Cognitive Development, 11, 571–603 (1996)

Generative Connectionist Networks and Constructivist Cognitive Development

Denis Mareschal
Oxford University
Thomas R. Shultz
McGill University

This article presents a novel computational framework for modeling cognitive development. The new modeling paradigm provides a language with which to compare and contrast radically different facets of children's knowledge. Concepts from the study of machine learning are used to explore the power of connectionist networks that construct their own architectures during learning. These so-called generative algorithms are shown to escape from Fodor's (1980) critique of constructivist development. We describe one generative connectionist algorithm (cascade-correlation) in detail. We report on the successful use of the algorithm to model cognitive development on balance scale phenomena; seriation; the integration of velocity, time, and distance cues; prediction of effect sizes from magnitudes of causal potencies and effect resistances; and the acquisition of English personal pronouns. The article demonstrates that computer models are invaluable for illuminating otherwise obscure discussions.

Computational developmental psychology is at the crossroads of what may at first appear to be two completely separate domains. Developmentalists wonder at the relevance of a discipline that draws on the tools of machine learning and computer science. How can computer models help to understand the complexities of a child developing in the real world? What similarities can there be between an infant exploring its new world and a machine trying to master an artificially created environment? Similarly, cognitive scientists and researchers in traditional artificial intelligence have tended to ignore developmental issues. Adult thought processes are complicated enough: Why should these problems be further clouded by the added task of trying to understand how cognition develops?

This work was funded in part by a scholarship from the Fonds pour la Formation de Chercheurs et l'aide à la Recherche (Quebec) and the Medical Research Council (UK), awarded to Denis Mareschal, and by a grant from the Natural Sciences and Engineering Research Council of Canada to Thomas R. Shultz. We thank David Buckingham, Jamie Henderson, and Michael Thomas for comments on an earlier version of the article.

Correspondence and requests for reprints should be sent to: Denis Mareschal, Department of Psychology, Exeter University, Washington Singer Laboratories, Perry Road, Exeter, EX4 4QG, UK. E-mail: dmaresch@singer.exeter.ac.uk.

This article challenges both these views. Computational developmental psychology is the product of a rich cross-fertilization. Nowhere is that more evident than in the study of cognitive development. Like many others (e.g., Boden, 1980; Karmiloff-Smith, 1992; Mehler & Dupoux, 1994; Rutkowska, 1993), we believe that studying development is a necessary part of any project that attempts to understand thinking. A developmental analysis accounts for how one level of competence can lead to another level (Boden, 1982). This defines constraints that, rather than confusing the issues, prune off sterile hypotheses and point to a smaller subset of plausible ones.

The value of computational modeling to the understanding of cognitive development has also been argued for elsewhere (e.g., Boden, 1989; Klahr & Wallace, 1976; Papert, 1963, 1980; Rutkowska, 1987; Shultz, 1991; Simon & Halford, 1995). Computer modeling provides a means of testing hypotheses over and above empirical work with children. Two clear advantages are the ability to test the self-consistency of a theoretical framework and the ability to work out the complex empirical implications of a given situation (Lewandowsky, 1993). Trying to formulate a developmental theory precisely enough to implement it as a computer program is in itself a learning experience because it forces the modeler to identify and evaluate the relevant environmental dimensions of the problem as well as the nature of any innate knowledge requirements.

Once a framework has been sufficiently refined, all the computer works out are the implications of a specified set of starting conditions. Any inconsistencies are made immediately obvious. In this way, the robustness of the framework's underlying assumptions can be tested. Results of the simulation can sometimes suggest empirical predictions to be evaluated with children. A successful computational model can tie together a large body of seemingly inconsistent data. This is often the case in developmental studies that have seen a sharp increase in empirical evidence, but little in the form of an overarching explanatory framework.

The primary purpose of this article is to present a novel framework for modeling cognitive development across a large number of task domains. The cascade-correlation generative connectionist algorithm (Fahlman & Lebiere, 1990) was initially developed as a learning algorithm for applied computer science purposes. However, we believe it is ideally suited for investigating children's cognitive development in natural domains. Learning occurs not just through observation, but also through an increase in the system's representational resources. This has clear implications for cognitive development.

The presentation of this new framework proceeds in two steps. First, we present theoretical arguments delimiting the ability of these types of systems to learn novel concepts. Second, we review an extensive array of task domains that have been successfully mastered by the learning algorithm.

These practical implementations sometimes lead to empirical predictions that have been or are being investigated. The appropriateness of the approach we are describing can also be evaluated by the resulting empirical work with children.

Perhaps the most immediate advance of the work reported here is that it constitutes a specific mechanistic description of how cognitive development can occur in the child. Most developmental research has focused on structural descriptions of a child's competence at any particular time but has said little about the underlying transition mechanisms (Beilin, 1994; Sternberg, 1981). This work attempts to redress that imbalance.

By applying a single mechanism to a wide range of well-established real world problem-solving tasks (the balance scale problem; seriation tasks; integration of velocity, time, and distance; and the identification of reference in the use of personal pronouns), this work also provides the beginnings of a unifying framework that embraces results across a number of divergent task domains. Indeed, a comparison of the different parameter settings and environmental biases required to complete the modeling exercises provides a common language with which to compare and contrast development on these tasks. Tasks that were designed to address radically different facets of a child's knowledge are seen to be intrinsically related in the way that information is gathered and processed.

Finally, the nature of the cascade-correlation learning algorithm argues for the plausibility of constructivist development. The models lend support to the argument that a system can develop by building more powerful representations out of previous representational structures. In its strongest reading, this modeling paradigm serves to refute Fodor's (1980) criticisms of constructivist development. Constructivism has come under attack from those more inclined toward nativism. They have argued that constructivism is an implausible approach because it is impossible to increase the representational power of any computational system (Bloom & Wynn, 1994; Fodor, 1975, 1980). Moreover, recent infant studies seem to be pointing to greater complexity in infant competence, suggesting that infants do possess very complex concepts almost immediately (e.g., Baillargeon, 1993; Spelke, 1994). We maintain that there are valid reasons for sustaining a belief in the constructivist hypothesis. Those reasons are supported and illustrated by computer models that constitute explicit counterexamples to the nonconstructivist positions.

A secondary purpose of this article (although no less important in our minds) is to illustrate the role of computational modeling in any developmental debate. Computational models can and do play a critical role in evaluating debates arising from developmental theory. To illustrate this point, we elaborate our arguments against the background of the debate between constructivists (such as Piaget) and nativists (such as Fodor).

Throughout the article, we discuss computational models that illustrate the different theoretical positions, thereby providing a concrete vehicle for evaluating those positions. Ultimately, we demonstrate that the computational approach is not as ominous as it may initially appear. It is well worth the time investment of becoming familiar with the discourse and techniques of the paradigm. Even very simple models can lead to unexpected results and clarifications.

Throughout the article, we use results from machine learning theory. These concepts are not just applicable to machines that learn, but to any physical system that learns. In other words, the results reported here are also applicable to human learning. The work developed by machine learning theorists about the limitations of the learnable and the computational resources required for learning have direct implications for the type of learning, the feasibility of learning, and the rate of learning in children. It is also our belief that importing these concepts into the study of cognitive development will inevitably result in a fertile source of future research and a better elucidation of how children make sense of the world we live in.

AN ILLUSTRATIVE ISSUE: CONSTRUCTIVISM

Kant's constructivist approach suggested a marriage between nativists and empiricists (Russell, 1961; Scruton, 1982). Knowledge was seen as having form and content. Content is acquired when interacting with the world. Form is the initial organizational structure that humans are endowed with at birth; it allows perceptual information to be cut up into meaningful categories that thus permit learning about the world. Initial knowledge includes an understanding of time, space, quantity, and causality. We construct our understanding of the world as these forms acquire content.

Piaget followed in this constructivist tradition (e.g., Piatelli-Palmarini, 1980; Siegler, 1986). He claimed that the learning process was bootstrapped by the innate presence of sensorimotor schemas. These are a form of knowledge structure that fuse together perceptual and motor experience. Through repetition, these structures are internalized and hierarchically integrated to form the basis of a more powerful representational system. Thus, according to Piaget, the child evolves through a series of progressively more powerful representational systems.

These ideas continue to resound in the works of contemporary neo-Piagetians (e.g., Case, 1985; Fischer, 1980; Fischer & Bidell, 1991) and even in the works of theorists who are usually placed in opposition to Piaget (e.g., Gelman, 1991). Yet strong arguments have been voiced claiming that Piaget's theory of development is inadequate, if not impossible. Perhaps the most striking of these objections is the paradox raised by Fodor (1980). Fodor's bottom line was that there is no known learning system that can develop new, more powerful representations. This argument is still advanced in the current literature (e.g., Bloom & Wynn, 1994).

