A Genetic Algorithm for Designing Neural Network Ensembles

Symone G. Soares  
Institute of Systems and Robotics (ISR-UC),  
Department of Electrical and Computer Engineering (DEEC-UC),  
University of Coimbra, Pólo II,  
3030-290 Coimbra, Portugal  
symonesoares@isr.uc.pt

Carlos H. Antunes  
Department of Electrical and Computer Engineering (DEEC-UC),  
and INESC Coimbra,  
University of Coimbra, Pólo II,  
3030-290 Coimbra, Portugal  
+351239851040  
ch@deec.uc.pt

Rui Araújo  
Institute of Systems and Robotics (ISR-UC),  
Department of Electrical and Computer Engineering (DEEC-UC),  
University of Coimbra, Pólo II,  
3030-290 Coimbra, Portugal  
rui@isr.uc.pt

ABSTRACT

Ensemble Methods (EMs) are sets of models that combine their decisions, or their learning algorithms, or different data to obtain good predictions. The motivations are the possibility of improving the generalization capability and the overall system performance. However, several issues are at stake in EM development, such as the design of models that disagree as much as possible on the same data, the selection of some of them and their optimal combination to enhance the robustness of the ensemble. Since there is no unified procedure to implement these steps, this paper proposes a new methodology to design Neural Network (NN) ensembles using a Genetic Algorithm (GA). Firstly, a set of NNs with high degree of diversity is produced. The aim is to draw a different training data set for each NN by applying bootstrap. The architecture of the NN is selected by varying the number of hidden neurons, activation functions and initialization of weights. Secondly, a GA is employed to select both the best subset of NNs and the optimal combination strategy for ensuring the accuracy and the robustness of the ensemble. Experiments on well-known data sets are reported to evaluate the effectiveness of the proposed methodology.

Categories and Subject Descriptors

1.2.6 [Artificial Intelligence]: Learning—Connectionism and neural nets; 1.2.8 [Artificial Intelligence]: Problem Solving, Control Methods, and Search—Heuristic methods.

General Terms

Algorithms, Performance, Design.

Keywords

Ensemble Methods, Neural Networks, Genetic Algorithms.

1. INTRODUCTION

Ensemble Methods (EMs) are sets of models that combine in some way their decisions, or their learning algorithms, or different data to achieve accurate predictions. Research has shown that an EM is generally more accurate than any single model, and the effectiveness of EMs has been shown in different benchmark data sets [7, 3, 19, 6, 2]. Neural Network (NN) ensembles have been widely investigated for both classification and regression problems. The main motivations are the possibility of improving the generalization capability and the overall system performance. Theoretical and empirical investigations have revealed that a good EM is one in which the models make different errors on the same data point [3]. Nowadays EMs represent one of the main research lines in Machine Learning [21].

In general, the design of an EM can be divided into three main steps: generation of candidate models, selection of some of them, and combination of them to produce the output of the EM. Since there is no unified procedure to implement these steps, different approaches including meta-heuristics have been proposed for automatic development of EMs [4].

For example, the Genetic Algorithm based Selective Ensemble (GASEN) trains NNs using bootstrap for increasing the diversity and then employs a Genetic Algorithm (GA) to select an optimum subset of NNs to constitute the ensemble. In this approach each NN is assigned a weight derived from the marginal improvement in the fitness associated with including this NN in the ensemble. Then the NNs whose weights are higher than a predefined threshold are selected to be included in the ensemble [27]. However, the NNs have fixed architectures and the combination strategies are limited by simple average and weighted average for regression and classification, respectively.

An automatic design of Neural Network Ensemble using Evolutionary Learning and Negative Correlation Learning (EENCL) is presented in [19]. Negative Correlation Learning (NCL) creates negatively correlated NNs using a correlation penalty term in the error function to encourage specialization and cooperation among the NNs. EENCL does not explore the linear combination between the NNs and the NN architectures are also predefined. Bayesian Artificial Immune System (BAIS) is an immune-inspired methodology to design EM of NNs with better generalization performance.
when compared to EENCL. BAIS introduces diversity in the architecture of the individual NNs. However, only one type of combination is performed to aggregate the NNs [8].

