November 1992

LU TP 92-23

Neural Networks in High Energy Physics

Carsten Peterson[1]

Department of Theoretical Physics, University of Lund
Sölvegatan 14A, S-22362 Lund, Sweden

Plenary talk presented at the "Computing in High Energy Physics", September 21 - 25, 1991, Annecy, France

Abstract:

The current status of using artificial neural networks in high energy physics is briefly reviewed. Examples of successful off-line applications for jet identification and tracking are presented. Also, non-classification applications like process control and mass reconstruction are discussed. For classification tasks the approach is demystified by stressing that the output can be interpreted as Bayesian probabilities.

In the optimization domain several approaches to track finding are discussed. In particular template matching approaches are emphazised – the elastic arms algorithm and the rotor model.

[1] thepcap@seldc52 (bitnet); carsten@thep.lu.se (internet).

1 Introduction

Artificial Neural Networks (ANN) are now being widely explored in high energy physics, in particular as classifiers in off-line data analysis. Also on-line triggers based on ANN are being installed. In this review talk I focus on those issues that I find particularly interesting, new and/or ignored. For more tutorial and complete treatments of the subject the reader is referred to refs. [1, 2]. In order to make the presentation somewhat self-contained very brief descriptions of the main algorithms and architectures are also included.

2 Feed-forward Architectures

2.1 Classifiers

For pattern recognition problems the so-called multi-layer perceptron (MLP) [3] is the most commonly used one. Here a functional mapping from input (x_k) to output (o_i) values is realized with a function F_i,

$$F_i(x_1, x_2, ...) = g[\sum_j \omega_{ij} g(\sum_k \omega_{jk} x_k)] \tag{1}$$

which corresponds to the feed-forward architecture of fig. 1. The parameters to be fitted are the "weights" ω_{ij} and ω_{jk}. The "transfer function" g is usually chosen as $g(x) = 0.5[1 + \tanh(x)]$. The

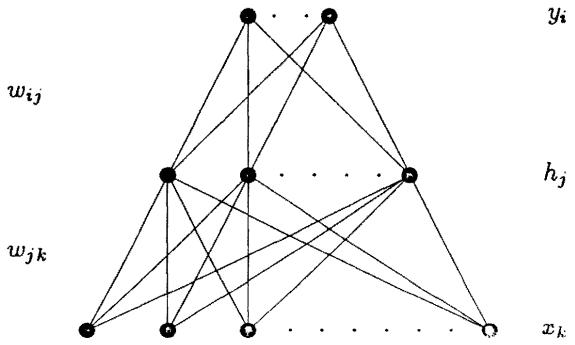

Figure 1: A one hidden layer feed-forward neural network architecture.

bottom layer (input) corresponds to the observed quantities x_k and the top layer to the features o_i. The mission of the so-called hidden layer is to build up an internal representation of the observed data. Each unit, or neuron, has the threshold behaviour given by $g(x)$. In fig. 1 only one hidden layer is displayed. The algorithm can be generalized to any number of hidden layers. Fitting to a

given data set (or "learning") takes place by minimizing e.g. a summed square error[2].

$$E = \frac{1}{2}\sum_i (o_i - t_i)^2 \tag{2}$$

with respect to the weights ω_{ij} and ω_{jk}, where t_i are the desired feature values (the *targets*) [3]. This is done by presenting all the **training patterns** repeatedly with successive adjustments of the weights. Once this iterative learning has reached an acceptable level in terms of a low error E, the weights are frozen and the ANN is ready to be used on patterns it has never seen before. The capability of the network to correctly characterize these **test patterns** is called **generalization** performance. The "vanilla" procedure for minimizing eq. (2) is gradient descent, where the weights ω_{ij} are iteratively updated according to

$$\Delta\omega_{ij} = -\eta\frac{\partial E}{\partial \omega_{ij}} \tag{3}$$

Updating Variants

There exist of course alternatives to eq. (3) for minimzing eq. (2). One is **Conjugate Gradient** (see e.g. ref. [4]). Virtually no improvements have been observed with this method for a number of different applications [5]. In using eq. (3) one might get stuck in local minima. One way of avoiding this is the Langevin updating method. Here one regards eq. (3) as a forward Euler approximation of the differential equations

$$\frac{d\omega_{ij}}{dt} = -\frac{\partial E}{\partial \omega_{ij}} \tag{4}$$

with $\Delta t{=}\eta{=}1$. Augmented with a white appropriately normalized Gaussian random noise ξ eq. (4) becomes the Langevin equation

$$\frac{d\omega_{ij}}{dt} = -\frac{\partial E}{\partial \omega_{ij}} + \xi(T) \tag{5}$$

The noise term facilitates avoiding local minima. Its width (or temperature T) is lowered during the updating process. This extension of the gradient descent method seems to be the most effective and robust one for difficult problems [6].

Architecture

Another issue is of course that of architecture. The number of weights (N_w) as compared to the number of training patterns (N_{tr}) determines the capability of the network to generalize.

In high energy physics "raw" calorimeter data contain many input nodes and consequently many weights, which poses a problem in this context. However in many cases one has symmetries in the input data (e.g. rotation, translation,..). If these can be incorporated into the architecture in one way or another the number of necessary input patterns of course decreases. In the case of translation, rotation and scale invariance the symmetries can of course be taken into account by appropriate Fourier preprocessing of data. There exist two alternative procedures, which are related to each other and more general, that deal with symmetries and reduce the effective number of weights, *local receptive fields* [3] and *weight clustering* [7]. We refer the reader to ref. [8] for a discussion of these issues.

[2] Other error measures exist (see e.g. ref. [2].

For off-line analysis it is often fruitful to preprocess the data. This can be done either by using application expertise in terms of choice of intelligent variables (various shape parameters etc. in jet tagging) or by some standard linear procedure like principal component analysis (PCA) [9] where only the directions of the high dimensional input space correponding to the leading eigenvalues are kept for further ANN processing.

Interpretation of Output Values

When using feedforward networks for test data the classifying output units need not take values that are exactly 1 or 0. How should these be interpretted? It has been shown that these can be interpretted as Bayesian probabilities [10] provided: (1) The estimation is accurate, (2) the outputs are of 1-of-M-type, (3) a square error or cross entropy error function is used. Recall that (see e.g. ref. [11]) a Bayesian probability $P(C_i|\vec{x})$ represents the conditional probability for a class C_i given input $\vec{x}$ Bayes rule tells us that it can be expressed as

$$P(C_i|\vec{x}) = \frac{P(\vec{x}|C_i)P(C_i)}{P(\vec{x})} \tag{6}$$

where $P(\vec{x}|C_i)$ is the conditional probability for producing the input vector $\vec{x}$ given the class C_i and $P(C_i)$ is the *a priori* probability of class C_i. $P(\vec{x})$ is the input probability distribution. In conventional Bayesian analysis $P(\vec{x}|C_i)$ is given by well-known parametric distributions, e.g. Gaussians and the training involves estimating the parameters. In the feed-forward ANN non-parametric training results in the left hand side of eq. (6) directly. The benefit from this Bayesian interpretation of the ANN output units is obvious - the latter can be subject to higher level decision analysis.

