ALEPH 93-152
PHYSIC 93-132
J. Proriol
22.9.1993

MULTI-MODULAR NEURAL NETWORKS FOR THE CLASSIFICATION OF E^+E^- HADRONIC EVENTS

J.PRORIOL
Laboratoire de Physique Corpusculaire
CLERMONT-FERRAND

ABSTRACT:Some multi-modular neural network methods of classification of e^+e^- hadronic events are presented.We compare the performances of the following neural networks:MLP (multi-layer perceptron),MLP and LVQ (learning vector quantization) trained sequentially,MLP and RBF (radial basis function) trained sequentially.We will introduce a MLP-RBF cooperative neural network.Our last study is a multi-MLP neural network.

1-INTRODUCTION

Various researchers have recently advocated the use of multi-modular neural networks [1,2,3,4,5,7],various networks were used:Multi-Layer Perceptrons(MLP),Time Delay Neural Networks(TDNN),Learning Vector Quantization(LVQ) or Radial Basis Functions(RBF).The most common applications have been character recognition[7] or speech [1,2,5].The idea underlying these efforts is to get improved performances from the cooperation of networks:for example,a MLP is first used on the raw noisy data,then a LVQ or a RBF network is trained on the MLP-extracted features and will be able to perform a better classification than the last layer of the MLP.

Most often these architectures are trained sequentially,one module after the other.However,such a training procedure is sub-optimal:global optimality can be obtained by cooperatively training all modules together [4].

Our aim is to classify very noisy data coming from High Energy Physics (HEP) problems [6]:events produced by the collisions of particles (e^+e^-) at the LEP accelerator at CERN

(Accepted by NUCLEAR INSTRUMENTS AND METHODS A)

(Geneva) and detected by the detector ALEPH.In a work reported previously [9],we compared the performances of conventional techniques (cut on the most discriminating variable,Discriminant Analysis,and various Neural networks:MLP and LVQ).

In this paper,we compare the performances of the following neural networks:MLP,MLP and LVQ trained seqentially,MLP and RBF trained sequentially .We will introduce a MLP-RBF cooperative neural network:the first part of the processing is done by the first layers of a MLP;the final classification is done by a RBF.Instead of using the usual initialization by a k-means method [7],the initialization is done with small random numbers.Our model is also different from [8]:we first use a MLP and then a RBF structure.Our last multi-modular neural network is a multi-MLP network.

In appendix,the algorithm of cooperative MLP and RBF is given.

2-MLP

Our main study is the reation at LEP energy:

$$e^+e^- \rightarrow Z_0 \rightarrow \text{quarks} \rightarrow \text{hadrons}.$$

With this reaction,it is easy to classify and study b quark events,but the study of the c quark events is also interesting.Then we consider the 3 classes:b quark events,c quark events and light quark events.The training of these 3 classes is difficult with natural training sets,the choice of a good training to classify c quark events has to be done.In all our attempts we'll take 10000 events for each class.

It is well known that a MLP classifies well very noisy data.The MLP is then used as classifier but can be used to extract the features and to feed another neural network.The architecture of the MLP is a 4 layers MLP:20 input neurons,two hidden layers with 20 neurons and 8 neurons,the output layer is a 3 neurons one.The outputs of the last hidden layer can then be used to feed a new neural network.

We have generated 150 variables for (e^+e^- -> hadrons) ALEPH events.Some variables are very classical ones and describe the shape of the event [10];the variables connected with the vertex detector data are very important [11].Some

other variables were designed to study neural network correlated problems [12,14].A Monte-Carlo program simulates fully the detector;the Monte-Carlo variables are very similar to data variables.The training set used 10000 events for each class.

The selection of the variables is difficult because their number of variables is important, the F-test method[9] was used to decrease the number of useful variables.The most discriminating variable is connected with the data of the vertex detector [11].The classification of the variables is done according to the F-test values and we eliminate too strongly correlated variables:we have kept 20 variables to feed the MLP[14].

The results of this study has already been presented in reference[14].The definition of some values: purity, efficiency,TOP coefficients were presented in some earlier papers[13,14,15].

In table 1 we recall some of these results.The different values are computed from the classification matrix:an event is classified to the class of the ouput neuron with the larger output value.When we consider 2 classes (b and udsc classes),the matrix is a 2x2 matrix;with 3 classes (b,c,uds classes),we get a 3x3 matrix.

3-MLP+LVQ

The number of outputs for a MLP is equal to the number of classes.We know other neural networks with improved outputs;it is the case of LVQ[16].

3-1 Theory

The LVQ neural network classifies correctly low dimensional and noisy free data.The high energy physics events have to be processed to lower the noise of the data:this is done by the 3 first layers of a 4 layers MLP.The MLP output layer is replaced by a LVQ.In our MLP study,we have chosen a 20-20-8-3 MLP network;the 8 values of the last hidden layer of the MLP network are the components of a vector $\mathbf{x}$.Some reference vectors $\mathbf{m}_i$ are chosen;several vectors can be assigned to each class.During the training,we compute the argument

$$j=\arg \mathrm{Min}_i \; \|\mathbf{x}-\mathbf{m}_i\|^2,$$

the vectors $\mathbf{m}_i$ are moved according LVQ1 and LVQ2.1 rules [7,16].This is a sequential training.In figure 1, we give the architecture of such a model.

During the test, the class of an event **x** is affected to the class of the nearest reference vector $\mathbf{m}_i$.

3-2 Initialization

The LVQ method works very well if a good initialization is done.In previous works,the k-means method was used [7].The moving center method [17],similar to k-means, was used in this work.Some reference vectors,usually chosen as the first events of the training set for each class, are initially affected,and gradually,these vectors are replaced by the the centers of gravity computed with the vectors of the learning sample.The convergence is very quick.

3-3 MLP+LVQ1

In a first attempt,the LVQ1 neural network was initialized with the moving center method with 21 reference vectors with 8 dimensions; 7 vectors are affected to each class.

The training was done with 10000 events for each class,the 8 variables for each event were taken from the third layer of the MLP.

The test was done with a sample of 73376 events.The percentage of well classified events for 3 classes (b,c,uds classes) is 72.61%,and when we consider only 2 classes (b and udsc classes),it is 91.39%.