This paper proposes a new methodology to design neural network ensembles for regression problems using GA. In the first phase, a set of NNs with high diversity degree is produced. For each NN, the proposed approach generates a different training data set by applying bootstrap and then selects the best NN architecture by varying the number of hidden neurons, activation functions and initialization of weights. In the second phase, a GA is employed to select both the best subset of NNs and the optimal type of combination for them. The individuals of the population are coded as a sequence of bits that contains information about the subset of NNs and the type of combination to aggregate them. For each individual the optimal combination strategy is determined to ensure the accuracy and robustness of the EM. Experiments on two well-known data sets are performed to evaluate the effectiveness of the proposed methodology.

The work is organized as follows. In Section 2, important issues for creating an EM are outlined. Section 3 reports the key issues in EM. Section 4 details the proposed methodology, named as Genetic Algorithm for Designing Neural Network Ensembles (GA-NNE). The experimental results are reported and analyzed in Section 5. Finally, Section 6 contains some concluding remarks.

2. CREATING AN ENSEMBLE METHOD

The combination strategy of the EM is of fundamental importance. Optimal combination can enhance the robustness and accuracy of an EM. In this paper, robustness is defined as the capability of the EM to perform well on new data points. The scope of this paper is restricted to the combination at the prediction level. Let us assume an initial data set $D_{init} = \{ (x_i, y_i) \}_{i=1}^k$ of size $k$, where $x_i \in \mathbb{R}^v$ ($v$ is the number of input variables) and $y_i$ is an output variable. $D_{init}$ is split as a training data set $D_{train} = \{ (x_i, y_i) \}_{i=1}^{k_1}$, a validation data set $D_{valid} = \{ (x_i, y_i) \}_{i=1}^{k_2}$, and a testing data set $D_{test} = \{ (x_i, y_i) \}_{i=1}^{k_3}$, of size $k_1$, $k_2$ and $k_3$, respectively, where $k = k_1 + k_2 + k_3$.

Bootstrap [9] is applied on the training data set to create a different training data set for each NN. Bootstrap can be used to expand upon a single realization of a distribution or to generate a data set that can provide a better understanding of the mean and variability of the original unknown distribution [15]. Several studies have applied bootstrap in EM modeling to create diverse models [10]. In this case, each NN of the ensemble is trained with a bootstrapped data set from the training data set.

Assuming a set of $n$ NNs for implementing an EM, the proposed approach trains each NN $j$ using a different bootstrapped data set. This is done by random drawing $k_1$ times with replacement from the training data set. The performance of an NN $j$ is evaluated by the Mean Square Error (MSE) between the estimated output $f_j()$ and real output $y$ using the validation data set.

After training and evaluation, the NNs are aggregated using the predictions of the testing data set. Considering $n$ the number of NNs in the ensemble and $F()$ the output of the EM, the proposed combinations are:

1. **Mean**: the output of the EM is obtained as the average of all models’ predictions:
   \[ F(x_i) = \frac{1}{n} \sum_{j=1}^{n} f_j(x_i). \]  

2. **Trimmed Mean**: the output of the EM is calculated as the trimmed mean of all models’ outputs. This approach removes the lowest models’ outputs and highest models’ outputs before calculating the mean, avoiding extreme values of the outputs. For $P\%$ trimmed mean, the mean is obtained by excluding $P\%/2$ from highest models’ outputs and $P\%/2$ from lowest models’ outputs. (This paper sets $P\%$ as 10%).

3. **Median**: the output of the EM is equivalent to the median among all the NNs’ outputs:
   \[ F(x_i) = \text{median} \{ f_j(x_i) \}. \]

4. **Weighted Mean**: the output of the EM is obtained by taking a weighted sum of the NNs’ outputs, where each weight $w_j$ of a NN $j$ is based on its accuracy. Here the accuracy of each NN is determined by the MSE using the validation data set. The EM’s output is expressed as:
   \[ F(x_i) = \sum_{j=1}^{n} w_j \cdot f_j(x_i). \]

Each weight $w_j$ is determined by [8]:
   \[ w_j = \frac{\text{adjusted MSE}_j}{\sum_{k=1}^{n} \text{adjusted MSE}_k}, \]

where
   \[ \text{adjusted MSE}_j = 1 - \text{average MSE}_j, \]
   \[ \text{average MSE}_j = \frac{\text{MSE}_j}{\sum_{k=1}^{n} \text{MSE}_k}. \]

The performance of the EM is evaluated by the MSE between the estimated output $F()$ and the actual output $y$ using the test data set $MSE_{test}$.

