated in chapter 4.

3.1. The probability distribution of the Mse

Let us take a *large* number of random initializations and calculate the final Mse for each initialization. The Mse value for each initialization can be assumed to be a realization of a random variable. It will vary in an interval $[Mse^{min}, Mse^{max}]$ and the probability density function will generally depend on a way the initial values of the weight vector have been chosen.

The number of training sweeps used to train the neural network has a significant influence on the distribution of the Mse values. An increase in the number of training sweeps will generally narrow the interval $[Mse^{min}, Mse^{max}]$ of the distribution. In a case of *perfect* training most of the values of the mean squared error will lie close to each other.

In figure 1 we present four distribution density functions of the Mse evaluated empirically for 1000 initializations. The standard back-propagation training algorithm of Rumelhart *et al.*[7] was used with η=0.2 and α=0.0. The data set and network architecture are described in paragraph 2.1 and 2.4. The number of training samples was 8+8 (8 per class). The number of testing samples to estimate the generalization error P^{test} was 1000+1000 (1000 per class). The density functions in this graph differ in the number of sweeps used to train classifier. Only one set of training patterns and one set of 1000 initial weight vectors was used to obtain all 4 histograms.

We see the number of the training sweeps is a very important factor to the shape and location of the density function f(Mse). In most cases the densities are not unimodal (figure 1c). It indicates that for this particular set of training patterns and architecture we have at least two different *local* minimums. A more detailed analysis of the weight vectors obtained after 200,000 training sweeps has shown that the weight vectors and decision boundaries corresponding to the left mode are different, while these for the right mode are close and can be supposed to belong to the same minimum.

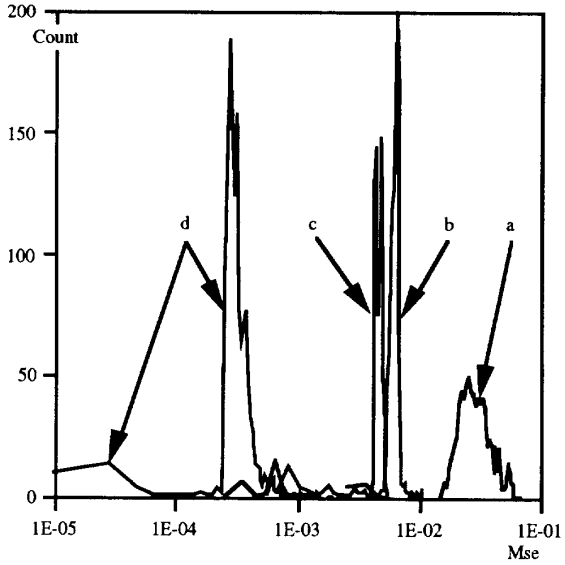

Fig. 1a, 1b, 1c, 1d: Histograms of the Mse distribution for different number of training sweeps. Two layer network with 2 inputs, 3 neurons in the hidden layer, 1 output, 8+8 training samples, the standard back-propagation training, η=0.2, α=0. Figures a, b, c and d are trained with 1000, 4000, 20000 and 200000 sweeps respectively.

Therefore to obtain a small value of the Mse, the use of one initialization and many learning sweeps is not sufficient. We have to use at least several initializations and apply a large number of the training sweeps.

An open unsolved question remains: when to stop to train the ANN classifier and how large the number of initializations should be chosen. It is difficult to obtain a correct answer to these questions before the neural network training process begins. It is probably true that the increase

in the number of training sweeps can not replace the number of initializations and vice versa.

