UvA-DARE (Digital Academic Repository)

Feed-forward neural networks for shower recognition: construction and generalization

H.M.A.Andree (et al), H.; Vermeulen, J.C.

DOI

[10.1016/0168-9002\(94\)01156-7](https://doi.org/10.1016/0168-9002(94)01156-7)

Publication date

1995

Published in

Nuclear Instruments & Methods in Physics Research Section A - Accelerators Spectrometers Detectors and Associated Equipment

[Link to publication](#)

Citation for published version (APA):

H.M.A.Andree (et al), H., & Vermeulen, J. C. (1995). Feed-forward neural networks for shower recognition: construction and generalization. *Nuclear Instruments & Methods in Physics Research Section A - Accelerators Spectrometers Detectors and Associated Equipment*, 355, 589. [https://doi.org/10.1016/0168-9002\(94\)01156-7](https://doi.org/10.1016/0168-9002(94)01156-7)

General rights

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations

If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (<https://dare.uva.nl>)

Feed-forward neural networks for shower recognition: construction and generalization

H.M.A. Andree ^{a,*}, W. Lourens ^a, A. Taal ^a, J.C. Vermeulen ^b

^a Department of Physics and Astronomy, Utrecht University, P.O. Box 80000, 3508 TA Utrecht, Netherlands

^b NIKHEF-H, P.O. Box 41882, 1009 DB Amsterdam, Netherlands

Received 10 March 1994

Abstract

Strictly layered feed-forward neural networks are explored as recognition tools for energy deposition patterns in a calorimeter. This study is motivated by possible applications for on-line event selection. Networks consisting of linear threshold units are generated by a constructive learning algorithm, the Patch algorithm. As a non-constructive counterpart the back-propagation algorithm is applied. This algorithm makes use of analogue neurons. The generalization capabilities of the neural networks resulting from both methods are compared to those of nearest-neighbour classifiers and of Probabilistic Neural Networks implementing Parzen-windows. The latter non-parametric statistical method is applied to estimate the optimal Bayesian classifier. For all methods the generalization capabilities are determined for different ways of pre-processing of the input data. The complexity of the feed-forward neural networks studied does not grow with the training set size. This favours a hardwired implementation of these neural networks as any implementation of the other two methods grows linearly with the training set size.

1. Introduction

In this paper we discuss strictly layered feed-forward neural networks for the recognition of energy deposition patterns in a calorimeter. This work is carried out within the context of the CERN-EAST collaboration (Embedded Architectures for Second-level Triggering, DRDC/RD-11) [10]. Feed-forward neural networks are well fitted for triggering purposes, as demonstrated in e.g. [7]. They belong to a class of universal approximators [16] and their pipeline architecture may allow for the high input data rates to be expected at future high-luminosity facilities like the Large Hadron Collider (LHC). Moreover their fixed, layered architecture provides for a constant response time, which makes them well suited for real time environments with critical time demands. This last characteristic is in sharp contrast to the behaviour of large interconnected neural networks, like Hopfield networks, which are less predictable due to undesirable features like strange attractors and a non constant relaxation time.

Two different data sets of Monte Carlo generated calorimeter patterns are used to construct shower classi-

fiers. For these sets, several methods for shower classification have been designed and studied by different groups within the CERN-EAST collaboration. Both classical methods and neural networks have been used [1–3,5,18,32]. Good performance with respect to the number of correctly classified patterns and the ability to handle a high average event rate of typically 100 kHz are essential for future trigger applications. The aim of this study is to constitute a systematic approach to yield neural network classifiers that implement relatively simple solutions in order to meet the requirements above. We evaluate the performance of the neural networks by using non-parametric statistical methods, with well defined error bounds, as quality measures. For training and testing the split data method is applied, i.e. the data set is divided into two parts: the training set and the test set. The training set is used for the construction of the classifier and the test set is used to measure its generalization capability. For the methods applied the generalization capability is compared as a function of the training set size and the method of pre-processing of the data sets.

Results obtained by applying Bayes' decision rule are the best one can hope to achieve. Since this rule is optimal, it is important to determine for any method its closeness to Bayes' decision rule. In most cases this rule cannot be applied directly, because the underlying probability densities are unknown. For the problems presented in this paper

* Corresponding author. Tel. +31 30 537556, fax +31 30 537555, e-mail andree@fys.ruu.nl.

four different data classification methods are applied. From the field of non-parametric statistics we used Parzen-windows [9,23], implemented by Probabilistic Neural Networks [30,31], to estimate the underlying probability densities. Another non-parametric classifier is the nearest-neighbour classifier [6,9], which performs a partitioning of the input space based on a distance measure with respect to a set of prototypes. For these two statistical methods the bounds of the generalization properties are well defined, therefore, they can be used as a quality measure for the generated neural networks.