High Energy Physics Applications

Identifying the origin of jets has been a major application area so far. Successful discrimination of gluon from quark jets have been reported both in both e^+e- annihilation [12][13][14] and hadron-induced reactions [15][16] using MLP. Identifying b-quarks in e^+e- annihilation was originally pursued in ref. [13] with encouraging results using four momenta of the leading hadrons with the MLP network. These results were later refined using preprocessed input variables like sphericity, transverse masses etc., giving rise to improved efficiencies and purities [17]. In fact the ANN approach to b-quark tagging has opened up the field of finding b-quarks by means of hadronic final states only. This general purpose detector attitude poses an inexpensive and competitive alternative to special purpose vertex detector and lepton tagging methods (see also ref [18] for new b-quark tagging applications). An ANN methology for finding t-quark jets at $p\bar{p}$ colliders have also been suggested, again with preprocessed input data [19].

In general the off-line ANN jet identification applications give results that are better than or equal to what is obtained with conventional methods. In the case of equal performance we feel that the ANN approach has the advantage of being a "black box", which implies that the number of man-hours needed to analyze experimental data is relatively small. Illuminating comparisons between conventional classifiers and ANN for off-line heavy quark tagging using "intelligent" variables can be found in refs. [20, 21, 22]. In fig. 2 such a comparison is shown for a efficiency/purity ratio. As can be seen the ANN method does very well in the comparison.

One should keep in mind that the jet identification results depend on the MC model used to generate the training data. Hence one should limit the training to hadrons located in regions of phase space where the different MC models differ as little as possible. In practice this typically means the most

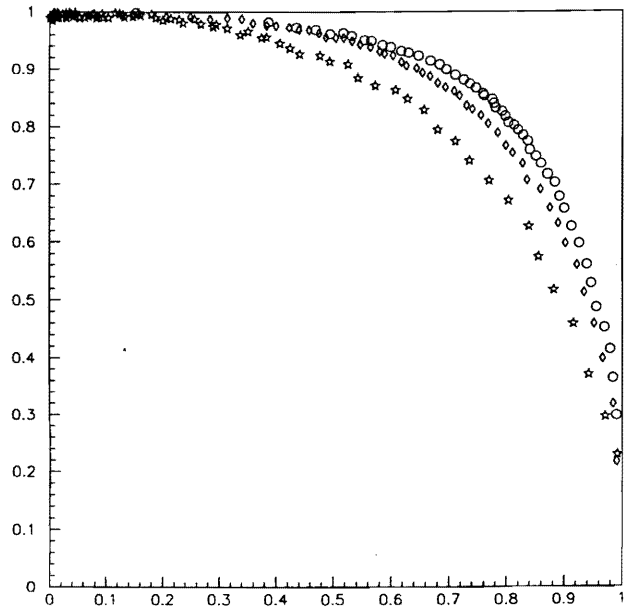

Figure 2: Purity/efficiency for b-quark discrimation (from ref. [20]) using single variable discriminant (star) and multivariate (lozenge) and ANN (open circle) respectively.

energetic hadrons in the jets.

For the next generation of accelerators (LHS, SSC) the availability of trigger algorithms that can be executed in real-time will be crucial since the event rate at these machines is of the order of one event per 10-100 ns. ANN with their intrinsic parallelism and simple processing units are almost ideal for this purpose. Indeed a hardware neural network based on the Intel ETANN chip [23] is being installed into the FNAL CDF detector for b-quark identification using lepton tagging [24].

2.2 Function Approximators

The same class of architectures can be used for cases where the output node is linear, representing any real number and not just a binary class. In this case the network serves as a general **function approximator**. The sigmoids turn out to be very efficient in capturing almost any non-linear behaviour. Impressive results in this area have been reported with chaotic non-linear times series (see eg. ref. [25]) and real-world problems (see e.g. ref. [26]). Here we report on two applications within high energy physics, where ANN are used as function approximators – mass reconstruction and accelerator process control.

Mass Reconstruction

When hunting for new particles or resonances one often encounters the problem of computing invariant masses out of expected decay products. An ANN method has been developed [27] where the invariant mass is computed from calorimetric information of hadronic decay particles for the process $W \rightarrow q\bar{q} \rightarrow$ hadrons in $\bar{p}p$ collisions. It this case a linear output unit representing m_W is used and the input consists of calorimetric signals. In order to prevent the network from learning

just one mass value (M_W) "by heart", W-bosons with fictive masses flatly distributed with 300 events per GeV in the interval [40,160] GeV were generated for training. For test set events with a normal mixture of W and Z^0 events with $M_W = 80.0$ GeV and $M_{Z^0} = 91.2$ GeV were generated.

The standard procedure for reconstructing masses is to use the "window" or cone algorithm to find two jets, defined as the two circular areas in the η, ϕ plane with radius $R^2 = \Delta\eta^2 + \Delta\phi^2$ with the largest $E_\perp$, and then to calculate their invariant mass. In ref. [28] it was found that the optimum choice of R is 0.8. This method was employed in ref. [27] to provide a comparison for the ANN algorithm[3]. The final value of the width from the ANN algorithm is 0.15 as compared with 0.19

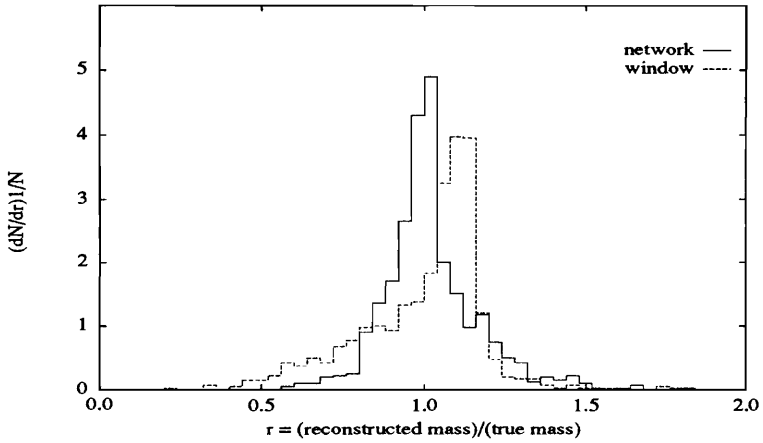

Figure 3: The reconstructed mass ($M_{W,Z}$) divided by the true mass ($M^0_{W,Z}$) using the neural network method (full line) and the conventional "window" method (dashed line) with $R = 0.8$.

when reconstructing the mass with the "window" method. In fig. 3 this distribution is shown for the test set with a normal mixture of events with W and Z distributed according to their true masses and widths. As can be seen from fig. 3 the neural network approach does significantly better than the conventional method, both with respect to peak position and width. The main reason for this is that it captures the gluon bremsstrahlung tails well and that it is more resistant to noise.