Fodor's argument relies on the idea that, in order for a new concept to be learned, the organism must be able to represent it so as to learn its domain of applicability. But if that concept can be represented, then the concept must already be available to the organism. The corollary of this line of argument is that only concepts that are somehow combinations of protoconcepts already available to the system can be learned. Thus, it is impossible to increase one's representational power. In a sense, learning just renders explicit a conceptual system that was inherent from the beginning.[1] Clearly, if Fodor's argument holds, then constructivist development cannot exist.

To illustrate Fodor's point, it is helpful to consider Drescher's (1991) computational model of infant sensorimotor development. Drescher picked up where Piaget left off and suggested a precise mechanism by which development could occur. The mechanism is implemented as part of a simulated organism in a microworld. Knowledge is stored in the form of a three-part action schema. The actions are fixed from the start and represent motor primitives comparable to the abilities of an infant. The context and result parts of the schema are progressively filled in as the organism discovers contingencies in its environment through the execution of an action. What is respectively stored in these registers is a summary of relevant aspects of the world before and after the action has been carried out. The context does not, therefore, describe a necessary set of conditions for the action to be carried out. Rather, it describes a hypothetical state of the world wherein if the action were to be carried out, the result part of the schema would become a true description of the world. Hence, the organism acquires knowledge in the form of counterfactual assertions that link environmental context to actions and perceptual consequences.

Drescher (1991) split learning into the two subtasks of identifying the relevant features and then identifying the necessary features for an event to occur. This so called "marginal attribution" mechanism is based on detecting significant changes in the probability of occurrence of an event. If the probability of occurrence significantly changes from its base rate, then the environmental context is deemed to be relevant. If several relevant contexts or results are discovered, new sibling schemas are created, each schema corresponding to a distinct context or result.

When an unreliable action schema is found to be locally consistent (in that it gives rise to the same results for a brief period of time), a new

[1]This argument should not be confused with the trivial claim that all data-driven learning systems are constrained by the raw sensory information they receive from the environment. Fodor's point is that the conceptual system will never increase in power from learning alone. By default, therefore, any increase in power would be due to some as yet unspecified innate mechanisms.

synthetic token is created to represent the still unknown feature of the environment that will make the schema reliable. These tokens pick out aspects of the environment that are either independent of perception or that the infant has not yet learned to coordinate in its perceptual and motor schemas. The proliferation and linking of these synthetic tokens supposedly leads to the emergence of a representation of external objects existing independently of perception.

At first sight, the results of this modeling are encouraging. The organism builds an intramodal network of sensory actions in each of the visual, tactile, and proprioceptive modes by chaining together action schemas. Moreover, it begins to construct intermodal schemas, thereby learning to coordinate two different sensory modes. However, the system does not progress very far along the Piagetian road of cognitive development (never quite reaching the concept of object), and where it does succeed, it does not seem to have been following the road map descibed by Piaget.

Even though the simulated micro-organism is endowed with a mechanism for recombining its knowledge elements, it fails to generate a more powerful representational system. In fact, it fails to generate a representation of permanent objects precisely because it cannot increase its representational power. Its knowledge remains bound to the perceptual input. This is because it has no means (in its initial state) of representing nonperceptually bound information. Even the synthetic items are perceptually bound in the sense that they are temporary fillers for a not fully understood environmental context. These schemas are replaced by more reliable perceptual schemas when they become available. Thus, the best that this model could hope to achieve is to elaborate a complex network of sensorimotor information. In the sections that follow, we describe a more appropriate approach to modeling.

THE CONNECTIONIST MODELING APPROACH

The first step of any computational approach is to identify a set of modeling tools appropriate for investigating the issue in question. In this case, connectionist methods are the ideal tool for modeling the development of cognitive processes because learning and development in connectionist networks depend on both the internal state of the system and the external state of the environment. Connectionist networks are profoundly interactivist in nature. Hence, this paradigm argues for a renewed focus on interactive processes in cognitive development (Bates & Elman, 1993; Karmiloff-Smith, 1993; Plunkett & Sinha, 1992).

When first approaching connectionism, one may be a little put off by the apparent complexities of its mathematical formalisms. The presentation in this section is not meant to be rigorous, but rather, to give the reader a sufficient grasp of the paradigm to evaluate the arguments we present later in the article. Interested readers may wish to consult Rumelhart and McClelland (1986) or Hertz, Krogh, and Palmer (1991) for excellent introductions to the more formal aspects of connectionist modeling. The key concept behind connectionist information processing is the idea of collective computation. The overall behavior of a network is not determined by any single element in it. Instead, it is a holistic approach in which the behavior of the network emerges as a result of the behavior of all its constituent parts. An analogy that may help to visualize this process is to think of a school of fish swimming in the sea. The overall undulation and flow of the school cannot be attributed to the swimming patterns of any single fish. It arises out of the complex interactions between all of the fish in the school. Similarly, the computational results of a network arise out of the complex interactions of the simple computations carried out by the elements of the network. It is the changing global behavior of the network that is taken to model the child's development.

A network is made up of simple units (Figure 1a). Each unit attends to the information being fed into it, and if it identifies a signal (that information exceeds some critical amount), then it emits a signal as well. Otherwise, the unit remains at its resting level of activation. The processing in any single unit is independent of the processing carried out by its neighbors.

Information flows between the units along weighted communication lines. It is the relative strengths of these weights that determine the global behavior of the network. If the weights change, the behavior changes. As a result, the network can adapt its behavior by changing its weights. Learning

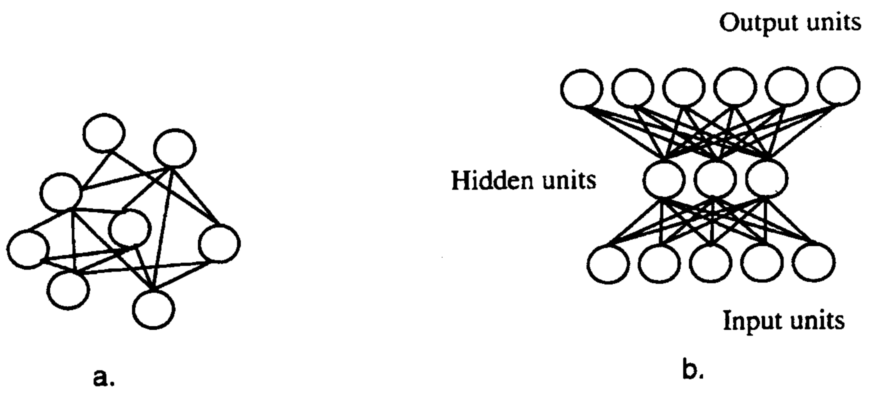

Figure 1. Two connectionist networks. Solid lines represent weighted communication lines and circles represent simple processing units. 1a shows a generic, unstructured network. 1b shows a network with six output, three hidden, and five input units.

in the network occurs by slowly changing the weights until the global behavior has reached some desired state.

Suppose now that some of the units receive information from the environment (input units) and other units send messages to the environment (output units; Figure 1b). The network can then learn to interact with the environment. More important, the units inside the network (hidden units) process the information flowing from the input units. As a result, activation across the hidden units collectively forms an internal re-representation of input information. A straightforward extension of this idea is to consider a network that is not actually interacting directly with the environment, but that is a specific module inside a larger system. The labeling of the units remains the same, but the input units now receive information from some other module, and the output units now send information out to some other module. The hidden units are now those units internal to the module across which is formed a collective re-representation of the information having entered the module. Moreover, there is no inherent reason why there cannot be more than one layer of hidden units with complicated connectivity between the layers. A large part of modeling with static connectionist networks (networks that have a fixed number of units and connections) is to identify an appropriate architecture for the task being modeled. Note, finally, that because all weights in the network can be changed, the learning of a task consists of learning an appropriate internal representation (adjusting the weights between the input units and the hidden units) and learning how to exploit that internal representation (adjusting the weights between the hidden units and the output units).

In typical networks, the amount of input entering a hidden unit is computed as the sum of the weighted activations of the units feeding into it:

$$X_i = \sum_j a_j w_{ij} \qquad (1)$$

where X_i is the net input to (hidden) unit i, a_j is the activation of sending unit j, and w_{ij} is the connection weight between units i and j. The net input to the hidden unit is then passed through a nonlinear squashing function:

$$y_i = \frac{1}{1 + e^{-X_i}} - 0.5 \qquad (2)$$

where y_i is the resulting activation of (hidden) unit i, e is the base of the natural logarithm, and X_i is the net input to that same unit i given by Equation 1.

