Researching on Multi-net Systems Based on Stacked Generalization

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

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

Abstract. Among the approaches to build a Multi-Net system, *Stacked Generalization* is a well-known model. The classification system is divided into two steps. Firstly, the level-0 generalizers are built using the original input data and the class label. Secondly, the level-1 generalizers networks are built using the outputs of the level-0 generalizers and the class label. Then, the model is ready for pattern recognition. We have found two important adaptations of *Stacked Generalization* that can be applyied to artificial neural networks. Moreover, two combination methods, *Stacked* and *Stacked+*, based on the *Stacked Generalization* idea were successfully introduced by our research group. In this paper, we want to empirically compare the version of the original *Stacked Generalization* along with other traditional methodologies to build Multi-Net systems. Moreover, we have also compared the combiners we proposed. The best results are provided by the combiners *Stacked* and *Stacked+* when they are applied to ensembles previously trained with *Simple Ensemble*.

1 Introduction

Perhaps, the most important property of a neural network is the generalization capability. The ability to correctly respond to inputs which were not used in the training set.

It is clear from the bibliography that the use of an ensemble of neural networks increases the generalization capability, [13] and [7], for the case of *Multilayer Feedforward* and other classifiers. The two key factors to design an ensemble are how to train the individual networks and how to combine them.

Although most of the methods to create a Multi-Net System are based on the *Ensemble approach* (*Boosting*, *Bagging*, *Cross-Validation*) [2,4] or in the *Modular approach* (*Mixture of Neural Networks*) [8,12], we have also analyzed other complex methodologies and models to perform an exhaustive comparison. *Stacked Generalization* is one of the most known alternatives to the traditional methods previously mentioned.

Stacked Generalization was introduced by Wolpert in 1994 [14]. Firstly, a set of cross-validated generalizers called *level-0 generalizers* are trained with the original input data and class label. Then, a set of generalizers called *level-1 generalizers* are trained using the information provided by *level-0* generalizers along with the class label. Unfortunately, *Stacked Generalization* can not be directly applied to artificial

L. Prevost, S. Marinai, and F. Schwenker (Eds.): ANNPR 2008, LNAI 5064, pp. 193–204, 2008.

neural networks because it can not be applied to methods that require being trained before classification.

Some authors like Ghorbani & Owrangh [3] and Ting & Witten [9,10] have proposed some versions based on the Wolpert's model which can be directly applied to neural networks. Moreover, we proposed in [11] two combination methods based on *Stacked Generalization* model, *Stacked* and *Stacked+*. Unfortunately there is not any comparison among them.

The original *Stacked Generalization* can be incorrectly identified as a method to combine an ensemble. In the original model specifications, the process to create the combiners highly depends on the process to create the experts so the *stacked models* can not be considered as a traditional ensemble. Moreover, the *stacked combiners* we proposed should not be considered as pure 2-leveled models.

The *Stacked Generalization* model applied to neural networks is shown in figure 1. We can see that the *level-0 generalizers* are called *Expert Networks* (*EN*) whereas the *level-1 generalizers* are called *Combination Networks* (*CN*).

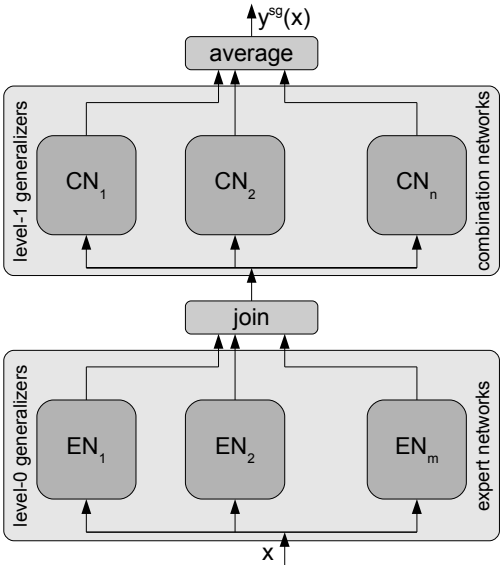

Fig. 1. Stacked Generalization Model

In this paper, we want to compare Ghorbani & Owrangh and Ting & Witten models with other ensemble models and with combiners *Stacked Stacked+*. To perform this comparison, ten databases from the UCI repository have been chosen.

