Evolutionary Design of Neural Networks for Classification and Regression[1]

Miguel Rocha*, Paulo Cortez†, José Neves*
*Dep. Informática, University of Minho, Portugal
†Dep. Sistemas de Informação, University of Minho, Portugal
E-mail: mrocha@di.uminho.pt pcortez@dsi.uminho.pt jneves@di.uminho.pt

Abstract

The *Multilayer Perceptrons (MLPs)* are the most popular class of *Neural Networks*. When applying *MLPs*, the search for the ideal architecture is a crucial task, since it should should be complex enough to learn the input/output mapping, without overfitting the training data. Under this context, the use of *Evolutionary Computation* makes a promising global search approach for model selection. On the other hand, *ensembles* (combinations of models) have been boosting the performance of several *Machine Learning (ML)* algorithms. In this work, a novel evolutionary technique for *MLP* design is presented, being also used an ensemble based approach. A set of real world classification and regression tasks was used to test this strategy, comparing it with a heuristic model selection, as well as with other *ML* algorithms. The results favour the evolutionary *MLP* ensemble method.

Keywords: Supervised Machine Learning, Multilayer Perceptrons, Evolutionary Algorithms, Ensembles.

1 Introduction

Neural Networks (NNs) are important *Machine Learning (ML)* techniques, denoting a set of connectionist models inspired in the behavior of the human brain. In particular, the *Multilayer Perceptron (MLP)* is a popular architecture, where *neurons* are grouped in *layers* and only *forward connections* exist. This provides a powerful base-learner, capable of nonlinear mappings [1]. When compared to other *ML* methods, *MLPs* are known to behave well in terms of predictive knowledge [2], and there has also been research in terms of explanatory knowledge (e.g. extracting rules from *MLPs*) [3].

However, one of the major issues when applying *MLPs* is the *topology* (i.e. connectivity) design. This is a complex and crucial task, with a strong impact in performance (a small network may provide poor learning capabilities, while a large one will overfit the data). It is common to address this task by trial-and-error procedures (e.g. exploring different number of hidden nodes), in a *blind* search strategy, which only goes through a small set of possible configurations. More elaborated methods have been proposed, such as *pruning* [4] and *constructive* [5] algorithms, although these perform *hill-climbing*, being prone to local minima.

An alternative is offered by *Evolutionary Computation (EC)*, which performs a global multi-point (or *beam*) search, quickly locating areas of high quality, even when the search space is very large and complex. The combination of *EC* and *NN*, often called *Evolutionary Neural Networks (ENNs)*, is a better candidate for the topology design, due to the characteristics of the error surface [6]:

- the number of nodes/connections is unbounded;
- the mapping from the structure to its performance is indirect;
- changes are discrete and can provide discontinuous effects in the *NN* behaviour; and
- similar topologies may present different performances.

In addiction, this approach is biologically more plausible; i.e., living creatures have succesfully adapted to their environments as a result of the interaction of *evolution* and *learning*.

Another emergent *ML* research area is related to the use of *ensembles*, where a set of models are combined to produce an answer, being often more accurate than individual learners [7]. One interesting way to build *NN* ensembles is based on *heterogeneous topologies* [8]. This approach can be easily adapted to *ENNs* with no computacional effort increase, since *ENNs* already use a population of *NNs* with different connectivities.

The present work presents a novel *ENN* to the design of *MLP* topologies, where a direct *MLP* representation (closer to the phenotype) is used. This approach will be

[1]This work was supported by the *FCT* project POSI/ROBO/43904/2002, which is partially funded by FEDER.

tested in classification and regression tasks, using both single and ensemble based models. Finally, results will be compared with a heuristic *NN* selection procedure, as well with other *ML* methods.

2 Materials and Methods

2.1 Data Sets

Eight classification and eight regression data sets were selected from the UCI *ML* repository [9], and its main features are listed in Table 1, namely the number of numeric (**Nu**), binary (**Bi**) and nominal (**No**, i.e. discrete with 3 or more labels) input attributes, as well as the number of examples (**Ex**) and classes (**Cl**). The regression tasks are identified by the symbol $\Re$ in the **Cl** column (last eight rows).

Table 1. A summary of the data sets used.