3-4 MLP+LVQ1+LVQ21

After the training of the LVQ1, the training of a LVQ2.1 is done with those reference vectors.The percentage of well classified events for 3 classes is 73.73%,and 91.59% for 2 classes.The performances of the network are improved.

4-MLP+RBF

Another way to improve the MLP results is to use a Radial Basis Function (RBF) network instead of a LVQ neural network.

4-1 Theory

RBF networks are multi-layer networks with a hidden layer of localized gaussian units.The output is fully connected to the hidden layer.The computation of the outputs is done according to the following relations:

$$a_i = \sum_j R_{ij} exp(-\frac{\|\mathbf{x}-\mathbf{x}^j\|^2}{(\sigma^j)^2})$$

where i is for the output units, $\mathbf{x}^j$ is the center of unit j and σ^j its width. We can normalize the a values and compute

$$b_i = a_i / (\sum_j a_j) \qquad (inorm=1)$$

The output O_i is computed by the relation

$$O_i = F(b_i) \text{ or } O_i = F(a_i)$$

where F can be a linear relation *(ifonc=0)* or a sigmoid relation *(ifonc=1)*.

In figure 2, the architecture of the sequential MLP and RBF is presented. The training algorithm of the RBF neural network is the second part of the MLP-RBF algorithm given in appendix.

4-2 Initialization

The $\mathbf{x}^j$ values are fixed in taking the vectors outputs of the last hidden layer of the MLP network. The σ^j values are taken equal to 0.75. The training of the R_{ij}, $\mathbf{x}^j$ and σ^j parameters is then done.

4-3 Results

With the same test set as before, the percentage of well classified events is 91.51% for 2 classes and 73.34% for 3 classes.

5 MLP-RBF (cooperative)

A sequential training is not generally optimal. Global optimality can be obtained by cooperatively training the MLP and the RBF modules.

5-1 Theory

The architecture is built up from two modules. First a MLP, with *l* layers of neurons; the MLP used for classification would have *l+1* layers, but here we have stripped the last layer, and only retained the features computed by the MLP which are used as inputs to the RBF. The RBF has two layers, the first one is a set of gaussian kernels, and the other one is the standard MLP output layer.

The feed-forward calculations are done according to the standard MLP and RBF equations; we propagate backwards to the RBF and the first layer of MLP the error computed on the output

layer.These relations are given in appendix and are based on standard methods of MLP back-propagation [18].

The initialization of the different weights is done randomly with small values;the σ values are initialized with a 0.75 value.

5-2 Results

The best results obtained with MLP-RBF give 73.51% for the percentage of classification for 3 classes and 91.69% for the classification into 2 classes.This result is the best one obtained with 2 networks.On figures 3 to 6 we give the ouputs of MLP-RBF and the curves of purity/efficiency.

6- THREE MLP

The last method uses a multi-network architecture:3 parallel MLP networks.

6-1 Theory

The 3 classes strongly overlap.To get a more efficient classification,it is interesting to follow a method used for the handwritten digits recognition [19]. The training is done for a set of groups of 2 classes:training b/c,training b/uds,training c/uds.The sets of variables are different for the 3 sets,and the outputs of the 3 MLP are different.During the test,the combination of the 3 outputs is done to get the final classification.On figure 7 the architecture of the system for the training and the test is presented.

In table 2 are presented the F-test values for the 3 sets of variables.The classification b/uds gives high F-test values,and then it is easy to distinguish a b event from an uds event.The c and uds classes are very similar.

6-2 Results

The best results were obtained in doing the product of the outputs of the 2 classes MLP.

The percentage of good classification is 73.73% for 3 classes and 91.91% for 2 classes.

7- CONCLUSION

The comparison of different types of multi-modular neural networks has been done and is presented in table 1.The most efficient ones seem the 3-MLP neural network and the cooperative MLP-RBF network;but the latter is easier to use than the 3-MLP network:the MLP-RBF is run like a MLP one.

8-REFERENCES

[1]Y.BENGIO,R.DE MORI:Glogal optimization of a neural-hidden Markov model hybrid.*Trans.on neural networks 3(1992)252*

[2]Y.BENNANI,P.GALLINARI:A modular connectionist architecture for text-independant talker identification.*IJCNN'91 vol 2,p 857*

[3]M.de BOLLIVIER,P.GALLINARI,S.THIRIA:Cooperation of neural networks and task decomposition.*IJCNN'91,Seattle,vol 2 p573*

[4]L.BOTTOU,P.GALLINARI:A framework for the cooperation of learning algorithms.In *Neural Information processing systems,NIPS'90,R.P.LIPPMANN,J.E.MOODY,D.S.TOURETZKY eds,Morgan Kaufmann,vol 3 (1991) p 781*

[5]H.BOULARD,N.MORGAN:A continuous speech recognition embedding MLP into HMM.*NIPS'89,D.Touretzky ed,Morgan Kaufman,vol 2 (1990) p 186*

[6]B.DENBY:The use of neural networks in high energy physics.*Neural Computation,(to appear)*

[7]F.FOGELMAN-SOULIE,B.LAMY,E.VIENNET:Multi-modular neural network architecture for pattern recognition:applications in optical character recognition and humanface recognition.*Int. Jl Pattern Recognition and Artificial Intelligence(to appear)*

[8]S.LEE: In Neural networks for signal processing,*B.Kosko ed,Prentice Hall,Englewood Cliffs (1992)*

[9]J.PRORIOL,J.JOUSSET,C.GUICHENEY,A.FALVARD,P.HENRARD,D.PALLIN ,P.PERRET:Tagging b quark events in ALEPH with neural networks.*In Proceedings of the Elba workshop:Neural networks:from biology to high energy physics. O.Benhar, C.Bosio, P.del Giudice,E.Tabet,eds. (1991)p 419*

B.BRANDL,A.FALVARD,C.GUICHENEY,P.HENRARD,J.JOUSSET,J.PRORIOL: Multivariate analysis methods to tag b-quark events at LEP/SLC *Nuclear Instruments and methods A324(1993)307*