This paper is organized as follows. Firstly, some theoretical concepts are briefly reviewed in section 2. Then, the databases used and the experimental setup are described in section 3. Finally, the experimental results and their discussion are in section 4.

2 Theory

2.1 The Multilayer Feedforward Network

The *Multilayer Feedforward* architecture is the most known network architecture. This kind of networks consists of three layers of computational units. The neurons of the first layer apply the identity function whereas the neurons of the second and third layer apply the sigmoid function. It has been proved that *MF* networks with one hidden layer and threshold nodes can approximate any function with a specified precision [1] and [5]. Figure 2 shows the diagram of the *Multilayer Feedforward* network.

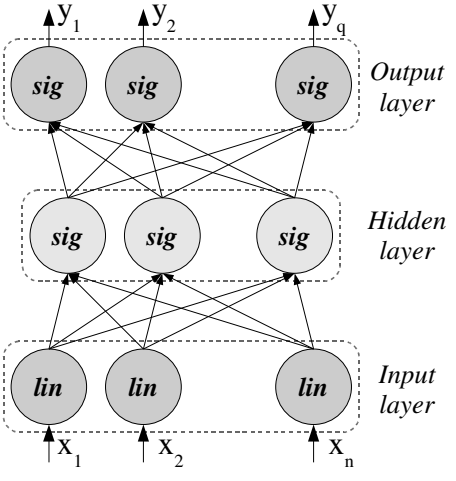

Fig. 2. Multilayer Feedforward Structure

In our experiments we have trained the networks with the following algorithm.

Algorithm 1. Neural Network Training $\{T, V\}$

```
Set initial weights values
for i = 1 to iterations do
    Train the network with the patterns from the training set T
    Calculate MSE over validation set V
    Save epoch weights and calculated MSE
end for
Select epoch with minimum MSE
Assign best epoch configuration to the network
Save network configuration
```

2.2 Traditional Multi-net Models

Simple Ensemble. A simple ensemble can be constructed by training different networks with the same training set, but with different random initialization [2].

Bagging. This method consists on generating different datasets drawn at random with replacement from the original training set [2].

Adaptive Boosting. In *Adaptive Boosting*, also known as *Adaboost*, the successive networks are trained with a training set selected randomly from the original set, but the probability of selecting a pattern changes depending on the correct classification of the pattern and on the performance of the last trained network [2].

Cross-Validation. We have used two different versions of k-fold cross-validation: *CVC* [2] and *CVCv2* [4]. In CVC, the training set is divided into k subsets being $k-1$ subsets used to train the network. In this case, the subset which is left out is not used in the training and validation process. In *CVCv2*, cross-validation is applied to the learning set to get k different subsets. In this case, $k-1$ subsets are used to train the network and the left one is used for validation.

In both cases, we can construct k classifiers with different sets by changing the subset that is left out.

2.3 Stacked Generalization

Stacked Generalization was introduced by Wolpert [14]. Some authors have adapted the Wolpert's method to use with neural networks like Ghorbani & Owrangh [3] and Ting & Witten [9,10].

The training in *Stacked Generalization* is divided into two steps. In the first one, the expert networks are trained. In the second one, the combination networks are trained with the outputs provided by the experts. However, there are some constraints related to the datasets used to train the expert and combination networks. The method is fully described in [14]. In fact, it has been suggested that *Stacked Generalization* is a sophisticated version of *Cross-Validation*.

Unfortunately, the method proposed by Wolpert can not be directly applied to generalizers that requires being trained before classificating patterns. Although this drawback, there are some authors that have adapted *Stacked Generalization* to specific classifiers. There are two versions that have to be taken into account: The version proposed by Ghorbani & Owrangh [3] and the version proposed by Ting & Witten [9,10]. Those authors described their procedure to create the different training sets for the experts and combination networks.

