Cogn Process (2005)
DOI 10.1007/s10339-005-0017-7

RESEARCH REPORT

Gerard O'Brien · Jon Opie

How do connectionist networks compute?

Received: 13 July 2005 / Revised: 19 July 2005Accepted: 19 July 2005

Abstract Although connectionism is advocated by its proponents as an alternative to the classical computational theory of mind, doubts persist about its *computational* credentials. Our aim is to dispel these doubts by explaining how connectionist networks compute. We first develop a generic account of computation—no easy task, because computation, like almost every other foundational concept in cognitive science, has resisted canonical definition. We opt for a characterisation that does justice to the explanatory role of computation in cognitive science. Next we examine what might be regarded as the "conventional" account of connectionist computation. We show why this account is inadequate and hence fosters the suspicion that connectionist networks are not genuinely computational. Lastly, we turn to the principal task of the paper: the development of a more robust portrait of connectionist computation. The basis of this portrait is an explanation of the representational capacities of connection weights, supported by an analysis of the weight configurations of a series of simulated neural networks.

Keywords Computation · Connectionism · Representation · Resemblance

Introduction

Connectionism was first widely recognised as a potential rival to the classical computational theory of mind nearly 20 years ago.[1] Yet, despite all that has since been written about this approach to cognition, we still lack a satisfactory account of how connectionist networks compute. Into this vacuum has crept doubt about connectionism's *computational* credentials. This doubt takes three forms. One is the view that connectionism, far from being a rival computational paradigm, is nothing more than a modern version of associationism, and hence suffers from all the well-known vices of this much older position.[2] A second is the claim that while connectionists typically interpret the states and activity of connectionist networks in representational terms, closer scrutiny reveals that these putative representations fail to do any explanatory work, and since there is "no computation without representation" (Pylyshyn 1984, p. 62), the connectionist framework is better interpreted non-computationally.[3] And a third is the suggestion that connectionist networks are better characterised as dynamical systems rather than computational devices.[4]

If connectionism is ever to stand as a serious alternative to the classical computational theory of mind, this doubt must be dispelled. And the only way to do this is to explain how connectionist networks compute. That is the task we have set ourselves in this paper. We begin by developing a generic account of *computation*—no easy task, since like almost every other foundational concept in cognitive science, computation has resisted canonical definition. In the face of this problem, we opt for a

Communicated by John Sutton

G. O'Brien (✉) · J. Opie
Discipline of Philosophy, University of Adelaide,
5005 Adelaide, SA, Australia
E-mail: gerard.obrien@adelaide.edu.au
URL: http://arts.adelaide.edu.au/humanities/gobrien/
Tel.: +61-8-8303-5298
Fax: +61-8-8303-5241

[1]We use "connectionism" generically to denote the explanatory framework that models human perceptual and cognitive processes in terms of the operations of neuron-like processing units connected together to form neural-like networks. While this explanatory framework has antecedents running back more than 50 years, we take the appearance of Rumelhart and McClelland (1986) and McClelland and Rumelhart (1986) as the moment when connectionism, in the guise of parallel distributed processing, came of age.
[2]This claim is most famously associated with Fodor (e.g. Fodor and Pylyshyn 1988; Fodor 2000, Ch. 3) but it pops up in a number of different places (Pinker 1997, pp. 112–131 and 2002, pp.78–83).
[3]See, e.g. Ramsey (1997).
[4]See, e.g. the various contributions to Port and Van Gelder (1995).

characterisation that captures the intended role of computation in cognitive science. Next we examine what might be regarded as the "conventional" account of connectionist computation. We show why this account is inadequate and hence fosters the kinds of doubt we have just enumerated. We then turn to the principal task of the paper: the development of a more robust portrait of connectionist computation. The basis of this portrait is an explanation of the representational capacities of connection weights, supported by an analysis of the weight configurations of a series of simulated neural networks. Once this explanation is in place, it will be apparent how connectionist networks compute.

Toward a proper understanding of computation in cognitive science

Computation is a concept so overused and so variously defined that we sometimes despair of it ever being meaningfully deployed. And yet we also believe that computation is the most important concept in all of cognitive science. Indeed, we would argue that without the concept of computation there is no cognitive (as distinct from behavioural, psychological, biological, or just plain physical) science in the first place. So something must be done.

In all that has been written about computation in cognitive science, two extreme characterisations are discernible. At one extreme is a depiction of computation in terms of the symbol manipulations of a digital computer; at the other is the claim that computation is simply a matter of implementing a function. We'll briefly say what's wrong with these proposals, before developing a middle ground characterisation that does justice to the explanatory role of computation in cognitive science.

Computation as symbol manipulation

Jerry Fodor is fond of remarking that there is only one important idea about how the mind works that anybody has ever had. This idea he attributes to Alan Turing:

> [G]iven the methodological commitment to materialism, the question arises, how a machine could be rational?...Forty years or so ago, the great logician Alan Turing proposed an answer to this question...Turing noticed that it isn't strictly true that states of mind are the only semantically evaluable material things. The other kind of material thing that is semantically evaluable is *symbols*... Having noticed this parallelism between thoughts and symbols, Turing went on to have the following perfectly stunning idea. "I'll bet", Turing (more or less) said, "that one could build a *symbol manipulating machine* whose changes of state are driven by the material properties of the symbols on which they operate (for example, by their weight, or their shape, or their electrical conductivity). And I'll bet one could so arrange things that these state changes are rational in the sense that, given a true symbol to play with, the machine will reliably covert it into other symbols that are also true". (Fodor 1992, p. 6)

The rest, as one says, is history. Cognitive science emerged as a discipline (or at least, a "multi-discipline") in the 1950s. What was novel about cognitive science (as opposed to those already established disciplines that study the mind, including neuroscience, psychology and philosophy) was its commitment to the *computational theory of mind*: the idea that cognitive processes are the symbol manipulations of a neurally realised digital computer.

At its inception, cognitive science thus embraced a Turing-inspired understanding of computation. Computation is what happens in a digital computer: a causal/mechanical process in which language-like representing vehicles are recognised and transformed in a semantically coherent fashion purely on the basis of their syntactic properties.

The problem with this characterisation is that it pays scant attention to the history of computer science. For more than 2000 years, theorists and practitioners have recognised a distinction between two forms of computation: *digital computation*, admirably formalised by Turing and others, and *analog computation*. The latter currently lacks a precise formal definition, but a quick survey of computer science textbooks of the 1950s and 60s reveals an intuitively clear demarcation: while digital computers employ semantically inert symbols (tokens that bear no resemblance to what they represent), analog computers employ internal models that physically or structurally resemble their representanda.[5] Analog computation is thus not properly conceived as symbol manipulation, but as a physical process driven by the structural properties of analog representational media.