[10]A. de RUJULA,J.ELLIS,E.G.FLORATOS,M.K.GAILLARD *NUCL PHYS B138(1978)37*

G.C. FOX,S.WOLFRAM *NUCL PHYS B149(1979)413*
S.BRANDT,U.D.DAHMEN *ZEIT PHYS C1(1979)61*
J.B. BABCOCK,R.E. CUTKOWSKY *NUCL PHYS B179(1980)113*
F.BARRIERO,S. BRANDT,M. GEBSER *NUCL INST METH A240(1985)237*
Sau Lan WU *NUCL PHYS B3(Proc supl) (1988)102*
P.HENRARD *Clermont Preprint PC/CT RI 88-08*
A. PUTZER and al *ALEPHnote 90-161*
[11]D.BROWN,M. FRANK Tagging b hadrons using track impact parameter *ALEPHnote 93-007*
[12]L.LÖNNBLAD,C.PETERSON,T.RÖGNVALDSON *NUCL PHYS B349(1991)675*
L.BELLANTONI and al *NUCL INST METH A310(1991)618*
[13]B.BRANDL,A.FALVARD,C.GUICHENEY,P.HENRARD,J.JOUSSET, J.PRORIOL
Multivariate analysis methods to tag b-quark events at LEP/SLC *NUCL INST METH A324(1993)307*
[14]J.PRORIOL Classification of hadronic events in e^+e^- collisions:applications to tag the quark and the number of jets *NUCL INST METH to appear*
[15]K.H.BECKS,F.BLOCK,J.BREES,P.LANGEFELD,F.SEIDEL:B-quark tagging using neural networks and multivariate statistical methods.A comparison of both techniques.
NUCL INST METH A329(1993)501
[16]T.KOHONEN:Self-organization and associative memory.
Springer-Verlag 3° edition Heideiberg (1989)
[17]G.SAPORTA:Probabilités,analyse des données et statistique
Editions Technip (Paris 1990)
[18]D.E.RUMELHART,J.L.Mc CLELLAND:Parallel distributed processing:explorations in the microstructures of cognition,
MIT Press,(1986)
[19]S.KNERR,L.PERSONNAZ,G.DREYFUS:From theory to silicon:an efficient procedure for the design of neural classifiers and its application to the automatic recognition of handwritten digits.*NEURO-NIMES'91,EC2,PARIS(1991)*

Acknowledgments.

This work was initiated by a suggestion of Professor F.FOGELMAN-SOULIE to apply to high energy phycics problems the multi-modular neural network architecture.Several discussions on neural networks turned out very stimulating.

This paper is a part of a work going on in the ALEPH collaboration:I would like to thank the physicists of the collaboration for their support.

APPENDIX

MLP-RBF ALGORITHM

The architecture of the neural network is a two modules one:a MLP module and a RBF one.The MLP generates features in its last hidden layer and these features are used by the RBF as inputs.

A MLP has L layers of neurons;the MLP used for classification has L+1 layers,but we have stripped here the last layer,and only retained the features computed by the MLP which are used as inputs to the RBF.Then a RBF has 2 layers of neurons:the first one is a set of gaussian kernels and the other is a standard MLP output layer.

Notations

Let us denote,for the MLP, W_{ij}^{λ} and T_i^{λ} the weights and bias of neuron i,in layer λ,x_i^{λ} its activation,y_i^{λ} its state and f_{λ} the transfer function,n_{λ} the number of neurons in layer λ.

The RBF activation of the first layer is x_i^{L+1},the state is y_i^{L+1};the activation of the output layer is x_i^{L+2},the state is y_i^{L+2},the transfer function is f_{L+2}.The m_{il} value is the weight vector of neuron i in layer L+1,the l value is taken in the interval l=1..n_L;the number of kernels is n_{L+1}, σ_i is the width of the kernel and the R_{kj} are the weights of neuron k in the output layer.The number of neurons in the output layer is n_{L+2}.

Feed-forward

The feed-forward activations of the MLP are given by:

$$x_i^{\lambda}=\sum_{j=1}^{n_{\lambda-1}} W_{ij}^{\lambda} y_j^{\lambda-1}+T_i^{\lambda} \qquad (\lambda=1..L)$$

$$y_i^{\lambda}=f_{\lambda}(x_i^{\lambda})$$

The states in RBF are computed with the formula:

$$x_i^{L+1}=\exp\left(-\frac{\sum_{l=1}^{n_L}(m_{il}-y_l^L)^2}{2\sigma_i^2}\right)$$

But the x values can be normalized.We denote by S the sum of the x in the first RBF layer:

$$S=\sum_{i=1}^{n_{L+1}} x_i^{L+1}$$

The outputs of this layer are for the 2 cases according to the option inorm:

inorm=1 then $y_i^{L+1}=x_i^{L+1}/S$

inorm=0 then $y_i^{L+1}=x_i^{L+1}$.

The activation of the output layer is

$$x_k^{L+2}=\sum_{j=1}^{n_{L+1}} R_{kj}y_j^{L+1}$$

The state of the output layer can be computed linearly or transformed by a transfer function f_{L+2}.According to the option ifonc

if ifonc=0 $y_k^{L+2}=x_k^{L+2}$,

if ifonc=1 $y_k^{L+2}=f_{L+2}(x_k^{L+2})$.

Back-propagation

The weights W, R, the m_i,the σ are modified in the following way[4,8].

The MLP and the RBF are trained to minimize the error

$$E(i)=(1/2)\sum_{k=1}^{n_{L+2}} c_k(\ t_k(i)-y_k^{L+2}(i))^2$$

where $y_k^{L+2}(i)$ is the output computed on layer L+2 for neuron k,when an example x(i) is presented on the input layer of MLP and $t_k(i)$ is the associated target ie x(i)class.The coefficient c_k is a weight correponding to a given class.

The different parameters can be represented in a column vector $\mathbf{V}_j=[W_{ij}^{\lambda},T_i^{\lambda},m_i,\sigma_i,R_{ik}]$.

The update parameter rule is:

$$\mathbf{V}_j^{new}=\mathbf{V}_j^{old}+\eta\Delta\mathbf{V}_j$$

where η is a positive constant named the training rate.