Neural networks consisting of linear threshold units are generated by a constructive learning algorithm, a modified version of the Patch algorithm [4]. A constructive learning algorithm tries to construct a neural network that classifies all patterns in the training set according to their output label [4,11–14,17,19–22,27,33]. As a non-constructive counterpart the back-propagation algorithm [29] is applied.

2. Statistical decision rules

2.1. Bayes strategy

Given a set P of observations, an input set can be constructed by assigning an output label θ to each pattern $\mathbf{p} \in P$. This results in an input set $\{(\mathbf{p}, \theta) | \mathbf{p} \in P \text{ and } \theta \in I\}$, where $I = \{1, 2, \dots, M\}$ is the index set of the set of different categories $\{C_i : i \in I\}$. The optimal classification is given by Bayes' decision rule, i.e. assign $\mathbf{p}$ to the label $\theta = i$ with the largest conditional probability:

$$\text{If for all } j \neq i \text{ } \Pr(\theta = i | \mathbf{p}) > \Pr(\theta = j | \mathbf{p}), \text{ decide } \mathbf{p} \in C_i, \quad (1)$$

where i and j are taken from the index set $I = \{1, 2, \dots, M\}$ of the set of different categories $\{C_i : i \in I\}$ and $\Pr(\theta = i | \mathbf{p})$ results from Bayes' law:

$$\Pr(\theta = i | \mathbf{p}) = \frac{\Pr(\mathbf{p} | \theta = i) \Pr(\theta = i)}{\sum_{j=1}^M \Pr(\mathbf{p} | \theta = j) \Pr(\theta = j)} = \frac{\Pr(\mathbf{p} | \theta = i) \Pr(\theta = i)}{\Pr(\mathbf{p})}, \quad (2)$$

where $\Pr(\theta = i)$ is the a priori probability of category C_i , $\Pr(\mathbf{p} | \theta = i)$ the probability of getting a measurement $\mathbf{p}$ from category C_i , and $\Pr(\theta = i | \mathbf{p})$ the a posteriori probability of a measurement $\mathbf{p}$ belonging to category C_i .

In practice all $\Pr(\mathbf{p} | \theta = i)$, and $\Pr(\theta = i)$ are unknown: a non-parametric classification problem. In fact no optimal classification algorithm can exist with respect to all possible probability densities of the pattern space. Bayes' rule results in the minimum expected risk, and therefore is the best one can achieve. Thus, for any learning algorithm it is important to determine its closeness to Bayes' rule.

2.2. Parzen-windows

In most cases only an input set is given and the underlying probability density of the patterns is unknown. Parzen [23] showed how to construct a family of estimators that asymptotically approach the unknown underlying probability density function $f_\theta(\mathbf{p})$ of patterns $\mathbf{p}$ of category C_θ . The estimate $f_{\theta;N}(\mathbf{p})$ of $f_\theta(\mathbf{p})$ from a training set of N independent samples $(\mathbf{p}, \theta)$ is the arithmetic mean of N window functions ψ :

$$f_{\theta;N}(\mathbf{p}) = \frac{1}{NV_N} \sum_{k=1}^N \psi\left(\frac{\mathbf{p} - \mathbf{p}_{\theta;k}}{h_N}\right), \quad (3)$$

wherein $\mathbf{p}_{\theta;k}$ is the k th training pattern from category C_θ , h_N the window width and V_N the window volume. As proven by Parzen and in Ref. [9] the expected error decreases as the estimate is based on a larger training set, i.e.

$$\lim_{N \rightarrow \infty} E[f_{\theta;N}(\mathbf{p})] = f_\theta(\mathbf{p}), \quad \lim_{N \rightarrow \infty} \text{Var}[f_{\theta;N}(\mathbf{p})] = 0. \quad (4)$$

A suitable window function is the Gaussian, thus we take the following estimate

$$f_{\theta;N}(\mathbf{p}) = \frac{1}{(2\pi)^{d/2} \sigma^d} \frac{1}{N} \sum_{k=1}^N \exp\left[-\frac{(\mathbf{p} - \mathbf{p}_{\theta;k})^2}{2\sigma^2}\right], \quad (5)$$

wherein N the total number of training patterns from category C_θ , d the dimensionality of the pattern space, and σ the window width (a smoothing parameter). A proper value of σ for a set of N patterns from category C_θ is easily found (by a binary search) for the problem discussed in this paper and the generalization (misclassification) is not severely affected by a small change in σ .

