HISTORY AND PHILOSOPHY OF NEURAL NETWORKS

J. MARK BISHOP

Abstract. This chapter conceives the history of neural networks emerging from two millennia of attempts to rationalise and formalise the operation of mind. It begins with a brief review of early classical conceptions of the soul, seating the mind in the heart; then discusses the subsequent Cartesian split of mind and body, before moving to analyse in more depth the twentieth century hegemony identifying mind with brain; the identity that gave birth to the formal abstractions of brain and intelligence we know as 'neural networks'.

The chapter concludes by analysing this identity - of intelligence and mind with mere abstractions of neural behaviour - by reviewing various philosophical critiques of formal connectionist explanations of 'human understanding', 'mathematical insight' and 'consciousness'; critiques which, if correct, in an echo of Aristotelian insight, suggest that cognition may be more profitably understood not just as a result of [mere abstractions of] neural firings, but as a consequence of real, embodied neural behaviour, emerging in a brain, seated in a body, embedded in a culture and rooted in our world; the so called 4Es approach to cognitive science: the *Embodied*, *Embedded*, *Enactive*, and *Ecological* conceptions of mind.

Contents

Key words and phrases. Artificial Intelligence, Connectionism, Embodiment, Neural Networks, Mind.

1. Introduction: the body and the brain

What force directs these actions? Where lies the body-magic that brings forth my world? Where sits this mind? If science can fully respond to these questions then it might be possible to one day simulate the human ability to act mindfully, with intelligence, on computer.

For much of the twentieth century the dominant paradigm of intelligence seated the mind in the brain; thus, if computers can model the brain then, theory goes, it ought to be possible to program computers to act intelligently. In the latter part of the twentieth century this insight - that intelligence is grounded in the brain - fuelled an explosion of interest in computational "neural networks" : high fidelity accurate simulations of the brain (cf. 'computational neuroscience') or engineering approximations used to control intelligent machines (connectionism). However, the view that intelligence is rooted solely in the brain is a relatively modern one and one that, in recent years, is being challenged by embodied approaches to artificial intelligence; a perspective that, in turn, can be traced back to the classical era.

The classical view of the mind was encompassed by the earlier notion of soul, leading Aristotle (in the De Motu Animalium, 350 B.C.E.) to famously enquire "how the soul moves the body, and what is the origin of movement in a living creature". In stark contrast to modern notions identifying mind with brain, Aristotle posited "the heart is the seat of the senses" - the sensorium - and the controller of both voluntary and involuntary movement; unlike, say, Alcmaeon and Hippocrates, in Aristotle's metaphysics of movement and tripartite division of the soul (into the appetitive, sensitive and rational parts; localised respectively in the liver, heart, and brain) there was simply no place for the brain in the casual chain that ultimately gave rise to animal behaviour.

Such classical ideas had influence well into the renaissance period until, in 1649, Descartes laid the foundations of a new - dualist - division of body (res extensa) and the immaterial soul/mind (res cogitans); the mind interacting with the material brain [the organ which controlled the body] - at an interface famously located in the pineal gland. Around this time support for this view of the brain - as the organ that controlled behaviour - found empirical support in the work of a contemporary of Descartes, the English physician Thomas Willis, who in 1667 began to identify links between the physical structure of the brain and certain pathological behaviours (e.g. epilepsy and other convulsive diseases).

Emerging from what later became known as the British Empiricist school of philosophy, John Locke was perhaps the first philosopher to define the self through a 'continuity of consciousness':

> .. that conscious thinking thing, (whatever substance, made up of whether spiritual, or material, simple, or compounded, it matters not) which is sensible, or conscious of pleasure and pain, capable of happiness or misery, and so is concerned for itself, as far as that consciousness extend.

In addition, in his 1690 'essay concerning human understanding' Locke famously suggested that at birth the mind was to be considered a blank slate ('tabula rasa'):

> Let us then suppose the mind to be, as we say, white paper, void of all character, without any ideas. How comes it to be furnished? I answer, in one word, from experience.

Thus, contrary to prevailing Cartesian philosophy (which held the basic logical propositions are innate), Locke maintained that we are born without innate ideas, and that all 'knowledge' comes from 'experience'; that knowledge is causally dependent on experience and that knowledge is justified solely by experience. Subsequently building on these intuitions in a supplementary chapter of the fourth edition of his essay, in 1690 Locke introduced the notion of an 'Association of Ideas' to label the principle accounting for the mental peculiarities of individuals.

In the philosophy of mind a 'theory of mind' typically attempts to explain the nature of ideas, concepts and other mental content in terms of the 'cognitive states' of underlying 'cognitive processes'. A cognitive state can thus encompass knowledge, understanding, beliefs etc. In a 'representational theory of mind' the cognitive states are conceived in terms of relations to 'mental representations' (which have content). In this view the underlying cognitive processes are simply understood in terms of 'mental operations' on the mental representations. In Locke's 'associationist theory of mind' this *association of ideas* - or associationism as it later became known - suggested that the mind is organised, at least in part, by principles of association and that items that 'go together' in experience will 'go together' in thought; subsequently David Hume refined Locke's generic notion of 'going together by association' by reducing it to three core empirical principles: identity, contiguity in time and place, cause and/or effect.

The associated ideas ('representations') could be memories, ideas, images, thoughts etc., with complex ideas being constructed from 'simples' and simple ideas being derived from sensations. Such raw sensations/perceptions were not governed and defined by principles of association, but were externally caused by things 'outside the head', which for Hobbes, Hume and Locke meant objects of the world. Hence for some time associationism was closely linked with the broad British Empiricist movement in philosophy.

Building on earlier ideas from Galileo and Descartes, Locke's epistemology famously identified objects [of reality] by their primary and secondary qualities. Examples of primary qualities include: solidity; extension; figure; number; motion; rest etc. Clearly the primary qualities have a direct link to their bearer; a primary quality says something about its bearer. E.g. If an object instantiates the primary quality of rest, then the object [of which it is a property] must be at rest. In this manner Locke suggested such primary qualities are essential to their bearers and are intrinsic qualities of their bearer and hence are fundamentally independent of the perceiving mind.

Conversely, Locke hypothesised the 'secondary qualities' to be the properties [of objects] that produce sensations in observers; things such as colour, sound, taste, smell etc. In this conception secondary qualities are the powers of objects - by configurations of their primary qualities - to cause experience in subjects; the effect objects have on some people. For Locke primary qualities exist in the world but secondary qualities exist only in the mind of the perceiver; there is no blueness or sweetness in the world, only extension in motion.

Thus Locke's associationism entails (i) compounding processes, where complex items are formed from simple items; (ii) decompositional processes, where complex items are

broken down into their simple elements and (iii) sequencing processes, where associations follow one another (e.g. in time). In sequencing processes groups of items (e.g. memories) follow one another in one of two ways: (a) by intrinsic association, whereby some items have a natural connection (i.e. a connection independent of the observer) with each other. E.g. chilli pepper and 'hotness' and (b) by extrinsic association, whereby some items have an observer dependent connection (either voluntarily or by happen chance). E.g. If the first person one fell madly in love with had long red hair, then one might thereafter associate these qualities with beauty, sex and love.

In his essay Locke suggested that such extrinsic associations had three properties: (i) they are either voluntary or happen by chance; (ii) the strength of the 'impression' of ideas can reinforce the association (i.e. powerful ideas may be forever linked in the mind. Cf. perception of beauty) and (iii) some items will 'go together' more easily than others; thus some will find mathematical associations easy, some artistic etc. Thus, although at birth the mind is a blank slate empty of ideas, individuals may find some 'ideas' easier to associate than others. These 'ideas' are conceived as mental representations; in this way we entertain particular ideas when in particular mental states. Thus the Scottish philosopher, historian and economist David Hume famously suggested that mental processes are merely sequences of such associated mental ideas. But what are such mental 'ideas' actually 'about'?

To what do 'ideas' actually refer? This, of course, is the problem of intentionality: 'how do mental ideas connect with [become to be about] things in the world?' Hume's response was to suggest that mental ideas are fundamentally representations 'like images'; thus positing a pictorial resemblance between idea and world. Unfortunately Hume's pictorial mechanism is not without problems. Firstly, it is both too general (not all ideas are like pictures; cf. justice) and not general enough (consider a picture of Eiffel Tower; this image may look like the Eiffel Tower, from one perspective, but that image is not necessary to the notion of the Eiffel Tower; the tower that Gustave Eiffel built); secondly, it provides no account of mental reference (resemblance is not sufficient for representation; a cartoon may look more like its creator than its subject, but can still represent the subject); thirdly, it offers no account of truth and falsity. Contra Tractatus Wittgenstein [2.17] "What the picture must have in common with reality in order to be able to represent it after its manner - rightly or wrongly - is its form of representation", images are not propositions; images [in themselves] are neither true or false.

Furthermore, although Hume's associationism - where mental/cognitive processes are simply defined as associations between representations - can easily accommodate [folk] psychological explanations of mind, it fundamentally lacks a workable account of representational content. I.e. what is it about a 'representation of a dog' that constitutes it as representing a dog rather than a banana (or anything else), rather than not being representational at all? Even in the present era [according to the American philosopher Mark Bickhard] the problem of "accounting for representational content is the central issue in contemporary naturalism: it is the major remaining task facing a naturalistic conception of the world. Representational content is also the central barrier to contemporary cognitive science and artificial intelligence: it is not possible to understand representation in animals nor to construct machines with genuine representation given current (lack of) understanding of what representation is" [15].

The preliminary theoretical base for a neural conception of mind (and hence also for contemporary neural networks) was independently proposed by Alexander Bain [10] and William James [61]. Central to their work is the notion that both thoughts and body activity resulted from neuronal processes in the brain; for Bain every possible activity required the firing of a distinct set of neurons. At the time the scientific community was skeptical of Bain's ideas because they appeared to require an inordinate number of neural connections within the brain (albeit it is increasingly apparent that the brain is an exceedingly complex organ and that the same brain 'hardware' can process multiple problems and multiple inputs - see Section 10.4.3).

In some ways James's theory was similar to Bain's, however in James's model he suggested that memories and actions resulted from electrical currents flowing between neurons; this was perceived as a significant advantage as, by focusing on the flow of electrical currents between neurons, it did not require as many distinct neural groups to store memories or motivate action.

1.1. **William James and neural associationism.** In "Minds, Brains, Computers" Harnish centrally views William James "The Principles of Psychology" [61] as suggesting that thinking (indeed all aspects of conscious life) had the following key properties:

- Thinking is conscious.
- Thinking is open to introspective examination.
- Thinking is private; 'my thought belongs with my other thoughts and your thought belongs with your other thoughts'.
- Thinking 'flows like a stream': a metaphor that gives rise to the idea of the 'stream of consciousness'; a concept with significant resonance in twentieth century literature, most famously in the works of Virginia Woolf and James Joyce.
- Thinking is 'about something' (i.e. it is fundamentally "intentional").
- Thinking is an evolved function (i.e. it is not a 'gift from god').

For James the key question in psychology is 'how does the mind solve the problem of what to think next?' The answer James arrived at was that thinking fundamentally operates using two general 'principles of association' which James sought to explain on a quasi-neurological basis:

- **Principle 1**: *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; a mechanism remarkably similar to the learning scheme outlined in 1949 by the Canadian psychologist Donald Hebb* [52] *(see Section 3.1).*
- **Principle 2**: *The amount of activity at any given point in the brain cortex is the sum tendencies of all the other points that discharge into it - a similar mechanism to that described in 1943 by McCulloch & Pitts* [77] *in their mathematical model of neural firing (see Section 2.1) - such tendencies being proportionate:*
  - *to the number of times the excitement of each point may have accompanied the point in question;*
  - *to the intensity of the excitements;*
  - *to the absence of any rival point, functionally disconnected with the first, into which the discharges may have been diverted.*

Using these principles James offers the following explanation of three types of associative mental processes involved in spontaneous thought: unrestricted association; partial association and focussed association.

Total association: in which there is unrestricted association between previous events and/or arbitrary concepts. E.g. James famously describes 'The memory of a walk followed by a romantic dinner'.

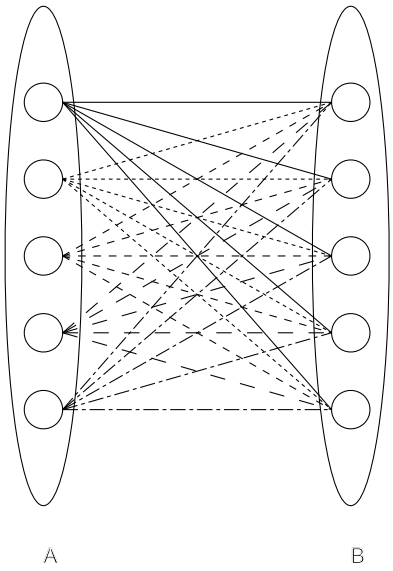

FIGURE 1. Total Association

Let **A** be the memory of the walk where the pattern of excitement distributed across neurons [a, b, c, d, e]. Let **B** be romantic thoughts where the pattern of excitement distributed across neurons [l, m, n, o, p]. To store this unrestricted total association James suggests the memories **A** and **B** must "vibrate in unison" hence '**A** must excite **B**' and '**B** must excite **A**', see Figure 1.

Partial association: in which only some of the past experiences have associated consequences; but 'why are some memories linked and not others?' James suggests that, "in no revival of past experience are all the items of thought equally operative in determining what the next thought shall be. Always some ingredient is prepotent over the rest." For James the item which is prepotent is the one which is most in our interest, "some one brain-process is always prepotent above its concomitants in arousing elsewhere".

James claims partial recall is the most common form of association-based recall and occurs when only some of past experiences have the required 'associated consequences'. To determine what experiences are [partially] associated together James outlines four principles; these are:

Habit: the more often something done the more likely it is to be associated.

Recency: more recent events are more likely to be recalled.

Vividness: the more intense an experience the more likely it is to be recalled.

Emotional congruity: similar emotion backgrounds are more likely to be associated together; hence feeling miserable makes it more difficult to recall times of joy.

At any time *the strongest principle of association* is that which pertains.

To see how these principles of association work imagine if one is thinking **A** and **A** is associated with **B**, (by say habit), then one will subsequently think **B** *unless a stronger principle applies.*

Focalised recall / association by similar sequences: in which not all memories are obviously associated ('go together'). As an example James considers: 'How might thoughts of a gas flame lead to thoughts of football?' James unfolds the process in this way: think of the gas-flame which conjures up visions of the moon [via a shared 'similarity of colour'; pale-whiteness]; this, in turn, is linked to the thoughts of football [via 'similarity of shape'; roundness]. So via focussed recall thought can move from **A** to **B** (from gas-flame to football) even though neither gas-flame nor football have any properties in common by themselves.

Contrasting the above examples of 'spontaneous thought' James also offers explanation for voluntary thought; a subject often thought problematic for associationists. James addresses the issue by offering an associationist account of the process of 'recall of a forgotten thing' (and claims a similar mechanism underlies means-end analysis).

To recall a forgotten item James envisages a situation where two groups of three memories [a,b,c] and [l,m,n] are fully associated (& interconnected) with the 'forgotten item' **Z**.

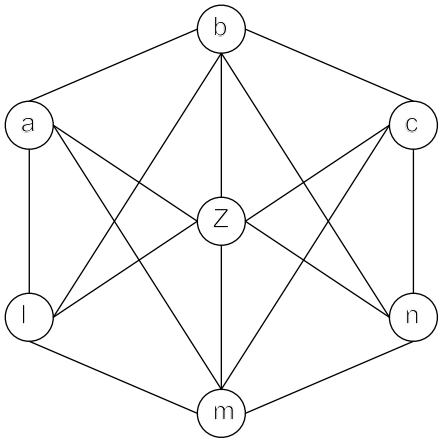

FIGURE 2. To recall a memory

By the process of total association activation of [a,b,c] will eventually propagate to activate [l,m,n] which together [a,b,c, l,m,n] will serve to activate the forgotten memory **Z**, see Figure 2.

Means-end analysis is performed in a similar way. If [a,b,c] is the goal states, its activation will eventually propagate to excite suggestion-states [l,m,n] which together will excite the solution **Z**, albeit in his analysis Harnish points out that James never

considers if all reasoning is performed via means end (or even if his version of means-end analysis will work in practice; for example, Harnish suggests that balancing a bank account would be difficult to describe in this way).

1.2. **The neuron: fine grain structure of the brain.** In "Aristotle's laptop: the discovery of our informational mind", Aleksander and Morton [9] suggest research into the building blocks of the brain effectively began with the discovery that all living things are made of cells. Thus, in 1838 the German botanist Matthias Schleiden first identified 'cell like' structures in plants; an idea later extended by physiologist Theodor Schwann who observed similar 'cellular structures' in the organs of animals; a discovery which, in turn, led to the development of 'cell theory': the idea that all living materials - whether in plants or animals - are composed of cells. In the context of the brain this posed the question: 'are the cells of the brain just a way of creating a fused functional structure or is there something special that the cells of the brain do (at the individual cell level) which is different to the function of other cells?'

Central to the search to resolve this issue was the ability to stain tissue such that it shows up better under microscopes (e.g. see Figure 3), research pioneered by the Bohemian anatomist Johannes Evangelista Purkinnje. Johannes worked at the University of Prague where he successfully made the then thinnest known slices of brain tissue; to make thes slices Purkinnje was amongst the first to use a device called a microtome; a tool to cut extremely thin slices of material.

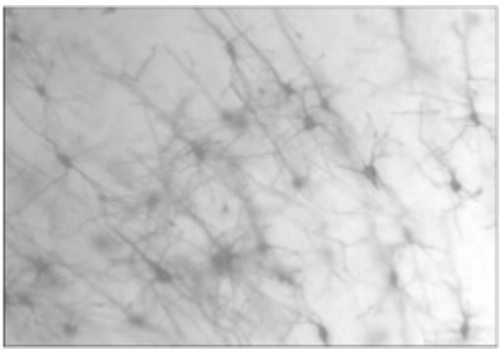

FIGURE 3. An example brain slice clearly showing neuron-bodies, the dendritic-tree and axon-fibres

In 1837 cell staining led to the discovery of the cellular structure of the cerebellum (a part of the brain at the back of the skull which is now known to be involved in the refinement of the control of the movement); in his research Purkinnje described and drew the structure of some large neurons (now known as Purkinje cells) labelled corpuscles and speculated that they had a role (collecting, generating and distributing) in intercellular communication.

Hence at this time there were two competing visions of brain operation - the reticularist and neuronist. The reticularist conception was that the cells of the brains formed a fused mechanism, functioning as a whole, whereas the neuronist conception postulated that brain function could be traced to the individual particular activity of brain cell

themselves : the 'neurons'. In particular, Otto Dieters & Camillio Golgi identified an 'axis cylinder' - the axon - as having a connective relationship to other cell dendrites.

A little later Camillio Golgi (1843-1926) successfully developed staining techniques using silver nitrate to identify nervous tissues; by so doing previously unseen fibres in the nervous tissue were brought to the fore and in so doing he clearly outlined the structure of neurons. These stains revealed beautiful network-like structures of many neurons. Around this time Santiago Ramon y Cajal (who eventually shared the Nobel prize with Golgi) went on to produce drawings of neural tissue which remain amongst the most impressive in the history of neuroscience. In so doing Cajal developed the fundamental insight that dendrites were conduits to the axons, which in turn connected to terminals on the dendrites of other neurons; and identified the neuron as a single element with, effectively, a one-way function.

Nonetheless, if neural doctrine sought to declare the neuron as a building brick of the brain, there still needs to be an interaction between the neurons to allow a network of neurons to develop a cooperative function. This idea shaped research in neurology during the last decade of the 19th century, when Sir Charles Sherrington conceived of the electrochemical neuron; outlining a way axons could electro-chemically 'make contact' with dendritic structure at a site we now identify as the 'synapse'.

In 1952 Alan Lloyd Hodgkin and Andrew Huxley suggested that in firing the axon initiates a charge polarity reversal which travels down the axon fibre to initiate neurotransmitter action at the synapse at the end of the fibre. In this way one neuron has the capacity to electro-chemically link with others and it became possible to think of the nervous system as a network of distinct cells that encode and transmit information throughout the brain. With this insight a modern conception identifying the mind with brain; and the brain with a large mass of distinct, but richly inter-linked neurons, arrived; the door was opened to *the mechanisation of thought.*

2. First steps towards modelling the brain

Early Cybernetic work in the field of machine intelligence began in various attempts to emulate neurological function and learning in animal brains and this history of Artificial Neural Networks (ANNs) begins with the excitement of the early pioneers in the 1940s and 1950s. I subsequently revisit some of the paths that led to neural networks falling out of favour in the late sixties, before highlighting a selection of key theoretical developments from the so called 'winter of connectionism' (throughout the 1970s and early 1980s) before foregrounding two novel insights in machine learning which overcame at least some of the limitations that Minsky and Papert had so devastatingly identified in 1969 [80]. The final section of this history reviews a small selection of some the more exciting recent developments in the field.

2.1. **The McCulloch-Pitts neuron model.**

The foundations of what has become known as 'neural computing', 'neural networks' or 'connectionism' were laid in 1943 by the neurophysiologist Warren McCulloch and the mathematician Walter Pitts [77]. In order to describe how the basic processing elements of the brain might function, McCulloch and Pitts showed how simple electrical circuits connecting groups of 'linear threshold functions' could compute various logical functions [77].

McCulloch and Pitts (MCP) realized that (a) neurons can receive positive or negative encouragement to fire, contingent upon the type of their 'synaptic connections' (excitatory or inhibitory) and (b) in firing the neuron has performed a 'computation' as, once the effect of the excitatory/inhibitory synapses are taken into account, it is possible to mathematically determine the net effect of incoming patterns of signal (i.e. the 'firings' from other neurons) on each neuron.

In their seminal paper "A logical calculus of ideas immanent in nervous activity" McCulloch and Pitts list five governing assumptions of neural operation, defining what has become known as the 'McCulloch-Pitts (MCP) neuron'. These are:

(1) the neuron is a binary device; it has two states of operation - firing or not firing.
(2) each neuron has a fixed threshold.
(3) the neuron can receive input from so called 'excitatory' synaptic inputs **all of equal weight**.
(4) the neuron can receive input from inhibitory synapses whose action is '**absolute**'; input on any one *inhibitory synapse* prevents the neuron from firing.
(5) there is a 'time quantum' over which to integrate all the synaptic inputs to the neuron.

Thus, neuronal operation proceeds as follows: during the time quantum, if there is no input on an inhibitory synapse, the number of excitatory inputs is summed and if this meets or exceeds the neural threshold, the neuron fires; if not, the neuron becomes inactive.

In this model McCulloch and Pitts provided a first (albeit very simplified) mathematical account of the chemical processes that define neuronal operation and in so doing they also realised that the mathematics that describe what the neuron is doing, exhibited exactly the same kind of logic that Shannon deployed for describing the behaviour of switching circuits: the calculus of propositions.

Hence, in their paper [77] McCulloch and Pitts demonstrated that if 'synapse polarity' is chosen appropriately, any single pattern of input can be 'recognised' by the neuron (i.e. cause the neuron to 'fire'). In other words, the central result of their paper is that logical truth tables of arbitrary complexity can be constructed by McCulloch-Pitts neurons. I.e. Any finite logical expression can be realised by a suitable network of McCulloch-Pitts neurons; indeed, some years later first Kleene [63] and subsequently Minsky [79] proved that rational-weighted recurrent neural networks with boolean activation functions are computationally equivalent to classical finite state automata.

The McCulloch-Pitts' result demonstrated that networks of artificial neurons could be mathematically specified which would perform 'computations' of immense complexity and power; in so doing, McCulloch and Pitts opened the door to a new form of computation; a form of computation based on the design of networks of artificial neurons rather than (Turing [125]) machine programming.

2.2. **The 'modern' McCulloch-Pitts neuron.** In the modern MCP threshold model adaptability comes from representing each synaptic junction by a variable (usually rational) valued weight W_i, indicating the degree to which the neuron should react to the ${}_i th$ particular input.

By convention positive weights represent excitatory synapses and negative inhibitory synapses; the neuron firing threshold is represented by a variable T. In modern use T is usually clamped to zero and a threshold implemented using a variable bias weight, b (a bias is simply a weight connected to an input clamped to $+1$).

In the MCP model the firing of the neuron is represented by the number $+1$ and not firing by 0. This is equivalent to the neuron representing a proposition as $TRUE$ or $FALSE$ for, as McCulloch and Pitts wrote [77], "in psychology .. the fundamental relations are those of two valued logic".

