Influence of weight initialization on multilayer perceptron performance

M. Karouia[1,2] T. Denœux[1]

R. Lengellé[1]

[1] Université de Compiègne – U.R.A. CNRS 817 Heudiasyc
BP 649 - F-60206 Compiègne cedex - France
mkarouia@hds.univ-compiegne.fr

[2] Lyonnaise des Eaux (LIAC)

Abstract

This paper presents a new algorithm for initializing the weights in multilayer perceptrons. This method is based on the use of feature vectors extracted by discriminant analysis. Simulations carried out with real-world and synthetic data sets show that the proposed algorithm allows to obtain a better initial state, as compared to random initialization. As a result, training time is reduced and lower generalization error can be achieved. Additionally, it is shown through numerical simulations that the generalization performance of networks initialized with the proposed method becomes less sensitive to network size and input dimension.

1 Introduction

Many researchers have emphasized the importance of initial weights in multilayer perceptron (MLP) training. Several initialization algorithms have been proposed, such as the use of prototypes [2]. The most obvious potential benefits of starting optimization from a good initial state are faster training and

higher probability of reaching a deep minimum of the error function. Additionally, it has been found that introducing prior knowledge in the initial weights may in some cases improve generalization performance [2, 8]. In this paper, a new approach to weight initialization is proposed, and its effect on generalization is demonstrated experimentally.

The starting point of this work is the relationship between MLPs and discriminant analysis (DA) pointed out by Gallinari [4]. It can be shown that training networks with one hidden layer using the quadratic error function is equivalent to maximizing a measure of class separability in the space spanned by hidden units. DA techniques aim at extracting features that are effective in preserving class separability. The algorithm presented in this paper (WIDA: Weight Initialization by Discriminant Analysis) proposes to use such features for initializing the weights in multilayer networks before training by standard back propagation (BP) or any other learning procedure. The performance of the WIDA method is then analyzed using several synthetic and real-world data sets. We examine the effect of weight initialization on the following aspects: convergence speed (training time), generalization error and sensitivity of generalization error to data dimensionality and number of hidden units.

2 The initialization method

2.1 Discriminant analysis

We consider a set $\mathcal{X}$ of N samples in a d-dimensional space. The samples are assumed to be partitioned into M disjoint subsets. Subset $\mathcal{X}_i$ of size N_i includes samples properly associated with class Ω_i. Let $\boldsymbol{x_{ij}}$ be the j-th d-dimensional sample vector from class Ω_i. The mean vector of class Ω_i is $\boldsymbol{m_i} = \frac{1}{N_i}\sum_{j=1}^{N_i}\boldsymbol{x_{ij}}$. The overall mean vector is $\boldsymbol{m} = \frac{1}{N}\sum_{i=1}^{M} N_i\boldsymbol{m_i}$. We define the *parametric within-class* scatter matrix W and the *parametric between-class* scatter matrix B respectively as:

$$W = \frac{1}{N}\sum_{i=1}^{M}\sum_{j=1}^{N_i}(\boldsymbol{x_{ij}} - \boldsymbol{m_i})(\boldsymbol{x_{ij}} - \boldsymbol{m_i})^T \quad (1)$$

$$B = \frac{1}{N}\sum_{i=1}^{M} N_i(\boldsymbol{m_i} - \boldsymbol{m})(\boldsymbol{m_i} - \boldsymbol{m})^T \quad (2)$$

where $(.)^T$ denotes transposition. Matrix W is assumed to be positive definite, so that W^{-1} exists. Matrix B is a positive semidefinite matrix with rank at most equal to $M-1$ (we assume that $d \geq M$). The sum of W and B gives the parametric global covariance matrix G.

In parametric discriminant analysis (PDA), we seek d-dimensional feature vectors $\boldsymbol{\tau}$ maximizing the Fisher's criterion $J(\boldsymbol{\tau})$:

$$J(\boldsymbol{\tau}) = \frac{\boldsymbol{\tau}^T B \boldsymbol{\tau}}{\boldsymbol{\tau}^T W \boldsymbol{\tau}} \tag{3}$$

Such features are obtained as the eigenvectors of $W^{-1}B$, each eigenvalue λ_i being equal to the Fisher criterion of its corresponding eigenvector $\boldsymbol{\tau_i}$ ($J(\boldsymbol{\tau_i}) = \lambda_i$).