From this perspective, Turing's great achievement was not that of conceiving the *idea* of computation, but of developing one very powerful means of mechanising computational processes. Drawing this distinction is important because once it is clear that the idea of computation is distinct from Turing's account of how computational processes might be mechanised, it is possible to investigate the former independent of the latter. What we therefore require is a characterisation of computation that captures more (if not all) of those processes that have earned this epithet over the last 2000 years.

Computation as implementing a function

In response to this demand, a different kind of characterisation of computation is now popular in cognitive

[5]See Truitt and Rogers (1960) for both a semi-formal account of analog computation along these lines, and for numerous examples of analog computers.

science. For example, in an influential article Churchland et al. (1990) have this to say on the subject:

> In a most general sense, we can consider a physical system as a computational system just in case there is an appropriate (revealing) mapping between some algorithm and associated physical variables. More exactly, a physical system computes a function $f(x)$ when there is (1) a mapping between the system's physical inputs and x, (2) a mapping between the system's physical outputs and y, such that (3) $f(x) = y$. (1990)

In this passage, however, it is not obvious that the reference to an "appropriate (revealing)" mapping is doing any real work. Once this is removed, what remains is the proposal that a computation is performed by some physical system just in case its causal operation can be interpreted as implementing some function. Chalmers (1994) summarises the idea as follows:

> A physical system implements a given computation when there exists a grouping of physical states of the system into state-types and a one-to-one mapping from formal states of the computation to physical state-types, such that formal states related by an abstract state-transition relation are mapped onto physical state-types related by a corresponding causal state-transition relation. (1994, p. 392)

The bottom line here, according to Chalmers, is that computation is simply " *an abstract specification of causal organisation*" (1994, p. 396, emphasis in original).

This characterisation does satisfy the desideratum we mooted above, given that it captures both analog and digital computation in its net. But it does so at a very great cost. Since all law-governed physical systems (and, granting determinism, this equates with all physical systems) are interpretable as implementing some function or other, we arrive at the unwelcomed conclusion that *all* physical systems are computational. And that would appear to render the concept of computation in cognitive science explanatorily vacuous.

Chalmers, for one, resists this conclusion:

> This objection expresses the feeling that if every process, including such things as digestion and oxidation, implements some computation, then there seems to be nothing special about cognition any more, as computation is so pervasive. This objection rests on a misunderstanding. It is true that any given *instance* of digestion will implement some computation, as any physical system does, but the system's implementing this computation is in general irrelevant to its being an instance of digestion.... With cognition, by contrast, the claim is that it is *in virtue* of implementing some computation that a system is cognitive. That is, there is a certain class of computations such that *any* system implementing that computation is cognitive (1994, p. 397).

Chalmers' reasoning fails to reassure, however. The concept of computation was originally introduced as a way of distinguishing two classes of causal processes: those characteristic of the vast majority of physical systems (e.g. intestines, microwave ovens, cups of tea, etc.), and those that are the preserve of intelligent systems alone. Computational processes are supposed to be *special* in some way—in a way, moreover, that provides us with some explanatory purchase with respect to the problem of intelligent behaviour. Since implementing a function is a ubiquitous feature of nature, choosing to characterise computation in this way repudiates the very motivation for introducing the concept into cognitive science in the first place.

Computation as content-shaped causal processing

What we need is a way of characterising computation that limns a middle path between the restrictiveness of digital computation and the promiscuity of abstract causal organisation. One way to do this is to re-visit the account of computation we get from digital computers, and consider whether this can be liberalised to some degree without falling prey to the problem of explanatory vacuity.

Digital computation, remember, is symbol manipulation: a causal/mechanical process in which language-like representing vehicles are recognised and transformed in a semantically coherent fashion purely on the basis of their syntactic properties. But as already remarked, the practice of computation has not historically been restricted to processes defined over symbols. Consider, for example, the familiar tactic of representing a physical variable, such as the velocity of a particle, using a curve on the plane. If we plot velocity on one axis, and time on the other, it is possible to compute distance travelled by measuring the area under the curve, or acceleration by constructing tangents to the curve. These are examples of analog computations which employ a non-symbolic representing vehicle.

Viewed from this less-restrictive perspective, there are two distinctive features of computational processes (as opposed to causal processes in general). First, they are associated with representing vehicles of some kind. Second, and more importantly, computational processes are shaped by the contents of the very representations they implicate. We thus arrive at the following characterisation:

> *Computations are causal processes that implicate one or more representing vehicles, such that their trajectory is shaped by the representational contents of those vehicles.*

Talk of representational content "shaping" the causal trajectory of computation is vague, of course. But this is deliberate. *Prima facie*, there are different ways of organising physical systems such that representational

content can play this role. In the case of digital systems, while computational operations only ever have access to the syntactic properties of symbols, the rules that govern these syntactic manipulations are nonetheless carefully crafted so as to ensure that they respect the contents of the symbols. In Dennett's memorable terms: digital computers are syntactic engines that behave as if they were semantic engines (Dennett 1987, p. 61). Analog computers, by contrast, are systems whose behaviour is driven not by content-sensitive rules, but by semantically "active" analog representations that physically or structurally resemble what they represent.[6]

Although this strategy of characterising computation in terms of operations shaped by representational contents is quite common in the literature[7] it does not find favour everywhere. Chalmers, for instance, has this to say:

> The original account of Turing machines by Turing (1936) certainly had no semantic constraints built in. A Turing machine is defined purely in terms of the mechanisms involved, that is, in terms of syntactic patterns and the way they are transformed.... To implement a Turing machine, we need only ensure that this formal structure is reflected in the causal structure of the implementation.... [W]hen computer designers ensure that their machines implement the programs that they are supposed to, they do this by ensuring that the mechanisms have the right causal organisation; they are not concerned with semantic content. In the words of Haugeland (1985), if you take care of the syntax, the semantics will take care of itself (1994, p. 399).

In our view, this represents a profound misreading of both Turing and Haugeland. Far from eschewing semantic considerations, computer science is in the business of designing and implementing formal operations that satisfy semantic constraints. In the passage of his classic text just prior to articulating his famous "formalists' motto" (quoted approvingly by Chalmers), Haugeland takes himself to be addressing the following question:

> Interpretation and semantics transcend the strictly formal—because formal systems as such must be self-contained. Hence to regard formal tokens as symbols is to see them in a new light: semantic properties are not and cannot be syntactical properties. To put it dramatically, interpreted formal tokens lead two lives: SYNTACTICAL LIVES, in which they are meaningless markers, moved according to the rules of some self-contained game; and SEMANTIC LIVES, in which they have meanings and symbolic relations to the outside world. The corresponding dramatic question then is this: how do the two lives get together? (1985, p. 100).