As $f_{i;N}(\mathbf{p})$ is an estimate for $\Pr(\mathbf{p} | \theta = i)$ and the probabilities of the different categories are taken to be equal, substitution of Eq. (2) in Eq. (1) yields

$$\text{If for all } j \neq i \text{ } f_{i;N}(\mathbf{p}) > f_{j;N}(\mathbf{p}), \text{ decide } \mathbf{p} \in C_i. \quad (6)$$

2.3. The nearest-neighbour rule

A class of classifiers that is easy to implement is based on the nearest-neighbour rule. Besides the 1-nearest-neighbour rule we also apply the k -nearest-neighbour rule. The k -nearest-neighbour rule assigns a pattern to the category most frequently present among its k nearest-neighbours.

If one defines R^* as the minimum expected risk or Bayes' risk, the risk R of the 1-nearest-neighbour classifier is bounded by

$$R^* \leq R \leq R^* \left(2 - \frac{M}{M-1} R^*\right), \quad (7)$$

where $M > 1$ is the total number of different categories [6,9].

3. Description of the constructive learning algorithm

The constructive learning algorithm we used is a modified version of the Patch algorithm. In the first part of this section we present a concise description of the original Patch algorithm as presented in Ref. [4]. The second part of this section contains the modifications with respect to the original Patch algorithm.

3.1. The Patch algorithm

The Patch algorithm is a constructive learning algorithm that generates strictly layered feed-forward neural networks consisting of linear threshold units. All patterns from the training set are correctly classified by a network generated by the Patch algorithm. This means that each layer has to be faithful: two patterns from the training set that have an identical internal representation on any layer must have an identical output label. The Patch algorithm generates networks for which the number of internal representations decreases with the number of layers that are generated. A network is considered to be completed if for the last layer generated a bijective mapping (i.e. an invertible mapping) exists between its internal representations and the output labels. The construction of a layer k is governed by a quality function

$$Q(k) = -Q_1(k) + \alpha Q_2(k). \quad (8)$$

Herein $Q_1(k)$ is the number of pairs of internal representations on the previous layer ($k-1$) with different output labels and an identical internal representation on layer k . If $Q_1(k)$ is equal to zero, layer k is faithful and no more neurons have to be added to this layer. $Q_2(k)$ is the number of pairs of internal representations on the previous layer ($k-1$) with an identical output label and an identical internal representation on layer k . The Patch algorithm tries to maximize the quality function $Q(k)$ with α chosen such that $Q_1(k)$ dominates the quality function. This maximization is done in weight space. Each neuron n in layer k , with a bias w_0 and a weight vector $w = (w_1, \dots, w_{d'})$, corresponds to a point in $d' + 1$ dimensional space. In case of a neuron in the first layer ($k=1$) d' is the dimension of the training patterns, otherwise it is the dimension of the internal representations on the previous layer ($k-1$). If we consider normalized neurons, i.e. $|n| = 1$, the neurons n correspond to points on a unit hypersphere in weight space. To each internal representation $\xi^{(i)}$ on the previous layer a hyperplane $h^{(i)}$, given by $g(\xi^{(i)}) = w_0 + w_1 \xi_1^{(i)} + w_2 \xi_2^{(i)} + \dots + w_{d'} \xi_{d'}^{(i)} = 0$, corresponds. These hyperplanes partition the surface of the unit hypersphere into patches (polyhedral regions). For any pair of points (neurons) on the same patch, the value of the quality function

$Q(k)$ equals. Therefore, to evaluate $Q(k)$ for any patch it suffices to choose an arbitrary representative from this patch. The algorithm now proceeds as follows:

- 1) Choose an initial neuron n and calculate the quality function $Q(k)$.
- 2) Choose a random direction n' in the $d' + 1$ dimensional space.
- 3) Calculate $Q(k)$ for all patches passed by the trajectory $n + \lambda n'$. The new neuron n is chosen from the patch with the largest value of $Q(k)$.
- 4) If a number (N_{opt}) of trajectories have been evaluated without an improvement of the quality, it is assumed that no better neuron will be found. Thus we add a neuron to the layer under construction that maximizes the quality function $Q(k)$. Otherwise continue with step 2.
- 5) Re-optimize all previously generated neurons in the layer under construction, based on the quality function $Q(k)$ including the last added neuron.
- 6) If the layer k is not faithful, $Q_1(k) \neq 0$, proceed with step 1.
- 7) If the number of different internal representations equals the number of target outputs, the network is completed. Otherwise, start the construction of a new layer and proceed with step 1.

