A Brief History of Connectionism

David A. Medler
Biological Computation Project
Department of Psychology, University of Alberta
Alberta, Canada, T6G 2E9

Abstract

Connectionist research is firmly established within the scientific community, especially within the multi-disciplinary field of cognitive science. This diversity, however, has created an environment which makes it difficult for connectionist researchers to remain aware of recent advances in the field, let alone understand how the field has developed. This paper attempts to address this problem by providing a brief guide to connectionist research. The paper begins by defining the basic tenets of connectionism. Next, the development of connectionist research is traced, commencing with connectionism's philosophical predecessors, moving to early psychological and neuropsychological influences, followed by the mathematical and computing contributions to connectionist research. Current research is then reviewed, focusing specifically on the different types of network architectures and learning rules in use. The paper concludes by suggesting that neural network research—at least in cognitive science—should move towards models that incorporate the relevant functional principles inherent in neurobiological systems.

1 The Connectionist Revolution

> This solution takes the form of a new associationism, or better, since it differs deeply and widely from that older British associationism, of a new connectionism. ([109], p. 4)

Connectionist research is firmly established within the scientific community. Researchers can be found in such fields as artificial intelligence [33][1], cognitive neuroscience [76], economics [117][121], linguistics [84], philosophy [48], and physics [47] to name but a few. It has even been suggested that connectionism represents a Kuhnian-like paradigm shift for psychology [98]. But, perhaps the field that has most benefited from connectionist research is the multidisciplinary field of cognitive science [8][19][96][69][108]. As Hanson and Olson have stated: "The neural network revolution has happened. We are living in the aftermath" ([42], p. 332).

Unfortunately, this revolution has created an environment in which researchers may find it difficult to keep up with recent advances in neural network research. Furthermore, the history of connectionist research is often overlooked, or at least misconstrued [81]. As a result, a view popular with current researchers is that connectionism really emerged in the 1980's—there is only brief mention of research before that time (e.g., [8], [48]).

Connectionism, however, has a very long past. In fact, one can trace the origin of connectionist ideas to the early Greek philosopher, Aristotle, and his ideas on mental associations. These ideas were elaborated by the British empiricists and then naturally extended by the founders of psychology. Neuropsychologists

[0]This work was supported by a Killam Scholarship. The author is now at the Center for the Neural Basis of Cognition, Carnegie Mellon University, Pittsburgh, PA. Updates, corrections, and comments should be sent to David A. Medler at medler@cnbc.cmu.edu.

then contributed to the growth of connectionism by trying to relate the processes of learning and memory to underlying properties of the brain. But, this is only half of the picture. The other half of the picture is filled in by those researchers engaged in mathematical research and early computing science who contributed to the formal, computational understanding of both the power and limitations of connectionist networks.

Although it might be argued that these past researchers were not true "connectionists" in today's terms, the ideas they put forth in the disciplines of philosophy, psychology, neuropsychology, mathematics, and computing science are fully embodied within today's connectionism. And, it is only through a review of the contributions made by each of these disciplines that we can place connectionism in its proper context today.

2 Connectionism and Cognitive Science

Before we begin with our definition of connectionism, a brief digression is required. As noted earlier, connectionism is used in many different fields of science. For example, connectionist networks have been used for aiding astronomical work [106], assisting medical diagnosis [20], regulating investment management [121], and controlling robotic limb movement [113]. Many of these systems, however, are approached from an engineering perspective; that is, the designers are only interested in making the networks as efficient as possible (in terms of network topology, correct responses, and generalization). Consequently, this attitude towards connectionism could be characterized as the "engineering" approach. In fact, it may just be this approach that Reeke and Edelman had in mind when they offered this blunt assessment of connectionist research:

> These new approaches, the misleading label 'neural network computing' notwithstanding, draw their inspiration from statistical physics and engineering, not from biology. ([37], p. 144)

Although the engineering approach to connectionist research is of interest and demands much attention, in this paper we will review connectionism from a different perspective—that of cognitive science. This second approach uses connectionism to answer questions pertaining to human cognition, from perceptual processes to "higher level" processes like attention and reasoning. That is, connectionist cognitive scientists are interested in drawing their inspiration from biology, not technology. Consequently, the goals of the engineering approach (e.g., minimizing network structure, improving generalization, etc.) are not necessarily those of the cognitive science approach to connectionism. To understand what these goals are, however, we need to understand what cognitive science is.

2.1 Cognitive Science

The "birth" of cognitive science is often traced back to the Symposium on Information Theory held on September 10-12, 1956 at M.I.T. [36]. There, researchers from various disciplines gathered to exchange ideas on communication and the human sciences. Three talks in particular, Miller's *The magical number seven*, Chomsky's *Three models of language*, and Newell and Simon's *Logic theory machine*, have been singled out as instrumental in seeding the cognitive science movement. Following these talks, a perception began to emerge that "human experimental psychology, theoretical linguistics, and computer simulations of cognitive processes were all pieces of a larger whole" (Miller, 1979; p. 9; cited in [36], p. 29). That is, there arose a belief that to understand the functioning of human cognition, one had to combine the efforts of several different disciplines. In fact, similar sentiments had been expressed previously in the literature by such researchers as Hebb [44] and Wiener [120].

> ... a proper explanation of these blank spaces on the map of science (can) only be made by a team of scientists, each a specialist in his own field but each possessing a thoroughly sound and trained acquaintance with the fields of his neighbors ... ([120]; p. 9)

Today, cognitive science can be defined as the interdisciplinary study of mind; It draws upon such diverse fields as Computing Science and Artificial Intelligence [15], Linguistics [80], Neuroscience [85], Philosophy [59], and Psychology [36], to name but a few. Although each discipline has its own unique interpretation of cognitive science, they are bound into a cohesive whole by a central tenet. This tenet states that the mind is an information processor; that is, it "receives, stores, retrieves, transforms, and transmits information" ([105], p. 1). This information and the corresponding information processes can be studied as patterns and manipulations of patterns. Furthermore, these processes posit representational or semantic states that are fully realized within the physical constraints of the brain.

Traditionally, this information processing approach has been characterized by the physical symbol system hypothesis of Newell and Simon [77] which forms the basis of the "classical" approach to cognitive science. Basically, the hypothesis states that cognition is based upon patterns of information, that these patterns of information can be represented as symbols, and that these symbols can be manipulated. Consequently, it is sometimes assumed that the architecture of the mind *is* the architecture of von Neumann style computers (e.g., [86]). In contrast, connectionism is often viewed as a radically different approach to studying the architecture of the mind, accounting for aspects of human cognition handled poorly by the traditional approaches (e.g., graceful degradation, content-addressable memory; [78]). What, then, are the properties of connectionism that distinguishes it from the traditional approach to cognitive science?

2.2 Connectionism Defined

Connectionism—within cognitive science—is a theory of information processing. Unlike classical systems which use explicit, often logical, rules arranged in an hierarchy to manipulate symbols in a serial manner, however, connectionist systems rely on parallel processing of sub-symbols, using statistical properties instead of logical rules to transform information. Connectionists base their models upon the known neurophysiology of the brain and attempt to incorporate those functional properties thought to be required for cognition.

What, then, are the functional properties of the brain that are required for information processing? Connectionists adopt the view that the basic building block of the brain is the neuron. The neuron has six basic functional properties [27]. It is an input device receiving signals from the environment or other neurons. It is an integrative device integrating and manipulating the input. It is a conductive device conducting the integrated information over distances. It is an output device sending information to other neurons or cells. It is a computational device mapping one type of information into another. And, it is a representational device subserving the formation of internal representations. Consequently, we would expect to find these functional properties within our artificial neural networks.

As an example, Rumelhart, Hinton, and McClelland [91] (p. 46) list eight properties that are essential to *Parallel Distributed Processing* (PDP) models.

- A *set of processing units*
- A *state of activation*
- An *output function* for each unit
- A *pattern of connectivity* among units
- A *propagation rule* for propagating patterns of activities through the network of connectivities
- An *activation rule* for combining the inputs impinging on a unit with the current state of that unit to produce a new level of activation for the unit.
- A *learning rule* whereby patterns of connectivity are modified by experience
- An *environment* within which the system must operate

These eight properties of PDP models map easily onto the six functional properties of the neuron. The processing unit is the neuron itself. The state of activation and the activation rule are part of the input and intergrative device of the neuron and the output function is simply the output of the neuron. The pattern of connectivity and propagation rule map onto the conductive function of the neuron. And, the learning rule and environment are part of the computational and representational functions of the neuron.

To be fair, though, PDP models are simply a subclass of connectionist models. Therefore, Bechtel and Abrahamsen [8] have reduced the above list to four properties that distinguish the different types of connectionist architectures. These four properties are:

1. The connectivity of units,
2. The activation function of units,
3. The nature of the learning procedure that modifies the connections between units, and
4. How the network is interpreted semantically.

The above properties of connectionist models can be summarized in three basic tenets. First, signals are processed by elementary units. Second, processing units are connected *in parallel* to other processing units. Third, connections between processing units are weighted. These three tenets are necessarily broad in their descriptions so as to accommodate all aspects of connectionism; however, further elaboration is given below.

For example, the processing of signals encompasses the receiving, transformation, and transmission of information. The signals themselves may be carried by electrical, chemical, or mechanical means. Furthermore, signals could be supplied from an external stimulus (such as light impinging on the retina) or from other processing units. The processing units (see Figure 1) may refer to neurons, mathematical functions, or even demons *à la* Selfridge [100]. Lastly, information may be encoded in the units either locally or in a distributed manner.

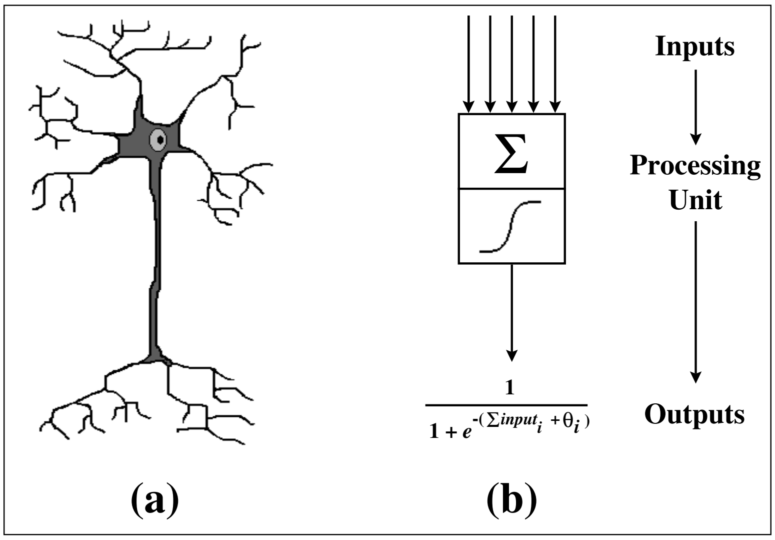

Figure 1: Different forms of processing units: (a) stylized sympathetic ganglion, (b) mathematical function.

Connections between units may or may not be massively parallel in the sense that every unit is connected to every other unit. Moreover, connections may be "feed-forward" (i.e., signals being passed in one direction only [92], [93]), or "interactive" (i.e., bidirectional passing of signals [66]).

Finally, the weights associated with the connections may be "hardwired", learned, or both. The weights represent the strength of connection (either excitatory or inhibitory) between two units. These three tenets allow a large spectrum of models (e.g., Selfridge's *Pandemonium* [100]; Rumelhart & McClelland's *Past-Tense Acquisition Model* [95]; Dawson's *Motion Correspondence Model* [18]) to fall within the classification of connectionist research.

To understand how these different models fit into connectionist research today, one needs to be aware of how connectionist research has developed. The best way of accomplishing this is to start at the beginning.

3 OLD CONNECTIONISM

We have chosen to analyze connectionism within the interdisciplinary realm of cognitive science. Consequently, we should not be surprised to find that connectionist research has an interdisciplinary origin. In fact, the essence of connectionism can be traced back to philosophy, psychology, mathematics, neuroscience, and computing science. It is only through a review of the contributions made by each of these disciplines that we can place connectionism in its proper context today.

3.1 Philosophical Roots

Although the popularity of connectionist research has grown considerably over the past decade, it is certainly not a new phenomenon. Aristotle (ca. 400 B.C.) has been cited [2] as the first scientist to propose some of the basic concepts of connectionism; that is, memory is composed of simple elements linked or connected to each other via a number of different mechanisms (such as temporal succession, object similarity, and spatial proximity). Furthermore, these associative structures could be combined into more complex structures to perform reasoning and memory access. Thus, a "well-specified outline of a perfectly viable computational theory of memory" ([2], p. 3) based on the interconnection of simple elements existed at least 2,400 years ago.

Moreover, many of the underlying assumptions of connectionism can be traced back to the ideas eminent in the philosophical school of materialism (e.g., la Mattrie, Hobbes), and the resulting school of British empiricism (e.g., Berkeley, Locke, Hume). Materialists held the view that nothing existed except for matter and energy, and that all human behaviour—including conscious thought—could be explained solely by appealing to the physical processes of the body, especially the brain (cf., Descartes' dualism). This lead to the empiricist view that human knowledge is derived ultimately from sensory experiences, and it is the association of these experiences that lead to thought [5][59]. Therefore, human cognition is governed by physical laws and can by studied empirically.

Within the empiricist tradition, accounting for psychological processes is known as associationism. The basic concepts of associationism are [8]:

1. mental elements or ideas become associated with one another through experience,
2. experience consists of such things as spatial contiguity, temporal contiguity, similarity, and dissimilarity of ideas,
3. complex ideas can be reduced to a set of simple ideas,
4. simple ideas are sensations, and
5. simple additive rules are sufficient to predict complex ideas composed from simple ideas

Although many associationist concepts are evident in the behaviourist movement in psychology, the cognitivist movement within psychology has dismissed associationism as inadequate to account for cognitive phenomenon such as recursive grammars (e.g., [10]).

Not surprisingly, with assumptions founded in associationist theories, connectionism has often been mistaken for associationism (e.g., [32], footnote 29), and subsequently dismissed as a viable theory of cognition. As pointed out by Thorndike [109], however, connectionism should not be confused for associationism. Rather, connectionism has borrowed concepts from associationism and has expanded them. For example, connectionism employs such concepts as distributed representations, hidden units, and supervised learning—concepts foreign to associationism [8].

In fact, Bechtel [7] points out that connectionism embodies a very distinctive characteristic that distinguishes cognitivism from behaviourism and associationism; specifically, connectionist modelers postulate that the connections between units provide structure in which mental activity occurs, and this structure is important for mediating future behaviour. Hence, connectionists are not repudiating cognitivism, they are simply providing an alternative to the standard rules and representation view of cognition. On the other hand, connectionism does embrace one very important aspect of associationism often missing from classical cognitive models; connectionism focuses on learning as a natural activity of the system being modeled. Consequently, Bechtel [7] concludes that connectionism may provide "a basis to draw together aspects of the two traditions that have generally been viewed as incommensurable" (p. 60).

3.2 Psychological Manifestations

With the emergence of psychology as a distinct field from philosophy, the ideas underlying connectionism became more refined and based on the known neurophysiology of the day. In fact, founding psychologists such as Spencer [103] and James [55] are often cited for early examples of connectionist networks—networks that combined associationist principles with neurology.

The appearance of the hardline behaviourist movement (e.g., [115], [102]), by all accounts, should have signaled the demise of connectionist ideas in psychology[1]. Surprisingly, however, it was behavioural psychologists (e.g., [109],[110],[51]), that finally made the distinction between associationism and connectionism [112]. Following the demise of behaviourism and the rise of cognitivism and symbolic processing, connectionist research all but disappeared from psychological literature. It has only recently become vogue once again.

But, for now, let us concentrate on psychology's contribution to connectionism.

3.2.1 Spencer's Connexions

In his two volume series entitled *The Principles of Psychology*, Herbert Spencer [103][104] laid out the foundations of what was then the emerging field of psychology. One of his central tenets was that a description of the nervous system was essential for the understanding of psychology. Thus, he devoted several sections of his text to describing neural structures and their functions. Part of this description included describing how connections may be formed—not only the connections between one neuron and another (see Figure 2), but also the connections between ideas and concepts. He even went so far as to state that "there is a fundamental connection between nervous changes and psychical states" ([103], p. 129).