The activation function described in Equation 2 and plotted in Figure 2 is typically called a sigmoid function. It describes how a single unit's response depends on the total amount of activation being fed into it. The response is nonlinear; the activation of this unit is not proportional to the

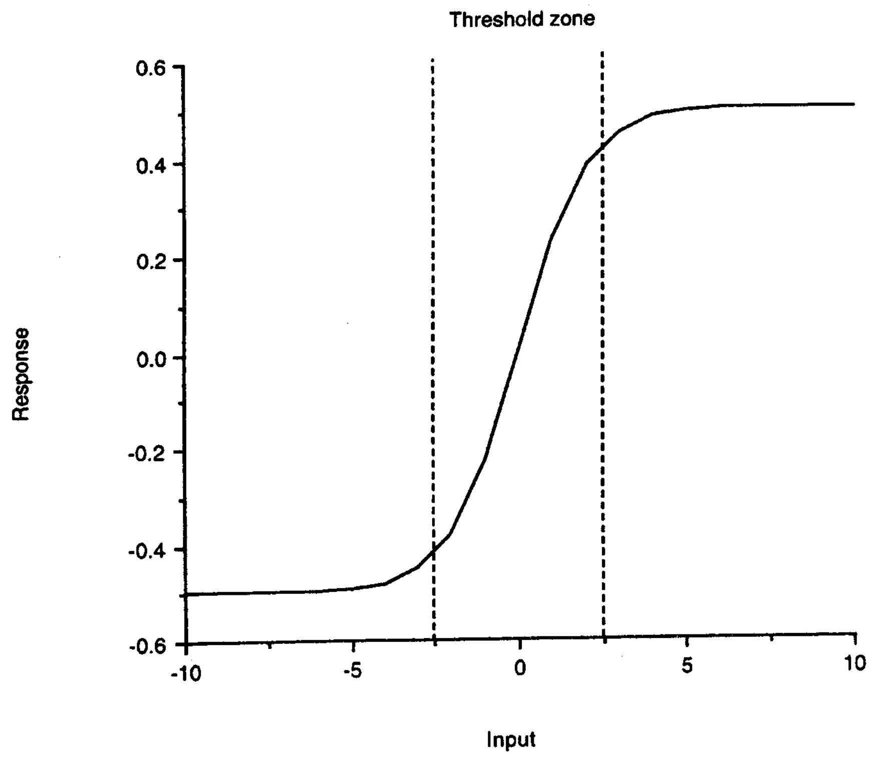

Figure 2. Plot of the sigmoid activation function in Equation 2. Units with such an activation function remain off until their net input exceeds a threshold.

signal it is receiving. In fact, this response is very similar to a threshold mechanism. When the net input is below a critical threshold zone, the unit remains at its resting level. If the net input continues to increase, it eventually reaches a threshold zone in which very small changes in input produce large changes in activation. Increasing the input even more (past the threshold zone) leads to no further activation changes. Hence, the unit tends to be either active or inactive. It is this quasibinary behavior of the unit that allows it to recode (or re-represent at a microcomputational level) the information fed into it. It is no longer possible to retrieve the exact nature of the input from the unit's activation level.

Connectionist networks with a fixed architecture (static connectionist networks) have been used to model vocabulary growth in children (Plunkett, Sinha, Moller, & Strandsby, 1992), performance on the balance scale task (McClelland, 1995), and the development of object permanence behaviors in infancy (Mareschal, Plunkett, & Harris, 1995). A full discussion of the range of tasks that have been modeled with static connectionist networks, and their implications for child development, can be found elsewhere (Elman, Bates, Karmiloff-Smith, Parisi, & Plunkett, 1996; Plunkett, Karmiloff-Smith, Bates, Elman, & Johnson, in press).

REPRESENTATIONAL CAPACITY OF A NETWORK

Because information in a network is internally re-represented across the hidden units, it seems intuitively plausible that the network's representational capacity is directly related to the number of hidden units it has. A number of mathematical proofs exist that explore different networks' abilities to represent certain relationships (e.g., Hornik, Stinchcombe, & White, 1989). We focus on Cybenko's (1989) proof because of its straightforward geometric interpretation. Cybenko's theorem states that a particular neural network architecture can be used to approximate arbitrarily well almost any input to output mapping (the underlying function must be bounded, i.e., all of its values must be finite). The proof of this theorem is rather involved, but essentially boils down to the following argument.

First, this theorem is limited to feedforward networks with a single layer of hidden units (e.g., Figure 1b). The weights in the network can take on any positive or negative value. The activation of the input units is determined by the input stimulus. The activation of each hidden unit is determined by a nonlinear monotonic function of its input and resting threshold. An example of such a function is the sigmoid function (Figure 2), but many others also satisfy the requirements of the theorem. The activation of an output unit is simply the weighted sum of the activation leading into it. Each output unit is superimposing scaled responses from the hidden units that fire in different areas of the input domain according to their threshold and the sign of the weights leading into them from the input units.

The threshold determines the midpoint at which there is a transition between the low state and the high state in the unit's activation. In Figure 2, the threshold is equal to 0. Modifications in the threshold result in moving the activation curve in Figure 2 back and forth along the input axis, thereby changing the total amount of weighted input required to invert the unit's state. Note that changing the sign of the threshold and the weights from the input units results in swapping the areas of the input domain that correspond to a high state with those that correspond to a low state (e.g., +0.5 in the negative input region and −0.5 in the positive input region for the function in Figure 2). In this case, the threshold would now determine a transition from a high state to a low state of activity.

What Cybenko (1989) did was to show that such a system can approximate any response function at the outputs arbitrarily well, simply by increasing the number of hidden units. That is, any response can be produced at a particular output unit by superimposing a sufficient number of finely balanced sigmoid functions. Figure 3 shows how this can be done. The curve in Figure 3a represents the response function to be approximated. Curves 3b through 3d show three sigmoid functions implemented by three successive hidden units. Finally, the curve in Figure 3e shows the result when those in 3b, 3c, and 3d are added together (by an output unit) with the appropriate

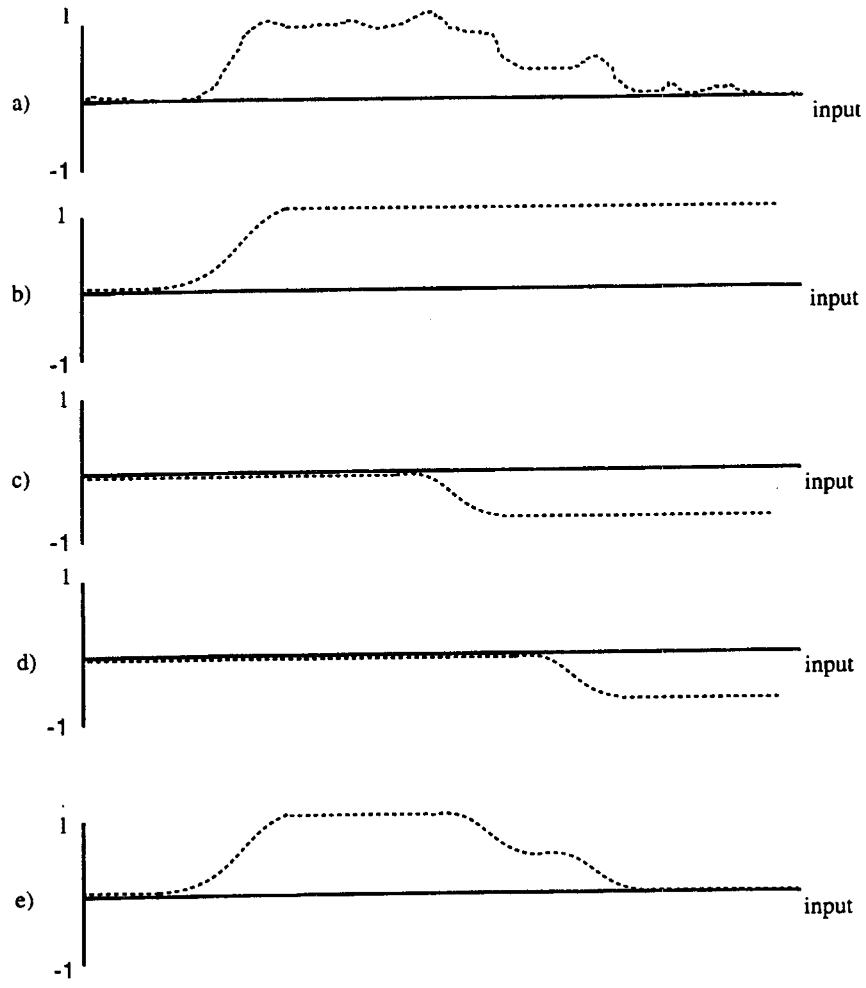

Figure 3. Approximation of a response function by superposition of sigmoid functions. The curves in b, c, and d are added to give an approximation (*e*) to the target function (a).

scaling weights. Clearly, the sum is not exactly the same as the target function, but finer detail could be obtained by adding in more sigmoid functions.

