A Recurrent Neural Network for Track Reconstruction in the LHCb Muon System

Giovanni Passaleva

Abstract–**In this work we describe an algorithm for muon track reconstruction in the LHCb Muon System. The algorithm is based on a recursive neural network known as Hopfield Network. The algorithm is particularly suitable in the harsh LHCb environment, where the expected number of hits per event in the muon detector is of the order of 2-3*10^3. The algorithm has been tested successfully on cosmic ray tracks, collected during the commissioning phase of the LHCb Muon System, and on simulated proton-proton collision events, showing excellent performances. The algorithm is currently used for standalone muon reconstruction in the Muon System online monitoring.**

I. INTRODUCTION

In the last two decades, neural networks have been widely used in high energy physics as they proved to be a very powerful analysis tool where large numbers of highly correlated physical observables are involved. A neural network is a set of processing units, the neurons, which are connected to each other. The status of each neuron depends on the status of all the other neurons connected to it and on the strength, or *weight*, of the connection. The most widely used class of neural networks is that of the *feed-forward* networks. These networks must be *trained* through a set of templates and then they are able to recognize certain features on the basis of what they have "learned". This class of neural networks is not well suited for track reconstruction because the number of possible patterns to be recognized is too large. For optimization problems, like track reconstruction, is instead better to use the so called *recurrent neural networks*. In a recurrent neural network, all the neurons are connected to each other. The classical model is due to Hopfield [1]. The network is made of a single layer of neurons (Fig. 1) all interconnected through a symmetric potential $w_{ij} = w_{ji}$ without auto–interactions (w_{ij} =0). The state of the *i-th* neuron is given by:

$$s_i(t+\Delta t) = sign\left(\sum_{ij} w_{ij} s_j(t) - \theta_i\right) \qquad (1)$$

where θ_i is an activation threshold. It can be shown [2] that with this update rule, the network evolves to a local minimum of the energy function

$$E = -1/2 \sum_{ij} w_{ij} s_i s_j + \sum_i s_i \theta_i \qquad (2)$$

As tracking is an optimization problem, we are interested in finding the global minimum of the energy function (2). It is therefore useful to put the system in a thermal bath at temperature T. In this case the states of the network have a Boltzmann distribution.

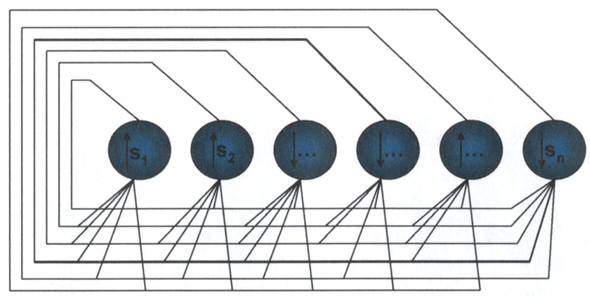

Fig. 1 Schematic diagram of a Hopfield network

It is possible to demonstrate that in the mean field theory approximation [3], the update rule (1) becomes:

$$v_i(t+\Delta t) = \tfrac{1}{2}\left[1 + \tanh\left(\frac{1}{T}\sum_{ij} w_{ij} v_j(t)\right)\right] \qquad (3)$$

$$v_i = \langle s_i \rangle_T$$

where v_i is the thermal average of the neuron status s_i. In this model, the status of the neurons is updated in a deterministic way. Fully stochastic evolution models exist, like the so called Boltzmann Machine, but they are extremely slow from the computing point of view.

Equation (3) describes a *synchronous* evolution of the network because at any time $t+\Delta t$ the status of the neurons depends on the status of all the neurons at a previous time t. An alternative approach is an *asynchronous* evolution, where the status of every neuron is updated in turn. The network evolution proceeds until the absolute minimum of the energy is reached. Typically a convergence condition is imposed like:

$$1/N_{neurons} \sum_i \left| s_i(n) - s_i(n-1) \right| < \delta \qquad (4)$$

where n is the iteration number and δ is the convergence parameter. The "answer" of the network is given by the final set of *active* neurons, i.e. those whose status is above the activation threshold θ_i.