3.2. Modifications of the Patch algorithm

By omitting the demand that every layer is faithful smaller networks with the same performance, i.e. generalization on the test set, can be generated. We therefore use a simple form of cross-validation. The training set is split into two parts and the quality function $Q(k)$ is maximized with respect to the first part. For the first layer a newly generated neuron is accepted if the generalization with respect to the second part improves by a certain factor. Otherwise the first layer is considered to be finished. For the following layers the modified Patch algorithm proceeds as the original Patch algorithm.

The convergence to the “optimal” patch is greatly enhanced, with respect to a random choice of direction, by applying Powell’s method [24]. It is a direction-set method for minimization of a function without the need to calculate derivatives. In step 2 the next trajectory $n + \lambda n'$ on

Fig. 1. A tower of the “spaghetti” calorimeter (left). Each tower consists of four cells for measuring electro-magnetic energy depositions, one cell for measuring hadronic energy depositions, and a wedge.

the hypersphere is determined by calculating n' from a set of conjugate directions.

The random choice of a neuron in a patch does not affect the size of the generated networks or the performance with respect to the training set. But the generalization of the generated networks does depend on the choice

of the neuron inside a patch. The best generalization can be expected from the neuron with maximal stability, i.e. the neuron with maximal distance to any neighbouring patch. We used the QuadProg algorithm by Ruján [28], which calculates the neuron of maximal stability, to optimize the selected neurons.

Fig. 2. Energy deposition patterns of an electron (a, b), pion (c, d), and light-quark jet (e, f) from data set 1, without pileup of minimum bias events. Shown is the deposition in the EM-layer (32×32 cells) with corresponding deposition in the H-layer (16×16 cells).

4. The back-propagation algorithm

The back-propagation algorithm [29] changes the weights and thresholds of the neurons in a fixed feed-forward architecture in order to minimize the cost function

$$Q = \frac{1}{2} \sum_{v=1}^N \sum_{i=1}^D (o_{v,i} - \tau_{v,i})^2, \quad (9)$$

wherein N the total number of patterns in the training set, D the number of output neurons of the network, $o_{v,i}$ the

output state of output neuron i when pattern v is presented to the network, and $\tau_{v,i}$ the target or desired output value of output neuron i for pattern v . To guarantee a decrease of the cost function, the update of weight w_{ij}^k , connecting neuron i in layer k with neuron j in layer $k-1$, is chosen to be

$$\Delta w_{ij}^k = -\eta \frac{\partial Q}{\partial w_{ij}^k}, \quad (10)$$

with η the learning rate. By applying this updating rule the back-propagation algorithm yields a multilayer network which is a minimum mean squared-error approximation to the Bayes' optimal decision rule [25,26].

The introduction of a momentum term (parameter α) [15,29] to the updating rule:

$$\Delta w_{ij}^k(t+1) = -\eta \frac{\partial Q}{\partial w_{ij}^k} + \alpha \Delta w_{ij}^k(t), \quad (11)$$

usually increases the speed of convergence. This extra term makes the change in weight w_{ij}^k at step $t+1$ more similar to the change at step t . There exist no general prescriptions for the selection of the values of parameters η and α .

5. The calorimeter data recognition problems

The training problems studied concern classification of energy depositions in a calorimeter. Two data sets, consisting of simulated events for two different calorimeters, have been used. The goal was to discriminate between electrons and hadrons.

5.1. Data set 1

The calorimeter of the first data set [2] is a design of a “spaghetti” calorimeter for experiments at high-energy hadron colliders, like LHC. This calorimeter consists of a staggered tiling of towers as depicted in Fig. 1. The Monte Carlo events are generated in a window of 16×16 towers covering an area of 0.48×0.48 in (η, ϕ) -space.

The calorimeter studied has a different granularity for the first and second layer, four electro-magnetic cells cover one hadronic cell, as depicted in Fig. 1. Each tower also has a wedge part to give the calorimeter a spherical geometry.