The answer that Haugeland goes on to develop is the fundamental basis of digital computation:

> The idea...is to design these formal systems so that they can be interpreted as axiomatic systems in the intuitive sense. That requires two things of the system (as interpreted);
>
> 1. the axioms should be true...; and
> 2. the rules should be truth preserving (1985, pp. 103–105).

In this light, the whole point of Haugeland's formalists' motto is to reinforce the message that it is only when the syntactically specified rules of the system are so crafted that they satisfy these semantic constraints, that "the semantics will take care of itself".

This isn't just an exercise in academic exegesis. Understanding the role of representational content in shaping computational processes is pivotal to understanding why the concept of computation arose in the first place. Intelligence is a rare commodity, and one that provokes a profound question: how is that some physical systems are capable of intelligent behaviour when the majority of systems in the universe are not? The concept of computation is supposed to provide some leverage here—intelligent systems are special because they alone engage in computation. But this answer won't suffice unless computational processes are themselves special. The characterisation developed above explains why they are (computational processes are shaped by the representational contents of the vehicles they implicate) and hence explains why the concept of computation is foundational for cognitive science.

With this characterisation of computation in place we can now turn to the principal task of the paper: that of explaining how connectionist systems compute. To satisfy this task we will need to show how representational content plays a role in shaping the trajectory of connectionist computational processes.

Connectionist computation: what's wrong with the conventional story?

It is possible to identify something of a consensus among proponents of connectionism as to the nature of computation in connectionist networks. The argumentative burden of this section is to establish that this "conventional" account of connectionist computation is unsatisfactory, and to explain why it has nurtured doubts about connectionism's computational credentials.

The conventional story

The characterisation of computation we developed in the preceding section emphasises the importance of

[6] See O'Brien (1999) for further discussion.

[7] See, e.g. Cummins and Schwarz (1991), p.64; Dietrich (1989); Fodor (1975), p. 27; and Von Eckardt (1993), pp. 97–116.

representation for computation. It is not surprising, therefore, that the conventional account of connectionist computation focuses on showing how activity across connectionist networks admits of a representational interpretation.

The story goes like this. A connectionist network is a collection of interconnected processing units (modelled on neurons), each of which has an *activation level* (modelled on a neuron's spiking frequency) that is communicated to other units in the network via modifiable, weighted connections (modelled on synapses). From moment to moment, each unit sums the weighted activation it receives, and generates a new activation level that is some threshold function of its current activity and that input. Via this process, a network transforms patterns of activity across its input layer into patterns of activity across its output layer. Altering the network's connection weights alters the activation patterns the network produces in response to its inputs. Consequently, a network can be taught to generate a range of target patterns in response to a range of inputs. These patterns of activity, because they are produced by a training regime that gradually shapes the network's responses so that it is successful in negotiating some task domain, are thought to constitute a form of information coding, often termed *activation pattern representation*. According to this account, therefore, connectionist networks compute by transforming activation pattern representations across their input units into activation pattern representations across their output units.[8]

But this account is superficial. What we really want to know is *how* connectionist networks are able to transform their input representations into appropriate output representations. It is at this point that the conventional story gets both more complicated and more interesting. The proffered explanation focuses on the fact that the hidden unit landscape of a trained network is partitioned into linearly separable regions, regions that capture the categorial distinctions necessary for generating a solution to the computational problem(s) posed by the inputs.

To illustrate this idea, consider a three layer, feedforward network designed by Laakso and Cottrell (2000) to perform colour categorisation (see Fig. 1). The task of this network is to take reflectance spectra—which provide a measure of the relative amounts of light reflected by an object across a range of wavelengths—and produce a colour judgment corresponding to that of a normal human observer. The inputs to the network are 523 reflectance spectra selected from a database produced at the University of Kuopio (anonymous 1995; Parkkinen 1989).[9] Each spectrum is measured over the 400–700 nanometre range in 5 nm "bins", and is thus a 61-dimensional vector of which the first component is the reflectance intensity at a wavelength of 400 nm; the second, the reflectance at 405 nm, and so on, through to the 61st component which is the reflectance at a wavelength of 700 nm. The input layer thus has 61 input units onto which are locked the amplitude values of the spectra. There are three units in the hidden layer, and five binary units in the output layer for encoding the relevant colour categories (red, green, blue, yellow, and purple). After training via backpropagation of errors, the network achieved better than 90% accuracy in its assignment of input spectra to colour categories. (See Laakso and Cottrell 2000, pp. 58–67 for further details.)

We reproduced these results by training a series of networks on the same data set. The activity at the hidden layer of a trained network can be portrayed as a three-dimensional activation space, in which the activity of each hidden unit is represented along one coordinate axis. For each input to the network, one gets a different pattern of activation on the hidden layer, and a corresponding point in activation space. We found that each colour-categorisation network partitions its activation space into linearly separable regions (in three-dimensions, these are regions that can be cleanly divided by a plane), such that the activation points corresponding to the various colour categories are located in distinct parts of the space (Fig. 2). This is typical of feedforward neural networks, and it is widely agreed that it is by virtue of organising their activation spaces in this way that such networks are able to correctly categorise their inputs.

This much about hidden unit activation pattern representation is common lore among connectionists. What is not always appreciated about hidden unit activation patterns, however, is that collectively they *structurally resemble* aspects of the task domain over which the network has been trained. Indeed, it is the existence of this structural resemblance relation that anchors the representational interpretation of activation patterns in the first place (O'Brien and Opie 2001; 2004). Since this structural resemblance theory of representational content will be important to the argument developed in the next section, we will pause here to examine it in some detail.

Resemblance is a fairly unconstrained relationship, because objects or systems of objects can resemble each other in a huge variety of ways, and to various different degrees. The most straightforward kind of resemblance involves the sharing of one or more physical properties. Thus, two objects might have the same colour, or mass, the same density, or electric charge, or be equal along a number of physical dimensions simultaneously. We shall refer to this kind of relationship as *first-order resemblance*.[10] A representing vehicle and its represented

[8]See, e.g. Bechtel and Abrahamsen (2002); Clark (1989), 1993; and Tienson (1987).

[9]These spectra were generated by measuring the reflectance profile of colour cards in the Munsell Book of Color (anonymous 1976), a set of cards that is used in standard psychometric tests of colour perception.