The different $\Delta\mathbf{V}_j$ values are given by the relation

$$\Delta V_j = -\frac{\partial E}{\partial V_j}$$

For the different parameters,we get the relations:

$$\Delta R_{kj} = c_k (t_k - y_k^{L+2}) f'_{L+2}(x_k^{L+2}) x_j^{L+1}$$

$$\Delta m_{jl} = \frac{x_j^{L+1}}{\sigma_j^2} (y_l^L - m_{jl}) \sum_{k=1}^{n_{L+2}} c_k (t_k - y_k^{L+2}) f'_{L+2}(x_k^{L+2}) R_{kj}$$

$$\Delta \sigma_j = \frac{x_j^{L+1}}{\sigma_j^3} \| y^L - m_j \| \sum_{k=1}^{n_{L+2}} c_k (t_k - y_k^{L+2}) f'_{L+2}(x_k^{L+2}) R_{kj}$$

The MLP is trained as usual,with the gradient-back-propagation rule [13],except that the error δ which is back-propagated from the RBF is:

$$\delta_j^L = -f'_L(x_j^L) \sum_{k=1}^{n_{L+1}} \Delta m_{kj}$$

$$\Delta W_{ji}^{L-1} = \delta_j^L . y_i^{L-1}$$

$$\Delta T_j^{L-1} = \delta_j^L$$

With these rules for parameters modification,the two modules are cooperatively trained [4],the architecture is globally optimal.The MLP is not trained to perform classification,but to derive the optimal features for the RBF.

Initialization

The MLP-RBF neural network has a structure very similar to the MLP one.The initialization can be done in the same way as a MLP.We first fix the structure of the network:number of layers,numbers of neurons.The weights W,R,m_{il} are initialized with small random numbers;the widths σ are initialized with constant values.This method is very simple and can be used by non specialists.

NAME	% 2 cl	% 3 cl	pur b	eff b	pur c	eff c	TOP b	TOP c	TOP uds
DISC ANALYS	91.0	67.2	85.8	71.0	29.6	54.6	0.98	0.33	-
MLP	91.4	71.2	81.3	79.4	33.6	51.7	0.98	0.44	0.93
MLP+LVQ1	91.3	72.6	81.0	79.3	34.9	48.6	0.97	0.45	0.93
MLP+LVQ1+LVQ21	91.5	73.7	82.3	78.4	36.0	46.1	0.98	0.45	0.92
MLP+RBF	91.5	73.3	81.9	78.5	35.6	47.2	0.98	0.44	0.93
MLP-RBF(coop)	91.6	73.5	82.7	78.5	35.9	49.0	0.98	0.45	0.93
3 MLP (prod)	91.9	73.7	84.0	77.8	35.9	48.4	0.98	0.44	0.93

Table 1:Results for different methods:

-Percentage of well classified events:2 classes

-Percentage of well classified events:3 classes

-Purity of the b and c sample events

-Efficiency of the b and c sample events

-TOP values for the 3 classes

The results are computed from the classification matrix.

variable N°	F-test 2*10000 b/c	F-test 2*10000 b/uds	F-test 2*10000 c/uds
1	7442	11022	1550
2	2647	5083	953
3	2508	4762	797
4	2483	3797	545
5	2186	3326	378
6	2070	3232	366
7	1422	2716	352
8	1083	1685	303
9	968	1675	295
10	761	1628	258
11	719	1389	250
12	718	1319	199
13	613	1180	145
14	605	999	121
15	529	710	121
16	502	660	102
17	494	644	101
18	469	590	98
19	448	484	93
20	444	408	65

Table 2:F-test values for the 3 sets of variables of 3-MLP

FIGURES CAPTIONS

Figure 1:Architecture of the MLP+LVQ neural network

Figure 2:Architecture of the MLP+RBF and the MLP-RBF neural networks

Figure 3:Histograms of the MLP-RBF neurons outputs for 3 classes:b quark events,c quark events,light quark(u,d,s) events

TE1:b quark events TE4:not b quark (u,d,s,d) events

TE2:c quark events TE5:not c quark (u,d,s,b) events

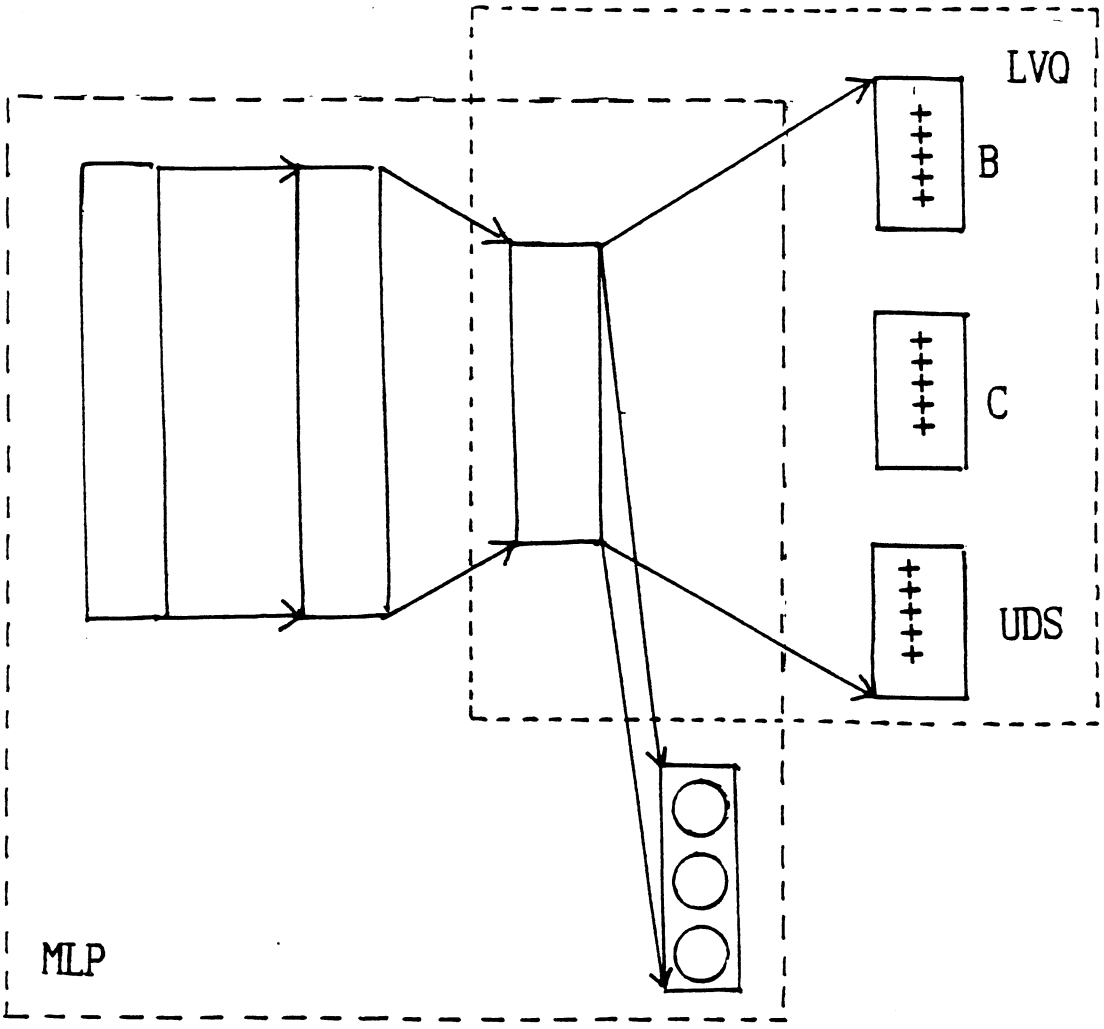
TE3:uds quark events TE6:not uds quark (b,c) events