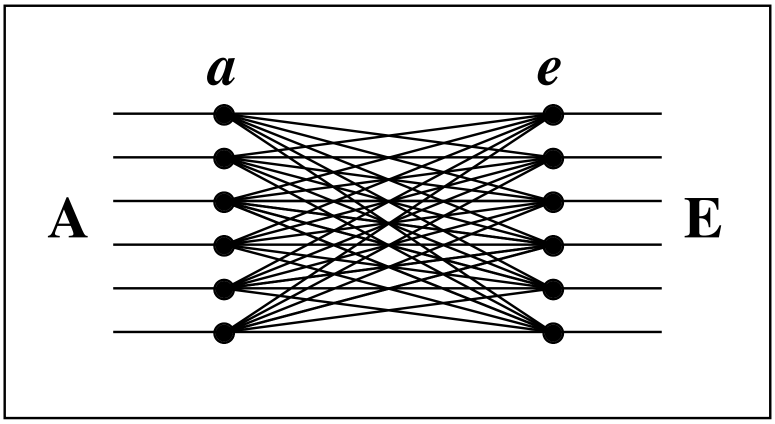

Figure 2: The needful connexions between afferent (A) and efferent (E) fibres to allow efficient transmission of a signal to move a muscle. Points a and e are where the afferent and efferent fibres diverge respectively (adapted from [103];vol. 1, Figure 7).

Using the growth of intelligence as an example, Spencer first identified those psychical aspects that define intelligence—the correspondence of internal relations with external relations. Intelligence grew as a function

[1] Watson proposed that psychology should only be interested in objective, observable behaviour: "the consideration of the mind-body problem affects neither the type of problem selected nor the formulation of the solution of that problem" [115]p. 166. This is exemplified by Skinner's view [102] that behaviour could be studied without any appeal to the brain.

of the differentiation of external events into ordered states of consciousness. Thus, changes in the psychical states could be linked directly to changes in the external order. As an infinite number of correspondences between internal and external events could exist over time, Spencer concluded that no general law could be stated for such a series of changes. Instead, a law of changes had to be sought in the small, immediately connected changes:

> When any state a occurs, the tendency of some other state d to follow it , must be strong or weak according to the degree of persistence with which A and D (the objects or attributes that produce a and d) occur together in the environment. ([103], pp. 408)

This law of connection also holds for events that co-exist in the world. If events A and B habitually coexist in the environment, then conscious states a and b must coexist as well. As neither A or B is antecedent or consequent, then state a is just as likely to induce state b as state b is to induce state a. Thus, the networks of connections could either be "feed-forward" (as in the case of a and d) or "interactive" (as in the case of a and b). As one last note, Spencer states that it is "the strengths of the connexion" ([103], p. 409) between the internal states and external events that is important. In other words, correct knowledge of the world is encoded within the connections of the brain.

3.2.2 James' Associative Memory

Further examples of early connectionist theory are also evident in William James' [55][56] treatment of psychology (interestingly enough, also a two volume set entitled *The Principles of Psychology*). James, like Spencer, was committed to the fact that psychological phenomenon could be explained in terms of brain activity—"*no mental modification ever occurs which is not accompanied or followed by a bodily change*" ([55], p. 5). In fact, James equated the analysis of neural functioning with the analysis of mental ideas.

> There is a complete parallelism between the two analyses, the same diagram of little dots, circles, or triangles joined by lines symbolizes equally well the cerebral and mental processes: the dots stand for cells or ideas, the lines for fibres or associations. ([55], p. 30)

The most obvious example of connectionist principles is James' associative memory model; the model consists of individual ideas that are connected in parallel such that recall of one idea is likely to cause the recall of related ideas. Thus, within this model, activation of event A with its component parts a, b, c, d, and e (e.g., attending a dinner party) caused activation of event B with its component parts l, m, n, o, and p (e.g., walking home through the frosty night) since all aspects of A were connected, or redintegrated, with all aspects of B (see Figure 3).

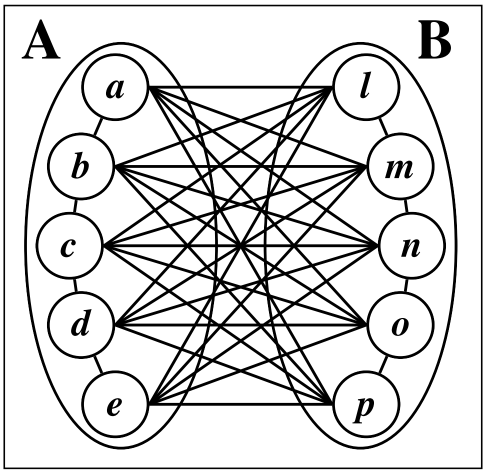

Figure 3: James' distributed memory model. Activation of event A causes activation of event B through weighted parallel connections.

James recognized that, all things being equal, any activation in such a network would unfortunately result in "the reinstatement in thought of the *entire* content of large trains of past experience"[2] ([55], p. 570). To counter this type of total recall, James proposed the law of interest: "some one brain-process is always prepotent above its concomitants in arousing action elsewhere" ([55], p. 572). Hence, not all connections in the brain are created equal. But, how are these connections modified, and hence the associations between memories learned? James proposed the law of neural habit:

[2] James quickly points out that only the minor personages within Dickens' and Eliot's novels possess this type of memory system.

When two elementary brain-processes have been active together or in immediate succession, one of them, on reoccurring, tends to propagate its excitement into the other. ([55], p. 566; his italics)

In other words, when two events occur repeatedly, the connection between the relevant brain-processes is strengthened (we shall see this notion reappear a little later in a more formal manifestation). Note that James is talking about modifying brain-processes physically and not simply strengthening the associations between ideas. Even now, we begin to see the borrowing and modification of associationist ideas to account for cognitive processes and learning in biological systems.

More importantly, these very simple concepts—weighted, modifiable, parallel connections—laid down over a century ago form the cornerstone of connectionism today.

3.2.3 Thorndike's Connectionism

Edward Lee Thorndike was a student of James; therefore, it is not surprising that he carried over some of the principles inherent in James' work. Although often considered one of the founding behaviourists (e.g., [82]), Thorndike was concerned with states of mind (cf., [115]), and how they changed with experience. More importantly, however, Thorndike can be considered one of the first true connectionists.

In his book, *The Fundamentals of Learning* [109], he differentiated between the principles of British associationism and what he had coined "new connectionism." He believed so much in this new connectionism that in 1949 he summarized what he considered his most important contributions to psychology under the title *Selected Writings from a Connectionist's Psychology* so that students may "know something of connectionist psychology" ([110], p. v).

Thorndike's connectionism can be viewed as a turning point where theories of neural association became sub-symbolic and graduated from merely implementational accounts to accounts of the functional architecture [112]. In other words, the neural connections became a substitute for, instead of a mechanism of, ideational processes. Thus, his computational descriptions of the fundamentals of learning were couched in the language of connectionist principles.

For example, to Thorndike, the most prevalent questions within learning theory were:

1. What happens when the same situation or stimulus acts repeatedly upon an organism—does the mere frequency of an experience cause useful modifications?

2. What happens when the same connection occurs repeatedly in a mind?

3. What effect do rewards and punishments have on connections, and how do they exert this effect?

In order to answer these questions, Thorndike proposed two different laws. The first law, the "Law of Exercise or Use or Frequency", states that all things being equal, the more often a situation connects with or evokes or leads to or is followed by a certain response, the stronger becomes the tendency for it to do so in the future. The second law, the "Law of Effect", states that what happens as an effect or consequence or accompaniment or close sequel to a situation-response, works back upon the connection to strengthen or weaken it. Thus, if an event was followed by a reinforcing stimulus, then the connection was strengthened. If, however, an event was followed by a punishing stimulus, then the connection was weakened. The principles underlying this law are very similar to the supervised learning techniques (such as error backpropagation) used in today's neural networks.

Finally, Thorndike anticipated the backlash against the principles of connectionism:

> Many psychologists would indeed deny that any system of connections was adequate to explain his behaviour, and would invoke powers of analysis, insight, purpose, and the like to supplement or replace the simple process of connection-forming by repetition and reward. ([109], p. 355)

Through a series of experiments, however, Thorndike [109] shows that there is "no sufficient reasons for ascribing any power over and above that of repetition and reward to any 'higher powers' or 'forms of thought' or 'transcendent systems' " (p. 382) and thus "justif[ies] the connectionist's faith" (p. 4).

3.2.4 Hull's Learning Rule

In 1943, Clark L. Hull[51] set for himself the task of elaborating the laws of behaviour from a molar level description of neural activity (since the results of molecular neurophysiology at the time were inadequate). As part of this elaboration, he described several functional properties of neural activity that he deemed important for organism survival. These include:

1. the afferent neural impulse (s_1) which is a non-linear function of the input it receives,
2. interactions between two or more afferent neural impulses (s_2 & s_3) which implies that behaviour to the same stimulus is not constant under all conditions, and
3. the spontaneous generation of nerve impulses which may account for the variability of behaviour to identical environments.

With these functional properties identified, Hull stated that the "supremely important biological process" of learning could be expressed in terms of modifying receptor-effector connections:

> The essential nature of the learning process may, however, be stated quite simply ... the process of learning consists in the strengthening of certain of these connections as contrasted with others, or in the setting up of quite new connections. (pp. 68-69)

The process of learning is wholly automatic—it occurs as the result of the interaction of the organism with its environment, both external and internal. Furthermore, the rules of learning must be capable of being stated in a clear and explicit manner without recourse to a guiding agent. Thus, Hull developed several empirically testable equations to describe the learning process. The one that concerns us the most for historical reasons is his formula for the growth of stimulus-response habits. This is simply the increase in the strength of connection (to a physiological maximum) between a stimulus and a response as a function of the number of reinforcing trials.

The growth of habit strength is dependent on three factors (p. 114):

1. The physiological limit or maximum (M),
2. The ordinal number (N) of the reinforcement producing a given increment to the habit strength ($\Delta^{\circ}_{S}H_R$),
3. The constant factor (F) according to which a portion ($\Delta_S H_R$) of the unrealized potentiality is transferred to the actual habit strength at a given reinforcement.

Thus, habit strength as a function of the number of reinforcement repetitions can be computed as follows

$$ {}^{N}_{S}H_R = M - Me^{-N \log \frac{1}{1-f}} \tag{1} $$

which generalizes over trials to

$$ \Delta H = f(M - H) \tag{2} $$

Hull is quick to point out that habit strength cannot be determined by direct observation; the strength of the receptor-effector connection can only be measured and observed indirectly. This is because the organization of the processes underlying habit formation are "hidden within the complex structure of the nervous system" (p. 102). Consequently, the only way of inferring habit strength is to note the associations between the antecedent conditions which lead to habit formation and the behaviour which is the consequence of these same conditions.

It has been pointed out [112] that this equation is a forerunner to the Rescorla-Wagner rule [87], which has been shown [107] to be essentially identical to the Widrow-Hoff [118] rule for training *Adaline* units (see Equation 6). Furthermore, this equation can be seen as a primitive form of the generalized delta rule for backpropagation in neural networks (see section 4.6, Equation 16)

3.3 The Neuropsychological Influence

Connectionist models derive their inspiration from neurophysiology. Consequently, it is appropriate to touch briefly on the neuropsychological influence exerted on connectionism. Following the pioneering work of such researchers as Sherrington and Cajal[3], researchers began to seek the neural correlates of learning and memory. From this research paradigm emerged two prominent figures in regards to the history of connectionism: Karl Lashley and Donald Hebb.

3.3.1 Lashley's Search for the Engram

One of the most intensive searches to localize memory traces—or engrams—within the brain was initiated by Karl Lashley in the 1920's. Lashley's studies involved training an animal to perform some specific task (such as brightness discrimination or maze orientation) and lesioning a specific area of the cortex either before or after training. Lashley then recorded the behavioural effects of cortical lesions on retention and acquisition of knowledge. In 1950 [58], he summarized 30 years of research into two principles:

- The *Equipotentiality* Principle: all cortical areas can substitute for each other as far as learning is concerned.
- The *Mass Action* Principle: the reduction in learning is proportional to the amount of tissue destroyed, and the more complex the learning task, the more disruptive lesions are.

In other words, Lashley believed that learning was a distributed process that could not be isolated within any particular area of the brain. Furthermore, it was not the location of the lesion that was important (within reason[4]), but the amount of tissue destroyed that determined the degree of behavioural dissociation. Although these two principles have been controversial since their publication, they do contribute to the field of connectionist research; specifically, to the ideas of distributed representations, multiple internal representations, and emergent network properties. In fact, recent lesioning experiments performed by connectionists (e.g., [29],[83]) would tend to agree with Lashley in terms of network processing being distributed and non-localized. Just as neuropsychologists have questioned Lashley's conclusions [53], however, the conclusions derived from experiments on lesioned connectionist networks are also being challenged [73]).

3.3.2 Hebbian Learning

Perhaps the most influential work in connectionism's history is the contribution of Canadian neuropsychologist, Donald O. Hebb (a student of Lashley). In his book, *The Organization of Behaviour* [44], Hebb presented a theory of behaviour based as much as possible on the physiology of the nervous system. Hebb reduced the types of physiological evidence into two main categories: (i) the existence and properties of continuous cerebral activity, and (ii) the nature of synaptic transmission in the central nervous system. Hebb combined these two principles to develop a theory of how learning occurs within an organism. He proposed that repeated stimulation of specific receptors leads slowly to the formation of "cell-assemblies" which can act as a closed system after stimulation has ceased. This continuous cerebral activity serves not only as a prolonged time for structural changes to occur during learning, but also as the simplest instance of a representative process (i.e., images or ideas).

The most important concept to emerge from Hebb's work was his formal statement (known as Hebb's postulate) of how learning could occur. Learning was based on the modification of synaptic connections between neurons. Specifically,

> *When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.* ([44], p.62; his italics)

[3]Sherrington was responsible for coining the term *synapse* to denote the structural and functional loci of interaction between neurons while Cajal was responsible for introducing the *neuron theory*—the nervous system is composed of neurons which are individual functional units.

[4]Lashley recognized that removing large portions of the visual cortex would prevent such things as brightness discrimination, but that this was due to the animal not being able to see, not to any deficit in learning or memory *per se*.

The principles underlying this statement have become known as Hebbian Learning. From a neurophysiological perspective, Hebbian learning can be described as a time-dependent, local, highly interactive mechanism that increases synaptic efficacy as a function of pre- and post-synaptic activity. Although the neurophysiology in Hebb's day was inadequate to support or deny Hebb's postulate, recent research has shown that Long-Term Potentiation (LTP) has those putative mechanisms required of Hebbian learning (e.g., [27]).

Within connectionism, Hebbian learning is an unsupervised training algorithm in which the synaptic strength (weight) is increased if both the source neuron and target neuron are active at the same time. A natural extension of this (alluded to by Hebb as the decay of unused connections) is to decrease the synaptic strength when the source and target neurons are not active at the same time. Hence, Hebbian learning can be formulated as:

$$w_{ij}(t+1) = w_{ij}(t) + net_i \, net_j \tag{3}$$

where

$w_{ij}(t)$ = the synaptic strength from neuron i to neuron j at time t

net_i = the excitation of the source neuron.

net_j = the excitation of the destination neuron.

There are serious limitations with Hebbian learning as stated (e.g., the inability to learn certain patterns), but variations of this simple algorithm exist today (e.g., Signal Hebbian Learning; Differential Hebbian Learning; [114]).

3.4 The Mathematical Influence

The next major formulation of connectionist theories can be attributed to McCulloch and Pitts [70]. In their seminal paper *A logical calculus of the ideas immanent in nervous activity*, they explicitly laid out the foundations of neural modelling in terms of propositional logic. To accomplish this, they simplified the activity of neurons into five functional states (p. 118):

1. The activity of the neuron is an "all-or-none" process.
2. A certain fixed number of synapses must be excited within the period of latent addition in order to excite a neuron at any time, and this number is independent of previous activity and position on the neuron.
3. The only significant delay within the nervous system is synaptic delay.
4. The activity of an inhibitory synapse absolutely prevents excitation of the neuron at that time.
5. The structure of the net does not change with time.