The method developed here could be of utmost importance when it comes to separating $H^0 \rightarrow W^+W^-$ produced at LHC/SSC energies from a background consisting of $W + jets$, $tt \rightarrow W + jets$ and $bb \rightarrow l + jets$. In this case one needs to reconstruct M_W.

Process Control

Using ANN for process control involves two steps:

- System identification (fixing the weights).
- Optimization of setpoints.

System identification. This falls within the function approximator domain described above. In

[3]In reality the the cone algorithm in JETSET [29] called LUCELL was used which is only slightly different from the one of ref. [28].

this case one has a set of outputs (o_i) representing the quantities to be optimized. These depend on control settings c_k, and measured state variables s_k

$$\vec{x} = (c_k; s_k) \tag{7}$$

This situation represents a static system (like pattern recognition). However, dynamical systems often have a time lag dependence that can be included by having quantities at time t as outputs, $o_i(t)$. Correspondingly earlier recordings of the outputs [$o_i(t-1)$,$o_i(t-2)$,...,$o_i(t-\tau)$] are fed into the network as inputs

$$\vec{x} = (c_k; s_k; o_i(t-\tau)) \tag{8}$$

The information in these delayed inputs contains to some extent those state variables that are not explicitly measured and included in the input.

Optimization of setpoints. Once the weights have been determined from fitting eq. (1) to training data the established model can be used to to optimize o_i with respect to the control settings c_k. This can be done either with exhaustive search, gradient descent, or some other approximate procedure. Needless to say the reliability of new settings will rely on to what extent the training data used for establishing the function covers the relevant parts of phase space well (see below).

In ref. [9] promising exploratory investigations were made predicting the CERN SPS ion source intensity.

3 Self-organization

The MLP networks discussed above assumes the knowledge about what features (o_i) are relevant from the outset and separates the data accordingly. There is also an alternative approach, *self-organization* where the network organizes the data into features without any external teacher (no output units) [31]. The underlying architecture consists of an input layer (x_k) and a layer of feature nodes denoted h_j (see fig. 4), where

$$h_j = g(\sum_k \omega_{jk} x_k) = g(\vec{\omega}_j \cdot \vec{x}) \tag{9}$$

For all patterns presented the weights are updated such that the angles between $\vec{\omega}_j$ and $\vec{x}$ are minimized. Also, topological ordering between the feature nodes h_j are introduced with a "mexican-hat" potential, such that neighbouring nodes in a plane react to similar features. Such a system has no "teacher" like the feed-forward MLP network above where answers are compared with correct values t_i. The results in this approach are extremely easy to analyze; the weight vectors $\vec{\omega}_j$ for the different feature nodes point in the direction of typical data in $\vec{x}$-space. These networks are natural extensions of the well known *k-means* clustering algorithm [30]. It is also interesting to note that with an appropriate learning rule the network in fig. 4 can host the pricipal component analysis (PCA) (see e.g. ref. [1]).

For classification purposes one can augment self-organization with supervised learning for fine tuning the units specific to certain features [31]. This is called *learning vector quantization* (LVQ) and amounts to learning correct answers and unlearning incorrect answers. As was demonstrated in ref. [32] this LVQ variant handles high dimensional problems less efficiently than the MLP.

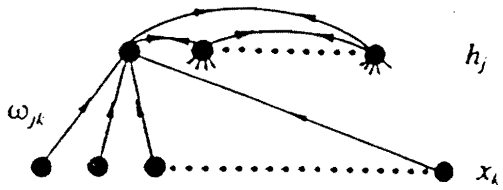

Figure 4: A one-layer self-organizing network. Lateral interactions between the feature nodes correspond to the "mexican-hat" potential.

In the area of heavy quark identification self-organizing networks have turned out to be fruitful given the power of these networks to illuminate the underlying structure [33]. Also subtle differences between different fragmentation models were discovered using a self-organizing network [33].

4 Track Finding

Feedback ANN approaches, and variations thereof, have shown quite some power in finding good approximate solutions to difficult optimization problems [34, 35, 36, 37, 38]. In refs. [39, 40] such an approach is pursued for toy-sized high energy physics track finding problems. The basic idea is to assign a decision element (neuron) s_{ij} between two signals i and j which is equal to 1 if i and j are connected and 0 otherwise. An energy function

$$E = E(s_{ij}) \tag{10}$$

is then constructed such that smooth tracks with no bifurcations correspond to global minima. In order to avoid local minima in eq. (10) the mean field approximation (MFT) is employed, in which eq. (10) is minimized by iteratively solving the MFT equations

$$v_{ij} = 0.5(1 + \tanh(-\frac{\partial E(v_{ij})}{\partial v_{ij}}\frac{1}{T}) \tag{11}$$

where "temperature" T and thermal averages $v_{ij} = \langle s_{ij} \rangle_T$ have been introduced. The temperature represents noise that is introduced to avoid local minima. The advantage of the MFT approach (eq. (11)) is that a true stochastic procedure can be emulated by the set of deterministic MFT equations (eq. (11)). The system "feels" its way towards good solutions.

In its original form this approach requires N^2 degrees of freedom for N signals with full potential connectivity between the signal points. In reality this is however never the case. In ref. [41] realistic cuts on the degrees of freedom were made on an application with real TPC data from the CERN ALEPH detector. The performance in terms of quality of the ANN algorithm solutions on this problem turned out to be compatible with that of the conventional one used in the ALEPH detector. With regard to execution speed the ANN approach is a winner, in particular for high multiplicity events [41].

Another neural approach is to have a rotor neuron [42] associated with each signal interacting in such a way that smooth tracks [43] are promoted. This method requires only O(N) degrees of freedom. We will return to this approach below.

Even though the neural approach [39, 40, 41] seems to work very well it may not be the optimal way to proceed for the particle physics track finding problem for the following reasons:

1. It only solves the combinatorial optimization part of the problem; assigns signals to tracks. One still has to make a fit to obtain the momenta. It is desirable to have an algorithm that does both these things in one "shot".