Single particles and jets have been generated by a Monte Carlo simulation at a fixed η and ϕ with a momentum distribution given by:

$$\frac{dN}{dP_t} = P_t \exp(-bP_t), \quad (12)$$

with $b = 0.02 \text{ GeV}^{-1}$. The centre of each energy deposition is chosen at random in a field of 2×2 cells in the centre field of 16×16 towers (i.e. 32×32 cells).

Fig. 3. The electron (a), pion (b), and light-quark jet (c) of Fig. 2, with pileup ($\langle n \rangle = 20$) of minimum bias events. The pileup only occurs in the electro-magnetic layer.

Fig. 4. The mean generalization capability ($\langle G \rangle$), for dataset 1, of electron vs. hadron recognition as a function of the training set size N . The patterns are centred with respect to the absolute maximum and logarithmic pre-processing is applied.

Showers due to electrons are characterized by a small lateral profile in the first layer, while only a relative small fraction of the total energy is deposited in the second layer. The second type of events, hadrons produced in jets, have in general larger lateral shower profiles as well as a larger fraction of the total energy deposited in the second layer. However, showers due to hadrons can also occur without energy deposition in the second layer. In Fig. 2 typical energy deposition patterns are shown.

Three different types of particles are distinguished; electrons, pions, and light-quark jets. Pions and light quark-jets will be referred to as hadrons.

For each type of particle this database contains 4000 energy deposition patterns. Besides 1000 clean events also events with pileup of n minimum bias events which are typical for LHC are used, with n following a Poisson distribution. Three different sets of events with pileup for each particle category were generated: sets of 1000 events contaminated with $\langle n \rangle = 5$, $\langle n \rangle = 10$, and $\langle n \rangle = 20$ events of the minimum bias type. In Fig. 3 the same energy deposition patterns as Fig. 2, are shown, but with pileup ($\langle n \rangle = 20$) contamination. The low-energy background contamination only occurs in the first layer.

5.2. Data set 2

The second data set consists of calorimeter events that satisfy the detector description referred to as “Eagle B” [18]. Electro-magnetic and hadronic cells are of the same size, $0.02 \times 2\pi/300$ in (η, ϕ) -space, and are precisely aligned. In contrast to the first data set only those Monte Carlo events, electrons and QCD-jets, are selected which passed a first-level trigger condition. The condition is defined by a trigger window of 4×4 cells, that slides in steps of 0.1 in both variables (η, ϕ) over the entire calorimeter range. Two adjacent electro-magnetic cells, both not touching the edge of the window, are defined as a cluster region. The remaining electro-magnetic cells (12) and all corresponding 16 hadronic cells form a veto region. The criterium used is a minimum of 25 GeV for clusters and a maximum of 5 GeV for the sum of all veto cells (after thresholding individual level-1 cells at 1 GeV).

Pileup was added after the first-level selection and only slightly distorted the energy spectrum in equal ways for both event types. The effect of the first-level filtering is that both event types only show an energy deposition in the electro-magnetic layer. Therefore, distinction by longitudinal deposition profile is no longer possible. Only the lateral shower profile can be used for discrimination between the different types of events. The energy depositions of electrons and QCD-jets are more similar as both fulfil the first-level condition.

6. Results

6.1. Pre-processing

Multivariate methods to find relationships within a data set can be enhanced by proper pre-processing of the data set before analysis. Examples of useful pre-processing operations can be found in e.g. Ref. [8]. The goals of pre-processing include: independence from scales of measurement, elimination of size effects, and elimination of abundance effects. Pre-processing of the patterns may improve the performance of the learning algorithm with respect to the size of the neural network, its generalization capabilities, and the CPU-time requirement of the learning algorithm. We applied the following pre-processing operations:

a) We either fed the energy E_i deposited in cell i of the calorimeter directly, or applied the non-linear operation $E'_i = \log(1 + E_i)$ for all cells in order to compress the dynamic range of the input set.

b) The patterns were translated such that the centre pixel of the electro-magnetic data field of each pattern coincides with the centre of gravity of the electro-magnetic part of the corresponding event. Alternatively, the patterns were translated such that the centre pixel of each pattern coincides with the absolute maximum of the electro-magnetic part of the corresponding event.

c) The centered patterns were clipped. For data set 1 input sets consisting of 34 dimensional patterns were chosen. Each pattern consists of a 5×5 window around

Fig. 5. The mean generalization capability ($\langle G \rangle$), for dataset 1, of electron vs. hadron recognition as a function of the training set size N . The patterns are centred with respect to the absolute maximum. No logarithmic pre-processing is applied.

the centre pixel of the electro-magnetic data field and a 3×3 window around the centre pixel of the corresponding hadronic data field. Concerning data set 2 each pattern consists of a 9×9 window around the centre pixel of the electro-magnetic data field only.