3. KEY ISSUES IN ENSEMBLE METHODS

Diversity is one of the key issues for the success of Ensemble Methods (EMs). Empirical results reveal that the performance of an EM is related to the diversity and accuracy of the models [5]. In the development of EMs, the information about diversity can be taken into account; i.e. the system may or may not explicitly try to optimize some metric of diversity during the design of the EM. These two approaches are classified as explicit and implicit diversity methods, respectively.

Bagging (short for Bootstrap Aggregation Learning) uses an implicit method. It randomly samples the training data by bootstrap to produce a different data set for each model [2]; at no point a measurement is taken to promote diversity. Bagging has proved to be a popular technique in many fields of application [6, 15]. Boosting performs an explicit method. It manipulates the training data distributions to ensure diversity in the set of models [11]. However, it is not guaranteed to be the “right” form of diversity.
According to Brown et al. [3], the diversity of the EM can be dealt with in three main ways: (i) by starting the learning with different conditions; (ii) by varying the set of models; and (iii) by modifying the trajectory used by the components in the search space.

The first way (i) aims to diversify the initial components of the individual models. For example, for EMs of NNs starting each NN with a different initialization of weights can increase the probability of continuing in a different trajectory with respect to the other NNs. Since usually the NNs are not diverse enough, this approach generally does not achieve good results [4].

The second approach (ii) attempts to alter the set of models available to the EM: firstly by manipulating the training data set that it receives (e.g., k-fold cross-validation [22], Bagging, Boosting or noise injection [10]); and secondly by modifying the model architecture itself (e.g., NNs with different architectures or different activation functions) or designing EM with heterogeneous learning algorithms (e.g., Bayesian, RBF NNs and Support Vector Machines) [6].

The last category of approaches to achieve diversity (iii) aims to alter the way the search space is traversed, leading different NNs to converge to different hypotheses. This technique can be subdivided into: evolutionary methods and penalty methods. The most popular penalty method for encouraging diversity in EMs is the NCL proposed by Liu et al. [18]. It introduces a correlation penalty term into the cost function of the NNs so that each NN minimizes its error together with the error correlation within the EM. Evolutionary algorithms can also evolve a population of NNs using some heuristic to maintain the diversity of the population. In [17] a multi-objective evolutionary approach is proposed to explore both accuracy and diversity between the members of the EM.

Evolutionary methods and penalty methods can also be hybridized. In [19], a penalty term is used to explore the interac- tion and diversity between the NNs and the evolutionary method addresses the issue of automatic determination of the number of NNs of the EM.

Evolutionary methods and other meta-heuristic optimization approaches, such as Genetic Algorithms [27], Particle Swarm Optimization [26] and Bayesian Artificial Immune System [8], have been used to select the best subset of models among a set of base models. In this setting some issues are at stake, such as how to produce the individual models, how to evaluate the performance of the models and what selection strategy to be performed [21].

Moreover, the way to aggregate the different outputs of the models is crucial to enhance the performance [24] and to balance the diversity between the models of the EM. However, most selection methods for ensembles consider a single combination strategy.

In order to address all the reported issues on the development of EMs, the proposed work performs (i) and (ii) to achieve diversity and, consequently, good generalization capability. The aim is to produce NNs using different training data sets by bootstrap and then choose the best architecture of each NN by varying the neurons of the hidden layer, the initialization of weights and the activation functions. Moreover, a GA is employed to select the best subset of NNs and the optimal combination strategy for them.

4. PROPOSED METHODOLOGY

The proposed method, named GA-NNE, aims to build neural network ensembles using a GA as detailed in Algorithm 1. Firstly, a set of candidate NNs with a high degree of diversity is produced. Then the GA is employed to select both the best subset of NNs to join the ensemble and the best type of combination for aggregating the NNs. Each individual of the population represents an ensemble to be implemented.

Algorithm 1 Genetic Algorithm for Designing Neural Network Ensembles (GA-NNE)

Input $n$: number of candidate NNs.

1. Divide the data set into training data set, validation data set and testing data set; generate $n$ copies from the training data set and apply bootstrap on the $n$ training data sets.

2. Select $n$ NNs for each manipulated training data set; the topology of each NN is chosen among a number of NNs based on the performance of the validation data set.

3. Set $t \leftarrow 1$. Consider $c$ the number of bits that represents the types of combinations, randomly produce an initial population $P_1$ with $m$ individuals of size $(n+c)$.