2. The neural approach is more general than what is needed for this problem since the parametric form of the tracks are known in advance – straight lines or helices. Any powerful algorithm should take advantage of this prior knowledge of the parametric form. This is the way a human being would do it. To illustrate this point we show in fig. 5a signal points generated from 15 straight tracks with 100% noise. The human method of spotting the tracks in the noisy environment is by inspecting the image in fig. 5 by holding it up and look for straight lines[4].

3. The number of degrees of freedom needed to solve the a N signal problem is large even with the connectivity restrictions imposed in refs. [40, 41]. For a problem with N signals and M tracks one should in principle only require $O(\mathrm{M})$ degrees of freedom.

4. As demonstrated in ref. [44] the neural approach is somewhat sensitive to noise. Again with prior knowledge of the parametric form one should be more robust with respect to noise (cf. pt. 2 above).

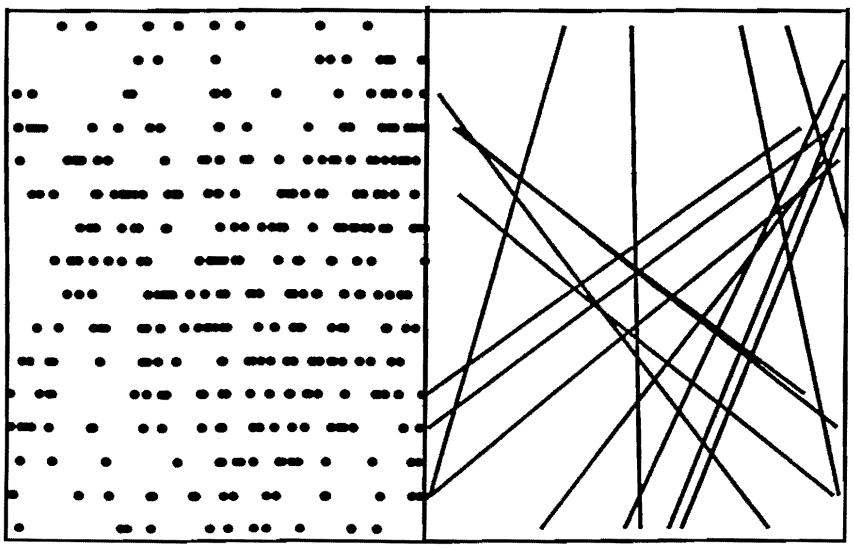

Figure 5: (a) Signal points from 15 generated straight tracks with 100% noise. (b) The corresponding solution. From ref. [44].

All these issues can be accounted for in a novel approach [45, 46, 47] based on so-called deformable templates or elastic nets [37]. A very similar approach was independently pursued in ref. [44].

[4] I encourage the reader to hold up the page to verify this.

4.1 The Elastic Arms Approach

The Strategy

The strategy is to try to match the observed events to simple parameterized models (templates) where the form of the models contains the ***a priori*** knowledge about the possible tracks, e.g. circles passing through the origin (the collision point). In addition, this formulation allows for some data points, those corresponding to noise, to be unmatched. The mechanism involved is closely related to redescending M-estimators used in Robust Statistics [48].

The algorithm works in two steps. First a Hough transform [11] is used to give initial conditions for the templates and to specify the number of templates required. Hough transforms are essentially variants of "histogramming" or "binning" techniques which have previously been applied to particle tracking. Given a set of M tracks given by the parameters $(\theta_a, \kappa_a, \gamma_a)$ in the case of helices from the Hough transform, the elastic arms method then takes over and resolves ambiguities and makes detailed fits to the signals. In ref. [45, 46] the derivation of the elastic arms approach was based on minimizing the following energy function.

$$E[s_{ia}; \vec{\pi}_a] = \sum_{i,a} s_{ia} M_{ia} + \lambda \sum_i \{\sum_a s_{ia} - 1\}^2 \tag{12}$$

where: (i) $\vec{x}_i$ labels the positions of the measurement points $(i = 1, ..., N)$), (ii) $\vec{\pi}_a$ the track parameters (in the case of helices $\vec{\pi}_a$=$(\theta_a,\kappa_a,\gamma_a)$), (ii) $M(\vec{x}_i, \vec{\pi}_a)$, which in what follows is abbreviated with M_{ia}, is a measure of distance between the i^{th} point and the a^{th} template, and (iii) s_{ia} is a binary decision unit (neuron) such that

$$s_{ia} = 1 \tag{13}$$

if the a^{th} arm goes through the i^{th} point and is zero otherwise. $E[s_{ia}, \vec{\pi}_a]$ is minimized with respect to s_{ia} and $\vec{\pi}_a$ subject to the global constraint that each point is either matched to a unique template or not matched. More precisely, given i there exist a unique a such that $s_{ia} = 1$. The second term in eq. (12) imposes a penalty λ if a specific point is unmatched to any circle. In order to avoid local minima when minimizing eq. (12) a Boltzmann distribution is introduced for E

$$P[s_{ia}, \vec{\pi}_a] = \frac{e^{-\beta E[s_{ia}; \vec{\pi}_a]}}{Z} \tag{14}$$

with β=1/T as the inverse temperature (noise level) and Z as the partition function. Integrating out the neuronic degrees of freedom (s_{ia}) subject to the constraint of eq. (13), yields a marginal distribution

$$P_M[\vec{\pi}_a]] = \frac{1}{Z} e^{-\beta E_{eff}[\vec{\pi}_a]} \tag{15}$$

where we have introduced the *effective* energy E_{eff} as

$$E_{eff}[\vec{\pi}_a] = -\frac{1}{\beta} \sum_i \log\{e^{-\beta\lambda} + \sum_a e^{-\beta M_{ia}}\} \tag{16}$$

Gradient descent on eq. (16) gives rise to the updating equations that converges to the tracking solutions. Note that both the the combinatorial and fitting parts of the problem (s_{ia} and $\vec{\pi}_a$ respectively) are being solved.