II. MUON TRACK RECONSTRUCTION IN LHCb

The LHCb Muon System [4][5] consists of five stations named M1–M5, equipped with Multi Wire Proportional

Manuscript received November 14, 2008.
G. Passaleva is with the Istituto Nazionale di Fisica Nucleare – Sezione di Firenze, Florence, 50019 Italy (telephone: +39-0554572596, e-mail: giovanni.passaleva @fi.infn.it).

978-1-4244-2715-4/08/$25.00 ©2008 IEEE

Chambers[1]. The five stations are divided into four Regions with different space granularities depending on the distance from the interaction point and from the beam line.

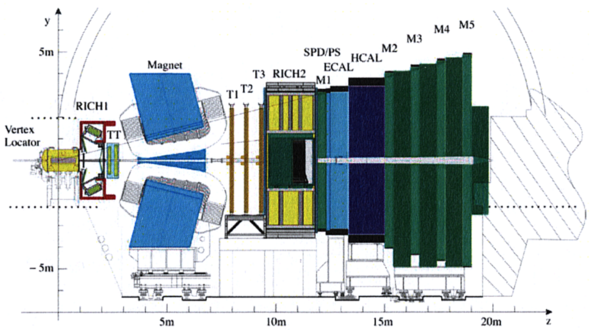

Fig. 2 The LHCb experiment. The Muon System is in green.

The tracks of the muons traversing the detector are straight lines roughly pointing to the interaction vertex. In the bending plane (xz plane) the pointing is very approximate due to the bending effect of the magnetic field. The track reconstruction problem therefore amounts to reconstructing straight lines in the two planes xz and yz, crossing the five Muon Stations.

A. Definition of neurons

In the framework of the model outlined in the introduction, it is useful to define a neuron as an oriented segment (vector) connecting two hits in the Muon System belonging to different Stations. By convention the neurons point in the outward direction from the interaction point. With these definitions, ideally a track is identified by a set of consecutive neurons connecting a trail of aligned hits. The neurons are characterized by a "tail" and a "head". The angle among two neurons is defined by their scalar product; the neurons are also characterized by their polar angles θ_{xz} and θ_{yz} in the xz and yz planes. Following the approach in [6]–[9], let us indicate with k and l the coordinates of the tail and the head of neuron i and with m,n those of neuron j. With this notation, the energy function (2) can be rewritten as:

$$E = -1/2\sum_{klmn}\left(w_{klmn} - bC_{klmn}\right)v_{kl}v_{mn} + \sum_i \theta_{kl}v_{kl} \quad (5)$$

where C_{klmn} is a constraint matrix and b is a Lagrange multiplier. We want only connections between neurons sharing a common hit, therefore:

$$w_{k\ln} = \delta_{lm} w_{klmn}. \quad (6)$$

Moreover, we do not want hits associated to more than one track (bifurcations are not allowed), thus:

$$C_{klmn} = \delta_{km}\left(1-\delta_{\ln}\right) + \delta_{\ln}\left(1-\delta_{km}\right). \quad (7)$$

With these definitions, (5) can be rewritten as

$$E = -\tfrac{1}{2}\Big[\sum_{k\ln} w_{k\ln}v_{kl}v_{\ln} - b\times \times\Big(\sum_{k\ln(n\neq l)} v_{kl}v_{kn} + \sum_{klm(m\neq k)} v_{kl}v_{ml}\Big)\Big] + + \sum_i \theta_{kl}v_{kl} \quad (8)$$

and the neuron update rule becomes:

$$v_{kl} = \tfrac{1}{2}\left[1 + \tanh\left(\frac{1}{T}\sum_n w_{k\ln}v_{\ln} - -\frac{b}{T}\left(\sum_{n\neq l} v_{kn} + \sum_{m\neq k} v_{ml}\right)\right)\right]. \quad (9)$$

The terms multiplied by the positive constant b are penalty terms that suppress configurations where the neurons originate from or enter into the same hit.

