Initialization of adaptive parameters in density networks

Włodzisław Duch, Rafał Adamczak, and Norbert Jankowski
Department of Computer Methods, Nicholas Copernicus University,
Grudziądzka 5, 87-100 Toruń, Poland.
E-mail: duch,raad,norbert@phys.uni.torun.pl

Abstract
Initialization of adaptive parameters in neural networks is of crucial importance to the speed of convergence of the learning procedure. Methods of initialization for the density networks are reviewed and two new methods, based on decision trees and dendrograms, presented. These two methods were applied in the Feature Space Mapping framework to artificial and real world datasets. Results show superiority of the dendrogram-based method including rotation.

I. Introduction

LEARNING in neural networks usually requires minimization of a cost function. Finding global minimum of a nonlinear function with many parameters is an NP-hard problem [1]. Good initialization of adaptive parameters may enable finding solutions to complex, real-world problems and may significantly decrease learning time. In this paper methods of initialization of the density estimation networks are discussed. Such networks usually employ localized transfer functions. For example, popular radial basis function (RBF) networks are frequently based on Gaussian functions. Architectures of density networks use almost exclusively single hidden layer, although a modular structure in which several of such one-layer networks cooperate, may also be used.

In the next section methods of initialization of density network parameters are discussed and new methods proposed. Using these methods for initialization of the Feature Space Mapping (FSM) network [2] we have obtained significant improvements both in the speed of convergence of learning procedures, accuracy of classification, and reduction of complexity of the network. These results are presented in the third section. Short discussion closes this paper.

II. Initialization of density network parameters

In the density networks centers of clusters and their dispersions should be determined. In RBF networks centers $\mathbf{C}_i$ are frequently selected randomly, with some fixed number of nodes K and initial dispersion $\sigma = d/\sqrt{2K}$ determined by the largest distance d between the centers. Once centers are selected weights are easily found by pseudoinverse or SVD solution of linear equations [1]. Unfortunately the use of such data-dependent parameters in density networks precludes the use of simple priors for probability distributions and thus makes Bayesian formulation of the learning algorithm [3] difficult.

There are many possibilities to select good centers: randomly selected prototypes may be improved using one of the learning vector quantization (LVQ) procedures [4]. Basically all LVQ methods are based on a self-organizing algorithm: $\mathbf{C}_i^{new} = \mathbf{C}_i^{old} + \eta(\mathbf{X}_k - \mathbf{C}_i^{old})$, where the vector $\mathbf{X}_k$ is drawn randomly from the training dataset. Some of these methods become quite sophisticated, with dynamical adjustment of learning rate η, update of several centers in some (dynamically defined and weighted) neighborhood of $\mathbf{X}$ (like in Self-Organized Mappings) and many other improvements [5]. After determination of centers initial dispersions are calculated using distances to the closest center belonging to different class. Tarassenko and Roberts [6] proposed probabilistic algorithm for determination of centers and dispersions in RBF networks. All such methods are computationally rather intensive, similar to the full learning and classification methods in their own right.

Constructive approaches to density networks, such as the Resource Allocation Network (RAN) [7], or function estimation approach [8], initialize a single unit only. In FSM [2] a few initial units are created using techniques based on decision trees, dendrograms and the variable resolution scale for data filtering. Transfer functions used in FSM [9], [10] include rectangular, biradial and Gaussian functions. Except for centers and dispersions parametrization of rotations is also desired. Rotation matrix in N dimensions theoretically needs only $N-1$ angles, but in practice it is very difficult to specify rotation parameters (we have found an interesting solution to this problem presented in [10]). Calculation of activation with full rotation matrix requires N^2 operations for matrix multiplication. Unconstrained transformation matrices are even worse since the number of adaptive parameters is of the order of N^2 per node, which for a large N is quite impractical.

We have followed another approach, which we also recommend for RBF and other density networks. Initialization procedure provides us with clusters at fixed angles. For skewed distribution of data, adaptation of positions and dispersions of such clusters allows for high accuracy with much smaller number of nodes. The total complexity of the model (measured by the number of network nodes) is restricted in the initialization step by assuming the maximum number of clusters allowed. First the data vectors are standardized to obtain zero mean and standard deviations:

$$\bar{X}_i = \frac{1}{N_v}\sum_{j=1}^{N_v} X_i^{(j)}; \quad \tilde{X}_i^{(j)} = (X_i^{(j)} - \bar{X}_i)/\sigma_i \tag{1}$$

$$\sigma_i = \sqrt{\frac{1}{N_v - 1}\sum_{j=1}^{N_v}(X_i^{(j)} - \bar{X}_i)^2} \tag{2}$$