PDA has two serious shortcomings. First, the maximum number of discriminant vectors is limited to $M-1$. When $M = 2$, PDA allows to extract only one discriminant vector. The second and more fundamental problem is the intrinsic parametric nature of PDA. When the class distributions are significantly non-normal, the use of PDA cannot be expected to accurately determine good features preserving the complex structure needed for classification.

Non-parametric discriminant analysis (NPDA) was introduced to overcome both of the aforementioned problems [3]. It is based on the use of a *non-parametric between-class* scatter matrix that measures between-class scatter on a local basis, using a k-nearest neighbor (k-NN) approach. Let us first consider the case where $M = 2$. Let $\boldsymbol{n_{il}}(\boldsymbol{x}) \in \mathcal{X}_\rangle$ $(l = 1, \ldots, k)$ be the k nearest neighbors in class Ω_i of an arbitrary sample $\boldsymbol{x} \in \mathcal{X}$. The local mean of class Ω_i (the sample mean of the k NNs from Ω_i to $\boldsymbol{x}$) is $\boldsymbol{m_{ki}}(\boldsymbol{x}) = \frac{1}{k}\sum_{l=1}^{k} \boldsymbol{n_{il}}(\boldsymbol{x})$. The non-parametric between-class scatter matrix is then defined as

$$\begin{aligned} \mathcal{B}_{12,k} &= \frac{1}{N}\Big(\sum_{\boldsymbol{x}\in\mathcal{X}_\infty} p_{12}(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{m_{k2}}(\boldsymbol{x}))(\boldsymbol{x} - \boldsymbol{m_{k2}}(\boldsymbol{x}))^T \\ &+ \sum_{\boldsymbol{x}\in\mathcal{X}_\in} p_{12}(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{m_{k1}}(\boldsymbol{x}))(\boldsymbol{x} - \boldsymbol{m_{k1}}(\boldsymbol{x}))^T\Big) \end{aligned} \tag{4}$$

The term $p_{12}(\boldsymbol{x})$ is defined as a function of the distances between $\boldsymbol{x}$ and its k-th nearest neighbor from each class [3]. Its role is to deemphasize the samples located far away from the class boundary.

By substituting B with $\mathcal{B}_{12,k}$ in Equation 3, we obtain a non-parametric Fisher's criterion $\mathcal{J}(\boldsymbol{\tau})$. The features maximizing $\mathcal{J}(\boldsymbol{\tau})$ can be obtained as the eigenvectors of $W^{-1}\mathcal{B}_{12,k}$. Since $\mathcal{B}_{12,k}$ is generally full rank, the number of discriminant vectors is not limited to $M-1$.

To extend NPDA to general M-class problems, two alternatives have been studied. The first one consists in considering M two-class problems or dichotomies. For each dichotomy, we take one class as Ω_1 and the other $M-1$ classes as Ω_2; d discriminant vectors are extracted by the above procedure. Afterwards, the best discriminant vectors can be chosen according to some selection procedure. The second alternative consists in defining a generalized non-parametric between-class scatter matrix as $\mathcal{B}_k = (1/N^2)\sum_{i<j} N_i N_j \mathcal{B}_{ij,k}$.

2.2 Application to weight initialization

The WIDA method consists in initializing the hidden unit weights as discriminant vectors extracted by non-parametric DA, and adding bias terms. Learning is then carried out in 3 steps:

1. the biases of hidden neurons are determined so as to maximize class separability in the space $\mathcal{H}$ spanned by hidden units. As shown in [5, 6], a suitable measure of class separability is $\mathrm{tr}(G_h^{-1}B_h)$, where G_h and B_h are respectively the total and between-class scatter matrices in $\mathcal{H}$.

2. the hidden-to-output weights are initialized randomly and trained separately to minimize the mean squared output error;

3. finally, further training of the whole network is performed using the standard back propagation algorithm.

3 Comparison to random initialization

The above initialization procedure was tested and compared to other methods using the following data sets:
Waveform data: it is a three-class synthetic problem in a 21-dimensional feature space. Training and test sets both contain 100 samples of each class [1].