B. Definition of weights

The definition of weights is the most critical part of the network construction. Ideally the weights should enhance connections among neurons that are well aligned and suppress configurations where two neurons enter in or exit from the same hit to avoid track bifurcations. Given the geometry of the Muon System and following the suggestion of [10], a good weight definition is given by:

$$w_{ij} = a\left(1-\sin\theta_{ij}\right)^{\beta}\left(1-\sin\varphi_{ij}\right)^{\gamma} \quad (10)$$

where θ_{ij} and φ_{ij} are the angles between the neurons i and j in the xz and yz planes, a is a positive constant and β and γ are two large exponents of the order of 10 or more.

To suppress track bifurcations, neurons that are connected "head–head" or "tail–tail" (i.e. coming from or entering into the same hit) are assigned a negative constant weight b. This can in general be different for head–head and tail–tail configurations. In fact, as the number of hits decreases going from M1 to M5, there are generally many more head–head configurations than tail–tail ones. As it is clear from the network behavior outlined in the introduction, negative weights can in principle switch off good neurons in an unwanted way. Therefore the balance between the positive and negative weights must be carefully tuned. Owing to the different granularities, the value of the exponents β and γ must in general be different for different Stations and Regions. Moreover, too large exponents would result in a high, momentum dependent, tracking inefficiency due to the large multiple scattering in the Muon System.

C. Network initialization

In a typical proton–proton collision, $\mathcal{O}(10^3)$ hits are released in the LHCb Muon System. Connecting all the possible hit pairs belonging to different Stations would result in an unaffordable number of neurons to deal with in the algorithm. On the other hand, normally, a good track should have an hit in each crossed Station so that in principle it is enough to connect only hits belonging to consecutive Stations, greatly reducing the number of created neurons. For physics events this is the basic preselection cut while for simple events like

[1] The innermost part of station M1 is equipped with GEM detectors.

cosmic rays this requirement can be released. The number of useful neurons can be further reduced exploiting the projectivity of tracks with respect to the beam interaction point. Therefore, only neurons forming a maximum angle θ_{xz}^{max} and θ_{yz}^{max} are retained. The value of θ_{xz}^{max} and θ_{yz}^{max} can depend in general from the Station and the Region, to allow for the different granularities and for the increasing multiple scattering going from M1 to M5. Moreover, θ_{xz}^{max} should be generally larger than θ_{yz}^{max} to take into account the effect of the magnetic field in the bending plane.

In order to balance the effect of negative and positive weights, the method described in [9] is followed. For every neuron, only the highest weight head–tail and the highest weight tail–head companions are kept, while all the other head–tail and tail–head weights are set to zero.

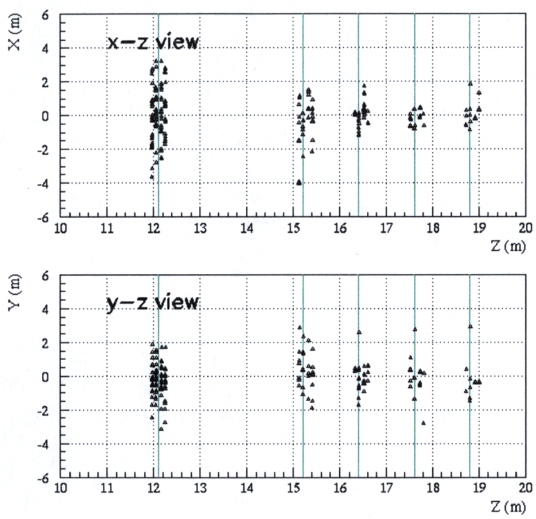

Fig. 3 Hits in the five Muon Stations from a $pp \rightarrow bb \rightarrow J\,\psi(\mu\mu)X$ simulated event.