A Simple Derivation

The resulting updating equations can also be obtained with an alternative and somewhat more intuitive way [47], where one in addition to s_{ia} introduces N *zero*-neurons s_{i0} ($i = 1, ..., N$), such that $M_{i0} = \lambda$ for all i. The energy function equivalent to eq. (12) which then reads

$$E[s_{ia}; \vec{\pi}_a] = \sum_{i=1}^{N} \sum_{a=0}^{M} s_{ia} M_{ia} \ , \tag{17}$$

should be minimized together with the ***Potts*** condition

$$\sum_{a=0}^{M} s_{ia} = 1 \quad \forall \, i \tag{18}$$

This condition is of course equivalent to eq. (13). Using gradient descent to minimize E with respect to $\vec{\pi}_a$, gives

$$\Delta \pi_a^{(k)} = -\eta_a^{(k)} \sum_i s_{ia} \frac{\partial M_{ia}}{\partial \pi_a^{(k)}} \tag{19}$$

Next we replace s_{ia} with its thermal average $v_{ia} = \langle s_{ia} \rangle_T$. The MFT equations for Potts neurons [40] analogous to the ones in eq. (11) for v_{ia} read

$$v_{ia} = \frac{e^{U_{ia}}}{\sum_{b=0}^{M} e^{U_{ia}}} \tag{20}$$

where the local field U_{ia} is given by

$$U_{ia} = -\frac{\partial E}{\partial v_{ia}} \beta = -\beta M_{ia} \tag{21}$$

Substituting this into eq. (20) yields

$$\begin{aligned} v_{ia} &= \frac{e^{-\beta M_{ia}}}{\sum_{b=0}^{M} e^{-\beta M_{ib}}} \\ &= \frac{e^{-\beta M_{ia}}}{e^{-\beta \lambda} + \sum_{b=1}^{M} e^{-\beta M_{ib}}} \end{aligned} \tag{22}$$

How Does the Algorithm Work?

How does this algorithm work? At the starting temperature T_H a set of template arms are placed according to the Hough transform values for the parameters $\vec{\pi}_a$. The templates are Gaussian distributions centered around the arm values where the width is given by the temperature (see fig. 6). Initially each arm can attract many signals. The relative importance of the different signals is measured by the Potts factor (eq. (22)). As the temperature is lowered the different arms are attracted to nearby signals more intensely.

The N *zero*-neurons v_{oa} are introduced because of the presence of noise signals in the data. It is therefore natural to interpret v_{i0} as the probability for signal i to be a noise signal, and v_{i0} is given by

$$v_{i0} = \frac{e^{-\beta \lambda}}{e^{-\beta \lambda} + \sum_{b=1}^{M} e^{-\beta M_{ib}}} \ . \tag{23}$$

With this interpretation v_{i0} can be used as a tool to identify non-fitted signals. If $v_{i0} \approx 1$ after the annealing of the algorithm ($T \rightarrow 0$), then i is a possible noise signal or a signal belonging to a track not included in the formalism. It is very instructive to inspect how the different v_{i0}'s develop with

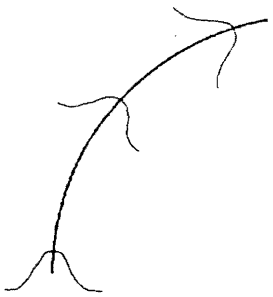

Figure 6: An elastic arm at temperature T.

decreasing temperature. If we choose λ to be the square of some typical distance representing the error in the initialization of the algorithm then, at high temperature, $v_{i0} > 0$ for most of the signals. As the temperature decreases "decisions" are made whether signal i belongs to a track or not. This can be seen in fig. 7 where the different v_{i0}'s are plotted against the number of iterations. The v_{i0}'s goes to either 0, meaning that signal i is assigned to a track, or 1, corresponding to i being a noise signal. Decisions are not made at a common temperature, instead the different v_{i0}'s converge at different temperatures (iteration steps). We also see that, already at a high temperature, $v_{i0} \approx 1$ for some signals, meaning that there is no initial arm close to these signals - they can form possible secondary tracks.

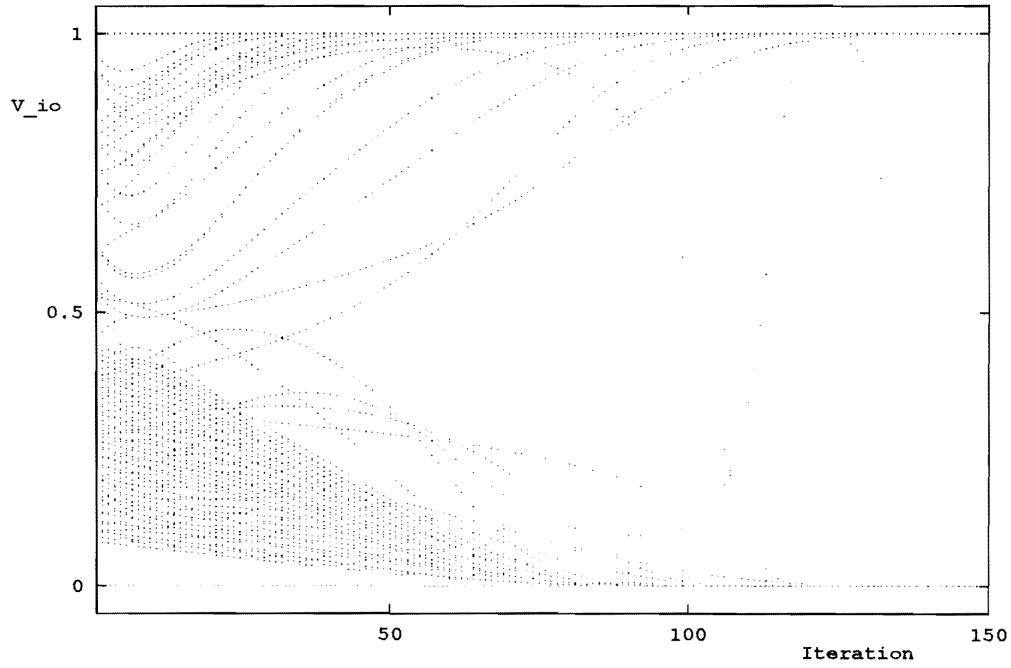

Figure 7: Development of the v_{i0}'s for a generated CERN DELPHI TPC event (371 signal points).

The elastic arms algorithm is also strongly related to Bayes' theorem as is discussed in ref. [46]. In refs. [45, 46] it is also shown how the Hough transform is a $\beta \mapsto \infty$ and $\lambda \mapsto 0$ limit of eq. (12) and how this approach is related to that of ref. [44], where also Hough transforms are used specify the number of tracks and initial parameters values.

Implementation Issues

The solutions are obtained by iteratively solving eqs. (22) by annealing the temperature at each step. The optimal updating rates $\eta_a^{(k)}$ are of course different for the different parameters. In ref. [47] an automated procedure for choosing these were developed, which is based on finding the natural metric for the problem.

