APPENDIX: Mathematical Description of the aiNet

Introduction

Due to the growth interest on the aiNet application, and many questions which arise from academic nature of user's research work, we prepared an appendix to the original User's Manual of the aiNet application, version 1.0. It begins with the explanation of theoretical background, where the derivation of the build-in method is presented. This is followed by simple explanation how and why is the aiNet an artificial neural network (ANN). The aiNet neuron structure is graphically presented, where it becomes clear that aiNet is an ANN. Finally, the list of references is given.

According to the authors' knowledge about ANN, their conviction, and also to the known existing tools (AN nets), the aiNet is a kind of ANN. It is based on a self-organizing system, called neural network-like system, presented by Grabec (see /GRAB90a/ and /GRAB90b/). It is similar to the method of the nearest neighbor, Learning Vector Quatization Network /KOHO88/, and also to the probabilistic neural network, proposed by Specht /SPEC88/. All of the above mentioned methods have the same basement and similar rules for describing various phenomena. On the other hand, they are different compared to each other in the sense like various paradigms used in backpropagation ANNs differ from each other, or various types of ANNs differ from each other.

One important thing that should be mentioned is how ANNs are related to the statistical methods. (see /PREC95/). Some of them have a little to do with statistics, some of them would not be considered as statistical methods, but most neural networks that can learn to generalize effectively from noisy data are similar or identical to statistical methods. For example, probabilistic neural nets are identical to kernel discriminant analysis. On the other hand, Kohonen's self organizing maps have no close relativities in the existing statistical literature, but self-organization of neurons, proposed by Grabec /GRAB90a/ is very similar to the Kohonen's self organization process and is based on statistical principles. Also, feedforward nets are a subset of the class of non-linear regression and discriminant models. Statisticians have studied the properties of this general class but had not considered the specific case of feedforward ANN before such networks became popular. It is obvious that it is very hard (or even impossible) to find a widely accepted definition which would be able to classify between ANNs and statistical methods. Nevertheless, the most important thing is that ANNs allow different view on problems which can not be solved by (exact) statistical methods due to their theoretical limitations.

And how is the aiNet related to the ANNs? According to one of the "soft" definitions of ANNs, the ANNs are networks of many simple processors "units", each possibly having a local memory. The processors are connected by unidirectional communication channels "connections", which carry numeric data. The units operate only on their local data and on the inputs they receive via connections. As will be shown the aiNet can be presented as a kind of neural network according to the above definition.

And how is the aiNet related to the statistics? It is actually derived from the statistical environment, which can be clearly seen from the derivation that follows.

Theoretical backgrounds of the aiNet

Basic principles and derivation

In formulating a modeler of a phenomena we assume that one partial observation of a phenomenon can be described by *L* variables m_i. One such observation can be written then as a vector, so called model vector:

$$\mathbf{mv} = (mv_1, mv_2, \ldots, mv_L) \qquad /1/.$$

To describe the whole phenomenon properly one needs to collect data from many observations of the same phenomenon. The complete phenomenon is then described by a sample of the *N* measurements (data base or knowledge base) that are described by a finite set of model vectors:

$$\text{model} = \left\{ \mathbf{mv}_1, \mathbf{mv}_2, \ldots, \mathbf{mv}_N \right\} \qquad /2/.$$

Expression /2/ can be written in matrix form as:

mv$_1$	=	m$_{11}$	m$_{12}$	**...**	m$_{1L}$
mv$_2$	=	m$_{21}$	m$_{22}$	**...**	m$_{2L}$
...		**...**			**...**
...		**...**			**...**
mv$_N$	=	m$_{N1}$	m$_{N2}$	**...**	m$_{NL}$

/3/.