D. Network evolution

After all the good neurons have been built and the corresponding weights defined, the network status is recursively updated until a global minimum is found. Both the synchronous and the asynchronous neuron update mechanisms have been studied. The synchronous approach tends to be too slow and often converges to local minima due to the strong correlation between the states at different times. Therefore, like in [9] the asynchronous method is followed here. By default all created neurons are switched on at the beginning. A random initial configuration has been studied as well; this

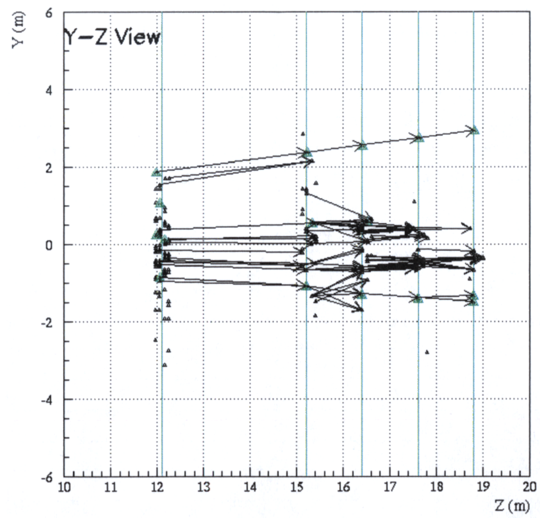

Fig. 4 Neurons created from the hits in Fig. 3 (yz plane). Green symbols are hits created bi muons.

solution however brings quite often to the unwanted suppression of good neurons. This is explained by the small number of neurons that generally form a track (for a good track traversing the five stations one expects only four neurons): if one or more good neurons are off at the beginning, it's very difficult or even impossible that they get switched on during the network convergence.

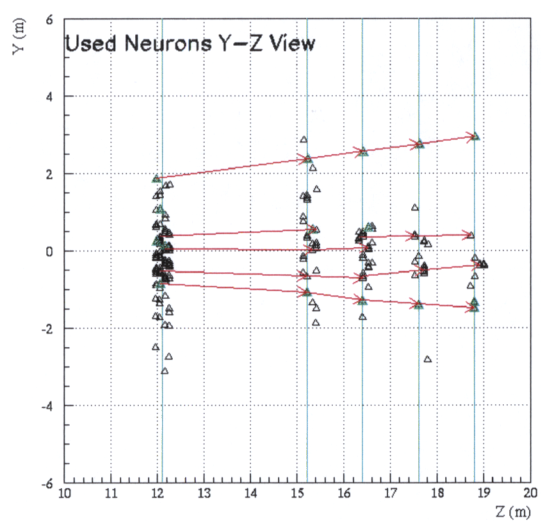

Fig. 5 Active neurons after the network convergence (yz plane). Green symbols are hits created by muons

The neuron status is updated according to equation (9), with the convergence condition (4)

In Table I, the values of the network parameters used in this study are summarized. Fig.3 shows the distribution of the hits in a typical $pp \rightarrow bb \rightarrow J/\psi(\mu\mu)X$ event from Monte Carlo simulation. In Fig. 4 the neurons created from the above hits are displayed while those remaining active after the network convergence are shown in Fig. 5. In all the previous figures, the green symbols represent the hits created by a muon.

TABLE I
NETWORK PARAMETERS

Parameter	value
a	8
b	5
β	5
γ	10
θ	0.5
T	5
δ	10^{-5}

The evolution of the neuron status and of the network energy are shown in Fig. 6 and Fig. 7.

