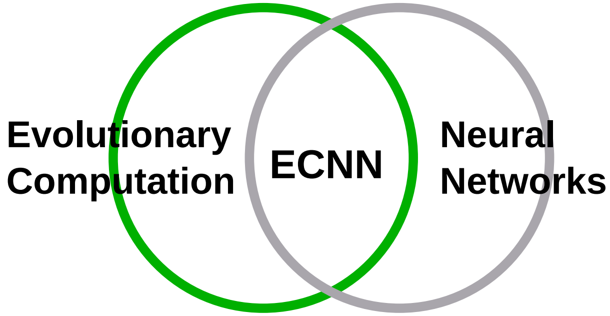
Overview of ECNN Combinations⋆



by **Paulo Cortez** and **Miguel Rocha**

pcortez@dsi.uminho.pt mrocha@di.uminho.pt

(Presentation available at: http://www.dsi.uminho.pt/~pcortez/ecnn.pdf)

Universidade do Minho, Portugal

Evolutionary Computation and Neural Networks Workshop - ECNN, EIS'2004, Madeira

⋆ The authors would like to acknowledge the support from FCT, given under the project POSI/ROBO/43904/2002, which is partially funded by FEDER.

1 – Evolutionary Computation

1.1 – Basic Concepts

- **Evolutionary Computation (EC)** denotes a family of optimization algorithms inspired in natural selection where:
  - a number of potential solutions to a problem makes an evolving **population**;
  - each individual codes a solution in a string (**chromosome**) of symbols (**genes**);
  - a numerical value (**fitness**) is assigned to each individual, which stands for the solution's quality;
  - new solutions are created through the application of **genetic operators** (e.g., **crossover** or **mutation**); and
  - the whole process evolves by stochastic **selection** that favors individuals with higher fitnesses.

1.2 – EC Variants

- In the last decades, several **EC** techniques have been developed;
- The differences (which sometimes are not clear) rely in the representation scheme used and the way new solutions are generated;
- The main variants include:
  - **Genetic Algorithms (GAs)** [Holland, 1975];
  - **Evolutionary Algorithms (GAs)** [Michalewicz, 1996];
  - **Evolutionary Programming (EP)** [Fogel et al., 1966];
  - **Evolutionary Strategies (ES)** [Rechenberg, 1973, Schwefel, 1981]; and
  - **Genetic Programing (GP)** [Cramer, 1985, Koza, 1989].

2 – Neural Networks

- **Neural Networks (NN)** are learning models that mimic the human central nervous system;
- An **NN** is made up by simple processing units (**neurons** or **nodes**) and interneuron synaptic strengths (**connection weights**), where the acquired knowledge is stored;
- **NNs** are appealing due to their capabilities to model complex, nonlinear, multi-dimensional data, even when noise is presented;
- The term **NN** is used to denote a family of models, each with its own **architecture** and **learning behavior**;
- **NN** popular types are:
  - **MultiLayer Perceptrons (MLPs)** [Minsky and Papert, 1969, Bishop, 1995];
  - **Radial-Basis Functions (RBFs)** [Broomhead and Lowe, 1988]; and
  - **Self-Organizing Maps (SOM)** [Kohonen, 1982].

3 – ECNN Combinations

3.1 – Why ECNN?

- The combination of **EC** and **NN**, also known as **Evolutionary Neural Networks** or **Genetic Algorithm and Neural Network (GANN)** systems, offers new possibilities to increase the power of adaptive approaches;
- Motivated by nature, where living creatures managed to survive in hazardous environments due to two main processes: **evolution** and **learning**;
- Key issues when applying **NNs** can be formulated as optimization problems (numerical and combinatorial).

3.2 – How to combine?