[10]We are here adapting some terminology developed by Shepard and Chipman (1970). They distinguish between first- and second-order *isomorphism*. Isomorphism is a very restrictive way of characterising resemblance, and hence we prefer to avoid this terminology (see O'Brien and Opie 2004).

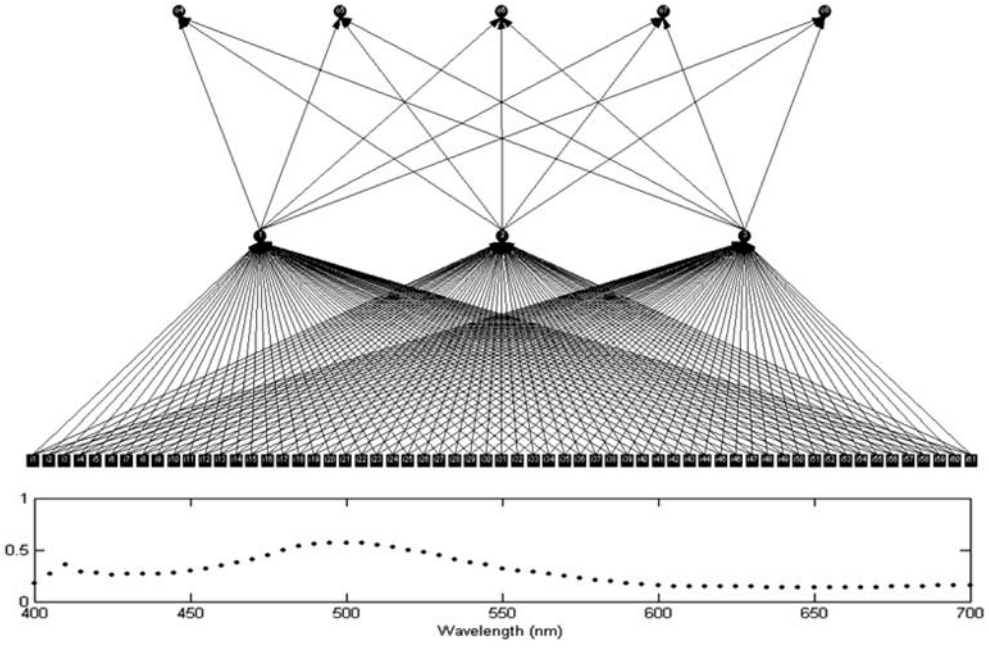

Fig. 1 The structure of the colour-categorisation network, showing an example of an input spectrum to be encoded on the input layer

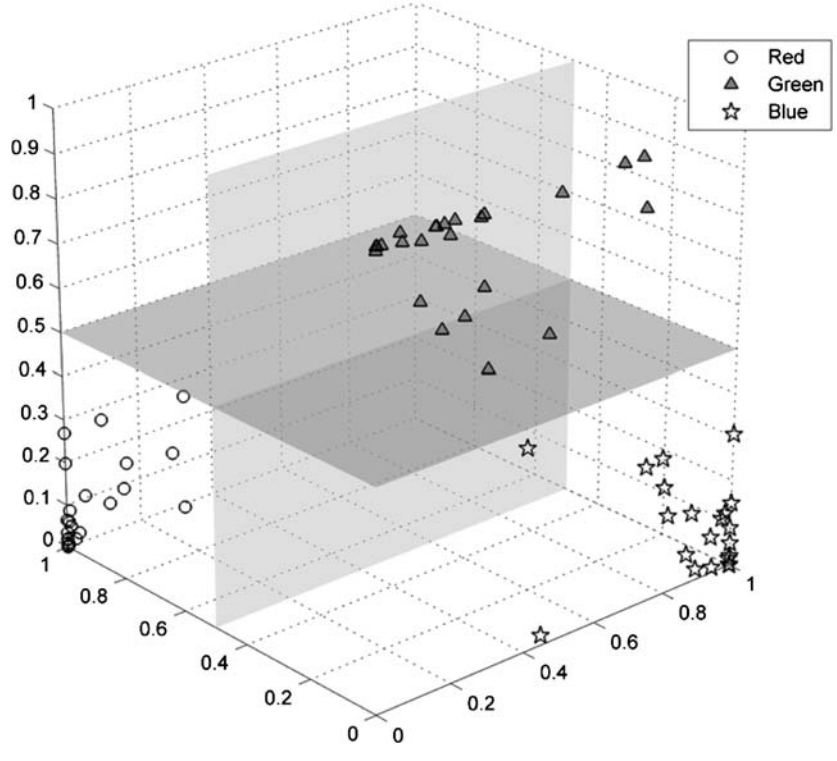

Fig. 2 Hidden unit activation space for one of the colour-categorisation networks

object resemble each other in this way if they have physical properties in common.

First-order resemblance is clearly unsuitable as a general ground of neural representation, since it is incompatible with what we know about the brain. It is quite obvious that our brains are capable of representing features of the world that are not replicable in neural tissue. There is, however, another kind of resemblance

available, which we shall refer to as *second-order resemblance*.[11] In second-order resemblance, the requirement that representing vehicles share physical properties with their represented objects can be relaxed in favour of one in which the *relations* among a system of representing vehicles mirror the *relations* among their objects. For example, a mercury thermometer can be used to represent temperature in virtue of the linear relationship between the length of a column of mercury and ambient temperature—variations in the one correspond systematically with variations in the other.

Although first-order resemblance cannot be the general ground of neural representation, the same is not true of second-order resemblance. Two systems can share a pattern of relations *without* sharing the physical properties upon which those relations depend. Second-order resemblance is actually a very abstract relationship. Essentially, nothing about the physical form of the relations defined over a system of representing vehicles is implied by the fact that it resembles a set of represented objects at second-order; second-order resemblance is a formal relationship, not a substantial or physical one.

As already foreshadowed, the form of second-order resemblance that is relevant in the present context is *structural resemblance*. One system *structurally resembles* another when the *physical* relations among the objects that comprise the first preserve some aspects of the relational organisation of the objects that comprise the second. Structural resemblance would seem to be the right second-order resemblance relation for explaining the representational content of connectionist representing vehicles. Hidden unit activation space is a *mathematical* space used by theorists to portray the set of activation patterns a network is capable of producing over its hidden layer. Activation patterns themselves are physical objects (patterns of neural firing, if realised in a brain), and thus distance relations in activation space actually codify *physical* relations among activation states. What is crucial here is that the set of hidden unit activation patterns generated across any trained-up connectionist network constitutes a system of representing vehicles whose physical relations sustain a second-order resemblance relation with respect to the task domain over which the network has been trained.

