Improving the Expert Networks of a Modular Multi-Net System for Pattern Recognition

Mercedes Fernández-Redondo[1],
Joaquín Torres-Sospedra[1], and Carlos Hernández-Espinosa[1]

Departamento de Ingenieria y Ciencia de los Computadores, Universitat Jaume I,
Avda. Sos Baynat s/n, C.P. 12071, Castellon, Spain
{redondo, jtorres, espinosa}@icc.uji.es

Abstract. A *Modular* Multi-Net System consists on some networks which solve partially a problem. The original problem has been decomposed into subproblems and each network focuses on solving a subproblem. The *Mixture of Neural Networks* consist on some expert networks which solve the subproblems and a gating network which weights the outputs of the expert networks. The expert networks and the gating network are trained all together in order to reduce the correlation among the networks and minimize the error of the system. In this paper we present the *Mixture of Multilayer Feedforward* (*MixMF*) a method based on *MixNN* which uses *Multilayer Feedfoward* networks for the expert level. Finally, we have performed a comparison among *Simple Ensemble*, *MixNN* and *MixMF* and the results show that *MixMF* is the best performing method.

1 Introduction

The most important property of an artificial neural network is its generalization capability, which is the ability to correctly respond to inputs which were not used to adapt its weights. The use of a Multi-Net system increases this ability. We can see in [1,2] that there are some approaches to build a multi-net system for pattern recognition.

The most studied approach is the *Ensemble of Neural Networks* or *Comittee Machine* (Figure 1). It consists on training different networks and combine their output vectors in a suitable manner to give the global output or final hypothesis of the classification system. In [3,4] we can see that the use of ensembles increases the generalization capability with respect to a single neural network.

Although most of the methods to create a Multi-Net System are based on the *ensemble approach* [5,6], in this paper we focus on the *Modular Network*. The *Modular Network* is based on the idea of "*divide and conquer*". The network divides the problem into subproblems and each subproblem tends to be solved by one network. We will deal with the *Mixture of Neural Networks* (*MixNN*) because is one of the most known *modular* methods and we think it could be improved. It consists of some expert networks which solve the subproblems and a gating network which is used to combine the outputs of the expert networks. Its basic diagram is also in Figure 1.

The original *Mixture of Neural Networks* (*MixNN*) [7,8] is based on a quite simple neural network architecture. We think that *MixNN* could perform better if the method was based on *Multilayer Feedforward* networks. In this paper we present a *Mixture*

S. Kollias et al. (Eds.): ICANN 2006, Part I, LNCS 4131, pp. 293–302, 2006.

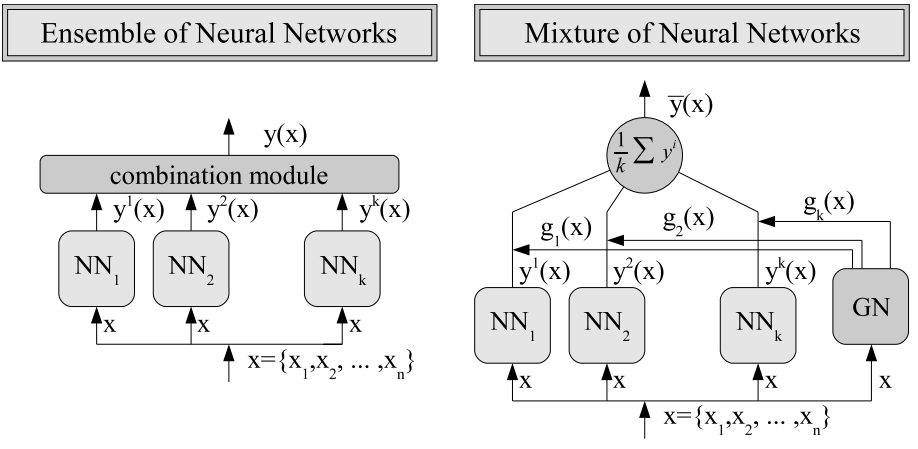

Fig. 1. Simple Ensemble and Mixture of Neural Networks diagrams

of Multilayer Feedforward Networks (*MixMF*) which is a modular approach based on Multilayer Feedforward networks trained with Backpropagation.