Using these principles, McCulloch and Pitts were able to show that *any* statement within propositional logic could be represented by a network of simple processing units. Furthermore, such nets have the *in principle* computational power of a Universal Turing Machine. "If any number can be computed by an organism, it is computable by these definitions, and conversely" (p. 128). Since all information processing can be characterized by a Turing Machine (e.g., [111]), it was assumed that human cognition could also be characterized by a Turing Machine. Consequently, McCulloch and Pitts concluded that:

> To psychology, however defined, specification of the net would contribute all that could be achieved in that field— even if the analysis were pushed to the ultimate psychic units or "psychons," for a psychon can be no less than the activity of a single neuron. (p. 131)

McCulloch and Pitts also proved that there is always an indefinite number of topologically different nets realizing any temporal propositional expression (TPE), although time discrepancies might exist between the different realizations. What this states is that there exists many different algorithms to compute the same function, or similarly, many different possible network configurations (say in terms of processing units and connections).

3.5 Early Computer Models of Connectionism

> Logically, if it were possible to construct non-living devices — perhaps even of inorganic materials — which would perform the essential functions of the conditioned reflex, we should be able to organize these units into systems which would show true trial-and-error learning with intelligent selection and the elimination of errors, as well as other behavior ordinarily classed as psychic. Thus emerges in a perfectly natural manner a direct implication of the mechanistic tendency of modern psychology. Learning and thought are here conceived as by no means necessarily a function of living protoplasm any more than is aerial locomotion. [52] pp. 14-15.

Perhaps the most influential event in the development of connectionism was the invention of the modern computer. Theories that could only be tested previously by observing the behaviour of animals or humans (e.g., [109][110][51]) could now be stated more formally and investigated on artificial computation devices. Hence, theory generation and refinement could now be accomplished faster and with more precision by using the empirical results generated by the computer simulations.

The computer and its influence on learning theory can be credited with producing both positive and negative press for connectionism. Selfridge's *Pandemonium* [100] and Rosenblatt's *Perceptrons* [89][90] did much to further the concepts of connectionism. The proofs on the limitations of simple perceptrons by Minsky and Papert [74], however, nearly caused the complete abandonment of connectionism.

3.5.1 Pandemonium

Recognizing that previous attempts to get machines to imitate human data had all but failed, Selfridge [100] proposed a new paradigm for machine learning. *Pandemonium* was introduced as a learning model that adaptively improved itself to handle pattern classification problems that could not be adequately specified in advance. Furthermore, whereas previous computer models relied on serial processing, Selfridge proposed a novel architecture to deal with the problem, parallel processing. The move to parallel processing was not an arbitrary one, but one motivated by two factors: (1) it is easier, and more "natural" to handle data in a parallel manner[5], and (2) it is easier to modify an assembly of quasi-independent modules than a machine whose parts interact immediately and in a complex way.

Pandemonium consists of four separate layers: each layer is composed of "demons" specialized for specific tasks. The bottom layer consists of data or image demons that store and pass on the data. The third layer is composed of computational demons that perform complicated computations on the data and then pass the results up to the next level. The second layer is composed of cognitive demons who weight the evidence from the computational demons and "shriek" the amount of evidence up to the top layer of the network. The more evidence that is accumulated, the louder the shriek. At the top layer of the network is the decision demon, who simply listens for the loudest "shriek" from the cognitive demons, and then decides what was presented to the network.

The initial network structure is determined a priori by the task, except for the computational level which is modified by two different learning mechanisms. The first mechanism changes the connection weights between the cognitive demons and the computational demons via supervised learning (all other connections within the network being fixed a priori). The weights are trained using a hill-climbing procedure in order to optimize the performance of the network. After supervised learning has run long enough to produce approximately optimal behaviour, the second learning mechanism is employed.

The second learning mechanism selects those computational demons that have a high worth (based on how likely they are to influence a decision), eliminates those demons that have a low worth, and generates new demons from the remaining good demons. Generation can be accomplished by either mutating a demon, or conjoining two successful demons into a continuous analogue of one of the ten nontrivial binary two-variable functions. It should be noted that this second mechanism may be one of the first genetic machine learning algorithms.

[5] "parallel processing seems to be the human way of handling pattern recognition" [99] p. 66.

Selfridge has demonstrated the effectiveness of *Pandemonium* on two different tasks: distinguishing dots and dashes in manually keyed Morse code [100], as well as recognizing 10 different hand-printed characters [99]. Thus a practical application of connectionist principles have been applied to pattern recognition. In fact, *Pandemonium* has been so successful as a model of human pattern recognition that it has been adopted and converted into a more traditional symbolic model (with connectionist principles appropriately ignored) by cognitive psychologists (e.g., [62])

3.5.2 The Perceptron

> The perceptron, more precisely, the theory of statistical separability, seems to come closer to meeting the requirements of a functional explanation of the nervous system than any system previously proposed. [89] p. 449.

Although originally intended as a genotypic model of brain functioning [89][90], the *perceptron* has come to represent the genesis of machine pattern recognition. Basically, the perceptron is a theoretically parallel computation device composed of (i) a layer of sensory units (S-unit) which transduce physical energy (e.g. light, sound, etc.) into a signal based on some transformation of the input energy, (ii) any number of layers of association units (A-unit) which have both input and output connections, and (iii) a final layer of response units (R-unit) which emit a signal that is transmitted to the outside world. Figure 4 shows different graphical representations of a perceptron system. Note that the same perceptron system can be expressed in terms of a network diagram, a set diagram, or even a symbolic diagram.

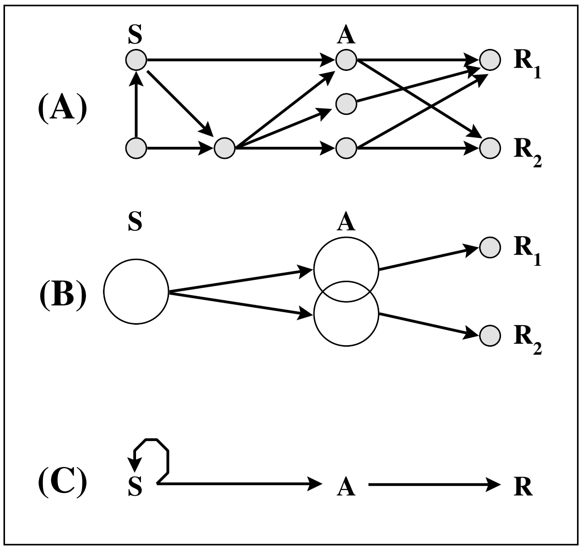

Figure 4: Different diagrams representing the same perceptron system. (A) Network diagram. (B) Set Diagram. (C) Symbolic Diagram. (Adapted from [90]; Figure 2, p. 86).

An elementary α-perceptron is defined, then, as a network in which S-units are connected to A-units (although not necessarily massively parallel), and all A-units are connected to a single R-unit, with no other connections being permitted. Furthermore, all connections are considered to have equal transmission rates, τ. The transfer function between units i and j at time t is expressed as

$$c^*_{ij}(t-\tau) = u^*_i(t-\tau)\, v_{ij}(t-\tau) \tag{4}$$

where $u^*_i(t-\tau)$ is the output of unit i at time t, and $v_{ij}(t-\tau)$ is the connection strength between units i and j at time t. Connection strengths from S- to A-units are fixed, while connection strengths from A- to R-units vary with the reinforcement history applied to the perceptron. Both A- and R-units have a threshold, θ, and emit a signal whenever the input signal, α, is equal to or greater than θ. We can assume that $\tau = 0$ without loss of generality, and thus, the reinforcement rule is

$$\Delta v_{ij} = u^*_i(t) \cdot \eta = \begin{cases} \eta \; if \; \alpha_i(t) \geq \theta, \\ 0 \; otherwise. \end{cases} \tag{5}$$

where η is of constant magnitude.

The theoretical importance of the elementary α-perceptron lies in the fact that, for binary inputs, a solution exists for every classification, $C(W)$, of all possible environments W. In other words, an elementary

α-perceptron is capable of solving *any* pattern classification problem expressed in binary notation. The proof is rather trivial:

1. For every possible pattern, S_i, in W, let there be a corresponding A-unit, a_i.
2. Make the connection, v_{ij}, between a_i and the corresponding sensory unit, s_j, excitatory (i.e., value equal to +1) if the pattern on that s_j is "on"; otherwise make the connection inhibitory (i.e., value equal to -1).
3. Set the threshold of a_i, θ, equal to the number of excitatory connections. Thus, a_i responds to one and only one pattern in W.
4. If S_i is a positive instance of $C(W)$ then make the connection from a_i to the R- unit positive (i.e. value equal to +1); otherwise make the connection negative (e.g., value equal to -1). With the threshold of the R-unit equal to zero, the network correctly classifies all S_i in W.

The above proof shows theoretically that any pattern classification problem expressed in binary notation can be solved by a perceptron network. As a concrete example, Figure 5 shows the network configuration for solving the XOR problem. The problem with this proof, however, is that it produces the final network structure, but does not indicate if the network could be trained to such a configuration. Consequently, Rosenblatt developed the *Perceptron Convergence Theorem* to show that an elementary α-perceptron using an error correction procedure is guaranteed to converge on a solution in finite time, providing that (i) a solution exists, (ii) each pattern is presented to the network at least twice, and (iii) the connections between the S-units and A-units are fixed.

S_i	A-Unit	*Response*
off off	a_0	*off*
on off	a_1	*on*
off on	a_2	*on*
on on	a_3	*off*

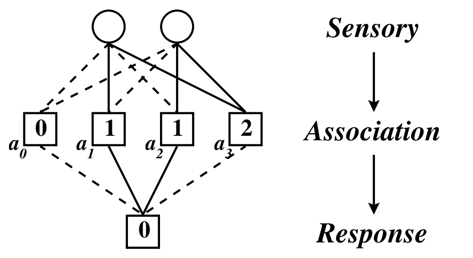

Figure 5: The solution for an elementary α-perceptron solving the XOR problem. Weights are either +1 (solid line) or -1 (broken line), whereas biases are indicated by the number within the processing unit.

Although theoretically very powerful, the practical problem with perceptrons was that there was no reliable method of adjusting the connections between the sensory (input) units and the association (internal) units. Hence, as a true learning network, perceptrons were limited to just a layer of sensory units connected directly to a layer of response units, with no intervening layers. With the output of the R-unit being monotonic (i.e., $u_i^*(t) = f(\alpha_i(t))$, where $\alpha_i(t)$ is the algebraic sum of all the inputs into unit u_i), the resulting networks were very limited in their computational power. Rosenblatt was quick to point this limitation out, although he left the proof up to the reader.

> It is left to the reader to satisfy himself that a system with less "depth" than an elementary perceptron (i.e., one in which S-units are connected directly to the R- unit, with no intervening A-units) is incapable if representing $C(W)$, no matter how the values of the connections are distributed. [90] p. 101.

3.5.3 Adaline

The next major formulation in learning rules for networks came from Widrow and Hoff [118]. They developed "*Adaline*" (first for *ada*ptive *line*ar, then *ada*ptive *li*near *ne*uron, and later *ada*ptive *line*ar element as neural models became less popular) as an adaptive pattern classification machine to illustrate principles of adaptive behaviour and learning. The learning procedure was based on an iterative search process, where performance feedback was used to guide the search process. In other words, a designer "trains" the system by "showing" it examples of inputs and the respective desired outputs. In this way, system competence was directly and quantitatively related to the amount of experience the system was given.

The typical Adaline unit, also called a "neuron element," is illustrated in Figure 6. It is a combinatorial logic circuit that sums the signals from weighted connections (gains), a_i, and then sends an output signal based on whether or not the internal signal exceeded some threshold. The threshold was determined by a modifiable gain, a_0, which was connected to a constant +1 source. As opposed to the usual convention of using signals of 0 and 1, the Adaline used input signals of -1 and +1 which meant a signal was *always* passed along a connection (unless the gain on the line was zero). Similarly, the gains on the connections were adjusted so that the output signals were exactly -1 or +1; therefore, classification was not simply correct, but *exactly* correct. This restriction on the outputs meant that learning continued even if the classification was simply correct as the summed inputs may not be exactly -1 or +1. This continued learning was an improvement over the simple perceptron which did not change its weights if the gross classification was correct.

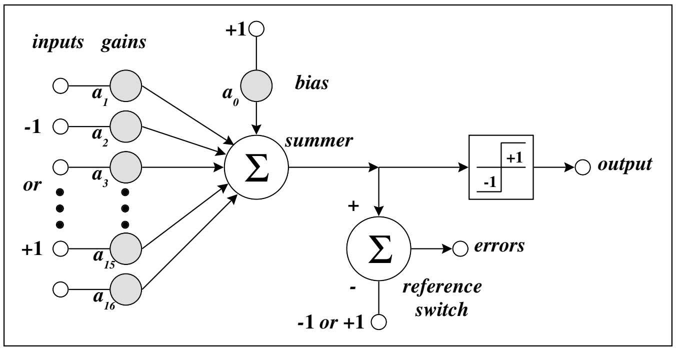

Figure 6: A typical Adaline unit. The inputs are sent along weighted connections (gains) to a summer which performs a linear combination of the signals. The output of the summer is compared to the value of the reference switch and the gains are adjusted by the same absolute value to produce an output of exactly -1 or +1.

The learning procedure is based on the error signal generated by comparing the network's response with the optimal (correct) response. For example, consider an Adaline unit with 16 input lines and a bias threshold. A pattern is presented over the 16 input lines, and the desired output is set into the reference switch (see Figure 6). If the error (computed as the difference between the summer and the reference switch) is greater than zero, then all gains including the bias are modified in the direction that will reduce the error magnitude by 1/17. Upon immediate representation of the pattern an error signal of zero would be produced. Another pattern can now be presented to the network and the connections modified[6]. Convergence is achieved when the error (before adaptation) on any given pattern is small and there are small fluctuations about a stable root mean-square value.

The Widrow-Hoff rule [107] is formulated as:

$$r_i = z[(t) - y(t)]x_i(t) \tag{6}$$

where t is the target pattern, $y(t)$ is the network's output, and $x_i(t)$ is the input to the network. Because this rule is dependent on an external teacher it is termed *supervised* learning. The Widrow-Hoff rule is also known as the *delta rule* because the amount of learning is proportional to the difference between the output and the target [91].

[6]Note that at this point, presenting the first pattern to the network would produce an error that was small but not necessarily zero.

3.5.4 Perceptrons Revisited (Minsky & Papert)

Although it was known for a decade that simple perceptrons were limited in their ability to classify some patterns, it was not until Minsky and Papert published *Perceptrons* in 1969 that the extent of these limitations were fully realized. In fact, it was with this publication that the connectionist tide was stemmed[7] (at least for a while). Instead of asking if neural networks are good, Minsky and Papert asked the question "what are neural networks *good for*?" This is clearly a computational level question aimed at identifying the limitations of the representational abilities of perceptron-like networks. As Minsky and Papert point out in their prologue to the 1988 edition of *Perceptrons*, "No machine can learn to recognize X unless it possesses, at least potentially, some scheme for *representing* X." (p. xiii; their italics).

Hence, their approach to the study of neural networks was based on studying the types of problems that were being proposed at the time—mainly visual pattern recognition [81]. In doing so, they discovered that some pattern recognition problems (e.g., distinguishing triangles from squares) were relatively easy and could be computed by simple networks. Conversely, some problems (e.g., determining if a figure was connected or not) were extremely difficult and required large networks to solve them. The main distinction between these two types of problems was not the size of the pattern space, but the concept of **order** [74], p. 30.

In general, the order of some function $\Psi(X)$ is the smallest number k for which we can find a set Φ of predicates satisfying

$$\begin{cases} |S(\varphi_p)| \leq k \;\forall\; \varphi \; in \; \Phi, \\ \qquad \Psi \in L(\Phi). \end{cases} \tag{7}$$

where φ is a simple predicate, and $L(\Phi)$ is the set of all predicates that are linear threshold functions. It should be noted that the order of Ψ is a property of Ψ alone, and not relative to any particular Φ. Functions that have an order of 1 are called "linearly separable" and can be solved by a single layer perceptron.

The types of pattern recognition problems that gave simple perceptrons trouble were those whose order was greater than 1. These types of problems are termed "linearly inseparable" and require a layer of processing units between the input and output units. At the time, however, there was no reliable method of training this intermediate level, and therefore perceptrons were limited to being trained on linearly separable problems only.