4. Evaluate the fitness of all individuals of $P_1$ using all available types of combinations.

5. Assign to each individual of $P_1$ the best type of combination based on the fitness.

6. Repeat:

   (a) Perform Roulette Wheel Selection to choose $p^\%$ of the individuals of $P_t$.

   (b) Apply crossover on the selected individuals to produce a population of offspring $O_t$.

   (c) Randomly mutate $p^\%$ of the individuals of $O_t$ on the first $n$ bits (the model part).

   (d) Calculate the fitness of all individuals of $O_t$ using all available types of combinations.

   (e) Assign to each individual of $O_t$ the best type of combination based on the fitness.

   (f) Select individuals for the next generation ($t + 1$):

      i. Consider a temporary population $P'_t \leftarrow (O_t \cup P_t)$ and $P_{t+1}$ the population for the next generation;

      ii. Elitism: assign to $P_{t+1}$ the $e$ individuals of $P'_t$ with the best fitness;

      iii. Perform Roulette Wheel Selection on $P'_t$ to select $(m-e)$ individuals for $P_{t+1}$.

   (g) Set $t \leftarrow t + 1$.

until $t = maxgenerations$;

As described in Section 2, the data set is split into training data set, validation data set and testing data set. Assuming that the aim is to produce $n$ candidate NNs, these
are trained using bootstrap data sets from the training data set. The NNs are implemented with one hidden layer using Multilayer Perceptron Neural Network (MLP NN) by Levenberg-Marquardt (LM) algorithm [13]. LM algorithm has become a standard technique for non-linear problems, widely adopted in a broad range of applications. It is a combination of the steepest descent and the Gauss-Newton methods.

The topology of each NN is chosen among a number of NNs based on the MSE of the validation data set. The number of NNs is obtained by varying the number of neurons in the hidden layer (from 1 to 10); two activation functions (linear and fast hyperbolic tangent) for the hidden layer and output layer; and three different initializations of weights. These techniques for initialization of weights are:

1. randomize the initial weights with a small number in an interval of \([-1/ni, 1/ni]\), where \(ni\) is the number of neuronal inputs [16];
2. Nguyen-Widrow initialization method, where the initial weights are set according to the NN learning [20];
3. uniform random initialization method in the interval of \([-0.5, 0.5]\) [25];

The encoding of the chromosomes is binary, as displayed in Figure 1. The representation is composed by two parts: models and type of combination. The first part contains information about the subset of NNs for composing the ensemble. Considering a set of \(n\) NNs \(\{net_1, \ldots, net_n\}\), each locus of the model part is related to the absence “0” or presence “1” of a model \(net_j\) in the ensemble. The second part refers to the type of combination to be used for aggregating the subset of models. Assuming that the number of types of combinations is \(z\), the number of bits \(c\) to represent them is given by \(c = \lceil \log_z^\ast \rceil\), where \(\lceil x \rceil\) is the smallest integer not lower than \(x\). As reported in Section 2, the number of types of combinations is four: mean, trimmed mean, median and weighted mean; their binary representations being “00,” “01,” “10” and “11,” respectively.

![Figure 1: Binary Representation of a Chromosome.](image)

Figure 2 displays an example of a chromosome representation using a set of ten models \(\{net_1, \ldots, net_{10}\}\). Figure 2a shows the binary encoding of the chromosome. Figure 2b displays the decoding of the same chromosome. The final subset of NNs \(\{net_1, net_2, net_3, net_4, net_5, net_6, net_7, net_8, net_9, net_{10}\}\) is selected according to the binary bits on the model part. In this example, the selected combination strategy is mean (i.e. “00”).

Other important issue is to define the fitness function. This work considers \(1/\text{MSE}^{test}\) as fitness function, where \(1/\text{MSE}^{test}\) is calculated as described in Section 2. In Algorithm 1, step (3) produces an initial population \(P_t\) with \(m\) individuals. In step (4), \(P_t\) is evaluated by the fitness function using all available types of combinations. In step (5), for each individual of \(P_t\), the best type of combination is assigned based on the fitness. The combination is an important step for designing a robust ensemble, and this approach ensures that the ensemble will be always built using the optimal combination.

![Figure 2: Example of a Chromosome - (a) Binary Chromosome Representation; (b) Decoding the Chromosome.](image)

In step (6), the proposed GA loops over \(t\) generations to make the population evolve. In sub-step 6a, \(P_t^{1\%}\) of the individuals of \(P_t\) are chosen using Roulette Wheel Selection. This selection method picks an individual of \(P_t\) to be a parent with a probability proportional to its fitness.