3.2. Selecting the best classifier from more initializations

We have trained m neural network classifiers, each time starting from a different starting point and for all m times use the same set of training patterns. The final mean squared errors Mse(1), Mse(2), Mse(3), , Mse(m) (a measurement on the training set) for all m neural network classifiers are measured. Suppose now we have a very large set of test patterns that can be used to evaluate performance measures (generalization errors P^{test}) of these m classifiers. Let $P^{test}(1)$, $P^{test}(2)$, $P^{test}(3)$, , $P^{test}(m)$ be the values of these measures. Then each neural classifier obtained after each separate training session is characterized by a vector:

$$\begin{bmatrix} Mse(j) \\ P^{test}(j) \end{bmatrix} \quad \forall j = 1,2,..,m \tag{5}$$

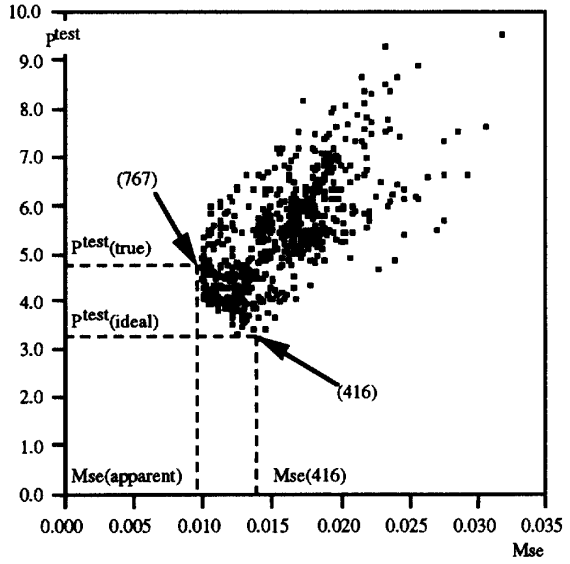

Fig. 2. A scatter diagram with horizontally the mean squared error and vertically the generalization error P^{test} (percentage). The data points correspond to trained networks, trained on the same learning set only from a different starting point. The measured correlation between Ptest and Mse is ρ=0.69.

In figure 2 an example scatter is given of 1000 of these vectors, corresponding to 1000 trained feed forward classifiers. Each classifier is trained with the same data and only differs in the random starting point. New 13-dimensional weight vectors were obtained after each random weight initialization followed by 5000 training sweeps (epochs). The standard back-propagation training algorithm of Rumelhart *et al.*[7] was used with η=0.3 and α=0.1. The data set distribution and network architecture are described in paragraphs 2.1 and 2.4. The number of training samples was 80+80 (80 per class). The number of testing samples to estimate the generalization error P^{test} was 1000+1000 (1000 per class).

In selecting the best network the normal thing to do is to use the estimates of the mean square error Mse(1), Mse(2), , Mse(1000). The *best* initialization appeared to be trial number 767. Its resulted Mse(min)=0.0098=Mse(767). The true error, the generalization error P^{test} of this classifier appeared to be $P^{test}(767)$=0.047. From the scatter diagram we see that the best initialization was number 416 with $P^{test}(416)$=0.032! Thus inexact selection of the *best* initialization increased the generalization error from 3.2% to 4.7%.

In the following we shall call the lowest value of the estimates Mse(j) the *apparent* error in neural network training. The actual error $P^{test}(j)$ of the corresponding network we shall call the *true* generalization error. The minimum generalization error (P^{err}) of the set will be called the *ideal* generalization error. For the above example the values are as follows:

Mse(apparent)	= Mse(767)	= 0.0098,
P(true)	= $P^{test}(767)$	= 0.047,
P(ideal)	= $P^{test}(416)$	= 0.032.

The difference D = P(true)-P(ideal) is the increase in generalization error due to non ideal selection of the best network initialization. The increase in the generalization error D depends on the set of vectors defined by equation 5.

This set depends on the architecture of the neural network classifier, distribution density function of the pattern vectors, the actual set of training pattern vectors used to obtain the estimates MSE(j), and of course on the training procedure used.