Let us assume that each model vector consist of two partial vectors: first vector represents the input variables of the phenomenon and the other vector the output variables. We will label the first one as vector **P**, and the other as a vector **Q**:

$$\begin{aligned} \mathbf{P} &= (m_1, m_2, \ldots, m_M, \#) \ \ldots \text{ input variables} \\ \mathbf{Q} &= (\#, m_{M+1}, m_{M+2}, m_L) \ \ldots \text{ output variables} \end{aligned} \qquad /4/.$$

Concatenation of both vectors gives the original model vector. This can be written as:

$$\mathbf{mv} = \mathbf{P} \oplus \mathbf{Q} = (m_1, m_2, \ldots, m_M, m_{M+1}, \ldots, m_L) \qquad /5/,$$

or in matrix form:

$\mathbf{mv}_1$	=	m_{11}	m_{12}	**...**	m_{1M}	$m_{1,M+1}$	**...**	m_{1L}
$\mathbf{mv}_2$	=	m_{21}	m_{22}	**...**	m_{2M}	$m_{2,M+1}$	**...**	m_{2L}
...		**...**						**...**
...		**...**						**...**
$\mathbf{mv}_N$	=	m_{N1}	m_{N2}	**...**	m_{NM}	$m_{N,M+1}$	**...**	m_{NL}

/6/.

Shadowed part belongs to the partial vector **Q**.

Vector **mv** (herein after labeled as **m**) is treated as a random variable, the properties of which are characterized by the density of the joint probability function:

$$f(\mathbf{m}) = \frac{d^L P}{dm_1\, dm_2 \ldots dm_L} \qquad /7/.$$

Let us assume that partial information about the phenomenon is given by truncated vector **P**. The problem is how to estimate the remaining (L-M) unknown components of the complementary vector **Q** on the basis of known information given by vector **P**. Because of the random nature of the sample vector **m**, there exist many possible realizations of vector **Q** which correspond to the known vector **P**. As the best solution we choose the estimator

$$\hat{\mathbf{Q}} = \hat{\mathbf{q}}(\mathbf{P}) \cong \mathbf{Q} \qquad /8/,$$

which minimizes the mean-square error, expressed as

$$\begin{aligned} D = E\left[\mathbf{Q} - \hat{\mathbf{q}}(\mathbf{P})\right]^2 &= \int \left[\mathbf{Q} - \hat{\mathbf{q}}(\mathbf{P})\right]^2 dP(\mathbf{m}) \\ &= \int_{\mathbf{Q}}\int_{\mathbf{P}} \left[\mathbf{Q} - \hat{\mathbf{q}}(\mathbf{P})\right]^2 f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{P}\, d\mathbf{Q} \end{aligned} \qquad /9/.$$

Where *dP(***m***)* is defined as (see also /7/):

$$dP(\mathbf{m}) = f(\mathbf{m})\, dm_1\, dm_2 \ldots dm_L = f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{P}\, d\mathbf{Q} \qquad /10/.$$

The unknown function $\hat{\mathbf{q}}(\mathbf{P})$ can be obtained by the standard variational procedure:

$$\begin{aligned} \delta D = &-2 \int_{\mathbf{Q}}\int_{\mathbf{P}} \left[\mathbf{Q} - \hat{\mathbf{q}}(\mathbf{P})\right] \delta\hat{\mathbf{q}}(\mathbf{P})\, f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{P}\, d\mathbf{Q} \\ &+ \int_{\mathbf{Q}}\int_{\mathbf{P}} \left[\mathbf{Q} - \hat{\mathbf{q}}(\mathbf{P})\right]^2 \delta f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{P}\, d\mathbf{Q} \end{aligned} \qquad /11/,$$

from where, while we assume $f(\mathbf{P} \oplus \mathbf{Q})$ is known function and therefore

$$\delta\, f(\mathbf{P} \oplus \mathbf{Q}) = 0 \qquad /12/$$

and while we know that minimum will be reached if $\delta\, D = 0$ we obtain:

$$\int_{\mathbf{Q}} \int_{\mathbf{P}} \left[\mathbf{Q} - \hat{\mathbf{q}}(\mathbf{P})\right] \delta\, \hat{\mathbf{q}}(\mathbf{P})\, f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{P}\, d\mathbf{Q} = 0 \qquad /13/.$$

We can rewrite the expression /13/ as:

$$\int_{\mathbf{P}} \delta\, \hat{\mathbf{q}}(\mathbf{P})\, d\mathbf{P} \int_{\mathbf{Q}} \left[\mathbf{Q} - \hat{\mathbf{q}}(\mathbf{P})\right] f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{Q} \qquad /14/.$$