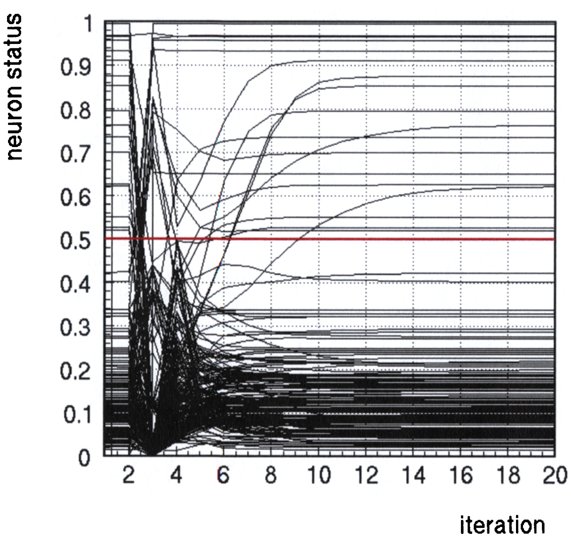

Fig. 6 evolution of the neuron status for the event in Fig. 3

E. Track selection and fitting

Once the minimum energy of the network has been identified, all the sets of neurons connected together are considered as track candidates. Only track candidates with at least three hits are considered for further processing. The hits of the selected track candidates are fitted to a simple straight line with a minimum χ^2 procedure separately in the two xz and yz projections and the resulting straight tracks are available for further analysis.

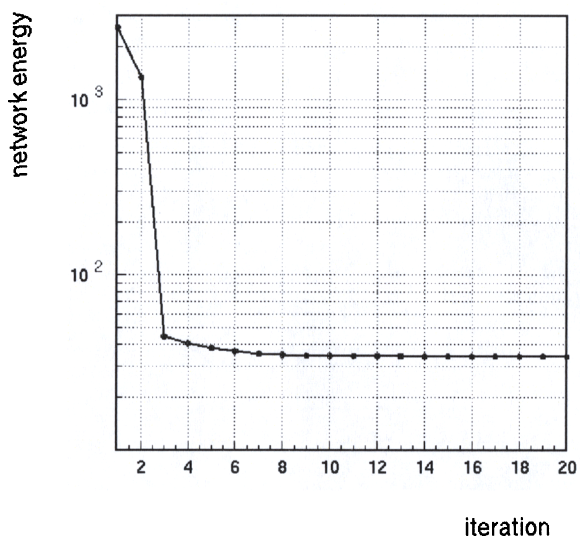

Fig. 7 Evolution of the network energy for the event in Fig. 3

III. PERFORMANCES

The performance of the neural network algorithm has been studied with simulated $pp \rightarrow bb \rightarrow J/\psi(\mu\mu)X$ events. The track reconstruction efficiency is about 97% including an implicit muon momentum cut at about 6 GeV/c due to the thick iron absorbers placed between the Muon Stations. For tracks with momentum larger than 10 GeV/c the reconstruction efficiency exceeds 99%.

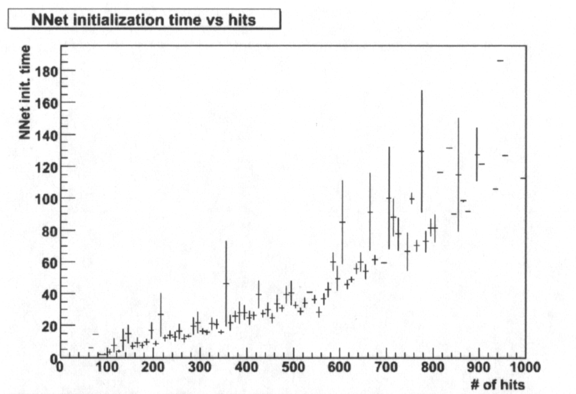

Fig. 8 Network initialization time as a function of the number of hits. Time is in milliseconds.

Tests of CPU time required by the neural network algorithm have been performed on a 2.8 GHz Xeon CPU. No attempt to optimize the algorithm from the point of view of the computing speed has been done. As shown in Fig. 8 and Fig. 9 the network initialization is the heaviest part from the CPU time point of view. As expected the computation time increases roughly quadratically with the number of hits in the event. The network convergence is instead rather fast and the computation time is approximately linear in the number of hits. The number of iterations required for the network convergence

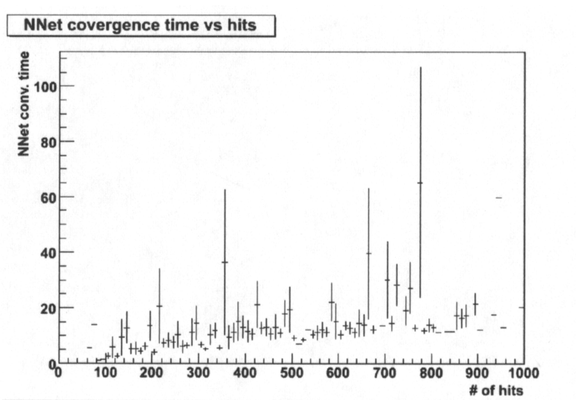

Fig. 9 Network convergence time as a function of the number of hits. Time is in milliseconds.