Consider, for example, the relationship between the set of hidden layer activation patterns generated by the colour-categorisation network and its task domain. Physical similarities and differences among these patterns of activity (which appear as relative distances in the activation space) correspond to similarities and differences among the reflectance spectra that the network is responding to (see Sect. 4 for a more detailed discussion).

The structural resemblance relation between hidden unit activation patterns and aspects of a connectionist network's task domain licenses an interpretation of the former as representing vehicles. This in turn appears to support the claim, made by the proponents of connectionism, that these networks are in the computing business. Why then have doubts about connectionism's computational credentials continued to linger in the cognitive science literature? It is to this issue that we will now turn.

What's wrong with the conventional story?

The conventional story about connectionist computation is elegant, but incomplete. Recall that a computational interpretation of connectionism must not only show that connectionist networks implement representing vehicles; it must also show how processing in networks is shaped by the representational contents of those vehicles. It is this latter requirement that the conventional story fails to satisfy.

To see this, consider the colour-categorisation network we described above. This network is required to sort spectra into colour categories, a task at which it succeeds because the network's hidden unit activation space is partitioned into regions corresponding to those categories. And it is a relatively simple exercise to map from regions in activation space to binary representations of colour on the network's output layer. Notice, however, that given any spectrum as input, it is the configuration of weights between the input and hidden layers that determines the resulting hidden layer activity. Furthermore, since they govern each and every such mapping, it is these weights that are responsible for the global structure of the hidden unit activity space. The representing vehicles on which the conventional story focuses—activation patterns across the hidden layer—are not causally implicated in these transformations. They are the *products*, not the *source*, of processing. And as such, their representational contents play no role in shaping the trajectory this processing takes.[12]

It is precisely this kind of analysis which invites the charge that connectionism is nothing more than a latter day version of associationism. This interpretation is quite consistent with a representational understanding of the activity across the layers of connectionist networks. It's just that it restricts connectionist networks to the mere association of "ideas", rather than the content-

[11] Bunge (1969), in a useful early discussion of resemblance, draws a distinction between *substantial* and *formal* analogy which is close to our distinction between first- and second-order resemblance. Two theorists who have kept the torch of second-order resemblance burning over the years are Palmer (1978) and Shepard (Shepard and Chipman 1970; and Shepard and Metzler 1971). More recently, Blachowicz (1997); Cummins (1996); Gardenfors (1996); Johnson-Laird (1983); O'Brien (1999) and Swoyer (1991), have all sought to apply, though in different ways, the concept of second-order resemblance to representation.

[12] This is not to deny that the physical relations among activation patterns on the hidden layer have a bearing on downstream processes, both at the output layer and in other networks. Our point is simply that this (diachronic) relational structure is governed by some *other* (synchronic) feature of the network, namely, the configuration of its connection weights.

driven forms of information processing that are necessary to explain intelligent behaviour.

There is a fairly standard riposte to this charge in connectionist circles. Connectionist networks implement two quite different kinds of representation: in addition to the information coded in activation patterns, which is transient and hence obliterated whenever the network is exposed to new input, information is coded in a long-term fashion in the network's connection weights. These weights, it is often claimed, constitute the network's memory. Since it is connection weights that govern the transformations of activity from layer to layer in a network, it thus appears that we do have a representational story to tell about the structures that shape the trajectory of connectionist processing.

The trouble with this response, however, is that we currently lack a representational analysis of connection weights comparable to the kind of analysis that is available for activation patterns. Consequently, the claim that connection weights represent a network's long-term knowledge is left unanchored, and commentators are justified in expressing doubts about this claim. Ramsey, for example, highlights what he takes to be a fundamental difference between connection weights and the rules that govern the symbol manipulations of digital computers:

> As the relevant content for this type of representation is the system's long-term knowledge...the most obvious point of comparison should be with the explicit rules that sometimes govern classical computation systems and are thought to encode those systems' knowledge base. Is there an explanatory pay-off in viewing connection weights as representations that is similar to the return we get when this is done with rules in classical models? I believe the answer is 'no' for the following reason. [In] classical models it is typically the case that causally distinct structures encode commands for specific stages of the computation... However, in trained connectionist models, this type of specificity is not possible. While it might be true that some connection weights contribute to some episodes of processing more than others, there is no level of analysis at which we can say a particular weight encodes a particular command or governs a specific algorithmic step in the computation. Instead, all the system's know-how is superimposed on all the weights with no particular mappings between the two. (1997, pp. 48–49)

Further rumination on this issue eventually leads Ramsey to conclude that "there doesn't appear to be any other level of understanding or explanatory motivation that requires us to view the weights as representations" (1997, p. 51), and he recommends that we view connectionist explanations of cognition as *dynamical* rather than computational (1997, p. 61).

The dialectical position, we think, is this. However strong our reasons for interpreting activation patterns in connectionist networks as representing vehicles, doubts will persist about connectionism's computational credentials unless Ramsey's challenge can be answered. What is required is a "level of understanding or explanatory motivation that requires us to view the weights as representations". It is time to meet this challenge.

Connection weight representation

We have seen that activation pattern representation is supported by a relation of structural resemblance between the patterns of activity in a connectionist network and the task domain in which that network operates. The proposal we explore here is that there is a more fundamental structural resemblance between the connection weights of such a network and its task domain; one that supports a species of representation we will call *connection weight representation*.[13]

Although, the relation of structural resemblance between a trained-up network's patterns of activity and its task domain is relatively easy to identify, the same cannot be said of any such relation between connection weights and task domain. If such a relation exists, it will require some teasing out. We will approach this problem by more closely examining the role of connection weights in connectionist processing.

Processing with connection weights

It is well-known that networks operating in the same domain, but trained-up with different initial assignments of connection weights, come to occupy different points in "weight space".[14] There is no simple relationship between the position in weight space occupied by a trained network and the task domain. We demonstrated this by training a group of 20, three-layer feedforward networks to perform at close to 100% accuracy on Laakso and Cottrell's colour-categorisation task. We then measured the pair-wise correlations among the (hidden layer) weight matrices of these networks (for a total of 190 comparisons). The set of correlations turned out to be randomly distributed about a mean of zero, confirming that there is no simple, first-order relationship

[13] In what follows, we develop this proposal by focusing solely on the connection weights between the input and hidden layers of feedforward networks. (We will reinforce this point by occasionally referring to the "hidden layer" connection weights: these are the weights that determine the activity across the hidden layer.) It is our view, however, that this proposal applies to connectionist systems more generally.