Stacked Generalization - Version 1. Ting & Witten proposed a version of *Stacked Generalization* that can be applied to the *Multilayer Feedforward* architecture [9,10]. The training set was randomly splitted into k equal subsets: $T = \{T_1, T_2, ..., T_k\}$. Then, $T^{-j} = \{T - T_j\}$ was used to train the experts networks and the experts output on T_j were used to train the combination networks.

Stacked Generalization - Version 2. Ghorbani & Owrangh proposed a version of *Stacked Generalization* that was applied directly to Artificial Neural Networks [3]. They applied cross-validation to create the different training sets of the experts by randomly splitting the training set into k equal subsets: $T = \{T_1, T_2, ..., T_k\}$. With this procedure,

k different classifiers can be built with different training sets by changing the subset that is left out as in *CVC*. Then, the outputs of the experts on T were used to train the combination networks.

Combining networks with Stacked and Stacked+. Moreover, *Stacked Generalization* can be used as a combiner of ensembles of neural networks as we proposed in [11] in which we introduced *Stacked* and *Stacked+*.

In those combiners, the networks of an ensemble previously trained with *Simple Ensemble* were used as expert networks. Then, the outputs provided by the ensemble on the original training set were used to train the combination networks. Additionally, in the combiner *stacked+*, the original input data was also used to train the combination network.

3 Experimental Setup

To test the performance of the methods described in this paper, we have used ensembles of 3, 9, 20 and 40 expert networks. In the case of two versions of *Stacked Generalization* we have used classification systems with 3, 9, 20 and 40 expert networks that have been combined by a single combination network. Finally, in the case of *Stacked* and *Stacked+*, we have used ensembles of 3, 9, 20 and 40 networks previously trained with *Simple Ensemble*, these networks have been combined by a single combination network.

In addition, the whole learning process have been repeated ten times, using different partitions of data in training, validation and test sets. With this procedure we can obtain a mean performance of the ensemble and the error calculated by standard error theory.

3.1 Databases

We have used ten different classification problems from the *UCI repository of machine learning databases* [6] to test the performance of the methods and combiners reviewed in this paper. These databases are:

- *Arrhythmia Database* (aritm)
- *Dermatology Database* (derma)
- *Ecoli Database* (ecoli)
- *Solar Flares Database* (flare)
- *Image segmentation Database* (img)
- *Ionosphere Database* (ionos)
- *Pima Indians Diabetes Database* (pima)
- *Haberman's Survival Data* (survi)
- *Vowel Database* (vowel)
- *Wisconsin Breast Cancer Database* (wdbc)

All the training parameters (Hidden units, Adaptation step, Momentum rate and Number of iterations of *Back-propagation*) have been set after an exhaustive trial and error procedure. Due to the lack of space, the parameters are omitted.

4 Results and discussion

4.1 Results

The main results of our work are presented in this subsection. Tables 1-5 show the performance of the ensembles of 3, 9, 20 and 40 networks trained with the traditional ensemble methods described in subsection 2.2.

Table 1. Simple Ensemble results

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	73.4 ± 1	73.8 ± 1.1	73.8 ± 1.1	73.8 ± 1.1
derma	97.2 ± 0.7	97.5 ± 0.7	97.3 ± 0.7	97.6 ± 0.7
ecoli	86.6 ± 0.8	86.9 ± 0.8	86.9 ± 0.8	86.9 ± 0.7
flare	81.8 ± 0.5	81.6 ± 0.4	81.5 ± 0.5	81.6 ± 0.5
img	96.5 ± 0.2	96.7 ± 0.3	96.7 ± 0.2	96.8 ± 0.2
ionos	91.1 ± 1.1	90.3 ± 1.1	90.4 ± 1	90.3 ± 1
pima	75.9 ± 1.2	75.9 ± 1.2	75.9 ± 1.2	75.9 ± 1.2
survi	74.3 ± 1.3	74.2 ± 1.3	74.3 ± 1.3	74.3 ± 1.3
vowel	88 ± 0.9	91 ± 0.5	91.4 ± 0.8	92.2 ± 0.7
wdbc	96.9 ± 0.5	96.9 ± 0.5	96.9 ± 0.5	96.9 ± 0.5