is shown in Fig. 10: the network usually converges in about 20-30 iterations with some tail which is potentially dangerous if computing speed is to be kept low. Even at the highest number of hits the algorithm requires less than 200 ms to reconstruct an event. Therefore, even though some optimization is certainly needed, the algorithm is suitable also for an application in an online monitoring task.

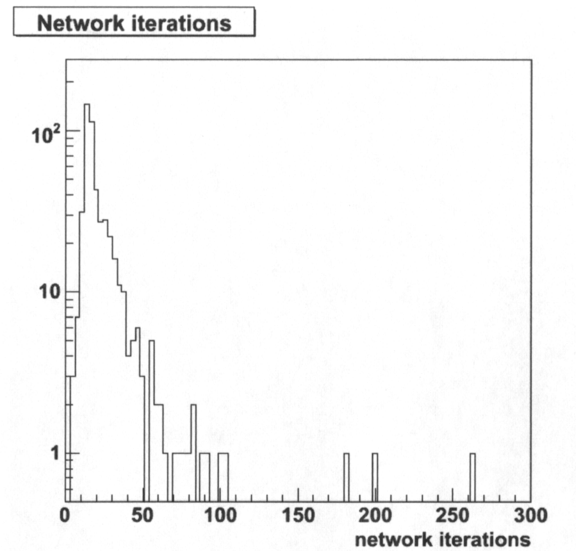

Fig. 10 number of iterations needed for network convergence.

IV. CONCLUSIONS

A track reconstruction algorithm for the LHCb Muon System based on a recurrent neural network has been described. The algorithm has a reconstruction efficiency of about 97%, but exceeds 99% for high momentum tracks. The CPU time required is relatively small and is compatible with both an offline and an online reconstruction task. A simplified version of the network discussed in this paper has been successfully used in the LHCb Muon System online monitoring for cosmic ray tracks reconstruction.

REFERENCES

[1] J J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities", *Proceedings of the National Academy of Sciences*, USA 78, pp. 2554-2558, reprinted in I. A. Anderson and E. Rosenfeld, eds, "Neurocomputing", Foundation of Research, Cambridge, M.I.T. Press 1988.
[2] J..J. Hopfield, D.W. Tank, *Biol. Cybernetics*, vol. 52, pp. 141., 1985.
[3] C. Peterson, J.R. Anderson, *Complex Systems*, vol. 1, pp. 995, 1987.
[4] LHCb Collaboration, "LHCb Muon System Technical Design Report", *CERN-LHCC* 2001-010, 2001; and "Addendum to the Muon System Technical Design Report", *CERN-LHCC* 2003-002, 2003.
[5] A.A. Alves *et al.*, LHCb Collaboration, "The LHCb Detector at the LHC", *JINST*, vol. 3, S08005, 2008.
[6] B. Denby, *Comput. Phys. Commun.*, vol. 49, pp. 429, 1988.
[7] B. Denby, S.L. Linn, *Comput. Phys. Commun.*, vol. 56, pp. 293, 1990.
[8] C. Peterson, *Nucl. Instrum. Meth. A*, vol. 279, pp. 537, 1989.

[9] G. Stimpfl-Abele and L. Garrido ,"Fast track finding with neural nets" *Comput. Phys. Commun.*, vol. 64, pp. 46, 1991.
[10] A. Pulvirenti, A. Badalà, R. Barbera, G. Lo Re, A. Palmeri, G.S. Pappalardo and F. Riggi, "Neural tracking in the ALICE Inner Tracking System", *Nucl. Instrum. Meth. A*, vol. 533, pp. 543, 2004.