6.2. Training set choices

For both data sets we tried training sets of different size (N). Concerning data set 1 each training set consists of a number of electrons (N_{el}), a number of pions (N_{pi}) and a number of light quark-jets (N_{lj}). For distinction of elec-

trons and hadrons, these numbers were in the range $N = 75$, i.e. $(N_{el}, N_{pi}, N_{lj}) = (25, 25, 25)$, to $N = 1500$, i.e. $(N_{el}, N_{pi}, N_{lj}) = (500, 500, 500)$.

In case of data set 2 the number of electron patterns N_{el} , and the number of QCD-jets N_{QCD} in the training sets were in the same range, $N = 50$, i.e. $(N_{el}, N_{QCD}) = (25, 25)$, to $N = 1000$, i.e. $(N_{el}, N_{QCD}) = (500, 500)$.

6.3. Generalization

For each method applied the mean generalization $\langle G \rangle$ is determined as a function of the training set size N . For a

Fig. 6. The mean generalization capability ($\langle G \rangle$), for dataset 1, of electron vs. hadron recognition as a function of the training set size N . The patterns are centred with respect to the centre of gravity and logarithmic pre-processing is applied.

Fig. 7. The mean generalization capability ($\langle G \rangle$), for dataset 1, of electron vs. hadron recognition as a function of the training set size N . The patterns are centred with respect to the absolute maximum. No logarithmic pre-processing is applied.

Fig. 8. The mean generalization capability ($\langle\langle G \rangle\rangle$), for dataset 2, of electron vs. QCD-jet recognition as a function of the training set size N . The patterns are centred with respect to the absolute maximum and logarithmic pre-processing is applied.

Fig. 9. The mean generalization capability ($\langle\langle G \rangle\rangle$), for dataset 2, of electron vs. QCD-jet recognition as a function of the training set size N . The patterns are centred with respect to the absolute maximum. No logarithmic pre-processing is applied.

Fig. 10. The mean generalization capability ($\langle\langle G \rangle\rangle$), for dataset 2, of electron vs. QCD-jet recognition as a function of the training set size N . The patterns are centred with respect to the centre of gravity and logarithmic pre-processing is applied.

Fig. 11. The mean generalization capability ($\langle\langle G \rangle\rangle$), for dataset 2, of electron vs. QCD-jet recognition as a function of the training set size N . The patterns are centred with respect to the centre of gravity. No logarithmic pre-processing is applied.

given training set size N , 20 training sets of size N are randomly drawn. Every classification method is “trained” on these 20 training sets. The mean generalization $\langle G \rangle$ of a method is the average of the generalizations on the corresponding test sets, i.e. the remaining patterns present in the data set but not in the training set.

6.4. Results for data set 1

Results for electron vs. hadron recognition are given in Figs. 4–7. The results shown in these figures are obtained for training sets of events without pileup contamination. The generalization capabilities are measured on test sets of clean data (left part of the figures) as well as test sets of data with pileup contamination (right part of the figures). Using a training set of clean data, no pileup contamination, and a test set of data with pileup ($\langle n \rangle = 20$) decreases the generalization for all methods applied. This deterioration in performance could not be compensated by training on data with pileup ($\langle n \rangle = 20$) only. Results for the different centering of the patterns, absolute maximum versus center of gravity, do not significantly differ. In the case of logarithmic pre-processing all methods perform better. The Patch algorithm, in contrast to the other methods, is almost insensitive to whether or not logarithmic pre-processing is applied. This is due to the search for the optimal patch on the hypersphere in weight space. Logarithmic pre-processing does only affect the shape of the patches. By applying logarithmic pre-processing the Patch algorithm and the back-propagation algorithm perform about equally well, both with respect to the generalization as well as to convergence to the absolute minimum. Almost all patterns in the training set are perfectly distinguished. The network sizes differ. The Patch algorithm yields networks of a

single neuron (i.e. larger networks did not contribute to a better generalization). For the back-propagation algorithm networks of at least 3 neurons were chosen. In the case of centering with respect to the absolute maximum the networks, trained by the back-propagation algorithm, consisted of a first layer of 2 neurons, and a single output neuron, whereas centering with respect to the center of gravity required networks consisting of 4 neurons. The other two methods, Probabilistic Neural Networks implementing Parzen-windows and the 1-nearest-neighbour approach, yield implementations that grow linearly with the number of training patterns.