While $\delta\, \hat{\mathbf{q}}(\mathbf{P}) \neq 0$, we obtain

$$\int_{\mathbf{Q}} \left[\mathbf{Q} - \hat{\mathbf{q}}(\mathbf{P})\right] f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{Q} = 0 \qquad /15/,$$

and finally

$$\int_{\mathbf{Q}} \mathbf{Q}\, f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{Q} = \int_{\mathbf{Q}} \hat{\mathbf{q}}(\mathbf{P})\, f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{Q} \qquad /16/.$$

This leads to the expression

$$\hat{\mathbf{q}}(\mathbf{P}) = \frac{\int_{\mathbf{Q}} \mathbf{Q}\, f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{Q}}{\int_{\mathbf{Q}} f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{Q}} \qquad /17/.$$

Conditional probability distribution is defined by

$$f(\mathbf{Q}|\mathbf{P}) = \frac{f(\mathbf{P} \oplus \mathbf{Q})}{f(\mathbf{P})} = \frac{f(\mathbf{P} \oplus \mathbf{Q})}{\int_{\mathbf{Q}} f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{Q}} \qquad /18/.$$

Then the expression /17/ can be written as

$$\hat{\mathbf{q}}(\mathbf{P}) = \int_{\mathbf{Q}} \mathbf{Q}\, f(\mathbf{Q}|\mathbf{P})\, d\mathbf{Q} \qquad /19/.$$

The expression /19/ is the optimal estimator of the conditional average.

In the experimental description of a physical phenomenon, the probability distribution function is not known analytically and therefore must be statistically estimated from observed data. Set of N independent but statistically equivalent partial observations of the phenomenon is defined as model vector data base. When a statistical weight 1/N is assigned to each observation the joint probability density can be expressed as

$$f(\mathbf{m}) = \frac{1}{N}\sum_{n=1}^{N}\delta(\mathbf{m} - \mathbf{m}_n) \qquad /20/,$$

where

$$\begin{aligned} \delta(\mathbf{m}) &= \prod_{i=1}^{L}\delta(m_i) \quad \ldots \text{ vector form} \\ \delta(x) &= \begin{cases} 1; \text{ if } x = 0 \\ 0; \text{ if } x \neq 0 \end{cases} \quad \ldots \text{ scalar form} \end{aligned} \qquad /21/,$$

Note: Please, do not confuse the δ function - expressions /20/, /21/ and expressions in further derivation - with the variational operator δ which was used in /11/, /12/, /13/, /14/.

The density of marginal probability distribution function in M-dimensional subspace is then given by

$$f(\mathbf{P}) = \frac{1}{N}\sum_{n=1}^{N}\delta(\mathbf{P} - \mathbf{P}_n) \qquad /22/,$$

where

$$\mathbf{P}_n = (m_{n1}, m_{n2}, \ldots, m_{nM}) \quad \text{and} \quad \delta(\mathbf{P}) = \prod_{i=1}^{M}\delta(m_i) \qquad /23/.$$

*f(***P***)* can be approximated by

$$f(\mathbf{P}) = \int_{\mathbf{Q}} f(\mathbf{P} \oplus \mathbf{Q})\, d\mathbf{Q} = \frac{1}{N}\sum_{i=1}^{N}\delta(\mathbf{P} - \mathbf{P}_i) \qquad /24/.$$

The conditional probability can then be represented by

$$f(\mathbf{Q}|\mathbf{P}) = \frac{\sum_{n=1}^{N}\delta(\mathbf{Q} - \mathbf{Q}_n)\,\delta(\mathbf{P} - \mathbf{P}_n)}{\sum_{n=1}^{N}\delta(\mathbf{P} - \mathbf{P}_n)} \qquad /25/.$$

The simple formula /25/ cannot be applied directly because of the δ function. We try to approximate the δ function by a smooth, regular function. Among various functions, the Gaussian function seems to be the most appropriate:

$$\delta(\mathbf{m}) \cong \gamma(\mathbf{m}) = \alpha\, e^{\frac{\|\mathbf{m}\|^2}{2\sigma^2}} = \alpha\, e^{\frac{-\sum_{i=1}^{N} m_i^2}{2\sigma^2}} \qquad /26/.$$

α in the expression is a constant, while the part $2\sigma^2$ is directly correlated to the so called penalty coefficient. How to choose the right value is another question and can be solved in the iteration procedure, as is done in the aiNet.