A network is limited in the number of functions it can implement, and the precision with which it can implement them, by the number of hidden units available to it. In the rest of this article, we discuss a special class of networks called generative connectionist networks. These networks increase their number of hidden units during learning. From what we have seen, it is clear that networks that increase the number of hidden units during learning increase their representational capacity during learning. This has direct implications for modeling cognitive development.

LEARNING REFORMULATED

Fodor's (1980) argument about the impossibility of constructivism suggests the need for a precise formulation of what is learnable—that is, a precise formulation of what constitutes learning so that the limits of what can be learned by a particular method can be defined. Within the domain of machine learning, Valiant (1984a, 1984b) presented a framework that explicitly addresses the acquisition of concepts. His learnability theory tries to establish the limits of the learnable. Learning itself is taken only to be the gradual self-driven acquisition of new knowledge.

Valiant (1984a, 1984b) argued that the study of learning involves identifying a learning procedure and investigating the class of concepts that the procedure can learn within a feasible time. For computational reasons, Valiant suggested that a feasible time scale is polynomial time. The number of time steps required to learn a concept is some polynomial function of the parameters in the learning procedure. That is, the time required to learn a concept is a finite sum of higher order terms of a parameter in the learning procedure (e.g., time $= 2 + 3n + 5n^2 + n^3$, where n is the number of weights in a network). This can be contrasted with the longer exponential time in which the number of time steps is some exponential function of the parameters. One advantage of limiting ourselves to studying concepts learned in polynomial time is the property of additivity. If two separate concepts can be learned, each in polynomial time, then learning some combination of these concepts can also be learned (in polynomial time) because adding together the time requirements for each step still produces a polynomial function of the parameters in the learning procedure. In other words, if it is possible to learn all the separate parts of a task in a feasible time, then it is also possible to learn the whole task in feasible time.

Valiant's (1984a, 1984b) second idea was to consider only probabilistic learning procedures. Learning happens through experience with examples of a concept. The examples are encountered according to some probability distribution determined by the environment. A teacher provides a signal specifying whether a particular encounter is an example of the concept. However, the teacher's information does not have to be perfect. It can be erroneous on a small proportion of examples. Under these conditions, Valiant presented a mathematical proof showing that such learning procedures are able to acquire a concept that has a high probability of closely approximating a target concept. This is sometimes referred to as Probably Approximately Correct (PAC) learning.

His formal proof assumed that the world is represented by a set of features with binary values (e.g., present or absent). A system is then said to have learned a concept if it can recognize almost all the examples of the concept in the environment (i.e., it has implemented an appropriate response function). An exemplar is described by a critical finite set of features (common to all exemplars) as well as an unlimited number of irrelevant features. Learning a concept consists in identifying the relevant and ignoring the irrelevant features of the concept. Learning is feasible because only the values of the relevant features have to be attended to.

The probabilistic formulation of learning makes it possible to define highly convergent learning. An exhaustive search of the universe is not required. This distinguishes the approach from the traditional machine learning ones in which concepts are induced from insufficient information.

In summary, the core of Valiant's work implies that feasible learning must occur within reasonable temporal bounds and is probabilistic in that it can always misclassify examples of a concept. At the end of learning, there is a high probability that the concept learned is approximately the target concept.

GENERATIVE ALGORITHMS CAN LEARN ANY LEARNABLE RELATIONSHIP

As noted earlier, connectionist networks with a single layer of hidden units can approximate arbitrarily well any input to output relationship given enough hidden units (Cybenko, 1989). In terms of learning, a problem arises because the more units there are in the network, the more weights there are in the network and, hence, the longer it takes to find an appropriate solution (if one can be found). In other words, although it can be shown theoretically that there exists a network that can reproduce any desired behavior, it is not clear that this network can be found or that the behavior can be learned by the network in feasible time.

Baum (1989) showed that by limiting our interests to input–output relationships that are learnable in polynomial time (as in Valiant's framework), it is always possible to find a connectionist network that will learn any such relationship. The structure of his argument is simple, although the details involve complex principles of graph theory. The first part of the argument is to show that any representation of a target relationship produced by any learning algorithm can also be implemented by a connectionist network in polynomial time. In other words, if a solution can be found by any means available, then that solution can be implemented as a neural network in polynomial time. Call this the implementation step. The second part of the argument makes use of Valiant's definition of *learnable:* Learnable concepts are those concepts learnable in polynomial time. Although we may not know how to learn the concept, we know that a procedure exists for learning the concept. Call this the learning step. It follows that any learnable concept can be acquired by a network that first uses an appropriate algorithm (the learning step) and then maps the resulting solution onto a corresponding network architecture (the implementation step). Because both steps in this process occur in polynomial time, their sum also occurs in polynomial time (cf. the foregoing additivity property in the description of polynomial time). Hence, the procedure made up of these two steps will satisfy Valiant's time requirement.

This awkward procedure is not the most natural way of training a network, but it does show that there exists at least one way for a neural network to learn any learnable relationship. The key is that the procedure has to be able to construct an appropriate architecture. In this broad sense, the family of generative neural networks defines a class of universal learners capable of learning any learnable concept (by Valiant's criteria). Baum's work provides an existence proof but it does not point to a specific learning mechanism.

GENERATIVE ALGORITHMS ESCAPE FODOR'S PARADOX

In light of these arguments, Quartz (1993) suggested that generative connectionist algorithms afford a means of escaping from Fodor's paradox. Clearly, because a network's representational power increases as the number of hidden units increases, generative algorithms increase their representational power through learning. There are functions that cannot be implemented by a smaller network but can be implemented by one containing additional units.[2]

Before moving on, we must point out two caveats to Quartz's argument. First, real implementations have limited resources. As a result, networks

[2]It could be argued that for Fodor, a system is fixed, such that the network at time (t) with (n) hidden units is not the same system as the network at time $(t+?)$ with $(n+1)$ hidden units. However, such a static definition of a system is very unlike biological systems. Moreover, this cannot be said of generative networks with a fixed total number of potential units to recruit in reserve. These units are part of the system even before they are installed, but do not contribute to computations carried out by the system. Hence, the system still develops through periods of increased computational power as the units are brought into the network.

cannot continue adding hidden units indefinitely and therefore do have an upper bound to the complexity of the concepts they can learn. In this restricted sense, the set of achievable concepts is determined at the onset of learning.

Secondly, Molenaar (1986) suggested that even static networks can escape from Fodor's paradox if a different measure of representational power is used. To understand Molenaar's suggestion requires a bit of a conceptual side-step. The key idea is that he is not discussing the power of feedforward networks as we have done so far. Rather, he is interested in describing the behavior of self-organizing networks of the type described by Grossberg (1982). These networks resemble the unstructured networks in Figure 1a. Activation flows in all directions rather than only forward. Structured oscillations can arise from the complex interactions of the units in these networks. Molenaar pointed out that these cycling patterns can arise and be modified as part of the learning process (i.e., as a result of small changes in the weights of the network). This, according to Molenaar, is a form of development in static networks that can escape Fodor's paradox.

The learning theory arguments described in the previous sections are all encouraging as to the potential power of generative algorithms, but they leave unanswered the question of a precise mechanism. In short, they provide a framework in which to deal with learning but reveal little about real world applications of that learning. To answer the question of how these ideas bear on the development of children's cognitive skills, specific mechanisms must be proposed, explored, and tested. Hence, we turn now to the description of a specific generative algorithm: cascade-correlation.

THE CASCADE-CORRELATION ALGORITHM

Cascade-correlation is a generative learning algorithm for connectionist networks (Fahlman & Lebiere, 1990). Like other generative algorithms, cascade-correlation constructs its own network topology during learning by adding new hidden units. There are two alternating, recurrent phases in cascade-correlation: an output phase in which the connection weights going into output units are adjusted in order to reduce the network's performance error and an input phase in which new hidden units are selected and installed in the network. The name cascade-correlation stems from the way that new hidden units are recruited and installed. In the input phase, the candidate hidden unit whose activations come to correlate best with the network's current error is selected for installation. Selected hidden units are installed into the network in a kind of cascade, such that each new hidden unit receives input from the input units and from any previous hidden units.