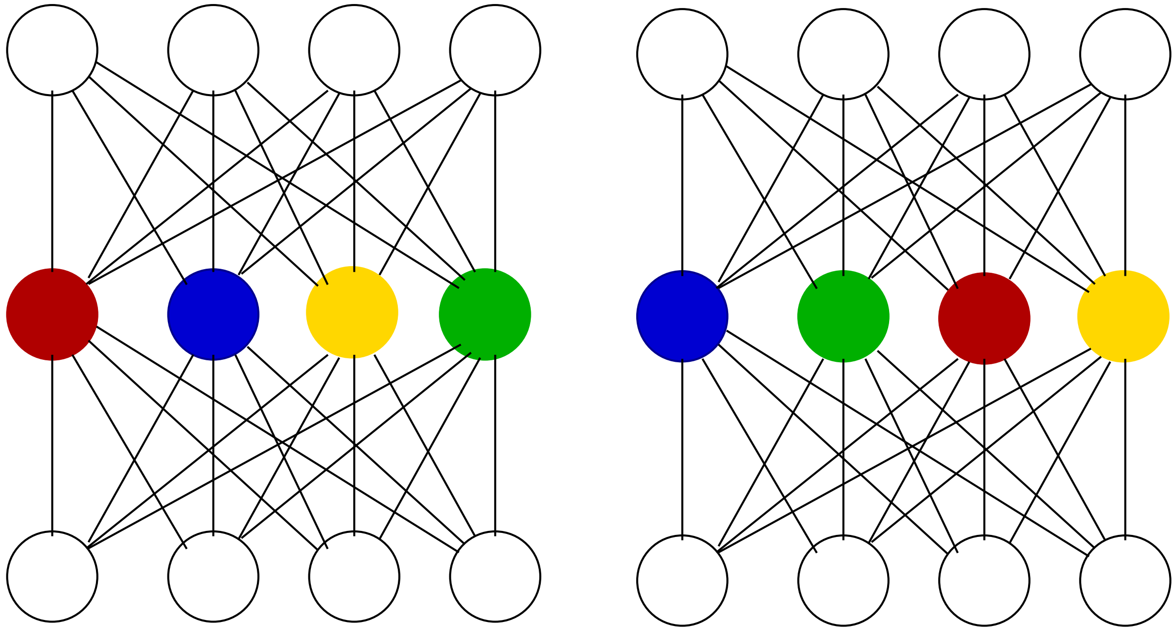
- **EC** to optimize **NN** (most used)
  - **Training**;
  - **Topology design**;
  - Simultaneous optimization of weights and topologies [Yao, 1999];
  - Ensembles of **NN**;
  - **NN** feeding/filtering/**preprocessing**; and
  - Post-processing **NN** outputs (e.g., knowledge extraction).
- **NN** to improve **EC** (nearly unexplored)
  - Individuals placed in lattice positions according to the **SOM** approach [Huhse and Zell, 2000]; and
  - To improve efficiency by simulating complex fitness functions with **NN** [Aguilar-Ruiz et al., 2003].

4 – Training

4.1 – Motivation

- **NN** training can be seen as an numerical optimization task;
- Several **gradient based algorithms** have been proposed (e.g. **Backpropagation**, **RPROP**) for **MLP training** (i.e., to adjust its weights);
- These methods are **local optimization** procedures, being often trapped in local minima of the error function;
- An alternative approach comes from the use of **EC**, since they are global multi-point search methods;
- Since no gradient information is required, **EC** can be used to train **Recurrent NNs** or in **Reinforcement Learning**;
- With a few minor changes, the same algorithm may be applied to train different types of **NNs**.

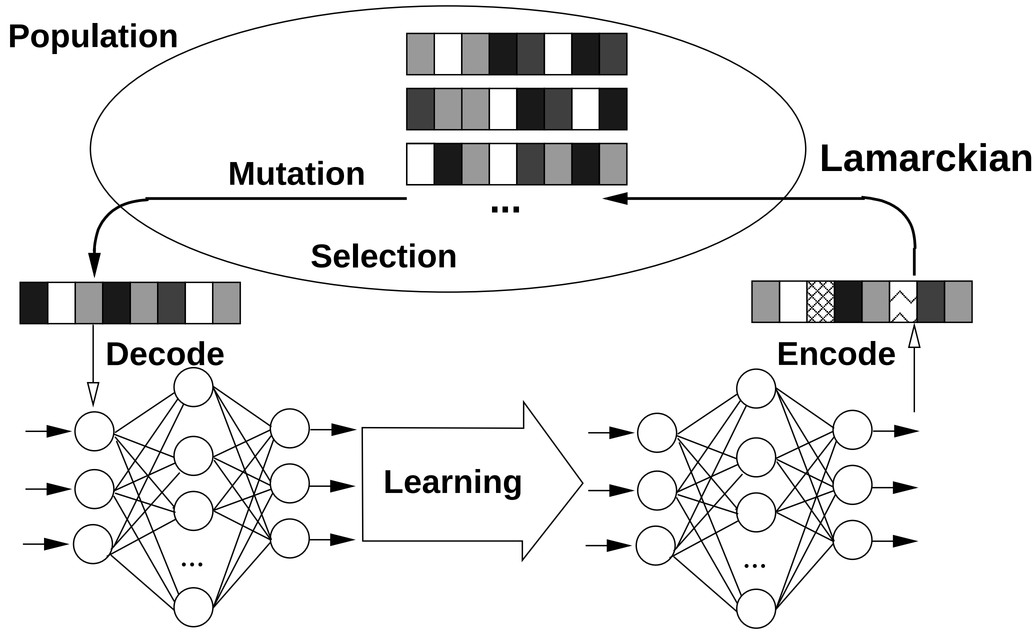
4.2 – Permutation problem



- Difficult to design good crossover operators, due to the **permutation problem** (several genomes may encode the same **NN**);
- Solutions:
  - Use of **ES**, **EP** or mutation operators [Rocha et al., 2003];
  - Try to analyze functionality of hidden nodes.