Activity at the i_{th} input to an n input neuron is represented by the symbol X_i and the effect of the i_{th} synapse by a weight W_i, hence the net effect of the i_{th} input on the i_{th} synapse on the MCP cell is thus $X_i \times W_i$. Thus the 'modern' MCP cell (see Figure 4) is denoted as firing if:

$$\sum_{i}^{n} X_i \times W_i + b \quad \geq \quad 0 \tag{1}$$

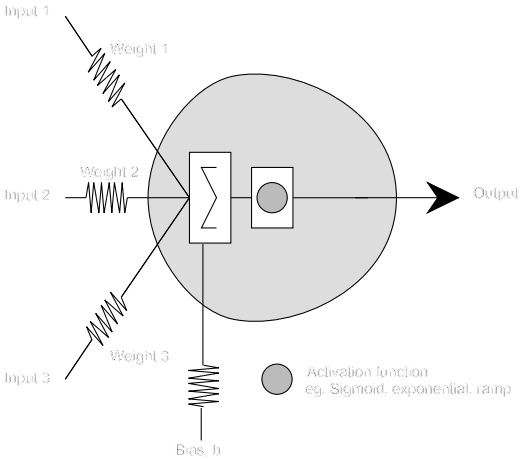

FIGURE 4. A schematic depicting the modern McCulloch-Pitts neuron model

NB. In a further modern generalisation of the MCP neuron, the MCP output is defined by an arbitrary function of the weighted sum of its input. This function is called the neuron's *activation function*. Example activation functions include: linear summation; the Heaviside (unit step or threshold) function, usually denoted by H, which is a discontinuous function whose value is 0 for negative argument and $+1$ for positive argument; and the oft deployed sigmoid function (which offers a continuous, differentiable, approximation to the Heaviside function).

2.3. **Artificial neural networks and neural computing.** An Artificial Neural Network, or ANN for short, is a connected group of MCP neurons. A *single layer neural network* only has one layer of MCP neurons between the input vector, X and the output vector O. In a *multi-layer neural network* the output of each neuron at the *output layer* will form one element of the output vector of the network; at a *hidden layer* (any layer of

neurons whose outputs do not constitute the output of the network) the neuron output will form the input to one or more neurons at a subsequent layer.

A *recurrent neural network* is a network where the output of one or more neurons is fed back to the input of neurons on that layer [or earlier layer(s)] in the architecture.

Thus neural computing fundamentally defines a mode of computing that seeks to include the style of computing used within the brain; it is a style of computing based on learning from experience as opposed to classical, tightly specified, algorithmic, methods (i.e. Aleksander and Morton [8] define Neural Computing as, *"the study of networks of adaptable nodes which, through a process of learning from task examples, store experiential knowledge and make it available for use."*).

2.4. **Computational and connectionist theories of mind.** The notion of mental representation is fundamental to any computational theory of mind, according to which cognitive states and processes are constituted by the instantiation, transformation and maintenance of information-bearing structures (i.e. 'representations'); these structures may be constituted 'symbolically' *in the Searlian sense* (i.e. as shorthand for a syntactic computational structure which does not imply meaning; thus the symbol structure 'CAT' may or may not denote 'a cute furry animal that purrs', but certainly does denote the structurally ordered collection of three formal shapes; the letters ''C', 'A' and 'T') or sub-symbolically (e.g. by vectors of real/rational numbers).

In a [Turing] computational theory of mind, the cognitive states are defined by [Turing] 'computational relations' to the underlying mental representations; and the cognitive processes are in turn thus simply defined by [Turing] computational operations on those mental representations. Strong computational theories of mind (cf. Searle [107]) further envisage the underlying mental representations to be fundamentally computational in character and hence fundamentally conceive the mind - encompassing all our thoughts, beliefs, intelligence, problem solving ability etc. - as but one form of suitably complex [Turing] machine computation.

In contrast the most generic connectionist theory of mind remains neutral on exactly what constitutes [connectionist] 'mental representations'; for example their structure may be fundamentally non-computational in character (i.e. a structure which cannot be generated via a classical Turing machine model of computation; perhaps, for example, a structure characterised by a chaotic dynamic system such as Hava Siegelmann's 'analogue shift map' [113]; a *super-Turing* mode of [information] processing) and/or the relation between cognitive state and mental representation is non-computational and/or the relationship between one cognitive state and the next is non-computational.

Conversely in a computational connectionist theory of mind, the cognitive states are simple computational relations to the underlying connectionist architecture and the underlying connectionist (mental) representations and the cognitive processes (the changes in the connectionist states) are fully Turing computational. E.g. in a classically connectionist neural network architecture, knowledge (e.g. the arity-zero predicate 'CAT()') can be considered as being represented either (a) by an activation value on a specific neuron or (b) distributed as a pattern of activation across groups of neurons.

However, uni-variate knowledge representation is limited to the representation of arity zero predicates and, as Dinsmore [38] and Fodor [41] suggest, this is too strong a restriction for representing the complexity of the real world and hence for modelling

general intelligent behaviour. However, more recent versions of connectionism (e.g. the NESTOR spiking neuron [Stochastic Diffusion] Neural Network [84] [86] can more easily represent knowledge of higher arity (see also Section 10.4.4).

2.5. **Connectionism as a special case of associationism.** By considering that the input nodes of an artificial neural network represent data from sensory transducers (the 'sensations'); the internal (hidden) network nodes encode ideas; the inter-node weights indicate strengths between these 'ideas' and the output nodes define behaviour, then we see a correspondence between connectionism and associationism (cf. Gardenfors [45] on the problem of modelling representations). However there are several ways in which this link might fail:

(1) In Associationist Networks the items that are associated together are specific idea(s) and in - local - connectionist networks (where one node encodes one item) this also holds; however, in ANNs that use a more distributed form of knowledge encoding, this correspondence no longer holds (see Figure 5).

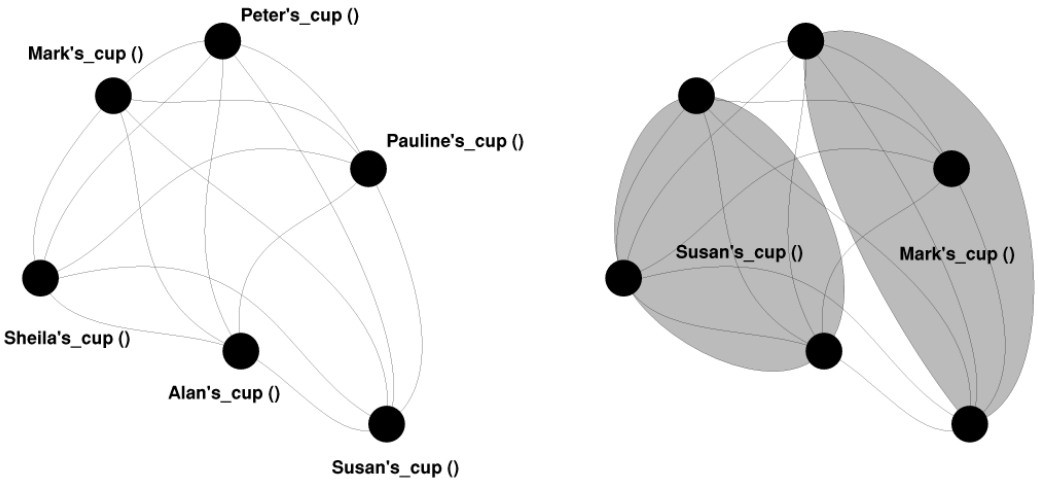

FIGURE 5. The link between Connectionism and Associationism

(2) Sensu stricto, in pure associationism 'ideas' are pure copies of 'sensations'. In any neural network where any of the weighted connections from the input to the first neuron(s) is not unity this is not true (as the input information will be scaled by the weight value).

2.6. **What functions can artificial neural networks perform?** Ever since McCulloch & Pitts first established that a network of MCP is Turing complete (i.e. it can *"compute only such numbers as a Turing Machine; second that each of the latter numbers can be computed by such a net"* [77]) researchers have sought to apply neural networks to mathematical tasks that, if carried out by humans, could be said to involve intelligence.

In contrast to conventional algorithmic approaches to intelligent problem solving by machine, ANNs are typically taught to solve particular problems by repeated exposure

to a set of input-output task exemplar vector pairs; the neural network *training set*. Subsequently, when deploying the artificial neural network on a real-world problem, the expectation is that a well designed neural network will *generalise* appropriately. I.e. The network should not only respond correctly to patterns that are in its training set (i.e. generate the specified output vector as defined by a relevant input-output vector pair from the training set), but also (a) generate appropriate responses to novel input vectors that lie *in-between* the input vector exemplars given in the training set (interpolation) and (b) generate appropriate output vectors for input vectors that *fall outside* the input space defined by the exemplar input-output vectors pairs in the training set (extrapolation).

ANNs have been used to model to three broad classes of complex function:

Vector association: an associative neural network is one that maps a given input vector to a particular output vector. Either a single layer or multi-layer neural network can be used to perform vector association as, given the networks *weight vector* (defining the synaptic weights of all the neurons in the network) and its *activation function(s)*, the operation of the network will be to map - *associate* - a given input vector to a particular output vector.

Vector classification: with judicious choice of weight vector, activation function(s) and a k element output vector, a network of MCP cells can be designed to map an input vector into one of k classes. Typically this is achieved by examining each element of the output vector $[1..k]$ with the class label being assigned to the element of the output vector that has [typically] the largest numeric value.

Function approximation: An n element input vector is mapped to an m element output vector, where the mapping defines some complex function. E.g. Given a three element real valued input vector [age, weight and height of a subject] a network could perhaps be trained to output a 1-D vector defining [an estimate] of their risk of suffering a medical condition such as diabetes or pulmonary embolism.

In addition to their ability to model complex functions neural networks are often claimed to have advantages over the alternative, algorithmic, problem solving approaches; these include:

- *offering a potential solution to the so called 100-step processing limit*: many complex problems involving intelligence take many millions of processing steps to solve on a powerful desktop computer however the brain operates with relatively slow processing units and it has been hypothesised that, in many tasks, there is time only for 100 neural processing operations to occur and it is not clear how to solve these problems in under this 100 step processing limit, without invoking the massive parallelism of ANNs in the brain;
- *being good at learning functions that do not obviously instantiate simple rules*: for example, learning so called 'intuitive' [sub-cognitive] non-verbal processes;
- *an ability to be resilient to minor changes (minor damage) to their structure*: that is, network performance typically degrades gracefully with small changes to the weight vector.

3. Learning: the optimisation of network structure

To enable a neural network to associate, classify or approximate a function appropriately we need to optimise its internal structure to minimise the network's *prediction error* between its output vector O and its target output vector T for a given input vector I. This the task of optimisation (or *learning*) in a neural network is the process whereby the network attempts to iteratively adjust all its weights and thresholds (its *structure*) so as to minimise the overall prediction error across all the pairs of input-output vectors in its *training set*. In a typical network there may be many millions of weight and threshold values and many thousands of input-output training vector pairs; a neural network *learning rule* is thus simply a - typically iterative - procedure for automatically calculating these values - there will usually be far too many to calculate by hand.

3.1. **Hebbian learning.** The publication of McCulloch and Pitts foundational work defining a simple mathematical model of a neuron was followed in 1949 by the publication of influential ideas on neural network learning by the psychologist Donald Hebb. In his monograph, 'The Organization of Behavior', Hebb suggested that 'neural pathways' [between simple cells/neurons] are strengthened each time that they are used and claimed the insight fundamental to understanding the ways in which humans learn; the underlying idea being to use correlations in the signalling activity of connected cells to cause an increase or decrease in connection strengths between them. This basic process has since become known as 'Hebbian learning' [52].

> Let us assume that the persistence or repetition of a reverberatory activity (or "trace") tends to induce lasting cellular changes that add to its stability.
>
> .. when an axon of cell **A** is near enough to excite cell **B** and repeatedly or persistently takes part in firing it, some growth process or metabolic changes take place in one or both cells such that **A**'s efficiency as one of the cells firing **B**, is increased.

I.e. When two neurons are simultaneously excited then the strength of the connection between them should be increased; as Fodor and Pylyshyn later remarked [41]:

> This should bring to mind the old Associationist principle that the strength of association between 'Ideas' is a function of the frequency with which they are paired 'in experience' and the Learning Theoretic idea that the strength of a stimulus-response connection is a function of the frequency with which the response is rewarded in the presence of the stimulus.

In Hebbian learning the task is to learn to associate a specified output vector **B** with an input vector **A**, the two vectors being fully connected by channels with interconnection weights W_{ij}. The learning scheme is concerned with the automatic evolution of connection weights, such that vector **A** forces the generation of vector **B**. The principle Hebb outlined was not sufficiently defined to build a complete model of fully autonomous weight learning, but a number of simple variants can trace their ancestry back to Hebb. One of the simplest Hebb variant is to "adjust the strength of the connection between units A_i, B_j in proportion to the product of their simultaneous activation". I.e. The change in weight connecting input I_i and output O_j is proportional (via a learning rate eta, η) to the product of their simultaneous activations:

$$\Delta W_{ij} \quad = \quad \eta I_i O_j \tag{2}$$

A neural network utilising such a rule has various interesting properties. One is that it does not require a perfect copy of the input to produce the required output, though the replicated response will be weaker (less accurate) as the patterns diverge. Similar effects are produced if the network is damaged, either by removing neurons or inter-connections.

Using the basic Hebbian rule, it is possible to modify connection weights automatically to generate *any single* linear mapping between the input and output vector. This is accomplished by making successive small modifications to the connection weights between each pair of units A_i, B_j, with the size of the weight change, the learning rate, set as a parameter of the learning process. If the weight changes are small enough, the Hebbian associator will eventually arrive at a set of weights that correctly generate the desired output vector when presented with the given input vector.

Hebbian learning demonstrated that it is possible to train a pattern associator to adjust its connection weights to form a desired mapping between input and output. However, in its most basic formulation the learning system is *unstable* and needs external regulation to stop weight adjustment once the desired output is reached. Kohonen [65] suggests a modification of the basic Hebb rule, introducing a simple forgetting factor for synaptic (weight) adjustment, to alleviate this problem:

$$\Delta w_{ij}(n) \quad = \quad \eta y_k(n) x_j(n) - \alpha y_k(n) w_{ij}(n) \tag{3}$$

3.2. **Rosenblatt's perception.** A figure, later to emerge as a controversial force in nascent field of connectionism, was Marvin Minsky. In 1954 Minsky published his doctoral dissertation on Neural Networks [78]. Four years later, a former classmate of Minsky's, Frank Rosenblatt, published a first description of his famous neuron-like computational element, *The Perceptron* [101], which uses both the fixed and learning versions of the McCulloch and Pitts neurons. In Rosenblatt's model the synapses became variable multipliers of the input which are systematically adjusted to remove error. At the time it could be clearly seen that there was great excitement in the field, with Rosenblatt claiming that [in the Class C Perceptron] "*for the first time we have a machine capable of having original ideas*" [101].

A few years later Rosenblatt's research led to the development of the *Perceptron Convergence Procedure (PCP)*, as outlined in his 1962 monograph, 'The Principles of Neurodynamics' [102]. The PCP was perceived as an important advance on the Hebb Rule for weight learning in complex networks; indeed it was claimed that [in the Perceptron] the 'rules that drive the brain mechanisms' are being uncovered' and the device notably garnered much press attention for having this 'brain-like' structure.

The full ('classical') Perceptron model can be divided into three layers (see Figure 6):

- The first layer of units defines the retina of the perceptron; this is comprised of a regular array of 'Sensory-Units' (or S-Units).
- The second layer of the perceptron is comprised of 'Association-Units' (or A-Units). The input to each A-Unit is the weighted sum of the output of a randomly selected set of S-Units. These weights are not learnt and hence do not change.

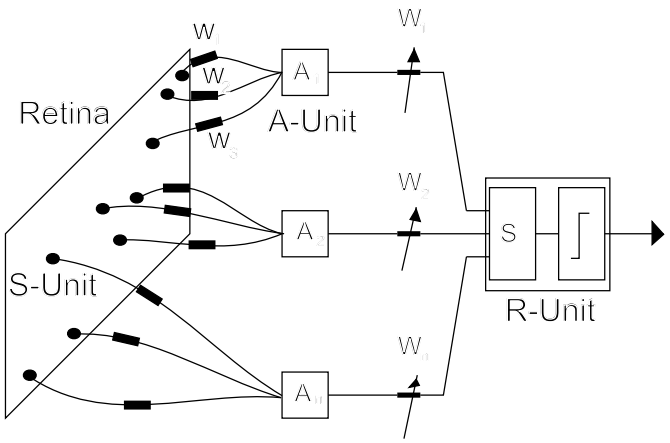

FIGURE 6. The structure of Rosenblatt's Perceptron

Because the weights are fixed, A-Units only respond selectively to particular patterns on the retina, effectively functioning as local feature detectors.

- The third layer of the perceptron consists of 'Response-Units' (or R-Units). Each R-Unit maintains a set of variable weighted connection to a specific set of A-Units. An R-Unit outputs [+1] if the sum of its weighted input is greater than a threshold T, [-1] otherwise. In some perceptron models, an active R-Unit will inhibit all A-Units not in its input set.

3.2.1. *Rosenblatt's 'Perceptron Convergence Procedure'.* In the perceptron convergence procedure Rosenblatt established that if the perceptron response to the current input vector is correct, then no change should be made to the weights to the R-Units; conversely, if the response of an R-Unit is incorrect then it is necessary to either:

- decrement all *active* weights - where an active weight is any weight connected to a non-zero input - if the R-Unit fires when it is not meant to **and** increase the threshold;
- increment active weights **and** decrement the threshold, if the R-Unit does not fire when it should.

3.2.2. *Rosenblatt's 'Perceptron Convergence Theorem'.* Simply stated Rosenblatt's - *fixed increment* - Perceptron Convergence Theorem asserts that the above procedure is *guaranteed* to find a set of weights to perform a specified mapping on a single layer perceptron *if such a set of weights exist* - e.g. if the problem is 'linearly separable'.

3.3. **The Widrow-Hoff (or 'simple delta') learning rule.** In 1960, Bernard Widrow and Marcian Hoff of Stanford developed an adaptive learning model called ADALINE (the ADAptive LInear NEuron - later to be renamed ADALINE: the 'ADAptive LINear Element') and subsequently refined as MADALINE (the Multiple ADAptive LINear Elements). In the ADALINE learning is effectively applied to a linear version of the McCulloch-Pitts neuron (with a simple Heaviside [or threshold] function, H, applied to define a binary output as required).

The ADELINE neuron consists of a set of weighted inputs W, a bias term θ and a summation element. The output of ADELINE is typically [a thresholded version of] this linear weighted sum of its inputs:

$$y = H(\sum_{i=1}^{n} I_i W_i + \theta) \tag{4}$$

Given an input vector I, an output vector O, a target vector T and a weight matrix W, ADELINE weight update is defined by a *simple 'delta' rule*:

$$\Delta W_{ji} = \eta(T_j - O_j)I_i \tag{5}$$

$$= \eta\delta_j I_i \tag{6}$$

Where W_{ji} is the change to be made to the weight linking the i_{th} input to the j_{th} output unit (given the current input/output training vector pair); η is defined as the learning rate constant and δj can be considered as an error term, δ, describing the difference between the desired output and the actual output.

ADALINE was originally developed to classify binary patterns; a subsequent development, MADALINE, was famously used as an adaptive filter designed to help eliminate echo on phone lines, and hence offers perhaps the first example of an Artificial Neural Network being applied to a 'real-world' engineering problem.

4. The fall and rise of connectionism

At the end of the sixties - only a few years after the pioneering work of McCulloch & Pitts, Rosenblatt and Widrow & Hoff - the iconoclastic pair of Minsky and Papert virtually single-handedly halted research in this nascent field of neural computing, with the publication of their 1969 critique 'Perceptrons' [80]. This work highlighted certain limitations to Rosenblatt's single-layer perceptron (SLP); in particular, they showed that SLPs are unable to calculate non-linearly separable functions such as the mathematical function of 'parity' or the topological function of 'connectedness'.

With the publication of 'Perceptrons' and early successes in the field of symbolic artificial intelligence - the approach favoured by Minsky and Papert - it rapidly became extremely unpopular to continue research into learning neural networks; the so-called 'winter of connectionism' had arrived. Furthermore, underlying the elegance of the Minsky and Papert critique there are stories of bad blood between Minsky and Rosenblatt, leading commentators such as Aleksander and Morton [9] to ponder, "Was it the fury of colleagues or a genuine desire for a truly scientific assessment that caused Seymour Papert and Marvin Minsky at MIT to launch an explicit attack on Perceptron-like systems?" Certainly Minsky and Rosenblatt had been to the same school in the Bronx and there were persistent rumours of very personal rivalries at play.

4.1. **The rise and rise of 'symbolic' artificial intelligence.**

If the overarching methodology of the first attempt to build machines that can 'think for themselves' had been to develop 'models of the brain', then the first roots of an alternative approach can be traced back to the 'Dartmouth Conference' of 1956, where the term 'Artificial Intelligence (AI)' was first coined. McCorduck gives an account of the Dartmouth Conference

in the book 'Machines who think [76], which offers a personalised history of AI based on interviews with key researchers. McCorduck observes that the Dartmouth Conference successfully gathered together all the major participants in AI at the time (a list which included Marvin Minsky, Nathaniel Rochester, John McCarth, Claude Shannon, Ray Solomonoff, Oliver Selfridge, Trenchard More, Arthur Samuel, Herbert A. Simon, and Allen Newell) for a month long 'brainstorming session':

> We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer [75].

Although the Dartmouth Conference did not achieve its main aim of facilitating a 'genuine pooling' of information and co-operation, and hence to John McCarthy (one of the key organisers) was not a complete success, it did offer the opportunity for two relatively unknown scientists, Allen Newell and Herb Simon to present their work.

Herbert Alexander Simon was an American polymath later to win the 1978 Nobel Prize in Economics for his pioneering research into decision-making processes within economic organisations and Allen Newell was a researcher in computer science and cognitive psychology at the RAND Corporation. Together with the RAND systems programmer John Clifford Shaw they developed the first 'classical' (these days occasionally more derisively termed 'good old fashioned') artificial intelligence program called the 'Logic Theorist'. This was a program encapsulating 'reasoning as search', 'ad hoc rules of thumb' called heuristics (borrowing a term from George Pólya) and 'list processing' (a computer data processing technique pioneered by Shaw).

Thus in operation the Logic Theorist explored a search tree, wherein the root node was an initial hypothesis and each of the branches followed a logical deduction (made using the rules of logic) from a node. If a theorem can be proved then it is clear that the goal state (i.e. the proposition to be proved) must lie somewhere in the tree, with the sequence of branches leading to it constituting the proof. As search trees of this form are prone to expand exponentially some means of pruning is required, or combinatorial explosion of branches will eventually make searching the tree intractable. In the Logic Theorist such branch trimming - of 'paths unlikely to reach the goal state' - was carried out by rote application of simple rules of thumb (heuristics).

In this manner the Logic Theorist was able to prove 38 of the first 52 theorems from Whitehead and Russell's Principia Mathematica [132]. Subsequently Newell and Simon refined the Logic Theorist into a system they called the 'General Problem Solver' [89]. And it was from these roots - the seeds of which being so fruitfully first sown at the Dartmouth conference - that an alternative style of artificial intelligence was ushered onto the scientific stage. In contrast to connectionism and machine learning this was to be a fundamentally new approach, explicitly defined by 'symbol processing' and 'heuristic

search'; as Dreyfus and Dreyfus later ironically observed [39], 'making a mind versus modelling the brain: artificial intelligence back at a branch point'..

4.2. **The rebirth of connectionism.** Throughout the 1970s and early 1980s, neural network research was at a low ebb with few people still active in the field; nonetheless important advances in the field continued to be made: Igor Aleksander refined Bledsoe and Browning's work on pattern recognition and 'reading by machine' [23] into a new 'weightless' (or n-tuple) neural network paradigm [5] (discussed in Section 4.3) and the continuing work of scientists such as Hopfield (see Section 5), Grossberg (see Section 6) and Kohonen (see Section 7) was influential in building momentum towards the eventual resurgence of interest in connectionism in the 1980s, ushered in by the 'Parallel Distributed Processing' (PDP) movement in psychology and the cognitive sciences (see Section 8).

The explosive resurgence of interest in connectionism took the AI community by surprise: for example, a 1987 conference on PDP methods, held by the Oxford Experimental Psychology Society, had an audience of 700, more than double the anticipated maximum attendance; whilst in San Diego, America, the first IEEE International Conference on Neural Networks attracted almost 2000 participants. As Dreyfus and Dreyfus later remarked, it was as though "Neural Network modelling may be getting a deserved chance to fail; as did the symbolic approach" [39].