Minsky and Papert [74] used a very simple and elegant example to show the practical limitations of perceptrons. The exclusive-or (XOR) problem (see Figure 5) contains four patterns of two inputs each; a pattern is a positive member of a set if either one of the input bits is on, but not both. Thus, changing the input pattern by one bit changes the classification of the pattern. This is the most simple example of a linearly inseparable problem (see Figure 7). A perceptron using linear threshold functions requires a layer of internal units to solve this problem, and since the connections between the input and internal units could not be trained, a perceptron could not *learn* this classification. And, if perceptrons failed on this small pattern set, what hope was there for larger pattern sets that were also linearly inseparable?

Furthermore, Minsky and Papert lay out other limitations of networks. For example, if a network is to solve a problem with order R, then at least one partial predicate φ must have as its support the *whole space* R. In other words, at least one internal unit must be connected to each and every input unit. This network configuration violates what is known as the "limited order" constraint. Another limitation that Minsky and Papert discuss is the growth of coefficients. For linearly inseparable problems, the coefficients (i.e., weights) can increase much faster than exponentially with $|R|$. This leads to both conceptual and practical limitations. Conceptually, although the behaviour of a network may be "good" on small problems, this behaviour may become profoundly "bad" when the problem is scaled up. Practically, for very large $|R|$, the amount of storage space required for the weights would overshadow the space required to simply represent the problem.

Although advances in neural network research have produced methods for training multiple layers of units (e.g., [92] [93]), many of Minsky and Papert's concerns remain unanswered. Networks using linear threshold units still violate the limited order constraint when faced with linearly inseparable problems (but

[7] Neural network research did not wane due to lack of interest, but because of lack of funding [81]

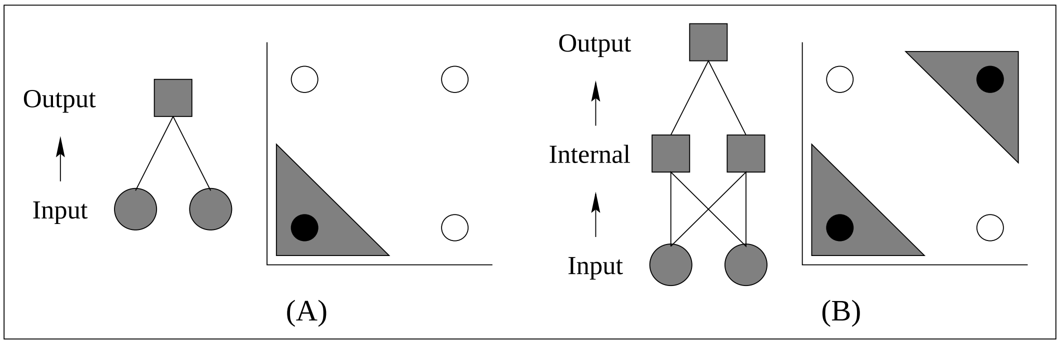

Figure 7: (A) Linearly separable problem—the pattern space requires a single hyperplane to make the proper classification; therefore, a network requires no hidden units. (B) Linearly inseparable problem—the pattern space requires two (or more) hyperplanes to make the correct classification; therefore, a network requires a layer of internal units.

see section 4.8). Furthermore, the scaling of weights as the size of the problem space increases remains an issue [31].

3.6 The Importance of Old Connectionism

The publication of *Perceptrons* by Minsky and Papert in 1969 has taken on almost a mythical aura—it has been likened to the huntsman being sent out to bring back the heart of Snow White [81]. Regardless of whether or not the work precipitated or merely coincided with the decline of connectionist research, it serves as a useful delineation between the "Old" and "New" connectionism.

The examples of connectionist networks provided in this section are often classified under the term "Old Connectionism". Old Connectionism is characterized by two different types of networks. The first are small, trainable networks, such as single layer perceptrons, that are computationally limited (i.e., cannot be trained to solve linearly inseparable problems). The second type of networks are large, computationally powerful networks that are mainly hardwired (although they could have a trainable layer of weights such as *Pandemonium*), and thus are limited in their learning ability. The problem with Old Connectionism was that it had no reliable way of combining these two different types of network architectures. To be an effective tool within cognitive science, researchers had to find a way of combining these two different types of networks.

Consequently, we are left with the question "Why should we be interested in Old Connectionism?" The first reason is purely academic. To understand the role of connectionism today we have to understand how the field has developed. By knowing the history of connectionism, not only are we in a position to counter the arguments against connectionism from the classical camp (e.g., knowing why connectionism is not associationism), but also we are in a position to evaluate claims from the connectionist camp that it may represent a paradigm shift [98]. To be effective researchers, we need to know both sides of the argument.

The second and more important reason is that by studying the development of connectionism we can appraise the strengths and weaknesses of the connectionist approach to information processing and adjust our course of inquiry accordingly. For example, we know that connectionist networks have the *in principle* power of a UTM [70], but we also know that perceptron-like single layer networks are limited in their computational power [74]. Thus, we should focus current research on multilayer networks. We know that there are guaranteed algorithms based very much on early behaviourist theorizing for training single layer networks [90], yet no such algorithm exists for multiple layer networks. Can the same be said of biological learning? Finally, we should stop working in the "biological vacuum" and heed the echoing call for models of learning to be based more on the known neurophysiology of the brain.

With the introduction to connectionism's interdisciplinary background from its philosophical roots to its computational apex completed, the current state of connectionism can now be evaluated.

4 New Connectionism

This section is concerned with describing connectionist systems in the post-*Perceptrons* era; that is, networks falling under the classification of "New Connectionism". New Connectionism is characterized by computationally powerful networks that can be fully trained. Such networks have often been hailed as providing a simple universal learning mechanism for cognition (but see [34]). Moreover, the learning algorithms embodied within new connectionist models have created very powerful information processors—they are both universal function approximators [16] and arbitrary pattern classifiers [63].

As stated earlier, we are living in the aftermath of the neural network revolution. As a consequence, the number of different connectionist architectures available to researchers today is immense; to discuss them all is beyond the scope of this paper. Instead, this section will focus on three specific architectures and provide a cursory examination of four other connectionist architectures.

It should be noted, however, that the demarcation between"Old" and "New" is somewhat tenuous. Following the publication of *Perceptrons*, there was a decrease in the number of researchers actively engaged in connectionist research; but, research did not cease. In certain respects, however, there was a change in the focus of connectionist research. Whereas previous researchers were interested in a connectionist theory of mind, the focus of research during the 1970's and early 1980's was more directed towards a connectionist theory of memory. This is exemplified by the work on associative memory models reported in Hinton and Anderson [45]. The models described in *Parallel Models of Associative Memory* were seen as a departure from standard memory models of the time for three distinct reasons (e.g., [97]):

1. The systems were assumed to have a neurophysiological foundation,
2. The systems offered an alternative to the "spatial" metaphor of memory and retrieval, and
3. The systems assumed a parallel, distributed-processing system that did not require a central executive to coordinate processing.

These researchers were aware of the limitations of connectionist models for learning linearly inseparable pattern classification tasks; consequently, the focus of their research was directed more towards how memory was stored and retrieved. In many ways, the work presented in Hinton and Anderson (1981) serves an important role by bridging the gap between *Perceptrons* and *Parallel Distributed Processing.*

4.1 Modern Connectionist Architectures

In this section, three different network architectures will be described in detail. After each of the main architectures is described, related network architectures will also be reviewed. These reviews will provide somewhat less detail as they are meant to provide only a cursory examination of comparable architectures.

The first architecture to be described in detail is James McClelland's [66] *Interactive Activation and Competition* (IAC) model of information retrieval from stored knowledge. Although early versions of the IAC architecture did not learn—hence, it could rightly be considered within the class of Old Connectionism as defined earlier—the model displays many characteristics of human cognition that are missing from classical symbolic models. Furthermore, a new learning rule for IAC networks is proposed within this section. Thus, it is included here in our description of modern connectionism.

Following the description of the IAC network, the work of two pioneering researchers in the field of neural network learning immediately following the Perceptrons era will be briefly reviewed. First, Stephen Grossberg's [39][40] instar and outstar configurations and his Adaptive Resonance Theory (ART) networks will be introduced. Second, we will cover Teuvo Kohonen's [57] self-organizing maps (which are now commonly referred to as Kohonen networks). Coupled with the new learning rule for the IAC networks, these architectures provide a link from the"Old" to the "New" Connectionism.

The second architecture to be covered in detail is the generic PDP architecture[8]; that is, a multi-layered network trained with Rumelhart, Hinton and Williams' [92][93] backpropagation algorithm. The generic PDP network is probably the most well known and most widely used architecture today it is estimated that about 70% of real-world network applications use the backpropagation learning algorithm [116]. Furthermore, the algorithm is suitable for both function approximation tasks and pattern classification problems.

One criticism leveled against the generic PDP architecture, however, is that is only capable of a static mapping of the input vectors. The brain, on the other hand, is not stateless but rather a high-dimensional nonlinear dynamical system [26]. Consequently, the recurrent network architecture pioneered by John Hopfield [47] will be briefly discussed. The basic characteristic of recurrent networks is that some processing activation (usually the output) at time t is re-used (usually as an input) at time t+1. Thus, a fully connected recurrent network is potentially a very powerful architecture for temporal processing; however, more efficient heuristics and algorithms for reliable learning are required.

The third architecture to be discussed specifically is a variation on the generic PDP architecture developed by Dawson and Schopflocher [24]. These value unit networks use the same basic learning algorithm as the generic PDP architecture, but use a non-monotonic activation function—the Gaussian—in their processing units. This new activation function has been shown to have certain theoretical and practical advantages over standard backpropagation networks. For example, value units are able to solve linearly inseparable problems much easier and with fewer hidden units than standard networks. Also, the hidden unit activations adopted by value unit network often fall into distinct "bands", allowing for easier interpretation of the algorithms being carried out by the network.

Finally, the last architecture to be briefly covered is the Radial Basis Function (RBF) network (e.g., [75]). The reason for covering the RBF architecture is that it and the value unit architecture are often confused. This is because both networks use a Gaussian activation function in their processing units. As the section will show, however, the networks are not equivalent.

4.2 Interactive Activation and Competition Models

McClelland's [66] *Interactive Activation and Competition* (IAC) model illustrates the power of a large network for retrieving general and specific information from stored knowledge of specifics. Although the network rightly falls into the category of Old connectionism as defined earlier (i.e., the network is hardwired and cannot "learn" new information), it is included in this section because it nicely illustrates those properties of an information processing system that are often overlooked in classical theories of cognitive science. These include graceful degradation, content-addressable memory, output availability, and iterative retrieval [78]. Furthermore, the network suggests that we may not need to store general information explicitly.

The basic IAC network consists of processing units that are organized into competitive pools. Connections within pools are inhibitory; this produces *competition* within the pools as strong activations tend to drive down weaker activations within the same pool. Connections between pools, however, are normally excitatory and bi-directional; thus, we have *interactive* processing. Units within the network take on continuous *activation* values between a minimum and a maximum, with their output normally equal to the activation value minus some threshold (although this can be set to zero without loss of generality). The basic mathematics of network functioning are fairly straight-forward (e.g., [41][68]).

Units within the IAC network compute their activation, a_i, based upon the unit's current activation and the net input. The net input arriving into a unit (see Equation 8) is calculated by summing the weighted activations sent from all other internal units connected to it, plus any external activation supplied by the environment. Thus, the net input to unit i that is connected to j other units is

$$net_i = \sum_j w_{ij}\ output_j + extinput_i \tag{8}$$

[8]The term 'generic' was first coined by [3] to describe this type of network architecture and is maintained here for consistency.

where w_{ij} is the weight (positive or negative) of the connection between unit i and unit j, and

$$output_j = [a_j]^+ = \begin{cases} a_j \ if \ a_j > 0, \\ 0 \ otherwise. \end{cases} \tag{9}$$

Once the net input to a unit has been calculated, the change in that unit's activation can be computed as follows:

$$\begin{aligned} &If \ (net_i > 0), \\ &\qquad \Delta a_i = (max - a_i)net_i - decay(a_i - rest). \\ &Otherwise, \\ &\qquad \Delta a_i = (a_i - min)net_i - decay(a_i - rest). \end{aligned}$$

where max, min, $decay$, and $rest$ are parameters supplied by the modeler. Normally, the parameters are set to $max = 1$, $min \leq rest \leq 0$, and $0 \leq decay \leq 1$. It is also assumed that a_i is initialized and remains within the range $[min, max]$.

With these equations in place, we can evaluate how Δa_i changes over time. For example, imagine that the input, net_i, to a unit is fixed at some positive value. As the activation, a_i, of a unit becomes greater and greater, Δa_i becomes less and less—when a_i reaches max then $\Delta a_i = -decay(a_i - rest) = -decay(max - rest)$. When a_i is equal to the resting level, then $\Delta a_i = (max - rest)net_i$. If we assume that $max = 1$ and $rest = 0$, then these equations reduce to $\Delta a_i = -decay$ when a_i is maximal and $\Delta a_i = net_i$ when a_i is minimal. Between these two extremes is the *equilibrium* point, where $\Delta a_i = 0$; that is, we can calculate the value of a_i such that given a constant net input, the unit's activation does not change with time. To determine the equilibrium point (assuming $max = 1$ and $rest = 0$), we simply set Δa_i to zero and solve for a_i which gives:

$$\begin{aligned} 0 &= (max - a_i)net_i - decay(a_i - rest) \\ 0 &= net_i - (a_i)(net_i) - (a_i)(decay) \\ 0 &= net_i - a_i(net_i + decay) \\ a_i &= \frac{net_i}{net_i + decay} \end{aligned} \tag{10}$$

This means equilibrium is reached when the activation equals the ratio of the net input divided by the net input plus the decay. Analogous results to Equation 10 are obtained when the net input is negative and constant. It should be noted that equilibrium is only reached when the net input to the unit is constant—if the net input changes with time, then equilibrium is not guaranteed (in practice, however, equilibrium is often achieved).

Having analyzed the mathematical basis of the network, we can now turn our attention to a more specific example of the IAC architecture. McClelland's [66] network is based on the bi-directional interconnection of nodes. A node is a simple processing device that accumulates excitatory and inhibitory signals from other nodes via weighted connections and then adjusts its output to other nodes accordingly. There are two different types of nodes in the network: *instance* and *property* nodes[9]. There is one instance node for each individual encoded in the network. The instance node has inhibitory connections to other instance nodes and excitatory connections to the relevant property nodes. The property nodes encode the specific characteristics of an individual. Property nodes are collected into cohorts of mutually exclusive values; nodes within a cohort have mutually inhibitory connections. Knowledge is extracted from the network by activating one or more of the nodes and then allowing the excitation and inhibition processes to reach equilibrium.

All information processing models take time, and the IAC model is no exception. Time is measured in "cycles", where a cycle consists of a node computing its activation level (dependent on previous activation,

[9] In reality, both of the nodes have the same physical characteristics and therefore only represent different types of information. It is often assumed, however, that instance nodes are 'hidden' from direct access whereas property nodes are not.

current excitatory and inhibitory signals being received, and a decay function) and then sending a signal to all connected nodes. During a cycle, all nodes are computing their activation levels in parallel. Furthermore, at the completion of any cycle, we can evaluate the current state of the network. This means that we can wait for the network to reach equilibrium and a definite answer, or we can ask what the network's best "guess" to a question is before equilibrium is reached.

The specific network reported [66] encodes information about the members of two gangs called the "Jets" and the "Sharks". Property cohorts include NAME, GANG, AFFILIATION, AGE, EDUCATION STATUS, MARITAL STATUS, and OCCUPATION. Figure 8 illustrates the network's architecture and the individual properties within each cohort. Note that McClelland's original network had 27 individuals encoded, while Figure 8 only encodes the properties of three individuals.

To illustrate how the network retrieves specific information, we will use Lance as an example. First, the name node "Lance" is activated by an external signal. The node then sends an inhibitory signal to all other name nodes, and an excitatory signal to the instance node for *Lance*. When the instance node receives enough stimulation, it sends an inhibitory signal to all other instance nodes, and an excitatory signal to the properties of Lance, specifically the nodes for "Jets", "20's", "J.H.", "Married", "Burglar", and the name node "Lance". These property nodes send out inhibitory signals to the other nodes within their cohorts and an excitatory signal back to the instance nodes to which they are connected (which means instance nodes other than *Lance* may be activated). Eventually, the network will settle into a state of equilibrium where the properties of Lance will be activated at a high level and all other properties will be relatively inhibited. This is an example of content-addressable memory.