The performance of the algorithm has been tested with simulated data from the CERN DELPHI TPC detector. A typical result is shown in fig. 8. It is encouraging to see how well the algorithm works. The arms do not confuse one another, even when passing close, or crossing each other. This is of course due to the Potts factor (eq. (22)), which in a sense "decides" which track each arm should be attached to and ignores the others. Not only does the algorithm exhibit good performance, it is

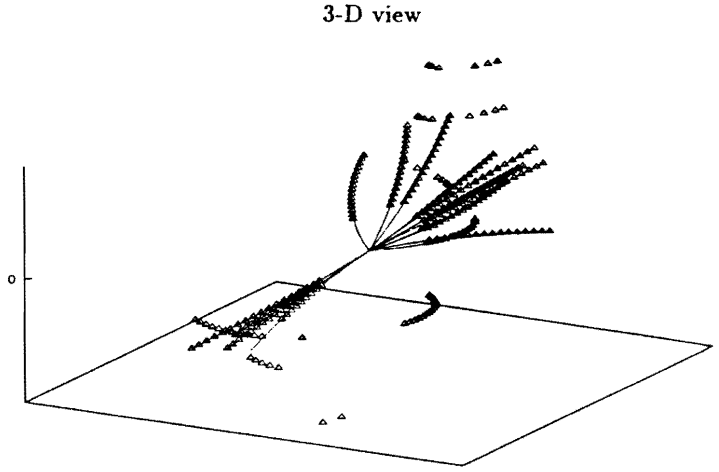

Figure 8: Result from Hough/Elastic arms algorithm with signals generated by CERN DELPHI TPC event generator (308 signal points).

also very cost effective in every respect. It scales like $N \times M$ and processes a typical DELPHI TPC event in one minute ona DEC 3100 workstation. The underlying F77 code only contains $O(300)$ lines.

Extensions of the Algorithm

So far all tracks emerging from other points than the assumed vertex position have been considered noise. This restriction is often unrealistic. In ref. [47] the formalism was extended to include, (i) secondary tracks coming from decaying particles and (ii) multiple vertex positions. In refs. [47, 49] it was also demonstrated how to handle the problem of mirror charges in the detectors.

Secondary tracks and **multiple vertex positions** are treated by having the vertex $\vec{r}_j^{\,o} =$

(x_j^o, y_j^o, z_j^o) $(j = 1, ..., K)$ as an additional parameter for the template with associated updating procedures. With this extension the algorithm could be used for vertex detection in general. This is most relevant at LHC/SSC luminosities, where one expects multiple events per bunch crossing.

Mirror Charges occur as ambiguities in some detectors where a coordinate for a given signal may be double-valued – it has a mirror signal, $(x, y) \rightarrow (x^+ \text{ or } x^-, y)$. This ambiguity problem can be solved [49] in the same manner as noise signals are handled, but on different levels of resolution. This is seen by rewriting the original energy function (eq. (12)) as

$$E(\{s_{ia}, s_{ia}^+, s_{ia}^-\}, \vec{\pi}_a) = \sum_{i=1}^{N}\sum_{a=1}^{M} s_{ia}\left(s_{ia}^+ M_{ia}^+ + s_{ia}^- M_{ia}^-\right) + \lambda \sum_{i=1}^{N}\left(\sum_{a=1}^{M} s_{ia} - 1\right)^2 , \qquad (24)$$

where the new variables s_{ia}^+ and s_{ia}^- are defined by

$$s_{ia}^+ \, (s_{ia}^-) = \begin{cases} 1 & \text{if signal } i^+ \, (i^-) \text{ is assigned to arm } a \\ 0 & \text{otherwise} \end{cases} , \qquad (25)$$

and with the corresponding distance measure M_{ia}^+ (M_{ia}^-). A derivation analogous to the above then gives new updating equations for the associated Potts neurons $\hat{v}_{ia}^+$ and $\hat{v}_{ia}^-$.

In summary this is a track finding method that combines the matching and the fitting problem into a single algorithm. It goes from coarse to fine resolution by using a variant Hough transform to intitialize a set of elastic arms. The approach is easy to adapt to specific situations. For example, suppose measurement precisions vary for different pad-layers. Then the formalism can be generalized to allow for different i-dependent λ's for the different pad-layers.

Although the core algorithm is extremely robust and generalizable to new situations, the initialization procedure has to be custom-made for different experimental configurations.

Particle physics tracking codes always end up "dirty" with ample of exceptions etc. The elastic arms approach has the advantage that the code based on it starts out from a clean base with global constraints built in. Nevertheless it is flexible to host a variety of experimental setups. The only "engineering" needed concerns the initialization. Furthermore the approach has the advantage of being intrinsically parallel, facilitating design of custom made hardware for real-time execution.

4.2 The Rotor Model Approach

A completely different neuronic approach to track finding is the rotor approach [43], where rotors $\vec{s}_i$ are associated with signal points. These rotors interact with each other (within some neighbourhood) and with line segment vectors $\vec{r}_{ij}$ (see fig. 9). A suitable energy function for lining up the rotors to form tracks is [43]

$$E = -\frac{1}{2}\sum_{ij}\frac{1}{|\vec{r}_{ij}|^m}\vec{s}_i\vec{s}_j - \frac{\alpha}{2}\sum_{ij}\frac{1}{|\vec{r}_{ij}|^m}(\vec{s}_i\vec{r}_{ij})^2 \qquad (26)$$

where the first term aligns neighbouring rotors to each other and the second term aligns rotors with neighbouring template segments. The factor $1/|\vec{r}_{ij}|^m$ suppresses non-local interactions. The balance between the two constraints is governed by the Lagrange parameter α.

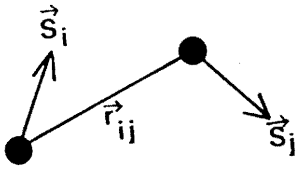

Figure 9: Rotors $\vec{s}_i$ and line segment vectors $\vec{r}_{ij}$.

Again the MFT technology can be applied for finding good minima. The MFT equations for this system read [42]

$$\vec{v}_i = \hat{u}_i \frac{I_1}{I_0}(|\vec{u}_i|) \tag{27}$$

where

$$\vec{u}_i = -\frac{\partial E}{\partial \vec{v}_i}\frac{1}{T} \tag{28}$$

and $\vec{v}_i =< \vec{s}_i >_T$ are the correponding mean field variables. I_1 and I_0 are Bessel functions. Note that $\vec{v}_i$ does not lie on the unit circle for $T \neq 0$. At high temperatures $\vec{v}_i$ will be in the vicinity of the origin. As the temperature is lowered it will "wander" out to the unit circle (see fig. 10a). Its final direction defines the solution (this is in sharp contrast to methods based on making small steps in $\vec{s}_i$, in which case one is is confined to moving on the circle). In fig. 10b we show the behaviour of $I_1/I_0(x)$. This threshold function is reminiscent of a tanh-function. These MFT rotor neuron technique has turned out to be very powerful in another angular optimization problem - placing charges on a sphere [42].