In the following sections I review some of the most notable neural network models developed during the so called 'winter of connectionism'; models which eventually led to the rebirth of the field via new multi-layer network architectures. These opened up exciting new developments in neural computing that finally enabled ANNs to overcome the challenging benchmark problems so elegantly defined just over a decade earlier by Minsky and Papert. I conclude the historical review with discussion of a selection of some notable recent developments in the field.

Following the brief review of neural computing I examine some of the core philosophical and technical issues which have been, and in some cases continue to be, problematic for the field. I commence this section with an examination of some of the technical challenges to the single layer perceptron first identified by Minsky and Papert and subsequently review some of the broader critiques of connectionism (as, for example, identified by Fodor and Pylyshyn [41]) before finally concluding with a review of three important philosophical critiques of computational connectionism that, at least in part, have motivated recent moves towards the embodied, embedded, enactive and ecological approaches to mind.

4.3. **The logical (or weightless) neural network.** I commence this review of neural network developments post publication of 'Perceptrons' with an examination of a relatively little known neural network architecture first invented by Bledsoe and Browning [23] and subsequently developed and refined over many years by Igor Aleksander and colleagues [6]; the building block of this network is now known as the 'logical' (or 'weightless') neuron.

Throughout the 1970s, Igor Aleksander was one of the very few scientists in the UK to continue researching neural computing following the crushing impact of the Minsky and Papert critique of single-layer perceptrons [80].

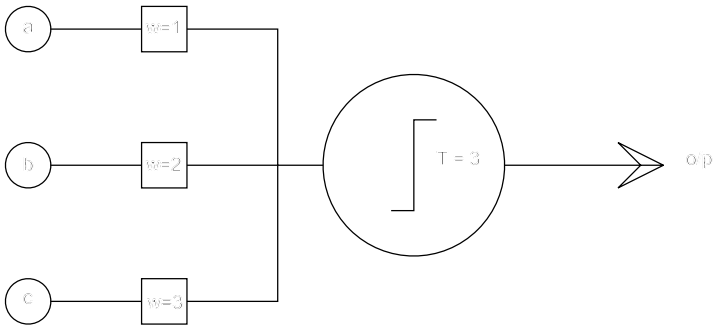

FIGURE 7. The Binary Discriminant Neuron

Building upon Bledsoe and Browning's earlier insights into n-tuple pattern classification [23], Aleksander noted that if the inputs and outputs to an MCP cell are constrained to be binary, the MCP cell computes the 'Binary Discriminant Function'; the operation of such a [Binary Discriminant] neuron (see Figure 7) being to classify (discriminate) its binary input data into one of two sets: TRUE or FALSE. Hence a BDN with n binary (0/1) inputs can classify 2^n possible binary patterns. For example, a BDN with three input lines, (a, b, c) can perform any binary discrimination between $2^3 = 8$ possible binary input pattern vectors (including functions Minsky and Papert had identified as hard for the single layer perceptron such as XOR and parity).

In a radical departure from a conventional neural model which uses real-valued weights (e.g. as found in the classical MCP cell), in 1970 Aleksander, deploying Bledsoe and Browning's *n-Tuple* sampling technique, outlined the operation of a 'weightless neuron' [5]; see Aleksander [6] for an early summary of these ideas. In contrast to the conventional MCP cell it is possible to fully model the behaviour of the Binary Discriminant Neuron simply by maintaining a truth-table (a list) of the desired [binary] input-output mappings; Aleksander's key insight was to realise that a simple hardware device that could maintain such a list is a RAM chip (see Figure 8). In this manner all that is needed to define an n input binary MCP cell is a RAM chip with 2^n memory locations.

To specify a memory location in a RAM chip a binary number is applied to its, n, ADDRESS input lines. To implement the function of the BDN the desired output value, given a specified n-Tuple [RAM address], is stored in the RAM at the location in the RAM memory specified by the n-Tuple [RAM address].

If for each of the 2^n possible binary input patterns to an n-input BDN this weightless 'learning' is carried out using a suitable RAM chip, then by simply causing the RAM to output the data stored in the RAM [at the address defined by the n-bit input vector to the BDN] the output of the RAM chip will perfectly replicate the logical function of the Binary Discriminant Neuron.

NB. The key differences between the two forms of Binary Discriminant Neuron (the logical neuron and the MCP implementation of a BDN) are that (i) the analogue MCP cell stores the required function as a list of n real-valued weight values whereas the RAM neuron stores the function as a list of 2^n binary values; (ii) a single analogue MCP cell can only compute Linearly Separable functions whereas a RAM neuron can perform any

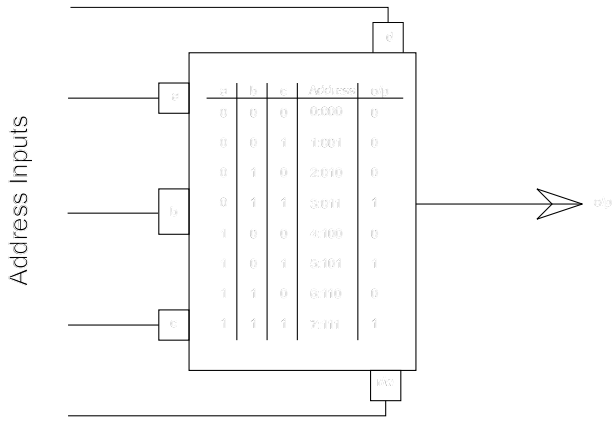

FIGURE 8. The use of a RAM chip as a Binary Discriminant Neuron

desired input/output mapping; in this sense the logical [RAM] neuron functions as an unconstrained, universal neuron.

To use a set of logical neurons as a [logical] neural network, the data to be classified are represented as a vector of m boolean values. This vector is subsequently sampled, according to some predefined fixed schema (typically either randomly or sequentially), with n such samples forming one n-Tuple address to one of the RAM neurons; with k ($k = mDIVn$) such RAMs required to map onto all the m bits of the Boolean input vector. NB. The input vector can be either exclusively sampled, such that each value is used only once, or oversampled by a factor of o, where each value is used o times.

To learn a pattern (n) boolean sampled values from the input vector are used to form an n-bit address into one of the network RAMs (using one such RAM per n-Tuple sample) and a boolean TRUE value is stored at this location in the RAM. To classify the binary data, the input vector is sampled as before; however, instead of storing a TRUE value at each of defined RAM locations, a count of the TRUE values stored at these addresses across all the RAMs in the network is maintained. This count is the system's response to the unknown input vector.

After training a network of k RAMs with one input vector it is trivial to see that a (re)presentation of the same input vector will result in a system output of k (as the $\sum k$ RAMs, each outputting a logical TRUE, will be k. If there is a '1-bit' difference in the input vector this will generate an address not seen by one of the RAMs in the network which would thus output FALSE. The response of the network to this pattern would thus be $(k - 1)$.

However the elegance of this type of network is that, after training with a *set of training patterns*, combinatorial and sampling effects ensure that the network is able to reliably classify patterns not in the training set with a maximal response; if these patterns are 'similar' to those in the training set, then the system has generalised successfully.

Figure 9 shows four training patterns (IMAGE ONE, TWO, THREE & FOUR) being learnt into a small logical network of four RAM neurons, which, by using a tuple (or sample size) or four, completely map onto each of the $4 = 16$ elements of the sixteen element [4x4] binary input vector. It can easily be seen that after this training there will be

eight patterns that will result in maximum FOUR RAMs firing (the four training images plus new unseen vectors defined by OR-ing together input vectors [2+3], [2+4], [3+4], [2+3+4]); for a comprehensive review of the capabilities, limitations and extensions of the weightless network paradigm see Ludemir et al [70].

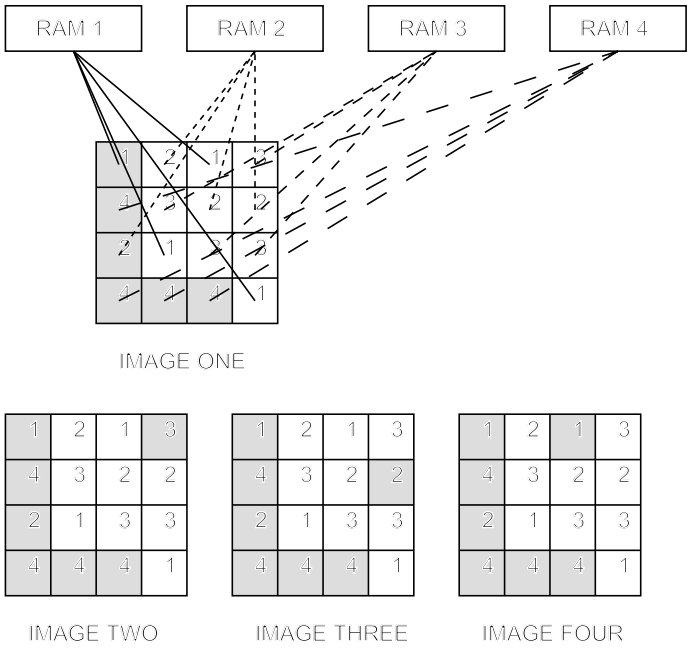

FIGURE 9. Generalisation in a Logical (or 'weightless') neural network

In this way a simple and cost effective route to instantiating large neural networks *in hardware* had been found and as early as the 1984 Aleksander and colleagues [7] were able to demonstrate such systems differentiating, for example, high resolution facial images at real-time video speed; a notable engineering achievement at the time.

5. Hopfield networks

One of the most influential researchers to continue working in the field of neural computing in the new era (i.e. post publication of the Minsky and Papert critique of perceptrons) was the physicist John Hopfield. Hopfield famously made the connection between ANNs and physics and in 1982, using ideas imported from statistical physics, analysed the behaviour of a simple binary network of MCP type cells in which the output of each neuron is connected to the inputs of all other neurons in the network. Hopfield considered that in such a network, at any given instant, the output of all the neurons should be considered together as a [transient] 'state' of the network [55].

Hopfield proved that on presentation of an arbitrary starting vector, such a network would go through a series of transient states before falling into one particular stable pattern of neural firing *after a finite number of iterations*. In this way the network functions as a [bi-directional] pattern associator, associating an arbitrary input vector

to an output vector. Hence, learning in the Hopfield network simply entails designing the weights of the network such that the net is forced to remain stable in certain preferred states. Hopfield suggested that a binary network consisting of N neurons, can reliably store $0.15N$ different stable energy states.

Hopfield networks - also known as 'resonance networks' or 'bidirectional associative memory (BAM)' - consist of a fully connected network of MCP cells with *bi-directional weighted connections*. In the simple Hopfield model the neuron can take on one of two states, on or off [-1, +1]. These states are defined by the application of a threshold MCP model:

$$\text{IF } (\sum_{i}^{n} X_i \times W_i + b > 0) \text{ THEN } X_i = +1 \text{ ELSE } X_i = -1 \tag{7}$$

Hopfield analysed both synchronous network (all neurons update together) and asynchronous networks (where each neuron fires independently, with a mean attempt rate of W attempts per second). Soon after Hopfield first published information on this class of devices, Hopfield nets were applied to difficult optimisation and constraint satisfaction problems (such as the task assignment) [57].

In operation the outputs of a simple binary Hopfield network are first clamped to respective elements of the input vector. The network is then unclamped and each neuron left to operate via application of the threshold MCP model (in asynchronous mode at W evaluations of the model per second) until the network arrives at a stable energy state at which point the outputs of each neuron define the network output vector.

To learn a binary class vector (or 'memory') in a binary Hopfield network (see Figure 10), the network weights need to be adjusted such that the memory is a stable energy state of the network. Hopfield showed that in a network of N neurons with M class archetypes (i.e. training vectors), for each vector in the training set and for each neuron in the network, the weight connecting neuron i with neuron j ($i \neq j$) should be updated by application of simple Hebbian learning as follows:

$$W_{ij} \quad = \quad \sum_{i=1}^{M} X_i^s \times X_j^s \tag{8}$$

.. where W_{ij} is the weight connecting neuron i to neuron j and X_i^s is the i_{th} element of training vector S. NB. Hopfield networks can be used to build auto-associative networks because the matrices produced by the Hebb rule or by computing the pseudo-inverse are symmetric.

If after training the weights of a Hopfield network are such that a set of vector archetype patterns are stable and the network is subsequently presented with an arbitrary input vector, the network will transit through a series of states before settling to one of the learnt vector archetypes (e.g. in a binary Hopfield network, the vector closest in Hamming distance to the input vector).

In retrospect it is clear that Hopfield's 1984 paper was important for several reasons:

(1) It describes a useful class of neural networks that are suitable for implementation in hardware (e.g. within three years of publication of the 1984 paper AT &

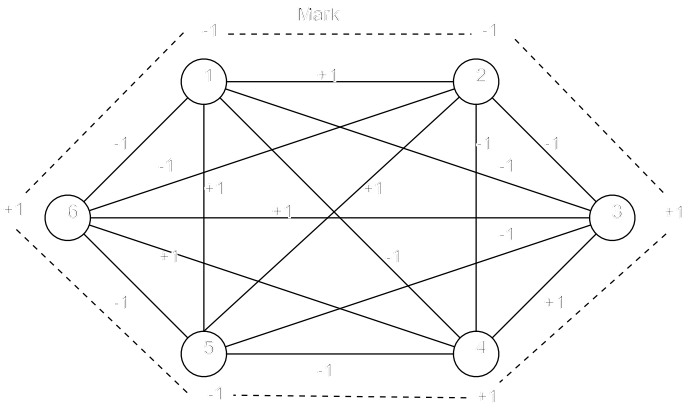

FIGURE 10. Weights in a binary Hopfield network to learn a binary vector representing 'Mark' [-1 -1 +1 +1 -1 +1]

Bell laboratories had announced the first hardware neural network chips based on Hopfield's ideas; Caltech colleague Carver Mead also developed silicon chips based on Hopfield's principles).

(2) Hopfield networks could be used on difficult optimisation problems [57].
(3) Hopfield outlined an original method of analysis of his networks (e.g. proving convergence) drawing on concepts from statistical physics (e.g. the concept of 'energy landscapes').
(4) Hopfield's work led to the development of *The Boltzmann Machine* [3]. The Boltzmann machine consists of simple processing units which are connected via bi-directional links. The links are represented by real valued weight vectors and the processing units can be either on or off. In this way the Boltzmann machine can be seen as a stochastic, generative extension to the Hopfield network.

NB. Although Boltzmann machines were one of the first examples of a neural network architecture capable of learning internal representations, due to a number of serious practical issues relating to learning time, Boltzmann machines with unconstrained connectivity have not proven useful for practical problems in machine learning (see [66]) and will not be discussed further herein.

6. The 'adaptive resonance theory' classifier

The Adaptive Resonance Theory (or ART) classifier belongs to the class of *unsupervised* learning Artificial Neural Networks. In the field of 'machine learning' the problem of unsupervised learning is that of trying to find hidden structure in unlabelled data; in neural networks it is the problem of adjusting neural network weights without reference to an explicit target output vector. The ART-1 classifier was first described in 1987 [28], however Adaptive Resonance Theory traces its lineage back to Grossberg's 1976 research in competitive learning [49].

In the context of connectionism, the ART classifier is unusual in that it has a relatively complex heterogeneous architecture. Unlike say, the Kohonen feature map (see Section 7), an ART network consists of several different types of functional unit ('neuron'), each

computing a specialised operation. A novel and important feature of ART is its ability to switch modes between *plastic* (the learning state in which the internal parameters of the net are modified) and stable (where the net acts as a fixed classifier), *without detriment to previous learning*. Effectively the ART classifier attempts to reconcile the input vector by searching through other category 'archetypes' according to their degree of similarity (as determined by the 'reset module' and the network 'vigilance parameter').

If the algorithm cannot find a suitable archetype, it creates a new one by adding an 'uncommitted neuron' to the recognition layer. The vigilance parameter has considerable influence on the system: higher vigilance (tending to 1.0) produces highly nuanced 'memories' (i.e. many, fine-grained categories), while lower vigilance results (around 0.4 or less) in more general, 'fuzzy' memories (i.e. fewer, more generic, categories).

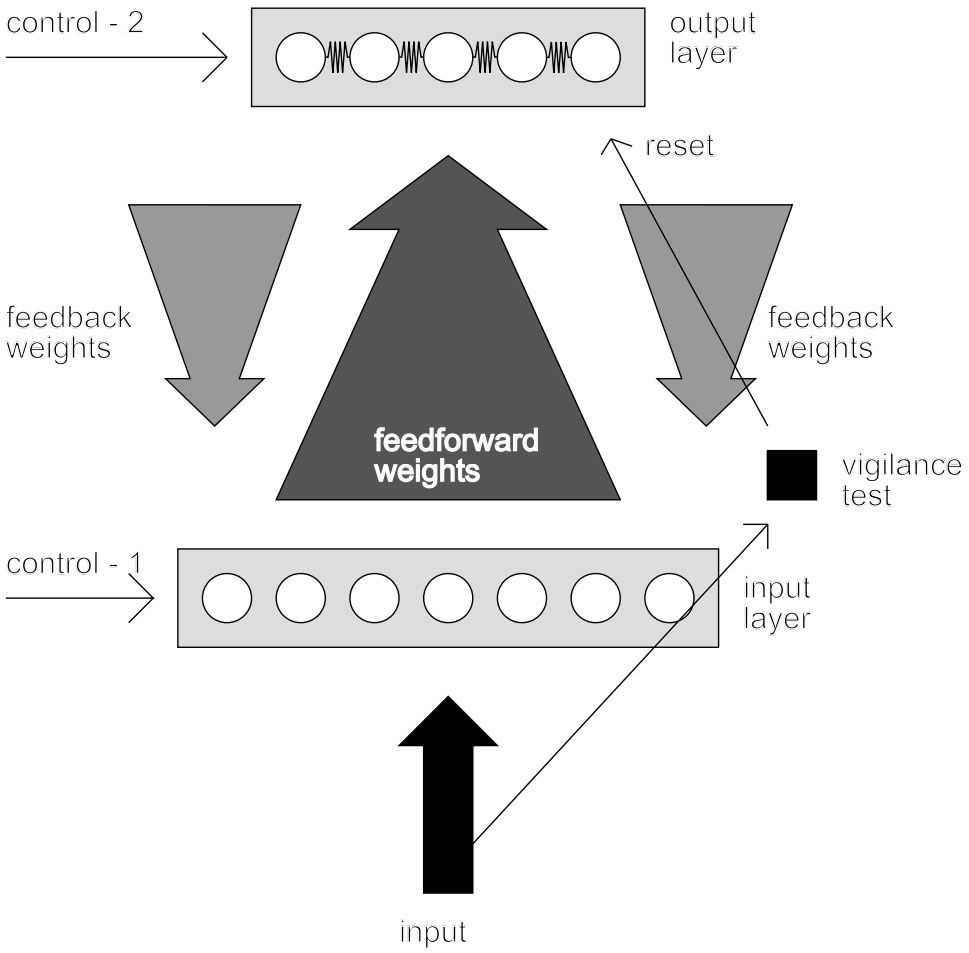

FIGURE 11. The binary ART classifier

Grossberg and Carpenter have developed several versions of ART (ART-1, ART-2, ART-3 etc) as the architecture evolved to become more refined and summarising all

is beyond the scope of this chapter; herein I will concentrate on the simplest ART-1 classifier [28].

The ART-1 network (see Figure 11) acts on binary data and comprises two layers (or 'fields') connected by weights:

(1) Real valued feedforward weights, W, connect the comparison field (the neural input layer) to the recognition field (or neural output layer).
(2) Binary feedback weights, T, connect the recognition field to the comparison field.

The neurons in the recognition field are also fully interconnected laterally via negative weights, so passing a negative quantity proportional to each output neuron activation to the inputs of all the other neurons in the output layer; in this way neurons in the recognition field exhibit lateral inhibition, so enabling each neuron in the recognition field to 'represent' a category (class archetype or'memory') into which input vectors are classified. Data-flow at each layer is controlled by the logical gating signals (control-1 and control-2). The 'reset' module effectively performs the comparison between input vectors and the network class exemplars and decides on adding new class exemplars as appropriate. Each neuron in the comparison field receives three [binary] inputs:

- The i_{th} component of the input vector.
- The i_{th} component of the feedback (class archetype) vector.
- The binary 'control-1' (or 'gain-1') signal.

6.1. **Data resonance.** After initialisation, each time an input vector X is presented to the ART classifier, a modified version of the input data 'resonates' between the comparison and recognition layers as it cycles through three phases: *recognition, comparison and search* until either a classification is made or a new class is assigned and a new node added to the recognition layer. The ART classifier then performs weight-update and becomes ready for new input.

Initialisation: On initialisation the vigilance threshold is set and weight values are set to initial values:
- The vigilance threshold r set to a value in range $0 < r < 1$.
- Feedforward weights are initialised to $W_i = \frac{1}{(1+n)}$ where n is the number of input neurons.
- Feedback weights are all initialised to one $T_i = 1$.

New input vector received: When a new input vector is received by the classifier control signals are set to the following values and operation moves to the *recognition phase*.
- Control - 1 is HIGH whenever valid input is present.
- Control - 2 is HIGH whenever valid input is present.

Recognition phase: Each node in the input layer receives input from three sources:
(1) The input data vector X_i.
(2) The feedback vector signal.
(3) The control-1 signal (which in the recognition phase is set to +1).

Each neuron in the comparison field layer outputs one *iff* two out of its three binary inputs are active and in this way a new *comparison vector* is generated.

The new comparison vector causes activation at the recognition layer by the usual weighted sum of inputs (via the real-valued feedforward weights, W). The

combined effect of the lateral inhibition at the output layer and the activation caused by the input ensure that for any given input only one output neuron is active. This 'winning node' is the neuron whose real valued input weights most closely match the current comparison vector; the control signal 'control-1' is set LOW as this occurs.

The winning node then passes its binary class exemplar back to the comparison layer by multiplying its winning activation level (+1) by its binary feedback vector, T.

Comparison phase: In the comparison phase there are two vectors active at the comparison layer: the data vector, X and the feedback vector, T. As the control-1 signal is now zero, the application of the 'two thirds rule' determines the new comparison vector (i.e. the state of each node at the comparison layer), Z is simply defined by the logical AND of (X AND T). The reset module now compares the new comparison vector Z to the input vector X. This process is evaluated by dividing the number of 'ones' in common between the binary comparison vector Z and the binary input vector X, by the total number of 'ones' in the input vector. I.e.

$$s = \frac{T \cdot X}{X \cdot X} \tag{9}$$

IF (s is greater than the vigilance threshold, r) THEN classification of the input vector is made to the winning recognition node j and ART enters the **weight-update** phase; otherwise the control signal 'control-2' is set LOW and ART enters the **search phase**.

Search phase: In the search phase the LOW control-2 signal causes the current winning recognition node, j to be disabled; at this point if there are no other nodes left in the recognition layer, then a new node is added to the recognition layer and ART enters the **weight-update** phase; otherwise control-1 reset to 1 and ART re-enters its **recognition phase**.

Weight-update phase: In ART weight-update the two sets of weights, T and W, are updated as follows:

- For the binary feedback weights T:

$$T_{ij}(t+1) = T_{ij}(t) \times X_i \tag{10}$$

- For the real valued feedforward weights W:

$$W_{ij}(t+1) = \frac{T_{ij}(t) \times X_i}{0.5 + \sum_{i=1}^{n} T_{ij}(t) \times X_i} \tag{11}$$

In this way ART-1 offers a novel solution to the problems of plasticity and stability in neural network learning. Whilst there remain unused nodes in the recognition layer pool, ART can continue to develop new classes as required by the presentation of novel input vectors (i.e. new memories/classes can always be added); conversely, the addition of a new class does not disturb previously learnt classifications made by the network (i.e. old memories remain stable). These are very attractive properties of the ART classifier.

7. The Kohonen 'feature-map'

Like the ART classifier, the Kohonen network also belongs to the class of *unsupervised* neural networks. In its original form it was significantly biologically inspired. A Kohonen network consists of an array of adaptable neurons - the outputs of which form a 'feature map'. Over time each neuron becomes 'tuned' to a particular input vector, by adaption of its weight vector. After training, similar input vectors will excite similar regions of this 'feature map'.

A Kohonen feature map can be considered as a fully inter-connected network of N MCP style cells (defined by a weight vector n), with in addition each neuron also having a further set of weighted inputs to an M element external input vector (defined by the weight vector m), see Figure 12.