In this section, we provide a reasonably detailed description of the cascade-correlation algorithm. The level of detail illustrates the nature of the precision with which one can specify the mechanisms involved in development. Details are discussed in terms of unit activations, the output phase, the input phase, and weight adjustment.

Computing Unit Activation

When a pattern of input activation is presented on the input units of a cascade-correlation net, that activation is passed forward to any hidden units and eventually to the output units. Effectively, each hidden and output unit examines its inputs and decides how active it is going to be. These properties are characteristic of many real neurons and can be computationally critical for the successful operation of artificial neural nets such as those created by the cascade-correlation algorithm. In most cases, hidden and output units have sigmoid activation functions, as described in Equation 2.

There is an option in cascade-correlation for output units to have linear activation functions. A linear activation function would mean that the net input to a receiving unit, as given in Equation 1, is directly translated into the receiving unit's activation. Such a linear activation function would be useful for output units that need to represent quantitative, as opposed to qualitative, values. The idea is to predict the amount of the output instead of whether the output is present or absent.

Unless otherwise specified, all our cascade-correlation models use hidden and output units with sigmoid activation functions. Occasionally, when learning quantitative rather than qualitative functions, they employ output units with linear activation functions.

The Output Phase

The output phase of cascade-correlation is characterized by adjusting the output-side weights of input and hidden units in order to reduce the network's error. The output-side weights are those connection weights leading into output units. The network's error is computed as the discrepancy between the network's predictions and what actually occurs in the environment. The network's prediction is specified by a pattern of activation across the output units. The actual environmental result is given by a target pattern of activation across the output units. More formally, error (E) is

$$E = \sum_{o} \sum_{p} \left(A_{op} - T_{op}\right)^2 \tag{3}$$

where Ao_p is the computed activation of output unit o on pattern p and T_{op} is the target activation for that unit on that pattern. A pattern is a particular pair of input and output activation values. A set of many such pattern pairs defines the learning environment of a cascade-correlation model.

The cascade-correlation algorithm tries to improve a network's predictive ability by minimizing E. This is done by adjusting all of the output-side connection weights. The method for such minimization is discussed later in more detail (see the Weight Adjustment section). For now, notice that the information that drives learning is more than just a binary reinforcement signal of right or wrong. Like many other feedback-driven connectionist algorithms, cascade-correlation requires the presentation of a fully specified target, in the form of the complete, required pattern of activations across the output units. In many cases of cognitive development, the assumption of the availability of such full information is warranted. The target (T_p) represents the way the environment actually appears to the child. The output activation (A_p) represents the child's prediction of what the environment would look like, for example, after some transformation is applied. Learning is then a matter of minimizing the absolute discrepancy between expectations and outcomes.

A cascade-correlation network begins life as a set (whose number is specified by the modeler) of only input units and output units initially connected with randomly chosen weights (Figure 4a). Error reduction through weight

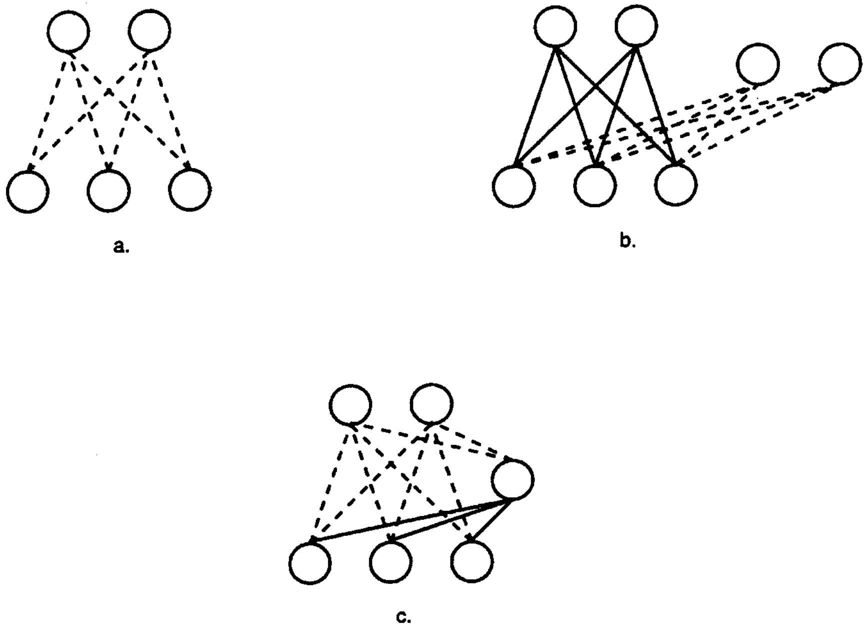

Figure 4. Three generic cascade-correlation nets. Input units are at the bottom, output units at the top. Modifiable connections are represented by dashed lines and nonmodifiable connections are represented by solid lines. 5a and 5c represent output phases in which only output-side weights are being adjusted. 5b represents an input phase in which two candidate hidden units are being trained and evaluated for their ability to detect network error.

adjustment proceeds until either E stagnates or some specified number of epochs pass without the problem having been learned. An epoch is a sweep through all of the input–output training patterns. At this point, because E is no longer being reduced, or if too much time has elapsed without the algorithm having learned to solve the problem, there is a shift to the input phase to increase the net's representational power by recruiting a new hidden unit.

The Input Phase

During the input phase, connection weights leading into the output units are temporarily frozen so that they can no longer change (cf. Figure 4b). An arbitrary number (eight, by default) of candidate hidden units are connected with random weights from the input units and any existing hidden units. The weights leading to each candidate unit (input weights) are then adjusted to maximize the absolute value of the modified correlation (C) between the activation of that unit and the error at the output units, across all patterns:

$$C = \frac{\sum_{o}\sum_{p}\left|\left(h_p - \langle h\rangle\right)\left(e_{op} - \langle e_o\rangle\right)\right|}{\sum_{o}\sum_{p}\left(e_{op} - \langle e_o\rangle\right)^2} \tag{4}$$

where h_p is the activation of the candidate hidden unit for pattern p, $<h>$ is the mean activation of the candidate hidden unit for all training patterns, e_{op} is the error at output o for training pattern p, and $<e_o>$ is the mean error at output o for all the training patterns.

Input training continues until C stagnates or a specified maximum number of epochs has elapsed. At this point, the candidate unit with the largest C is installed into the network, and all other candidate units are discarded. The input weights to the newly installed hidden unit are then frozen so that they can no longer change, and the new hidden unit is allowed to send output to all of the output units. The algorithm returns to the output phase with the added power of a new hidden unit that is particularly good at detecting the network's current error. Note that, thanks to the correlation training mechanism of the candidate hidden units, the new unit has been specifically trained to encode (or internally re-represent) some feature of the input (the environment) that still leads to error in the network's performance.

Weight Adjustment

Both input and output phases require that connection weights be adjusted. In the input phase, this weight adjustment is in the service of maximizing the correlations between candidate hidden unit activations and the residual network error. In the output phase, the weight adjustment is in the service of minimizing network error. The weight adjustment algorithm is the same in both phases of cascade-correlation. It is called *quickprop* (Fahlman, 1988), because it is much quicker than back-propagation (the conventional technique for training multilayered static networks). The main reason for the added speed is that quickprop uses second- as well as first-order information to adjust weights. An overview of quickprop is provided here in the context of error minimization.

The network's error, given in Equation 3, can be viewed as a function of each weight in the network. The assumption is that there is an optimal value for each weight in order to minimize error. That is, the weight cannot be either too large or too small for the error to be minimized. It is assumed that in the neighborhood of the current weight value the function approximates a parabola with arms opening upward (Figure 5). A parabola, like any function, is at a minimum (or maximum) when its first derivative is 0. The exact shape of the function in Figure 5 is unknown, but the slope at any particular value of X_i for sigmoid units may be computed as:

$$\text{slope} = \frac{\partial E}{\partial w} = \frac{\partial E}{\partial A}\frac{\partial A}{\partial x}\frac{\partial x}{\partial w} = (T - A)((0.5 - A)A)a_j \qquad (5)$$

where A is a computed output activation, T is the corresponding target output activation, and a_j is the activation of sending unit j.[3] If Equation 5 computes a negative slope (corresponding to a downward slope), then we know that the weight is too small and we must increase the weight to continue reducing error. But if Equation 5 computes a positive slope (corresponding to an upward slope) then we know that the weight is too large and must be decreased to continue reducing error (see Figure 5). The amount of weight change is proportional to the slope, so the magnitude of the weight change is larger on a steeper slope than on a smaller slope. A general problem with using only slope information, however, is that we do not know exactly how big a weight change to make. If the slope is changing rapidly, then we should be cautious and not rely too much on the current measurement of the slope. Conversely, if the slope is not changing or is changing very slowly, then we can be more confident that the current measurement of the slope is representative of that region of the weight space. When being cautious it is more appropriate to make small weight changes, whereas when being confident it is more appropriate to make large weight changes.