Task	Inputs			Ex	Cl
	Nu	Bi	No		
Balance	4	0	0	625	3
Bupa	6	0	0	345	2
Car	0	0	6	1728	4
Cmc	5	3	1	1473	3
Dermatology	34	0	0	366	6
Ionosphere	34	0	0	351	2
Sonar	60	0	0	104	2
Yeast	7	1	0	1484	10
Abalone	7	0	1	4177	$\Re$
Auto-mpg	5	0	2	398	$\Re$
Autos	17	3	5	205	$\Re$
Breast-cancer	1	4	4	286	$\Re$
Heart-disease	6	3	4	303	$\Re$
Housing	12	1	0	506	$\Re$
Servo	2	0	2	167	$\Re$
WPBC	32	0	0	194	$\Re$

2.2 Neural Networks

Before feeding the *MLPs*, the data was preprocessed: a *1-of-C* encoding (one binary variable per class) was applied to the nominal attributes and all input values were rescaled within the range $[-1, 1]$. For example, the **safety** attribute from the task **car** was encoded according to: *low* $\rightarrow$ 1 -1 -1, *med* $\rightarrow$ -1 1 -1 and *high*$\rightarrow$ -1 -1 1.

Regarding the outputs, the discrete variables were normalized within the range $[0, 1]$ (using also a *1-of-C* encoding for the nominal attributes). Therefore, the predicted class is given by the nearest class value to the node's output, if one single node is used (binary variable), otherwise the node with the highest output value is considered. On the other hand, regression problems will be modeled by one real-valued output, which directly represents the *dependent* target variable.

The *MLPs* used make use of biases and sigmoid activation functions, with one hidden layer, containing a variable number of nodes. A different approach was followed for the regression tasks, since outputs may lie out of the logistic output range ($[0, 1]$). In this case, the *logistic* function was applied on hidden nodes, while the output ones used shortcut connections and *linear* functions, to scale the range of the outputs (Figure 1). This solution avoids the need of filtering procedures, which may give rise to information loss and has been successfully adopted in other regression applications, such as *Time Series Forecasting* [10].

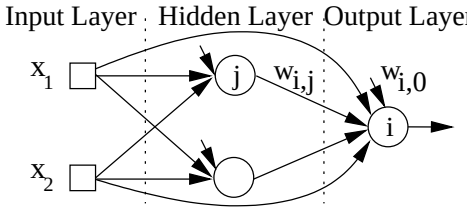

Fig. 1. A $2-2-1$ *MLP* topology with bias and shortcuts.

The initial weights will be randomly set within the range $[-1, 1]$. Then, the *RPROP* algorithm [11] is selected for training, due to its faster convergence and stability, being stopped after a maximum of 200 epochs or when the error slope is approaching zero.

Two distinct accuracy measures were adopted: the *Percentage of Correctly Classified Examples (PCCE)*, used in classification tasks; and the *Normalized Root Mean Squared Error (NRMSE)*, applied in the regression ones. These metrics are given by the equations:

$$\begin{aligned} PCCE &= \frac{\sum_{i=1}^{N} 1\,,\, if(T_i = P_i)}{N} \times 100\,(\%) \\ NRMSE &= \frac{\sqrt{\frac{\sum_{i=1}^{N}(T_i - P_i)^2}{N}}}{\frac{\sum_{i=1}^{N} T_i}{N}} \times 100\,(\%) \end{aligned} \quad (1)$$

where N denotes the number of examples; P_i, T_i the predicted and target values for the i-th example.

An *Heuristic* approach (*HNN*) to model selection is defined by a simple trial-and-error procedure, where fully connected *MLPs*, with a number of hidden nodes ranging from 0 to 20, are trained. Then, the topology with the lowest validation error (computed over non training data) is selected. The trained *MLPs* will also be used to build an *Ensemble* (*HNNE*), where the output is given by the average over all 21 *MLPs*.

2.3 Evolutionary Neural Network

In this work, an *Evolutionary Algorithm* with a *direct* representation is embraced, where the genotype is the whole *MLP*. Each individual of the initial population is set by choosing a random number of hidden nodes (between 0 and 10). Then, each possible connection is set with probability of 50%. New individuals are bred by structural *mutation*, which works by adding or deleting a random number (from 1 to 5) of nodes or connections. The population size was set to 20, being the *selection* done by converting the fitness value (the error computed over a *validation* set) into its ranking, and then applying a roulette wheel scheme, being used a substitution rate of 50%. Finally, the *ENN* is stopped after 20 generations.