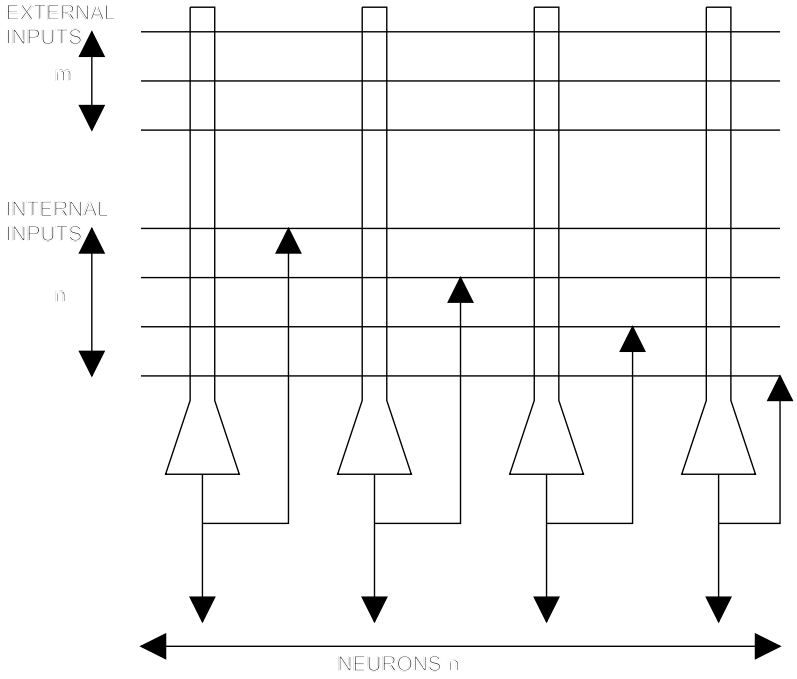

Figure 12. The Kohonen Feature Map

$$o_i = f^a(f_i^m(t) + f_i^n(t-1)) \tag{12}$$

$$f_i^m(t) = \sum_i x_i m_i \text{ for all external inputs} \tag{13}$$

$$f_i^n(t) = \sum_i o_i n_i \text{ for all internal connections} \tag{14}$$

f^a is typically a sigmoid squashing function whose output is always positive. The n internal weights are clamped as a function of the sinc operator and distance, (see Figure 13).

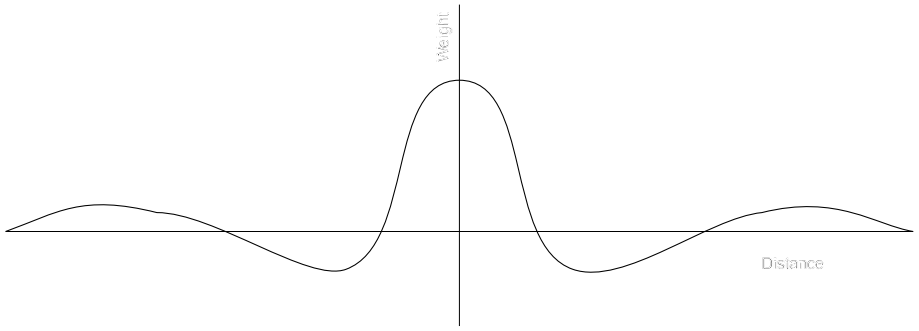

FIGURE 13. The 'Mexican hat' operator, or sinc function, defining internal weight values as a function of distance

7.1. **Learning in a Kohonen feature map.** Learning in the Kohonen feature map proceeds as follows:

(1) All neural weights are initialised to small random values.
(2) On presentation of an input vector from the training set, the first task of the network is to find the focus neuron, X_c; this is the neuron c whose external weights M most closely resemble the input vector X (e.g. the Euclidean distance between the input vector and the neuron external weight vector is minimised):

$$X_c = MIN(DIST(X, M)) \tag{15}$$

E.g. For Euclidean Distance:

$$|x - m| = |x_1 - m_1| + |x_2 - m_2| + |x_3 - m_3| + ... \tag{16}$$

Where $|a - b|$ is the modulus of $(a - b)$.

(3) Update the weight vectors that lie within a neighbourhood $N_c(t)$ of c such that:

$$m_i(t + 1) = m_i(t) + \alpha(t)(x(t) - m_i(t)) \tag{17}$$

Where $N_c(t)$ and $\alpha(t)$ are empirically defined by monotonically decreasing functions of time, e.g. $N_c(t) = \phi/t$.

7.2. **An artificial example: classifying pairs of real valued random input vectors.** To illustrate Kohonen weight learning the external weights for a 2D Kohonen feature map, with a 2D external input vector, is first initialised with small random weights $[0 \leq x < 1]$ and then repeatedly presented with pairs of random values in the interval $[0 \leq m < 1]$. Network behaviour is illustrated by plotting the weights of each neuron as a point on the plane [0..1, 0..1] with lines linking topologically adjacent neurons. Initially the plot shows the network weights as randomly distributed and thence the plot as a random mesh of points; over time as Kohonen learning takes place, the external weights to each neuron adapt and the network unfolds such that neurons are distributed evenly and preserve their topological relationships on the plane, (see Figure 14).

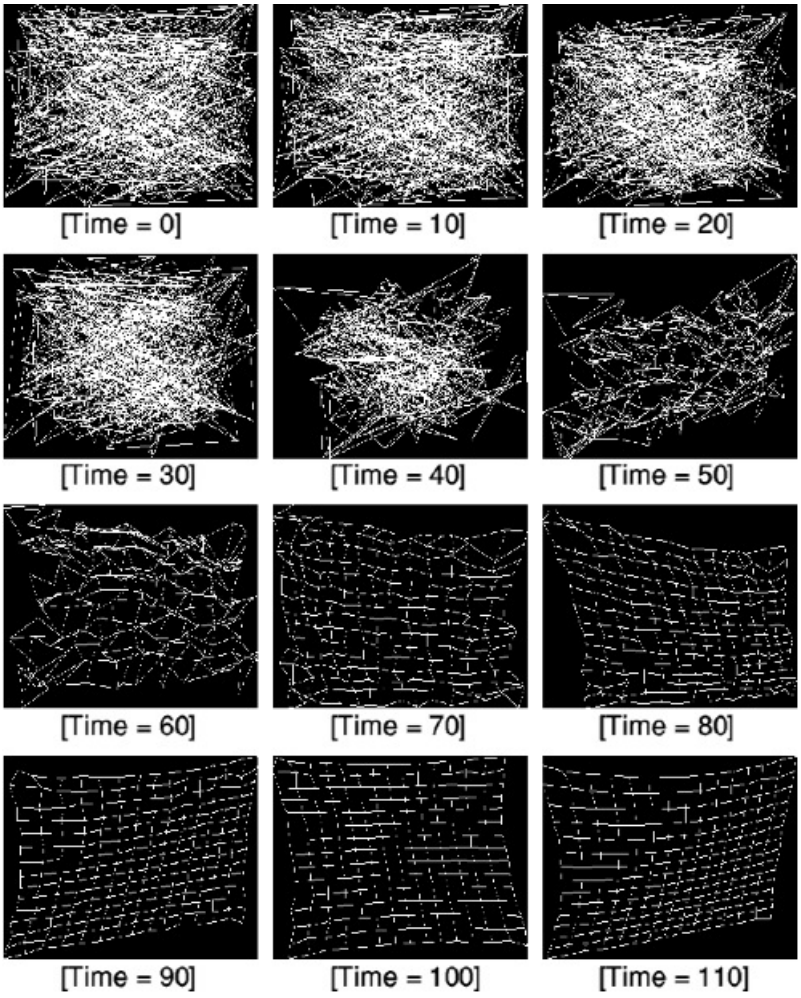

FIGURE 14. An illustration of Kohonen network weight adaption with random pairs of real-valued inputs

7.3. **Practical applications.**

Dimension Reduction: using a Kohonen Feature Map. A series of n-dimensional input vectors can be mapped, (classified), onto a two dimensional feature map. As the network decides the classification without external supervision it is an *'unsupervised' learning algorithm.* A Kohonen network 'student classifier' may have n different inputs corresponding to grades in particular subjects etc. and m output neurons. After training the map would classify students into one of m 'attainment' groups.

Speech Recognition: using a Kohonen feature map. Each neuron is taught to respond most strongly to a specific phoneme with the input vector defined by the Fourier transform of the speech signal. In use:

- At each sample period a specific neuron will be most active.

- If this is consistent over a given time window (e.g. 4 samples out of 7) then that phoneme on the map is classified as active.
- A word is identified as a [recognisable and learnt] trajectory of phonemes across the feature map over a specified time window.

7.4. **Supervised feature-map learning.** The standard Kohonen methodology is paradigmatically an *unsupervised* learning algorithm; however, there may be occasions when it is desired to associate particular input vectors with particular nodes on the feature map and 'learning vector quantisation' (LVQ) as one algorithm to achieve this. LVQ is a *supervised learning algorithm* that can be used to fine tune such a feature map (e.g. by the use of additional, new, training vectors to improve performance of individual neighbourhoods). The algorithm works as follows:

(1) Select training vectors with a known classification and present to the network.
(2) Calculate winning neuron (the focus neuron, c)
(3) Modify the weight vector of focus neuron as follows:
    If the network correctly classified the input:

$$m_i(t+1) = m_i(t) + \alpha(t)(x(t) - m_i(t)) \tag{18}$$

    If the network incorrectly classified the input vector:

$$m_i(t+1) = m_i(t) - \alpha(t)(x(t) - m_i(t)) \tag{19}$$

8. The multi-layer perceptron

One of the key moments in neural network research, which led to a "renaissance" in the field of artificial neural network research in the 1980s, was the development of a learning rule that can be applied to multi-layer neural networks; networks with so called 'hidden neurons' (i.e. processing nodes not directly connected to the network output). The rule which gained most attention at the time - being widely popularised by Rumelhart, Hinton, Williams [104] and the PDP group at the University of California - was called back propagation (or the *Generalised Delta rule*) which propagates an error term back through the network - from the output units to the input units - making weight changes such that overall prediction error of the net is minimised.

Historical note: in his book on Support Vector Machines, Vapnik cites Bryson et al 1963 [27] with the first publication of the back propagation algorithm; however it wasn't until 1974, in the PhD research of the control engineer Paul Werbos, that the procedure was applied in the context of neural networks [130]. Coincidentally, around the same time as the work of the PDP group, back propagation was also independently (re)discovered by Le Cun and published (in French) in 1985 [67].

A full introduction to the back propagation learning rule is beyond the scope of this chapter, however a brief description is offered below.

8.1. **Back propagation (or the generalised-delta rule).** Many problems in approximation theory can be viewed as a process of curve fitting, i.e., the fitting of lines [or surfaces] to a set of points in a 2 or more dimensional space.

- In 2-D space - lines are fitted to points.
- In 3-D space - planes are fitted to points.
- In n-D space - hyperplanes are fitted to points.

It is possible to view learning in a multi-layer network (MLP) as the process of curve fitting that occurs as the MLP adapts its structure to model the training data. If there are K (weights + biases) in the MLP network then, over time, the learning process traces out a path in (K+1) dimensional weight/error space. If the learning process is good, at the end of the learning process the network will arrive at a low point in this error / weight space.

The generalised delta rule (GDR) works by performing gradient descent in error / weight space and is a generalisation of the standard delta (or Widrow-Hoff) learning rule discussed in Section 5 above. That is, using the GDR, after each training pattern has been presented, the resulting network prediction error on that pattern is computed. This is calculated by comparing the actual output of the network with the desired output as specified by the current training pattern. Subsequently each weight in the network is modified by moving down the error gradient towards the minimum (for that input/output pattern pair). The 'gradient descent' learning technique involves changing each weight in proportion to the negative of the derivative of the error (as defined by the current training vector pair).

The basic form of the generalised delta rule is identical to the simple delta rule discussed in Section 3.3 above):

$$\Delta W_{ji} = \eta \delta_j I_i \tag{20}$$

At the output layer the error signal is:

$$\delta_j = \eta (T_j - O_j) f'_j net_j \tag{21}$$

and for a hidden layer the error signal is:

$$\delta_j = f'_j net_j \sum \delta_k W_{kj} \tag{22}$$

Where k represents the index of each output node, $f'_j net_j$ is the derivative of a *semi-linear* activation function acting on the j_{th} unit, which maps the total input to the unit to an output value. NB. A semi-linear function must be differentiable, continuous, monotonic and non-linear. Thus, using a sigmoid activation function we get:

$$o_j = \frac{1}{1 + e^{-net_j}} \tag{23}$$

Where $net_j = \sum_i W_{ji} O_i + \theta_j$ and θ_j is a bias term which can be learned like any other weight by considering it connected to a unit which is permanently on. The derivative of f is thus:

$$\frac{do_j}{dnet_j} = o_j(1 - o_j) \tag{24}$$

Hence for an output unit the error signal is:

$$\delta_j = (t_j - o_j) o_j (1 - o_j) \tag{25}$$

and for a hidden unit the signal is:

$$\delta_j \quad = \quad o_j(1-o_j)\sum \delta_k W_{kj} \tag{26}$$

8.1.1. *The learning rate eta.* The above learning procedure requires only that the change in weight is proportional to $\frac{dE}{dW}$, whereas true gradient descent requires infinitesimally small steps to be taken. The constant of proportionality is the *learning rate* constant, η. The larger this value, the larger the changes in weight at each iteration and the faster the network learns. However if the learning rate is too large, then the network will go unstable and oscillate. The PDP group suggested that one way to increase the learning rate, without leading to oscillation, is to introduce a momentum term. I.e.

$$\Delta W_{ji}(n+1) \quad = \quad \eta\delta_j o_i + \alpha\Delta W_{ji}(n) \tag{27}$$

.. where n indexes the current input/output presentation number, η is the learning rate constant and α is the momentum constant. In effect the 'momentum' term defines how much past weight changes affect the current direction of movement, providing a force analogous to momentum in weight space, which acts to filter out high frequency variations in the error surface. For example, if the network arrives at a gradually descending ravine in weight error space, the steepest error gradient may be mainly across, rather than down, the ravine. The use of a momentum term tends to filter out such sideways movement, while compounding movement down the ravine.

8.1.2. *One learning iteration of the Generalised Delta rule.* In one learning iteration of the Generalised Delta learning rule the following sequence of events occur:

- The network is presented with an input pattern (all the input units of the network are set to the required values).
- The input vector is used to compute the output values by feeding forward through the net and computing activation values for all the other units.
- The output vector for this pattern is compared to the required, or target, pattern and the error term is calculated for every output unit.
- Error terms are recursively propagated backwards through the net to the other units in proportion to the connection strengths between the units.
- The weights are then adjusted in such a way as to reduce the error terms, by performing gradient descent in weight error space, in a similar manner to the simple delta rule.
- The process is repeated for all the input/target pattern pairs in the training set.

The process of presenting to the network all the patterns over which it is to be trained is defined as an epoch of learning. Training continues for as many epochs as are necessary to reduce the overall error to an acceptably low value.

9. Radial basis function networks

A radial basis function (RBF) network consists of an *input layer; a hidden layer and an output layer.* At the input layer the nodes distribute the network input vector to all the nodes in the hidden layer; at the hidden layer each node contains a centre and a function called a 'basis function'; finally each node at the output layer multiplies the

activation of each hidden layer node by a coefficient and sums the resulting values to produce a network output, see Figure 15.

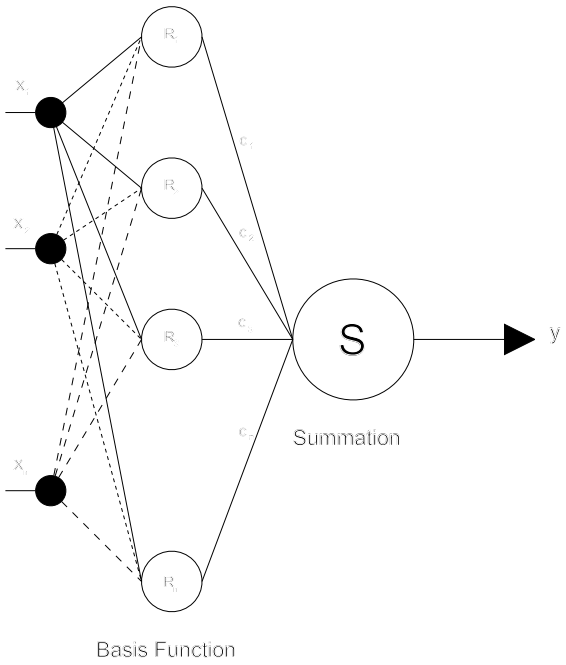

FIGURE 15. The Radial Basis Function network

A RBF network is defined by two parameters:

- A set of 'Centres' each defined by a vector, R, of the same dimensionality as the RBF network's input vector; a centre is associated with each node of the RBF hidden layer.
- A set of coefficients $\{c_{ij}\}$ which weights the connection between the i_{th} node in the RBF hidden layer and the j_{th} node in the RBF output layer.

In the RBF network the basis functions take the place of the activation function of a MCP cell in an MLP network. Each basis function ϕ operates on a scalar value and returns a scalar value, which is the output of the hidden layer node. Examples of commonly used basis functions include (a) the thin plate spline: $\phi(x) = x^2 log(x)$ and (b) the Gaussian function $\phi(x) = e^{-\frac{x^2}{2w}}$ (where w represents the width of the Gaussian).

At the hidden layer node each node calculates the Euclidean distance between the current input vector, x, and the node's centre vector, R. The j_{th} node's basis function operates on the resulting distance to produce the output of the j_{th} hidden layer node:

$$o_j = \phi(||x - R||) \tag{28}$$

At the output layer each node multiplies the 'activation values' of each hidden layer node o by a scalar coefficient, c_i and sums the resultant values to produce the RBF

output value for that node y_i. Thus, for each node in the output vector the output value of the RBF is defined as:

$$y_i = \sum_{1}^{m} c_{ij} o_i \tag{29}$$

..where m is the number of nodes in the hidden layer, o_i represents the output of the i_{th} hidden layer node and c_{ij} is the coefficient weighting the connection between the i_{th} hidden node and the jth output layer node.

The overall output equation for each output node of the RBF is thus:

$$y_i = \sum_{1}^{m} c_{ij} \phi(||x - R||) \tag{30}$$

.. where R_i is the centre belonging to the i_{th} hidden layer node.

9.1. **Learning in an radial basis function network.** Before RBF training commences, the centres, R, must be chosen. Centres must be chosen to represent the training data set and the correct choice of centres is critical for good RBF performance :- too many or too few decreases network accuracy.

Various heuristics have been tried to determine the optimum number and position of centres; these include: positioning centres according to a simple distribution (e.g. a uniform or Gaussian distribution over data space); or choosing a distribution related to the training data distribution (e.g. via the application of a suitable clustering algorithms). The only remaining parameters which need to be trained are the coefficients c (as all the other RBF values are known).

The input and output training sets provide pairs of values for x and y. For each input-output xy pair, an equation is produced:

$$y = c_1\phi(x, R_1) + c_2\phi(x, R_2) + c_3\phi(x, R_3) + ...c_n\phi(x, R_n) \tag{31}$$

.. where $\phi(x, R_i) = \phi i(||x - R||)$. As there is usually more than one pattern in the training set a group of equations is produced:

$$y_1 = c_1\phi(x_1, R_1) + c_2\phi(x_1, R_2) + c_3\phi(x_1, R_3) + ...c_n\phi(x_1, R_n) \tag{32}$$
$$y_2 = c_1\phi(x_2, R_1) + c_2\phi(x_2, R_2) + c_3\phi(x_2, R_3) + ...c_n\phi(x_2, R_n) \tag{33}$$
$$y_3 = c_1\phi(x_3, R_1) + c_2\phi(x_3, R_2) + c_3\phi(x_3, R_3) + ...c_n\phi(x_3, R_n) \tag{34}$$
$$y_m = c_1\phi(x_m, R_1) + c_2\phi(x_m, R_2) + c_3\phi(x_m, R_3) + ...c_n\phi(x_m, R_n) \tag{35}$$

And as there will [usually] be many more training pairs than there are unknown coefficients we have an over determined set of equations.

$$y_1 = \phi(x_1, R_1) \cdot \underline{c} \tag{36}$$
$$y_2 = \phi(x_2, R_2) \cdot \underline{c} \tag{37}$$
$$y_3 = \phi(x_3, R_3) \cdot \underline{c} \tag{38}$$
$$y_m = \phi(x_m, R_m) \cdot \underline{c} \tag{39}$$

In matrix form this can be more succinctly expressed as $\underline{y} = \underline{A} \times \underline{c}$. Solving this equation simply involves finding the inverse of $\underline{\mathrm{A}}$. There are a variety of techniques that can be used to produce this inverse and hence solve for the coefficients $\underline{\mathrm{c}}$; these include:

- LU (Lower Upper) decomposition; factors a matrix as the product of a lower triangular matrix L and an upper triangular matrix U.
- QR decomposition; factors a matrix as the product of an orthogonal matrix Q and an upper triangular matrix R.
- Singular value decomposition.

Singular decomposition is generally regarded as being the preferred method, as it can accommodate cases where the training patterns are closely related to each other.

In summary, RBFs have some similarities to perceptrons:

- they learn from a set of training set exemplars;
- they are able to generalise from training data;
- they store information in a distributed manner;
- and they map well onto parallel processing hardware.

.. and some differences:

- MLPs are defined by weights and thresholds; RBFs by centres and coefficients;
- training in MLPs is relatively slow and computationally expensive compared to RBFs;
- perceptrons weight and sum their inputs before passing them through an activation function; RBFs pass each input through a basis function before weighting and summing.

However, because RBF networks can learn problems which a single layer perceptron cannot (e.g. the so called 'hard' problems) they are considered more powerful than SLPs.

10. Recent developments in neural networks

With the development of the multi-layer perceptron and the widespread adoption of the back-propagation learning rule (and the alternative basis-function framework developed by Broomhead and Lowe) some of the key criticisms of simple single layer Perceptions from Minsky and Papert were finally addressed and the subsequent decades witnessed an explosion of papers detailing the application of neural network technology in different fields; these include: system identification and control [59]; game-playing and decision making [119]; pattern recognition [6]; sequence recognition (gesture, speech, handwritten text recognition); medical diagnosis; financial applications (automated trading systems); mobile robot localisation [62] [19]; data mining (knowledge discovery); colour recipe prediction [18] and e-mail spam filtering etc.

I will complete this section with a brief summary of some of the recent key developments in artificial neural networks technology.

10.1. **Support vector machines.** Similar to the RBF, sensu stricto the support vector machine (SVM) is a machine learning architecture rather than a neural network. The original SVM was co-invented by Corinna Cortes and Vladimir Vapnik [36] in 1995 and rapidly because the machine learning algorithm of choice for the cognoscenti (until the

more recent development and popularisation of so called 'deep learning networks' by Hinton and Schmidhuber).

Given a set of training examples belonging to two classes, the SVM algorithm builds a model that assigns new examples into one class or the other, such that the examples of the separate categories are divided by a clear gap that is as wide as possible. SVMs perform non-linear classification using the so called 'kernel trick' by which they implicitly map their inputs into higher-dimensional feature spaces.

10.2. **Reinforcement learning.** Reinforcement learning defines a broad area of machine learning research concerned with describing how agents ought to take actions in an environment so as to maximise 'cumulative reward'. Effectively reinforcement learning, by a system of trial and error, works out what is the best action to take in any given situation. Feedback is given by reward or punishment - a positive reward is given for successfully achieving the task; a punishment (or 'negative reward') is given for any action that impedes successful completion of the task.

The first neural network implementation of reinforcement learning was developed in the work of Barto et al in the early 1980s [11] with more advanced reinforcement learning schemes, such as Q-learning, following in the 1990s. Q-learning algorithms typically store the expected reinforcement value associated with each situation-action pair, in a look-up table (Q-table), but this approach often fails to scale well. In 1997 Touzet suggested replacing the Q-table with an MLP network or, most effectively with a Kohonen feature map; as Touzet concludes [124] "we may have found with this implementation the 'ultimate' Q-learning implementation".

Reinforcement learning has been used in numerous difficult application areas including: mobile robotics, game playing, pole balancing etc.

10.3. **Artificial recurrent neural networks.** The artificial recurrent neural network (ARNN) defines a class of neural network where connections between neurons form a directed cycle. This enables the network to create an internal state which allows it to exhibit complex dynamic temporal behaviour akin to memory, which in turn enables the ARNN to more easily process arbitrary sequences of inputs than non-recurrent architectures such as the Multi-Layer Perceptron. Albeit, by appropriate coding of input to a MLP (e.g. $I_{t=0}, I_{t-1}, I_{t-2}..I_{t-k}$) it is *possible* to enable MLPs to processes temporal data, but not as *naturally* as ARNNs.