As it stands eq. (26) assumes no parametric form of the tracks. It has demonstrated in ref. [51] that the requirement of circular tracks can be naturally incorporated by replacing eq. (26) by the expression

$$E = -\frac{1}{2}\sum_{ij} \vec{s}_i \vec{s}'_j = -\frac{1}{2}\vec{s}_i T_{ij} \vec{s}_j \tag{29}$$

where T_{ij}=$T(\phi_{ij})$ is the rotation matrix for the angle ϕ_{ij} between $\vec{r}_{ij}$ and the $\hat{x}$ axis emerging from signal i. Hence $\vec{s}'_j$ in eq. (29) is the mirror reflection of $\vec{s}_j$ relative to $\vec{r}_{ij}$ Ref. [51] reports on successful use of this energy function together with the MFT equations (27) on a multiwire proportional counter tracking problem.

Figure 10: (a). MFT development of $\vec{v}_i$ from $T = \infty$ to $T = 0$. (b). $I_1/I_0(|\vec{u}_i|)$ as a function of $|\vec{u}_i|$.

5 Summary

It is clear that ANN are here to stay, both in general and with respect to high energy physics applications. Within 5 years from now one will find ANN in standard software packages, both commercial and public domain ones. Also, the special purpose hardware implementations will play an important role in intelligent data processing.

What is really New?

A question that arises is what ANN really has brought to the table as compared to standard approaches.

- **Classifiers.** The muli-layered perceptron (MPL) is a generic non-linear extension of standard discrimination methods and it thus contains the latter as a special case.
- **Function Approximators.** Again the inclusion of hidden layer(s) makes the the ANN approach a generalization of linear predictors etc..
- **Self-organization.** Self-organizing networks host the conventional *k-means* clustering algorithm as a special case. In addition these networks also have the capability of grouping similar features nearby in a metric space, which facilitates the interpretations. Principal component analysis can also be formulated as a self-organizing neural system.
- **Combinatorial Optimization.** Whereas in the domains listed above the ANN approach are non-linear extensions of existing technologies, the situation is different when finding good solutions to optimization problems. These problems are mapped onto energy fuctions very similar to those of magnetic systems. Hence tools from statistical mechanics can be exploited. In addition to the simulated annealing method one uses the mean field approximation. With this technique the systems feel their ways in a "fuzzy" manner towards good solutions. This is in contrast to most other methods, where discrete moves take place in solution space.

High Energy Physics

The progress of exploiting ANN in high enregy physics has been somewhat slow. Partly this conservatism is due to the a misconception that ANN approaches contain an element of "black box magic" as compared to conventional approaches. I hope I have convinced the reader that this is not the case. Statistical interpretation of the answers makes the ANN approach as well-defined to use as the discriminant ones.

Off-line uses of ANN for classification problems are presently dominated by jet tagging. Very successful results exist for gluon and heavy quark identification. The performance seems to be better than or equal to that of conventional methods. The public domain F77 software package (`JETNET 2.0`) [52] has been commonly used in many of the applications discussed in this talk.

Parallel execution with custom made hardware is natural for ANN. Indeed, such real-time devices are being installed [24].

Novel pplications domains like mass reconstruction and process control look very promising.

In the area of track finding variants of ANN, elastic arms (or net) algorithms appears to be very powerful given their flexibility and noise resistance.

References

[1] J. Hertz, A. Krogh and R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, Ca. (1991).

[2] C. Peterson and T. Rögnvaldsson, "An Introduction to Artificial Neural Networks", *Proceedings of the 1991 Cern School of Computing, Ystad, Sweden*, CERN Yellow Report 92-02. *LU TP 91-23.*

[3] D. E. Rumelhart and J. L. McClelland (eds.) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition (Vol. 1)*, MIT Press (1986).

[4] See e.g. W.P. Press, B.P Flannery, S.A. Teukolsky and W.T. Vettering, *Numerical Recipes, The Art of Scientific Computing*, Cambridge University Press, Cambridge, (1986).

[5] L. Lönnblad, C. Peterson and T. Rögnvaldsson, in preparation.

[6] T. Rögnvaldsson, in preparation.

[7] S.J. Nowlan and G.E. Hinton, "Simplifying Neural Networks by Soft Weight-Sharing", University of Toronto Computer Science preprint 1991 (to appear in Neural Computation).

[8] C. Peterson, "Neural Networks in High Energy Physics - Algorithms and Results" (invited talk), *Proceedings of the Second International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics, January 1992, L'Agelonde, France*, eds. D. Perret-Gallix, World Scientific (Singapore 1992).

[9] C. Peterson, Rögnvaldsson and E. Wildner, "Controlling Ion Sources with Artificial Neural Networks", *CERN PS/OP 92-34 (LU TP 92-32).*

[10] M.D. Richard and R.P. Lippmann, "Neural Network Classifiers Estimate *a Posteriori* Probabilities", *Neural Computation* **3**, 461 (1991).

[11] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, John Wiley and Sons, New York (1973).

[12] L. Lönnblad, C. Peterson and T. Rögnvaldsson, "Finding Gluon Jets with a Neural trigger", *Physical Review Letters* **65**, 1321 (1990).

[13] L. Lönnblad, C. Peterson and T. Rögnvaldsson, "Using Neural Networks to Identify Jets", *Nuclear Physics* **B 349**, 675 (1991).

[14] I. Scabai, F. Czakó and Z. Fodor, "Quark and Gluon Jet Separation using Neural Networks", ITP Budapest Report 477 (1990).

[15] P. Bhat, L. Lönnblad, K. Meier and K. Sugano, "Using Neural Networks to Identify Jets in Hadron-Hadron Collisions", *Proc. of the 1990 DPF Summer Study on High Energy Physics Research Directions for the Decade, Colorado, 1990.*

[16] B. Denby, et al., "Neural Networks at the Tevatron", FERMILAB-CONF-92-269-E (Oct 1992).

[17] L. Bellantoni, J. Conway, J. Jacobsen, Y.B. Pan and S.L. Wu, "Using Neural networks with Jet Shapes to Identify b Jets in e^+e^- Interactions", CERN-PPE/91-90 (submitted to Nuclear Instruments and Methods A).

[18] B. Brandl et al., *Proceedings of the Second International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics, January 1992, L'Agelonde, France*, eds. D. Perret-Gallix, World Scientific (Singapore 1992);F. Seidel, *ibid*; J. Jousset, *ibid.*

[19] H. Baer, D.D. Karatas and G.F. Giudice, "Snagging the Top Quark with a Neural Net", *Physical Review* **D46**, 4901 (1992).