This scalable *ENN* is able to search through any kind of *MLP* connectivity, ranging from linear models to complex nonlinear *MLPs*. The *ENN Ensemble (ENNE)* will be built using the best 20 individuals (with lower validation error) obtained during the evolutionary process, being the output computed as the average of the *MLPs*.

3 Results

The *NN/EC* experiments were conducted using a software package developed in *Java* by the authors. The other techniques were computed using *WEKA ML* software (with its default parameters) [12]:

- *J48* – a classification decision tree based on the C4.5 algorithm;
- *M5P* – a regression decision tree (M5 algorithm);
- *IB5* – a *5-Nearest Neigboor*;
- *KStar* – an instance based algorithm; and
- *SVM* – a *Support Vector Machine*.

For each model, 10 runs of a 5-fold cross-validation process [13] (stratified in the classification tasks) were executed. This means that in each of these 50 experiments, 80% of the data is used for learning and 20% for testing. Regarding the *MLP* based approaches, the learning data was divided into training (50% of the original dataset) and validation sets (30%). Tables 2 and 3 show the average errors of the 10 runs for each learning model and classification/regression task. In both tables, the last row averages the global behaviour of each technique.

First, the classification results will be analyzed. The *NN* based approaches (last four columns) are competive when compared with the other *ML* algorithms. The few exceptions are the **dermatology** and **sonar** tasks, where the *SVM* and *KStar* get the best results. As expected, the *ENN* outperforms the *HNN*, with a 1% increase in performance. Furthermore, the ensemble approaches (*HNNE* and *ENNE*) obtain better results when compared with the single based methods (1.4% improvement in both cases). Indeed, the *ENNE* reveals the best overall behaviour, outperforming all other algorithms in 5 of the 8 tasks.

A similar scenario occurs in the regression tasks. In general, the *NN* methods are better than the other *ML* algorithms, although the *M5P* outperforms the *HNN* and *HNNE* approaches. When compared with the *ENN*, the *HNN* is outperformed by a wider difference (2.5%). As before, the ensembles behave better, although the impact is higher with the *ENNE* (1.5% improvement, being the best method in 5 tasks) than with the *HNNE* (0.7% improvement).

4 Conclusions

The surge of bio-inspired techniques, such as *Multilayer Perceptrons (MLPs)* and *Evolutionary Computation (EC)*, has created new exciting possibilities for the field of *Machine Learning (ML)*. Considering ensembles of learning models to improve its accuracy has also been a focus of attention by the research community. In this work, an evolutionary approach to *MLP* topology design was presented, considering single and ensemble combinations, being tested in supervised learning tasks (e.g., classification and regression).

The results obtained confirm than *Evolutionary Neural Network (ENN)* approach outperforms a heuristic trial-and-error *MLP* design procedure (*HNN*), as well as other *ML* algorithms (e.g. *K-Nearest Neigboor*). However, this improvement in performance has the handicap of increasing the computational complexity (the *HNN* requires only a tenth of the *ENN* computational effort).

On the other hand, the use of the *EC* population structure to construct ensembles is a recent research field. Indeed, the proposed *ENN Ensemble (ENNE)*, based on the average of the outputs from the best *EC* individuals (or *MLPs*), is competitive. The *ENNE* has the advantage of presenting the best overall performance while requiring the same computational effort, when compared with the single based *ENN*.

In future work, it is intended to explore similar approaches with different neural architectures (e.g., *Recurrent Neural Networks*). Moreover, more elaborated ensembles should be considered, by designing fitness functions which reward specialization [14].

References

[1] Haykin, S. (1999) Neural Networks - A Compreensive Foundation. 2nd ed. Prentice-Hall

[2] Quinlan, J.R. (1994) Comparing Connectionist and Symbolic Learning Methods. In: Hanson, S. et al. (eds.) Computation Learning: Theory and Natural Learning Systems, MIT Press, pp. 445-456

Table 2. The classification results ($PCCE$ values, in %).