4. Correlation Between Mse and P^{test}

If the classifier is trained for only few training epochs, the Mse is comparatively large and Mse varies over a wide interval. We have seen in the previous section that the interval becomes more narrow and shifts to the left (see figure 3) as the number of the training sweeps increases. There is no reason to expect that when the Mse is relatively large, we shall obtain a network with small generalization error. Contrary, when Mse is large the generalization error will be large too and if Mse will be small, one can hope to obtain a small value of P^{test}. Therefore in a case of insufficient learning one can expect a wider distribution of Mse-P^{test} and a large positive correlation between them. Assume now that in each training session the classifier is trained sufficiently, using a *large* number of training sweeps.

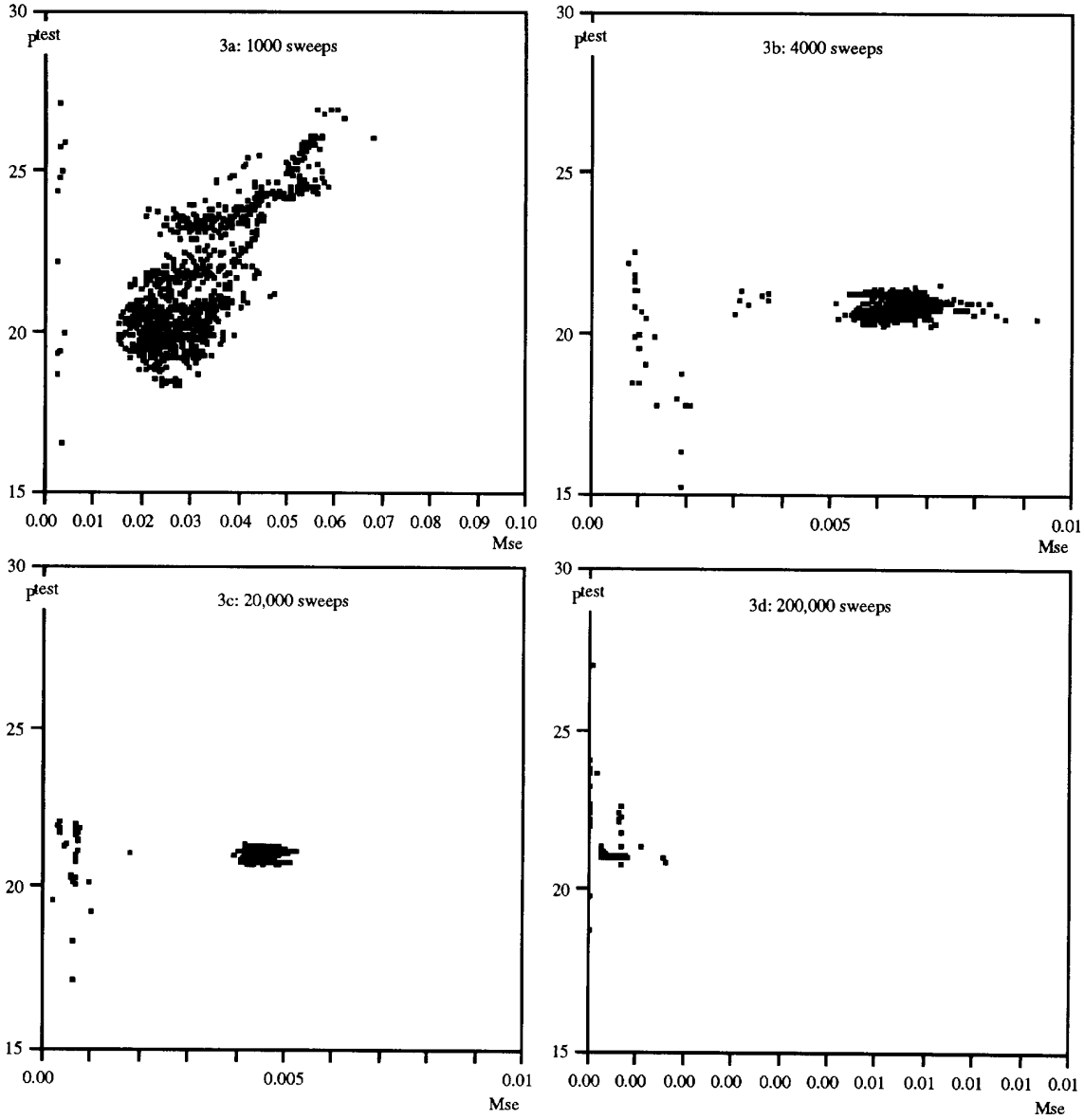

Fig. 3a, 3b, 3c, 3d. Four scatter diagrams Mse versus P^{test} (percentage) obtained for different number of the training sweeps, based on 1000 initializations. For the experimental settings see figure 1. The measured correlation's between P^{test} and Mse are respectively ρ_a=0.71, ρ_b=0.31, ρ_c=0.06, ρ_d=-0.22. Note the difference in scale of figure 3a.

In this situation we shall obtain classifiers only from the very left segment of the cloud Mse-P^{test} (see figure 2) and one can expect to obtain a lower correlation between Mse and P^{test}.

This deduction is confirmed by experimental observations. In figure 3 we present a series of bi-variate distributions Mse-P^{test} depicted for the same experimental conditions as the histograms presented in figure 1. We see that with an increase in the number of the training sweeps, two clusters become distinguishable. One may hope they correspond to two different minimums. A more detailed analysis of the weight vectors obtained after 200000 training sweeps indeed indicated that the weights that correspond to the left cluster produce different decision boundaries. These solutions correspond either to different local minimums or to different parts of a *flat* area (or saddle structure) of the loss function

(equation 2). All the weight vectors that correspond to the cluster on the right side produce a similar decision boundary, so one can expect that they correspond to one (local) minimum.

One can obtain even negative correlation between Mse and P^{test}. Such a situation we have observed in figure 3d where for 200000 training sweeps the correlation is ρ=-0.22. In this case an increase in the number of initializations (also in the number of the training sweeps) only results in the selection of a classifier with a *larger* generalization error.