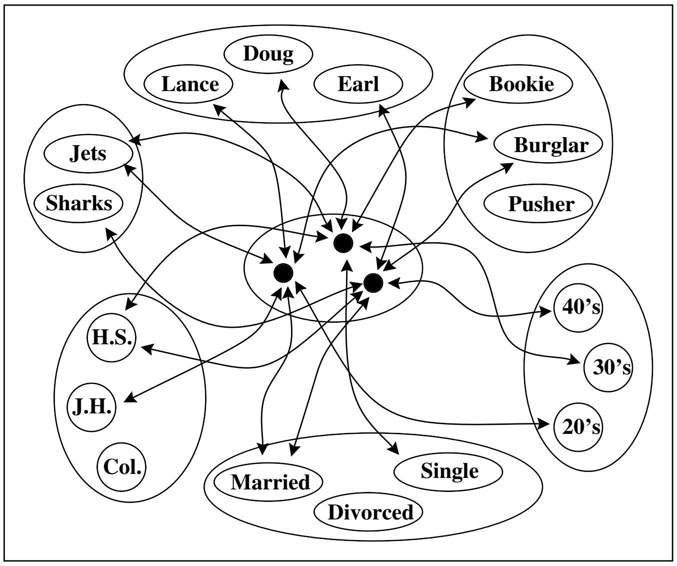

Figure 8: A much reduced version of McClelland's (1981) "Jets" and "Sharks" network for illustrative purposes. The solid nodes are "instance units" (one for each gang member) while the hollow nodes are "property units" that encode specific characteristics. Inhibitory connections are not shown.

To retrieve general information from the network, we can activate one of the other property nodes. For example, if we wished to find out the average characteristics of members in the Jets Gang, we would simply activate the "Jets" unit and look for the property nodes with the highest amount of activation in each cohort. It turns out that the average member of the Jets is in his 20's, has a Junior High education, is single, and is equally likely to be a pusher, a bookie, or a burglar. Furthermore, the network will also tell us who the gang members of the Jets are. This general information is not encoded specifically anywhere within the network; therefore, the model "has no explicit representation that the Jets tend to have these properties" ([66], p. 171) and yet this information is available.

Finally, the IAC model is able to handle incomplete or missing data and even perform when the network has been damaged. If the network is given incorrect information (e.g., inquire about who was a Shark, in their 20's, single, a burglar, and had a Junior High education) it will return with the best match. In this case, the network returns the individual Ken who fits all the characteristics except for education level [68]. Furthermore, if we sever the connection between the instance node *Lance* and the property "Burglar", the network is still able to return a value of "Burglar" when the property node "Lance" is activated even though there is no direct connection. In effect, Lance will activate other individuals who are similar to him, and thus the network "guesses" at Lance's profession by default assignment.

The IAC network illustrates quite effectively content-addressable memory (e.g., retrieval of the properties of Lance by supplying his name only), output availability (e.g., assessing the state of the network at the end of a cycle), iterative retrieval (e.g., finding the average property of a Jets member from all other possible properties), and graceful degradation (e.g., retrieval of information when connections are severed or incorrect information is given)—properties required in a model of human information processing [78]. This model also questions the classical view that we explicitly store generalizations.

4.2.1 A Possible Learning Mechanism for IAC Networks

Although IAC models are an important contribution to the connectionist's tool bag, the initial models still suffered from the inability to learn and, hence, rightly fall into the classification of Old Connectionism. The ability of the IAC networks to incorporate so many of the characteristics of human information processing, however, make it difficult to dismiss the architecture for lack of a learning mechanism. Consequently, a possible learning mechanism for the IAC networks is proposed here.

In devising a new learning mechanism, we would want to incorporate as many of the known neurophysiological properties (both theoretical and empirical) of learning as possible. The first modification would be to add a Hebbian-like learning mechanism to increase or decrease the weighted connections between nodes, so that those nodes that are active together become more strongly connected (either inhibitory or excitatory), and those nodes that are seldom active together weaken their connection [44]. The second modification would be to limit the maximum possible weight of any connection. The idea behind this restriction comes from Hull's [51] growth of habit strength and Minsky and Papert's[74] observation that network weights often grow without bound and there is no evidence that biological neural networks behave in this manner. A third property to be incorporated into a possible learning mechanism would be to prevent weights from shifting sign; that is, weights that are positive remain positive, and weights that are negative remain negative. Finally, a decay process should be added to the learning mechanism to account for the "use it or lose it" property evident in real neural circuits (e.g., [27]). Consequently, learning in an IAC network could be accomplished by adding the following equation to the mathematics of the architecture:

$$\begin{aligned} &If(w_{ij} > 0) \\ &\quad \Delta w_{ij} = \eta(wmax - w_{ij})a_i a_j - wdecay(w_{ij}). \\ &Otherwise \\ &\quad \Delta w_{ij} = \eta(wmin + w_{ij})a_i a_j - wdecay(w_{ij}). \end{aligned} \tag{11}$$

where $wmax \geq max$, $wmin \leq min$, and $wdecay \geq 0$ are parameters specific to the network weights, η is a learning parameter, and the unit activations a_i and a_j are assumed to fall into the range $[min, max]$ as before. What Equation 12 states is that the change in weight is equal to some proportion of the unrealized weight potential (cf., Hull's growth of habit strength) minus some decay process. Note that this equation guarantees that as the inactivity between nodes persists, weights will approach but never cross zero; in other words, weights that are inhibitory remain inhibitory, and weights that are excitatory remain excitatory. Thus, the network is an unsupervised learning algorithm based on self-organizing principles. A similar learning rule—although less encompassing—for IAC networks has been proposed and tested [11] to train a face recognition network.

Therefore, with these modifications, the IAC architecture could be said to bridge the gap between Old and New Connectionism. But, the IAC network is still somewhat limited in the knowledge it can represent; for example, while the architecture represents semantic knowledge quite well, the architecture probably is not suitable for controlling limb movement. Consequently, we need to explore other architectures and learning methods, and the best place to start is with the forerunners to the most common network and learning algorithm today.

4.3 Grossberg's Instars and Outstars

Many of the ideas commonly used in artificial neural networks today can be attributed to Stephen Grossberg [39]. One such contribution is the instar and outstar configurations, which were originally proposed as models of certain biological functions. Basically, instars are neurons fed by a set of inputs through synaptic weights, while outstars are neurons driving a set of weights. Instar neurons and outstar neurons are capable of being interconnected to form arbitrarily complex networks.

The purpose of instar neurons is to perform pattern recognition. Each instar is trained to respond to a specific input vector **X** and to no other. This is accomplished by adjusting the weight vector **W** to be like the input vector. The output of the instar is the sum of its weighted connections (see Equation 14). This calculation can be seen as the dot product of the input vector and the weight vector, which produces a measure of similarity for normalized vectors. Therefore, the neuron will respond most strongly to the pattern for which it was trained.

An instar is trained using the formula ,

$$w_i(t+1) = w_i(t) + \alpha[x_i - w_i(t)] \tag{12}$$

where,

$w_i(t)$ = the weight from input x_i

x_i = i^{th} input

α = training rate coefficient which should be set to 0.1 and then gradually reduced during the training process.

Once trained, the instar will respond optimally to the input vector **X**, and respond to other vectors that are similar to **X**. In fact, if you train it over a set of vectors representing normal variations of the desired vector, the instar develops the ability to respond to any member of that class.

The outstar works on a complementary basis to the instar. It produces a desired excitation pattern for other neurons whenever it fires. To train the outstar, its weights are adjusted to be like a desired target vector

$$w_i(t+1) = w_i(t) + \beta[y_i - w_i(t)] \tag{13}$$

where β is the training rate coefficient which should start at 1 be slowly reduced to 0 during training. Ideally, the outstar neuron should be trained on vectors that represent the normal variation of the desired vector. Thus, the output excitation pattern from the neuron represents a statistical measure of the training set and can converge to the ideal vector even if it has only seen distorted versions of the vector.

4.4 Grossberg's Adaptive Resonance Theory

It would be hard to mention Grossberg without making a least a brief mention about *Adaptive Resonance Theory* (ART). ART was initially introduced by Grossberg [40] as a theory of human information processing: it has since evolved into a series of real-time neural network models that perform supervised and unsupervised category learning, pattern classification, and prediction [12].

The simplest ART network is a vector classifier—it accepts as input a vector and classifies it into a category depending on the stored pattern it most closely resembles. Once a pattern is found, it is modified (trained) to resemble the input vector. If the input vector does not match any stored pattern within a certain tolerance, then a new category is created by storing a new pattern similar to the input vector. Consequently, no stored pattern is ever modified unless it matches the input vector within a certain tolerance. This means that an ART network has both plasticity and stability; new categories can be formed when the environment does not match any of the stored patterns, but the environment cannot change stored patterns unless they are sufficiently similar.

There are many different variations of ART available today. For example, ART1 performs unsupervised learning for binary input patterns, ART2 is modified to handle both analog and binary input patterns, and ART3 performs parallel searches of distributed recognition codes in a multilevel network hierarchy. ARTMAP combines two ART modules to perform supervised learning while fuzzy ARTMAP represents a synthesis of elements from neural networks, expert systems, and fuzzy logic [12]. Other systems have been developed to suit individual researcher's needs; for example, Hussain and Browse [54] developed ARTSTAR which uses a layer of INSTAR nodes to supervise and integrate multiple ART2 modules. The new architecture provides more robust classification performance by combining the output of several ART2 modules trained by supervision under different conditions.

4.5 Kohonen Networks

A Kohonen network [57] can be characterized as a self-organizing map used for pattern recognition. It differs from the generic PDP architecture in several ways (see Section 4.6). First, application of an input vector to the network will cause activation in all output neurons: the neuron with the highest value represents the classification. Second, the network is trained via a non-supervised learning technique. This poses a rather interesting problem. As the training is done with no target vector, it is impossible to tell *a priori* which output neuron will be associated with a given class of input vectors. Once training is completed, however, this mapping can easily be done by testing the network with the input vectors. A typical Kohonen network is illustrated in Figure 9.

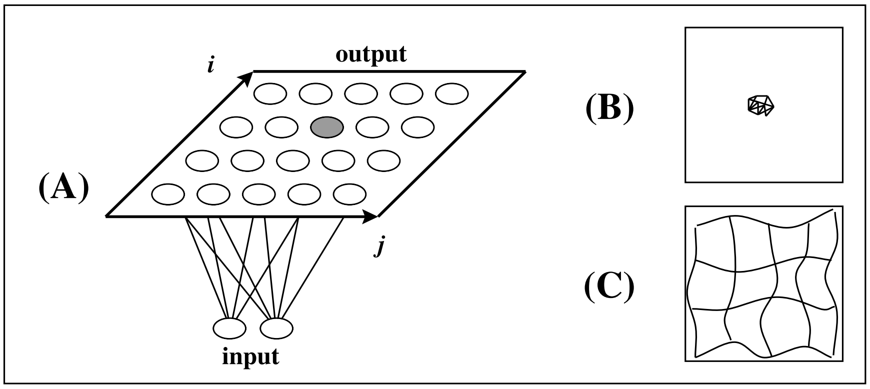

Figure 9: (A) A Kohonen network with two inputs mapping onto a 4 x 5 output field. (B) Randomized weight structure before training. (C) Typical weight structure following training.

The n connection weights into a neuron are treated as a vector in n-dimensional space. Before training, the vector is initialized with random values, and then the values are normalized to make the vector of unit length in weight space. The input vectors in the training set are likewise normalized.

The algorithm for training a Kohonen network can be summarized as follows:

1. Apply an input vector X to the network.
2. Calculate the distance D_j (in n dimensional space) between X and the weight vectors W_j of each neuron.
3. The neuron that has the weight vector closest to X is declared the winner. Use this weight vector W_c as the center of a group of weight vectors that lie within a distance of d from W_c
4. Train this group of vectors according to

$$W_j(t+1) = W_j(t) + \alpha[X - W_j(t)]$$

for all weight vectors within a distance d of W_c. Note the similarity between this equation and Equation 12.

5. Perform steps 1 through 4 for each input vector.

As training proceeds, the values of d and α are gradually reduced. It is recommended by Kohonen that α start near 1 and reduce to 0.1, whereas d can start as large the greatest distance between neurons and reduce to a single neuron. Furthermore, the number of training cycles should be approximately 500 times the number of output neurons to ensure statistical accuracy.

Because the input and weight vectors are normalized they can be viewed as points on the surface of a unit hypersphere. The training algorithm therefore adjusts the weight vectors surrounding the winning neuron to be more like the input vector. In other words, the algorithm tends to cluster weight vectors around the input vector.

Such adaptive units can be organized into a layer to produce a feature map. A feature map is a nonlinear method of representing the original signal space and resembles the topographic maps found in many areas of the brain [88]. The feature map is produced by the unsupervised training of the adaptive units which gradually develop into a spatially organized array of feature detectors whence the position of the excited units signal statistically important features of the input signal. Consequently, more frequently occurring stimuli will be represented by larger areas in the map than infrequently occurring stimuli.

Kohonen maps and unsupervised learning are but one way of training connectionist networks. But, if both the input and corresponding output patterns are known *a priori*, then supervised learning can be used. The most common supervised learning algorithm is the backpropagation algorithm used to train generic PDP networks.

4.6 The Generic PDP Network

As we saw in Section 3.5.2, an elementary α-perceptron has the *in principle* power to solve any pattern classification problem expressed in binary notation, whereas a network with less depth is limited in its computational power. This increase in computational ability derives from the fact that a multilayer network can theoretically carve a pattern space into an arbitrary number of decision regions [63]. Furthermore, it can be shown that such networks are also universal function approximators—that is, they are able to solve any function approximation problem to an arbitrary degree of precision [16][43][49]. These results are specific to the network architecture alone, and not to the learning rule used to train the networks.

Thus, we need to make a distinction between the network architecture and the learning rule. This move serves a dual purpose. First, it allows us to make claims about the computational power of networks regardless of the training procedure used. Second, we can evaluate the learning rule independent of the network architecture. The consequence of making this distinction between architecture and learning rule is that it allows us to (i) address concerns about the "biological plausibility" of certain learning algorithms (e.g., backpropagation) without compromising the interpretation and final results of the trained network, and (ii) determine if differences in network performance are due to architectural discrepancies or modifications of the learning algorithm. Therefore, we will first define the generic connectionist architecture, and then define the learning rule.

4.6.1 The Generic Connectionist Architecture

The building block for the generic connectionist architecture is the artificial neuron (see Figure 1). The functional properties of the artificial neuron mimic those of actual neurons; that is, the neuron receives and integrates information, processes this information, and transmits this new information (e.g., [27][60]). Mathematically, the input function to the neuron is expressed in Equation 14; net_{pj} is a linear function of the output signals, o_{pi}, from units feeding into j with weighted connections, w_{ij}, for pattern p.

$$net_{pj} = \sum_i w_{ij} o_{pi} \tag{14}$$

The output function of the neuron is a non-linear function of its input and is expressed in Equation 15. Note that the training rules typically used require that the activation function of the artificial neuron be differentiable and monotonic. Consequently, the most common function used is the logistic or sigmoid function which compresses the range of the net input so that the output signal lies between 0 and 1. This function allows the network to process large signals without saturation and small signals without excessive attenuation. Thus, in Equation 15, o_{pj} is the output of the neuron, net_j is the input, and θ_j is the "bias" of the unit which is similar in function to a threshold.

$$o_{pj} = f(net_{pj}) = (1 + e^{-net_{pj}+\theta_j})^{-1} \tag{15}$$

Units that use a function such as the logistic have an *order* of 1 [74] and are able to carve a pattern space into two distinct regions (see Figure 10). Thus, networks using this form of activation function can solve linearly inseparable problems without any hidden units. These networks have been termed *Integration Devices* by Ballard [6], and generic PDP nets by Anderson and Rosenfeld [3].

The power of these simple units emerges when they are connected together to form a network, or *multi-layer perceptron* (MLP). The most common MLP is a feed-forward architecture consisting of an input layer, an internal or hidden layer, and an output layer (see Figure 7B); such networks are often referred to as three-layer networks, although this nomenclature is not always agreed upon[10]. Units in one layer propagate their signals to units in the next layer through uni-directional, weighted connections. Normally, connections do not exist within layers, nor do they transcend more than one layer (i.e., from the input layer directly to the output layer); however, exceptions do exist.