4.3 – Training Approaches

- First attempts used **binary** representations, making use of **GAs**;
- **Real-valued representations** have been proposed, enlarging the set of genetic operators;
- **Hybrid approaches**, such as **Lamarckian optimization**, where each individual is improved by local training, being the new weights encoded back into the chromosome [Rocha et al., 2003].

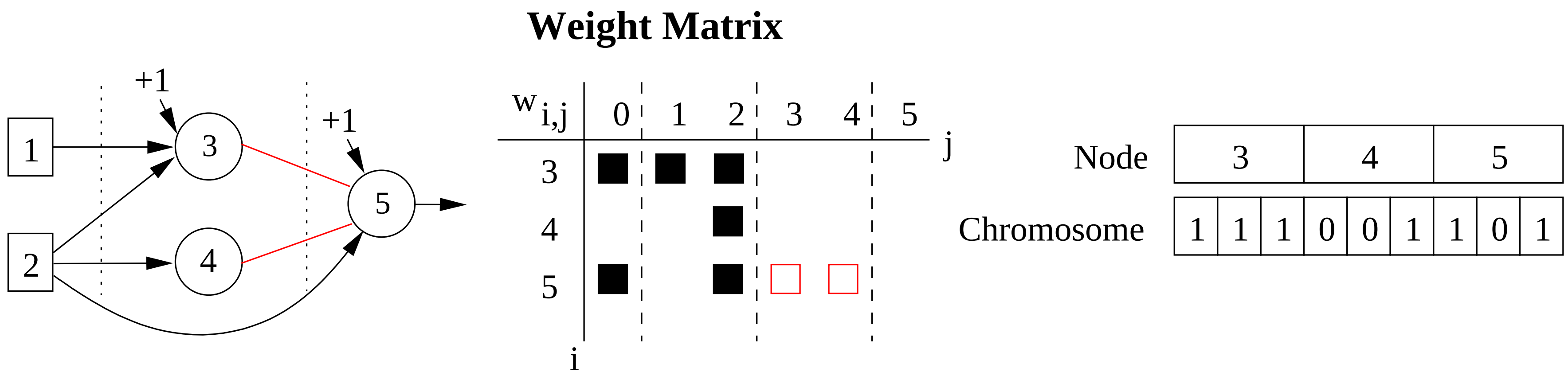


5 – Topology Design

5.1 – Motivation

- The design of optimal **NN** architecture can be formulated as a search problem, presenting a set of characteristics that favor the use of **EC**, such as [Yao, 1999]:
  - **Nondifferentiable** surface, since changes in nodes and/or connections are discrete and have discontinuous effect on the **NN**'s performance;
  - Similar architectures may present different performances (**deceptive** surface) and different networks may present similar performances (**multimodal** surface);
- The critical issues are the **topology representation** and the **NN** evaluation (**fitness function**).

5.2 – Representation



- **Strong** - direct, low-level encoding of connections (most used) or nodes.
  - Used for small networks, prevents network from growing to large;
  - More efficient.
  - E.g. Time Series Forecasting [Cortez et al., 2001].
- **Weak** - indirect, high-level encoding (e.g. construction rules, fractals).
  - not every structure is probable, favors regular networks;
  - better scalability and biological plausibility;
  - E.g. **Cellular Encoding** [Gruau and Whitley, 1993].

5.3 – Fitness Function

- Simple metrics: training error, adding a penalty due to training time, ...;
- Yet, there are two main issues:
  - **Generalization** - How to avoid overfitting?
    - Estimate the error over a **validation set** (not used in training), by using **hold-out**, **K-fold** of **bootstrapping**;
    - Penalize complexity: **weight decay** or **BIC** criterion [Cortez et al., 2001]
  - **Noisy Fitness** - Due to the random initialization of weights.
    - Use of average estimate of **several runs**, although this increases the computational effort;
    - **Simultaneous topology/weight evolution** may alleviate this drawback, although it is more sensitive to overfitting.

6 – Preprocessing

- Feature Subset Selection: selecting a subset of features from a larger set of attributes;
- Search space can be large and other techniques such as **PCA** or **Forward Selection** may fail;
- One easy representation is binary coding, one bit per feature;
- Ensembles can be defined by defining populations of **NNs** with different set of input features [Guerra-Salcedo and Whitley, 1999].