The sign and value of the correlation between Mse and P^{test} depend on how well the training pattern vectors reflect the general population. When training patterns have a different structure than the general population (this relative to the considered network architecture), than one can expect the correlation to be small or even be negative. The distribution of Mse-P^{test} depends on how well the training samples represent the statistics of the general population, which is impossible to predict without some additional information. It is thus impossible to predict from the training samples only, whether the correlation will be positive or negative. This is a well-known fact in classical statistics; from the learning samples it is impossible to predict if the true mean squared error is larger or smaller than the mean squared error measured on this learning set.

The set of the training pattern vectors is usually assumed to be a random one. Therefore one can assume the correlation discussed here to be a random variable too. We investigated the correlation for 4 different 40+40 training sets, trained for 4000 epochs. The correlation found for these experiments differed between -0.10 and 0.88. We want to note that some of these values are largely dominated by outliers. The distributions of Mse-P^{test} for these experiments are however different. This confirms that individual peculiarities of the particular set of training samples play an important role on the form of the distribution Mse-P^{test}. In a case of small or negative correlation between Mse and P^{test} an excessive increase in the number of initializations is not useful but is even harmful. Sometimes even an excessive increase in the number of the training sweeps causes an increase in the generalization error.

This phenomenon is called *over-training*. It is caused by the fact that the minimization criterion (Mse) is different from that used to evaluate the final performance of the classification rule (the generalization or test sample error). After a certain point, a longer training of the network will increase the effect of over-training. Although the Mse is still decreasing, the generalization error P^{test} increases. This results into a negative correlation between Mse and P^{test}. The value of the correlation between Mse-P^{test} is probably a good *quantitative* measure for over-training, in the context of using a feed forward network for pattern recognition.