Table 2. Bagging results

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	74.7 ± 1.6	75.9 ± 1.7	75.9 ± 1.7	74.7 ± 1.5
derma	97.5 ± 0.6	97.7 ± 0.6	97.6 ± 0.6	97.6 ± 0.6
ecoli	86.3 ± 1.1	87.2 ± 1	87.1 ± 1	86.9 ± 1.1
flare	81.9 ± 0.6	82.4 ± 0.6	82.2 ± 0.5	82 ± 0.6
img	96.6 ± 0.3	96.7 ± 0.3	97 ± 0.3	97.1 ± 0.3
ionos	90.7 ± 0.9	90.1 ± 1.1	89.6 ± 1.1	90 ± 1.1
pima	76.9 ± 0.8	76.6 ± 0.9	77 ± 1	77 ± 1.1
survi	74.2 ± 1.1	74.4 ± 1.5	74.6 ± 1.7	74.2 ± 1.3
vowel	87.4 ± 0.7	90.8 ± 0.7	91.3 ± 0.6	91.2 ± 0.8
wdbc	96.9 ± 0.4	97.3 ± 0.4	97.5 ± 0.4	97.4 ± 0.3

Tables 6-7 show the results we have obtained with the combiners *Stacked* and *Stacked+* on ensembles previoulsly trained with *Simple Ensemble*.

Tables 8-9 show the results related to the first version of *Stacked Generalization*. These tables show the performance of the experts combined as an ensemble and the performance of the whole model.

In a similar way, tables 10-11 show the results related to the second version of *Stacked Generalization*.

Table 3. Adaptive Boosting results

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	71.8 ± 1.8	73.2 ± 1.6	71.4 ± 1.5	73.8 ± 1.1
derma	98 ± 0.5	97.3 ± 0.5	97.5 ± 0.6	97.8 ± 0.5
ecoli	85.9 ± 1.2	84.7 ± 1.4	86 ± 1.3	85.7 ± 1.4
flare	81.7 ± 0.6	81.1 ± 0.7	81.1 ± 0.8	81.1 ± 0.7
img	96.8 ± 0.2	97.3 ± 0.3	97.29 ± 0.19	97.3 ± 0.2
ionos	88.3 ± 1.3	89.4 ± 0.8	91.4 ± 0.8	91.6 ± 0.7
pima	75.7 ± 1	75.5 ± 0.9	74.8 ± 1	73.3 ± 1
survi	75.4 ± 1.6	74.3 ± 1.4	74.3 ± 1.5	73 ± 2
vowel	88.4 ± 0.9	94.8 ± 0.7	96.1 ± 0.7	97 ± 0.6
wdbc	95.7 ± 0.6	95.7 ± 0.7	96.3 ± 0.5	96.7 ± 0.9

Table 4. CVC results

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	74 ± 1	74.8 ± 1.3	74.8 ± 1.3	73.4 ± 1.9
derma	97.3 ± 0.7	97.6 ± 0.6	97.3 ± 0.6	97.3 ± 0.6
ecoli	86.8 ± 0.8	87.1 ± 1	86.5 ± 1	86.8 ± 0.9
flare	82.7 ± 0.5	81.9 ± 0.6	81.7 ± 0.7	81.7 ± 0.7
img	96.4 ± 0.2	96.6 ± 0.2	96.8 ± 0.2	96.6 ± 0.2
ionos	87.7 ± 1.3	89.6 ± 1.2	89.6 ± 1.3	88.3 ± 1
pima	76 ± 1.1	76.9 ± 1.1	76.2 ± 1.3	76.6 ± 1
survi	74.1 ± 1.4	75.2 ± 1.5	73.8 ± 0.9	74.6 ± 1
vowel	89 ± 1	90.9 ± 0.7	91.9 ± 0.5	92.2 ± 0.8
wdbc	97.4 ± 0.3	96.5 ± 0.5	97.4 ± 0.4	96.8 ± 0.5