The conditional average is approximated by

$$f(\mathbf{Q}|\mathbf{P}) \cong \frac{\sum_{i=1}^{N} \delta(\mathbf{Q} - \mathbf{Q}_i)\,\gamma(\mathbf{P} - \mathbf{P}_i)}{\sum_{i=1}^{N} \gamma(\mathbf{P} - \mathbf{P}_i)} \qquad /27/.$$

When this function is inserted into expression /18/ we obtain

$$\begin{aligned}
\hat{\mathbf{q}}(\mathbf{P}) &= \int_{\mathbf{Q}} \mathbf{Q} \frac{\sum_{i=1}^{N} \delta(\mathbf{Q} - \mathbf{Q}_i)\gamma(\mathbf{P} - \mathbf{P}_i)}{\sum_{i=1}^{N} \gamma(\mathbf{P} - \mathbf{P}_i)}\, d\mathbf{Q} \\
&\cong \sum_{j=1}^{N} \mathbf{Q}_j \frac{\sum_{i=1}^{N} \delta(\mathbf{Q}_j - \mathbf{Q}_i)\gamma(\mathbf{P} - \mathbf{P}_i)}{\sum_{i=1}^{N} \gamma(\mathbf{P} - \mathbf{P}_i)} \qquad /28/. \\
&= \frac{1}{\sum_{i=1}^{N} \gamma(\mathbf{P} - \mathbf{P}_i)} \sum_{j=1}^{N} \mathbf{Q}_j \sum_{i=1}^{N} \delta(\mathbf{Q}_j - \mathbf{Q}_i)\gamma(\mathbf{P} - \mathbf{P}_i)
\end{aligned}$$

We can see that

$$\delta(\mathbf{Q}_j - \mathbf{Q}_i) = \delta_{ij} \quad \text{where} \quad \delta_{ij} = \begin{cases} 1; i = j \\ 0; i \neq j \end{cases}$$

and therefore /30/.

$$\sum_{j=1}^{N} \delta_{ij}\, \gamma(\mathbf{P} - \mathbf{P}_j) = \gamma(\mathbf{P} - \mathbf{P}_i)$$

Finally we obtain the following expression:

$$\hat{\mathbf{q}}(\mathbf{P}) = \frac{1}{\sum_{j=1}^{N} \gamma\left(\mathbf{P} - \mathbf{P}_j\right)} \sum_{i=1}^{N} \mathbf{Q}_i \, \gamma\left(\mathbf{P} - \mathbf{P}_i\right) = \sum_{i=1}^{n} \mathbf{Q}_i \, c_i \qquad /31/,$$

where

$$c_j = \frac{\gamma\left(\mathbf{P} - \mathbf{P}_j\right)}{\sum_{i=1}^{N} \gamma\left(\mathbf{P} - \mathbf{P}_i\right)} \qquad /32/.$$

Now we have the final expression for the optimal estimator. Presented procedure corresponds to an associative recognition of some unknown properties of the phenomenon on the basis of incomplete observation or experience, obtained by previous complete observation. The derived procedure can be considered as a kind of a primitive intelligence and it is applicable (as it will be shown) to the development of ANNs. In one of such treatments a particular model vector $\mathbf{m}_n$ is characterized as a neuron. When driven by a particular input vector **P**, the neuron is excited as described by the amplitude c_n and contributes to the complete output of the network $\hat{\mathbf{q}}$. It should be noted that in our further explanation, the aiNet will be presented in a more “natural” way, similar to the existing representations of the ANNs.

Graphical presentation of the aiNet

Figure A.1 shows the graphical presentation of the aiNet, and as can be seen it is very similar to the other ANNs. To compare it exactly with other supervised ANNs, we should distinguish between training and prediction phase. In the aiNet is the training phase very quick and corresponds to the presentation of the model vectors (loading the data base or aiNet data file) to the network - aiNet (see also: probabilistic neural network - PNN from NeuralWare, Inc. /NEUR91/). Prediction phase corresponds to the calculation of values of processing elements and calculating the unknown output values of prediction vector (in case of prediction) or output values of model vectors (in case of filtration of verification for determination of penalty coefficient value).