Fahlman's (1988) solution in quickprop was to supplement first-order slope information with the second derivative of error with respect to weight, also known as curvature. Curvature is an index of how fast slope changes with respect to changes in weight. The more the curvature, the faster the slope changes as weight changes. Curvature can be estimated as

[3]An advantage of using sigmoid (and asigmoid, and hyperbolic tangent) activation functions, as opposed to other nonlinear activation functions, is that their slopes can usually be expressed as a simple combination of the original activation function.

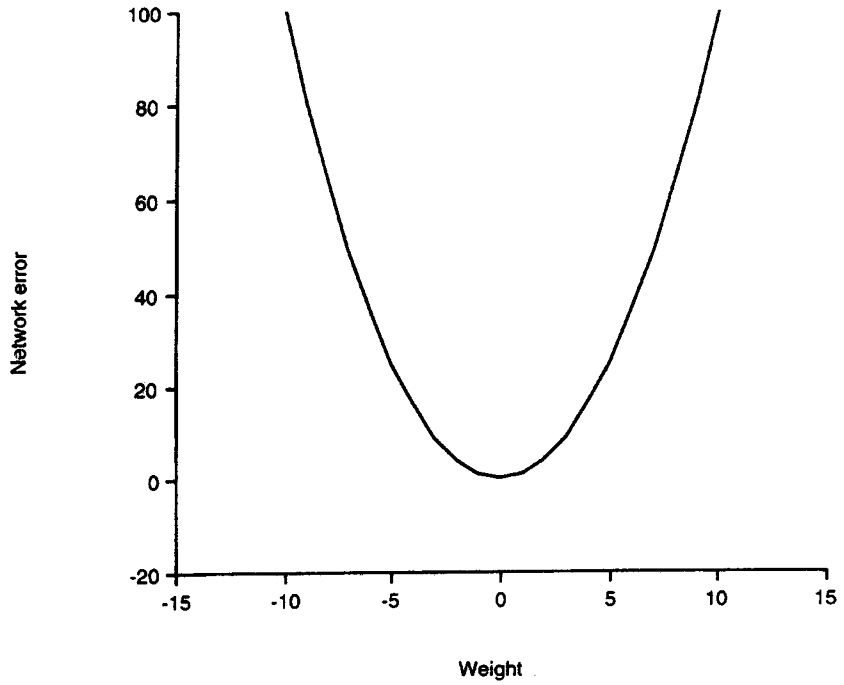

Figure 5. A hypothetical curve relating network error to the size of a connection weight. The learning algorithm tries to find a set of optimal-sized connection weights to minimize network error.

$$c = \frac{s_1 - s_2}{w_1 - w_2} \tag{6}$$

where c is curvature of the error function, s_1 is the slope at Time 1, s_2 is the slope at Time 2, w_1 is the weight at Time 1, and w_2 is the weight at Time 2, and where Time 1 and Time 2 are two consecutive steps in time. In simple words, the quickprop algorithm uses the slopes and weights of the error function at two separate epochs, the current epoch and the previous to estimate curvature; that is, curvature is estimated as the amount of slope change per weight change.

The weight adjustment is then computed as

$$\Delta w = \frac{-s_2}{c} = \frac{s_2(w_2 - w_1)}{s_1 - s_2} \tag{7}$$

where Δw is the connection weight adjustment, s_2 is the current slope, and c is curvature as defined in Equation 6. In other words, weight adjustment is a direct function of slope and an inverse function of curvature. As in the back-propagation algorithm, we increase weight when the slope is negative and decrease weight when the slope is positive, and we make a larger weight change on a steeper slope. But we now also make a larger weight change with a smaller curvature and a smaller weight change when the curvature is large. A steep slope signals that we are not yet near the error minimum. A smaller slope signals that we should make a small weight change because we may be approaching the error minimum. In either case, slope is scaled by the curvature. The more such derivative information that we have about the error function, the more rapidly we can alter weights to reach the minimum error. This presentation of quickprop is still somewhat simplified. There are complications to deal with the first epoch of weight change and other computational considerations (see Fahlman, 1988, for more detail).

A PSYCHOLOGICAL INTERPRETATION OF CASCADE-CORRELATION

How are these mathematics and this computational machinery related to psychological processing and cognitive development? One way to answer this question is to compare cascade-correlation with what is perhaps the most prominent psychological model of cognitive development, Piaget's (1977) adaptational model. Piaget held that the child develops cognitively by adapting to the environment. Piaget further specified that adaptation has two contrasting poles: assimilation and accommodation. In assimilation, the child transforms external information so that it fits into her or his existing knowledge structures. In accommodation, the child transforms her or his internal knowledge structures so that they fit external information. Both assimilation and accommodation are supposed to occur to some degree in every cognitive encounter, but there are times when one aspect dominates over the other. Often they occur in recurring phases, for example, attempted assimilation, accommodation, and successful assimilation. Assimilation and accommodation have long been considered interesting ideas about cognitive development, but their operation was never adequately specified in Piagetian theory (Boden, 1980, 1982).

We have argued that assimilation corresponds to a situation in which a network more or less successfully generalizes to patterns it has not been trained on (Shultz, Schmidt, Buckingham, & Mareschal, 1995). This happens when the novel patterns are sufficiently similar to patterns that the network has already learned. The contrasting process of accommodation corresponds to the network having to alter its underlying structure by recruiting new hidden units. Such underlying structural changes provide the net with representational power that it previously lacked, allowing it to represent and process novel patterns without error.

We also argued that cascade-correlation provides a way to think about the intermediate process of assimilative learning, which Piaget had no way of conceptualizing (Shultz et al., 1995). Assimilative learning would presumably occur when the child learns new material without undergoing drastic cognitive change. Not every piece of learning should require a qualitative structural change or increase in representational power. In cascade-correlation, assimilative learning would occur through the process of quantitative adjustment of connection weights, as it occurs in the output phase, without the recruitment of any new hidden units.

Because these processes have a precise mathematical and computational interpretation in cascade-correlation, they are now sufficiently specified that they can be investigated more thoroughly and systematically than in the past.

CASCADE-CORRELATION MODELS OF COGNITIVE DEVELOPMENT

We turn next to a brief review of some cascade-correlation models of cognitive development. A detailed description of the dynamics of the networks within the different task domains can be found in the articles dedicated to each of the models. Successful models have so far included balance scale phenomena; seriation; the integration of velocity, time, and distance cues; prediction of effect sizes from the magnitudes of causal potencies and effect resistances; and acquisition of English personal pronouns.

Balance Scale

In the balance scale task, a child is presented with a rigid beam on which a number of pegs have been placed at different distances to the left and right of a fulcrum. The experimenter places some number of equally valued weights on a peg on the left side and on a peg on the right side. The child's task is to predict what will happen when supporting blocks are removed. Will the scale tip to the left, to the right, or will it balance?

Siegler (1976, 1981) found that children progress through four distinct rule-based stages on this task between the ages of 5 and 17 years of age. Children in Stage 1 predict outcomes on the basis of how many weights have been placed on each side. In Stage 2, children continue to use weight information and begin to use distance information when the two sides have equal weights. By Stage 3, they are using weight and distance about equally, but become confused when one side has greater weight and the other side has greater distance. In the final Stage 4, children perform correctly on a wide range of balance scale problems, suggesting to some that they may be using the so-called torque rule. The torque on one side is product of weight and distance. The torque rule would involve computing and comparing the torques on each side of the fulcrum.

The four-stage sequence of performance on the balance scale has been simulated in cascade-correlation nets, as has the so-called torque difference effect (Ferretti & Butterfield, 1986), wherein balance scale problems with large torque differences are easier for children to solve than problems with small torque differences (Shultz, Mareschal, & Schmidt, 1994; Shultz & Schmidt, 1991). The torque difference is the absolute difference between the torque on one side and the torque on the other side.