Table 5. CVCv2 results

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	76.1 ± 1.6	75.4 ± 1.5	74.3 ± 1.2	77 ± 0.8
derma	98 ± 0.3	97.3 ± 0.5	97.5 ± 0.6	97.2 ± 0.6
ecoli	86.8 ± 0.9	86.6 ± 1	86.6 ± 1.1	86.3 ± 0.9
flare	82.5 ± 0.6	82.1 ± 0.5	81.8 ± 0.4	82.2 ± 0.5
img	96.9 ± 0.3	97 ± 0.3	97.03 ± 0.17	96.7 ± 0.3
ionos	89.7 ± 1.4	90.4 ± 1.3	91 ± 0.9	92 ± 1
pima	76.8 ± 1	76.8 ± 1.1	76.7 ± 0.8	76.1 ± 0.9
survi	74.1 ± 1.2	73 ± 1	73.6 ± 1	73.4 ± 1.2
vowel	89.8 ± 0.9	92.7 ± 0.7	93.3 ± 0.6	92.9 ± 0.7
wdbc	96.7 ± 0.3	96.8 ± 0.3	95.9 ± 0.6	96 ± 0.5

Table 6. Simple Ensemble - Stacked Combiner results

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	75.4 ± 1.4	75.1 ± 1.2	73.8 ± 1.3	73.9 ± 1.4
derma	97.2 ± 0.7	97.5 ± 0.7	97.5 ± 0.7	97.6 ± 0.7
ecoli	86.6 ± 0.9	86.8 ± 1.1	86.8 ± 0.9	86.8 ± 1.1
flare	81.4 ± 0.6	81.1 ± 0.5	81.4 ± 0.6	81.3 ± 0.8
img	96.5 ± 0.2	96.6 ± 0.3	97 ± 0.2	97.2 ± 0.2
ionos	92 ± 0.8	92.9 ± 1	92.7 ± 1.1	92.4 ± 1
pima	76.1 ± 1	76.1 ± 1.1	76.4 ± 0.9	75.9 ± 0.9
survi	74.4 ± 1.4	73.8 ± 1.5	73.8 ± 1.3	74.1 ± 1.2
vowel	89.4 ± 0.8	92.3 ± 0.5	93.3 ± 0.6	94.2 ± 0.8
wdbc	97.1 ± 0.5	97.2 ± 0.4	97.2 ± 0.5	97.2 ± 0.5

Table 7. Simple Ensemble - Stacked+ Combiner results

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	74.4 ± 1.4	73.6 ± 1.7	74.7 ± 1.1	74.5 ± 1.3
derma	97.2 ± 0.7	97.3 ± 0.7	97.5 ± 0.7	97.6 ± 0.7
ecoli	86.8 ± 1	86.3 ± 1.2	86.8 ± 1.1	86.8 ± 1
flare	81.9 ± 0.4	81.7 ± 0.7	81.5 ± 0.7	81.1 ± 0.7
img	96.7 ± 0.3	96.8 ± 0.3	97 ± 0.3	96.8 ± 0.2
ionos	92 ± 0.9	92.7 ± 1	92.9 ± 1.2	92.4 ± 1.2
pima	76.1 ± 1	75.7 ± 1	75.9 ± 1.2	75.9 ± 1
survi	73.9 ± 1.4	73.6 ± 1.4	73.8 ± 1.2	73.9 ± 1.4
vowel	89.8 ± 0.8	92.3 ± 0.6	93.3 ± 0.7	94.1 ± 0.7
wdbc	97.2 ± 0.5	97.4 ± 0.5	97.3 ± 0.5	97.3 ± 0.5

