28/1/88 Full Class Tutorial K.R.II

A Brief Overview of the Study of Neural Computation
Jay Buckingham
January 1988

1. Motivation

Animals which have nervous systems as their behaviour control systems are quite successful at surviving in a changing environment. To be fair, many other simpler biological organisms also do well at that, but those animals with nervous systems exhibit many behaviours which we find so interesting that we try to emulate them in the systems we build. Such animals can move around in their environment, find food, evade predators, etc. And they do these things in a way that is both robust and adaptive. We have not been nearly so successful in building systems which perform these 'survival' functions.

So we might ask what do nervous systems do and how do they do it? Further, how can we characterize their behaviour? What are they good at and what are they poor at? What tasks can they not do at all? And in what ways do they make mistakes?

These questions are topics of active research. In a very hand-waving gesture we can say that nervous systems are good at pattern recognition, sensory-motor control, and learning, while they have a hard time with complex symbolic tasks like doing integral calculus problems. That is, they have evolved for survival in the real world.

Two commonly expressed aims of artificial intelligence are:

- To study the nature of intelligence.
- To build systems which exhibit intelligence in their behaviour.

The study of the behaviour of nervous systems serves both of these. Since these systems generate behaviours that are robust and adaptive enough to enable animals to survive in a changing world their study serves the first aim. Recall that to date we have not been able to build systems which can do that, so if we can understand how biological nervous systems do it we may learn some principles which will enable us to build artificial systems which exhibit some of those properties (second aim).

2. Background

In this section I want to show that people have been working on this for a long time and note some of the major players. Though people involved in this area have come from many disciplines you can roughly divide them into those primarily interested in studying the nervous system and those interested in computation. Often ideas developed by neuroscientists have been taken up by people working on computation.

2.1. Early Work

2.1.1. Cajal

Cajal was an anatomist working in the late nineteenth century on the morphology of nerve cells. He produced many drawings of various sorts of nerve cells and brain structures.

2.1.2. Sherrington

Sherrington was a physiologist working in the same period as Cajal. He studied the patterns of connectivity in brain structures and coined the term *synapse* for the site of contact between nerve cells.

2.1.3. The Basic Model

From the work of these and other neuroscientists there emerged a model of the nerve cell or neuron which looks like a tree. It gathers input from other neurons or the world along its 'roots', called dendrites. The

input is collected at the cell body, or soma. Then the output signal of the neuron is carried away from the soma along the 'trunk', called the axon. The axon, which may split into subbranches, terminates in buds which lie close to the dendrites of other neurons.

This structure - an axon terminal bud apposed to a dendrite - is the synapse. The axon and dendrite are separated by a small gap; signals flowing along the axon do not jump across and induce a signal in the dendrite, rather they cause chemical messengers known as neurotransmitters to be released from the axon bud which flow across the gap to the dendrite. These transmitters can either excite or inhibit the activity of the post-synaptic cell.

To be sure, there is much more detail which can be added to this basic model in light of more recent research, but it is still useful.

2.2. Hebb - The Basis of Learning

Hebb (1949) postulated that the efficacy of a synapse between two neurons would increase if there were simultaneous activity in both of them. That is, the weight of the connection between the two neurons increases. It is widely believed that this is the basis of learning.

2.3. Neurally Inspired Computation

2.3.1. Turing

He was interested in building an 'electronic brain', but was limited by the technology of the time.

2.3.2. von Neumann

Though his name gets associated with register based architectures - the von Neumann Machine - he explored a number of computational models, especially cellular automata. In cellular automata, each 'cell' can have take one of a finite number of states. The state of a cell at time $t+1$ is a function of its state and the state of its neighbors at time t. Conway's Life program is a popular example. Von Neumann showed that cellular automata are very powerful computationally, but that it is difficult to characterize their behaviour. That is, it is hard to program them.

2.3.3. Rosenblatt

He is famous for his work on perceptrons - a neural network abstraction which can classify input patterns which are linearly separable into two groups. They typically have a single layer of cells or nodes which get input from the environment and an output unit which gets input from these. Each of these nodes can have state '0' or '1'. Patterns from the environment are presented and the output node is supposed to take state '1' if the pattern is in one class and state '0' if it is in the other.

Part of the appeal of perceptrons was that there was a learning algorithm (for a limited class of problems) called *perceptron convergence*, independently discovered by Rosenblatt and Nilsson among others. Given a problem they could solve, the Perceptron Convergence Theorem guaranteed that this algorithm would change the weights connecting the nodes in such a network and their thresholds so that the network would solve the problem in an efficient manner.

Again, perceptrons are an abstraction. As Rosenblatt wrote (1962):

> Perceptrons are not intended to serve as detailed copies of any actual nervous system. They're simplified networks, designed to permit the study of lawful relationships between the organization of a nerve net, the organization of its environment, and the "psychological" performances of which it is capable.