These cascade-correlation networks were trained to solve balance scale problems with five weights and five pegs on each side of the fulcrum. Of the 625 possible balance scale problems of that size, 100 were randomly selected for the initial training set. Balance scale problems were described on the input units in terms of the number and position of the weights on each side of the fulcrum. The behavior of the scale (tip left, tip right, or balance) was described as targets for the output units. There was a strong bias in the training set in favor of so-called equal distance problems in which the weights were placed at equal distances from the fulcrum, but with varying numbers of weights. This reflects the assumption that, although children have many experiences in lifting different numbers of objects, and noticing that more objects create more torque, they have relatively little experience at placing objects at differing distances from a fulcrum (McClelland, 1989). In each epoch of training a new balance scale problem was randomly selected, with the same bias toward equal distance problems, and added to the training set. This reflects the assumption that children gradually encounter more instances of the problem being learned.

Diagnosis of network behavior mimicked the way such diagnosis is done with children, by examining the pattern of performance across six different types of balance scale problems (Siegler, 1976, 1981). These test problems were selected independently from the training problems and were balanced for torque difference so as to provide a clearer diagnosis of rule following.

Previous models of balance scale performance had failed to capture Stage 4 and did not attempt the torque difference effect. Indeed, the torque difference effect seems particularly awkward for symbolic rule-based models of the balance scale because such rules are typically sensitive to the direction of weight and distance differences but not to their amounts (Langley, 1987; Newell, 1990; however, see Schmidt and Ling, 1996, for a possible solution to this problem). Even a connectionist model of the balance scale failed to reach a stable Stage 4 level of performance (McClelland, 1989), and it is noteworthy that this model involved a static, rather than a dynamically generated, network topology. McClelland's static network model is able to capture the torque difference effect, but it does require hidden units segregated for weight versus distance information in order to capture the proper stage progressions (McClelland, 1995; Schmidt & Shultz, 1991).

Seriation

The four stages of development on Piaget's (1965) seriation task have also been simulated with cascade-correlation nets (Mareschal & Shultz, 1993). In the seriation task, the child is asked to sort by length a set of sticks of different lengths that are arranged in a random fashion. In Stage 1, children move the sticks about randomly or seem unable to make any move. In Stage 2, children sort a few sticks, but seem unable to complete the entire array. By Stage 3, they achieve a complete sort by a trial and error process, where moves often have to be corrected. Finally, in Stage 4, children complete a full sort without errors, by using a systematic procedure such as moving the smallest out of order stick to its correct position.

The seriation simulations also managed to capture well-known perceptual effects on seriation tasks such as the tendency for seriation difficulty to increase with decreases in size differences among the sticks (Elkind, 1964; Kingma, 1984). A novel prediction was generated when it was discovered that cascade-correlation nets did better with more disordered arrays. This prediction was subsequently confirmed with young children, who were found to also have relatively more difficulty choosing a move when the array became less disordered (Mareschal & Shultz, 1996).

Modular networks were required to learn to seriate and to capture the relevant psychological phenomena. One cascade-correlation net was trained to select the stick to be moved, and another cascade-correlation net was trained to insert the stick in its correct position. Both modules received the same input, namely the current configuration of the array of sticks. Current positions of the different length sticks were coded on the input units. The position of the stick to be moved was coded on the output units of the *which* module, and the position to which a stick should be moved was coded on the output units of the *where* module.

The networks were trained and tested on arrays of six different lengths. Small biases in the training set toward smaller and nearly ordered arrays produced the best results. It is reasonable to assume that smaller arrays are more likely to occur in the child's environment than large arrays, and that nearly ordered arrays are more likely to elicit an attempt to finish the sort than are highly disordered arrays. During testing, after an array was presented to the network and a move computed and executed, the resulting new array was presented as input. Sorting continued in this way until the sort was completed or a move was repeated. This corresponds to the child doing a sort on the table, rather than in his or her head. In other words, it is not necessary for the child to remember the position of each stick; it is only necessary to look at the current array and make a move and to keep doing this until the sort is finished. Seriation stages in the networks were diagnosed just as they are with children.

Rule-based models have had much worse luck capturing seriation phenomena. They did succeed in modeling the systematic nature of seriation performance at each of the four stages, but did not manage transitions between stages, perceptual effects, or the variation typical of children's seriation performance (e.g., Baylor, Gascon, Lemoyne, & Pothier, 1973; Young, 1976).

Velocity, Time, and Distance

Cascade-correlation nets have also simulated rule-based stages in the integration of velocity, time, and distance information (Buckingham & Shultz, 1994). In classical physics, velocity = distance/time. Thus, distance 15 velocity × time, and time = distance/velocity. Given the task of learning to predict one dimension (e.g., velocity) from knowledge of the other two (e.g., time and distance), nets typically progressed through an identity stage (e.g., velocity = distance), followed by an additive stage (e.g., velocity = distance + time), and finally the correct multiplicative stage (e.g., velocity = distance/time). Many of these stages have been found with children (Wilkening, 1981), and others remain as predictions for future psychological research, for example, an additive stage in distance inferences, an identity stage in time inferences, and a multiplicative stage in velocity inferences. The model also predicts continuous improvement in inferences within any given stage.

This was a case in which the networks used a single output unit with a linear activation function, because the idea was to estimate the amount of the unknown dimension, say, velocity. Five different levels of velocity, time, and distance were used. Input units coded the amounts on two of these dimensions, and the target value of the third, unknown dimension was coded on the output unit. No biases in the training patterns were required to capture these stages. Interestingly, static back-propagation networks could not capture this stage progression (Buckingham & Shultz, 1995). Either these static nets had insufficient computational power (in terms of number of hidden units) to reach the final multiplicative stage or too much power to capture the intermediate additive stage. It would appear that, at least in this case, the need to pass through periods of more limited representational power, as opposed to having immediate access to full representational power, may be critical when modeling children's cognitive development.

Effect Size

A linear output unit was also used in simulations of the prediction of effect sizes from the magnitudes of causal potencies and effect resistances (Shultz et al., 1995). Predicting the magnitude of a physical effect often requires integration of information on the potency of the cause and the resistance to the effect's occurrence. In some cases, potency and resistance are combined by subtraction and in other cases by division. A number of psychological regularities have been found on such tasks (Zelazo & Shultz, 1989). As they mature, children show an increase in the number of levels of potency and resistance used and a gradual convergence on the correct integration rule. They also acquire the subtraction rule before the division rule and temporarily overgeneralize subtraction to division problems. All of these phenomena were captured by cascade-correlation nets (Shultz et al., 1995). Problems were created by combining six levels of potency with six levels of resistance using either subtraction or division rules, as in the psychological research. Potency, resistance, and rule type were coded on the input units and effect size was coded on the output unit. The psychological regularities were captured with a variety of different input coding techniques. In addition, the networks generalized well when trained on a randomly selected two thirds of the problems and tested on the remaining one third.

Personal Pronouns

In addition to capturing these phenomena involving the child's understanding of physical events, cascade-correlation nets have been successfully applied to the acquisition of English personal pronouns. Many children acquire these pronouns without notable errors, whereas others show persistent reversal errors in which they refer to themselves as *you* and to others as *me*. The presence of such reversal errors is related to the lack of opportunity to overhear speech that is not addressed directly to the child.

All of these regularities were simulated with cascade-correlation nets (Shultz, Buckingham, & Oshima-Takane, 1994). The nets were trained to predict the correct pronoun as output given input information on the speaker, the addressee, and the referent. In effect, the nets succeeded in learning the correct semantic rules underlying pronoun use, namely that a first person pronoun refers to the person using it and a second person pronoun refers to the person who is addressed when it is used. The nets also showed sensitivity to the type of speech in the training patterns, such that acquisition could be relatively error-free in the case of a predominance of nonaddressed speech or characterized by persistent reversal errors in the case of a predominance of directly addressed speech. Errorless generalization was particularly evident when the network could overhear speech involving a number of other people, say an aunt and uncle in addition to the parents (Takane, Oshima-Takane, & Shultz, 1995).

These networks were trained in two phases, mimicking the way that pronoun interventions were done with young children, in the form of the so-called me–you game (Oshima-Takane, 1988). In the first phase, the network was exposed to speech uttered by parents. For example, the mother might address the child, point to herself, and say "me." Or, the mother might address the father, point to the father, and say "you." The former is an example of speech addressed directly to the child; the latter an example of speech overheard by the child. This first phase of training was typically biased in favor of different amounts of addressed or overheard speech. In the second phase, it was the network's turn to speak, in simulation of a child playing the me–you game. The question was how long the network would take to learn correct pronoun use when addressing other people, as a function of the amount of addressee or overheard speech it had experienced in Phase 1.