In section 2 we describe the *MixNN* and the *MixMF*. In subsection 3.1 we describe the databases used in our experiments. The results we have obtained are in subsection 3.2. We have also calculated the mean *Percentage of Error Reduction* to compare the methods, these results appear in subsetion 3.3.

2 Theory

In this section, we describe the *Simple Ensemble*, the *Mixture of Neural Networks* and the *Mixture of Multilayer Feedforward* Networks.

2.1 Simple Ensemble

Simple Ensemble is a method to build an ensemble of neural networks which consist on training some networks with different weights initialization. The mean of the outputs are applied to get the output of the ensemble.

$$\overline{y}_{class}(x) = \frac{1}{k} \cdot \sum_{net=1}^{k} y_{class}^{net}(x) \tag{1}$$

The output yielding the maximum of the averaged values is chosen as the correct class.

$$h_{simpleensemble}(x) = \arg \max_{class=1,\ldots,q} \overline{y}_{class}(x) \tag{2}$$

Where q is the number of classes, k is the number of networks in the ensemble.

2.2 Mixture of Neural Networks

The *Mixture of Neural Networks* (*MixNN*) is a method to build a Multi-Net System based on the *Modular approach*. It consists on training *k* expert networks and a gating network. The input *x* is applied to the expert networks and the gating network. The modular network output is:

$$\overline{y}_{class} = \sum_{net=1}^{k} g_{net} \cdot y_{class}^{net} \tag{3}$$

Where the output of the expert networks is described in equation 4 and the output of the gating networks is described in equation 5

$$y_{class}^{net} = x^T \cdot w_{class}^{net} \tag{4}$$

$$g_{net} = \frac{\exp\left(x^T \cdot a_{net}\right)}{\sum_{j=1}^{k} \exp\left(x^T \cdot a_j\right)} \tag{5}$$

The output yielding the maximum of the averaged values is chosen as the correct class.

$$h_{MixNN}(x) = \arg \max_{class=1,\ldots,q} \overline{y}_{class}(x) \tag{6}$$

The neural networks used in *MixNN* are quite simple, in Figure 2 we show the diagram of an expert network.

To adapt the weights of the expert networks and the gating network, we have used the objective function described in equation 7.

$$L = \ln\left(\sum_{net=1}^{k} g_{net} \cdot \exp\left(-\frac{1}{2} \cdot \left\|d - y^{net}\right\|^2\right)\right) \tag{7}$$

The equations used to adapt the weights of the expert networks w and the gating network a are:

$$w_{class}^{net}\left(ite+1\right) = w_{class}^{net}\left(ite\right) + \eta \cdot h_{net} \cdot \left(d - y_{class}^{net}\right) \cdot x \tag{8}$$

$$a^{net}\left(ite+1\right) = a^{net}\left(ite\right) + \eta \cdot h_{net} \cdot \left(h_{net} - g_{net}\right) \cdot x \tag{9}$$

where:

$$h_{net} = \frac{g_{net} \cdot \exp\left(-\frac{1}{2}\left|d - y^{net}\right|^2\right)}{\sum_{j=1}^{k}\left(g_j \cdot \exp\left(-\frac{1}{2}\left|d - y^{j}\right|^2\right)\right)} \tag{10}$$

To train the networks of the classification system we have used the following algorithm:

Algorithm 1. Mixture of Neural Networks

```
Random initialization of networks
for ite = 1 to Iterations do
    for each pattern from training set do
        for net = 1 to k do
            Adapt expert weights w^net
        end for
        Adapt gating weights a
    end for
    Calculate L_ite over Validation set
    Save weights
end for
Select iteration with maximum L (best iteration)
Set best iteration weights to network
Save final configuration
```

2.3 Mixture of Multilayer Feedforward Networks

Mixture of Multilayer Feedforward Networks (*MixMF*) is the new method we propose to build a modular network. *MixMF* is an approach of *MixNN* where the expert networks are *Multilayer Feedforward* networks with one hidden layer and threshold nodes. *Multilayer Feedforward* networks are more accurate than the expert networks used in *MixNN* [9], but the training process is slower. In Figure 2 we show the diagram of a *MixMF* expert network with a single hidden layer.