[14] The weight space of a network is a Euclidean vector space in which each of the network's connection strengths is represented as the position along a distinct coordinate axis. The dimensionality of this space corresponds to the number of connections in the network. Once can picture training a network as a journey through weight space, and different final positions in the space as alternative ways of dealing with the task demands.

between these networks (see Fig. 3). Since the networks themselves are not related in any straightforward way, it appears unlikely that each bears some simple relationship to the task domain over which they operate.

It remains a live possibility, however, that connectionist networks embody some (higher-order) internal structure that warrants a representational understanding of their connection weights. To explore this possibility we need to take a closer look at how connectionist networks process their inputs.

The key players in network processing are what we call *fan-ins*. A fan-in is the vector of weights modulating the effect of incoming activity on a particular hidden unit. Within any feedforward network there is one fan-in per hidden unit, each corresponding to a row of the network's hidden layer weight matrix (see Fig. 4). Fan-ins effect the transformation of the network's input space into its hidden unit activation space. More specifically, each fan-in determines how one hidden unit responds to input, by way of a product of input activation and fan-in values. This product is then modified by the hidden unit's activation function to produce the value along a single coordinate in activation space. It is thus a network's fan-ins that interface directly with the structure of the vectors coded at the input layer, and which ultimately determine the structure of activation space. Accordingly, if we are to discover any structural resemblance between a network's connection weights and its task domain it is the fan-ins on which we should focus.

Connection weights as representing vehicles

Given the crucial role of fan-ins in network processing, we offer the following proposal: the fan-ins in the hidden layer of a successful connectionist network structurally resemble aspects of the network's task domain.

To investigate this conjecture we trained a series of three-layer feedforward networks to solve the colour-categorisation problem using a subset of Laakso and Cottrell's original data: about 25 each of the spectra normally classified as red, green, and blue, respectively. Each network had 61 input units and three hidden units. We represented the fan-ins of the trained networks using weight diagrams and compared these with the means of the red, green and blue input data sets.

A typical example is shown in Fig. 5. The three fan-ins are depicted on the right, the mean spectra on the left. One immediately notices a striking similarity between the fan-ins of this network and the means of the data sets. The shape of the fan-in for hidden unit 2, for example, corresponds nicely to the shape of the mean spectrum of the 25 inputs that normal observers classify as red. Likewise, the fan-in for hidden unit 3 resembles the mean of the "green" spectra, and the fan-in for hidden unit 1 resembles the mean of the "blue" spectra. What this indicates is that, for each fan-in, the relative magnitudes of its component weights mirror the relative amplitudes of the various wavelengths comprising one of the mean spectra. Since this mirroring is a similarity at the level of relations, rather than properties, it is an instance of second-order resemblance. And since it is grounded in the physical relations among the fan-in weights (i.e. their relative magnitudes), it is a structural resemblance.

In the previous section we saw that it is a relation of structural resemblance that anchors a representational interpretation of hidden unit activation patterns. We've just seen (Fig. 5) that there is a structural resemblance between the fan-ins of the colour-categorisation network

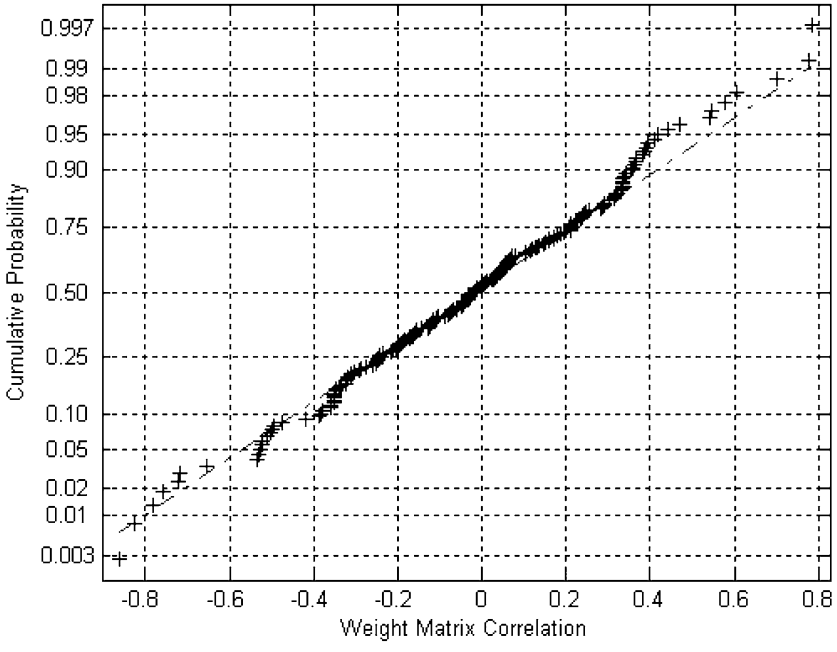

Fig. 3 A plot of cumulative probability against weight-matrix correlation. A good fit to the straight line indicates a normal distribution