Table 8. Stacked Generalization Version 2 Results - Expert networks as Ensemble

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	72.5 ± 1.5	72.4 ± 1.5	72.4 ± 1.5	72.4 ± 1.5
derma	96.6 ± 0.7	96.8 ± 0.6	96.6 ± 0.9	96.3 ± 0.9
ecoli	86 ± 1	86 ± 0.9	85.7 ± 1	85.4 ± 1.1
flare	82 ± 0.6	81.8 ± 0.6	81.8 ± 0.6	81.9 ± 0.6
img	96.5 ± 0.2	96.6 ± 0.2	96.6 ± 0.3	96.6 ± 0.2
ionos	88 ± 1.3	89 ± 1.2	88.9 ± 1.2	89.1 ± 1.1
pima	77.4 ± 0.8	77.2 ± 0.7	77.4 ± 0.7	77.2 ± 0.8
survi	74.9 ± 1.4	74.9 ± 1.4	75.1 ± 1.4	75.3 ± 1.4
vowel	86.7 ± 0.7	88.6 ± 0.6	89.7 ± 0.7	89.8 ± 0.4
wdbc	96.9 ± 0.6	97 ± 0.6	97 ± 0.6	97 ± 0.6

Table 9. Stacked Generalization Version 2 Results - Whole model

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	74.8 ± 1.5	74 ± 2	75 ± 2	74.9 ± 1.9
derma	96.6 ± 0.9	96.5 ± 1	96.5 ± 1	96.6 ± 1.1
ecoli	85 ± 1.2	84.3 ± 1	85.4 ± 1.3	84.6 ± 1.1
flare	81.8 ± 0.7	81.8 ± 0.7	82 ± 0.8	81.8 ± 0.7
img	96.9 ± 0.2	97.1 ± 0.3	97.1 ± 0.3	97.2 ± 0.2
ionos	89.4 ± 1.2	90.6 ± 0.9	90.6 ± 0.7	91.1 ± 1
pima	76.3 ± 1.1	76.2 ± 1.1	76.5 ± 0.9	75.9 ± 1.4
survi	73.1 ± 1.2	73.4 ± 0.6	73.1 ± 1	73.4 ± 1.1
vowel	87 ± 0.5	90.7 ± 0.6	92.6 ± 0.8	92.7 ± 0.7
wdbc	96.7 ± 0.6	96.6 ± 0.5	96.6 ± 0.5	96.5 ± 0.6

Table 10. Stacked Generalization Version 2 Results - Expert networks as Ensemble

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	74.5 ± 1.3	74.3 ± 1	74.3 ± 1.4	74 ± 1
derma	97.5 ± 0.7	97.8 ± 0.6	97.3 ± 0.6	97.3 ± 0.4
ecoli	87.5 ± 0.9	87.2 ± 1	87.2 ± 0.9	86.6 ± 1.2
flare	82.1 ± 0.5	81.9 ± 0.7	82.1 ± 0.6	81.7 ± 0.7
img	96 ± 0.3	96.7 ± 0.2	96.9 ± 0.3	96.7 ± 0.3
ionos	88.9 ± 0.9	89.1 ± 0.9	89.7 ± 1.6	88.9 ± 1.5
pima	76.1 ± 1.2	76.8 ± 0.9	77.4 ± 1	75.8 ± 0.6
survi	74.1 ± 1.4	74.1 ± 1.5	75.1 ± 1.1	74.4 ± 1.3
vowel	86.9 ± 0.5	90.6 ± 0.5	91.1 ± 0.6	91.5 ± 0.7
wdbc	96.8 ± 0.5	97 ± 0.5	96.9 ± 0.5	96.7 ± 0.6

Table 11. Stacked Generalization Version 2 Results - Whole Stacked Model

	3 Nets	*9 Nets*	*20 Nets*	*40 Nets*
aritm	74.9 ± 1.3	74.5 ± 1	72.9 ± 1.3	75.6 ± 1
derma	97.5 ± 0.7	97.6 ± 0.7	97.3 ± 0.6	97.3 ± 0.4
ecoli	86.5 ± 0.9	86.3 ± 0.9	84.4 ± 1.1	86 ± 1.1
flare	81.6 ± 0.9	82.1 ± 0.8	81.4 ± 0.9	81.7 ± 0.7
img	97 ± 0.2	96.8 ± 0.3	96.8 ± 0.2	97 ± 0.2
ionos	90 ± 0.9	91 ± 1	90.1 ± 1.2	89.3 ± 1.5
pima	76.4 ± 1	76.7 ± 1	76.5 ± 1.4	74.5 ± 1
survi	73.9 ± 1.1	72.6 ± 1.1	72.8 ± 1.5	73.8 ± 1
vowel	88.1 ± 0.6	92.2 ± 0.5	92.8 ± 0.7	93.6 ± 0.6
wdbc	96.5 ± 0.5	96.9 ± 0.4	96.5 ± 0.6	96.6 ± 0.6