In sub-step 6b, the selected parents are combined to form a population of offspring \(O_t\). The best results have been achieved with uniform crossover using a random mask, where two parents \((\text{parent}_1\text{ and parent}_2)\) produce two offspring \((\text{offspring}_1\text{ and offspring}_2)\). The crossover is only performed on the bits of the model part. A mask is a vector of binary bits with length \(n\). When the bit in the mask is “0,” the corresponding bit in \(\text{parent}_1\) is passed to \(\text{offspring}_1\) and the corresponding bit in \(\text{parent}_2\) is passed to \(\text{offspring}_2\). When the bit in the mask is “1,” the corresponding bit in \(\text{parent}_1\) is passed to \(\text{offspring}_2\) and the corresponding bit in \(\text{parent}_2\) is passed to \(\text{offspring}_1\) [14].

In sub-step 6c, \(P_t^{50\%}\) of random individuals of \(O_t\) are selected to be mutated only on the model part. The aim is to change randomly one or more bits of each individual to foster diversity. In sub-step 6d and sub-step 6e, \(O_t\) is evaluated and the best type of combination is assigned to each individual as described above.

An important issue is to pick the individuals for the next generation \((t + 1)\). The computational experiments revealed that elitism provides better results. The algorithm selects \(e\) individuals with best fitness \(\text{fitness}\) of both populations \(O_t\) and \(P_t\), and then it assigns these \(e\) individuals to the next generation population \(P_{t+1}\), where \(P_{t+1}\) has size \(m\). The other \((m - e)\) individuals are chosen by Roulette Wheel Selection. After \(t\) generations, the best individual of the population of the last generation is chosen as the solution to the problem.

5. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed methodology, several experiments have been carried out and their results are summarized below.

5.1 Data Set Descriptions

The tests are performed using two data sets available on Luís Torgo’s website [23]: Boston Housing and Friedman.
Friedman is a nonlinear prediction problem [12]. It is an artificial data set with 10 independent predictor variables \( x = [x_1, x_2, \ldots, x_{10}] \) each of which is uniformly distributed over \([0, 1]\). The function is given by:

\[
y = 10 \sin(\pi x_1 x_2) + 20 \left( x_3 - \frac{1}{2} \right)^2 + 10 x_4 + 5 x_5 + \epsilon.
\]

The dimensionality of the covariance space is increased, so that instead of one noise \( \epsilon \), there are now more five independent variables \( x_6, \ldots, x_{10} \). The data set includes 40768 samples.

The Boston Housing is a well-known data set [1]. The goal is to predict housing values in areas of Boston. The data set contains 13 input variables (mainly socio-economic), 1 output variable (median housing price) and 506 samples.

The tests are evaluated using 10-fold cross-validation, where each data set is split into 10 subsets. The result is the average of the results of the 10 subsets. So each subset is in turn used as testing data set while the samples of the other 9 subsets are randomly divided as a training data set and a validation data set (10%). Therefore, results of MSE\text{test} using 10-fold cross-validation are given by averaging the MSE of all 10 testing subsets.

### 5.2 Experiment 1 - Candidate Models

Before starting the experiments, it is important to evaluate the performance of the individual NNs. For this purpose, tests were done by varying the number of neurons in the hidden layer. Other parameters were set according to popularity, such as, Nguyen-Widrow method as initialization of methods, fast hyperbolic tangent as activation function for the hidden layer and linear as activation function for the output layer. Figure 3 displays the accuracy of the NNs, where the MSE is calculated using 10-fold cross-validation.

![Figure 3: Performance of the Individual NNs.](image)

According to the results provided by the individual NN, it can be seen that the performance on unseen data (e.g. validation and testing data set) is poor when compared to the training data set. Moreover, NNs with lower number of neurons in the hidden layer have better generalization capability.

A set of 20 candidate NN models is generated using only steps (1) and (2) of Algorithm 1. The training of the NNs stops when the MSE on the training data set reaches \(10^{-4}\) or the number of iterations reaches 500.

