Optimal Genetic Representation of Complete Strictly-Layered Feedforward Neural Networks

Spyros Raptis, Spyros Tzafestas, and Hermione Karagianni

Intelligent Robotics and Automation Laboratory
Division of Signals, Control and Robotics
Department of Electrical and Electronic Engineering
National Technical University of Athens
Zographou Campus, 15773, Athens, GREECE
tzafesta@softlab.ece.ntua.gr

Abstract. The automatic evolution of neural networks is both an attractive and a rewarding task. The connectivity matrix is the most common way of directly encoding a neural network for the purpose of genetic optimization. However, this representation presents several disadvantages mostly stemming from its inherent redundancy and its lack of robustness. We propose a novel representation scheme for encoding complete strictly-layered feedforward neural networks and prove that it is optimal in the sense that it utilizes the minimum possible number of bits. We argue that this scheme has a number of important advantages over the direct encoding of the connectivity matrix. It does not suffer from the curse of dimensionality, it allows only legal networks to be represented which relieves the genetic algorithm from a number of checking and rejections, and the mapping from the genotypes to phenotypes is one-to-one. Additionally, the resulting networks have a simpler structure assuring an easier learning phase.

1 Introduction

As stochastic search processes, genetic algorithms (GA’s) provide no estimation on the time required to locate an adequate solution to a given problem. They use very limited (if any) a priori information on the specific problem they are addressing. In this sense, they are inadequate for on-line execution in time-sensitive problems.

On the contrary, GA’s have very often been used as a high-level tool to evolve other systems that are more suited for on-line performance. Indicative examples include the definition of the input partitions or other parameters of a fuzzy system, the determination of appropriate values for the parameters of another GA, the design of an appropriate set of detectors for pattern recognition, etc. The evolutionary design of neural networks by genetic means may very well be placed in this context.

In spite of the intense research in the area of neural networks, globally applicable rules for their design are still missing and their development is still based on heuristics and rules of thumb. Design parameters include the network topology, the determination of its connectivity pattern, the selection of the neuron activation functions, the training algorithm used to calculate the weights of the connections, etc.
The design of a neural network by genetic means restates the problem in the context of an optimization process. Automatically evolving a network that is “optimal” (in the sense of a certain criterion) is particularly attractive since it offers a general methodology for the design of a “well-behaving” neural system.

The work in the field of evolving neural networks is quite extensive and certain surveys are also available (e.g. [1]). The various approaches may be roughly divided into categories based on different characteristics of the process.

Based on the genotypic representation used we can identify (a) direct encoding schemes (e.g. [2], [3], [4]) where the GA uses a simple and easily decodable representation of the neural network such as its connectivity matrix and (b) indirect encoding schemes (e.g. [5], [6], [7]) such as production rules, grammars, etc. where decoding is not so trivial. Based on the type of network being evolved, we may identify approaches for the design of (a) feedforward networks or (b) recurrent networks. The selection of the required class of neural networks is, clearly, problem-dependent. A last classification can be identified that is based on the type of network parameters being evolved. We can identify methods that (a) optimize only the network topology, (b) optimize the values of the synaptic weights and other parameters of a fixed network, or (c) optimize both the topology and the parameter values of the network.

In the remainder of the paper we will only consider the case of complete strictly-layered feedforward neural networks. A neural network is said to be layered if its hidden neurons are organized in layers. No connection can appear among neurons of the same layer. A network is strictly-layered if it is layered and the neurons of a layer can only accept inputs from neurons of the immediately preceding layer. A network is complete if every neuron of a layer is connected to all the neuron in the previous layer.

2 Connectivity Matrices

The connectivity matrix is the most commonly used representation of a neural network. It is a square binary matrix, $C$, of dimension equal to the total number of neurons in the network (including input, output and hidden neurons). Each element of the matrix, say $C(i,j)$, indicates the presence or absence of a connection from neuron $i$ to neuron $j$. For a feedforward network the connectivity matrix has an upper-triangular form.

In the connectivity matrix of a feedforward neural network, the number of elements that can be non-zero is given by:

$$n_{\text{gene}} = (n+h)(m+h) - \frac{1}{2}h(h+1)$$ (1)

where $n$ is the number of network inputs, $m$ is the number of network outputs, and $h$ is the number of hidden neurons.

The representation of neural connectivity patterns through their respective connectivity matrix contains redundant information. Thus, while the two networks displayed in Fig. 1 are functionally equivalent, their respective connectivity matrices differ. Of course, by appropriate renumbering (i.e. exchanging columns and rows) the