Figure 4:MLP-RBF purity/efficiency for b quark events

Figure 5:MLP-RBF purity/efficiency for c quark events

Figure 6:MLP-RBF purity/efficiency for u,d,s quark events

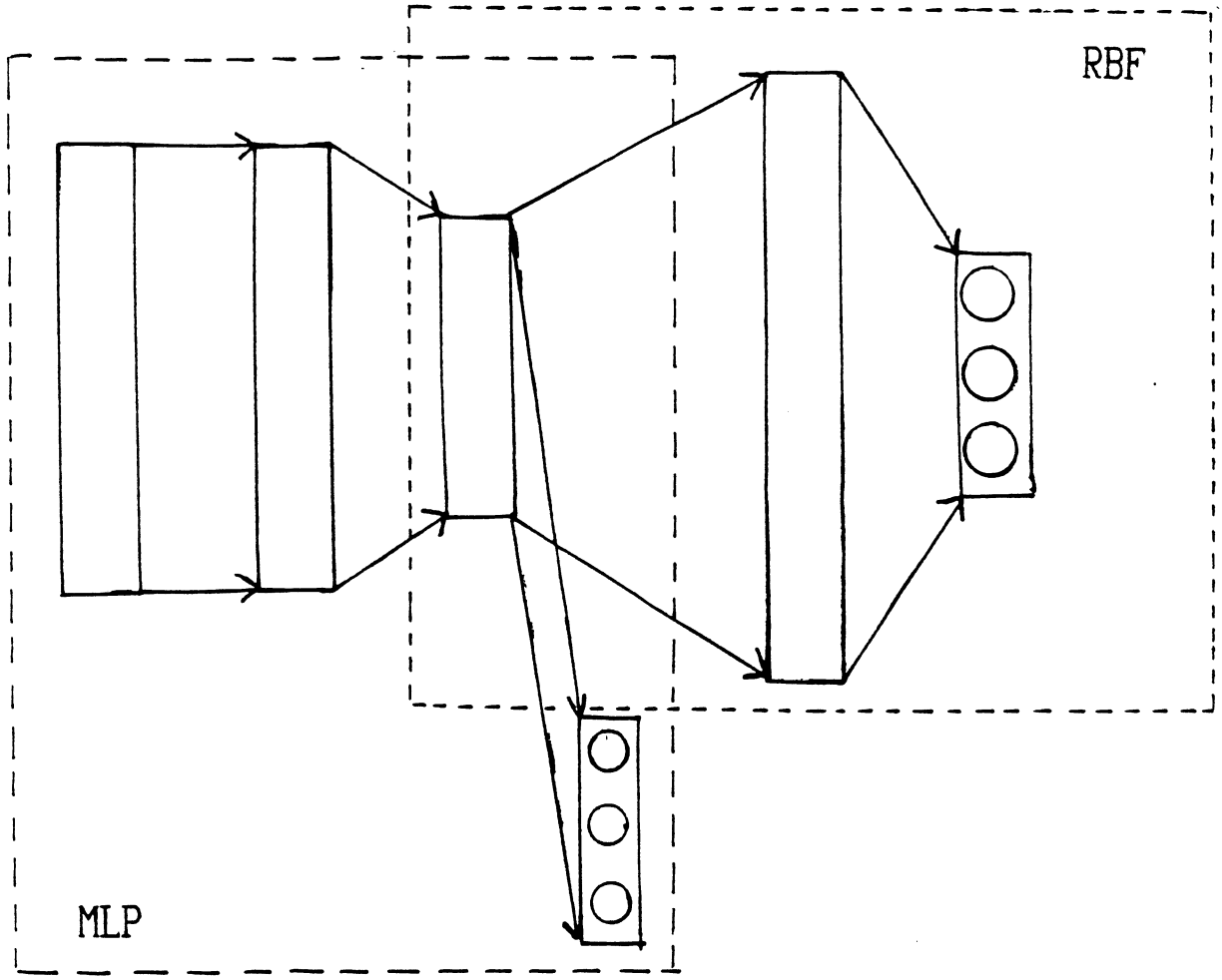
Figure 7:Architecture for the training and the test for the 3-MLP neural network



MLP(1) MLP(2) MLP(3) MLP(OUTPUT)

LVQ(1) LVQ(OUTPUT)

FIGURE 1



MLP(1) MLP(2) MLP(3) MLP(OUTPUT)

RBF(1) RBF(2) RBF(OUTPUT)

