8701132

Anderson, Charles William

LEARNING AND PROBLEM SOLVING WITH MULTILAYER CONNECTIONIST SYSTEMS

University of Massachusetts Ph.D. 1986

University Microfilms International 300 N. Zeeb Road, Ann Arbor, MI 48106

LEARNING AND PROBLEM SOLVING
WITH MULTILAYER CONNECTIONIST SYSTEMS

A Dissertation Presented

By

CHARLES WILLIAM ANDERSON

Submitted to the Graduate School of the
University of Massachusetts in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 1986

Department of Computer and Information Science

This research was supported by the Air Force Office of Scientific Research and the Avionics Laboratory (Air Force Wright Aeronautical Laboratories) through contracts F33615-77-C-1191, F33615-80-C-1088, and F33615-83-C-1078.

LEARNING AND PROBLEM SOLVING
WITH MULTILAYER CONNECTIONIST SYSTEMS

A Dissertation Presented

By

CHARLES WILLIAM ANDERSON

Approved as to style and content by:

Andrew G. Barto

Andrew G. Barto, Chairperson of Committee

Paul E. Utgoff

Paul E. Utgoff, Member

M. A. Arbib

Michael A. Arbib, Member

William Kilmer

William L. Kilmer, Outside Member

Conrad A. Wogrin

Conrad A. Wogrin, Department Chair
Computer and Information Science

DEDICATED TO

STACEY AND JOSEPH

ACKNOWLEDGEMENTS

Research in areas where there are many possible paths to follow requires a keen eye for crucial issues. The study of learning systems is such an area. Through the years of working with Andy Barto and Rich Sutton, I have observed many instances of "fluff cutting" and the exposure of basic issues. I thank both Andy and Rich for the insights that have rubbed off on me. I also thank Andy for opening up an infinite world of perspectives on learning, ranging from engineering principles to neural processing theories. I thank Rich for showing me the most important step in doing "science"—simplify your questions by isolating the issues.

Several people contributed to the readability of this dissertation. Andy spent much time carefully reading several drafts. Through his efforts the clarity is much improved. I thank Paul Utgoff, Michael Arbib, and Bill Kilmer for reading drafts of this dissertation and providing valuable criticisms. Paul provided a non-connectionist perspective that widened my view considerably. He never hesitated to work out differences in terms and methodologies that have been developed through research with connectionist vs. symbolic representations. I thank Stephen Judd, Brian Pinette, Jonathan Bachrach, and Robbie Jacobs for commenting on an early draft and for many interesting discussions.

I thank Andy, Rich, Harry Klopf, Michael Arbib, Nico Spinelli, Bill Kilmer, and the AFOSR for starting and maintaining the research project that supported the work reported in this dissertation. I thank Susan Parker for the skill with which she administered the project. And I thank the COINS Department at UMass and the RCF Staff for the maintenance of the research computing environment. Much of the computer graphics software used to generate figures of

this dissertation is based on graphics tools provided by Rich Sutton and Andy Cromarty.

Most importantly, I thank Stacey and Joseph for always being there to lift my spirits while I pursued distant milestones and to share my excitement upon reaching them. Their faith and confidence helped me maintain a proper perspective.

ABSTRACT

LEARNING AND PROBLEM SOLVING WITH MULTILAYER CONNECTIONIST SYSTEMS

September 1986

Charles William Anderson

B.S., University of Nebraska

M.S., University of Massachusetts

Ph.D., University of Massachusetts

Directed by: Professor Andrew G. Barto

The difficulties of learning in multilayered networks of computational units has limited the use of connectionist systems in complex domains. This dissertation elucidates the issues of learning in a network's *hidden* units, and reviews methods for addressing these issues that have been developed through the years. Issues of learning in hidden units are shown to be analogous to learning issues for multilayer systems employing symbolic representations.

Comparisons of a number of algorithms for learning in hidden units are made by applying them in a consistent manner to several tasks. Recently developed algorithms, including Rumelhart, et al.'s, error back-propagation algorithm and Barto, et al.'s, reinforcement-learning algorithms, learn the solutions to the tasks much more successfully than methods of the past. A novel algorithm is examined

that combines aspects of reinforcement learning and a *data-directed* search for useful weights, and is shown to out perform reinforcement-learning algorithms.

A connectionist framework for the learning of strategies is described which combines the error back-propagation algorithm for learning in hidden units with Sutton's AHC algorithm to learn evaluation functions and with a reinforcement-learning algorithm to learn search heuristics. The generality of this hybrid system is demonstrated through successful applications to a numerical, pole-balancing task and to the Tower of Hanoi puzzle. Features developed by the hidden units in solving these tasks are analyzed. Comparisons with other approaches to each task are made.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Connectionist systems embody a framework for decision-making based on an active form of knowledge representation. They are composed of simple computational units interconnected by pathways that transmit numerically-valued signals rather than complicated symbolic messages. The connectionist framework stems both from psychological theories of how the brain represents associations among concepts and from the modeling of neural networks. Current methods and applications of connectionist systems constitute a paradigm at a level between symbolic and neural representations. Although connectionist and symbolic representations are not on the same level, they are subject to analogous problems concerning the development of a representation. ***Representation development*** is the process whereby modifications are made to a representation by the addition, removal, or alteration of the representation's components, whether these components are symbolic terms or numerically-valued features.