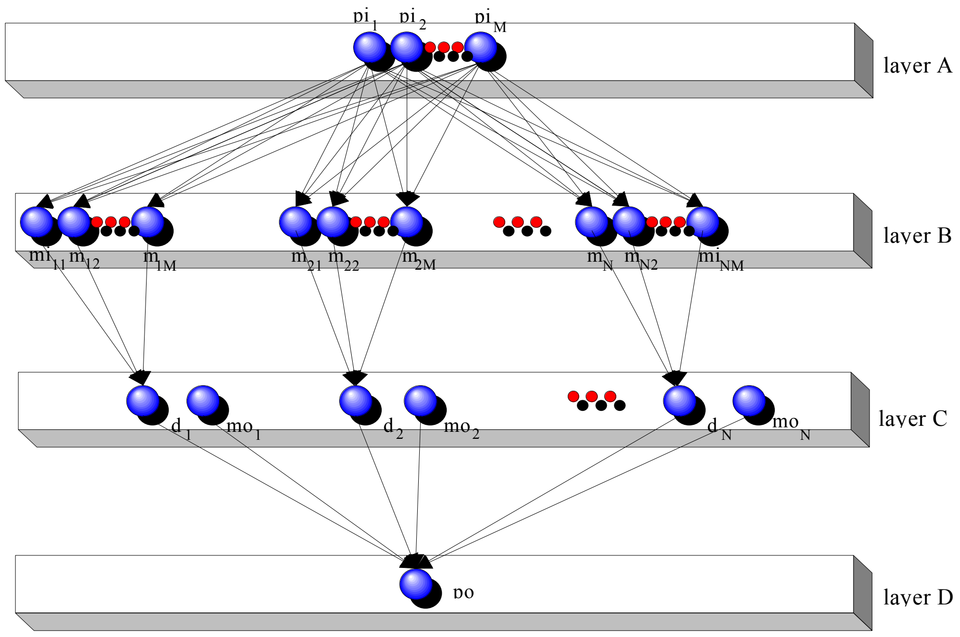

Figure A.1: *Graphical presentation of the aiNet.*

Notations in Figure A.1 have the following meaning:

- p prediction vector,
- m model vector,
- i indicates the neuron, belonging to the input variable,
- o indicates the neuron, belonging to the output variable.
- N number of model vectors,
- M number of input variables of the phenomenon,
- K number of output variables of the phenomenon (K is equal to 1 in presented case, and is omitted),
- pc penalty coefficient

As it was mentioned earlier, training (or learning) the aiNet ANN corresponds to the presentation of the model vectors to the net. The weights on connections are either equal to one or equal to zero. The expression for weight adaptation can be written as:

$$w_{ij} = \overline{w}_{ij} \delta_{kj} ,$$

where $\overline{w}_{ij}$ is equal to 1.0, and δij is defined in /30/.

The following expressions, which describe the aiNet in the feedforward - prediction mode, are based completely on the derived expression /31/. It should be noted that such network has two hidden layers (layer B and layer C). The number of neurons in layer B is equal to the product of the number of all model vectors N and the number of input variables M (N M), while the number of neurons in the layer C is equal 2 times the number of model vectors. For the sake of simplicity, we will assume that we have only one output (unknown) variable.

Network works in prediction mode according to the following scheme:

- *layer A:* value of the neuron: $X_i^A = p_i,$
  transfer function: *linear*
  output value of the neuron: $Y_i^A = f(X_i^A) = X_i^A.$
- *layer B*: value of the neuron: $X_{ij}^B = \sum_{k=1}^{M} \left(Y_k^A - mi_{ij}\right) \delta_{kj}$
  transfer function: *linear*
  output value of the neuron: $Y_{ij}^B = f\left(X_{ij}^B\right) = \left(X_{ij}^B\right)^2$
- *layer C:* value of the neurons <u>type d</u>: $X_i^C = \sum_{j=1}^{M} Y_{ij}^B$
  transfer function: *linear*
  output value of the neuron: $Y_i^C = f\left(X_i^C, pc\right) = e^{\frac{-X_i^C}{pc}}$
  value of the neuron <u>type mo</u>: $\overline{X}_i^C = mo_i,$
  transfer function: *linear*
  output value of the neuron: $\overline{Y}_i^C = f(\overline{X}_i^C) = \overline{X}_i^C = mo_i.$