2.3.4. Minsky

Minsky's PhD thesis was on learning as embodied in a machine with hundreds of vacuum tubes and thousands of connections. Later he became more interested in studying *what* constitutes intelligent behaviour than in *how* such behaviour is actually generated in biological systems. That is, symbolic programming. In Minsky and Papert (1972):

> Thinking is based on the use of SYMBOLIC DESCRIPTIONS and description-manipulating processes to represent a variety of kinds of KNOWLEDGE -- about facts, about processes, about problem-solving, and about computation itself ... The ability to solve new problems ultimately requires the intelligent agent to conceive, debug, and execute new procedures.

In their book Perceptrons, Minsky and Papert laid out a detailed analysis of the computational properties of perceptrons. They showed that such simple networks had severe limitations with respect to the class of problems for which they can compute solutions and that the computational resources required grow exponentially as problems scale up. At the same time many improvements in symbolic processing were being made - better lisp systems, the resolution theorem proving technique, etc. Symbolic programming looked like the way to go. Interest in neural networks waned. During the seventies most AI workers were exploring the capabilities of symbolic representations.

Ideas from traditional computer science and symbolic programming began to influence psychology. Indeed, Minsky and Papert wrote (1972):

> Until recently there was a serious shortage of ways to describe more procedural aspects of behaviour.
>
> The community of ideas in the area of computer science makes a real change in the range of available concepts. Before this, we had too feeble a family of concepts to support effective theories of intelligence, learning, and development.

2.4. Back to Brains

Although most of the computing community were pursuing symbolic programming, neuroscientists were still interested in the brain and during this period a number of new experimental techniques were pioneered from which emerged a much more detailed and complex picture of both individual neurons and entire brain structures. And some workers made progress with respect to characterizing the functioning of brain structures.

2.4.1. Marr

In the late sixties and early seventies Marr developed theories which attempted to characterize what certain brain regions are capable of doing. He concentrated on the cerebellum, the neocortex, and the archicortex (hippocampus) (Marr 1969, 1970, 1971). His work provides a good example of theoretical analysis of real nervous systems.

2.4.2. Longuet-Higgins and Willshaw

About the same time, Longuet-Higgens and Willshaw in the Department of A.I. at Edinburgh, were exploring the properties of distributed representations such as the associative net. Willshaw became more interested in biological systems and devoted most of the decade to doing theoretical neuroscience.

2.4.3. Koch and Poggio

They have been studying the behaviour of individual neurons in detail. In particular, they have worked to characterize what computations they can do and how they do them.

2.5. Parallel Distributed Processing

2.5.1. Hinton, Rumelhart and McClelland

Hinton was also in the Department of A.I. at Edinburgh in the early seventies working on distributed representations. He did some post-doc work at UCSD and while there inspired Rumelhart and McClelland to explore this area. Together they coined the term *parallel distributed processing* to describe the field.

They have continued to work in this area and have been involved in the development of some of the newer (post-perceptron) learning schemes for networks such as the Boltzmann Machine and the back propagation

algorithm. These algorithms will work for networks with more complex architectures than single layer perceptrons nets and have been responsible for much of the recent interest in the computing community.

Rumelhart and McClelland are cognitive psychologists who are involved with developing PDP models and using them to describe cognitive behaviour. In effect, they are working to provide a new set of metaphors for describing behaviour instead of the mechanistic ones which came from computer science.

2.5.2. Hopfield

He is a physicist who noticed that the equations which describe the behaviour of magnetic orientations in matter could also be used to describe the behaviour of a certain class of abstract neural network. These equations basically tell you about stable states - of the magnetic orientations or of the network. Such networks can be used as associative memories because if you start one with a state vector that is close (using some information theoretic metric) to a stable state, it will settle into that stable state. This class of networks has come to be known as *Hopfield nets*, although Hopfield is just one of many people who have 'invented' them.

3. Characteristics of Neural Networks

I have noted some of the history of the study of neural computation, but not really said much about what it is yet. People talk about the behaviour of neural networks - either biological or artificial, about 'settling' into solutions, about stable states, about distributed representations, etc. What do they mean?

3.1. An Example

To give an idea about what is meant when we talk about the 'behaviour' of a network let's work through a very simple example.

The following network has five nodes connected by some directed, weighted arcs. Each node has a state, either 0 or 1. The state of a node is determined by the total input coming into it, which is given by:

$$\textit{Total input to node}_j = \sum_i w_{ij}\, s_i$$

Where w_{ij} is the weight of the arc from node i to node j, and s_i is the state of node i.

There are many possible functions which give node state as a function of the total input. A simple one is:

Node state at time t+1 =
0 if total input (at time t) < 0
1 if total input (at time t) ≥ 0

The state of the network is simply a vector of the states of the individual nodes.

Let us specify the arcs by simply stating the matrix of the weights. A number in row i, column j denotes the weight of the arc connecting node i to node j.

$$\begin{bmatrix} 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & -2 & 0 & -2 \\ 0 & -2 & 0 & 0 & 2 \\ -2 & 0 & 0 & 0 & 0 \\ 0 & -2 & 2 & 0 & 0 \end{bmatrix}$$

I primarily use this notation as I cannot incorporate nice drawings into this file easily.

This defines a Hopfield network. This net is not intended to compute any pre-conceived function, but rather just to show how one state goes to the next.