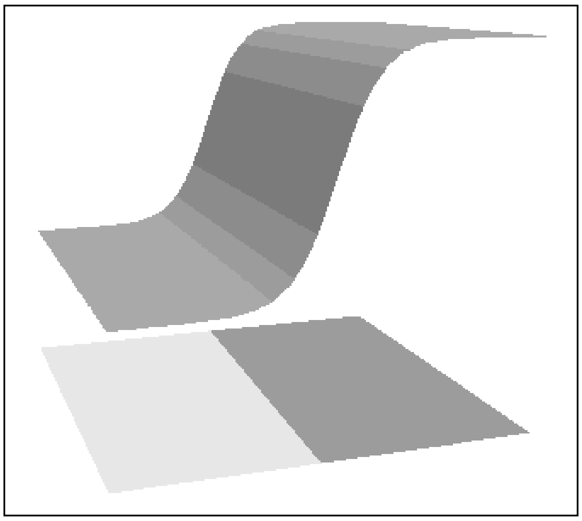

Figure 10: A monotonic activation function—such as the logistic—divides a pattern space into two distinct regions

Furthermore, it is assumed for simplicity that processing within the network occurs in discrete time intervals. It is further assumed that all processing is done in parallel; that is, all signals pass through the connections from one layer to the next at the same time and all units in a layer process their activations at the same time. Thus, the speed of processing—in terms of how long it takes the network to solve a problem—is directly proportional to the number of layers in the network, not the number of processing units. A three layer network with 5,000 units theoretically takes the same number of time steps to compute its function as a three layer network with five units (practically, this is not the case as networks are often modeled using serial computers). Consequently, parallel processing in neural networks is often hailed as a solution to the 100-step constraint[11] plaguing classical models.

The major advantage of multilayer networks over single layer networks is that they can theoretically carve a pattern space into an arbitrary number of decision regions and therefore solve any pattern classification problem [63], overcoming one of the limitations cited by Minsky and Papert [74]. Furthermore, it can be shown that such networks are also universal function approximators—that is, they are able to solve any function approximation problem to an arbitrary degree of precision [16][43][49]. It should be noted that

[10]For example, Wasserman [114] argues that since input units do not compute any function, they should not be counted as a layer; therefore, he calls these two-layer networks.

[11]The 100-step constraint is based on the processing speed of neurons [30]. Most complex behaviours occur in a few hundred milliseconds—this means entire behaviours are executed in less than a hundred time steps as opposed to the millions of time steps required by classical models.

although these proofs of function approximation are theoretically powerful, they are not necessarily tractable from a practical sense. This reason for this is two-fold: (i) in order to determine the requisite weights for the model, these proofs assume a highly representative sample of the range and domain of the function, and (ii) no *effective* procedure is typically given for arriving at the requisite set of weights.

Although MLP's have the required computational competence for cognitive scientists to find them interesting, their real allure lies in their ability to learn. Various training techniques have been proposed previously (e.g., Selfridge's supervised and genetic learning; Rosenblatt's reinforcement rule; Widrow and Hoff's Delta Rule), but they have all been limited to training only one layer of weights while keeping the other layers constant. What connectionism needed to move into the mainstream was a general learning rule for networks of arbitrary depth. In this way, a relatively simple network with a generic learning algorithm can be applied to a wide-range of different tasks.

Consequently, the next section will introduce the Generalized Delta Rule—also known as the standard backpropagation algorithm—and two variations on the rule that use nonmonotonic activation functions within the processing units. Furthermore, radial basis function networks will also be introduced as they use the same general framework but are differentiated by the net input function that they calculate and the activation function used within the processing units.

4.6.2 The Generalized Delta Rule

Papert's [81] likening of *Perceptrons* to the huntsman being sent out to bring back Snow White's heart is appropriate, for the huntsman did not return with the heart of Snow White, but the heart of a deer. Similarly, connectionism was not slain by *Perceptrons*, it was just quietly minding its time until its prince came. And, for connectionism, Prince Charming turned out to be the *Generalized Delta Rule* (GDR).

The GDR can be considered one of the most significant contributions to connectionist research: It has allowed the training of multilayer networks. In fact, the work of Rumelhart, Hinton, and Williams [92][93] is often cited as the catalyst for the strong resurgence of connectionist research in the latter half of the 1980's (e.g., [8][48]). As the name implies, the GDR is a generalization of the Widrow-Hoff Delta Rule for training networks of Adaline units [119]. The training procedure, however, is commonly referred to as backpropagation of error, or backpropagation (backprop) for short.

Although Rumelhart et al. are often credited with popularizing the GDR, the learning rule itself was derived previously on three separate independent occasions, first by Werbos in 1974[12], then by Parker in 1982[13] and finally by LeCun in 1986[14]. In fact, the GDR is simply a basic form of backpropagation. In its more general form [116], backpropagation contributes to the *prediction* and *control* of large systems (in terms of optimal planning and reinforcement learning), and not simply to supervised learning as is often assumed. Consequently, backpropagation can be applied to any differentiable, sparse, nonlinear system—it is not restricted to any specific form of MLP, nor is it restricted to artificial systems. The main advantage of backpropagation over traditional methods of error minimization is that it reduces the cost of computing derivatives by a factor of N, where N is the number of derivatives to be calculated. Furthermore, it allows higher degrees of nonlinearity and precision to be applied to problems.

Werbos [116] notes that since backpropagation is used in so many different applications, its actual definition has often become muddled and confused. Therefore, he offers these two standard definitions (p. 135, his italics):

1. Backpropagation is a procedure for *efficiently* calculating the derivatives of some output quantity of a nonlinear system, with respect to all inputs and parameters of that system, through calculations proceeding *backwards* from outputs to inputs. It permits "local" implementation on parallel hardware (or wetware).

[12] *Beyond regression: New tools for prediction and analysis in the behavioral sciences.* Masters thesis, Harvard University, Boston, MA.

[13] *Learning logic*, Invention Report S81-64, File 1, Office of Technology Licensing, Stanford University, Stanford, CA.

[14] Learning processes in an asymmetric threshold network. In E. Bienenstock, F. Fogelman Souli, & G. Weisbuch (Eds.), *Disordered systems and biological organization.* Berlin: Springer.

2. Backpropagation is any technique for adapting the weights of parameters of a nonlinear system by somehow using such derivatives or the equivalent.

What we are concerned with, however, is the special form of backpropagation for training neural networks. Werbos [116] calls this the *basic* form of backpropagation, although most researchers today simply refer to it as backprop. The GDR, as applied to neural networks, is a supervised learning algorithm (cf., Widrow & Hoff's delta rule—Equation 6) used to adjust the weights in an MLP in accordance with the *Principle of Minimal Disturbance*[15]. To begin, a training vector is presented to the network via the input units and the activations are then passed through weighted connections to the hidden units. The net input function to the hidden units is computed (Equation 14), the activation function is applied, then the output signal is generated (Equation 15) and propagated to the output units. The output units then use Equations 14 and 15 to produce a final output signal, o_{pj}, which is compared to the desired target output, t_{pj}. The total error, E, is defined in Equation 16, where p is an index over the patterns being presented, j is an index over output units, o is the actual state of the output, and t is the desired (target) state of the output.

$$E = \frac{1}{2} \sum_p \sum_j (t_{pj} - o_{pj})^2 \tag{16}$$

Learning is defined therefore as the minimization of this error term by gradient descent through an error surface in weight space. Gradient descent is described by the relation $\mathbf{W}_{k+1} = \mathbf{W}_k + \mu(-\nabla_k)$ where $\mathbf{W}_k$ is a weight vector, μ is a parameter that controls stability and rate of convergence and ∇_k is the value of the gradient of the sum squared error (SSE) surface at $\mathbf{W}_k$. Consequently, to begin gradient descent, an initial weight vector, $\mathbf{W}_0$, is defined and the gradient of the error surface at this point is measured. Weights are then altered in the direction opposite to the measured gradient, producing a new weight vector based upon the above relation. Every time this procedure is repeated with a newly calculated weight vector, $\mathbf{W}_k$, the SSE is caused to be reduced on average and moves towards a minimum. Because the true gradient is often impractical and inefficient to obtain, the instantaneous gradient is often computed based on the square of the instantaneous error. The instantaneous gradient is used because it is an unbiased estimate of the true gradient and is easily computed from single data samples [119].

Therefore, to minimize E by gradient descent, the partial derivative of E with respect to each weight within the network needs to be computed. For a given pattern, p, this partial derivative is computed in two passes: a forward pass using Equations 14 and 15, and a backward pass which propagates the derivatives back through the layers; hence, backpropagation of error. The backward pass begins by first differentiating Equation 16 which gives

$$\frac{\partial E_p}{\partial o_{pj}} = t_{pj} - o_{pj}$$

and then applying the chain rule to compute

$$\frac{\partial E_p}{\partial net_{pj}} = \frac{\partial Ep}{\partial o_{pj}} \cdot \frac{\partial o_{pj}}{\partial net_{pj}}.$$

The second term of the above equation is produced by differentiating Equation 15 which gives

$$\frac{\partial o_{pj}}{\partial net_{pj}} = f_j'(net_{pj}) = o_{pj}(1 - o_{pj}).$$

Therefore, the effect on the error due to a change in the total input to an output unit is known. But, as the total input is simply a linear function of the output from previous layers and the related connection weights,

[15] *Principle of Minimum Disturbance*: Adapt to reduce the output error for the current training pattern, with minimal disturbance to responses already learned [119](p. 719). It is noted that unless this principle is followed, it is difficult to store the required pattern responses simultaneously; hence, learning becomes problematic.

the effect on the error due to a change in the previous outputs and weights can be computed. For a weight w_{ij} from unit i to unit j, the derivative is

$$\frac{\partial E_p}{\partial w_{ij}} = \frac{\partial E_p}{\partial net_{pj}} \cdot o_{pi}$$

and the effect of all connections emanating from unit i is simply

$$\frac{\partial E_p}{\partial o_{pi}} = \sum \frac{\partial E_p}{\partial net_{pj}} \cdot w_{ij}.$$

Thus, two different error signals can be defined depending on if the unit is an output unit or an internal unit. For an output unit, the error signal is

$$\delta_{pj} = (t_{pj} - o_{pj}) f'_j(net_{pj}) \tag{17}$$

whereas for an internal unit, the error signal becomes

$$\delta_{pj} = f'(net_{pj}) \sum_k \delta_{pk} w_{kj}. \tag{18}$$

Hence, weights in the network are changed by

$$\Delta_p w_{ij} = \eta \delta_{pj} o_{pi} \tag{19}$$

where η is a learning parameter to scale the weight change, and Equation 17 is used for output units and Equation 18 for internal units. Finally, learning can be improved by adding a momentum term, α, which uses the previous weight changes to influence the current changes

$$\Delta_p w_{ij}(t) = \eta \delta_{pj} o_{pi}(t) + \alpha(\Delta_p w_{ij}(t-1)). \tag{20}$$

The weights of the network can be updated after every pattern presentation, or after the entire pattern set has been presented. Typically, training of the network continues until convergence is reached. For function approximation problems, convergence is measured by a sufficiently small total sum of squared errors (SSE) as computed by Equation 16. For pattern classification problems, convergence is attained when the network correctly classifies all input patterns. The performance of the network is normally assessed by the number of "sweeps" or "epochs" the network uses to solve the problem, where a sweep is defined by the single presentation of the entire training set.

Thus, the GDR overcomes the earlier limitations of Old Connectionism by allowing multilayer networks to be trained on any information processing problem. As Minsky and Papert [74] point out in their Epilogue to *Perceptrons*, however, many problems still exist with the GDR and the generic PDP architecture. One problem is that the GDR searches through an error space using gradient descent; although gradient descent on average moves towards a minimum it is not guaranteed to move towards a global minimum. In other words it is neither dependable nor efficient, though there are techniques for trying to improve this [116]. Another problem is that networks with only one layer of hidden units trained with the GDR still must violate the limited order constraint to solve linearly inseparable problems.

Although basic backpropagation is powerful enough to solve a wide variety of problems, much work is done on improving the performance of artificial neural networks especially in regards to three specific characteristics:

1. *Generalization*: the ability to predict data outside the original training set,

2. *Learning Speed*: increasing the convergence rate, especially for systems learning from real-time experience, and

3. *Fault Tolerance*: the ability to perform despite noise or breakage.

The general rule of thumb—at least from an engineering perspective—for the first two characteristics is to make the networks as simple as possible: use fewer connections and smaller weights. The third characteristic, on the other hand, is trickier to pin down. For example, the ability to perform despite noise can be seen as the ability to generalize; thus, smaller network structure would seem to be the answer. Performance despite breakage, however, requires larger network structures with some form of redundancy built in.

As mentioned earlier, one problem with generic PDP networks is that they are static; that is, previous inputs have no effect on new inputs (except during the training period). Consequently, the standard generic PDP architecture may not be appropriate for modeling some time dependent tasks, such as recognizing a pattern of sounds as forming a word. Therefore, the recurrent network architecture will be briefly introduced.

4.7 Recurrent Networks

A recurrent network is defined as one in which either the network's hidden unit activations or output values are fed back into the network as inputs. Figure 11 shows one possible structure for a recurrent network. In this network, inputs are received from an external source, passed to a hidden layer, and then on to the output layer. The signal from the output layer is passed to an external source, as well as back to a state layer which then acts as an input layer (along with the actual input layer) to the hidden layer on the next pass.

As the output of the network at time (t) is used along with a new input to compute the output of the network at time $(t+1)$, the response of the network is dynamic. That is, the network's response can be stable (successive iterations produce smaller and smaller output changes until the outputs become constant) or unstable (the network's outputs never cease changing). This stability issue proved a problem for early researchers, but Cohen and Grossberg [14] devised a theorem showing that at least a subset of recurrent networks were guaranteed to produce outputs with stable states. Stable networks are typified by weight matrices that are symmetrical along the main diagonal, with diagonal weights of zero (i.e., $w_{ij} = w_{ji}, w_{ii} = 0$).

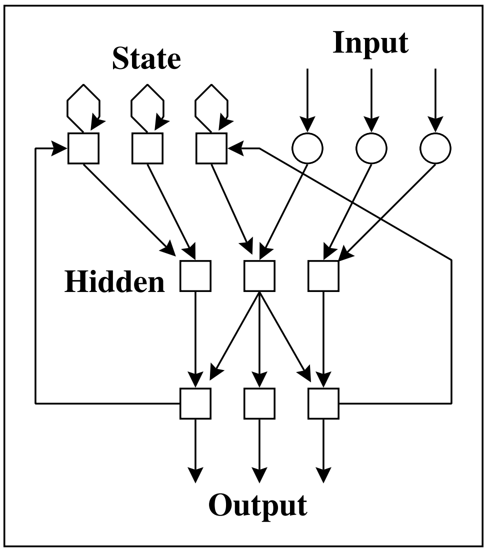

Figure 11: Recurrent network architecture. Connections from output to state units are one-for-one. Note that not all connections are shown.

Much of the early work on recurrent networks was pioneered by John Hopfield [47]. In fact, some have argued that it was because of Hopfield's stature as a well-known physicist that neural network research was made respectable again [3]. Hence, certain configurations of recurrent networks are referred to as Hopfield nets. One problem that plagued earlier versions of Hopfield networks, though, was that the networks tended to settle into local minimum instead of the global minimum. To combat this problem, one can change the weights statistically instead of deterministically. This technique is known as simulated annealing, and networks trained using this method are known as Boltzmann machines [46].

It has been shown that recurrent networks can simulate finite state automata [13] and that one can construct a second-order recurrent network such that internal deterministic finite-state automata state representations remain stable [79]. Furthermore, it has been proven that finite size recurrent networks can simulate any multi-stack Turing Machine in real time and non-deterministic rational nets can simulate non-deterministic Turing Machines [101].

Adding recurrent connections to the generic PDP architecture is but one way of improving the performance of such networks. Another way is to use different activation functions within the processing units.

Such an approach was taken by Dawson and Schopflocher [24] when developing the value unit architecture.

4.8 Value Unit Networks

Despite their immense theoretical power as universal function approximators and arbitrary pattern classifiers, networks trained with the GDR suffer from severe practical training problems. Networks are prone to local minima and notoriously slow if they do find a solution. One reason for this behaviour is the limitations imposed on the processing units by the GDR processing units must have a function that is both differentiable and monotonic. Consequently, the most commonly used activation function for processing units is the *logistic* (see Equation 15). This choice of activation function is normally motivated by engineering principles; for example, the logistic function is chosen because it fulfills the requirements of the learning rule, while similar functions—such as *tanh*—are chosen simply for their ability to improve performance in terms of learning speed over the logistic.