Table 1 reports the MSE on the testing data set if all the candidate models are used to design ensembles. In this paper, we refer to this technique as Ensembling All, i.e. it aggregates all the models of the ensemble without performing the optimization technique to select the best subset of models. The candidate models are employed to design ensembles using mean, trimmed mean, median, and weighted mean as type of combination. In the Friedman data set, all the types of combinations have the same performance. In the Boston Housing data set, trimmed mean outperforms the other types of combinations.

### Table 1: Results of MSE\text{test} using Ensembling All. In the table, all the MSE values have been multiplied by \(10^3\).

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Type of Combination</th>
<th>mean</th>
<th>tr. mean</th>
<th>median</th>
<th>wt. mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Friedman</td>
<td>1.000</td>
<td>2.000</td>
<td>2.000</td>
<td>2.000</td>
<td>2.000</td>
</tr>
<tr>
<td>Boston H.</td>
<td>5.000</td>
<td>4.900</td>
<td>5.100</td>
<td>5.000</td>
<td>5.000</td>
</tr>
</tbody>
</table>

### 5.3 Experiment 2 - GA-NNE

In this approach, Algorithm 1 was implemented to select the best subset of NNs and the best type of combination. The models of the Ensembling All using 20 NNs are also employed here as candidate models. Several experiments were performed by varying the GA parameters, such as number of the mutated bits (1 and 2), the mutation probability \(p_m\) (5%, 10% and 15%), the selection probability \(p_s\) (60% and 100%) and the number of individuals of the population (20, 40 and 60). The crossover probability is 1. The number of individuals selected by elitism was set as 1.

Table 2 and Table 3 show the experimental results of the GA-NNE using the Friedman and Boston Housing data sets on 20 runs. For both data sets, some characteristics are noticed. For example, 1 mutated bit has inferior results when compared to 2 mutated bits; mutation probability of 10% has the same performance when compared to mutation probability of 15%, and these probabilities outperform a probability of 5%; selection probability of 60% is more efficient than selection probability of 100%; and, improvements are observed when the number of individuals increases from 20 to 40 and 60.

Figure 4 shows the GA properties of the run with min value of the experiment with best performance for Friedman and Boston data set. On Table 2 and 3, these runs are displayed in bold and they have values of 1.788 and 2.431, respectively. Mean is the average of the MSE\text{test} of all individuals of the population in a generation (i.e. 1/fitness) and Best is the best value of MSE\text{test} of all individuals of the population in a generation. As it can be seen, no improvements are shown after generation 500 for the Friedman and 700 for the Boston Housing. The percentages of selection of each type of combination on all runs are reported in Table 4. For both data sets median is the best strategy.
Table 2: Experimental Results of the GA-NNE Using the Friedman Data Set. In the table, all the MSE values have been multiplied by $10^3$.

<table>
<thead>
<tr>
<th>No. of Mutated bits</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mutation Probability ($p_{m}^%$)</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>10%</td>
<td>15%</td>
</tr>
<tr>
<td>Selection Probability ($p_{s}^%$)</td>
<td>60%</td>
<td>100%</td>
<td>60%</td>
<td>100%</td>
<td>60%</td>
<td>60%</td>
<td>60%</td>
<td>60%</td>
</tr>
<tr>
<td>No. of Individuals ($m$)</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>40</td>
<td>60</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Mean</td>
<td>1.798</td>
<td>1.799</td>
<td>1.794</td>
<td>1.794</td>
<td>1.792</td>
<td>1.792</td>
<td>1.793</td>
<td>1.793</td>
</tr>
<tr>
<td>S.D.</td>
<td>0.004</td>
<td>0.003</td>
<td>0.003</td>
<td>0.002</td>
<td>0.002</td>
<td>0.002</td>
<td>0.002</td>
<td>0.001</td>
</tr>
<tr>
<td>Min</td>
<td>1.790</td>
<td>1.795</td>
<td>1.788</td>
<td>1.788</td>
<td>1.788</td>
<td>1.788</td>
<td>1.788</td>
<td>1.788</td>
</tr>
<tr>
<td>Max</td>
<td>1.806</td>
<td>1.808</td>
<td>1.802</td>
<td>1.798</td>
<td>1.797</td>
<td>1.797</td>
<td>1.797</td>
<td>1.797</td>
</tr>
</tbody>
</table>

Table 3: Experimental Results of the GA-NNE Using the Boston Housing Data Set. In the table, all the MSE values have been multiplied by $10^3$.