Data resolution: If the number of data vectors is large the data is first filtered decreasing its resolution. Calculations are frequently done with 4 byte representation of floating numbers giving data resolution of $r = 10^{-7}$ for data renormalized to the $[0, 1]$ interval. For the resolution r all data in an interval $[a, a + r)$ are equivalent to one value. All data values X_i are converted first to the maximum range of $[0, 2^{32}]$ using the formula:

$$X_r = C(X - X_{\min}); \quad C = \frac{2^{32}}{(X_{\max} - X_{\min})} \tag{3}$$

To obtain the resolution of $r = 1/2^k$ we have to leave k significant bits in X_r. This is done by shifting the bits, for example converting back to original representation:

$$X(k) = X_{\min} + \frac{1}{C}\left[\text{Round}\left(\frac{X_r}{2^{33-k}}\right) \cdot 2^{33-k}\right] \tag{4}$$

For $k = 1$ all data $\mathbf{X}$ are mapped to two clusters and for $k = 32$ original data is recreated. The highest cluster corresponds to $X_r = 11\ldots1$ (in the binary representation) and the lowest to $X_r = 00\ldots0$, setting the scale for the largest distance. The smallest distance is defined by the resolution of the data. The number of significant bits is increased until the number of clusters reaches the maximum allowed. Since this is done at a relatively small-grain scale, just to find reduced set of vectors for clusterization, it is a good approximation, requiring very simple processing of data.

Decision trees: At a resolution of k bins the number of data vectors in each bin is counted. All adjacent bins with non-zero vectors in them form a cluster in one dimension, defining positions and initial cluster sizes. The resolution is increased up to the noise level (defined in a global way for the input data by the user) or until the total number of initial clusters reaches allowed value. Suppose that the cluster starting from bin i takes l units of size s. Then position and size for the x-component in the original scale is:

$$C_x = X_{min} + s(i + \frac{l}{2}); \quad \sigma_x = \frac{1}{2}ls; \quad s = |X_{max} - X_{min}|/k \qquad (5)$$

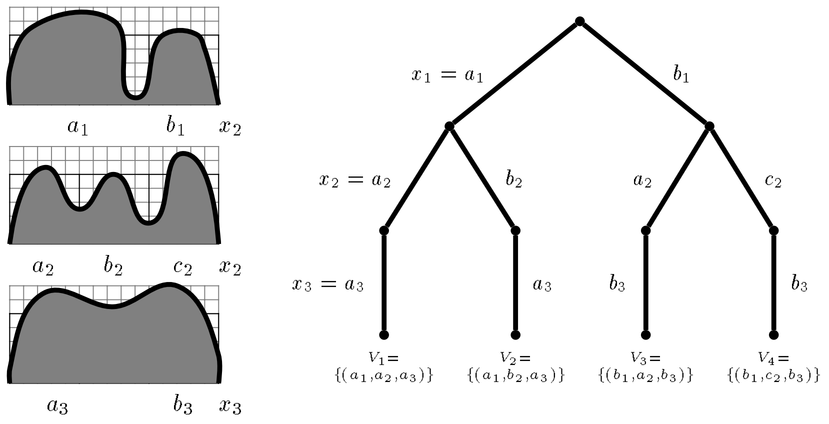
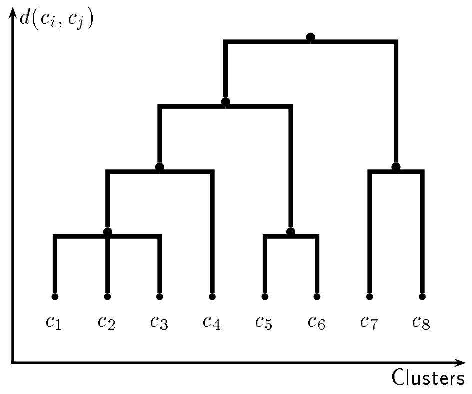


Fig. 1. Illustration of the decision tree method of initial clusterization.

Each of these one-dimensional clusters may be a projection of many separate N-dimensional clusters. In the initialization phase vectors are read and a loop over all dimensions started. In the first dimension the vector will belong to some cluster C_1^i, in the second dimension to C_2^j (this already defines a two-dimensional cluster) and in the N-th dimension to C_N^k. Each N-dimensional cluster is characterized by a chain of lower-dimensional clusters, $C_1^i \rightarrow C_2^j \rightarrow \ldots C_N^k \rightarrow$, therefore this initialization procedure creates a decision tree. In the leafs of this tree the number of vectors belonging to the N-dimensional cluster are counted (for some classifiers this information is used to set the "mass" of clusters). If the number of clusters at any level exceeds the maximum allowed the search is terminated and results from more coarse-grained initialization are taken. Finally, once all clusters are found distances between them are computed and those clusters that are closer than a specified threshold are merged into one. This step allows to take into account skewed distributions that are not recognized as single clusters by the search algorithm that effectively divides the input space into cuboids. Positions and sizes of cuboids belonging to these clusters are memorized as numerical values.