FIGURE 2

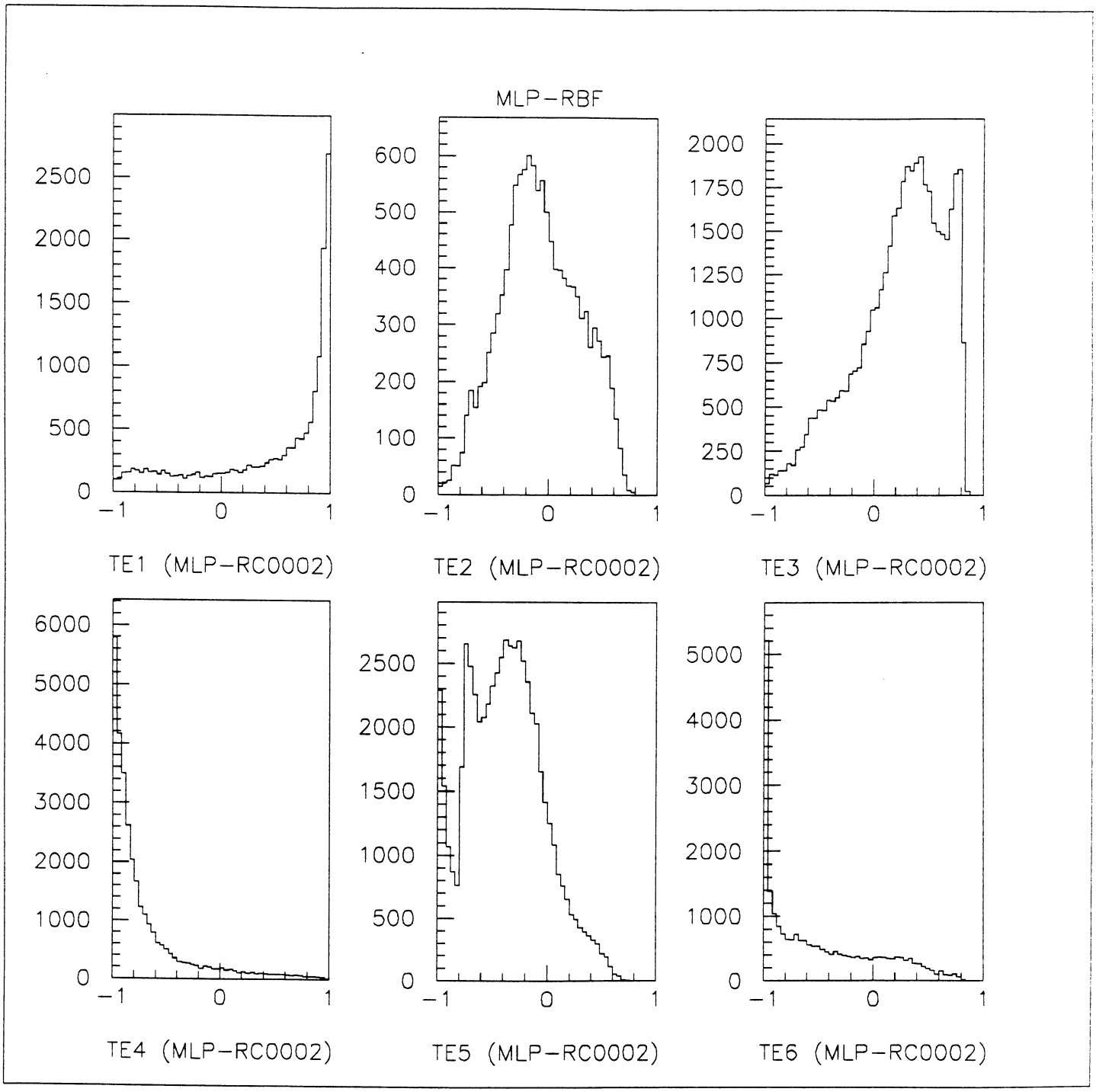

MLP-RBF
TE1 (MLP-RC0002)
TE2 (MLP-RC0002)
TE3 (MLP-RC0002)
TE4 (MLP-RC0002)
TE5 (MLP-RC0002)
TE6 (MLP-RC0002)