The performance of the back-propagation algorithm was measured only for logarithmic pre-processing. Without logarithmic pre-processing the back-propagation algorithm always converged to a local minimum far away from the optimal solution. No measures were undertaken to escape from a local minimum, since the probability to enter a better minimum is very small for the given training sets. Also k -nearest-neighbour classifiers ($k = 11$, $k = 21$) were tried. In all cases the 1-nearest-neighbour classifier performed best, and $k = 11$ performed better than $k = 21$.

6.5. Results for data set 2

The above discussion also applies to the results for data set 2, shown in Figs. 8–11. Only the generalization performance is less due to the method used to generate this data set. As discussed before the events in this data set fulfil a first-level trigger condition. Both event types only show an energy deposition in the electro-magnetic layer. This in contrast to the events of data set 1, where a large energy deposition in the hadronic layer is a feature of hadronic events only. Furthermore for data set 2 a smaller number

Fig. 12. The mean generalization ($\langle G \rangle$), for dataset 2, as a function of the Parzen-window widths σ_{electron} and $\sigma_{\text{QCD-jet}}$. The patterns are centered with respect to the center of gravity and logarithmic pre-processing is applied. The generalization is measured on clean data (without pileup) for training set size $N = 50$.

of events is available. The networks generated by the Patch algorithm consist of a single neuron. In the case of centering with respect to the absolute maximum the networks, trained by the back-propagation algorithm, consisted of 4 neurons, whereas centering with respect to the center of gravity required networks consisting of 3 neurons.

As mentioned before, for each category a proper value of the Parzen-window width σ (Eq. 5) was easily found. For training sets of $N = 50$ (N_{el} , N_{QCD}) = (25, 25), with logarithmic pre-processing and centering with respect to the centre of gravity applied, the dependency of the generalization on σ for both categories is shown in Fig. 12. The surface shown in Fig. 12 has one single peak and is monotonously decreasing in all directions. For both data sets and all methods of pre-processing the dependence was characterized by a single peak.

7. Conclusions

In this paper pattern recognition by means of feed-forward neural networks is studied. Networks consisting of linear threshold units are generated by a constructive learning algorithm. Networks consisting of analogue neurons are trained by the back-propagation algorithm. Both algorithms perform about equally well with respect to the network size, but only if proper pre-processing is applied. Only simple forms of pre-processing are tried, more sophisticated forms of pre-processing may improve the generalization capabilities of the trained neural networks. The Patch algorithm is rather insensitive to the logarithmic pre-processing. Furthermore the Patch algorithm yields small networks and, if the optimization parameter is large enough, it converges to an optimal solution. The Patch algorithm was able to generate networks consisting of a single neuron with a performance close to that of the more complex solutions found so far. In the case of the back-propagation algorithm the search for the optimal network size is difficult and time consuming. Back-propagation is not guaranteed to converge to a global minimum. The required values of the algorithm parameters depend on the network size as well as the training set and its size.

As the bounds of the generalization properties for the statistical methods, Parzen-windows and nearest-neighbour classifiers, are well defined they can be used as a quality measure for the generated neural networks. For the statistical methods the complexity of a hardware implementation grows linearly with the training set size; as each pattern in the training set is represented by a component in the hardwired implementation. This in contrast to the implementation of the networks generated by the Patch algorithm and the back-propagation algorithm. For the training sets considered in this paper the complexity of a hardwired implementation of these neural networks does not depend on the training set size. This favours to hardwire

the networks from the Patch algorithm and the back-propagation algorithm above those from the other methods.

Acknowledgements

The authors thank R.K. Bock, spokesman of the EAST/RD-11 collaboration of CERN, for the opportunity and support given to fulfil this study.

This work is made possible by financial support from the Foundation for Fundamental Research on Matter (FOM).