But, we could also adopt a different perspective and choose an activation function based upon neurophysiological evidence. Evidence from single-unit recordings (that is, record the output of the neuron with respect to its input) suggests that there are at least two functionally different types of neurons in the brain in regards to their output encodings [6]. This can be illustrated by comparing the recordings from neurons that function as a basic part of the oculomotor system to neurons in the visual areas of the cortex.

The first type of neurons—for example, those in the servo system controlling eye movement—have linear outputs whose *firing rate* is proportional to a scalar parameter such as the rate of eye rotation. These neurons could be characterized as *summation* or *integration* devices [6] and have the equivalent activation function as the logistic used in artificial neurons. The outputs of such neurons have two features—larger values mean more frequent pulses, and the output is one dimensional. From a physiological perspective, these neurons use *frequency* encoding. In other words, neurons using a monotonic activation function could be viewed as encoding *variables*.

In contrast, neurons in visual areas of cortex use fundamentally different encodings for their output. These neurons have multidimensional receptive fields[16]; that is, if the input stimulus is within a receptive field, the neuron will increase its firing rate, otherwise it remains at its baseline firing rate. The firing rate is specifically determined by the degree of match between the stimulus and receptive field—the stronger the match, the stronger the firing rate. From a physiological perspective, neurons with this type of firing pattern use *spatial* or *place* encoding. In other words, neurons using a nonmonotonic activation function could be viewed as encoding *values*. Consequently, Ballard [6] terms these neurons *value units*.

As "the value unit way of representing information seems to be a property of most cortical cells" [6], p. 68, the logical move—from a cognitive science perspective—would be to incorporate this type of activation function into a connectionist network.

4.8.1 The Value Unit Architecture

In considering a nonmonotonic activation function for artificial neurons, the most likely choice would be the *Gaussian*. Such an activation function is readily apparent not only within the cones of the eye [17], but also within the tuned neurons in the visual cortex [50]. From a computational perspective, the Gaussian

$$o_{pj} = G(net_{pj}) = e^{-\pi(net_{pj}-\mu_j)^2} \qquad (21)$$

where net_{pj} is the same as in Equation 14 and μ_j is the bias of the activation function, has the advantage of being able to carve a pattern space into three decision regions (see Figure 12) while still satisfying the GDR's requirement that the function be differentiable. Consequently, such an activation function could be said to be *limited* order 2; it is of limited order because the planes are restricted to parallel cuts in the pattern space.

[16]In this respect, a receptive field is defined in terms of *all* the neuron's inputs, including possible feedback connections from neurons in other parts of the cortex. This is in contrast to the normal interpretation of receptive field which limits itself to the inputs from a specific stimulus.

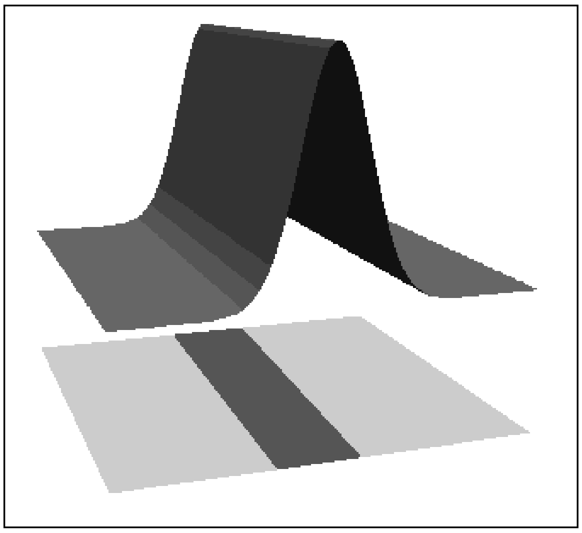

Figure 12: A non-monotonic activation function—such as the Gaussian—carves a pattern space into three regions

The nonmonotonicity of the activation function buys the value unit networks certain theoretical and practical advantages over standard integration device networks. For example, the fact that a single value unit can subdivide a pattern space into three regions by placing two parallel hyperplanes within the pattern space means that the processing power of the unit is increased. Whereas standard integration device networks require the same number of hidden units as the order of the problem (e.g., a 4-parity problem is order 4 and therefore requires four hidden units), networks with value units in both the hidden and output layers require considerably fewer. In fact, for problems such as parity which require parallel cuts of the pattern space, the number of hidden units needed is

$$(order\ div\ 2) - 1$$

where the operation *div* returns the quotient. Therefore, a solution to the XOR problem can be represented in a network without any hidden units, and a solution to the 4-parity problem can be represented in a network with only one hidden unit! Moreover, the added processing power of the value unit means that the limited order constraint need not be violated; that is, it is possible to solve the parity problem without any hidden unit connected to every input unit.

On the other hand, the nonmonotonicity of the activation function has the potential to limit the value unit architecture. For example, value units actually require hidden units to solve linearly separable problems [71]. Furthermore, because the value unit uses a nonmonotonic activation function, it is not uniquely invertible, and therefore it has been suggested that theoretically, value units are not suitable for function approximation [24]. On the other hand, the RBF unit which also uses a nonmonotonic activation function is routinely used for function approximation [75][64]. As will be shown, however, the basis underlying the RBF network is different than the value unit.[17]. In all other respects, however, the value unit architecture is the same as the generic PDP architecture, including its ability to be fully trained by a variation of the GDR.

4.8.2 Modifying the Generalized Delta Rule

Normally, replacing the $f'(net_{pj})$ in Equations 17 and 18 with the first derivative of the Gaussian causes the network to fall into a local minimum which asserts that some property of the pattern space p is not true, but fails to assert the some property of p is true. To avoid these local minima, Dawson and Schopflocher added a second term to the standard GDR's error function to produce a new cost function, C_p. The first component of Equation 22 is the standard cost function used in the backpropagation algorithm and measures the failure of the network to match the observed output o_{pj} with the target output t_{pj}. The second component of C_p measures the network's failure to set $net_{pj} = \mu_j$ (a component of the Gaussian) when the desired output is equal to 1: It essentially prevents the unit's activation from falling towards either negative or positive infinity.

$$C_p = \frac{1}{2}\sum_{j=1}^{n}(t_{pj} - o_{pj})^2 + \frac{1}{2}\sum_{j=1}^{n} o_{pj}(net_{pj} - \mu_j)^2 \tag{22}$$

The new learning rule is therefore based on changing the weights such that the cost function in Equation 22 is minimized. Consequently, the desired weight change for the connection originating from unit i and

[17] In fact, recent results have shown that, in practice, value units can perform some types of function approximation tasks[71]

terminating at unit j for given pattern p can be computed as

$$\Delta_p w_{ij} = \eta(\delta_{pj} - \epsilon_{pj}) I_{pi} \tag{23}$$

where $\delta_{pj} = (-t_{pj} - o_{pj})G'(net_{pj})$ is equivalent to δ_{pj} in Equation 19 with the exception that the derivative is of the Gaussian and not the logistic. The ϵ_{pj} term is equal to $T_{pj}(net_{pj} - \mu_j)$ and is the augmented error-minimization function from Equation 22. Similarly, the unit's bias term can be modified using the equation

$$\Delta_p \mu_j = -\eta(\delta_{pj} - \epsilon_{pj}) \tag{24}$$

by treating the parameter μ_j as a connection weight between output unit j and some other unit whose activation is always 1. Furthermore, it can be shown that, at the end of training, the error function minimized is equivalent to that of the GDR's. These modifications allow a network trained with the backpropagation algorithm to use non-monotonic activation functions.

4.8.3 Value Unit Performance

Value unit networks have been applied to a wide variety of pattern classification tasks, from "toy" problems [24] [25] [72], to diagnosing Alzheimer's patients from SPECT data [20], to identifying logical problems [9], to classifying mushrooms [22]. One of the surprising aspects of the value unit architecture is that, from an engineering perspective, they have been shown to converge faster and more reliably on linearly inseparable problems than the more traditional MLPs that use monotonic activation functions. Furthermore, value unit networks show better generalization, and better ability to be "scaled up" from toy problems.

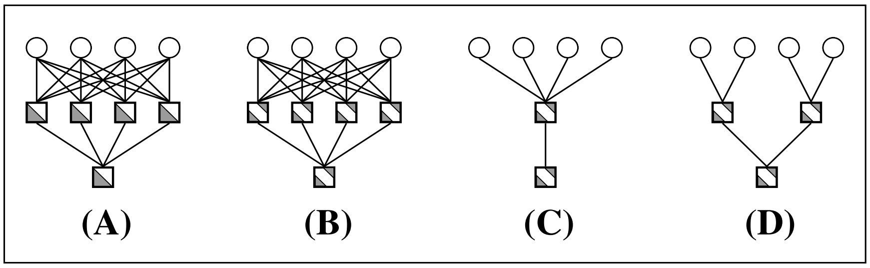

Figure 13: Network architectures for solving the 4-parity problem. Network (A) is an intergration device, while networks (B), (C), and (D) are value unit networks. Note that (A) and (B) have identical network structure (except for the processing unit), (C) requires only one hidden unit to solve the 4-parity problem, and (D) does not violate the limited order constraint.

To quickly show the processing power of value units over standard integration devices on linearly inseparable problems, a small experiment was conducted using four different network architectures to solve the 4-parity problem (see Figure 13). The first network was a standard integration device network with four hidden units (A). The second network had the same structure (i.e., 4 hidden units), but used value units instead (B). The third network (C) used the minimal value unit architecture of one hidden unit to solve the problem. Finally, the fourth network used two value units, but did not violate the limited order constraint (D); that is, no hidden unit was connected to every input unit. All networks were trained to a criterion of 0.0025 and had an upper limit of 10,000 sweeps imposed. Table 1 reports the results of the experiment; specifically the percentage of converged networks (total of 50 individual runs for each architecture), and the minimum, median, and maximum number of sweeps to reach convergence. As can be seen, all value unit networks clearly outperformed the integration device network.

The first thing to note is that all value unit networks, regardless of network topology, outperformed the integration device network both in terms of convergence rate and speed of convergence. In fact, when the

Table 1: Performance of Networks Trained on the Four Parity Problem

ARCHITECTURE	CONVERGENCE %	MIN	MED	MAX
A - Integration Device	24	1929	3607	9246
B - Value Unit	100	46	213	621
C - Minimum Value Unit	88	98	267	620
D - Limited Value Unit	64	133	134	209

integration device network did converge—which was only approximately one quarter of the time—it took an order of magnitude longer to do so.

Consequently, from a computational perspective, value unit networks show more competence than integration device networks in solving linearly inseparable problems such as parity. Furthermore, many different types of network topologies can be used to solve the 4-parity problem (see section 3.4). As it has been argued that networks are algorithms [94], this means that the different network topologies are different algorithmic descriptions for solving the 4-parity problem. Choosing the correct algorithm (network architecture) simply becomes a matter of comparing the computational competence between systems we are modeling. Finally, it should be noted that the fourth value unit network architecture (D) satisfies the Minsky and Papert's [74] limited order constraint, effectively addressing one of their concerns about neural networks.

4.9 The Radial Basis Function Network

One of the misconceptions surrounding the value unit architecture is based upon its use of a Gaussian activation function. This is because another network architecture, the *Radial Basis Function* (RBF) network [75] uses a similar activation function. That, however, is where the similarities end. The RBF network is a three-layer feedforward network that uses a linear transfer function for the output units and a nonlinear transfer function (normally the Gaussian) for the hidden units. The input layer simply consists of n units connected by weighted connections $\{\mu_{ij}\}$ to the hidden layer and a possible smoothing factor matrix $\{\sum_j\}$. A hidden unit can be described as representing a point $\mathbf{x}$ in n-dimensional pattern space. Consequently the net input to a hidden unit is a distance measure between some input, $\mathbf{x}_p$, presented at the input layer and the point represented by the hidden unit; that is, $net_j = \|\mathbf{x} - \mathbf{x}_p\|$. This means that the net input to a unit is a monotonic function as opposed to the nonmonotonic activation function of the value unit. The Gaussian is then applied to the net input to produce a *radial* function of the distance between each pattern vector and each hidden unit weight vector. Hence, a RBF unit carves a hypersphere within a pattern space whereas a value unit carves a hyperbar.

In general, an RBF network can be described as constructing global approximations to functions using combinations of basis functions centered around weight vectors. In fact, it has been shown that RBF networks are universal function approximators [38]. Practically, however, the approximated function must be smooth and piecewise continuous. Consequently, although RBF networks can be used for discrimination and classification tasks (see [64] for some examples), binary pattern classification functions that are not piecewise continuous (e.g., parity) pose problems for RBF networks[75]. Thus, RBF networks and value unit networks are not equivalent.

4.10 The Importance of New Connectionism

The major turning point in connectionist research occurred with the discovery of methods for training multilayer networks. With this discovery, connectionist models not only had the computational power to answer those questions interesting to cognitive science, but also had a method of *learning* how to answer those questions. Thus, there is an explicit distinction between network architectures and the learning rules used to train them within new connectionism.

By understanding the different types of architectures and learning rules, researchers are in a position to choose the appropriate type of network to solve specific problems. For example, if one wanted to solve a pattern recognition problem that was linearly separable, then an integration device network would be appropriate. If the problem was linearly inseparable, however, then the value unit architecture would be more appropriate.

It should be noted that the field of connectionism is ever-evolving, and new architectures and learning rules are being constantly developed. For example, McCaughan [65] has trained networks that use a sinusoidal activation function and has found that such networks can solve both linearly separable and inseparable problems with relative ease. In a different approach to connectionism, Zemel [122][123] has applied minimum description length analysis to connectionist networks in order to optimize their internal representations. Similarly, Bayesian theories are being incorporated in connectionism in order to understand aspects of rationality in human cognition [67] and to guide unsupervised learning of higher order structure in patterns [61]. As these new techniques are building upon the previous research presented in this paper, they will not be elaborated here. Instead, we will conclude with what we have learned about the history of connectionism, and its possible future directions.

5 Conclusions

Connectionism has a long and varied past—it has borrowed and incorporated ideas from philosophy, psychology, neuroscience, mathematics, and computing science. By studying the history of connectionism, we place ourselves in a knowledgeable position to support or deny claims about connectionism. For example, we now know that claims about connectionism merely being another form of associationism [32] are false. Furthermore,claims that connectionism may offer a Kuhnian-like paradigm shift for psychology [98] are not necessarily true either, especially when connectionism's rather long history is considered. On the other hand, we can support the claim that connectionist networks have the computational power to be a valuable tool within cognitive science.

We know where connectionism has come from, and what the current state of connectionism is—but, where should connectionism be headed? Within cognitive science, there have been recent calls for a third generation[18] of neural network models to be developed (e.g.,[35]). It has been argued that this third generation of networks should take the "neural" a little more seriously, and incorporate as many known neurobiological principles as possible. That is, these "neuromorphic" networks should transcend the simplified components, layered architectures, and limited scale of the first and second generation networks. Results [24][72] [21] have shown that such principles can be successfully applied to network architectures and learning rules. Furthermore, these neuromorphic networks often result in unanticipated improvements in performance and interpretability [9][23] over standard networks. Thus, we may have lots to learn by returning to the "neural" roots of connectionism.

Acknowledgments

The author would like to thank M. R. W. Dawson, A. Kingstone, C. D. Heth, D. B. McCaughan, P. Thagard, and two anonymous reviewers for helpful comments.

[18] In a very general sense, first generation networks fall within Old Connectionism while second generation networks fall within New Connectionism.

References

[1] K. Aizawa. Connectionism and artificial intelligence: History and philosophical interpretation. *Journal of Experimental Artificial Intelligence*, 4:295–313, 1992.

[2] J. A. Anderson, A. Pellionisz, and E. Rosenfeld, editors. *Neurocomputing 2*. MIT Press, Cambridge, MA, 1990.

[3] J. A. Anderson and E. Rosenfeld, editors. *Neurocomputing*. MIT Press, Cambridge, MA, 1988.

[4] M. A. Arbib, editor. *The Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge, MA, 1995.

[5] B. Aune. *Rationalism, Empiricism, and Pragmatism: An Introduction*. Random House, New York, 1970.

