On connectionism and dynamical systems theory : different doesn't have to mean separate

Joe Blaylock

January 10, 2008

1 Introduction

To many in the cognitive science community, old-guard computational theories of the mind seem increasingly inadequate to explain the vast richness of lived experience. Phenomena such as insect locomotion, face recognition, or the prediction of rhythmic patterns have eluded plausible explanation in the Sense-Think-Act framework. In addition, the symbol-grounding problem and related issues continue to haunt representational realists. Increasingly, scientists are appealing to alternative, *emergentist* paradigms, to explore issues in cognition. The dominant emergentist models in cognitive science are connectionism and dynamical systems theory (DST). In this paper, I intend to defend the position that these models, while sharing many similarities, are not the same. Further, I suggest that they are both valid and valuable in different contexts.

2 The Same...But Different

Both dynamical systems theory and connectionism take the position that cognition is a high-level, emergent phenomena deriving from the interaction of a large number of smaller, individuated components. They both tend to favor systems which can be described by collections of differential equations playing out over either a series of discrete time steps or continuously over some bounded range in time. Practitioners in both camps rely on simulations using digital computers to test ideas and guide further research.

While it is not true that all dynamical systems can be simulated using any neural network[1], there exists a large class of neural networks which can emulate the behavior of any discrete or continuous-valued dynamical system to any arbitrary degree of accuracy. They can also emulate the behavior of

[1] Artificial neural networks are *the* tool of choice in connectionism.

any other neural network. Likewise, while dynamical systems theoreticians don't consider most neural networks to be dynamically interesting, preferring to instead concentrate on higher-level couplings, systems of differential equations can capture the behavior of any neural network.

It sounds as if I'm arguing for logical identification: for a point of view asserting that either connectionism or dynamical systems theory should be considered a subset of the other viewpoint. When I first proposed the topic for this paper, I thought that I was. In Beer[1], Dr. Beer asserts that a good metaphor for thinking about the relationship between dynamical systems theory and connectionism is to consider the relationship between Newtonian physics and Relativistic physics. As written, Beer subtly implies that dynamical systems theory is necessarily superior to connectionism in both predictive power and pragmatic usefulness. If one reads his analogy uncarefully, one might think that he refers to Special Relativity, and that connectionism and dynamical systems theory are *almost* the same. This is what led my thinking in the direction of identification.

From Beer[2], however, it is clear that this is incorrect. Beer most ardently intended that readers think of General Relativity, with its grossly different ideas about the nature of time and space. He wanted to point up the differences between dynamical systems theory and connectionism as being paradigmatic[2], rather than cosmetic. He has said that, just like classical and modern physicists, connectionists and dynamical systems theoreticians ask different questions, seek different types of confirmation, and tolerate different errors.

Connectionism is close to the computational theory of mind that it seeks to replace in that it tends to be disembodied. While this is not a necessary feature of connectionist models, it is traditional for the neural networks employed to be free from context, and to rely on some external filter to translate raw data into network inputs, and to translate network outputs into useful behavior. Designs tend to focus on symbol grounding and the manipulation of intermediate representations.

The artificial neural networks of connectionism are rule-bound systems of simple artificial neurons, arranged into collections called layers, which allow information to flow only in a directed fashion. They act as filters or trans-

[2] In its proper Kuhnian sense.

formers from some set of inputs to some set of outputs. Designs vary by their number and style of neurons, the number and style of interconnections between them, and by the algorithm employed to adjust the network's outputs. Typically, connectionists concentrate their efforts on matters of network architecture and training algorithm employed, and tend towards a preference for working models over rigorous analysis. We can caricature their emphasis as being on *construction* rather than explanation.

Traditionally (although not necessarily), the cognitive models of dynamical systems theory tend to rely on their context. That is, they tend to be both situated (in an environment) and embodied. Their sensation of the (possibly virtual) environment, the state space path of the system, and the behavior of the model (and subsequent impact on the environment) form a fluid whole. Thus, dynamical systems theoreticians model phenomena using collections of differential equations that describe how coupled systems change over time. They concern themselves not with the underlying architecture[3], but with paths through high-dimensionality state spaces, and the ways that those paths change as the system and the context that the system

[3]whether it be neurons or Chinese nationals

is embedded in change.

Like Oroborus, part of the input consumed by the dynamical system is its own output. Restricting study to a single part of such a framework is artificial, so dynamical systems theoreticians concentrate on the relationships betweent the parts, and on applying mature mathematical tools to help them understand the dynamics of those relationships. In caricature, they emphasize *explanation* rather than construction.