<table>
<thead>
<tr>
<th>No. of Mutated bits</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mutation Probability ($p_{m}^%$)</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>5%</td>
<td>10%</td>
<td>15%</td>
</tr>
<tr>
<td>Selection Probability ($p_{s}^%$)</td>
<td>60%</td>
<td>100%</td>
<td>60%</td>
<td>100%</td>
<td>60%</td>
<td>60%</td>
<td>60%</td>
<td>60%</td>
</tr>
<tr>
<td>No. of Individuals ($m$)</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>40</td>
<td>60</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Mean</td>
<td>2.502</td>
<td>2.503</td>
<td>2.455</td>
<td>2.457</td>
<td>2.443</td>
<td>2.442</td>
<td>2.440</td>
<td>2.440</td>
</tr>
<tr>
<td>S.D.</td>
<td>0.037</td>
<td>0.042</td>
<td>0.019</td>
<td>0.016</td>
<td>0.010</td>
<td>0.009</td>
<td>0.008</td>
<td>0.005</td>
</tr>
<tr>
<td>Min</td>
<td>2.458</td>
<td>2.455</td>
<td>2.431</td>
<td>2.439</td>
<td>2.431</td>
<td>2.431</td>
<td>2.432</td>
<td>2.431</td>
</tr>
<tr>
<td>Max</td>
<td>2.594</td>
<td>2.607</td>
<td>2.510</td>
<td>2.494</td>
<td>2.459</td>
<td>2.460</td>
<td>2.461</td>
<td>2.452</td>
</tr>
</tbody>
</table>

Figure 4: Mean of the fitness of the population and the fitness of the best individual of the population on a generation - (a) Friedman data set; (b) Boston Housing data set.

Table 4: GA-NNE - Percentage of Selection of the Types of Combinations.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Type of Combination</th>
<th>mean</th>
<th>tr. mean</th>
<th>median</th>
<th>wt. mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Friedman</td>
<td>1.00%</td>
<td>0.00%</td>
<td>93.50%</td>
<td>5.50%</td>
<td></td>
</tr>
<tr>
<td>Boston H.</td>
<td>10.00%</td>
<td>0.00%</td>
<td>79.50%</td>
<td>10.50%</td>
<td></td>
</tr>
</tbody>
</table>

5.4 Experiment 3 - Comparisons of the Strategies for Ensembling

In this Subsection, the proposed methodology GA-NNE is compared to Bagging, *Ensembling All* and GASEN. In Bagging, the ensemble is composed by 100 NNs trained using bootstrap resampling from the training data set. The NNs are trained with Levenberg-Marquardt algorithm and they have fixed architecture (MLP NN) and parameters. In this paper, these parameters are set as: 5 neurons in the hidden layer; initialization of weights proposed by Kasabov et [16]; fast hyperbolic tangent activation function in the hidden layer; linear function in the output layer; the training stops when the value of the MSE on the training data set reaches $10^{-4}$. The combination strategy is mean.

GASEN is an approach for selecting models from a set of available NNs to constitute an ensemble using the GA. This method selects NNs to be included in the ensemble based
on the evolving weights. Initially, the NNs are assigned as random weights and finally the approach selects those NNs which have weights higher than a predefined threshold $\lambda$ to join the ensemble. The tests are performed using the code available at the website [27]. The number of candidate NNs is 20. Other parameters are set to the default values: NNs trained with Backpropagation; MLP architecture with one hidden layer; 5 neurons in the hidden layer; hyperbolic tangent sigmoid as activation function in the hidden layer; linear function in the output layer; stopping criterion when the number of epochs reaches 100; number of individuals of the population equal to 20; threshold ($\lambda$) for selecting the NNs as $1/20$.

Table 5 and Table 6 report the main results for both Friedman and Boston Housing data sets. For Bagging and GASEN the experiments involved 20 runs. In Ensembling All, the results of the ensemble using 20 NNs are reported, i.e. the same results obtained in Subsection 5.2. The strategy with best performance is displayed in bold in Table 5 and Table 6.

For both data sets, Bagging presents worse results when compared to the other approaches. This can be explained as follows. Firstly, the diversity of the ensemble is achieved just by manipulating the training data set. Secondly, no selection method is employed to choose the best subset of NNs and type of combination. On the other hand, Ensembling All has better performance when compared to Bagging. The success of Ensembling All is explained by the two levels of diversity: (i) by using different initializations of weights and (ii) by altering the set of NNs accessible to the ensemble (i.e. by manipulating the data set and varying the architectures of the NNs).