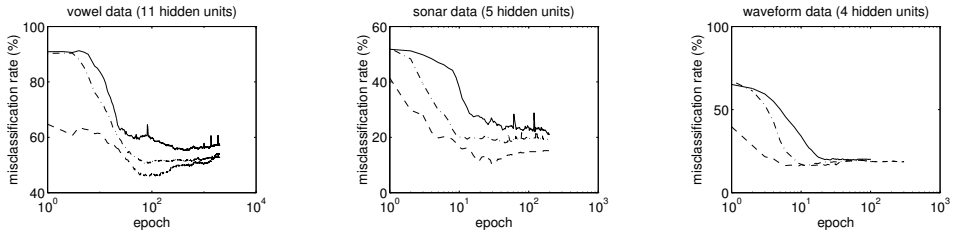

Figure 1: Mean test misclassification rate as a function of training cycles (averages over 10 trials). — : random; - - : WIDA; -.- : prototype method.

Vowel data: training and test data have 10 features and are partitioned in 11 classes. We used 528 randomly chosen samples for training and the 462 remaining samples for the test. A complete description of this data is given in [7].

Sonar data: this is a real-world classification task [7] with 60 features and 2 classes. Training and test data are both of size 104.

The network weights were initialized with the WIDA algorithm, the prototype method and randomly. For each classification task, the number n of hidden units was varied from 2 to $n_{max} \leq d$. Training and test misclassification error rates were computed after each learning cycle. The algorithm was run 10 times for each value of n and each initialization method. Figure 1 shows the evolution of mean error rates as a function of time for the three tasks. The means of the best error rates obtained at each trial by the three methods are represented in Figure 2 as a function of n.

As expected, these results show that the WIDA method provides "good" initial solutions in terms of misclassification error. This results in faster training, although the gain in not very important because we used an accelerated version of back-propagation. The main advantage of our method happens to be a better generalization performance for all three classification tasks. The test error rates obtained with the WIDA method were always significantly lower than those obtained with random initialization (and, to a lesser extent, with the prototype method).

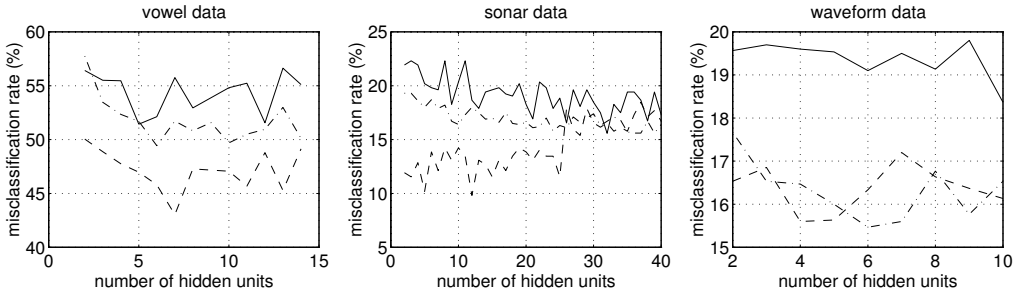

Figure 2: Mean test misclassification rate as a function of$n$ (averages over 10 trials). — : random; - - : WIDA; -.- : prototype method.

4 Influence of dimensionality and network size

The influence of dimensionality and number of weights on generalization performance was studied experimentally using a set of discrimination tasks similar to that used in [8]. Each task consists in discriminating between two multivariate Gaussian classes. Both classes have identity covariance matrix. The class mean vectors are $\boldsymbol{m}_1 = (2, 0, \ldots, 0)$ and $\boldsymbol{m}_2 = -\boldsymbol{m}_1$. This parameterization allows to keep the Mahalanobis distance, and hence the theoretical Bayes error rate, to constant values. Training sets of 120 samples (60 in each class) and test sets of 4000 samples (2000 in each class) were randomly generated.

The two initialization procedures tested were the WIDA method and random initialization. The number n of hidden units was varied from 2 to 10, and the data dimension d from 10 to 100 with a step of 10. For each of the $2 \times 9 \times 10$ configurations, the learning algorithm was run 10 times. The mean misclassification error rates were computed over the 10 trials. Figure 3 shows the obtained mean misclassification rates with 95% confidence intervals as a function of d and n.