FIGURE 3

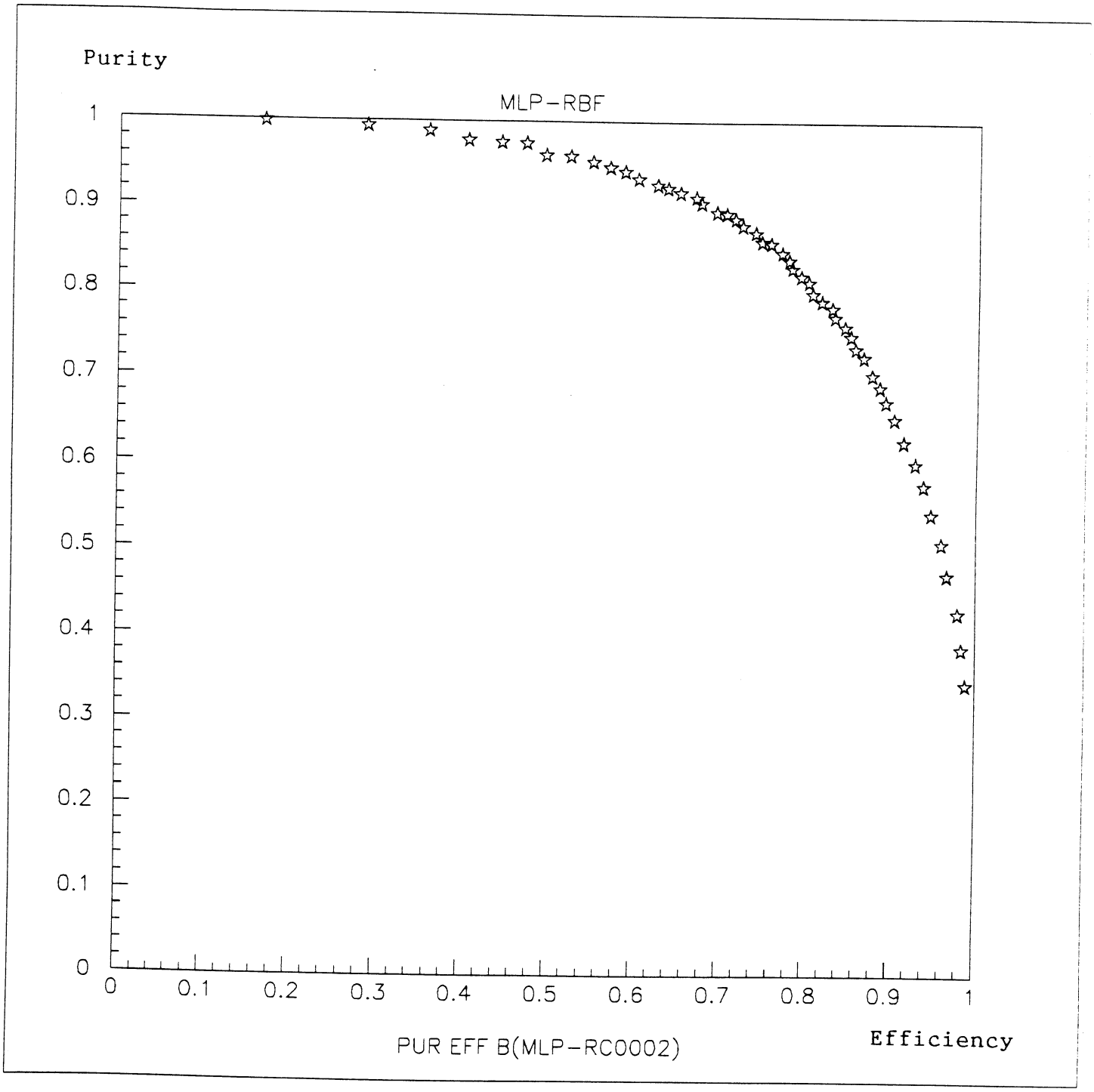

Purity
MLP-RBF
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
PUR EFF B(MLP-RC0002)
Efficiency

FIGURE 4

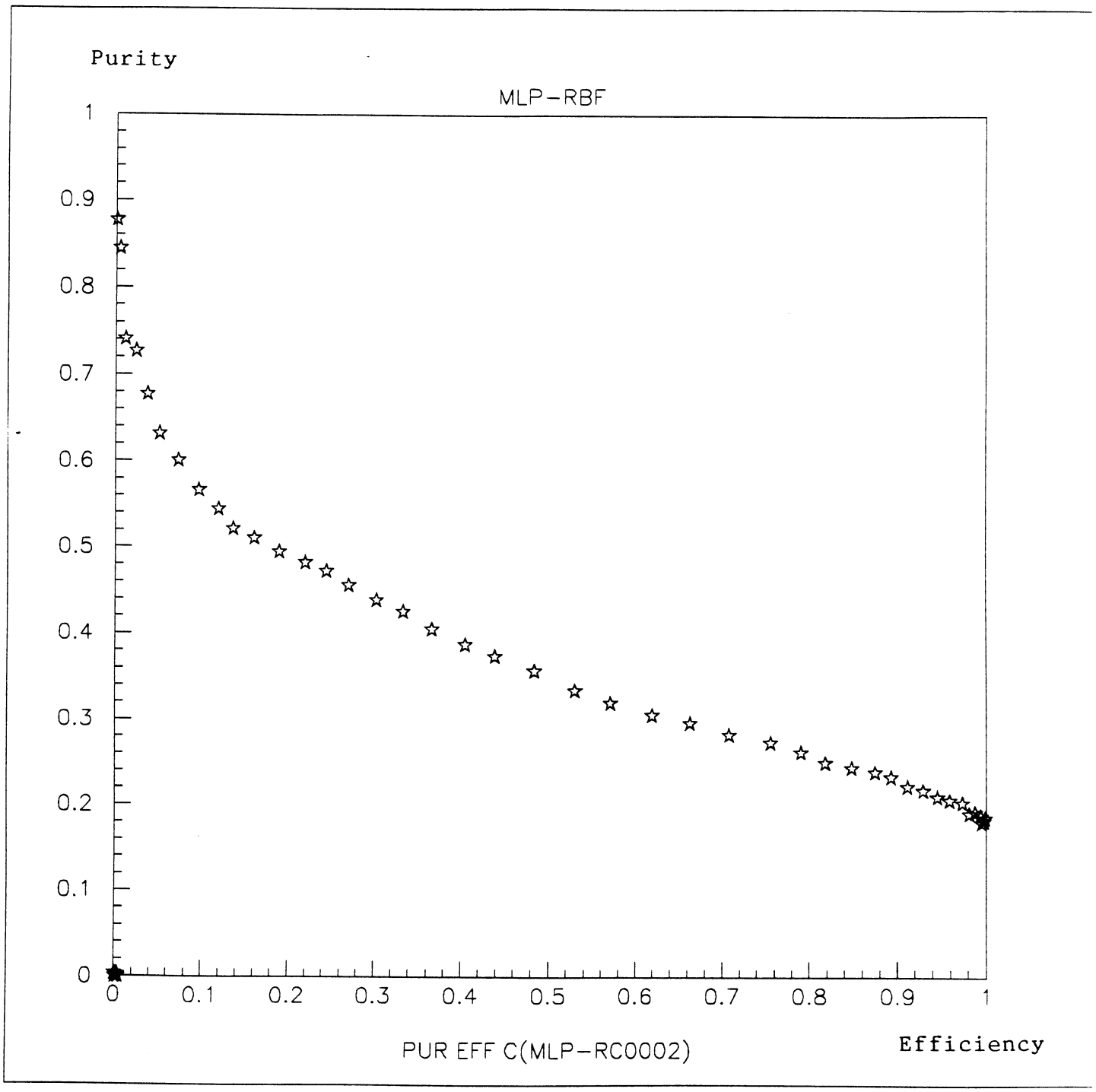

Purity
MLP-RBF
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
PUR EFF C(MLP-RC0002)
Efficiency