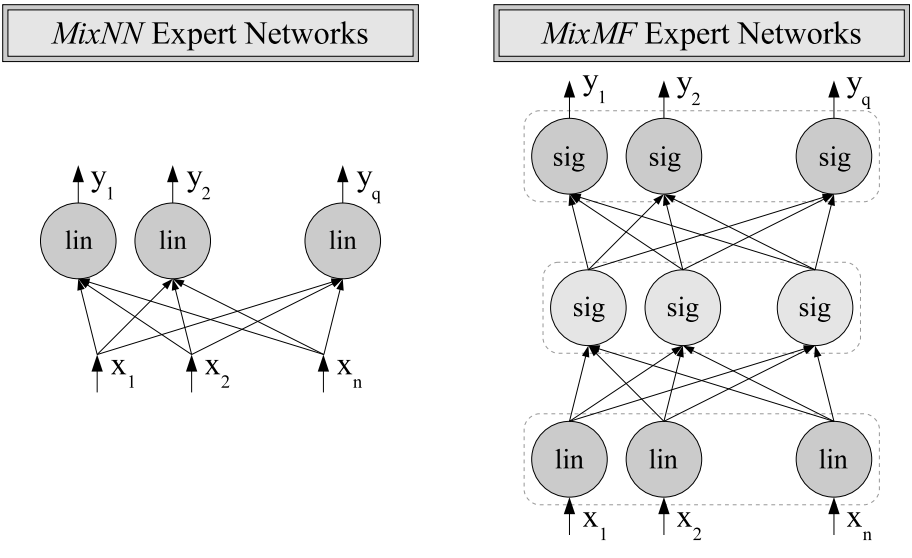

Fig. 2. Expert Network Diagram - MixNN and MixMF

In this case we have used a modified version of Algorithm 1 which take into account that the expert networks are Multilayer Feedforward Networks with one hidden layer and threshold nodes and they are trained with Backpropagation. In this subsection we describe the equations used on the *MixMF* learning process.

In order to adapt the weights of the expert networks and the gating network we have used the objective function described in equation 7. The equations used to adapt the input layer weights of the expert networks wi, the hidden layer weights of the expert networks wh and the gating network a are the following ones:

$$wh_{j,k}^{net}\left(ite+1\right) = wh_{j,k}^{net}\left(ite\right) + \eta \cdot h_{net} \cdot \delta_{k}^{net} \cdot ho_{j}^{net} \tag{11}$$

$$wi_{i,j}^{net}\left(ite+1\right) = wi_{i,j}^{net}\left(ite\right) + \eta \cdot h_{net} \cdot \delta_{j}^{'net} \cdot x_{i} \tag{12}$$

$$a^{net}\left(ite+1\right) = a^{net}\left(ite\right) + \eta \cdot h_{net} \cdot \left(h_{net} - g_{net}\right) \cdot x \tag{13}$$

where:

$$h_{net} = \frac{g_{net} \cdot \exp\left(-\frac{1}{2}\left|d - y^{net}\right|^{2}\right)}{\sum_{j=1}^{k}\left(g_{j} \cdot \exp\left(-\frac{1}{2}\left|d - y^{j}\right|^{2}\right)\right)} \tag{14}$$

$$\delta_{k}^{net} = \left(d_{k} - y_{k}^{net}\right) \cdot \left(1 - y_{k}^{net}\right) \cdot \left(y_{k}^{net}\right) \tag{15}$$

$$\delta_{j}^{'net} = ho_{j}^{net} \cdot \left(1 - ho_{j}^{net}\right) \cdot \sum_{h=1}^{m} \delta_{h}^{net} \cdot wh_{j,h}^{net} \tag{16}$$

3 Experimental Testing

In this section, we describe the experimental setup, the datasets we have used in our experiments and we show the results we have obtained. Finally, we compare the results we have obtained with *Simple Ensemble*, *MixNN* and *MixMF* on the different datasets.