5. Discussion and Conclusions

The weight initialization plays an important role in the ANN training. One initialization is often not sufficient for the ANN training. To be sure one must perform several training sessions each time beginning from a different initial weight vector. The optimal number of initializations needed to find a good classifier, is heavily dependent on the probability distributions of the inputs, as well as the particularities of the used learning set.

An excessive increase in the number of initializations or training sweeps does not always lead to a better classifier. There is no way to predict the usefulness of an increase in the number of sweeps or an increase of the number of initializations. The only recommendation is to use a sufficiently large number of the training patterns. In Raudys[4] [5] the influence of the sample size on the model selection in pattern recognition is investigated. It is shown there that the increase in the classification error due to improper model selection, is of the order of a standard deviation of the re-substitution estimate P^{train} (probability of error on the learning set) of the classification error used to choose the best initialization. The standard deviation for p^{train} is:

$$\sigma^{train} = \sqrt{\frac{P^{train} \cdot (1 - P^{train})}{n}} \qquad (6)$$

If $p^{train} << 1$ than the recommendation follows:

$$n >> 1/ p^{train} \qquad (7)$$

The above equation can be used to determine a number of the training samples necessary to avoid (possibly only to detect) unpleasant effects of over-training. The number of training vectors determined from the above inequality is *not* the number of the samples necessary to train a neural classifier with good generalization. This depends on the complexity of the classifier (the number of inputs d, the number of neurons in the hidden layer) as well as on a complexity of the *pattern* space see Raudys and Jain[6].

This last result will only ensure that the increase in the generalization error caused by the non ideal selection of the best initialization will be sufficiently small. It does not give any guaranty on the generalization capabilities of the neural classifier.

The correlation between the mean squared error (Mse) and the generalization error (P^{test}) is proposed as a quantitative measure for a phenomenon often revered to as *over-training*. If a strong negative correlation exists between these two, multiple initializations and the selection of the initialization with the lowest Mse, will lead to the selection of the *worst* classifier. This suggests that the network complexity and training do not match to the data complexity and the number

of training samples. This can arise due to a particularity of a training set or be more systematic for a certain distribution and network choice.

ACKNOWLEDGMENT

This work was sponsored by the Dutch Government as a part of the SPIN-FLAIR-DIAC project, and by the Foundation of Computer Science in the Netherlands (SION) with financial support from the Dutch Organization for Scientific Research (NWO).

REFERENCES

[1] K. Funahashi, "On the Approximate Realization of Continues Mappings by Neural Networks.", Neural Networks, Vol 2 page 183-192, 1989.

[2] K. Hornik, "Multilayer Feedforward Networks are Universal Approximators", Neural Networks, Vol 2 page 359-366, 1989.

[3] J.F. Kolen and J.B. Pollack, "Back Propagation is Sensitive to Initial Conditions", Advances in Neural Information Processing Systems 3, edited by Lippmann, Moody and Touretzky, Morgan Kaufmann Publishers, San Mateo, page 860, 1991.

[4] S.Raudys, "An influence of sample size on the accuracy of model selection in pattern recognition". Statistical problems of control, Issue 50, (S.Raudys ed.), Inst. Math.& Cybernetics press, Vilnius, pp.9-30 (in Russian).

[5] S.Raudys,"On accuracy of model selection in data analysis" Proc. of III-rd Int. Conf. on data Analysis and Informatics. INRIA Press, Paris, 1987, pp. 91-98.

[6] S.Raudys and A.Jain, "Small sample Problems in Designing Artificial Neural Networks, Artificial Neural Networks and Statistical Patters recognition, Old and New Connections, Sethi and Jain editors, Elsevier Science Publishers B.V,pp. 33-50.

[7] D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation", in "Parallel Distributed Processing: Exploring in the Microstructure of Cognition", Vol 1, D.E Rumelhart and J.L. McClelland (Eds.), Cambridge, MA: MIT Press, pp 318-362.