With respect to the Friedman data set, GASEN has better generalization capability than Ensembling All. On the contrary, for the Boston Housing data set, GASEN has worse performance when compared to Ensembling All. This reveals that, in some cases, it is better ensembling all models than applying a selection method. However, GASEN has smaller computational cost than Ensembling All. In Ensembling All, the search for the best NN architecture in a pool of available NNs has an expensive time cost.

The experiments reveal that a subset selection of models when combined with a selection of the type of combination (e.g. GA-NNE) can produce good results in comparison with ensemble all models (e.g. Ensembling All). Moreover, as can be observed, GA-NNE outperforms GASEN, because GASEN designs an ensemble using only one type of combination and with NNs that have fixed architectures.

The average number of ensemble models used by GASEN and GA-NNE is presented in Table 7, where the results are obtained over the 20 runs. From the experimental results, it can be seen that GASEN selects less models for ensembling when compared to GA-NNE.

### Table 5: Experimental Results of Different Methods Using the Friedman Data Set. In the table, all the MSE values have been multiplied by $10^3$.

<table>
<thead>
<tr>
<th>MSE</th>
<th>Bagging</th>
<th>Ensembling All</th>
<th>GASEN</th>
<th>GA-NNE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean tr. mean median wt. mean</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>7.501 2.100</td>
<td>2.100 2.100</td>
<td>2.100 2.100</td>
<td>1.927 1.792</td>
</tr>
<tr>
<td>S.D.</td>
<td>0.022 -</td>
<td>- -</td>
<td>- -</td>
<td>0.035 0.002</td>
</tr>
<tr>
<td>Min</td>
<td>7.466 -</td>
<td>- -</td>
<td>- -</td>
<td>1.870 1.788</td>
</tr>
<tr>
<td>Max</td>
<td>7.543 -</td>
<td>- -</td>
<td>- -</td>
<td>1.980 1.797</td>
</tr>
</tbody>
</table>

### Table 6: Experimental Results of Different Methods Using the Boston Housing Data Set. In the table, all the MSE values have been multiplied by $10^3$.

<table>
<thead>
<tr>
<th>MSE</th>
<th>Bagging</th>
<th>Ensembling All</th>
<th>GASEN</th>
<th>GA-NNE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean tr. mean median wt. mean</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>7.849 5.000</td>
<td>4.900 5.100</td>
<td>5.000 5.000</td>
<td>6.349 2.440</td>
</tr>
<tr>
<td>S.D.</td>
<td>0.107 -</td>
<td>- -</td>
<td>- -</td>
<td>0.876 0.005</td>
</tr>
<tr>
<td>Min</td>
<td>7.666 -</td>
<td>- -</td>
<td>- -</td>
<td>5.461 2.431</td>
</tr>
<tr>
<td>Max</td>
<td>8.059 -</td>
<td>- -</td>
<td>- -</td>
<td>9.015 2.452</td>
</tr>
</tbody>
</table>

### Table 7: Average Number of Models in the Ensemble.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>GASEN</th>
<th>GA-NNE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Friedman</td>
<td>4.160</td>
<td>5.325</td>
</tr>
<tr>
<td>Boston H.</td>
<td>4.895</td>
<td>4.985</td>
</tr>
</tbody>
</table>

6. CONCLUSIONS

Ensemble Methods (EMs) of Neural Networks (NNs) are powerful techniques for improving the generalization capabilities and overall system performance. However, several issues are at stake in EM development, such as generation of candidate NNs that disagree as much as possible on the same data point, the selection of some NNs and the optimal combination to enhance the robustness of the EM.

Since there is no unified procedure to implement these steps, this paper proposed a methodology for automatic design of NN ensembles using Genetic Algorithm (GA), named GA-NNE. Firstly, a set of NNs with high degree of diversity is designed. Secondly, a GA is employed to select both the
best subset of NNs and the optimal type of combination for them.

This paper compared GA-NNE with other approaches, such as Bagging. Ensembling All and GASEN. The tests using two well-known data sets have shown the effectiveness of the GA-NNE. The experiments have shown that our approach outperforms other methodologies. This is explained by the high diversity degree of the NNs. The diversity is achieved by using NNs with different numbers of hidden neurons, activation functions, initializations of weights and training data sets. The selection of the optimal type of combination was fundamental to enhance the accuracy and robustness of the EM.

7. REFERENCES