Interest in connectionist systems has grown in recent years for several reasons. The inherent parallelism of connectionist systems can result in fast decision-making. Also, the use of connectionist systems as models of cognitive processes has met with some success. For example, Rumelhart and McClelland's (1986) connectionist system models the verb-tense learning behavior of children, a behavior often presented as an argument for the symbolic representation of explicit rules. A third reason for the growing interest in connectionist systems is the recent progress towards a solution to the problem of learning in multilayer connectionist systems, overcoming an obstacle that has been a major criticism of the

connectionist paradigm supported by Minsky and Papert's (1969) analysis of the limitations of the perceptron (Rosenblatt, 1962).

The issues of learning in connectionist systems, however, are far from resolved. The work reported in this dissertation addresses three pressing learning issues:

1. relationships among various approaches to learning in multilayer connectionist systems,
2. the *structural* credit-assignment problem (defined below), and
3. learning in cases in which the desired output of the system is unknown.

The relationships among learning algorithms for multilayer connectionist systems are elucidated by reviewing them within a framework based on a categorization of methods for structurally assigning credit. Numerical learning algorithms not originally presented as connectionist learning algorithms are also discussed within this framework. Such a consistent review is much needed in a field where researchers come from diverse backgrounds, as is the case for connectionist learning research.

Some of the better-known learning algorithms for multilayer connectionist systems are studied by applying them to the multiplexer learning task (described in Chapters II and IV). In comparing the performances of different algorithms, consideration was given to details such as consistency in the training procedure, optimization of the algorithms' parameters, measures both for the performance level during learning and at the conclusion of a learning run, and statistical confidence intervals for all data. Such careful comparisons rarely appear in the literature but are necessary for drawing significant conclusions.

The third issue listed above is discussed in terms of reinforcement-learning methods, as described later in this chapter. To date, research with reinforcement-learning methods has focused on single-layer learning systems (Barto and Anandan, 1985; Barto, Sutton, and Anderson, 1983; Barto, Sutton, and Brouwer, 1981;

Barto and Sutton, 1981), though their potential use for learning in multilayer connectionist systems has been demonstrated in several small examples (Anderson, 1982; Barto, 1985; Barto, Anandan, and Anderson, 1986; Barto and Anderson, 1985; Barto, Anderson, and Sutton, 1982). In this thesis, reinforcement-learning methods are combined with a learning algorithm for multilayer systems to develop an example of a multilayer connectionist system for the learning of problem-solving strategies. This system is demonstrated on a pole-balancing control task and on a Tower of Hanoi puzzle.

Connectionist Systems

Connectionist systems generally consist of a collection of computational units, sometimes described as neuron-like in their input-output behavior. Each unit receives a number of input signals, or *input components*, whose numerical values constitute the unit's *input vector*, and the unit applies an *output function* to its input to generate output values. *Networks* of units are constructed by connecting the output of some units to the input of other, or the same, units. A network is said to interact with an *environment* by receiving a vector of numerical values from the environment and producing an output vector that acts upon the environment. Thus a unit's input components can originate either from the network's environment or from the output of another unit; a unit's output can be passed on to another unit or it can become a component of the network's output.

A unit's output function is parameterized by a vector of numerical weights, one weight for every input component. For a given network structure, it is the values of these weights that determines the input-output behavior of the network. A *learning algorithm*[1] for a connectionist system is a method for updating the

[1] "Algorithm" is not used here in the strictest sense of Knuth's (1973) definition: finiteness is not assumed. Knuth suggests that such procedures be called "computational methods", but for brevity we will use "algorithm." Also, by using the label "learning algorithm" we do not imply the existence of a proof of convergence to a desirable outcome.

values of the system's weights based on the performance of the system. For reviews of current connectionist research see Feldman and Ballard (1982), Hinton and Anderson (1981), McClelland and Rumelhart (1986), and Rumelhart and McClelland (1986).

The output functions performed by the units are usually one of a small set of functions. The most common function is the linear threshold function used by the pioneers of adaptive networks (e.g., Farley and Clark, 1954; McCulloch and Pitts, 1943; Rosenblatt, 1962; Widrow, 1962). A unit implementing this function is sketched in Figure 1, where $x_1, x_2, \ldots, x_n$ are the components of the unit's n-

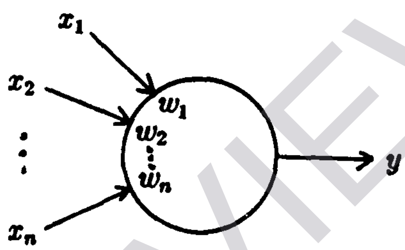

Figure 1: A Linear Threshold Unit

dimensional input vector, $w_1, w_2, \ldots, w_n$ are the unit's weights, and y, the unit's output, is defined as

$$y = \begin{cases} 1, & \text{if } \sum_{i=1}^{n} x_i w_i > 0; \\ 0, & \text{otherwise.} \end{cases}$$

The threshold of this unit is zero, but it can be considered to be parameterized if one of the input components is constant (so that the value of the corresponding weight becomes the negative of the threshold[2]). Such a unit can discriminate (i.e., produce different output values for) sets of input patterns that are linearly-separable (can be separated by an $n-1$ dimensional surface). For example, a unit receiving two input components discriminates input vectors by a line, as shown

[2]Using a constant input is a standard technique for implementing a variable threshold. See Nilsson (1965) for further details.