For this reason we have trained multiple classification systems of 3, 9, 20 and 40 MF networks with *Simple Ensemble*, *MixNN* and *MixMF* on the eight problems described in subsection 3.1. For the expert networks of *Simple Ensemble* and *MixMF* we have used the training parameters described in table 1. In addition, we repeated ten times the whole learning process, using different partitions of data in training, validation and test sets. With this procedure we can obtain a mean performance of the ensemble for each database and an error in the performance calculated by standard error theory.

3.1 Datasets

We have used eigth different classification problems from the *UCI repository of machine learning databases* [10] to test the performance of methods. The databases we have used are: Cylinder Bands Database (band), Australian Credit Approval (cred), Solar Flare Database (flare), Glass Identification Database (glas), The monk's problem 1 (mok1), Congressional Voting Records Database (vote), Wisconsin Breast Cancer Database (wdbc).

Table 1. MF training parameters and Performance of a Single Network

Database	Hidden	Iterations	Step	Momentum	Performance
band	23	5000	0.1	0.05	72.4 ± 1.0
cred	15	8500	0.1	0.05	85.6 ± 0.5
flare	11	10000	0.6	0.05	82.1 ± 0.3
glas	3	4000	0.1	0.05	78.5 ± 0.9
mok1	6	3000	0.1	0.05	74.3 ± 1.1
survi	9	20000	0.1	0.2	74.2 ± 0.8
vote	1	2500	0.1	0.05	95.0 ± 0.4
wdbc	6	4000	0.1	0.05	97.4 ± 0.3

3.2 Results

In this subsection we present the experimental results we have obtained with the Multi-Net Systems trained with *Simple Ensemble* (Table 2), *Mixture of Neural Networks* (Table 3) and *Mixture of Multilayer Feedforward Networks* (Table 4).

Table 2. Simple Ensemble results

Database	3 Nets	9 Nets	20 Nets	40 Nets
band	73.5 ± 1.2	72.9 ± 1.5	73.8 ± 1.3	73.8 ± 1.3
cred	86.5 ± 0.7	86.4 ± 0.7	86.6 ± 0.7	86.5 ± 0.7
flare	81.8 ± 0.5	81.6 ± 0.4	81.5 ± 0.5	81.6 ± 0.5
glas	94 ± 0.8	94 ± 0.7	94 ± 0.7	94.2 ± 0.6
mok1	98.3 ± 0.9	98.8 ± 0.8	98.3 ± 0.9	98.3 ± 0.9
survi	74.3 ± 1.3	74.2 ± 1.3	74.3 ± 1.3	74.3 ± 1.3
vote	95.6 ± 0.5	95.6 ± 0.5	95.6 ± 0.5	95.6 ± 0.5
wdbc	96.9 ± 0.5	96.9 ± 0.5	96.9 ± 0.5	96.9 ± 0.5

Table 3. *Mixture of Neural Networks* results

Database	3 Nets	9 Nets	20 Nets	40 Nets
band	72.7 ± 2.2	74.4 ± 1.3	74 ± 1.9	75.5 ± 1.3
cred	86.8 ± 0.5	86.9 ± 0.5	86.5 ± 0.6	86 ± 0.5
flare	81.5 ± 0.5	81.7 ± 0.5	81.7 ± 0.6	81.8 ± 0.6
glas	89.4 ± 1	91.2 ± 1.1	90.2 ± 1.3	91 ± 1.1
mok1	87.8 ± 2.2	93.6 ± 2.6	93.6 ± 2.1	93.9 ± 2.5
survi	72.3 ± 1.2	72.6 ± 0.9	73.8 ± 0.9	73.6 ± 1.2
vote	95 ± 1.2	96.1 ± 0.6	96.1 ± 0.6	96.5 ± 0.7
wdbc	94.7 ± 0.5	94.9 ± 0.4	95.1 ± 0.6	94.6 ± 0.5

3.3 Interpretations of Results

Comparing the results showed in tables 2, 3 and 4 we can see that we have got better results with *MixMF*. We can also see that the improvement in performance using