The lineage of ARNNs can be traced to Hopfield's architecture (albeit Hopfield's architecture was not engineered to process *sequences* of patterns). As a generalisation of the fully recurrent Hopfield network, each neuron has a *directed* connection to every other neuron (specified by a 'real-valued' weight). Some of the neurons in the network will be defined inputs, some outputs, and the rest hidden neurons.

The ARNN consists of a finite number of neurons. The activation of each processor $i = 1..N$ is updated by the equation:

$$x_i(t+1) = \sigma(\sum_{j=1}^{N} a_{ij} x_j(t) + \sum_{j=1}^{M} b_{ij} u_j(t) + c) \tag{40}$$

.. where N is the number of neurons, M is the number of external input signals, x_j are the activations of the neurons, u are the external inputs, and a_{ij}, b_{ij}, and c are the real valued weights.

In pioneering work Hava Siegelmann and Eduardo D. Sontag [114] proved that an ARNN with a finite number of neurons, rational valued weights (contra full precision real number-valued weights), with the function u being the simplest possible 'sigmoid' (the saturated-linear function):

$$\sigma(x) := \begin{cases} 0 & \text{if x} < 0 \\ x & \text{if } 0 \leq x \leq 1 \\ 1 & \text{if x} > 1 \end{cases} \tag{41}$$

.. defines a model of computation that is polynomially related to Turing machines; more precisely, given any multi-tape Turing machine, one can simulate it in real time by some network with rational weights. They subsequently generalised their proof to show that the use of [just one] irrational values for weights results in a machine with super-Turing power:

> We prove that neural networks can recognize in polynomial time the same class of languages as those recognized by Turing machines that consult sparse oracles in polynomial time (the class P/poly); they can recognize all languages, including of course noncomputable ones, in exponential time.

In response to criticism of the use of ARNNs with irrational (real valued) weights, Siegelmann remarks:

> In a natural analog computation process, one starts from initial conditions that constitute the (finitely describable) input, and the system evolves according to the specific equations of motion to its final position, which constitutes the output. The evolution is controlled by the exact equations of motion with the exact physical constants. The analog physical system 'solves' the equations of motion exactly. For example, planetary motion is used to measure time with very high precision although we know the gravitational constant G only to two digits. The planets, of course, revolve according to the exact value of G, irrespective of its measurement by humans.

In subsequent work from 1995 Siegelmann [113] further describes a highly chaotic dynamical system she termed the 'Analog Shift Map (ASM)', which is proved to have computational power beyond the Turing limit (i.e. it is 'super-Turing'), computes exactly like ARNNs and conjectures that it can be used to describe other natural [chaotic] physical phenomena and hence that ARNNs can accurately model other chaotic physical phenomena.

10.3.1. *Reservoir Computing and Echo-State Networks.* A reservoir computing system typically consists of a [fixed] randomly configured dynamical system, the dynamics of which map an input vector into a higher dimensional space, from where a simple learnt read-out mechanism maps the resulting 'reservoir' state to the desired output. The

reservoir itself typically consists of a collection of random, recurrently connected non-linear processing units, with its overall dynamics being driven by the input vector. Because the reservoir parameters are fixed, learning is only performed at the output layer and hence can be made very efficient.

An example of reservoir computing is the 'Echo State Network (ESN)'. The ESN consists of a non-trivial (typically 50-1000 unit) artificial recurrent neural network with a very sparsely connected hidden layer where the weight inter-connectivity vector is randomly assigned. In this way the reservoir can exhibit dynamic memory by maintaining its activation even in the absence of the input vector. In principle, the ARNN can learn to approximate any given continuous function with arbitrary accuracy.

Because there are no cyclic dependencies between the trained readout connections training an ESN becomes a simple linear regression task and hence learning is very fast. In their Science article of 2004 Jaeger and Haas report [60]:

> On a benchmark task of predicting a chaotic time series, accuracy is improved by a factor of 2400 over previous techniques. The potential for engineering applications is illustrated by equalizing a communication channel, where the signal error rate is improved by two orders of magnitude.

10.3.2. *Continuous Time Recurrent Neural Network (CTRNN).* A CTRNN [13] deploys a system of ordinary differential equations to model the effects on a neuron of incoming spike trains to the neuron. In typical engineering applications CTRNNs are often deployed in conjunction with 'evolutionary learning algorithms' to learn the network parameters. Because of the simplicity of the 'evolutionary learning' algorithm (which can be configured to simultaneously learn network architecture (number of neurons) and inter neuron weights) and because CTRNNs do not model neural activations at the spike train level, they are relatively simple to simulate and compute and hence have been deployed in various real-world application areas (e.g. evolutionary robotics, vision and minimally cognitive behaviours).

10.4. **The spiking neuron neural network.** In contrast to classical ANN models, spiking neurons do not simply aggregate individual action potentials as a mean firing rate but act on temporal sequences of spikes. Like classical connectionist models, spiking neural networks (SNNs) have been studied both for their computational/engineering properties and as models of neurological processes. In contrast to the MCP model, in addition to neuronal and synaptic weight, Spiking neurons fundamentally embrace time in their operation. Typically a spiking neuron model only 'fires' when an internal state (akin to neuronal 'membrane potential') reaches a specific value; at this point the neuron fires and transmits a signal to other neurons which, in turn, will increase or decrease their potentials. Various schemes exist for encoding the outgoing spike train as a real-value number, specifying either the frequency of spikes, or the timing between spikes, to encode information.

10.4.1. *The 'Integrate and Fire' Neuron.* The 'Leaky Integrate-and-Fire' (LIF) neuron [47] is perhaps the simplest spiking neuron model and in its simplest form is simply modelled as a "leaky integrator" of its input $I(t)$:

$$\tau_m \frac{dv}{dt} = -v(t) + RI(t) \tag{42}$$

.. where $v(t)$ represents the membrane potential at time t, τ_m is the membrane time constant and R is the membrane resistance.

Thus equation (42) simply describes a resistor-capacitor (RC) circuit where the 'leakage' term is due to the resistor and the integration (of $I(t)$) is due to a capacitor wired in parallel to the resistor.

In the LIF model, the so called firing (or spiking) events are not explicitly modelled; however, when the membrane potential $v(t)$ reaches a certain threshold v_{th} (its 'spiking threshold'), it is forced to a lower value v_r (its 'reset potential') and the leaky integration process (described by Equation 42) starts again with the initial value v_r.

It is possible to make the LIF model slightly more biologically plausible by adding an *absolute refractory period* Δ_{abs} to cut in immediately after $v(t)$ reaches v_{th}. During this period, $v(t)$ is clamped to v_r and the leaky integration process is re-initiated after a delay of Δ_{abs} following the 'spike'.

10.4.2. *The Hodgkin-Huxley model.* In 1963 the English physiologists and biophysicists Alan Lloyd Hodgkin and Andrew Huxley received the Nobel Prize in Physiology/Medicine for their 1952 explanation and model of the ionic mechanisms underlying the initiation and propagation of action potentials in the squid giant axon [54]. The Hodgkin-Huxley model defines a set of nonlinear differential equations that, compared to classical ANNs, provide a 'hi-fidelity' approximation to some of the electrical characteristics of real brain neurons. In particular the Hodgkin-Huxley model uses a set of nonlinear differential equations to describe how action potentials in neurons are initiated and propagated.

Although for many years the large computational demands of the Hodgkin-Huxley model limited its application in large neural network computational simulations, recent advances in computing technology, for example the IBM Blue Gene system, have enabled the model to be deployed in very large scale simulations of animal, and human, brains. For example, the Michael Hines NEURON simulator (as used in Henry Markram's Blue Brain and Human Brain Project) deploys the Hodgkin-Huxley (HH) axon ion channel model to describe how action potentials in neurons are initiated and propagated.

The Blue Brain project was headed by the founding director Henry Markram [and co-directed by Felix Schürmann and Sean Hill]. Markram's involvement was in part motivated by his child's autism and his failure as a neuroscientist to understand the condition, which prompted a life-long drive to better understand the human brain. Using an IBM Blue Gene supercomputer running Michael Hines's NEURON software, the Blue Brain project did not simply consist of a large Artificial Neural Network, but fundamentally centred around a "biologically realistic" model of neurons based on the Hodgkin-Huxley model.

In July 2011 the Blue Brain project delivered a cellular mesocircuit of 100 neocortical columns with one million cells in total, with an (entire) cellular rat brain model expected to be completed in 2014, simulating around one hundred million cells. Flushed with their early success Markram's team confidently predict that a functional cellular human brain (with a simulation requirement of over one hundred billion cells) will be possible by 2023.

In this context in January 2013 the EU 'Future and Emerging Technologies Flagship Initiative' awarded in excess of one billion euro (over ten years) to Markram's 'Human Brain Project', with a modest stated goal of using a powerful supercomputer to recreate everything known about the human brain. Nature journal observed this to be "a hugely ambitious goal that has been met with some scepticism".

10.4.3. *Liquid State Machines.* A Liquid State Machine (LSM) consists of a large collection of neurons (nodes) in which each neuron receives time varying input from external sources (the inputs) as well as from other neurons. Neurons are usually randomly and recurrently connected to each other (but simple feed-forward variants are possible). The recurrent nature of the connections turns the time varying input into a spatio-temporal pattern of activations across the network neurons. The spatio-temporal patterns of activation are mapped to the desired output function by simple output neurons.

Like the Echo-state network, the theoretical approach deployed by the LSM suggests that to approximate a given input/output behaviour F on particular functions of time (or spike trains) it is simply necessary to randomly pick some sufficiently complex reservoir of recurrent circuits of spiking neurons, and then merely adapt the weights of a single pool of feedforward spiking neurons to approximate the desired target output behaviour [2], (F). And further that there exists a simple local learning rule that can adapt such pool of spiking neurons to approximate any given specific continuous function [72].

As Maass and Markram report [73]

> .. this approach has the advantage that the same recurrent circuit can be used simultaneously - with the help of additional other readout functions that can be trained independently - to compute in parallel different outputs from the same input $u(\cdot)$. This provides a new paradigm for parallel computation in real-time on time-varying input that appears to be rather attractive from the biological point of view. In other words: m different filters $F_1..F_m$ can be implemented with the same recurrent circuit (i.e. the same L), yielding a giant saving of hardware (i.e. neurons).

10.4.4. *Multi-variate Spiking Networks.* NESTOR is an example of a spiking neuron connectionist architecture whose constituent neurons inherently operate on rich (bi-variate) information encoded in spike trains, rather than at a simple mean firing rate. NESTOR was first proposed by Nasuto and Bishop in 1998 [83] as a spiking neuron architecture designed to locate an object (memory) projected onto an artificial retina (see Figure 16):

> The NEural STochastic diffusion search netwORk (NESTOR) consists of an artificial retina, a layer of fully connected matching neurons and retinotopically organised memory neurons. The bi-variate information output from retina/memory cells is encoded as a spike train consisting of two qualitatively different parts: a tag determined by its relative position on the retina/memory and a tag encoding the feature signalled by the cell. This information is processed by the matching neurons which act as spatiotemporal coincidence detectors; the task of NESTOR is to locate an object (memory) projected onto its artificial retina.

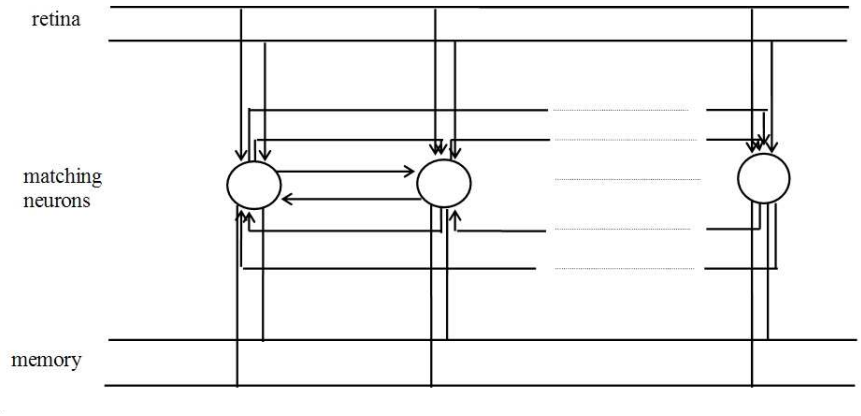

FIGURE 16. Structure of NESTOR

It is important to note that matching neurons obtain input from both artificial retina and memory and thus their operation is influenced by both bottom-up and top-down information. As Mumford notices [81], systems which depend on interaction between feedforward and feedback loops are quite distinct from models based on Marrś feedforward theory of vision.

Thus matching neurons are fully connected to both retina and memory neurons and accept for processing new information, contingent on their internal state (defined by the previously accepted spike train).

Each matching neuron maintains an internal representation (a hypothesis) defining a potential location of the memory on the retina and in operation simply conjoins the positional tags of the incoming spike trains from the retina/memory (corresponding to their retinotropic positions), with its own hypothesis and, dependent upon the result, distributes its successful or unsuccessful hypothesis to other matching neurons.

Effectively NESTOR is a connectionist implementation of the swarm intelligence (SI) paradigm Stochastic Diffusion Search, (SDS) [17]; a simple agent based SI matching process whose operation depends on cooperation and competition in a population of agents which are realised in NESTOR as the matching neurons which instantiate a form of dynamic assembly knowledge encoding.

Because NESTOR processes information as Inter-Spike Intervals, it naturally represents and processes high arity predicates avoiding one of the highlighted limitations of classical connectionism as a model of cognitive processes (see Section 11.8).

10.5. **Deep learning.** In the context of connectionism, 'Deep Learning' defines a class of artificial neural network algorithms that attempt to learn training data at multiple levels, corresponding to different levels of abstraction. The first form of deep learning algorithm can be traced to Fukushima's Neocognitron [44] but the term 'deep learning'

did not really begin to emerge widely until the work of Hinton, who in 2007 demonstrated that "It is much more sensible first to learn a generative model that infers the hidden variables from the sensory data and then to learn the simpler mapping from the hidden variables to the labels" [53]. In this work Hinton concluded that "a combination of three ideas leads to a novel and effective way of learning multiple layers of representation.

- The first idea is to learn a model that generates sensory data rather than classifying it.
- The second idea is to learn one layer of representation at a time. This decomposes the overall learning task into multiple simpler tasks and eliminates the inference problems that arise in directed generative models.
- The third idea is to use a separate fine tuning stage to improve the generative or discriminative abilities of the composite model."

Deep learning algorithms have been shown to perform extremely well on difficult machine learning problems. For example, in 2010 Ciresan et al demonstrated that standard back propagation learning, applied in deep non-linear networks, can outperform all previous techniques on the MNIST handwritten digit benchmark, even without unsupervised pre-training of hidden layer [34]. Similarly Hinton reports other versions of the approach being successfully applied to tasks as diverse as de-noising images, retrieving documents, extracting optical ow, predicting the next word in a sentence and predicting what movies people will like [53].

The potential of deep learning is particularly powerfully demonstrated in the work of the 'Google Brain' team, led by Andrew Ng and Jeff Dean who, in 2007, created a deep learning network that successfully learned to recognise higher-level concepts (e.g. human faces and cats) from unlabelled video streams [88].

11. "What artificial neural networks cannot do .."

I conclude this chapter with a brief discussion of six philosophical issues relating to connectionism; issues I have loosely delineated 'epistemic' and 'ontological':

'Epistemic' issues: An examination of three 'philosophico-technical' problematic themes relating to computational connectionism:
(1) Minsky and Papert's 1969 critique of single layer perceptrons.
(2) Fodor and Pylyshyn's 1988 critique of connectionism per se.
(3) Nasuto's 1998 critique of uni-variate knowledge representation in classical connectionism.

'Ontological' issues: An examination of three modes pertaining to computational connectionism:
(1) Searle's 1980 critique of computational 'understanding'.
(2) Penrose's 1989 critique of computational [mathematical] 'insight'.
(3) Bishop's 2002 critique of computational 'consciousness'.

11.1. **What the [single layer] perceptron cannot do.**

The first of the 'epistemic' themes relates to the single layer perceptron. One year at the tale end of the sixties, 1969, witnessed the publication of what perhaps must rate as one of the most influential texts ever published in the field of neural computing '*Perceptrons: an introduction to computational geometry*' by Marvin Minsky and Seymour Papert (albeit although

'Perceptrons' was certainly hugely influential, some commentators have suggested that it plays much the same role in the evolution of A.I. as 'The Necronomicon' does in the world of H.P. Lovecraft - as a book often cited but rarely read).

Ironically, at the time of its publication, some viewed 'Perceptrons' not so much as a text which would effectively close-off the field of connectionism for the next decade or more, but as a text which would help establish the nascent field of neural computing - and perception studies in particular - on much firmer, more mathematically rigorous, ground. Thus Jurg Nievergelt's contemporaneous review (IEEE Transaction on computers, June 1969, pp. 572) concludes:

> Minsky and Papert's 'Perceptrons' is an important contribution to the growing stock of mathematical models of computing devices. It establishes perceptron theory on a rigorous foundation, *and will probably generate a revival of interest in perceptrons.* It should also attract the attention of automata theorists to the subject, who may in the past have been distracted by the lack of mathematical rigor with which perceptrons have usually been treated.

Nievergelt identifies the main focus of the book as an analysis of the computational power of the single layer perception (SLP) and reports its two key conclusions as being: (a) that a [single layer] diameter limited perceptron cannot compute the 'connectedness' predicate and (b) that a single layer [fixed order] perceptron cannot compute the k-input 'parity' (XOR) problem.

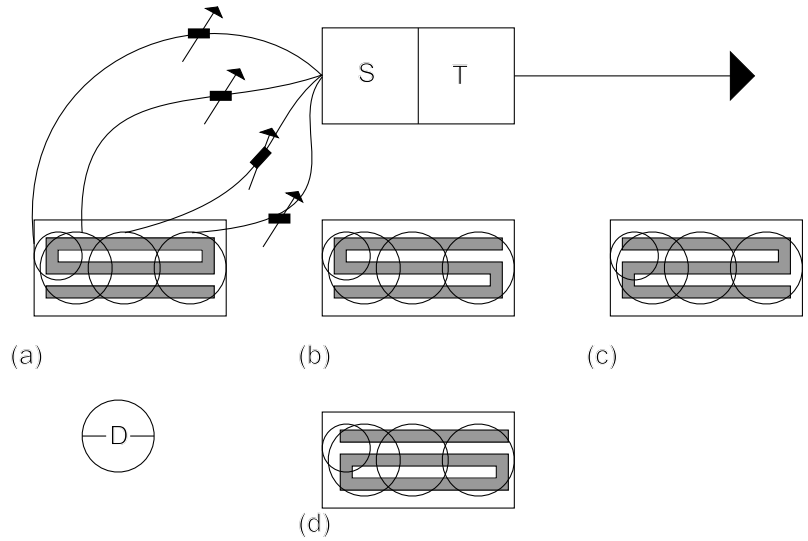

FIGURE 17. Can a diameter limited Perceptron compute the 'connectedness' predicate

11.2. **The 'connectedness' predicate.** A 'diameter limited perceptron (DLP)' is a perceptron in which the inputs to its A-Units must all fall within a receptive field of size D.

In Figure 17 the A-Units can naturally be divided into three groups: those on the *left*, the *middle* and the *right* of the image and only images (b) and (c) are connected. Hence to compute the 'connectedness' predicate the weights of the perception must be configured such that the perceptron will fire only on presentation of images (b) and (c).

- In considering images (a) & (c) it is apparent that only the left group A-units are positions to identify the difference between the two classes; thus to identify the connected image there need to be higher weights activated by the left A-Units in image (c) than image (a).
- Similarly, in images (a) & (b) it is only the right group that can tell the difference, hence to identify connectedness there need to be higher weights activated by the right A-Units in image (b) compared to image (a).
- However the above two requirements give image (d) higher activation than either image (b) or image (c); this implies that if a threshold is found that can classify (b) & (c) as connected, then it will also - *incorrectly* - classify image (d) as connected.

Thus the diameter limited perceptron cannot compute the 'connectedness' predicate.

11.3. **The 'order' of a perceptron.** The order of a perceptron is defined as the largest number of inputs to any of its A-Units; for perceptrons to be useful this 'order' must be fixed at some value smaller than the size of the retina. Consider a simple problem: to design a perceptron to recognise (fire) if there is one or more groups of $[2 \times 2]$ black pixels on its input retina, (see Figure 18).

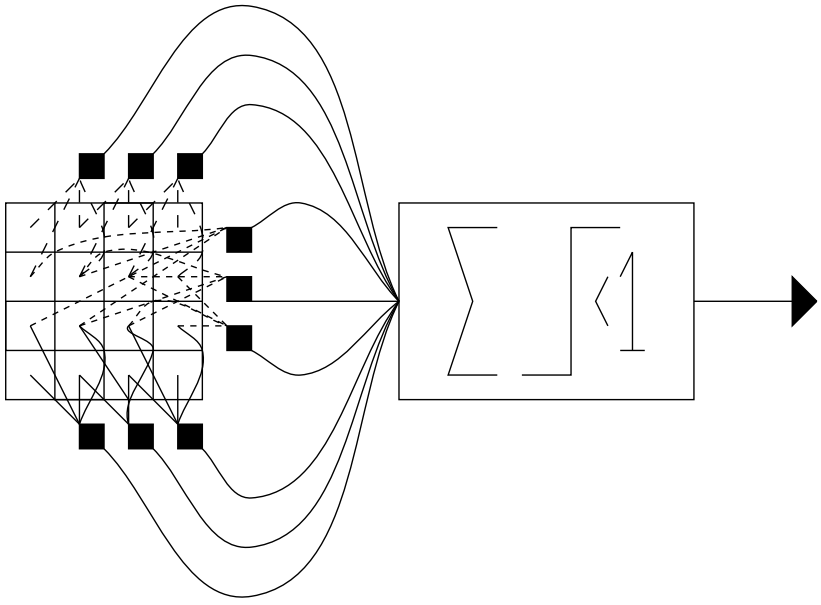

FIGURE 18. The $[2 \times 2]$'blob' detecting Perceptron

This 'blob detection' problem requires that the perceptron has as many A-Units as there are pixels on the retina (fewer duplications due to edge effects). Each A-Unit

covers a $[2 \times 2]$ square and computes the AND of its inputs. Clearly, if all the weights to the R-Unit are unity and the threshold is just lower than unity, then the perceptron will fire if there is a black square anywhere on the retina. The order of a perceptron to solve this problem is thus four $O(4)$ and it is clear that this remains constant irrespective of the size of the retina.

11.4. **The 'odd-parity' problem.** To compute the 'odd parity' problem the perceptron must fire if the number of active elements on its retina sums to an odd number.

11.4.1. *Can an order (1) perceptron solve the odd parity problem?* Consider the 3-input odd parity problem, (see Figure 19). An 'order (1) perceptron' is functionally equivalent to a simple MCP cell. To solve the odd parity problem an order (1) perceptron with three A-Units (A_1, A_2 and A_3) must fire - output one - when the number of active elements on the retina is odd.

ID		S1	S2	S3		R
R0		0	0	0		0
R1		0	0	1		1
R2		0	1	0		1
R3		0	1	1		0
R4		1	0	0		1
R5		1	0	1		0
R6		1	1	0		0
R7		1	1	1		1

FIGURE 19. The 3-input 'odd parity' function

The desired perceptron response for a retina of size three is depicted in Figure 19. Thus to depict the function described in Figure 19 the perceptron must learn a set of weights, W_1, W_2, W_3 that satisfy the following constrain:

$$(W_1 S_1 + W_2 S_2 + W_3 S_3 > T) = Response \tag{43}$$

Case (R1) implies $W_3 > T$ since it is the only active weight. Similarly case (R2) implies $W_2 > T$. However case (R3) requires $W_3 + W_2 < T$. This is a contradiction, hence an order (1) perceptron cannot solve the three input parity problem.

11.4.2. *Can an order (2) perceptron) solve odd parity?* To determine if there is any order(2) perceptron that can solve 3-input odd parity it is necessary to show that a 'mask perceptron' cannot solve 3-input odd parity, as a mask perceptron is more general than any perceptron with specific 'hand engineered' A-Units.

A mask perceptron of order (2) has one A-unit for every possible pair of points of its input (three) retina. I.e. There is one mask for each of the possible two-input

combinations; for a retina of size three $[S_1, S_2, S_3]$ there are three possible combinations of two inputs mapping onto the input retina $S_{[1..3]}$: $a_{[S_1S_2]}, b_{[S_1S_3]}$ and $c_{[S_2S_3]}$; and for each pair of inputs a, b and c there are four two input mask A-units each of which will detect one of the four possible two input patterns: [0 0] [0 1] [1 0] [1 1].