[20] J. Proriol, these Proceedings.

[21] B. Brandl et al., "Multivariate Analysis Methods to Tag b Quark Events at LEP/SLC", PCCF-RI 92-02 (to appear in *Nuclear Instruments and Methods A*).

[22] K.H. Becks, F. Block, J. Drees, P. Langefeld, and F. Seidel, "B-quark Tagging using Neural Networks and Multivariate Statistical Methods - a Comparison of Both Techniques", Wuppertal Preprint *WU B 93-4*.

[23] M. Holler, S. Tam, H.Castro and R. Beson, "An Electrically Trainable Artificial Neural Network (ETANN) with 10240 "Floating Gate" Synapses". In *IJCNN International Joint Conference on Neural Networks*, II:191-196 (1989).

[24] B. Denby et. al., "Neural Networks for Triggering", FERMILAB-CONF-90/20 (1990); see also B. Denby, these Proceedings.

[25] A. Lapedes and R. Farber, "Nonlinear Signal Processing using Neural Networks: Prediction and System Modeling", Los Alamos Report LA-UR 87-2662 (1987).

[26] A. Weigend, B.A. Huberman and D. Rumelhart, "Predicting the Future: A Connectionist Approach", *International Journal of Neural systems* **3**, 193 (1990).

[27] L. Lönnblad, C. Peterson and T. Rögnvaldsson, "Mass Reconstruction with a Neural Network", *Physics Letters* **B278**, 181 (1992).

[28] J. Alitti et al., "A Measurement of Two-jet Decays of the W and Z bosons at the CERN $\bar{p}p$ Collider", *Zeitschrift für Physik* **C 49**, 17 (1991).

[29] T. Sjöstrand, JETSET 7.3 program and manual see e.g. B. Bambah et. al., QCD Generators for LEP, CERN-TH.5466/89 (1989);
T. Sjöstrand, "The Lund Monte Carlo for Jet Fragmentation and e^+ e^- Physics - JETSET version 6.2", *Computer Physics Communications* **39**, 347 (1986);
T. Sjöstrand and M. Bengtsson, "The Lund Monte Carlo for Jet Fragmentation and e^+ e^- Physics - JETSET version 6.3 - An Update", *Computer Physics Communications* **43**, 367 (1987).

[30] J. MacQueen. "Some Methods for Classification and Analysis of Multivariate Observations", in L. M. LeCam and J. Neyman (eds.), *Proc. 5th Berkeley Symposium on Math. Stat. and Prob.*, U. California Press, Berkeley (1967).

[31] T. Kohonen, "Self Organized Formation of Topologically Correct Feature Maps", *Biolological Cybernetics* **43**, 59 (1982);
T. Kohonen, *Self-organization and Associative Memory*, third edition, Springer-Verlag, Berlin, Heidelberg (1990).

[32] T. Rögnvaldsson, "Pattern Discrimination Using Feed- forward Networks - a Benchmark Study of Scaling Behaviour", *LU TP 92-18* (to appear in *Neural Computation.*

[33] L. Lönnblad, C. Peterson, H. Pi and T. Rögnvaldsson, "Self-organizing Networks for Extracting Jet Features", *Computer Physics Communications* **67**, 193 (1991).

[34] J.J. Hopfield and D.W. Tank, "Neural Computation of Decisions in Optimization Problems", *Biological Cybernetics* **52**, 141 (1985).

[35] C. Peterson, "Parallel Distributed Approaches to Combinatorial Optimization Problems - Benchmark Studies on TSP", *Neural Computation* **2**, 261 (1990).

[36] C. Peterson and B. Söderberg, "A New Method for Mapping Optimization Problems onto Neural Networks", *International Journal of Neural Systems* **1**, 3 (1989).

[37] R. Durbin, and D. Willshaw, "An Analog Approach to the Travelling Salesman Problem Using an Elastic Net Method", *Nature.* **326**, 689 (1987).

[38] C. Peterson and B. Söderberg, "Artificial Neural Networks and Combinatorial Optimization Problems", *LU TP 92-15* (to appear in *Local Search in Combinatorial Optimization*, eds. E.H.L. Aarts and J.K. Lenstra, New York 1992: John Wiley & Sons).

[39] B. Denby, "Neural Networks and Cellular Automata in Experimental High Energy Physics", *Computer Physics Communications* **49**. 429. 1988.

[40] C. Peterson, "Track Finding with Neural Networks", *Nuclear Instruments and Methods* **A279**, 537 (1989).

[41] G. Stimpfl-Abele and L. Garrido, "Fast Track Finding with Neural Nets", *Computer Physics Communication* **64**, 46 (1991).

[42] L. Gislen, C. Peterson and B. Söderberg, "Rotor Neurons - Formalism and Dynamics", *Neural Computation* 4, 727 (1992).

[43] C. Peterson, "Neural Networks and High Energy Physics", *Proc. of International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics, Lyon Villeurbanne, France, March , 1990*, eds. D. Perret-Gallix and W. Woijcik, Editions du CRNS (Paris 1990).

[44] M. Gyulassy and H. Harlander, "Elastic Tracking and Neural Network Algorithms for Complex Pattern Recognition", *Computer Physics Communications*, **66** 31 (1991).

[45] A. Yuille, K. Honda and C. Peterson, "Particle Tracking by Deformable Templates", *Proceedings of 1991 IEEE INNS International Joint Conference on Neural Networks*, Vol. 1, pp 7-12, Seattle, WA (July 1991).

[46] M. Ohlsson, C. Peterson, and A. Yuille, "Track Finding with Deformable Templates - The Elastic Arms Approach", *Computer Physics Communications* **71**, 77 (1992).

[47] M. Ohlsson, "Extensions and Explorations of the Elastic Arms Algorithm", *LU TP 92-28* (submitted to *Computer Physics Communications*)

[48] P.J. Huber, *Robust Statistics*, John Wiley and Sons (New York 1981).

[49] R. Blankenbecler, in preparation.

[50] M. Regler and R. Frühwirth, "Reconstruction of Charged Tracks", *Proc. Advanced Study Institute on Techniques and Concepts in High Energy Physics, St. Croix*, Plenum, Rochester (1989).

[51] G. Ososkov, these Proceedings; A.A. Glazov et, al., Dubna preprint *JINR E10-92-352.*

[52] L. Lönnblad, C. Peterson and T. Rögnvaldsson, "Pattern Recognition in High Energy Physics with Artificial Neural Networks - JETNET 2.0", *Computer Physics Communications* **70**, 167 (1992) [Program and manual available via Email request].