Hybrid AI and Machine Learning Systems

Logan R. Kearsley
Dec. 20, 2006
Computer Systems Lab, 2006-2007

December 20, 2006

Abstract

Many different Artificial Intelligence and Machine Learning architectures and algorithms have been developed, each with their own strengths and weaknesses that make them particularly suited to certain classes of problems. Two approaches to artificial intelligence are dealt with here: subsumption architecture, and neural networks. S.A.'s are good at building up complex behaviors from sets of simpler ones, when a problem can be broken down into independent peices. Neural networks are good at memorizing associations and making inferences about new data based on stored memories, but can take impractically long to train when the datasets are large. Performance on various tasks can be improved by combining the two approaches to take advantage of the strengths of both.

Keywords: AI, machine learning. neural networks, subsumption architecture

1 Introduction

The purpose of this project is to design a system that combines the capabilities of neural networks and subsumption architectures to produce a more flexible and versatile hybrid system. In this paper I describe a set of basic library functions for the manipulation of subsumption architectures and neural networks, and their use in building a hybrid system, which is then evaluated against a test problem.

This project serves as a starting point for further investigations of the improvements that can be made by in AI models by using multiple approaches that cover each other's weak points. The results could be applied to almost any problem that requires high adaptability and dealing with 'fuzzy' input, such as character or speech recognition with multiple users or real-world robot navigation.

2 Background

Most AI and Machine Learning research to this point has consisted of pursuing separate single methods, either to maximize utility for a single problem type, or to duplicate biological models. To my knowledge, little research has been done in hybrid systems that combine the best aspects of multiple other methods to produce a highly versatile AI/ML system without necessarily trying to model biological nervous system functions. A similar research project was carried out by Julian Togelius at the University of Sussex, focusing on the control of a mobile robot, in 2003. Hybrid neural networks have been studied which incorporate symbolic representations into neural network programs for, among other things, speed and ease of control while retaining the robustness and generalization capabilities of pure neural nets since the late 80s.

The subsumption architecture model was first used in 1984 and invented by Alexandre Parodi. Subsumption architectures make use of multiple behavior layers which process inputs and produce outputs independently, with some layers capable of temporarily overiding or subsuming the actions of others. This allows for low-level, high-priority reflex layers to deal with immediate problems, while higher layers control the mid-term and global goals of the agent. Brooks modified the original hierarchical layer model to produce a competitive architecture in which each layer competes for priority with a master scheduler, rather than having its priority fixed.

3 Neural Network Development

Requirements: The neural network should be able to memorize any arbitrary set of input/output associations, where inputs and outputs are in the form of lists of numbers between 0 and 1.

Overview: The neural network code is written in C, using the gcc compiler. Functions are available to generate, save, and load from a file networks of arbitrary dimensions, to query a given network with an input vector for the associated output vector, and to teach new associations to the net in cycles.

Limitations: The primary limitations on this project have been time to do the necessary debugging and testing.

Iterative Evaluation Plan: I have used the Spiral Lifecycle Model to develop this project. Each new component was exhaustively tested before adding on the next one, which often revealed previously inconsequential flaws in that required re-visiting the development of older components.

Research Theory and Design criteria: The central algorithm for this project is the back-propagation neural network learning algorithm. Back-propagation is a method of sequentially updating the weights in each layer of a multi-layer, simply connected network based on the errors in the following layer. At the output layer, errors are calculated for each neuron based on the difference between the actual output and the target, and weights are updated accordingly. These errors are then distributed among each of the neurons in the next layer up, weighted by the output of each of those neurons, and the process is repeated to propagate the errors and corrections backwards through the entire network.

Testing and analysis: The primary test for this program consists of providing the network with a list of input/output associations and run training cycles until the network either converges on the correct solution, or it becomes clear that it will not (for example, by converging on an incorrect solution or beginning to diverge). We can thus test the efficiency of the program by examining how many epochs are required to fully train the network. In the event that the network did not converge, small numbers of training epochs were run, with verbose output of the input and output values of all layers of the network in the process in order to identify where and why incorrect adjustments are being made.