FIGURE 5

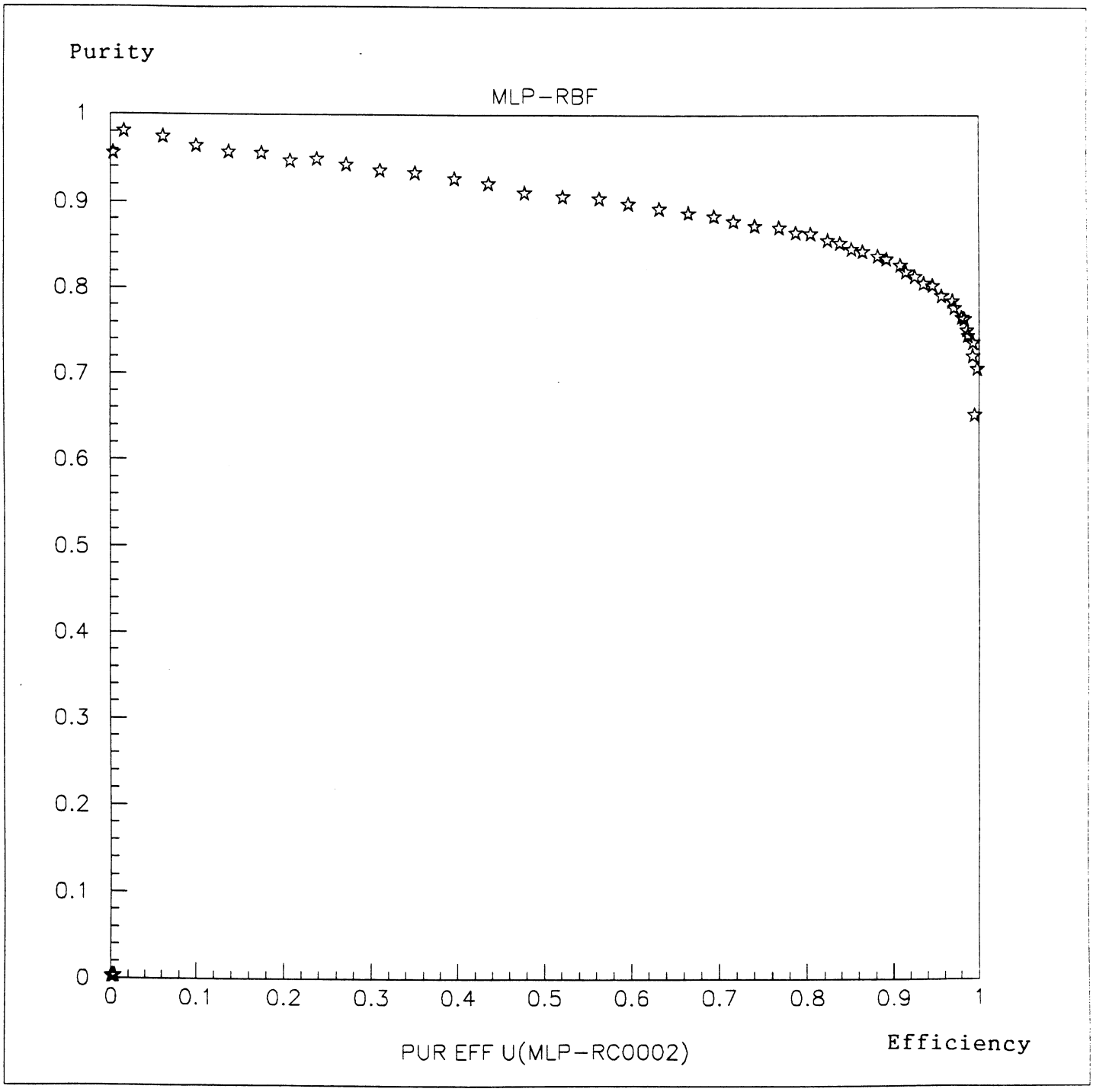
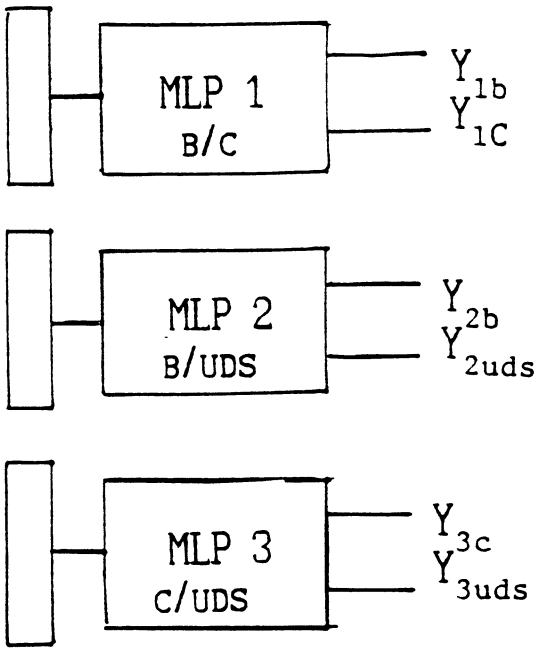

Purity
MLP−RBF
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
PUR EFF U(MLP−RC0002)
Efficiency

FIGURE 6

LEARNING



TEST

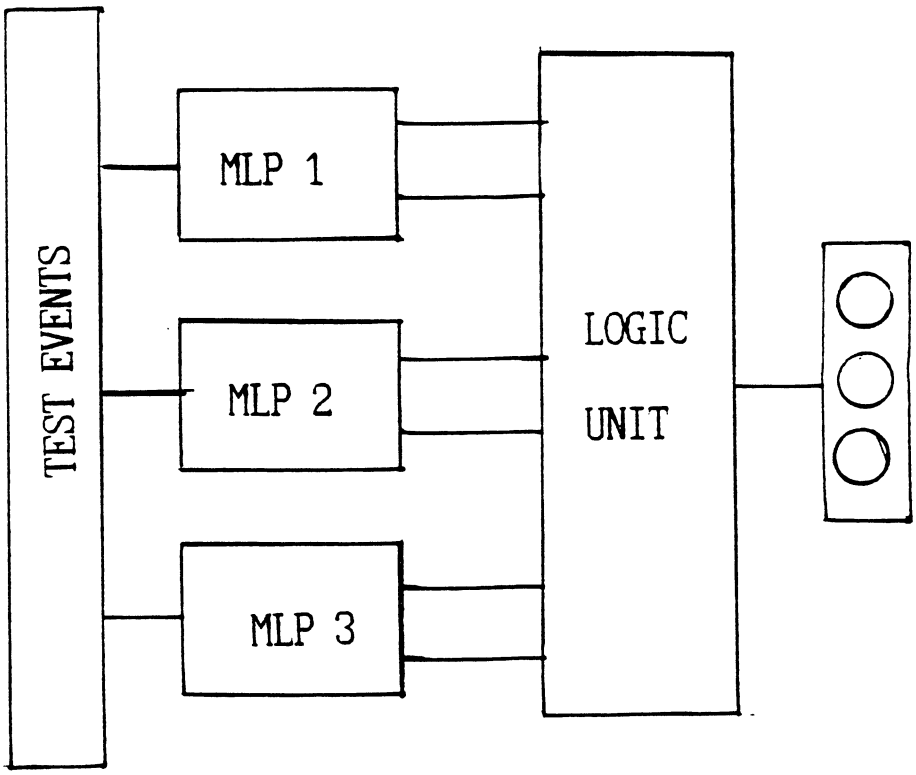


FIGURE 7