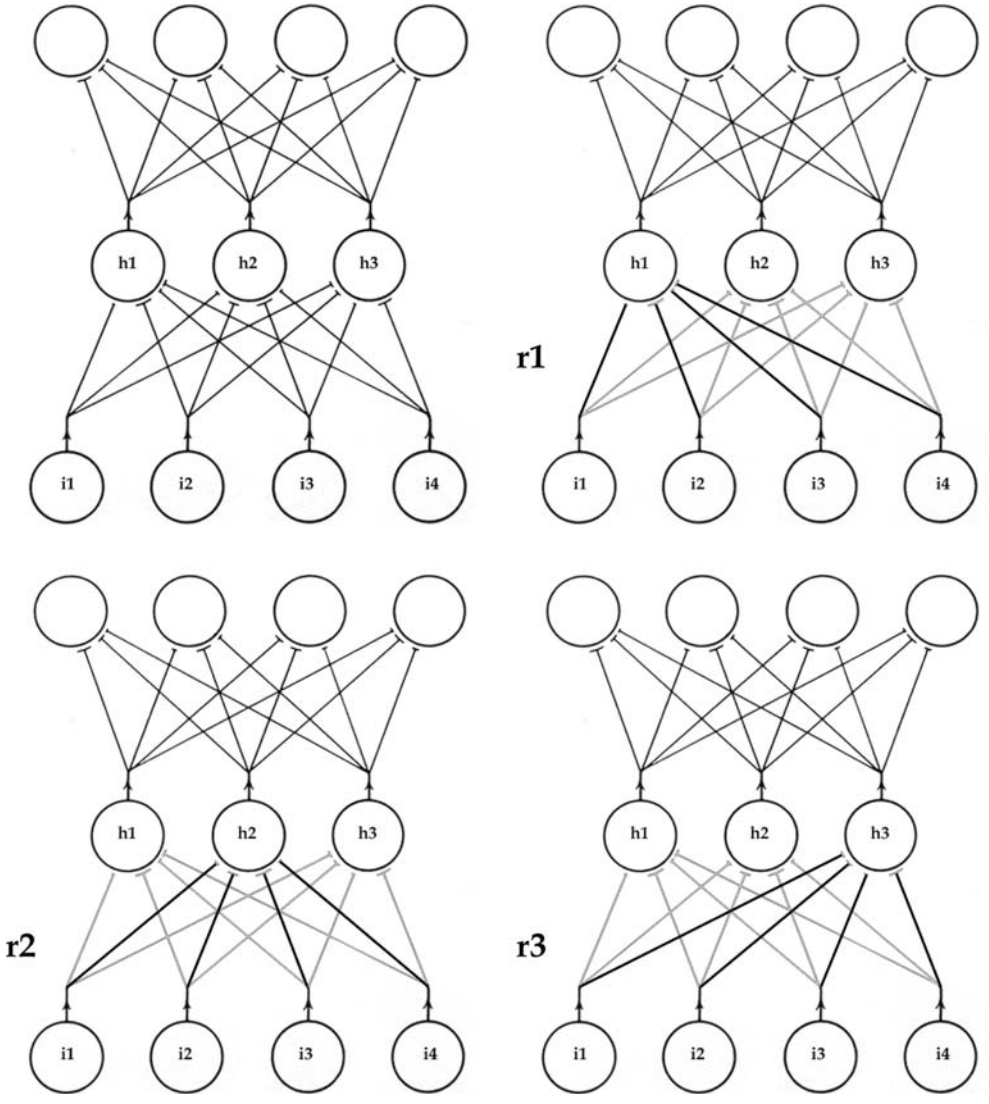

Fig. 4 A simple network with and without its three fan-ins (r1, r2, & r3) highlighted

and the task domain over which it operates. That resemblance licenses an interpretation of fan-ins (and their component weights) as representing vehicles.

Connectionist computation: the real story

The characterisation of computation we offered above suggests that connectionist systems must satisfy two conditions if they are to count as computational devices: (i) they must implicate representing vehicles of some kind, and (ii) the contents of those vehicles must shape the causal processes that occur in connectionist processing. We established that connection weights may legitimately be interpreted as representing vehicles, at least for a significant class of connectionist systems. It remains to show that the contents of this species of vehicle influence the trajectory of connectionist processing.

We have noted the crucial role of fan-ins in transforming a network's inputs into hidden layer activation. It is the "shape" of these vectors that govern the respective activities of the hidden units they influence, by way of the so-called "dot product" of weights and input activation. Taking a dot product is a well-known way of measuring the similarity of two vectors.[15] Each fan-in is, in effect, a filter looking for input with a particular shape. The dot product indicates the extent to which a given input matches a particular fan-in filter, as does the activity of the corresponding hidden unit. Input that is presented to the colour-categorisation network, for example, is filtered through three fan-in vectors, thereby modifying the activation of the three units in the hidden layer. Activity in the hidden layer thus reflects the degree of similarity between the input spectra and the fan-ins. Correlatively, hidden unit activation space forms a three-dimensional map that allows us to compare the filtered versions of the input spectra, one with the other.

[15]The dot product of two vectors in a Euclidean space is at a maximum when the angle between them is zero, and decreases as the angular separation between them increases.

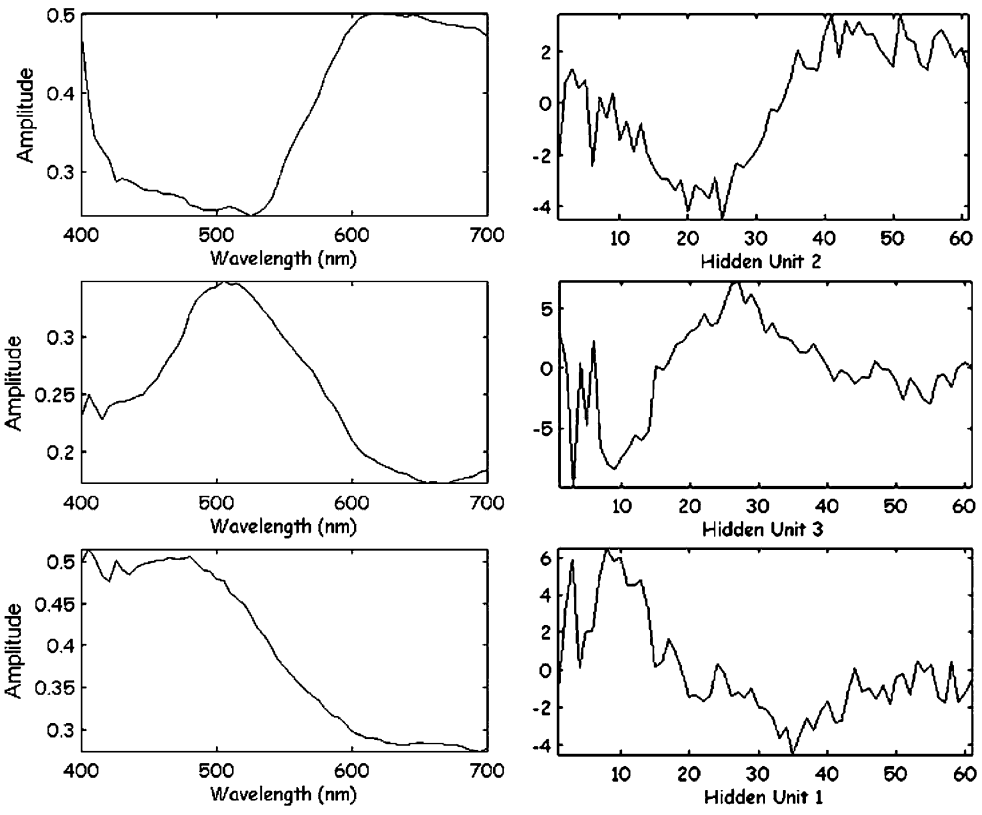

Fig. 5 On the *left* are the mean spectra of the three classes of inputs; those classified (from *top to bottom*) as red, green and blue. On the *right* are the three fan-ins of the network, with weight value on the *y*-axis and input index on the *x*-axis

Now the final piece of the puzzle is in place. We have shown that the fan-ins in the colour-categorisation network structurally resemble aspects of the task domain, namely, the mean spectra of the three classes of input (red, green and blue). That resemblance warrants us in regarding those fan-ins, and their component weights, as representing vehicles. But, we have also shown that it is this same resemblance, embodied in the physical structure of the fan-ins, that drives the causal processes within the network. It is by virtue of their resemblance to global features of the input data that the fan-in vectors contrive to transform reflectance spectra into a map of categorial colour, and thereby solve the problem posed to the network. Representational content is in the driver's seat here, as we require, and it appears that connectionist networks are genuine computational devices.