Thus each one of these four possible input combinations is detected by the mask, and each of the twelve (3x4) A-Units will only fire if that particular combination is present at its positions on the input retina. If it can be proved that no O (2) mask perceptron can accomplish the task, then no other O (2) perceptron can.

The firing patterns of the twelve A-Units, for the eight possible input patterns (R0 .. R7), are shown in Figure 20; where A_n is the weight to A-Unit A_n.

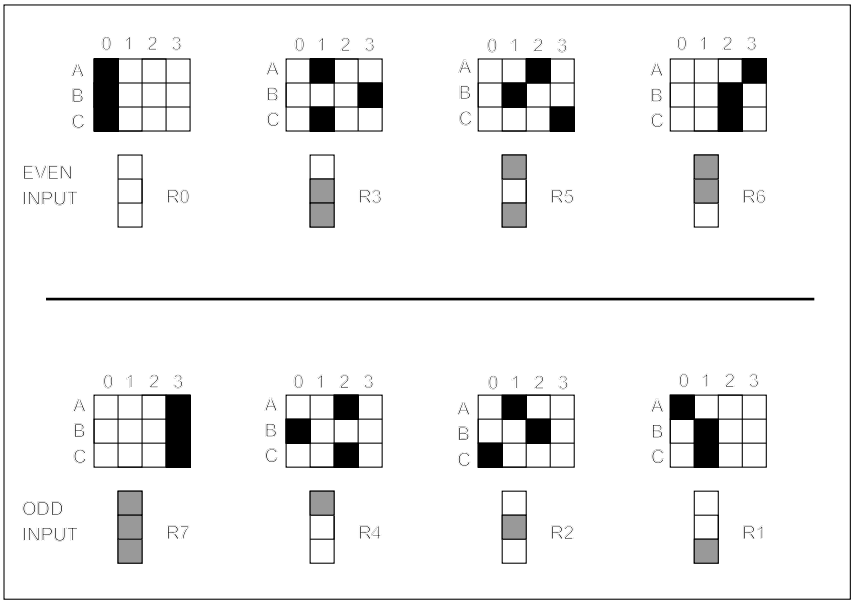

FIGURE 20. Response of A-Unit masks to possible input patterns

$$(44)(R7 > T \& R0 < T) \implies (R7 > R0) \implies (A3 + B3 + C3 > A0 + B0 + C0)$$
$$(45)(R4 > R5) \implies A2 + B0 + C2 > A2 + B1 + C3 \implies B0 + C2 > B1 + C3$$
$$(46) \qquad R4 > R6 \implies A2 + B0 > A3 + B2$$
$$(47) \qquad R2 > R3 \implies B2 + C0 > B3 + C1$$
$$(48) \qquad R2 > R6 \implies A1 + C0 > A3 + C2$$
$$(49) \qquad R1 > R3 \implies A0 + B1 > A1 + B3$$
$$(50) \qquad R1 > R5 \implies A0 + C1 > A2 + C3$$

Adding inequalities [45 .. 50] together :

$$2A0 + A1 + A2 + 2B0 + B1 + B2 + 2C0 + C1 + C2 > A1 + A2 + 2A3 + B1 + B2 + 2B3 + C1 + C2 + 2C3$$

$$\implies 2A0 + 2B0 + 2C0 > 2A3 + 2B3 + 2C3 \tag{51}$$

$$\implies (A0 + B0 + C0 > A3 + B3 + C3) \tag{52}$$

But (52) this contradicts (44), hence the task is not solvable by an order (2) mask perceptron.

11.4.3. *Can an order (3) perceptron solve odd parity?* It is trivial to solve 3-input odd parity with a perceptron of order (3). To perform this function it is merely necessary to ensure there is one A-Unit tuned to detect each example of odd parity on the retina, with the threshold of the R-Unit set low enough such that the R-Unit will fire *iff* any one A-Unit becomes active.

In their book *Perceptrons: an introduction to computational geometry*, Minsky and Papert famously proved that what has been shown for a retina of size 3 is true for any K-input retina; the order of K-bit parity detection is at least K.

11.5. **Linearly separable problems.** Two sets of points in a two-dimensional space are linearly separable if they can be completely separated by a single line; generalising, two sets of points are linearly separable in n-dimensional space if there exists a hyperplane that can separate them.

The output function of a two input MCP cell can be represented in two dimensions, using the x-axis for one input and the y-axis for the other. The firing equation for a two input MCP cell ($X_1W_1 + X_2W_2 > T$) can be represented by a line dividing the two dimensional input space into two areas. Hence a two input MCP cell can classify any two-input function that can be separated by a straight line dividing the input space. Figure 21 clearly illustrates that the two input OR function is linearly separable.

$$X_1W_1 + X_2W_2 = T \tag{53}$$

$$X_2 = \frac{T - X_1W_1}{W_2} \tag{54}$$

11.6. **Linearly inseparable problems.** There are many problems that cannot be linearly divided; the so called 'Hard Problems' of perceptron learning. Perhaps the most famous example of this class of problem is the 'XOR' problem, (see Figure 21).

Figure (22) demonstrates that the two input XOR function is not linearly separable in two dimensions; however, the XOR problem *can* be made linearly separable by adding another input. Thus, given two inputs (a) and (b), adding a third input c (where $c = a \wedge b$) maps the problem into a three dimensional input space. Figure 23 clearly shows that in this 3D input space, there exists a plane to separate the two classes and hence that the two input XOR problem is linearly separable in three dimensions.

In their book 'Perceptrons', Minsky & Papert demonstrated that there were many simple tasks that could not be performed by Single Layer Perceptrons (SLPs) of fixed order, although each of the problems was relatively trivial to compute using 'conventional' algorithmic methods. It was this result that many believe instigated the mass migration of research effort away from machine learning and artificial neural network paradigm, towards what is now colloquially termed 'Good Old Fashioned Artificial Intelligence'

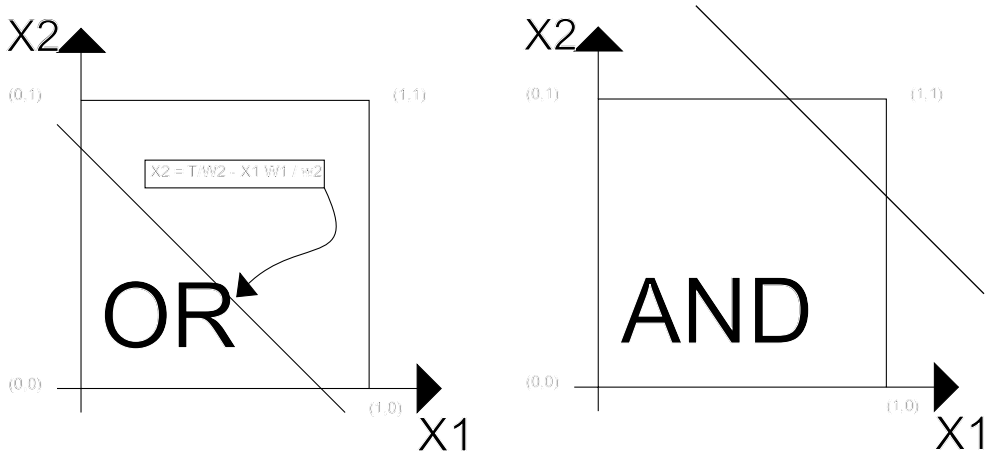

FIGURE 21. The two input OR and AND functions are linearly separable

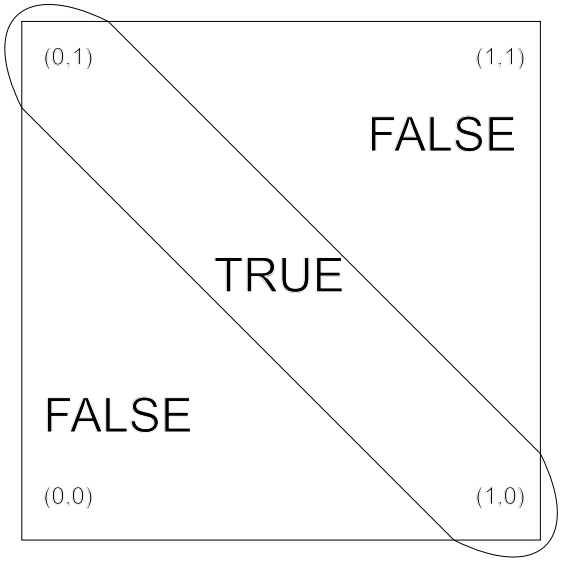

FIGURE 22. The two input XOR function is not linearly separable

(GOFAI). It was not until the development of efficient learning rules for multi-layer networks that Minsky & Papert's 'hard problems' became [relatively] 'easy' problems for artificial neural networks to solve.

11.7. **Fodor & Pylyshyn.** In 1988, in a comprehensive, trenchant and hugely influential critique by Fodor and Pylyshyn [41], the authors "threw a scare into the field of connectionism, at least for a moment. Two distinguished figures, from the right side of the tracks, were bringing the full force of their experience with the computational approach to cognition to bear on this young, innocent field" [29]. In particular, Fodor and Pylyshyn (ibid) highlighted that:

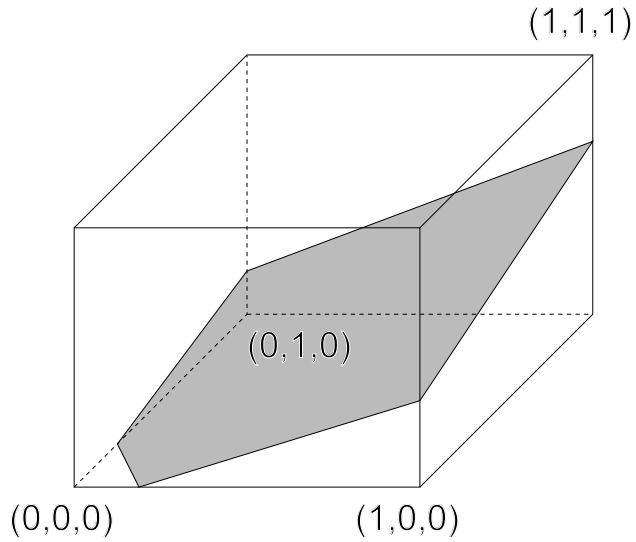

FIGURE 23. Mapping the two input XOR function into a 3D space

.. Classical and connectionist theories disagree about the nature of mental representation; for the former, but not for the latter, mental representations characteristically exhibit a combinatorial constituent structure and a combinatorial semantics. Classical and connectionist theories also disagree about the nature of mental processes; for the former, but not for the later, mental processes are characteristically sensitive to the combinatorial structure of the representations on which they operate.

In his paper 'The constituent structure' [116], Paul Smolensky summarises the central arguments of Fodor and Pylyshyn's paper as follows:

(1) Thoughts have composite structure. By this they mean things like: the thought that John loves the girl is not atomic; it is a composite mental state built out of thoughts about John, loves, and the girl.
(2) Mental processes are sensitive to this composite structure. For example, from any thought of the form $p \wedge q$ - regardless of what p and q are - we can deduce p.

Fodor and Pylyshyn elevate (1) and (2) to the status of defining the Classical View of Cognition, and they want to say that this is what is being challenged by the connectionists. I will later argue that they are wrong, but now we continue with their argument ..

.. Now for Fodor and Pylyshyn's analysis of connectionism. They assert that in (standard) connectionism, all representations are atomic; mental states have no composite structure, violating (1). Furthermore, they assert, (standard) connectionist processing is association which is sensitive only to statistics, not to structure-in violation of (2). Therefore, they conclude, (standard) connectionism is maximally non-Classical; it

violates both the defining principles. Therefore connectionism is defeated by the compelling arguments in favour of the Classical View.

In practice this yields the following problems for connectionism: 'productivity', 'systematicity', 'compositionality and inferential coherence':

Productivity: Typically in symbolic approaches to AI, linguistic competence - following Noam Chomsky (1968, "Language and mind") - is conceived as generative; this allows the AI system to generate and understand a potentially unbounded number of sentences.

Fodor and Pylyshyn state the productivity problem as follows:

> There is a classical productivity argument for the existence of combinatorial structure in any rich representational system (including natural languages and the language of thought). The representational capacities of such a system are, by assumption, unbounded under appropriate idealization; in particular, there are indefinitely many propositions which the system can encode. However, this unbounded expressive power must presumably be achieved by finite means. The way to do this is to treat the system of representations as consisting of expressions belonging to a generated set. More precisely, the correspondence between a representation and the proposition it expresses is, in arbitrarily many cases, built up recursively out of correspondences between parts of the expression and parts of the proposition. But, of course, this strategy can operate only when an unbounded number of the expressions are non-atomic. So linguistic (and mental) representations must constitute symbol systems. So the mind cannot be a PDP.

Systematicity: You don't find minds that are prepared to infer that "John went to the store" from (a) "John, Mary, Susan and Sally went to the store", but not from (b) "John, Mary and Susan went to the store".

Compositionally: It is, for example, only insofar as 'the girl', 'loves' and 'John' make the same semantic contribution to "John loves the girl", that they make to "the girl loves John", that understanding the one sentence implies understanding the other; in this way compositionally is also closely related to systematicity.

It is clear from their definitions that productivity and systematicity arguments are essentially the same and are fundamentally related to compositionality. However, classical connectionism recognises no combinatorial structure in its mental representations (e.g. it doesn't fit semantic and syntactic structure into its representations) which makes it very difficult to see how to instantiate systematicity and productivity on a classical connectionist architecture; leading Fodor and Pylyshyn [41] to conclude:

> ...if you need structure [compositionality] in mental representations anyway to account for the productivity and systematicity of minds, why not postulate mental processes that are structure sensitive to account for the coherence of mental processes? Why not be a Classicist, in short?

Subsequent 'refutations' of the Fodor and Pylyshyn's critique generally took two forms: argument, or counterexample.

In 1990 David Chalmers [29] remarked, "One is reminded of Nietzsche's observation: *It is not the least charm of a theory that it is refutable*" and subsequently offered up perhaps the simplest a priori refutation of Fodor and Pylyshyn's 'systematicity' critique:

(1) In F&P's argument that no connectionist models can have compositional semantics, there is no escape clause excluding certain models (such as Classical implementations) from the force of the conclusion. (By observation.)
(2) If F&P's argument is correct as it stands, then it establishes that no connectionist model can have compositional semantics. (From (1).)
(3) But some connectionist models obviously do have compositional semantics; namely, connectionist implementations of classical models. (By observation, accepted by all.)
(4) Therefore, F&P's argument is not correct as it stands. (From (2), (3).)

Chalmers concludes: "Fodor and Pylyshyn's arguments establish that compositionality exists, but for their arguments [...] to succeed, they would need to establish a rather stronger claim: that compositionally is everything", but this is precisely the claim Chalmers' argument (above) aims to refute.

Whether all of Fodor and Pylyshyn's carefully nuanced arguments are refuted quite as easily as Chalmers suggests remains a moot point; nevertheless, it is interesting to observe - twenty five years after their work was first published - the relatively small number of *fully connectionist* planning systems, proof provers, game playing etc. This offers at least partial empirical evidence that, to date, the philosophico-technical issues raised by Fodor and Pylyshyn (so simple to resolve using classical approaches) remain relatively difficult nuts for connectionists to crack. Thus, although a very strong neural network playing backgammon program was developed by Tesauro [119], this program effectively used the neural network for board evaluation (albeit recent work from Tanay et al [118] demonstrates a fully agent based system capable of playing a strong game of HeX).

11.8. **The representational power of uni-variate neural networks.** *"Nothing seems more certain to me than that people someday will come to the definite opinion that there is no copy in the ... nervous system which corresponds to a particular thought, or a particular idea, or memory"*, (Wittgenstein, 1948).

Over the hundred years since the publication of William James' Psychology [61], neuroscientists have attempted to define the fundamental features of the brain and its information processing capabilities in terms of mean firing rates at points in the brain cortex' (neurons) and computations. After Hubel and Wiesel [58], the function of the neuron as a specialised feature detector was treated as established doctrine. From this followed the functional specialisation paradigm, mapping different areas of the brain to specific cognitive function, reincarnating an era of modern phrenology (which Fodor [40], perhaps ironically, describes as,"*a pseudoscience focused on measurements of the human*

skull, based on the concept that the brain is the organ of the mind, and that certain brain areas have localized, specific functions or modules").

Connectionism mapped well onto the above assumptions. Its emergence is based on the belief that neurons can be treated as simple computational devices. The initial boolean McCulloch-Pitts model neuron [77] was quickly extended to allow for analogue computations. Further, the assumption that information is encoded in the mean firing rate of neurons was a central premise of all the sciences related to brain modelling.

Over the last half century such 'classical' connectionist networks have attracted significant interest. They are routinely applied to engineering problems [120], and as metaphors of concepts drawn from neuroscience, have also been offered as models of both high [103] and low level [4] [127] cognition. However the classical connectionist models of high level cognition have also been strongly criticised [38] and the situation at the domain of low level neural modelling is little better [1].

More recently spiking neuron, pulsed neural networks have begun to attract attention [71]. Such models no longer aggregate individual action potentials as a mean firing rate but act on temporal sequences of spikes. Like the classical connectionist models spiking neuron neural networks have been studied both for their computational/engineering properties and as models of neurological processes [126].

Although temporal coding of spike trains lends itself more readily to multi-variate information encoding than rate encoding, both are typically discussed in a uni-variate framework - an observation that also applies to classical connectionist frameworks. Furthermore, although such networks can represent 'type' knowledge (e.g. of the form 'Apple iPhone') say by the activation of a single neuron (or spatially distributed across the activation of a group of neurons), because their constituent neurons merely process uni-variate information, they can only easily instantiate 'tokens' in a very similar manner (e.g. by activation on a particular node [or group of neurons]); in other words, they typically process only 'arty zero' knowledge (an 'arity zero' predicate is one without an argument).

However, Nasuto et al argue [83] that arity zero predicates are insufficient for representation of many complex relationships. For example, such limitations make it difficult to interpret and analyse the network in terms of causal relationships; specifically it is difficult to imagine how such a system could develop symbolic representations and quantified logical inference [105]. Such deficiencies in the representation of complex knowledge by classical neural networks have long been recognised [111] [41] [12] [94].

Taking into account the above considerations in 1998 Nasuto et al [83] proposed NESTOR: a spiking neuron connectionist architecture whose constituent neurons inherently operate on rich (bi-variate) information encoded in spike trains, rather than as a simple mean firing rate (see Section 10.4.4). NESTOR offers an elegant method of representing the specific member of a class (a token) by the use of the arity one predicate CLASS (INDIVIDUAL); in general this requires the use of bi-variate information to identify both the CLASS and the INDIVIDUAL (and multi-variate information to encode ever higher order predicates).

One of the most pervading concepts underlying computational connectionist models of information processing in the brain is linear input integration of rate coded univariate information by neurons. After a suitable learning process this results in neuronal

structures that statically represent knowledge as a vector of real valued synaptic weights in Euclidean space $\mathbb{R}^N$. Although this general framework has contributed to the many successes of connectionism, for all but the most basic of cognitive processes, a more complex, multi-variate dynamic neural coding mechanism is required; one which would avoid spatially binding knowledge to a particular neuron (or group of neurons) and such an architecture has been instantiated in the NESTOR spiking neuron network.

11.9. **The Chinese room argument.** The first of the three 'ontological' modes relates to the possibility of genuinely instantiating 'understanding' in a computational connectionist system.

Perhaps the most well known critic of computational theories of mind is John Searle. His best-known work on machine understanding, first presented in the 1980 paper 'Minds, Brains & Programs' [107], has become known as the Chinese Room Argument (CRA). The central claim of the CRA is that computations alone are not sufficient to give rise to cognitive states, and hence that computational theories of mind cannot fully explain human cognition. More formally Searle stated that the CRA was an attempt to prove the truth of the premise:

- Syntax is not sufficient for semantics;

Which, together with the following two axioms:

- Programs are formal, (syntactical);
- Minds have semantics, (mental content);

... led Searle to conclude that 'programs are not minds' and hence that computationalism - the idea that the essence of thinking lies in computational processes and that such processes thereby underlie and explain conscious thinking - is false [110].

In the CRA Searle emphasises the distinction between syntax and semantics to argue that while computers can act in accordance to formal rules, they cannot be said to know the 'meaning' of the symbols they are manipulating, and hence cannot be credited with 'understanding' the results of the execution of programs those symbols compose. In short, Searle claims that while computational connectionism may *simulate* aspects of human cognition, it can never *instantiate* it. NB. Some commentators insist the CRA relates purely to linguistic understanding; it is my contention that a close reading of Searle's paper reveals that it targets *any* aspect of [machine] 'understanding'; specifically - in the 'Robot reply' - the understanding of 'actions in the world'.

In a 2002 volume further developing analysis on the CRA [20] comment ranged from Selmer Bringsjord who observed the CRA to be "arguably the 20th century's greatest philosophical polarizer" [25], to Rey who claims that in his definition of Strong AI, Searle "burdens the [Computational Representational Theory of Thought (Strong AI)] project with extraneous claims which any serious defender of it should reject" [99]. Nonetheless, continued academic interest in the argument thirty three years after it was first proposed - see [98] [129] [117] [100] [42] [43] [90] [46] - offers testament to Searle's core claim 'that programs are not sufficient for mind'.

It is beyond the scope of this chapter to do more than scratch the surface of the extensive literature on the CRA other than to note that, to date, the two most widely discussed responses to the CRA have been the 'Systems reply' and the 'Robot reply', (for a broad selection of essays detailing these and other critical arguments see Preston and

Bishop's edited collection '*Views into the Chinese room*' [95]); however in '*A view into the Chinese room*' Bishop [21] summarised Searle's Chinese Room Argument (CRA) as follows:

> In 1977 Schank and Abelson published information [106] on a program they created, which could accept a simple story and then answer questions about it, using a large set of rules, heuristics and scripts. By script they referred to a detailed description of a stereotypical event unfolding through time. For example, a system dealing with restaurant stories would have a set of scripts about typical events that happen in a restaurant: entering the restaurant; choosing a table; ordering food; paying the bill, and so on. In the wake of this and similar work in computing labs around the world, some of the more excitable proponents of artificial intelligence began to claim that such programs actually understood the stories they were given, and hence offered insight into human comprehension.
>
> It was precisely an attempt to expose the flaws in the statements emerging from these proselytising AI-niks, and more generally to demonstrate the inadequacy of the Turing test [*in what has become known as the 'standard interpretation' of the Turing test a human interrogator, interacting with two respondents via text alone, has to determine which of the responses is being generated by a suitably programmed computer and which is being generated by a human; if the interrogator cannot reliably do this then the computer is deemed to have 'passed' the Turing test*] which led Searle to formulate the Chinese Room Argument.
>
> The central claim of the CRA is that computations alone cannot in principle give rise to understanding, and that therefore computational theories of mind cannot fully explain human cognition. More formally, Searle stated that the CRA was an attempt to prove that syntax (rules for the correct formation of sentences:programs) is not sufficient for semantics (understanding). Combining this claim with those that programs are formal (syntactical), whereas minds have semantics, led Searle to conclude that 'programs are not minds'.
>
> And yet it is clear that Searle believes that there is no barrier in principle to the notion that a machine can think and understand; indeed in MBP Searle explicitly states, in answer to the question 'Can a machine think?', that 'the answer is, obviously, yes. We are precisely such machines'. Clearly Searle did not intend the CRA to target machine intelligence *per se*, but rather any form of artificial intelligence according to which a machine could have genuine mental states (e.g. understanding Chinese) purely in virtue of executing an appropriate series of computations: what Searle termed 'Strong AI'.
>
> Searle argues that understanding, of say a Chinese story, can never arise purely as a result of following the procedures prescribed by any computer program, for Searle offers a first-person tale outlining how he

could instantiate such a program, and act as the Central Processing Unit of a computer, produce correct internal and external state transitions, pass a Turing test for understanding Chinese, and yet still not understand a word of Chinese.

Searle describes a situation whereby he is locked in a room and presented with a large batch of papers covered with Chinese writing that he does not understand. Indeed, the monoglot Searle does not even recognise the symbols as being Chinese, as distinct from say Japanese or simply meaningless patterns. Later Searle is given a second batch of Chinese symbols, together with a set of rules (in English) that describe an effective method (algorithm) for correlating the second batch with the first, purely by their form or shape. Finally he is given a third batch of Chinese symbols together with another set of rules (in English) to enable him to correlate the third batch with the first two, and these rules instruct him how to return certain sets of shapes (Chinese symbols) in response to certain symbols given in the third batch.