This algorithm improves FSM classification in comparison with purely constructive approach [11]. In highly dimensional spaces even a small increase in the number of data bins in each dimension creates many clusters. To avoid it one should increase resolution in selected dimensions first, using histograms to determine which dimensions (features) allow for the best discrimination. For some datasets this may be difficult, especially if the input space has high dimension and many input features are dominated by noise. Skewed distribution of data clusters in the input space create problems in finding good division of features. Decision-tree based initialization method is computationally inexpensive but in some cases difficult to apply.



, Illustration of the decision tree method of initial clusterization and dendrogram method of clusterization.

Dendrograms: In decision-tree algorithm data resolution is progressively increased. Better results are sometimes obtained using algorithm in which data resolution is progressively decreased, using a modified dendrogram-type method. Initially each training vector is a separate cluster. To determine which vectors $\mathbf{X}_i, \mathbf{X}_j$ should be merged distance matrix $d(\mathbf{X}_i, \mathbf{X}_j)$ is computed. Minimal distance is found in the distance matrix and the closest clusters replaced by an averaged cluster. This procedure is repeated until the number of clusters becomes sufficiently small or the distances between remaining clusters become larger than assumed threshold. To avoid large matrices and high computational costs for large datasets again filtering of data vectors is used to create groups of vectors replaced by averaged cluster positions.

Rotations: Once all vectors are assigned to clusters one can build a covariance matrix for each cluster, diagonalize it and use its eigenvectors to set up the initial rotation and dispersion parameters. To find the rotation angles of the main axis of the cluster a straight line $X_i = a_i X_1$ is fitted using all vectors $\mathbf{X}^{(k)}$ belonging to the cluster $\mathbf{C}$:

$$a_i = \frac{\sum_{k\in C} X_1^k X_i^k}{\sum_{k\in C} X_i^k X_i^k} \qquad (6)$$

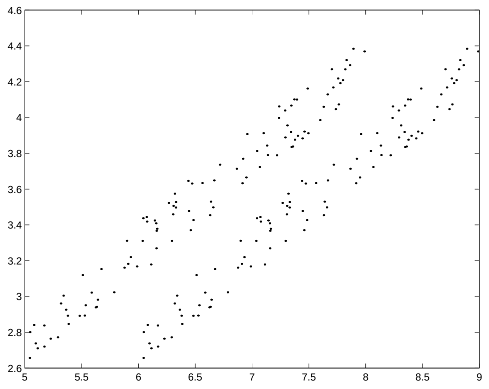
These coefficients are equal to tangens of angle between the main cluster axis and X_1 axis. To find dispersions vectors belonging to the cluster are rotated in the (X_1, X_2), (X_1, X_3), .. (X_1, X_N) plane using 2×2 transformations:

$$\begin{pmatrix} X_1^{k'} \\ X_i^{k'} \end{pmatrix} = \begin{pmatrix} \cos(-\arctan(a_k) & -\sin(-\arctan(a_k) \\ \sin(-\arctan(a_k) & +\cos(-\arctan(a_k) \end{pmatrix} \begin{pmatrix} X_1^k \\ X_i^k \end{pmatrix} \qquad (7)$$

This transformation requires only $4(N-1)$ multiplications. Maximum and minimum values of X_i^k define dispersions in each dimension. We have used the dendrogram approach with rotations for initialization of FSM obtaining very good results.

III. Results of computer simulations

Consider an artificial two-dimensional data containing two skewed clusters (Fig.2). Using rectangular basis functions with decision tree initialization 69.5% of points were correctly classified – this is a difficult case for decision tree method (histograms are almost uniform). After initialization using dendrogram method without rotations the percentage of correctly classified points increased to 88%. Adding rotations and determination of cluster sizes all data points were correctly classified. In all cases two clusters were created.



, Artificial data with two skewed clusters used in the preliminary experiments.