layer D: value of the neuron: $X^D = \sum_{i=1}^{N} Y_i^C \quad ,$

$$\overline{X}^D = \sum_{i=1}^{N} Y_i^C \, \overline{Y}_i^C$$

transfer function: *linear*
output value of the neuron: $po = Y^D = f\left(X^D, \overline{X}^D\right) = \frac{X^D}{\overline{X}^D}$

Why is the aiNet an ANN?

Finally we can give a simple explanation, why the aiNet **is** an ANN. There are a few necessary characteristics which should be satisfied, if the network is to be classified as an ANN.

- *ANN consists of many simple processors*: this can be observed from Figure A.1!
- *Each processor (unit, neuron) has its own local memory*: this can be deduced from expressions!
- *The processors are connected by unidirectional communication channels, which carry numeric data*: this can be seen from Figure A.1 and from expressions. It should be mentioned, that connections (weights) are not changed; they have values either 0 or 1!
- *The units operate only on their local data and on the inputs they receive via connections*: this can be seen from Figure A.1 and from expressions!
- *While units operate on their local data, ANN can work in parallel mode*: this can be observed from Figure A.1 and from expressions!

As can be deduced, the aiNet presented in the Figure A.1 and described with above expressions, fulfills all demands to be classified as an ANN.

It should be noted here that classic comparison of biological and artificial NN (see the above example) is done on very low level which corresponds to the state when ANNs are simulated on a hardware (and by software, too). On the contrary, the functionality of ANNs can be compared to the human thinking in general, which corresponds to very high level comparison: We collect information from outside world, and then we produce conclusions on the basis of these data using a kind of simple statistical regression. While we are limited in an available conciseness memory space, we make a kind of self-organization mapping, when the number of information becomes to high; each information represents the model vector about the phenomenon and might correspond to the one single neuron (see /GRAB90b/). According to this we have only limited (optimal) number of information of the phenomenon in the conciseness memory at once. But, if we do not have problems with the physical memory space (or hard disk space) on the computers, we do not need the self-organization process, which is usually very complicated iterative (learning) process. Avoiding self-organization process will have consequence only on the time used for the prediction, and never on the quality of the predictions.

References

/GRAB90a/ Grabec, I. **Self-Organization of Neurons Described by the Maximum-Entropy Principle**, Biol. Cybern., 63, pp. 403-409, 1990.

/GRAB90b/ Grabec, I. **Modeling of Natural Phenomena by a Self-Organizing System**, Proc. ECPD NEUROCOMPUTING, Vol. 1, no. 1, 1990.

/GRAB90c/ Grabec, I**., Prediction of a chaotic time series by a self-organizing neural network**, Dynamic Days, Dusseldorf, 1990.

/GRAB91/ Grabec, I. & Sachse, W., **Automatic modeling of physical phenomena: Application to ultrasonic data**, J. Appl. Phsy. **69** (9), 1991.

/GRAB93/ Grabec, I., **Optimization of kernel-type density estimator by the principle of maximal self-consistency**, Neural Parallel & Scientific Computations, **1**, pp. 83-92, 1993

/SPEC88/ Specht, D. F., **Probabilistic Neural Networks for Classification, Mapping or Associative Memory**, ICNN, Conference Proc., 1988.

/KOHO88/ Kohonen, T., et al**., Statistical Pattern Recognition with Neural Networks: Benchmark Studies**, Proceedings of the 2nd Annual IEEE International Conference on Neural Networks, Vol. 1, 1988.

/PREC95/ Prechelt, L., Neural-Net-FAQ, URL: http://wwwipd.ira.uka.de/-prechelt/FAQ/neural-net-faq.html, see **Answers to the Questions** (No. 11, Last Modified: 1995/03/02), 1995.

/NEUR91/ **Neural Computing**, NeuralWare, Inc., 1991.