4.2 General Measurements

We have calculated the mean Increase of Performance (IoP eq.1) and the mean Percentage of Error Reduction (PER eq.2) across all databases to get a global measurement to

Table 12. General Results

Mean Increase of Performance

method	*3 nets*	*9 nets*	*20 nets*	*40 nets*
Simple Ensemble	0.69	1.01	1.04	1.16
Bagging	0.84	1.44	1.51	1.34
CVC	0.66	1.24	1.13	0.95
CVCv2	1.26	1.33	1.3	1.51
Adaboost	0.3	0.85	1.14	1.26
stacked on simple ensemble	1.13	1.44	1.51	1.59
stacked+ on simple ensemble	1.12	1.26	1.58	1.57
SG Ver.1 - Experts	0.28	0.56	0.64	0.62
SG Ver.2 - Experts	0.57	1.08	1.31	0.9
SG Ver.1 - Whole model	0.29	0.67	1.05	1.01
SG Ver.2 - Whole model	0.76	1.19	0.67	1.07

Mean Percentage of Error Reduction

method	*3 nets*	*9 nets*	*20 nets*	*40 nets*
Simple Ensemble	5.58	8.38	8.08	9.72
Bagging	6.85	12.12	13.36	12.63
CVC	6.17	7.76	10.12	6.47
CVCv2	10.25	10.02	7.57	7.48
Adaboost	1.32	4.26	9.38	12.2
stacked on simple ensemble	8.48	12.04	13.43	14.6
stacked+ on simple ensemble	9.4	11.8	14	13.89
SG Version 1 - Experts	0.7	3.46	3.52	2.78
SG Version 2 - Experts	3.44	9.22	9.48	6.31
SG Version 1 - Whole model	1.4	3.91	6.07	5.94
SG Version 2 - Whole model	5.57	10.14	4.86	7.82

compare the methods (Table 12). A negative value on these measurements means that the the ensemble performs worse than a single network.

$$IoP = Error_{SingleNet} - Error_{Ensemble} \tag{1}$$

$$PER = 100 \cdot \frac{Error_{SingleNet} - Error_{Ensemble}}{Error_{SingleNet}} \tag{2}$$

4.3 Discussion

Before the results discussion, the main results have been resumed in table 13 in which the best performance for each database is shown along with the method and number of networks we got it with.

In the resumed table, we can see that any version of the *Stacked Generalization* model do not appear on it. Although there are two cases in which the experts of *Stacked Generalization Ver.2* as ensemble is the best method, these experts are trained as in

Table 13. Best Resutls

	Performance	*Method*	*Networks*
aritm	77 ± 0.8	*CVCv2*	40
derma	98 ± 0.3	*CVCv2*	3
ecoli	87.5 ± 0.9	*SG Version 2 - Experts*	3
flare	82.7 ± 0.5	*CVC*	3
img	97.3 ± 0.3	*Adaboost*	9
ionos	92.9 ± 1	*Stacked*	9
pima	77.4 ± 1	*SG Version 2 - Experts*	20
survi	75.4 ± 1.6	*Adaboost*	3
vowel	97 ± 0.6	*Adaboost*	40
wdbc	97.5 ± 0.4	*Bagging*	20

CVC. We can extract from the table that the best individual results are provided by *Adaboost* and *Cross-Validation*.

Finally, we can see in the global measurements tables that the best results are got by *Cross-Validation* methods, Bagging and the combiners *Stacked* and *Stacked+*.

5 Conclusions

In this paper we have elaborated a comparison among traditional methodologies to build ensembles, two combiners based on *Stacked Generalization* and two stacked models proposed by Ting & Witten (SG Ver.1) by and Ghorbani & Owrangh (SG Ver.2).