Results of initial clusterization for 5 datasets are presented in Table I. Classification errors reported here are of course reduced further by optimization of FSM network parameters, but it is interesting to note that these results are frequently already of rather high quality. Except for galaxies all other data was obtained from the UCI repository [13]. Iris dataset contains 150 cases in 3 classes. After initialization with Gaussian functions including rotations only 4 classification errors are made (97.3% accuracy), which is a better results than many classifiers give on this dataset. For vowel 990 samples of 10-dimensional vectors should be divided into 11 classes. 3 clusters give already 77% accuracy on the training and 50% accuracy on the test data, while the best classification results (k-NN) give only 56% on the test data. Using rotations and Gaussian functions accuracy on the training set is increased to 84.4 %. The StatLog version of the **satellite image data** generated from Landsat Multi-Spectral Scanner [13] has a total of 6435 vectors with 36 attributes each (0 to 255 range) divided into 6 classes. Initialization with rotated clusters gave 80% accuracy, while best RBF or MLP solutions give 89%. The DNA database, with 3186 vectors, has 180 attributes and 3 classes. In this case k-NN gives 85% accuracy while our initialization gave 83%. The data for galaxies were obtained from the ESO-LV catalog (all details of this dataset are described in [14]) and contains 5217 cases, 13 features and two classes. MLP gives 90% accuracy here while our best initialization gives 86%.

TABLE I
Classification results (% of errors) after initialization of FSM network using decision tree method (DT), dendrograms without cluster rotation (D) and with rotation (DR). Results were obtained using rectangular functions (+R) and Gaussian functions (+G). Number of created clusters is given in parenthesis.

Dataset	DT+R	D+R	DR+R	DT+G	D+G	DR+G
Iris	65 (4)	80 (4)	87	65	95	97.3
Vovel	9 (31)	53 (22)	73	1	68	84.4
Satimage	45 (14)	23 (6)	24	14	77	80.0
DNA	0 (3)	9 (3)	19	0	83	76.1
Galaxies	36 (4)	14 (2)	12	36	86	84.4

IV. Summary and discussion

Two new methods of initialization of density networks have been proposed, one based on decision trees and the other on dendrograms. These methods are used for initialization of FSM network with rectangular, biradial and Gaussian functions in applications to classification problems and to logical rule extraction [12]. FSM constructive algorithm starts from a number of prototype clusters adding more networks nodes as and where required, but networks created from scratch are not so efficient as networks developed from initial structures found by clusterization. Introduction of rotations for some datasets (but not for all) significantly reduces the classification error. For many classification problems FSM network after initialization required little extra tuning. Final network structures became more compact than those obtained with the straightforward constructive algorithm and the classification accuracy was higher. Initialization methods presented in this paper may also be used in other type of density networks, such as RBF or RAN networks [1].

Acknowledgments: Support by the Polish Committee for Scientific Research, grant 8T11F 00308, is gratefully acknowledged.

References

[1] C. Bishop, Neural networks for pattern recognition. (Clarendon Press, Oxford 1995)
[2] W. Duch, G.H.F. Diercksen, Feature Space Mapping as a universal adaptive system, *Computer Physics Communications* 87 (1995) 341–371
[3] D.J. MacKay, A practical Bayesian framework for backpropagation networks, *Neural Comput.* 4 (1992) 448-472
[4] T. Kohonen, Self-organizing maps. (Heidelberg Berlin, Springer-Verlag 1995).
[5] C. Chinrungrueng, C.H. Sequin, Optimal adaptive K-means algorithm with dynamics adjustment of learning rate, *Trans. Neural Net.* 6 (1995) 157–169

[6] L. Tarassenko, L. Roberts, Supervised and unsupervised learning in radial basis function classifiers. *IEEE Vision and Image Signal Proc.* 141 (1994) 210–216

[7] J. Platt, A resource-allocating network for function interpolation. *Neural Comput.* 3 (1991) 213-225

[8] V. Kadirkamanathan, M. Niranjan, A function estimation approach to sequential learning with neural networks. *Neural Comput.* 5 (1993) 954-975

[9] W. Duch, N. Jankowski, New Neural Transfer Functions, *J. of Applied Mathematics and Computer Science* (submitted 1997)

[10] R. Adamczak, W. Duch and N. Jankowski, New developments in the Feature Space Mapping model, *Proc. third conference on neural networks and their applications*, Kule, Poland, 14-18.10.1997 (this volume)

[11] W. Duch, R. Adamczak, N. Jankowski, Improved memory-based classification, in: *Proc. EANN'96*, London, 17-19.06.1996 (ed. A.B. Bulsari, S. Kallio, D. Tsaptsinos), pp. 447-450

[12] W. Duch, R. Adamczak, K. Grąbczewski, Extraction of crisp logical rules using constrained backpropagation networks. *ICANN'97*, Houston, 9-12.6.1997 (in print); Logical rules for classification of medical data using ontogenic neural algorithm, *EANN'97*, Stockholm, 16-18.06.1997 (in print)

[13] C.J. Mertz, P.M. Murphy, UCI repository, http://www.ics.uci.edu/pub/machine-learning-databases.

[14] O. Lahav, A. Naim, L.Sodre Jr. and M. C. Storrie-Lombardi, Neural computation as a tool for galaxy classification: methods and examples, Institute of Astronomy, Cambridge, Technical report CB3 OHA (1995)