This is a very satisfying result for proponents of connectionism. It enables us to meet Ramsey's challenge, because we now have a robust explanatory motivation for viewing connection weights as representations. And this in turn puts to bed the lingering doubts about connectionism's computational credentials.

Connectionist networks are capable of successfully negotiating their task domains because they structurally resemble them—a resemblance relation they gradually acquire in the course of training. This structural resemblance relation is sustained at two different levels of description. It is sustained (diachronically), by the set of activation patterns that are produced across a network's hidden units in response to its various inputs, and, more importantly, it is sustained (synchronically) by the higher order structure of the network's hidden layer connection weights.

These two kinds of structural resemblance support an interpretation of activation patterns and connection weights as different species of representing vehicle. And these two kinds of representing vehicle shape the trajectory of connectionist processing in different ways. Activation pattern representations shape the impact that one network has on other networks or motor mechanisms to which it is connected. Connection weight representations, by contrast, are responsible for the production of these activation pattern representations in the first place.

This last point is important because it secures a computational understanding of connectionist processing, at least according to the characterisation we have developed in this paper. The causal operations that generate a hidden unit activation pattern implicate one or more representing vehicles (the fan-in connection weights) and the trajectory of this process is shaped by the representational content of these vehicles (since it is the structural resemblance relation that determines the representational content of the fan-ins). Connectionist networks are not merely association engines or dynamical systems; they are full-blooded computational mechanisms. And they compute by exploiting relations of structural resemblance between their connection weights and their target domains.

References

Anonymous (1976) Munsell book of color: matte finish collection. Munsell Color Company, Inc

Anonymous (1995) Kuopio color database. http://www.lut.fi/ltkk/tite/research/color/lutcs_database.html

Bechtel W, Abrahamsen A (2002) Connectionism and the mind: parallel processing, dynamics, and evolution in networks. Blackwell, Oxford

Blachowicz J (1997) Analog representation beyond mental imagery. J Philosophy 94:55–84

Bunge M (1969) Analogy, simulation, representation. Revue-Internationale-de-Philosophie 23:16–33

Chalmers DJ (1994) On implementing a computation. Mind Mach 4:391–402

Churchland PS, Koch C, Sejnowski T (1990) What is computational neuroscience? In: Schwartz E (eds) Computational neuroscience. MIT Press, Cambridge

Clark A (1989) Microcognition: philosophy, cognitive science, and parallel distributed processing. MIT Press, Cambridge

Clark A (1993) Associative engines: connectionism, concepts, and representational change. MIT Press, Cambridge

Cummins R (1996) Representations, targets, and attitudes. MIT Press, Cambridge

Cummins R, Schwarz G (1991). Connectionism, computation and cognition. In: Horgan T, Tienson J (eds). Connectionism and the philosophy of mind. Kluwer, Dordrecht

Dennett D (1987) The intentional stance. MIT Press, Cambridge

Dietrich (1989) Semantics and the computational paradigm in cognitive psychology. Synthese 79:119–141

Fodor JA (1975) The language of thought. Harvester Press, London

Fodor JA (1992) The big idea: can there be a science of the mind? Times Literary Supplement July 3: 5–7

Fodor JA (2000) The mind doesn't work that way: the scope and limits of computational psychology. MIT Press, Cambridge

Fodor JA, Pylyshyn ZW (1988) Connectionism and cognitive architecture: a critical analysis. Cognition 28:3–71

Gardenfors P (1996) Mental representation, conceptual spaces and metaphors. Synthese 106:21–47

Johnson-Laird P (1983) Mental models. Harvard University Press

Laakso A, Cottrell G (2000) Content and cluster analysis: assessing representational similarity in neural systems. Philos Psyc 13:47–76

McClelland JL, Rumelhart DE (eds) (1986) Parallel distributed processing: explorations in the microstructure of cognition, Vol. 2. MIT Press, Cambridge

O'Brien G (1999) Connectionism, analogicity and mental content. Acta Analytica 22:111–131

O'Brien G, Opie J (2001). Connectionist vehicles, structural resemblance, and the phenomenal mind. In: Veldeman J (eds). Naturalism and the phenomenal mind, a special issue of Communication and Cognition. 34: 13–38

O'Brien G, Opie J (2004) Notes towards a structuralist theory of mental representation. In: Clapin H, Staines P, Slezak P (eds) Representation in mind: new approaches to mental representation. Elsevier

Palmer S (1978) Fundamental aspects of cognitive representation. In: Rosch E, Lloyd B (eds) Cognition and categorization. Lawrence Erlbaum

Parkkinen JPS, Hallikainen J, Jaaskelainen T (1989) Characteristic spectra of Munsell colors. J Opt Soc A 6(2):318–322

Pinker S (1997) How the mind works. Norton, New York

Pinker S (2002) The blank slate: the modern denial of human nature. Viking, New York

Port R, van Gelder TJ (1995) Mind as motion: explorations in the dynamics of cognition. MIT Press, Cambridge

Pylyshyn ZW (1984) Computation and cognition: toward a foundation for cognitive science. MIT Press, Cambridge

Ramsey W (1997) Do connectionist representations earn their explanatory keep? Mind Lang 12(1):34–66

Rumelhart DE, McClelland JL (eds) (1986) Parallel distributed processing: explorations in the microstructure of cognition, vol. 1. MIT Press, Cambridge

Shepard R, Chipman S (1970) Second-order isomorphism of internal representations: shapes of states. Cog Psychol 1:1–17

Shepard R, Metzler J (1971) Mental rotation of three-dimensional objects. Science 171:701–703

Swoyer C (1991) Structural representation and surrogative reasoning. Synthese 87:449–508

Tienson J (1987) Introduction to connectionism. South J Philos 26:1–16

Truitt TD, Rogers AE (1960) Basics of analog computers. John F. Rider

Von Eckardt B (1993) What is cognitive science? MIT Press, Cambridge