Table 4. *Mixture of Multilayer Feedforward* results

Database	3 Nets	9 Nets	20 Nets	40 Nets
band	75.5 ± 1.9	74.2 ± 2	74.7 ± 1.7	73.8 ± 1.6
cred	85.9 ± 0.5	86.7 ± 0.7	86.5 ± 0.7	86.8 ± 0.5
flare	82.1 ± 0.6	81.9 ± 0.6	81.6 ± 0.6	81.7 ± 0.6
glas	94.6 ± 1	94.6 ± 1.2	94.2 ± 1.3	95 ± 1.2
mok1	99.3 ± 0.8	99.3 ± 0.8	98.8 ± 0.9	100 ± 0
survi	74.6 ± 1.3	74.9 ± 1.2	74.6 ± 1.1	75.1 ± 1.2
vote	96.1 ± 0.6	96.1 ± 0.6	96.1 ± 0.6	95.8 ± 0.6
wdbc	96.9 ± 0.5	96.9 ± 0.5	96.9 ± 0.5	96.9 ± 0.5

MixMF depends on the database and the number of networks used in the multinet system. For instante, in database *mok1 MixMF* has an increase of performance with respect to *MixNN* of 11.50% for the 3-network system, but only has an increase of 5.12% for the 20-network system.

In general, we can also see in these tables that the *MixMF* increase of performance with respect to *MixNN* is higher than with respect to *Simple Ensemble*, for instance in databases *mok1*, *glas* and *survi*.

The increase of performance we have shown is an *absolute* measure so we cannot see how important is the increase of performance with respect to the error. To have a *relative* measure and information about the error reduction, we have also calculated the percentage of error reduction PER of the multinet systems with respect to a *Single Network*. We have used equation (17) to calculate the PER value.

$$PER = 100 \cdot \frac{Error_{singlenetwork} - Error_{multinet}}{Error_{singlenetwork}} \tag{17}$$

The PER value ranges from 0%, where there is no improvement by the use of a particular multinet system method with respect to a single network, to 100%. There can also be negative values, which means that the performance of the multinet system is worse than the performance of the single network. This new measurement is relative and can be used to compare more clearly the different methods.

Table 5 shows PER values for the ensembles trained with *Simple Ensemble*. Tables 6 and 7 shows PER values for the modular networks trained with *MixNN* and *MixMF*.

According to the results showed in tables 5-7, we can see that, in general, *Mixture of Multilayer Feedforward Networks* is the best performing method, and *Mixture of Neural Netorks* is the worst method for 5 databases.

Furthermore, we have calculated the mean increase of performance with respect to a *Single Network* (Table 8) and the mean PER (Table 9) across all databases for each method to get a global measurement.

According to the mean PER, *MixMF* is the best performing method. The highest difference between *MixMF* and *Simple Ensemble* is in the 9-network ensemble where the mean PER increase is 3.3%. The highest difference between original *MixMF* and *MixNN* is in the 3-network ensemble where the mean PER increase is 23.90%. In Figure we can see more the diference on PER among all the methods.

Table 5. Simple Ensemble PER

Database	3 Nets	9 Nets	20 Nets	40 Nets
band	3.8	1.84	5.14	5.14
cred	6.52	5.41	7.08	6.52
flare	−1.68	−2.8	−3.36	−2.8
glas	72.09	72.09	72.09	73.02
mok1	93.19	95.13	93.19	93.19
survi	0.38	0	0.38	0.38
vote	12.59	12.59	12.59	12.59
wdbc	−19.24	−19.24	−19.24	−19.24

Table 6. *Mixture of Neural Networks* PER

Database	3 Nets	9 Nets	20 Nets	40 Nets
band	1.19	7.1	5.79	11.05
cred	8.12	8.68	5.97	2.77
flare	−3.19	−2.13	−2.13	−1.63
glas	50.69	59.06	54.41	58.13
mok1	52.33	75.21	75.21	76.18
survi	−7.37	−6.13	−1.67	−2.29
vote	0	22.59	22.59	30
wdbc	−102.7	−95.77	−88.85	−106.16

