Chapter 36
Database Analysis with ANNs by means of Graph Evolution

Daniel Rivero
University of A Coruña, Spain

Juan R. Rabuñal
University of A Coruña, Spain

Julián Dorado
University of A Coruña, Spain

Alejandro Pazos
University of A Coruña, Spain

ABSTRACT

Traditionally, the development of Artificial Neural Networks (ANNs) is a slow process guided by the expert knowledge. This expert usually has to test several architectures until he finds one suitable for solving a specific problem. This makes the development of ANNs a slow process in which the expert has to do much effort. This chapter describes a new method for the development of Artificial Neural Networks, so it becomes completely automated. Since ANNs are complex structures with very high connectivity, traditional algorithms are not suitable to represent them. For this reason, in this work graphs with high connectivity that represent ANNs are evolved. In order to measure the performance of the system and to compare the results with other ANN development methods by means of Evolutionary Computation (EC) techniques, several tests were performed with problems based on some of the most used test databases in Data Mining. These comparisons show that the system achieves good results that are not only comparable to those of the already existing techniques but, in most cases, improve them.

INTRODUCTION

One of the most widely used techniques in Data Mining are Artificial Neural Networks (ANNs). They are very used in Data Mining because of their ability to solve problems related to different aspects such as classification, clustering or regression (Haykin, 1999). These systems are, due to their interesting characteristics, powerful techniques used by the researchers in different environments (Rabuñal, 2005).

ANNs are systems that try to emulate the behaviour of the nervous system. Therefore, as the biological nervous system is composed by a high number of neurons connected between them, an ANN is also composed by several artificial neurons with a high connectivity. Even each artificial neuron performs a very easy mathematical function with the inputs that it received from other neurons or from the environment, the result of a

DOI: 10.4018/978-1-4666-2455-9.ch036

high number of neurons connected between them can be a very complicated function. In fact, ANNs are universal approximators: any mathematical function can be reproduced by means of using ANNs.

However, the development of ANNs is not an easy task. This process can be divided into two parts: architecture development and training and validation. The architecture development determines not only the number of neurons of the ANN, but also their connectivity. The training will determine the connection weights for such architecture. This second step is automatically performed by different training algorithms. However, the first step, architecture development, given that the architecture of the network depends on the problem to be solved, is usually performed by the use of a manual process, meaning that the expert has to test different architectures until he finds the one that returns the best results.

Therefore, if the expert wants to develop an ANN to solve a specific problem, he will have to design several networks and train several times each network (due to the stochastic nature of the training algorithms) in order to determine which one of these architectures is the best one. This is a slow process due to the fact that architecture determination is a manual process, although techniques for relatively automatic creation of ANNs have been recently developed.

This work presents a new technique that automatically develops ANNs, so that no human participation will be needed. This technique is a modification of the Genetic Programming (GP) algorithm in order to allow it to evolve graphs instead of trees.

BACKGROUND

Genetic Programming

GP (Koza, 92) is based on the evolution of a given population. Its working is similar to a GA. In this population, every individual represents a solution for a problem that is intended to be solved. The evolution is achieved by means of the selection of the best individuals – although the worst ones have also a little chance of being selected – and their mutual combination for creating new solutions. This process is developed using selection, crossover and mutation operators. After several generations, the population is expected to contain some good solutions for the problem.

The GP encoding for the solutions is tree-shaped, so the user must specify which are the terminals (leaves of the tree) and the functions (nodes that have children) for being used by the evolutionary algorithm in order to build complex expressions. These can be mathematical (including, for instance, arithmetical or trigonometric operators), logical (with Boolean or relational operators, among others) or other type of even more complex expressions.

The wide application of GP to various environments and its consequent success are due to its capability for being adapted to numerous different problems. Although the main and more direct application is the generation of mathematical expressions (Rivero, 2005), GP has been also used in others fields such as filter design (Rabuñal, 2003), knowledge extraction (Rabuñal, 2004), image processing (Rivero, 2004), etc.

Using Evolutionary Computation Techniques for the Development of ANNs

The development of ANNs has been widely treated with very different techniques in AI. The world of evolutionary algorithms is not an exception, and proof of it is the large amount of works that have been published is this aspect using several techniques (Nolfi, 2002; Cantú-Paz, 2005). These techniques follow the general strategy of an evolutionary algorithm: an initial population with different types of genotypes encoding also different parameters – commonly, the connection