References

- [1] H.M.A. Andree, G.T. Barkema, A.J. Borgers, M. Kolstein, W. Lourens, A. Taal, J.C. Vermeulen and L.W. Wiggers; Feedforward neural networks for second-level triggering on calorimeter patterns, Proc. Conf. on Computing in High Energy Physics (CHEP92), Annecy, France, Sept. 1992, p. 21.
- [2] J. Badier, R.K. Bock, C. Charlot and I.C. Legrand, Benchmarking Architectures with SPACAL Data, EAST-note 91-10.
- [3] J. Badier et al. IEEE Trans. Nucl. Sci. NS-40 (1993) 45.
- [4] G.T. Barkema, H.M.A. Andree and A. Taal, Network: Computation in Neural Systems 4 (1993) 393.
- [5] R.K. Bock, J. Carter and I.C. Legrand, A calorimeter feature extraction algorithm adapted for a DSP network running in a data driven mode, EAST-note 94-10.
- [6] T.M. Cover and P.E. Hart, IEEE Trans. Inf. The. 13 (1967) 21.
- [7] B. Denby, Tutorial on Neural Network Applications in High Energy Physics: A 1992 Perspective. Proc. 2nd Int. Workshop on Software Engineering, Artificial Intelligence and Expert Systems in High Energy and Nuclear Physics, La Londe-les-Maures, France, Jan. 1992, p. 287.
- [8] P.G.N. Digby and R.A. Kempton, Multivariate Analysis of Ecological Communities (Chapman and Hall, 1987).
- [9] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis (Wiley, New York, 1973).
- [10] EAST (RD-11), Proposal and Status Report: CERN/DRDC 90-56 (1990) and CERN/DRDC 92-11 (1992).
- [11] S.E. Fahlman and C. Lebiere, The Cascade-Correlation Learning Architecture, Advances in Neural Information Processing Systems 2 (Morgan Kaufmann, (1990) p. 524.
- [12] M. Frean, Neural Computation 2 (1990). 198.
- [13] S.I. Gallant, IEEE Trans. Neural Networks NN-1 (1990) 179.
- [14] M. Golea and M. Marchand, Europhys. Lett. 12 (1990) 205.
- [15] J. Hertz, A. Krogh and R.G. Palmer, Introduction to the Theory of Neural Computation (Addison-Wesley, 1991).
- [16] K. Hornik, M. Stinchcombe and H. White, Neural Networks 2 (1989) 359.
- [17] S.A.J. Keibek, G.T. Barkema, H.M.A. Andree, M.H.F. Savenije and A. Taal, Europhys. Lett. 18 (1992) 555.
- [18] G. Klyuchnikov, R.K. Bock, A. Gheorghe, W. Krischer, M. Nessi and A. Watson, A second-level trigger, based on calorimetry only, EAST-note 92-23 (1992).
- [19] M. Marchand, M. Golea and P. Ruján, Europhys. Lett. 11 (1990) 487.

- [20] D. Martinez and D. Estève, *Europhys. Lett.* 18 (1992) 95.
- [21] M. Marchand and M. Golea, *Network: Computation in Neural Systems* 4 (1993) 67.
- [22] M. Mézard and J.-P. Nadal, *J. Phys. A. Math. Gen.* 22 (1989) 2191.
- [23] E. Parzen, *Ann. Math. Statistics* 33 (1962) 1065.
- [24] W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, *Numerical Recipes in C. The Art of Scientific Computing*. (Cambridge University Press, 1988).
- [25] M.D. Richard and R.P. Lippmann, *Neural Computation* 3 (1991) 461.
- [26] D.W. Ruck, S.K. Rogers, M. Kabrisky, M.E. Oxley and B.W. Suter, *IEEE Trans. Neural Networks* NN-1 (1990) 296.
- [27] P. Ruján and M. Marchand, *Complex Systems* 3 (1989) 229.
- [28] P. Ruján, *J. Phys. I France* 3 (1993) 277.
- [29] D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing Vols. 1 and 2*. (MIT Press, 1986).
- [30] D.F. Specht, *Neural Networks* 3 (1990) 109.
- [31] H. Schiøler and U. Hartmann, *Neural Networks* 5 (1992) 903.
- [32] J.M. Seixas, L.P. Caloba, M.N. Souza, A.L. Braga, A.P. Rodrigues and H. Gottschalk, *Neural Networks Applied to a Second-Level Trigger Based on Calorimeters*, EAST-note 93-17;
- [33] J.A. Sirat and J-P. Nadal, *Network: Computation in Neural Systems* 1 (1990) 423.