Table 7. *Mixture of Multilayer Feedforward* PER

Database	3 Nets	9 Nets	20 Nets	40 Nets
band	11.05	6.44	8.44	5.14
cred	2.22	7.56	5.97	8.12
flare	−0.28	−1.07	−2.91	−2.13
glas	74.88	74.88	73.02	76.74
mok1	97.08	97.08	95.13	100
survi	1.51	2.79	1.51	3.41
vote	22.59	22.59	22.59	15
wdbc	−18.85	−18.85	−18.85	−18.85

Table 8. Mean Increase of Performance with respect to *Single Network* across all databases

Method	3 Nets	9 Nets	20 Nets	40 Nets
S.E	5.17	5.1	5.19	5.21
MixNN	3.67	3.99	3.93	4.17
MixMF	5.62	5.63	5.48	5.69

Table 9. Mean Percentage of Error Reduction with respect to *Single Network* across all databases

Method	3 Nets	9 Nets	20 Nets	40 Nets
S.E.	20.96	20.63	20.98	21.1
MixNN	−0.12	8.58	8.92	8.51
MixMF	23.77	23.93	23.11	23.43

4 Conclusions

In this paper we have reviewed the *Mixture of Neural Networks* a modular network based on a quite simple architecture of neural networks.

Finally, we have proposed *Mixture of Multilayer Feedforward Networks*, a modular method based on *Mixture of Neural Networks* and *Multilayed Feedforward*.

Moreover, we have trained Multi-Net Systems of 3, 9, 20 and 40 networks with *Simple Ensemble*, *Mixture of Neural Networks* and *Mixture of Multilayer Feedforward* in order to test the performance of the new method and cover a wide spectrum of the number of networks in the classification system. The results showed that the performance of *Mixture of Multilayer Feedforward* was better but the improvement by the use of *Mixture of Multilayer Feedforward* depends on the database.

Futhermore, we have obtained the mean *Percentage of Error Reduction* across all databases and the mean *Increase of Performance*. According to these global measures, the method we have proposed *Mixture of Multilayer Feedforward* performs better than *Simple Ensemble* and it permorms by far better than the original *Mixture of Neural Network*. In general, the *Mixture of Multilayer Feedforward* is the best performing method.

We can conclude that the *Mixture of Neural Networks* variation we have proposed in this paper is better than the original *Mixture of Neural Networks* because it uses a better neural networks architecture to build the expert networks. Moreover, the *Mixture of Multilayer Feedforward* performs better than *Simple Ensemble* because training process and the gating network reduces the correlation among the networks.

Acknowledgments

This research was supported by project *P1·1B2004-03* of Universitat Jaume I - Bancaja in Castellón de la Plana, Spain.

References

1. Sharkey, A.J., ed.: Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems. (1999)
2. Dara, R.A., Kamel, M.S.: Sharing training patterns among multiple classifiers. In Roli, F., Kittler, J., Windeatt, T., eds.: Multiple Classifier Systems. Volume 3077 of Lecture Notes in Computer Science., Springer (2004) 243–252
3. Tumer, K., Ghosh, J.: Error correlation and error reduction in ensemble classifiers. Connection Science **8**(3-4) (1996) 385–403

4. Raviv, Y., Intratorr, N.: Bootstrapping with noise: An effective regularization technique. Connection Science, Special issue on Combining Estimators **8** (1996) 356–372
5. Hernandez-Espinosa, C., Fernandez-Redondo, M., Torres-Sospedra, J.: Ensembles of multilayer feedforward for classification problems. In: Neural Information Processing, ICONIP 2004. Volume 3316 of Lecture Notes in Computer Science. (2005) 744–749
6. Hernandez-Espinosa, C., Torres-Sospedra, J., Fernandez-Redondo, M.: New experiments on ensembles of multilayer feedforward for classification problems. In: Proceedings of International Conference on Neural Networks, IJCNN 2005, Montreal, Canada. (2005) 1120–1124
7. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Computation **3** (1991) 79–87
8. Jordan, M.I., Jacobs, R.A.: Hierarchical mixtures of experts and the EM algorithm. Technical Report AIM-1440 (1993)
9. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience (2004)
10. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998)