As shown in Figure 3, the generalization performance of randomly initialized networks degrades for large values of d and n. This dependency of test error rate on the number of parameters to be estimated is well-known in the Pattern Recognition and Neural Network literature as the "peaking phenomenon" [8]. This phenomenon happens to be less important, in this case, when the initial weights are determined by discriminant analysis. The rate of increase of test error rate as a function of d is smaller, and practically

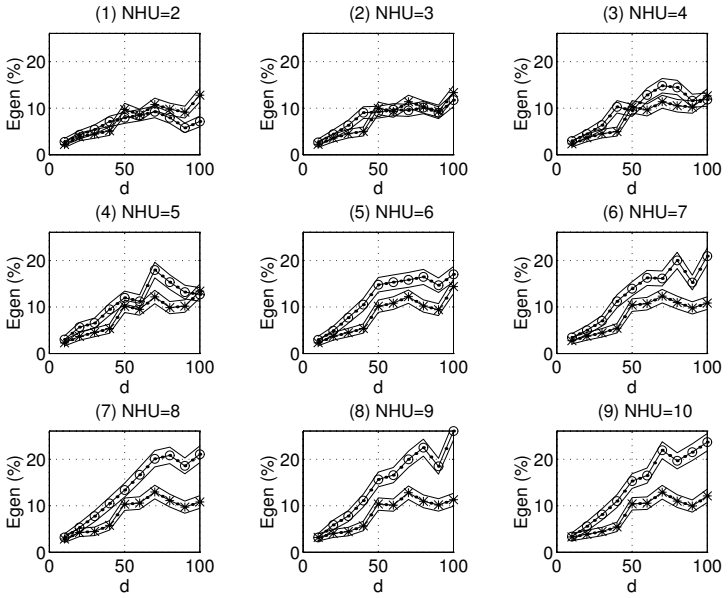

Figure 3: Mean test misclassification rate and 95 % confidence interval as a function of data dimension and number of hidden units (averages over 10 trials). -*- : WIDA initialization, -o- : random initialization, d = data dimension, Egen = generalization error, NHU = number of hidden units (n).

independent from n for $2 \leq n \leq 10$.

This finding can be interpreted by remarking that the WIDA method provides the learning algorithm with prior information concerning the data structure, in the form of discriminant axes. This allows to search only a certain region of weight space, in which weight vectors lead to relatively "simple" discrimination boundaries. In that sense, careful initialization can be seen as performing some kind of regularization. This is consistent with the theoretical and experimental analysis performed by Raudys [8] in the case of linear classifiers, showing that suitable selection of initial weights may cancel the influence of dimensionality on expected probability of misclassification.

5 Conclusion

A new weight initialization procedure for multilayer perceptrons has been presented. This procedure consists in using class-separability preserving feature vectors as the initial hidden layer weights. Biases and output weights are then optimized separately, before fine tuning of all network parameters is performed by a standard back-propagation algorithm. This scheme has been applied to several real-world and artificial discrimination tasks, and has been shown to yield lower generalization error as compared to random initialization and (to a lesser extent) to the procedure proposed in [2]. Experimental results also suggest that the introduction of prior knowledge about the data structure in the form of discriminant vectors reduces the harmful effect of excessive parameters on the expected probability of misclassification. Our current work aims at combining this initialization procedure with a constructive training algorithm.

References

[1] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1984.

[2] T. Denœux and R. Lengellé. Initializing back-propagation networks with prototypes. *Neural Networks*, 6(3):351–363, 1993.

[3] K. Fukunaga. *Introduction to statistical pattern recognition.* Electrical Science. 2nd. edition, Academic Press, 1990.

[4] P. Gallinari, S. Thiria, F. Badran, and F. Fogelman-Soulie. On the relations between discriminant analysis and multilayer perceptrons. *Neural Networks*, 4:349–360, 1991.

[5] R. Lengellé and T. Denœux. Optimizing multilayer networks layer per layer without back-propagation. In I. Aleksander and J. Taylor, editors, *Artificial Neural Networks II*, pages 995–998. North-Holland, Amsterdam, 1992.

[6] R. Lengellé and T. Denœux. Training MLPs layer by layer using an objective function for internal representations. *Neural Networks (to appear)*, 1995.

[7] P. M. Murphy and D. W. Aha. *UCI Repository of machine learning databases [Machine-readable data repository].* University of California, Department of Information and Computer Science., Irvine, CA, 1994.

[8] Raudys S. Why do multilayer perceptrons have favorable small sample properties ? In E. S. Gelsema and L. N. Kanal, editors, *Pattern Recognition in Practice IV*, pages 287–298, Amsterdam, 1994. Elsevier.