7 – ECNN Workshop

7.1 – Evolutionary Computation

- “Artificial Life Optimization over Complex Networks” - M. Lucchetti, M. Annunziato, R. Huerta and L. Tsimring
- “An Evolutionary Algorithm for Manipulator Path Planning” - R. Corsepius
- “Evolving Strategy for Game Playing” - J. Hynek

7.2 – ECNN combinations

- “Advanced Evolutionary Design of Generalized Recurrent Neural Networks” - A. Dobnikar and S. Vavpotic
- “Ensembles of Artificial Neural Networks with Heterogeneous Topologies” - M. Rocha, P. Cortez and J. Neves
- “A Lamarckian Model Combining Levenberg-Maquardt Algorithm and a Genetic Algorithm” - P. Pires and P. Castro
- “Evolving Modular Neural Networks to Solve Challenging Control Problems” - S. Doncieux and J. Meyer
- “Hierarchical Evolutionary Algorithm in the Rule Extraction from Neural Network” - U. Markowska-Kaczmar and R. Zagorski
- “Genetic Algorithms with Fitness & diversity -Guided Adaptive Operating Probabilities and Analysis of its Convergence” - L. Meiyi, C. Zixing and S. Guoyun

8 – Open Discussion: "The Future of ECNN Combinations"

- **NN** learning in changing environments;
- Reinforcement learning (e.g. RoboCup simulation league);
- Recurrent **NNs**;
- Ensembles;
- ...

References

[Aguilar-Ruiz et al., 2003] Aguilar-Ruiz, J., Mateos, D., and Rodriguez, D. (2003). Evolutionary Neuroestimation of Fitness Functions. In Pires, F. and Abreu, S., editors, *Progress in Artificial Intelligence, EPIA 2003 Proceedings, LNAI 2902*, pages 74–83, Beja, Portugal. Springer.

[Bishop, 1995] Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

[Broomhead and Lowe, 1988] Broomhead, D. and Lowe, D. (1988). Multivariable functional interpolation and adaptative networks. *Complex Systems*, 2:321–355.

[Cortez et al., 2001] Cortez, P., Rocha, M., and Neves, J. (2001). Evolving Time Series Forecasting Neural Network Models. In *Proceedings of the Thirtd International Symposium on Adaptive Systems: Evolutionary Computation and Probabilistic Graphical Models (ISAS 2001)*, pages 84–91, Havana, Cuba.

[Cramer, 1985] Cramer, N. (1985). A representation for the adaptive generation of simple sequential programs. In Grefenstette, J. J., editor, *International Conference on Genetic Algorithms and Applications*, pages 183–187.

[Fogel et al., 1966] Fogel, L., Owens, A., and Walsh, M. (1966). *Artificial Intelligence Through Simulated Evolution*. John Wiley, New York.

[Gruau and Whitley, 1993] Gruau, F. and Whitley, D. (MIT Press, 1993). Adding learning to the cellular development of neural networks: Evolution and the baldwin effect. *Evolutionary Computation*, 3(1):213–233. MIT Press.

[Guerra-Salcedo and Whitley, 1999] Guerra-Salcedo, C. and Whitley, D. (1999). Feature Selection Mechanisms for Ensemble Creation: a Genetic Search Perspective. In *Proceedings of GECCO-99 Workshop on Data Mining with Evolutionary Algorithms: Research Directions*.

[Holland, 1975] Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. PhD thesis, University of Michigan, Ann Arbor.

[Huhse and Zell, 2000] Huhse, J. and Zell, A. (2000). Evolutionary strategy with neighborhood attraction. In Bothe, H. and Rojas, R., editors, *Neural Computation 2000*, pages 363–369. ICSC Academic Press.

[Kohonen, 1982] Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69.

[Koza, 1989] Koza, J. (1989). Hierarchical genetic algorithms operating on populations of computer programs. In Sridharan, N. S., editor, *Proceedings of Eleventh International Joint Conference on Artificial Intelligence IJCAI-89*, volume 1, pages 768–774. Morgan Kaufmann.

[Michalewicz, 1996] Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, USA, third edition.

[Minsky and Papert, 1969] Minsky, M. and Papert, S. (1969). *Perceptrons*. MIT Press, Cambridge, MA.

[Rechenberg, 1973] Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Holzboog Verlag, Stuttgart.

[Rocha et al., 2003] Rocha, M., Cortez, P., and Neves, J. (2003). Evolutionary Neural Network Learning. In Pires, F. and Abreu, S., editors, *Progress in Artificial Intelligence, EPIA 2003 Proceedings, LNAI 2902*, pages 24–28, Beja, Portugal. Springer.

[Schwefel, 1981] Schwefel, H.-P. (1981). *Numerical Optimization of Computer Models*. Wiley.

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