[6] D. H. Ballard. Cortical connections and parallel processing: Structure and function. *Behavioral and Brain Sciences*, 9:67–120, 1986.

[7] W. Bechtel. Contemporary connectionism: Are the new parallel distributed processing models cognitive or associationist? *Behaviorism*, 13:53–61, 1985.

[8] W. Bechtel and A. Abrahamsen. *Connectionism and the Mind: An Introduction to Parallel Processing in Networks*. Blackwell, Cambridge, MA, 1991.

[9] I. S. N. Berkeley, M. R. W. Dawson, D. A. Medler, D. P. Schopflocher, and L. Hornsby. Density plots of hidden value unit activations reveal interpretable bands. *Connection Science*, 7:167–186, 1995.

[10] T. G. Bever, J. A. Fodor, and M. Garrett. A formal limitation of associationism. In T. R. Dixon and D. L. Horton, editors, *Verbal Behavior and General Behavior Theory*, pages 582–585. Prentice-Hall, Englewood Cliffs, NJ, 1968.

[11] A. M. Burton. Learning new faces in an interactive activation and competition model. In V. Bruce and G. W. Humphreys, editors, *Object and Face Recognition*, pages 313–348. Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.

[12] G. A. Carpenter and S. Grossberg. Adaptive Resonance Theory (ART). In Arbib [4], pages 79–82.

[13] A. Cleermans, D. Servan-Schreiber, and J. L. McClelland. Finite state automata and simple recurrent networks. *Neural Computation*, 1:372–381, 1989.

[14] M. A. Cohen and S. Grossberg. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:815–826, 1983.

[15] A. Collins and E. E. Smith. *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*. Morgan Kaufmann, San Mateo, CA, 1988.

[16] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2:303–314, 1989.

[17] J. Davidoff. Color perception. In Arbib [4], pages 210–215.

[18] M. R. W. Dawson. The how and why of what went where in apparent motion: Modeling solutions to the motion correspondence problem. *Psychological Review*, 98:569–603, 1991.

[19] M. R. W. Dawson. *Understanding Cognitive Science*. Blackwell, Oxford, 1998.

[20] M. R. W. Dawson, A. Dobbs, H. R. Hooper, A. J. B. McEwan, J. Triscott, and J. Cooney. Artificial neural networks that use SPECT to identify patients with probable Alzheimers disease. *European Journal of Nuclear Medicine*, 21:1303–1311, 1994.

[21] M. R. W. Dawson, S. C. Kremer, and T. N. Gannon. Identifying the trigger features for hidden units in a pdp model of the early visual pathway. In Elio [28], pages 115–120.

[22] M. R. W. Dawson and D. A. Medler. Of mushrooms and machine learning: Identifying algorithms in a PDP network. *Canadian Artificial Intelligence*, 38:14–17, 1996.

[23] M. R. W. Dawson, D. A. Medler, and I. S. N. Berkeley. PDP networks can provide symbolic models that are not mere implementations of classical theories. *Philosophical Psychology*, 10:25–40, 1997.

[24] M. R. W. Dawson and D. P. Schopflocher. Modifying the generalized delta rule to train networks of non-monotonic processors for pattern classification. *Connection Science*, 4:19–31, 1992.

[25] M. R. W. Dawson, D. P. Schopflocher, J. Kidd, and K. S. Shamanski. Training networks of value units. In *Proceedings of the Ninth Canadian Artificial Intelligence Conference*, pages 244–250, 1992.

[26] K. Doya. Recurrent networks: Supervised learning. In Arbib [4], pages 796–800.

[27] Y. Dudai. *The Neurobiology of Memory.* Oxford University Press, Oxford, 1989.

[28] R. Elio, editor. *Proceedings of the Tenth Canadian Conference on Artificial Intelligence*, San Mateo, CA, 1994. Morgan Kaufmann.

[29] M. J. Farah. Neuropsychological inference with an interactive brain: A critique of the "locality" assumption. *Behavioral and Brain Sciences*, 17:43–104, 1994.

[30] J. A. Feldman and D. H. Ballard. Connectionist models and their properties. *Connection Science*, 6:205–254, 1982.

[31] D. Feldman-Stewart and D. J. K. Mewhort. Learning in small connectionist networks does not generalize to large networks. *Psychological Research*, 56:99–103, 1994.

[32] J. A. Fodor and Z. W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28:3–71, 1988.

[33] S. I. Gallant. *Neural Network Learning and Expert Systems.* MIT Press, Cambridge, MA, 1993.

[34] C. R. Gallistel. The replacement of general-purpose theories with adaptive specializations. In M. S. Gazzaniga, editor, *The Cognitive Neurosciences*, pages 1255–1267. MIT Press, Cambridge, MA, 1995.

[35] D. Gardner, editor. *The Neurobiology of Neural Networks.* MIT Press, Cambridge, MA, 1993.

[36] H. Gardner. *The Mind/'s New Science.* Basic Books, New York, 1985.

[37] J. George N. Reeke and G. M. Edelman. Real brains and artificial intelligence. In S. R. Graubard, editor, *The Artificial Intelligence Debate.* MIT Press, Cambridge, MA, 1988.

[38] F. Girosi and T. Poggio. Networks and the best approximation property. *Biological Cybernetics*, 63:169–176, 1990.

[39] S. Grossberg. Classical and instrumental learning by neural networks. *Progress in Theoretical Biology*, 3:51–141, 1974.

[40] S. Grossberg. Adaptive pattern classification and universal recoding: I. parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23:121–134, 1976.

[41] S. Grossberg. A theory of visual coding, memory, and development. In E. L. J. Leeuwnberg and H. F. J. M. Buffart, editors, *Formal Theories of Visual Perception*. Wiley, New York, 1978.

[42] S. J. Hanson and C. R. Olson. Neural networks and natural intelligence: Notes from Mudville. *Connection Science*, 3:332–335, 1991.

[43] E. J. Hartman, J. D. Keeler, and J. M. Kowalski. Layered neural networks with gaussian hidden units as universal approximations. *Neural Computation*, 2:210–215, 1990.

[44] D. O. Hebb. *The Organization of Behaviour*. John Wiley & Sons, New York, 1949.

[45] G. E. Hinton and J. A. Anderson, editors. *Parallel Models of Associative Memory*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1981.

[46] G. E. Hinton and T. J. Sejnowski. Learning and relearning in Boltzman machines. In Rumelhart et al. [96], pages 282–317.

[47] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79:2554–2558, 1982.

[48] T. Horgan and J. Tienson. *Connectionism and the Philosophy of Psychology*. MIT Press, Cambridge, MA, 1996.

[49] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[50] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurons in the cat's striate cortex. *Journal of Physiology (London)*, 148:574–591, 1959.

[51] C. L. Hull. *Principles of Behavior*. Appleton-Century-Crofts, New York, 1943.

[52] C. L. Hull and H. D. Baernstein. A mechanical parallel to the conditioned reflex. *Science*, 70:14–15, 1929.

[53] W. S. Hunter. A consideration of Lashley's theory of the equipotentiality of cerebral action. *Journal of General Psychology*, 3:455–468, 1930.

[54] T. S. Hussain and R. A. Browse. ARTSTAR: A supervised adaptive resonance classifier. In Elio [28], pages 121–130.

[55] W. James. *The Principles of Psychology*, volume 1. Dover, New York, 1890/1950.

[56] W. James. *The Principles of Psychology*, volume 2. Dover, New York, 1890/1950.

[57] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[58] K. Lashley. In search of the engram. *Symposia of the Society for Experimental Biology*, 4:454–482, 1950.

[59] J. Leiber. *An Invitation to Cognitive Science*. Basil Blackwell, Cambridge, MA, 1991.

[60] I. B. Levitan and L. K. Kaczmarek. *The Neuron*. Oxford University Press, Oxford, 1991.

[61] M. S. Lewicki and T. J. Sejnowski. Bayesian unsurvised learning of higher order structure. In *Advances in Neural Information Processing Systems*, volume 9, pages 529–535, 1997.

[62] P. H. Lindsay and D. A. Norman. *Human Information Processing*. Academic Press, New York, 1972.

[63] R. P. Lippman. An introduction of computing with neural nets. *IEEE ASSP Magazine*, April:4–22, 1987.

[64] D. Lowe. Radial basis function networks. In Arbib [4], pages 779–782.

[65] D. B. McCaughan. On the properties of periodic perceptrons. In *Proceedings of the 1997 International Conference on Neural Networks*, pages 188–193, 1997.

[66] J. L. McClelland. Retrieving general and specific information from stored knowledge of specifics. In *Proceedings of the Third Annual Meeting of the Cognitive Science Society*, pages 170–172, 1981.

[67] J. L. McClelland. Connectionist models and Bayesian inference. In M. Oaksford and N. Chater, editors, *Rational Models of Cognition*, chapter 2. Oxford University Press, Oxford, 1998.

[68] J. L. McClelland and D. E. Rumelhart, editors. *Explorations In Parallel Distributed Processing.* MIT Press, Cambridge, MA, 1988.

[69] J. L. McClelland, D. E. Rumelhart, and the PDP Research Group, editors. *Parallel Distributed Processing*, volume 2:Psychological and Biological Models. MIT Press, Cambridge, MA, 1986.

[70] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[71] D. A. Medler. *The Crossroads of Connectionism: Where Do We Go From Here?* PhD thesis, University of Alberta, 1998.

[72] D. A. Medler and M. R. W. Dawson. Training redundant artificial neural networks: Imposing biology on technology. *Psychological Research*, 57:54–62, 1994.

[73] D. A. Medler, M. R. W. Dawson, A. Kingstone, and E. Panasiuk. The locality assumption revisited: The relation between internal structure and behavioral dissocations. *Cognitive Science*, *under review*, 1998.

[74] M. Minsky and S. A. Papert. *Perceptrons: An Introduction to Computational Geometry.* MIT Press, Cambridge, MA, expanded edition, 1988/1969.

[75] J. Moody and C. J. Darken. Fast learning in networks of locally tuned processing units. *Neural Computation*, 1:281–294, 1989.

[76] M. C. Mozer, P. W. Halligan, and J. C. Marshall. The end of the line for a brain-damaged model of unilateral neglect. *Journal of Cognitive Neuroscience*, 9:171–190, 1997.

[77] A. Newell and H. A. Simon. Computer science as empirical inquiry: Symbols and search. *Communications of the ACM*, 19:113–126, 1976.

[78] D. A. Norman. Reflections on cognition and parallel distributed processing. In McClelland et al. [69], pages 531–546.

[79] C. W. Omlin and C. L. Giles. Constructing deterministic finite-state automata in recurrent neural networks. *Journal of the ACM*, 43:937–972, 1996.

[80] D. N. Osherson and H. Lasnik, editors. *Language: An Invitation to Cognitive Science*, volume 1. MIT Press, Cambridge, MA, 1990.

[81] S. Papert. One AI or many? *Daedalus*, 117:1–14, 1988.

[82] W. D. Pierce and W. F. Epling. *Behavior Analysis and Learning.* Prentice-Hall, Englewood Cliffs, NJ, 1995.

[83] D. C. Plaut and T. Shallice. Perseverative and semantic influences on visual object naming errors in optic aphasia: A connectionist account. *Journal of Cognitive Neuroscience*, 5:89–117, 1993.

[84] K. Plunkett. Connectionist approaches to language acquisition. In P. Fletcher and B. MacWhinney, editors, *The Handbook of Child Language*, pages 36–72. Blackwell, Oxford, 1995.

[85] M. I. Posner, editor. *Foundations of Cognitive Science*. MIT Press, Cambridge, MA, 1989.

[86] Z. W. Pylyshyn. *Computation and Cognition*. MIT Press, Cambridge, MA, 1984.

[87] R. A. Rescorla and A. R. Wagner. A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In A. H. Black and W. F. Prokasy, editors, *Classical Conditioning II: Current Research and Theory*, pages 64–69. Appleton-Century-Crofts, New York, 1972.

[88] H. Ritter. Self-organizing feature maps: Kohonen maps. In Arbib [4], pages 846–851.

[89] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[90] F. Rosenblatt. *Principles of Neurodynamics*. Spartan, Washington, DC, 1962.

[91] D. E. Rumelhart, G. E. Hinton, and J. L. McClelland. A general framework for parallel distributed processing. In Rumelhart et al. [96], pages 45–76.

[92] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In Rumelhart et al. [96], pages 318–362.

[93] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[94] D. E. Rumelhart and J. L. McClelland. Levels indeed! a response to Broadbent. *Journal of Experimental Psychology: General*, 114:193–197, 1985.

[95] D. E. Rumelhart and J. L. McClelland. On learning the past tense of English verbs. In McClelland et al. [69], pages 216–271.

[96] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, editors. *Parallel Distributed Processing*, volume 1:Foundations. MIT Press, Cambridge, MA, 1986.

[97] D. E. Rumelhart and D. A. Norman. Parallel associative models of memory: Where do they fit? In Hinton and Anderson [45].

[98] W. Schneider. Connectionism: Is it a paradigm shift for psychology? *Behavior Research Methods, Instruments, and Computers*, 19:73–83, 1987.

[99] O. G. Selfrdige and U. Neisser. Pattern recognition by machine. *Scientific American*, 203:60–67, 1960.

[100] O. G. Selfridge. Pandemonium: A paradigm for learning. In D. V. Blake and A. M. Uttley, editors, *Proceedings of the Symposium on Mechanisation of Thought Processes*, pages 511–529, London, 1959. H. M. Stationary Office.

[101] H. T. Siegelmann and E. D. Sontag. Analog computation via neural networks. *Theoretical Computing Science*, 131:331–360, 1994.

[102] B. F. Skinner. *The Behavior of Organisms*. Appleton-Century-Crofts, New York, 1938.

[103] H. Spencer. *The Principles of Psychology*, volume 1. D. Appleton and Company, New York, 3rd edition, 1855/1910.

[104] H. Spencer. *The Principles of Psychology*, volume 2. D. Appleton and Company, New York, 3rd edition, 1855/1910.

[105] N. A. Stillings, M. H. Feinstein, J. L. Garfield, E. L. Rissland, D. A. Rosenbaum, S. E. Weisler, and L. Baker-Ward. *Cognitive Science: An Introduction*. MIT Press, Cambridge, MA, 1987.

[106] M. C. Storrie-Lombardi and O. Lahav. Astronomy. In Arbib [4], pages 107–110.

[107] R. S. Sutton and A. G. Barto. Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review*, 88:135–171, 1981.

[108] P. Thagard. *Mind: Introduction to Cognitive Science*. MIT Press, Cambridge, MA, 1996.

[109] E. L. Thorndike. *The Fundamentals of Learning*. Teachers College, Columbia University, New York, 1932.

[110] E. L. Thorndike. *Selected Writings from a Connectionistś Psychology*. Greenwood Press, New York, 1949.

[111] A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, Series 2*, 42:230–265, 1936.

[112] S. F. Walker. A brief history of connectionism and its psychological implications. *AI & Society*, 4:17–38, 1990.

[113] J. Walter and K. Schulten. Implementation of self-organizing neural networks for visuo-motor control of an industrial robot. *IEEE Transactions on Neural Networks*, 4:86–95, 1993.

[114] P. D. Wasserman. *Neural Computing: Theory and Practice*. Van Nostrand Reinhold, New York, 1989.

[115] J. B. Watson. Psychology as the behaviorist views it. *Psychological Review*, 20:158–177, 1913.

[116] P. J. Werbos. Backpropagation: Basics and new developments. In Arbib [4], pages 134–139.

[117] H. White. Economic prediction using neural networks: The case of IBM daily stock returns. In *Proceedings of the IEEE International Conference on Neural Networks*, pages II–451–II–459, San Diego, 1988.

[118] B. Widrow and M. E. Hoff. Adaptive switching circuits. In *1960 IRE WESCON Convention Record*, pages 96–104, New York, 1960. IRE.

[119] B. Widrow and M. A. Lehr. Perceptrons, Adalines, and backpropagation. In Arbib [4], pages 719–724.

[120] N. Wiener. *Cybernetics*. John Wiley & Sons, New York, 1948.

[121] A. D. Zapranis and A. N. Refenes. Investment management: Tactical assest allocation. In Arbib [4], pages 491–495.

[122] R. S. Zemel. *A Minimum Description Length Framework for Unsupervised Learning*. PhD thesis, University of Toronto, 1993.

[123] R. S. Zemel. Minimum description length analysis. In Arbib [4], pages 572–575.