Unknown to Searle, the people outside the room call the first batch of Chinese symbols 'the script', the second set 'the story', the third 'questions about the story' and the symbols he returns they call 'answers to the questions about the story'. The set of rules he is obeying they call 'the program'. To complicate matters further, the people outside the room also give Searle stories in English and ask him questions about these stories in English, to which he can reply in English.

After a while Searle gets so good at following the instructions, and the 'outsiders' get so good at supplying the rules he has to follow, that the answers he gives to the questions in Chinese symbols become indistinguishable from those a true Chinese person might give.

From an external point of view, the answers to the two sets of questions, one in English the other in Chinese, are equally good; Searle, in the Chinese room, has passed the Turing test. Yet in the Chinese language case, Searle behaves 'like a computer' and does not understand either the questions he is given or the answers he returns, whereas in the English case, *ex hypothesi*, he does. [To highlight the difference consider Searle is passed a joke first in Chinese and then in English. In the former case Searle-in-the-room might correctly output appropriate Chinese ideograms signifying "HA HA" whilst remaining phenomenologically unmoved whilst in the latter, if the joke is funny, he may laugh out loud and 'feel the joke' within]. Searle contrasts the claim posed by some members of the AI community - that any machine capable of following such instructions can genuinely understand the story, the questions and answers - with his own continuing inability to understand a word of Chinese; for Searle the Chinese symbols forever remain 'ungrounded'.

NB. The 'symbol-grounding' problem' [50] is closely related to the problem of how words (symbols) get their meanings. On its own the meaning of a word on a page is 'ungrounded' and merely looking it up in a dictionary doesn't help ground it. If one

attempts to look up the meaning of an unknown word in a [unilingual] dictionary of a language one does not already understand, one simply wanders endlessly from one meaningless definition to another (a problem not unfamiliar to young children); like Searle in his Chinese room, the search for meaning remains forever 'ungrounded'.

The thirty three years since its inception [107] have seen many reactions to the Chinese Room Argument from the computational, cognitive science, philosophical and psychological communities, with perhaps the most widely held being based on what has become known as the 'Systems Reply'. This concedes that, although the person in the room doesn't understand Chinese, the entire system (of the room, the person and its contents) does.

Searle finds this response entirely unsatisfactory and responds by allowing the person in the room to internalise everything (the rules, the batches of paper etc) so that there is nothing in the system not internalised within Searle. Now in response to the questions in Chinese and English there are two subsystems, the native English speaking Searle and the internalised Chinese room but all the same, he [Searle] continues to understand nothing of Chinese, and a fortiori neither does the system, because there isn't anything in the system that is not just a part of him.

11.9.1. *Brain Simulation and the Chinese room.* In [107] Searle presciently anticipated the construction of a large, hi-fidelity simulation of the brain as envisaged in Henry Markram's Human Brain Project; in considering connectionist systems Searle writes that instead of a monolingual man in a room shuffling symbols we should imagine that:

> .. the man operates an elaborate set of water pipes with valves connecting them. When the man receives the Chinese symbols, he looks up in the program, written in English, which valves he has to turn on and off. Each water connection corresponds to a synapse in the Chinese brain, and the whole system is rigged up so that after doing all the right firings, that is after turning on all the right faucets, the Chinese answers pop out at the output end of the series of pipes.
>
> Now where is the understanding in this system? It takes Chinese as input, it simulates the formal structure of the synapses of the Chinese brain, and it gives Chinese as output. But the man certainly doesn't understand Chinese, and neither do the water pipes, and if we are tempted to adopt what I think is the absurd view that somehow the conjunction of man and water pipes understands, remember that in principle the man can internalise the formal structure of the water pipes and do all the "neuron firings" in his imagination. The problem with the brain simulator is that it is simulating the wrong things about the brain. As long as it simulates only the formal structure of the sequence of neuron firings at the synapses, it won't have simulated what matters about the brain, namely its causal properties, its ability to produce intentional states.

Some years later, in his invention of a 'Chinese gym' Searle [108] turned his fire to Artificial Recurrent Neural Networks and dynamic systems:

> What is more, the connectionist system is subject even on its own terms to a variant of the objection presented by the original Chinese room argument. Imagine that instead of a Chinese room, I have a Chinese gym: a hall containing many monolingual, English speaking men. These men would carry out the same operations as the nodes and synapses in a connectionist architecture, as described by the Churchlands, and the outcome would be the same as having one man manipulate symbols according to a rulebook. No one in the gym speaks a word of Chinese and there is no way for the system as a whole to learn the meaning of any Chinese words. Yet with appropriate adjustments, the system could give the correct answers to Chinese questions.

And in [131] Michael Wheeler offers a simple reconceptualisation of Searle's Chinese gym such that it fully targets even the 'Super-Turing' ARNN's described by Siegelmann:

> The counter-move that we might expect from Searle would be a simple re-staging of the Chinese gym as the dynamical Chinese gym, a hall in which many interacting monolingual English speakers perform the kinds of varied state-transition functions and time-dependent message-sending operations standardly performed by the units in a dynamical neural network. One can almost hear Searle's scorn: no one in the dynamical gym speaks a word of Chinese, and there is no way for the system as a whole to learn the meanings of Chinese words.

Thus the Chinese room and its connectionist extensions - first to a network of water pipes and secondly to the Chinese gym - suggest that no computational simulation of the brain will ever 'genuinely understand [Chinese]' and hence that computational connectionism must fail to provide an adequate model for cognition. As Wheeler summarises (ibid), "The reason why the shift to dynamical neural networks runs headlong into the dynamical Chinese gym is, of course, that that shift, as radical as it may be, still leaves us firmly within the **formalist** conceptual framework identified earlier".

11.10. **Computations and understanding: Gödelian arguments against computationalism.** Gödel's first incompleteness theorem states that, *"any effectively generated theory capable of expressing elementary arithmetic cannot be both consistent and complete. In particular, for any consistent, effectively generated formal theory F that proves certain basic arithmetic truths, there is an arithmetical statement that is true, but not provable in the theory."* The resulting true but unprovable statement $G(g)$ is often referred to as 'the Gödel sentence' for the theory, (albeit there are infinitely many other statements in the theory that share with the Gödel sentence the property of being true but not provable from the theory).

Arguments based on Gödel's first incompleteness theorem (initially from John Lucas [69] were first criticised by Paul Benacerraf [14] and subsequently extended, developed and widely popularised by Roger Penrose [91] [92] [93]) typically endeavour to show that for any such formal system F, humans can find the Gödel sentence $G(g)$ whilst the computation/machine (being itself bound by F) cannot. In [92] Penrose develops a subtler reformulation of this vanilla argument that purports to show that 'the human

mathematician can "see" that the Gödel Sentence is true for consistent F even though the consistent F cannot prove $G(g)$'.

A detailed discussion of Penrose's own take on the Gödelian argument is outside the scope of this chapter (for a critical introduction see [30] and for Penrose's response see [93]). Nonetheless, it is important to note that Gödelian style arguments, purporting to show computations are not necessary for cognition, have been extensively (e.g. Lucas maintains a web page <http://users.ox.ac.uk/~jrlucas/Godel/referenc.html> listing over fifty such criticisms) and vociferously critiqued in the literature (see [96] for a review); nevertheless interest in them - both positive and negative - continues to surface, (e.g. [121] [24]).

11.11. **Dancing with pixies.** Many people hold the view that, 'there is a crucial barrier between computer models of minds and real minds: the barrier of consciousness' and thus that computational connectionist simulations of mind (e.g. the huge, hi-fidelity simulation of the brain currently being instantiated in Henry Markram's 'Human Brain Project' - see Section 10.4.2) and 'phenomenal (conscious) experiences' are conceptually distinct [123].

But is consciousness a prerequisite for genuine cognition and the realisation of mental states? Certainly Searle believes so, "the study of the mind is the study of consciousness, in much the same sense that biology is the study of life" [109] and this observation leads him to postulate a 'connection principle' whereby, "... any mental state must be, at least in principle, capable of being brought to conscious awareness". Hence, if computational machines are not capable of enjoying consciousness, they are incapable of carrying genuine mental states and computational connectionist projects must ultimately fail as an adequate model for cognition.

In this final section of the chapter I briefly review a simple reductio ad absurdum argument that suggests there may be problems in granting phenomenal (conscious) experience to any computational system purely in virtue of its execution of a particular program; if correct the argument suggests either that strong computational accounts of consciousness (and a fortiori all computational connectionist accounts) must fail OR that panpsychism is true.

The argument - the 'Dancing with Pixies' (DwP) reductio - derives from ideas originally outlined by Hilary Putnam [97], Tim Maudlin [74], John Searle [108] and subsequently criticised by David Chalmers [31], Colin Klein [64] and Ron Chrisley [32] [33] amongst others, (for early discussion of these themes see Minds and Machines 4(4), 'What is Computation?', November 1994). In what follows, instead of seeking to justify Putnam's claim that "every open system implements every Finite State Automaton" (FSA) and hence that "psychological states of the brain cannot be functional states of a computer", I will simply establish the weaker result that, over a finite time window, every open physical system implements the trace of a Finite State Automata Q on fixed, specified input (I). That this result leads to panpsychism - the belief that the physical universe is composed of elements each of which is conscious - is clear as, equating FSA Q(I) to a specific computational system that is claimed to instantiate phenomenal states as it executes, and following Putnam's procedure, identical computational (and ex hypothesi phenomenal) states can be found in every open physical system.

Formally DwP is a reductio ad absurdum argument that endeavours to demonstrate that:

- IF 'an appropriately programmed computer really does instantiate genuine phenomenal states'
- THEN 'panpsychism holds'
  - *However, against the backdrop of our immense scientific knowledge of the closed physical world, and the corresponding widespread desire to explain everything ultimately in physical terms, panpsychism has come to seem an implausible view...*
- HENCE we are led to reject the assumed claim.

In Science and Science Fiction the hope is periodically reignited that a computer system will one day be conscious in virtue of its execution of an appropriate program; indeed in the UK the EPSRC awarded an Adventure Fund grant of £493,000 to a team of Robot-eers and Psychologists at Essex and Bristol, led by Owen Holland, with a goal of instantiating machine consciousness through appropriate computational internal modelling.

In contrast, below I outline a brief reductio style argument based on [20] [22] that either suggests such optimism is misplaced or that panpsychism is true.

In his 1950 paper, 'Computing Machinery and Intelligence', Turing defined Discrete State Machines (DSMs) as "machines that move in sudden jumps or clicks from one quite definite state to another", and explained that modern digital computers fall within the class of them. An example DSM from Turing is one that cycles through three computational states (Q_1, Q_2, Q_3) at discrete clock clicks. Such a device, which cycles through a linear series of state transitions 'like clockwork', may be implemented by a simple wheel-machine that revolves through 120^0 intervals.

By labelling the three discrete positions of the wheel (A, B, C) we can map computational states of the DSM (Q_1, Q_2, Q_3) to the physical positions of the wheel (A, B, C) such that, for example, $(A \implies Q_1; B \implies Q_2; C \implies Q_3)$. Clearly this mapping is observer relative: position A could map to Q_2 or Q_3 and, with other states appropriately assigned, the machine's function would be unchanged. In general, we can generate the behaviour of any K-state (input-less) DSM, $f(Q) \implies Q'$, by a K-state wheel-machine (e.g. a digital counter), and a function that maps each 'counter' state C_n to each computational state Q_n as required.

In addition, Turing's machine may be stopped by the application of a brake and whenever it enters a specific computational state a lamp will come on. Input to the machine is thus the state of the brake, $(I = ON|OFF)$, and its output, (Z), the state of the lamp. Hence the operation of a DSM with input is described by a series of contingent branching state transitions', which map from current state to next state $f(Q, I) \implies Q'$ and define output (in the Moore form) $f(Q') \implies Z$.

However, (over a finite time interval), defining the input to the device entails that such contingent behaviour reverts to clockwork, $f(Q) \implies Q'$. E.g. If Turing's DSM starts in Q_1 and the brake is OFF for two clicks, its behaviour, (execution trace), is fully described by the sequence of state transitions, $(Q_1; Q_2; Q_3)$. Hence, over a finite time window, if the input to a DSM is defined, we can map from each counter state C_n to each computational state Q_n, as required. And, following Putnam, in [20] I similarly

demonstrate how to map any computational state sequence with defined input onto the [non-repeating] internal states generated by any Open Physical System (OPS) (e.g. a rock).

Now, returning to a putative conscious robot: at the heart of such a beast there is a computational system - typically a microprocessor; memory and peripherals. Such a system is a DSM. Thus, with input to the robot fixed over a finite time interval, we can map its execution trace onto the state evolution of any digital counter or, ibid, any OPS. Hence, if the state evolution of a DSM instantiates phenomenal experience, then so must the state evolution of any OPS and we are inexorably led to a panpsychist worldview whereby disembodied phenomenal consciousnesses (aka 'pixies') are found everywhere.

12. Conclusions and perspectives

In Bishop [22] I review the three '*ontological*' arguments (discussed herein) that purport to show that computations are neither necessary nor sufficient for cognition; specifically that the execution of a computational connectionist simulation of the brain cannot instantiate genuine understanding or phenomenal consciousness (qua computations) and hence that there are limits to the use of the computational connectionist simulations in cognitive science. But perhaps this conclusion is too strong? E.g. How do the *a priori* arguments discussed herein accommodate the important results being obtained through computational neuroscience to cognition?

There are two responses to this question. The first suggests that there may be principled reasons why it may not be possible to adequately simulate all aspects of neuronal processing via a computational system; there are bounds to a [Turing machine based] computational neuroscience. Amongst others this position has been espoused by: (i) Penrose (see Section 11.10); (ii) Copeland who claims the belief that "the action of any continuous system can be approximated by a Turing Machine to any required degree of fineness ... is false" (Copeland's argument is detailed, but at heart he follows an extremely simple line of reasoning: consider an idealised analogue computer that can add two reals (a, b) and output one if they are the same, zero otherwise. Clearly either (a) or (b) could be non-computable numbers (in the specific formal sense of non Turing-computable numbers). Hence, clearly there exists no Turing machine that, for any finite precision (k), can decide the general function $F(a = b)$; see [35] for detailed discussion of the implications of this result); and (iii) those who suggest that (Turing) computational connectionist systems can only simulate 'well behaved' chaotic systems (e.g. Peter Smith outlines results from chaos theory which describe how 'Shadowing Theorems' limit the set of chaotic functions that a Turing machine can accurately model to those that are 'well-behaved' - albeit this will encompass most dynamical systems which contract phase space volumes [115]).

However in Section (11.10) I emphasised that Gödelian style arguments, purporting to show computations are not necessary for cognition, have been **extensively** criticised in the literature; and are endorsed by only a few, albeit in some cases very eminent, scholars. Nonetheless, Siegelmann [112] is confident that even if Gödelian argument is valid, ARNNs offer a simple, formal, reconciliation between computational connectionism and the mind:

> Our model may also be thought of as a possible answer to Penrose's recent claim [91] that the standard model of computing is not appropriate for modelling true biological intelligence. Penrose argues that physical processes, evolving at a quantum level, may result in computations which cannot be incorporated in Church's Thesis. The analog neural network does allow for non-Turing power while keeping track of computational constraints, and thus embeds a possible answer to Penrose's challenge within the framework of classical computer science.

A second response emerges from the Chinese room and the Dancing with Pixies *reductio*. It acknowledges the huge value that the computational metaphor plays in current psychology and neuroscience and concedes that a future computational neuroscience may be able to simulate *any* aspect of neuronal processing and offers insights into all the workings of the brain. However, although such a computational neuroscience will result in deep understanding of cognitive processes it insists on a fundamental ontological division between the *simulation of a thing* and *the thing itself*.

I.e. We may simulate the properties of gold using a computer program but such a program does not automatically confer upon us riches (unless of course the simulation becomes duplication; an identity). Hence Searle's famous observation that "*.. the idea that computer simulations could be the real thing ought to have seemed suspicious in the first place because the computer isn't confined to simulating mental operations, by any means. No one supposes that computer simulations of a five-alarm fire will burn the neighbourhood down or that a computer simulation of a rainstorm will leave us all drenched. Why on earth would anyone suppose that a computer simulation of understanding actually understood anything?*" [107]

Both of the above responses accommodate results from computational neuroscience, but the second specifically highlights continued shortcomings of any **purely formal** - vis a vis computational connectionist - account of cognition. If this is correct, perhaps it is finally time for Cognitive Science to take the body [and its physical and social embodiment] (c.f. Gibson [48], Varela et al [128], Bickhard & Terveen [16], Thompson [122] and Deacon [37]) a little more seriously ..

Acknowledgements. I would like to thank Hisao Ishibuchi for patience well beyond the call of duty. I would also like to thank the three anonymous reviewers for their help with the manuscript; Ron Chrisley for his many interesting criticisms and observations regarding the 'Dancing with Pixies' *reductio* and Slawek Nasuto for his insight, encyclopaedic knowledge and helpful comments over many years. In preparing this chapter I extracted some elements from my previous work, notably [21] [86] & [22]. General background and historical context was, in part, derived from Aleksander & Morton [9], Harnish [51] and, of course, Wikipedia.

Glossary

McCulloch-Pitts neuron: a first - grossly simplified - mathematical model of neuron firing.
Neural Network: a network of adaptable nodes which, through a process of learning from task examples, store experiential knowledge and make it available for use.

Learning: is the process whereby the neural network attempts to iteratively optimise all its weights and thresholds (its *structure*) so as to minimise the overall prediction error across all the pairs of input-output vectors in its *training set*.
Recurrent Neural Networks: a class of neural network where connections between units form a directed cycle.
Supervised Learning Algorithm: the process of (typically iteratively) inferring network function from labeled training data.
Spiking Neural Networks: are networks which incorporate the concept of time into their behaviour.
Unsupervised learning algorithm: the task of finding 'hidden structure' (network function) in unlabelled [training] data
Reservoir computing: defines network function via a trained 'readout mechanism' over a fixed, richly connected, pool of processing nodes (the reservoir).

Nomenclature

ART: Adaptive Resonance Theory
ANN: Artificial Neural Network
BDN: Binary Discriminant Neuron
BP: Back Propagation [learning algorithm]
DwP: Dancing with Pixies
GDR: Generalised Delta Rule
LVQ: Learning Vector Quantisation
MCP: McCulloch-Pitts [neuron]
MLP: Multi-Layer Perceptron
NESTOR: NEural STochastic diffusion search netwORk
RAM: Random Access Memory
RBF: Radial Basis Function
SLP: Single Layer Perceptron
SDS: Stochastic Diffusion Search
SI: Swarm Intelligence
SNN: Spiking Neuron Networks

References

[1] Abbot, L., (1990), Learning in Neural Network Memories, Network: Computation in Neural Systems 1, pp. 105-122.
Offers a critique of neural modelling in classical connectionist models of low level cognition.

[2] Auer, P, Burgsteiner, H. & Maass, W., (2002), Reducing communication for distributed learning in neural networks, in: J. R. Dorronsoro (Ed.), (2002), Proc. of the International Conference on Artificial Neural Networks ICANN 2002, Vol. 2415 of Lecture Notes in Computer Science, pp. 123-128, Springer.
Auer suggests that to approximate a given input/output behaviour F on particular functions of time (or spike trains), it is simply necessary to randomly pick some sufficiently complex reservoir of recurrent circuits of spiking neurons, and then merely adapt the weights of a single pool of feedforward spiking neurons to approximate the desired target output behaviour.

[3] Ackley, D.H., Hinton, G,E & Sejnowski, T,J., (1985). A Learning Algorithm for Boltzmann Machines. Cognitive Science: 9 (1), pp. 147?169.
First description of Boltzmann machine learning algorithm for neural networks with hidden layers.

[4] Ahmad, S., (1991), VISIT: An efficient computational model of human visual attention, PhD Thesis, University of Illinois, USA.
Another critique of neural modelling in classical connectionist models of low level cognition.

[5] Aleksander, I, (1970), Microcircuit learning networks: hamming distance behaviour, Electronics Letters 6 (5).
Early introduction to the idea of using small microcircuit devices to implement Bledsoe and Browning's n-tuple recognition systems.

[6] Aleksander, I. & Stonham, T.J., (1979), "Guide to Pattern Recognition using Random Access Memories", Computers & Digital Techniques 2 (1), pp. 29-40, UK.
Early review of the n-tuple or weightless network paradigm.

[7] Aleksander, I, Thomas, W.V. & Bowden, P.A., (1984),"WISARD a Radical Step Forward in Image Recognition", Sensor Review, UK.
A description of a real-time vision system built using weightless networks.

[8] Aleksander, I & Morton, H., (1995), An Introduction to Neural Computing, Cengage Learning EMEA.
Introductory text on neural networks.

[9] Aleksander, I & Morton, H., (2012), Aristotle's laptop: the discovery of our informational mind, World Scientific Publishing, Singapore.
Research text outlining Aleksander and Morton's idea of the 'informational mind'.

[10] Bain, A, (1873), Mind and Body. The theories of their relation, D.Appleton & Co, NY.
In which Bain suggests that our thoughts and body activity result from interactions among neurons within the brain.

[11] Barto, A.G., Sutton, R. S. & Anderson, C. W. , (1983), Neuron like adaptive elements that can solve difficult learning control problems, IEEE Transactions on Systems, Man and Cybernetics SMC:13: pp. 834-846.
The classic introduction to reinforcement learning.

[12] Barnden, J., Pollack, J. (eds.), (1990), High-Level Connectionist Models, Ablex, Norwood NJ, USA.
Edited volume which includes critiques of neural modelling in classical connectionist models of high level cognition.

[13] Beer, R. D., (1995), On the dynamics of small continuous-time recurrent neural networks, Adaptive Behaviour 3(4), pp. 469-509.
A key introduction to Continuous Time Recurrent Neural Networks.

[14] Benacerraf, P., (1967), God, the Devil & Gödel, Monist 51.
Benacerraf's response to Lucas's Gödelian argument against materialism.

[15] Bickhard, M. H. (1993). Representational Content in Humans and Machines. Journal of Experimental and Theoretical Artificial Intelligence 5, pp. 285-333.
Mark Bickard highlighting the problem of symbol grounding in cognitive science.

[16] Bickhard, M. H. & Terveen, L.,(1995). Foundational Issues in Artificial Intelligence and Cognitive Science, North Holland.
Introduction to Mark Bickhard's interactionist account of cognition.

[17] Bishop, J.M.: Stochastic Searching Networks, Proc. 1st IEE Int. Conf. on Artificial Neural Networks, pp. 329-331, London, UK.
First description of the 'Stochastic Diffusion Search' algorithm.

[18] Bishop, J.M., Bushnell, M.J. & Westland, S., (1991), The Application of Neural Networks to Computer Recipe Prediction, Color, 16 (1), pp. 3-9.
Successful application of neural networks to colour recipe prediction.

[19] Bishop, J.M., Keating, D.A. & Mitchell, R.J., (1998), A Compound Eye for a Simple Robotic Insect, in Austin, J., RAM-Based Neural Networks, pp. 166-174, World Scientific.
The use of weightless neural network in mobile robot localisation.

[20] Bishop, J.M., (2002), Dancing with Pixies: strong artificial intelligence and panpyschism. in: Preston, J. & Bishop, J.M., (eds), (2002), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Oxford University Press, Oxford UK.
First publication of the Dancing with Pixies (DwP) reductio against machine consciousness.

[21] Bishop, J M, (2004), A view inside the Chinese room, Philosopher 28 (4), pp. 47–51.
Summary of Searle's Chinese room argument.

[22] Bishop, J.M. (2009), A Cognitive Computation fallacy? Cognition, computations and panpsychism, Cognitive Computation 1(3), pp. 221-233.
Critique of the ubiquitous computational metaphor in cognitive science.

[23] Bledsoe, W.W. & Browning, I., (1959), Pattern recognition and reading by machine, Proc. Eastern Joint Computer Conference, pp. 225-232.
First description of the n-tuple method for pattern classification later developed into the 'weightless neural network' paradigm by Igor Aleksander.

[24] Bringsjord, S., & Xiao, H., (2000), A refutation of Penrose' Gödelian case against artificial intelligence, J. Exp. Theo. Art. Int. 12, pp. 307-329.
Selmer Bringsjord's critique of Roger Penrose's Gödelian argument against artificial intelligence.

[25] Bringsjord, S., & Noel, R., (2002), Real Robots and the Missing Thought-Experiment in the Chinese Room Dialectic. in: Preston, J. & Bishop, J.M., (eds), (2002), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Oxford University Press, Oxford UK.
Selmer Bringsjord's exposition of a 'missing thought experiment' in Searle's Chinese room argument.

[26] Broomhead, D. S. & Lowe, D., (1988), Radial basis functions, multi-variable functional interpolation and adaptive networks (Technical report). Royal Signals and Radar Establishment RSRE 4148, UK.
First exposition of the Radial Basis Function network.