Firstly, we can notice that *Bagging* and *Cross-Validation* are the best methods among the results related to the traditional ensemble methods according to the general results.

Secondly, we can see that the stacked combiners, *Stacked* and *Stacked+*, considerably improve the results got by the ensembles trained with *Simple Ensemble*. Moreover, these combiners on *Simple Ensemble* provide better results than some traditional ensemble methods. The best overall general measurements is provided by applying *Stacked* to a 40-network ensemble trained with *Simple Ensemble*.

Thirdly, we can see that the second version of the original *Stacked Generalization* model is better than first one. We can also see that the experts trained with the second version are also better. In fact, the first version is not on the top of the best performing methods, being similar to *Adaboost*.

Finally, comparing the results related to the first and second version of the original *Stacked Generalization* model, we can see that the performance of the whole stacked models is similar to the performance of their experts as ensemble. There is not a high increase of performance by the use of the combination network. Incredibly, the increase of performance got by the combiners *Stacked* and *Stacked+* with respect to their experts is considerable higher.

In conclusion, *Cross-Validation*, *Adaboost* and the combiners *Stacked* and *Stadked+* on *Simple Ensemble* are the best ways to build a *Multi-Net system*. *Adaboost* and the *Cross-Validation* methods got the best individual results whereas the general results show that the best Multi-Net system is provided by the 40-network version of *Simple*

Ensemble combined with *Stacked*. The two versions of the original *Stacked Generalization* model do not get neither the best individual performance for a database nor the best global results. Moreover, there are some cases in which the two versions of the original *Stacked Generalization* perform worse than a *Simple Ensemble*.

References

1. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, Inc., New York (1995)
2. Fernndez-Redondo, M., Hernndez-Espinosa, C., Torres-Sospedra, J.: Multilayer feedforward ensembles for classification problems. In: Pal, N.R., Kasabov, N., Mudi, R.K., Pal, S., Parui, S.K. (eds.) ICONIP 2004. LNCS, vol. 3316, pp. 744–749. Springer, Heidelberg (2004)
3. Ghorbani, A.A., Owrangh, K.: Stacked generalization in neural networks: Generalization on statistically neutral problems. In: Proceedings of the International Joint conference on Neural Networks, IJCNN 2001, Washington D.C., USA, pp. 1715–1720. IEEE, Los Alamitos (2001)
4. Hernndez-Espinosa, C., Torres-Sospedra, J., Fernndez-Redondo, M.: New experiments on ensembles of multilayer feedforward for classification problems. In: IJCNN 2005 proceedings, pp. 1120–1124 (2005)
5. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience, Chichester (2004)
6. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
7. Raviv, Y., Intratorr, N.: Bootstrapping with noise: An effective regularization technique. Connection Science, Special issue on Combining Estimators 8, 356–372 (1996)
8. Sharkey, A.J. (ed.): Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems. Springer, Heidelberg (1999)
9. Ting, K.M., Witten, I.H.: Stacked generalizations: When does it work? In: International Joint Conference on Artificial Intelligence proceedings, vol. 2, pp. 866–873 (1997)
10. Ting, K.M., Witten, I.H.: Issues in stacked generalization. Journal of Artificial Intelligence Research 10, 271–289 (1999)
11. Torres-Sospedra, J., Hernndez-Espinosa, C., Fernndez-Redondo, M.: Combining MF networks: A comparison among statistical methods and stacked generalization. In: Schwenker, F., Marinai, S. (eds.) ANNPR 2006. LNCS (LNAI), vol. 4087, pp. 210–220. Springer, Heidelberg (2006)
12. Torres-Sospedra, J., Hernndez-Espinosa, C., Fernndez-Redondo, M.: Designing a new multilayer feedforward modular network for classification problems. In: WCCI 2006 proceedings, pp. 2263–2268 (2006)
13. Tumer, K., Ghosh, J.: Error correlation and error reduction in ensemble classifiers. Connection Science 8(3-4), 385–403 (1996)
14. Wolpert, D.H.: Stacked generalization. Neural Networks 5(6), 1289–1301 (1994)