3 Different. . . But Similar Enough?

Despite these differences, perhaps Smith and Samuelson[5] are on to something. I'd like to suggest that the proper relationship between connectionism and dynamical systems theory is not as competing scientific paradigms, with the necessity that one will win and one will lose. Perhaps they relate to one another rather in the way of different computer programming languages. Equivalent in power, but having vastly different context-dependent efficiencies[4], they might appear to users to be less general than they really are, and

[4]Programming language users generally use the term *expressiveness* for this concept, although they may mean different, sometimes conflicting things, when they use this term.

less compatible.

But first, what do I mean by power, and what do I mean by efficiency? By *power*, I mean the ability of some language to express computation. Loosely, any[5] computer programming language are capable of implementing a Turing machine. Symmetrically, because they're built for the finite hardware of the real world, every computer programming language should be implementable on a Turing machine. In this sense, then, every computer programming language is formally equivalent to a Turing machine, and this is what programmers mean when they say that a language is "Turing complete". To understand what I mean by *efficiency*, consider the formal equivalence between Church's lambda calculus and Turing's machines. Any algorithm which can be expressed in one can be expressed in the other. Thus, they have equivalent power. However, there is a natural sense in which the lambda calculus is more efficient: it is simply easier to do a variety of useful operations in the lambda calculus than it is using a Turing machine.

At a higher level of abstraction, let us consider the programming lan-

---

For more on this issue, cf. Felleisen[3].

[5] . . . useful . . .

guages LISP and FORTRAN. Automata theory tells us that they are equivalent in power, yet they appear to be very different in practice. Both their syntax and their semantics diverge wildly. They're designed to make different kinds of jobs easier (that is, to express different efficiencies) and provide their users with very different high-level structures for thinking about problems. While LISP is very close to the lambda calculus and makes use of structures like arbitrary lists and closures, FORTRAN is intended as a rough abstraction of the computer's hardware, and emphasizes matrix operations.

These differences constrain the types of choices that programmers make using these languages. They shape the way that designers think about problems, the way that connect functioning is confirmed, and the way optimization is thought about. For programmers having limited time or other resources, it constrains what is actually possible. The choice of programming language becomes an important one, not because one language is better than another per se, but because the choice of language has a deep impact on the nature of the work undertaken, and the programmer's relationship to it.

As discussed above, at least some artificial neural networks can emulate any dynamical system, and any artificial neural network is describable as a dynamical system. So clearly, dynamical systems theory and connectionism offer equivelent power, in the computer programming sense. By asking different questions and seeking different types of explanation, then, it can be said that connectionism and dynamical systems theory [6] are expressing different efficiencies for different classes of problems.

4 Parting Thoughts

Understood this way, we see that the differences between dynamical systems theory and connectionism appear to be paradigmatic, but may be, in some sense, illusory. They encourage their users to think in very different ways about problems, to ask different questions and to seek different kinds of explanation. But we needn't suppose that this makes them *fundamentally* incompatible ways of looking at problems in cognition. Rather, like the computer scientist who applies the appropriate programming language paradigm to some problem at hand, or like Smith & Samuelson[5] building models

[6] Which are really just tools for thought in the same way that programming languages are tools for thought.

across emergentist paradigms to use all of the explanatory tools available, we can synthesize these different theoretical frameworks. Rather than using them as scaffolding, letting us build at the cost of mobility, we can can use them as vehicles to take our thinking to wholly new places.

References

[1] Dynamical approaches to cognitive science, Randall D. Beer; Trends in Cognitive Sciences Vol. 4, No. 3, pps. 91-99, March 2000

[2] Private Conversation, Randall D. Beer; 30 November 2007

[3] Matthias Felleisen. On the Expressive Power of Programming Languages. In Proc. European Symposium on Programming, in LNCS 432, pages 134–151. Springer-Verlag, 1990. http://citeseer.ist.psu.edu/felleisen90expressive.html

[4] Garson, James, "Connectionism", The Stanford Encyclopedia of Philosophy (Spring 2007 Edition), Edward N. Zalta (ed.). http://plato.stanford.edu/archives/spr2007/entries/connectionism/

[5] Different is good: connectionism and dynamic systems theory are complementary emergentist approaches to development, Linda B. Smith and Larissa K. Samuelson. http://www.indiana.edu/ cogdev/labwork/smithandsam.pdf

[6] Connectionism and dynamic systems: are they really different?, Esther Thelen and Elizabeth Bates http://www.psychology.uiowa.edu/cd-conference/Thelen-2003.pdf

[7] Bottles, Rebecca Walpole, http://www.99-bottles-of-beer.net/language-common-lisp-141.html

[8] 99 Bottles of Beer, Alex Ford, http://www.99-bottles-of-beer.net/language-fortran-77-760.html