[27] Bryson, A.E., (1963), Optimal programming problems with inequality constraints. I: Necessary conditions for extremal solutions, J. AAAI, 1 (11), pp. 2544-2550.
Vapnik cites this paper by Bryson et al as the first publication of the 'back propagation' learning algorithm.

[28] Carpenter, G. & Grossberg, S., (1987), A massively parallel architecture for a self organising neural pattern recognition machine, Computer Vision, Graphics and Image Processing: 37, pp. 54-115.
Early publication of the ART-1 classifier.

[29] Chalmers, D., (1990), Why Fodor and Pylyshyn Were Wrong: The Simplest Refutation, In Proceedings of The Twelfth Annual Conference of the Cognitive Science Society, pp. 340-347.
David Chalmers' rebuttal of Fodor and Pylyshyn's critique of connectionism.

[30] Chalmers, D., (1995), Minds, Machines, And Mathematics A Review of Shadows of the Mind by Roger Penrose, PSYCHE, 2: 9. *David Chalmers' critique of Roger Penrose's Gödelian argument against artificial intelligence.*

[31] Chalmers, D.J., (1996), The Conscious Mind: In Search of a Fundamental Theory, Oxford University Press, Oxford, UK.
David Chalmers' classic monograph espousing his 'psycho-functionalist' theory of mind.

[32] Chrisley R., (1995), Why Everything Doesn't Realize Every Computation, Minds and Machines: Minds and Machines Volume 4, pp. 403-420.
Ron Chrisley addresses Hilary Putnam's critique of functionalism.

[33] Chrisley R., (2006), Counterfactual computational vehicles of consciousness, Toward a Science of Consciousness (April 4th-8th 2006), Tucson Arizona, USA.
Ron Chrisley addresses Bishop's 'Dancing with Pixies' reductio against machine consciousness.

[34] Ciresan, D.C., Meier, U. , Gambardella, L. M. Schmidhuber, J., (2010), Deep Big Simple Neural Nets For Handwritten Digit Recognition, Neural Computation 22 (12): pp. 3207?3220.
Deep learning networks applied to hand writing recognition.

[35] Copeland B.J., (1997), The broad conception of computation, American Behavioral Scientist 40 (6), pp. 690- 716.
Copeland discusses foundations of computing and outlines so called 'real-valued' super-Turing machines.

[36] Cortes, C. & Vapnik, V.N., (1995), Support-Vector Networks, Machine Learning 20 (3), pp. 273 - 297.
Early discussion of the Support Vector machine.

[37] Deacon, T, (2012), Incomplete Nature: How Mind Emerged from Matter, W. W. Norton & Company.
All encompassing monograph emphasising the importance of appropriate embodiment for the emergence of mind.

[38] Dinsmore, J., (1990), Symbolism and Connectionism: a difficult marriage, Technical Report TR 90-8, Southern University of Carbondale, USA.
The limitations of uni-variate knowledge representation in classical connectionist systems to arity zero predicates.

[39] Dreyfus, H.L. & Dreyfus, S.E., (1988), Making a mind versus modelling the brain: artificial intelligence at a branch point, Artificial Intelligence 117: 1.
Historical review of the battle between connectionism and GOFAI in cognitive science.

[40] Fodor, J.A., (1983), Modularity of Mind: An Essay on Faculty Psychology. The MIT Press.
In which Fodor postulates a vertical and modular psychological organisation underlying biologically coherent behaviours.

[41] Fodor, J.A., Pylyshyn, Z.W., (1988), Connectionism and Cognitive Architecture: a critical analysis, Cognition 28, pp. 3-72.
The classic critique suggesting serious limitations of connectionist systems in comparison to symbol processing systems.

[42] Freeman, A. (2003), Output still not really convinced, The Times Higher (April 11th 2003), UK.
' *Discussion of the Chinese room argument.*

[43] Freeman, A., (2004), The Chinese Room Comes of Age: a review of Preston & Bishop, Journal of Consciousness Studies 11 (5/6), pp. 156-158.
Discussion of the Chinese room argument and Preston/Bishop's edited text.

[44] Fukushima, K, (1980), Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics 36 (4), pp. 193?202.
The Neocognitron: one of the first 'deep learning' connectionist systems.

[45] Gardenfors, P, (2004), Conceptual Spaces as a Framework for Knowledge Representation, Mind and Matter 2 (2), pp. 9-27.
. On the problem of modelling representations.

[46] Garvey, J., (2003), A room with a view?, The Philosophers Magazine (3rd Quarter 2003), p. 61.
Discussion of the Chinese room argument and Preston/Bishop's edited text.

[47] Gerstner, W. & Kistler, W., (2002), Spiking Neuron Models: Single Neurons, Populations, Plasticity. Cambridge University Press, UK.
Classic text on spiking neuron connectionist modelling.

[48] Gibson, J.J., (1979), The Ecological Approach to Visual Perception. Boston: Houghton Mifflin.
Gibson's classical text outlining the radical 'ecological approach' to vision and visual perception.

[49] Grossberg, S., (1976), Adaptive Pattern Classification and Universal recording: II. Feedback, expectation, olfaction, and illusions, Bio. Cybernetics 23: pp. 187-202.
Early exposition of the ART classifier.

[50] Harnad S., (1990), The Symbol Grounding Problem. Physica D 42, pp. 335–346.
Harnad's key reference on the classical 'symbol grounding' problem.

[51] Harnish, S., (2001), Minds, Brains, Computers: An Historical Introduction to the Foundations of Cognitive Science, Wiley-Blackwell.
Excellent general introductory text to Cognitive Science.

[52] Hebb, D.O. (1949). The Organisation of Behaviour. New York: Wiley & Sons.
The original exposition of Hebbian learning.

[53] Hinton, G, (2007), Learning multiple layers of representation, Trends in Cognitive Science 11 (10), pp. 428-434.
Hinton's discussion of deep learning.

[54] Hodgkin, A. L. & Huxley, A. F., (1952), A quantitative description of membrane current and its application to conduction and excitation in nerve. Journal of Physiology 117 (4), pp. 500?544.
The classic paper on the detailed Hodgkin-Huxley model of the [squid] neuron.

[55] Hopfield, J..J., (1982), Neural networks and physical systems with emergent collective computational abilities, Proc. Nat. Acad. Sci. 79 (8), pp. 2554-2558.
Outlines the use of methods from physics to analyse properties of neural systems.

[56] Hopfield, J. J. (1984) Neurons with graded response have collective computational properties like those of two-state neurons. Proc. Nat. Acad. Sci. 81, pp. 3088-3092.
The binary Hopfield neural network.

[57] Hopfield, J. J. and Tank, D. W., (1985), Neural computation of decisions in optimization problems. Biological Cybernetics 55, pp. 141-146.
The use of Hopfield networks in optimisation.

[58] Hubel, D.H., Wiesel, T.N. (1962), Receptive fields, binocular interaction and functional architecture in the cat's visual cortex, Journal of Physiology 160 (1), pp, 106-154.
Hubel & Wiesel's classic paper on neuron behaviour.

[59] Irwin, G., Warwick, K. & Hunt, K., (eds), (1995), Neural Network Applications In Control. IET Digital Library, UK.
A selection of neural network applications.

[60] Jaeger , H. & Haas, H. (2004), Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication, Science 304: 5667, pp. 78-80.
On why echo state networks learn very efficiently.

[61] James, W, (1891), Psychology (briefer course), Holt, New York, USA.
Early classic text on the principles of psychology.

[62] Katevas, N., Sgouros, N.M., Tzafestas, S., Papakonstantinou G., Beattie, P., Bishop, J.M., Tsanakas, P & Koutsouris D., (1997), The Autonomous Mobile Robot SENARIO: A Sensor-Aided Intelligent Navigation for Powered Wheelchairs, IEEE Robotics and Automation 4 (4), pp. 60-70.
The use of Stochastic Diffusion in mobile robot localisation.

[63] Kleene, S. C., (1956), Representation of events in nerve nets and finite automata. In Automata Studies 34, Annals of Mathematics Studies, pp. 3?42. Princeton University Press, Princeton, NJ.
Before Minsky, Kleene proved that rational-weighted recurrent neural networks with boolean activation functions are computationally equivalent to classical finite state automata.

[64] Klein, C., (2004), Maudlin on Computation, (working paper).
On Maudlin's model of the implementation of computation.

[65] Kohonen, T., (1988), Self-Organising Maps, Springer.
Introduction to Kohonen feature maps.

[66] Kohonen, T, Barna, G. & Chrisley, R., (1988), Statistical Pattern Recognition with Neural Networks, In: Proceedings of the IEEE Conference on Neural Network 1: 61-68, 24-27 July 1988, San Diego, CA.
A discussion of practical issues relating to the Boltzmann machine and comparison of performance with other neural architectures.

[67] Le Cun, Y., (1985), Une procedure d'apprentissage pour reseau a seuil asymmetrique (a Learning Scheme for Asymmetric Threshold Networks), Proceedings of Cognitiva 85, pp. 599-604, Paris, France.
Independent derivation of the back propagation learning algorithm.

[68] Lighthill, J., (1973), Artificial Intelligence: A General Survey, in Artificial Intelligence: a paper symposium, Science Research Council, UK.
UK government-funded study providing a critique of GOFAI methods.

[69] Lucas, J.R., (1961), Minds, Machines & Gödel, Philosophy, 36.
Lucas outlines the use of Gödelian argument against materialism.

[70] Ludermir, T.B., de Carvalho, A., Braga, A.P. & de Souto, M.C.P., (1999), Weightless neural models: a review of current and past works, Neural Computing Surveys 2, pp. 41-61.
Recent review of the weightless neural network paradigm.

[71] Maass, W. & Bishop, C., (eds.), (1999), Pulsed Neural Networks, The MIT Press, Cambridge MA, USA.
Classic text on spiking neural networks.

[72] Maass, W, Natschlager, T. & Markram, H., (2002), Real-time computing without stable states: A new framework for neural computation based on perturbations, Neural Computation 14 (11), pp. 2531-2560.
Learning rules for Liquid State machines.

[73] Maass, W. & Markram, H., (2004), On the computational power of circuits of spiking neurons, Journal of Computer and System Sciences 69 (4), pp. 593?616.
On the use of the same connectionist hardware to compute different functions.

[74] Maudlin, T., (1989), Computation and Consciousness, Journal of Philosophy 86, pp. 407-432.
On the execution of computer programs.

[75] McCarthy, J., Minsky, M., Rochester, N. & Shannon, C., (1955), A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence. In: McCarthy, J., Minsky, M., Rochester, N. & Shannon, C., (2006) A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence August 31 1955, AI MAgazine 27(4): 12-14.
Reprint of the proposal for the 1956 Dartmouth Conference, along with the short autobiographical statements of the proposers.

[76] McCorduck, P., (1979), Machines who think, Freeman & Co., NY.
An account of the Dartmouth Conference.

[77] McCulloch, W.S., Pitts, W., (1943), A logical calculus immanent in nervous activity, Bulletin of Mathematical Biophysics 5, pp. 115-133.
On the computational power of the McCulloch-Pitts neuron model.

[78] Minsky, M., (1954), Neural Nets and the brain model problem. PhD thesis, Princeton, USA.
Early work from Minsky on a connectionist theory of mind.

[79] Minsky, M. L. (1967). Computation: finite and infinite machines. Prentice-Hall, Inc., Upper Saddle River, NJ, USA. *After Kleene, Minsky proved that rational-weighted recurrent neural networks with boolean activation functions are computationally equivalent to classical finite state automata.*

[80] Minsky, M & Papert, S., (1969), Perceptions: an introduction to computational geometry, The MIT Press.
Devastatining critique of the power of single layer perceptrons.

[81] Mumford, D., (1994), Neural Architectures for Pattern-theoretic Problems. In: Koch, Ch., Davies, J.L. (eds.), (1994), Large Scale Neuronal Theories of the Brain. The MIT Press, Cambridge MA.
In which Mumford observes that systems which depend on interaction between feedforward and feedback loops are quite distinct from models based on Marr's feedforward theory of vision.

[82] Nasuto, S.J., Bishop, J.M., Lauria, S., (1998), Time Complexity Analysis of the Stochastic Diffusion Search, Neural Computation '98, Vienna, Austria.
Derivation of the computational time complexity of stochastic diffusion search.

[83] Nasuto, S.J., Bishop, J.M.:, (1998), Neural Stochastic Diffusion Search Network - a theoretical solution to the binding problem, Proc. ASSC2, pp. 19-20, Bremen, Germany.
Examination of the binding problem through the lens of stochastic diffusion search.

[84] Nasuto, S.J., Dautenhahn K., Bishop, J.M., (1999), Communication as an Emergent Metaphor for Neuronal Operation. In: Nehaniv, C., (ed), (1999), Computation for Metaphors, Analogy, and Agents, Lecture Notes in Artificial Intelligence 1562, pp. 365-379. Springer.
Outlines a new metaphor grounded upon stochastic diffusion search for neural processing.

[85] Nasuto, S.J., Bishop, J.M., (1999), Convergence of the Stochastic Diffusion Search, Parallel Algorithms 14 (2), pp. 89-107.
Proof of the convergence of stochastic diffusion search to global optimum.

[86] Nasuto, S.J., Bishop, J.M. & De Meyer (2009), Communicating neurons: a connectionist spiking neuron implementation of stochastic diffusion search, Neurocomputing 72: pp. 4-6, pp. 704-712.
Review of a communication metaphor - based upon stochastic diffusion search - for neural processing.

[87] Newell, A., & Simon, H.A., (1976), Computer science as empirical inquiry: symbols and search, Communications of the ACM 19 (3), pp. 113-126.
Outlines the Physical Symbol System Hypothesis.

[88] Ng, A. & Dean, J., (2012), Building High-level Features Using Large Scale Unsupervised Learning, Proc. 29th Int. Conf. on Machine Learning, Edinburgh, Scotland, UK.
Deep learning neural network that successfully learned to recognise higher-level concepts (e.g. human faces and cats) from unlabelled video streams.

[89] Newell, A., Shaw, J.C. & Simon, H.A., (1960), A variety of intelligent learning in a General Problem Solver, in Yovits, M.C & Cameron, S., (eds), 'Self Organising Systems', Pergamon Press, NY.
Early discussion of the General Problem Solver.

[90] Overill, J., (2004), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Journal of Logic and Computation 14 (2), pp. 325-326.
Critique of the Chinese room argument.

[91] Penrose, R., (1989), The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics, Oxford University Press, Oxford UK.
Roger Penrose's text criticising computational artificial intelligence from a Gödelian perspective.

[92] Penrose, R., Shadows of the Mind: A Search for the Missing Science of Consciousness, Oxford University Press, Oxford UK.
Roger Penrose's second text criticising computational artificial intelligence from a Gödelian perspective.

[93] Penrose, R., Beyond the Doubting of a Shadow A Reply to Commentaries on Shadows of the Mind, PSYCHE, 2 (23).
Roger Penrose replies to criticism of his Gödelian arguments.

[94] Pinker, S. & Prince, A., (1988), On Language and Connectionism: Analysis of a Parallel Distributed Processing Model of Language Acquisition. In: Pinker, S., Mahler, J. (eds.), (1988), Connections and Symbols, The MIT Press, Cambridge MA.
Deficiencies in the representation of complex knowledge by classical neural networks .

[95] Preston, J. & Bishop, J.M., (eds), (2002), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Oxford University Press, Oxford UK.
Edited collection of essays on John Searle's Chinese room argument.

[96] Psyche, Symposium on Roger Penrose's Shadows of the Mind, http://psyche.cs.monash.edu.au/psyche-index-v2.html, PSYCHE VOL 2.
. Special issues on Roger Penrose's Gödelian arguments against AI.

[97] Putnam, H., (1988), Representation & Reality, Bradford Books, Cambridge MA.
The appendix contains Putnam's refutation of functionalism.

[98] Rapaport, W.J., (2006), Review of Preston, J. & Bishop, J.M., (eds), (2002), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Oxford University Press, Oxford UK. In: Australian Journal of Philosophy, 94(1), pp. 129-145.
Review of work on the Chinese room argument.

[99] Rey, G, (2002), Searle's Misunderstanding of Functionalism and Strong AI. In: Preston, J. & Bishop, J.M., (eds), (2002), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Oxford University Press, Oxford UK.
Criticism of Searle's Chinese room argument.

[100] Richeimer, J., (2004), Review of Preston, J. & Bishop, J.M., (eds), (2002), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Oxford University Press, Oxford UK, Philosophical Books 45 (2), pp.162-167.
Review of work on the Chinese room argument.

[101] Rosenblatt, F, (1958), Two theorems of statistical separability in the Perceptron. In: The Mechanisation of thought processes: proceedings of a symposium held at the National Physical Laboratory 1, pp. 419-449, HMSO London.
Early work by Frank Rosenblatt on the power of the Perceptron.

[102] Rosenblatt, F, (1962), The Principles of Neurodynamics, Spartan Books, New York.
Introduction to Perceptrons.

[103] Rumelhart, D.E., McClelland, J.L.., (1986), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, The MIT Press, Cambridge MA.
Seminal text which helped re-awaken interest in neural networks post Minsky & Papert's critique.

[104] Rumelhart, D.E.., Hinton, G.E. & Williams, R.J., (1986), Learning representations by back-propagating errors, Nature 323 (6088), pp. 533-536.
Independent derivation of the back-propagation algorithm.

[105] Russel, B., (1905), On Denoting, Mind XIV(4), pp. 479-493, Oxford UK.
On quantified logical inference.

[106] Schank, R C & Abelson, R P, (1977), Scripts, plans, goals and understanding: An inquiry into human knowledge structures, Psychology Press.
On the 'machine understanding' of stories.

[107] Searle, J. R., (1980), Minds, brains, and programs. Behavioral and Brain Sciences 3 (3): 417-457, Cambridge, UK.
The Chinese room argument.

[108] Searle, J., (1990), Is the Brain a Digital Computer?, Proceedings of the American Philosophical Association 64, pp.21-37.
Is computational syntax reducible to physics?.

[109] Searle, J., (1992), The Rediscovery of the Mind, pp. 227, MIT Press, Cambridge MA.
On computations and mind.

[110] Searle, J., (1994), The Mystery of Consciousness, Granta Books, London UK.
Interviews and reflections with a selection of twentieth century thinkers on consciousness.

[111] Sejnowski, T.J., Rosenberg, C.R., (1987), Parallel networks that learn to pronounce English text. Complex Systems 1, pp. 145-168.
Example of an ANN which learns to pronounce English text.

[112] Siegelmann, H. T., (1993), Neural and Super-Turing Computing, Minds and Machines 13, pp. 103?114.
On Siegelmann's claim that recurrent neural networks have super-Turing computational power.

[113] Siegelmann, H. T., (1995), Computation Beyond the Turing Limit, Science 268 (5210), pp. 545-548.
On Siegelmann's claim that recurrent neural networks have super-Turing computational power.

[114] Siegelmann, H.T & Sontag, E.F., (1984), Analog computation via neural networks, Theoretical Computer Science 131, pp. 331-360. *On Siegelmann's claim that recurrent neural networks have super-Turing computational power.*

[115] Smith P., (1998), Explaining Chaos. Cambridge University Press, Cambridge, UK.
Includes discussion of the 'shadowing theorems' in chaos.

[116] Smolensky, P., (1988), Connectionist Mental States: A Reply to Fodor and Pylyshyn , Southern Journal of Philosophy 26 (supplement), pp.137-161.
A refutation of Fodor and Pylyshyn's criticism of connectionism.

[117] Sprevak, M.D., (2005), The Chinese Carnival, Studies in the History & Philosophy of Science 36, pp. 203-209.
Reflections on Searle's Chinese room argument.

[118] Tanay, T., Bishop, J.M., Spencer, M.C., Roesch, E.B. & Nasuto, S.J., (2013), Stochastic Diffusion Search applied to Trees: a Swarm Intelligence heuristic performing Monte-Carlo Tree Search. In: Bishop, J.M & Erden Y.J, (2013), (eds), Proceedings of the AISB 2013: Computing and Philosophy symposium,'What is computation?', Exeter, UK.
Describes the use of a swarm intelligence paradigm - stochastic diffusion search - to play the strategic game of HeX.

[119] Tesauro, G. 1989. Neurogammon wins Computer Olympiad. Neural Computing 1, pp. 321-323.
Playing backgammon using neural networks.

[120] Tarassenko, L., (1998), A Guide to Neural Computing Applications, Arnold, London.
Practical introduction to neural computing.

[121] Tassinari, R.P. & D'Ottaviano, I.M.L., (2007), Cogito ergo sum non machina! About Gödel's first incompleteness theorem and Turing machines, CLE e-Prints, Vol. 7 (3).
Discussion of Penrose arguments against artificial intelligence.

[122] Thompson, E, (2010), Mind in Life, Belknap Press.
Thompson's claim that mind is causally related to life.

[123] Torrance, S., Thin Phenomenality and Machine Consciousness, in R.Chrisley, R.Clowes and S.Torrance, (eds), (2005), Proc. 2005 Symposium on Next Generation Approaches to Machine Consciousness: Imagination, Development, Intersubjectivity and Embodiment, AISB05 Convention. Hertfordshire: University of Hertfordshire, pp. 59-66.
On differences between brain simulations and real brains.

[124] Touzet, C., (1997), Neural Reinforcement Learning for Behaviour Synthesis. in: N. Sharkey, N., (ed) (1997), Special issue on Robot Learning: The New Wave, Journal of Robotics and Autonomous Systems 22 (3/4), pp. 251–281.
In reinforcement learning Touzet details replacing the Q-table with an MLP network or Kohonen network.

[125] Turing, A. M., (1937), [Delivered to the Society in November 1936], "On Computable Numbers, with an Application to the Entscheidungsproblem". Proceedings of the London Mathematical Society. 2 42. pp. 23-65, London, UK.
Introduction to the Turing machine and its application to the Entscheidungsproblem.

[126] Van Gelder, T., Port, R.F. (1995), It's About Time: an overview of the dynamical approach to cognition. In: Port, R.F. & Van Gelder, T. (eds.), (1995), Mind as Motion, pp. 1-45, The MIT Press, Cambridge MA.
Introduction to the dynamical system theory of mind.

[127] Van de Velde, F., (1997), On the use of computation in modelling behaviour, Network: Computation in Neural Systems 8, pp. 1-32.
Critique of neural modelling in classical connectionist models of low level cognition.

[128] Varela, F.J., Thompson, E. & Roesch, E., (1991), The Embodied Mind: Cognitive Science and Human Experience. The MIT Press.
Varela et al's seminal introduction to the so called 'embodied concept' of mind.

[129] Waskan, Jonathan A., (2005), Review of Preston, J. & Bishop, J.M., (eds), (2002), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Oxford University Press, Oxford UK. Philosophical Review, 114: 2, (2005).
Further reflections on the Chinese room argument.

[130] Werbos, P., (1974), Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences, PhD thesis, Harvard University.
'Usually' identified as the first derivation of the back propagation algorithm.

[131] Wheeler, M, (2002), Change in the Rules: Computers, Dynamical Systems, and Searle. In: Preston, J. & Bishop, J.M., (eds), (2002), Views into the Chinese Room: New Essays on Searle and Artificial Intelligence, Oxford University Press, Oxford UK.
In which Mike Wheeler uses the Chinese room argument to target all purely formal theories of mind.

[132] Whitehead A.N. & Russell, B., (1910), Principia Mathematica (3 volumes), Cambridge University Press, Cambridge UK.
An attempt to derive mathematics from 'first principles'.

[133] Wittgenstein, L., (1948), Last Writings on the Philosophy of Psychology (vol. 1). Blackwell, Oxford, UK.
The writings Wittgenstein composed during his stay in Dublin between October 1948 and March 1949, one of his most fruitful periods, which informed the Philosophical Investigations.

Biographical sketches

Dr. Mark Bishop is *Professor of Cognitive Computing* at Goldsmiths, University of London; Director of the Goldmsiths centre for Radical Cognitive Science and Chair of the UK society for Artificial Intelligence and the Simulation of Behaviour (AISB). He has published over 140 articles in the field of Cognitive Computing: *its theory*, where his

interests focus on the foundations of the swarm intelligence paradigm 'Stochastic Diffusion Search'; *its application*, where he has worked on industrial problems relating to autonomous robotics, neural networks and colour; and *its philosophical foundations*, where he developed the 'Dancing with Pixies (DwP)' thought experiment, a novel reductio ad absurdum argument against the possibility of computational machine consciousness; he has also written extensively on John Searle's 'Chinese room' argument.

Dept. Computing, Goldsmiths, University of London, New Cross, London