Task	J48	IB5	KStar	SVM	HNN	ENN	HNNE	ENNE
Balance	78.1	87.6	88.3	87.7	94.8	95.8	95.7	**96.5**
Bupa	64.8	60.7	65.9	58.0	68.4	69.0	68.5	**69.7**
Car	91.3	92.3	87.1	93.5	97.4	98.2	98.3	**98.7**
Cmc	51.2	47.2	49.6	48.4	50.6	54.2	52.1	55.9
Dermatology	95.7	96.6	94.5	**97.4**	94.9	95.5	95.9	96.8
Ionosphere	89.4	84.6	84.0	87.9	88.9	89.6	92.3	**92.6**
Sonar	72.5	80.5	**85.2**	76.7	79.9	79.0	80.9	81.2
Yeast	56.0	57.1	53.1	56.6	58.2	59.3	59.9	**60.2**
Mean	74.9	75.8	76.0	75.8	79.1	80.1	80.5	**81.5**

Table 3. The regression results ($NRMSE$ values, in %).

Task	M5P	IB5	KStar	SVM	HNN	ENN	HNNE	ENNE
Abalone	24.3	25.3	24.8	25.0	**23.2**	**23.2**	**23.2**	**23.2**
Auto-mpg	**11.8**	15.1	14.6	13.5	14.2	13.0	12.2	12.2
Autos	**13.3**	21.4	21.6	13.8	14.6	16.1	14.2	14.9
Breast-cancer	40.8	40.8	43.6	44.0	42.6	40.6	47.4	**40.2**
Heart-disease	21.2	**21.0**	25.5	21.5	21.9	21.7	22.2	21.5
Housing	18.4	22.2	18.0	22.6	18.4	17.3	16.6	**16.0**
Servo	50.4	60.4	67.6	70.5	60.8	45.0	49.6	**39.4**
Wpbc	73.2	73.6	98.2	**71.7**	75.4	74.4	80.3	**71.7**
Mean	33.0	35.0	39.2	35.3	33.9	31.4	33.2	**29.9**

[3] Setiono, R. (2003) Techniques for Extracting Classification and Regression Rules from Artificial Neural Networks. In D. Fogel and C. Robinson, (eds.), Computational Intelligence: The Experts Speak, IEEE Press/Wiley, pp 99–114

[4] Thimm, G. and Fiesler, E. (1995) Evaluating pruning methods. In: Proc. of the Int. Symp. on Artificial Neural Networks, pp 20–25

[5] Kwok, T. and Yeung, D. (1997) Constructive algorithms for structure learning in feedforward neural networks for regression problems problems: A survey. IEEE Transactions on Neural Networks, 8(3):630–645

[6] Yao, X. (1999) Evolving Artificial Neural Networks. In: Proc. of the IEEE, 87(9): 1423-1447

[7] Dietterich, T. (1997) Machine Learning Research: Four Current Directions. AI Magazine, 18(4):97–136

[8] Rocha, M., Cortez, P. and Neves, J. Ensembles of Artificial Neural Networks with Heterogeneous Topologies. In: Proc. of the 4th Symposium on Engineering of Intelligent Systems (EIS2004). ICSC Academic Press

[9] Blake, C. and Merz, C. (1998) UCI Repository of Machine Learning Databases, University of California

[10] Cortez, P., Rocha, M. and Neves, J. (2001) Evolving Time Series Forecasting Neural Network Models. In: Proc. of the 3rd Int. Symposium on Adaptive Systems: Evolutionary Computation and Probabilistic Graphical Models (ISAS 2001), pp. 84–91

[11] Riedmiller, M. (1994) Supervised Learning in Multilayer Perceptrons - from Backpropagation to Adaptive Learning Techniques. Computer Standards and Interfaces 16

[12] Witten, I. and Frank, E. (2000) Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann

[13] Kohavi, R. (1995) A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In: Proc. of the Int. Joint Conference on Artificial Intelligence (IJCAI)

[14] Liu, Y., Yao, X. and Higuchi, T. (2000) Evolutionary Ensembles with Negative Correlation Learning. IEEE Transactions on Evolutionary Computation, 4(4):380–387