Why These Simulations Work

The basis for rule-like behavior in cascade-correlation nets, as found in all of these simulations, is the ability of these nets to extract statistical regularities from the learning environment. These include simple linear regularities as well as more complex nonlinear regularities. Simple linear regularities included the use of weight information on the balance scale; identity rules in the integration of velocity, time, and distance cues; the subtraction rule in predicting effect size; and mere imitation of pronoun use resulting in reversal errors. More complex nonlinear regularities that required the recruitment of hidden units included the torque rule on the balance scale; correct ratio rules in integrating velocity, time, and distance cues; and the correct semantic rules for personal pronouns.

Learning of rule-like behaviors in psychologically realistic stage sequences is a matter of both domain specific factors like environmental bias and task modularization, as well as domain general factors like a summative activation rule (see Equation 1) and the recruitment of hidden units. Following Flavell (1971), we consider stages to refer to periods of qualitatively distinct behaviors organized in a more or less invariant sequence (Shultz, 1991). Environmental bias favoring equal distance problems forced balance scale nets to focus on weight information to the temporary exclusion of distance information. Modular nets were required for generating seriation phenomena. Use of an activation rule that sums the inputs to units was important in producing early additive rules on the balance scale, velocity, time, and distance judgements, and effect size predictions. Recruitment of hidden units was important in eventually moving on to nonlinear rules, such as the torque rule on the balance scale; the correct ratio rules in integrating velocity, time, and distance cues; and the normative semantic rules for personal pronouns. Note that domain general features are those that would be common to all cascade-correlation models.

Perceptual effects reflect the continuous nature of network computations in tasks where quantitatively described items are being mapped to a qualitative comparison. In such cases, different sources of quantitative information must be compressed to reach a qualitative decision. Whenever the relevant quantitative inputs are large and clear, the qualitative decision is easier. This characterizes the torque difference effect on the balance scale and stick size differences in seriation.

An attraction of cascade-correlation is that it provides a means of modeling both qualitative and quantitative changes in processing mechanisms during development. The respective roles of these two modes of development is a critical question in contemporary developmental debates (Keil, 1990). Static connectionist models provide only quantitative change in the underlying processing mechanisms (McClelland, 1989), whereas most rule-based models provide only qualitative change in the underlying processing mechanisms (van Geert, 1991). Cascade-correlation provides a means of evaluating the need for either type of development and the interactions that may arise between them. Learning through the modification of the output weights corresponds to an underlying quantitative change in processing mechanisms. The network is integrating new information within an existing knowledge structure. Hidden unit recruitment corresponds to a qualitative change in processing mechanisms. The resulting network is computationally more powerful than it was before, capable of forming representations that were previously impossible. Note that qualitative changes in behavior, which are often used to mark stage transitions, can be due to either quantitative adjustment of weights (as in the effect size and seriation simulations) or qualitative increases in network power (as in the balance scale, velocity–time–distance, and pronoun simulations). The use of cascade-correlation networks permits assessment of which sort of underlying change is responsible for the surface changes in behavior.

We believe that it is this added developmental flexibility that has resulted in cascade-correlation models surpassing the performance of static network models. In the balance scale simulations, cascade-correlation did better than static back-propagation networks in two respects: reaching and staying in Stage 4, and not needing segregated weight and distance information in the layer of hidden units (an extra, programmer-designed constraint). In the velocity, time, and distance modeling, a wide range of back-propagation networks failed to produce the same stages successfully modeled with cascade-correlation. The back-propagation networks designed with too little power were unable to reach to final multiplicative stages, whereas those designed with too much power were unable to reach to final multiplicative stages, whereas those designed with too much power were unable to capture the intermediate additive stages. Only a progressive increase in power has been capable of showing the appropriate developmental course. A similar point in the realm of grammar learning was made by Elman (1993). In his case, recurrent back-propagation nets had to either receive progressively more complex sentences or grow in working memory capacity to learn an English-like grammar.

All of these interpretations of cascade-correlation network behaviors are more fully discussed elsewhere (Shultz et al., 1995).

CONCLUSION

In this article, we set out to illustrate the role of modeling in the study of cognitive development, demonstrate that cascade-correlation is an appropriate model of cognitive development, and to argue for the plausibility of constructivism.

Modeling was shown to play the role of a thought experiment with which to test and illustrate theories. Drescher (1991) tried to implement a constructivist mechanism for infant cognitive development, but the actual implementation failed to generate novel concepts. Its failure could be directly related to Fodor's (1980) argument against the possibility of constructivism. This model was a concrete example with which it became possible to pinpoint the potential shortcomings of a constructivist paradigm.

Cascade-correlation is a generative algorithm used for implementing constructivist style development. Here, the model was used to illustrate the effectiveness of Quartz's (1993) argument at overcoming Fodor's paradox. By demonstrating its applicability in a diverse range of cognitive developmental tasks, it is possible to explore the extent to which Quartz's claims constitute a general argument. We reported successful modeling results pertaining to the development of problem solving, reasoning about the physical world, and semantic development. The range of these successes strongly supports Quartz's claim that generative neural nets can escape Fodor's paradox by building more powerful representations.

We believe the models reported in this article provide strong support for the claim that computational models enrich the study of cognitive development. Indeed, the modeling work with cascade-correlation has led to a number of novel findings concerning the interaction of perceptual features and a child's performance on a cognitive task (e.g., effect of disorder on the seriation task, effect of torque-difference on the balance scale task). The return to further empirical studies may result in new data that will need to be incorporated in the model. Still different predictions may then arise. Thus, the modeling–experimentation dialectic process constitutes a series of ever better approximations.

Cascade-correlation not only accounts for behavioral data and perceptual effects, but it constitutes a working model of transition mechanisms in cognitive development. Because of the inherent difficulties in studying transitions in development, only a small fraction of developmental research is devoted to it (Sternberg, 1981). Our use of cascade-correlation is an attempt to redress that imbalance by providing a rigorous account of transition. A strong mechanistic reading of our models would suggest that there exist, in the child's cognitive architecture, direct analogies to the constituent elements of cascade-correlation. A weaker reading is one in which cascade-correlation is seen only as an illustrative example taken from the family of generative algorithms. The models can be interpreted on their own ground or used to illustrate other developmental theories such as Piaget's genetic epistemology expressed in terms of assimilation and accommodation.

Throughout this article we have supported a constructivist approach to development. We chose to highlight this issue because it was a clear example of the arguments we were presenting. Of course, the models we presented do not guarantee that constructivist development occurs in children. What they do show is that such development is possible and offer an avenue for escaping Fodor's paradox.

Although cascade-correlation reflects some of what we know about the brain (cf. Quartz & Sejnowski, in press), it is important to understand that the models are not meant as neural models of the corresponding high-level tasks. Rather, they are models of how constraints on cognitive resources can guide the developmental paths through which children evolve. Development is driven by the interactions of the child with the environment, but is constrained by the child's present representations and the mechanisms available to the child for developing novel and more powerful representations.

Finally, applying Valiant's framework to learning in both children and generative connectionist networks focuses attention not simply on what can be learned, but also on what cannot be learned. Different generative algorithms differ in the types and order of relationships that they can or cannot learn in feasible time. They all possess simple conceptual structures early in learning, but only a few will develop an architecture that can learn a specified complex conceptual structure. This suggests a means of comparing their relative adequacy as models of cognitive development. In terms of child development, it suggests a shift in research away from the current trend of searching for ever more precocious simplified conceptual representations and back to an examination of concepts that cannot be assimilated at various ages.

REFERENCES

Baillargeon, R. (1993). The object concept revisited: New directions in the investigation of infants' physical knowledge. In C.E. Granrud (Ed.), *Visual perception and cognition in infancy* (pp. 265–318). London: Erlbaum.

Bates, E., & Elman, J.L. (1993). Connectionism and the study of change. In M. Johnson (Ed.), *Brain development and cognition: A reader* (pp. 623–642). Oxford: Blackwell.

Baum, B.E. (1989). A proposal for more powerful learning algorithms. *Neural Computation, 1,* 201–207.

Baylor, G.W., Gascon, J., Lemoyne, G., & Pothier N. (1973). An information-processing model of some seriation tasks. *Canadian Psychologists, 14,* 167–196.

Beilin, H. (1994). Mechanisms in the explanation of developmental change. *Advances in Child Development, 25,* 327–352.

Bloom, P., & Wynn, K. (1994). The real problem with constructivism. *Behavioral and Brain Science, 17,* 707–708.

Boden, M.A. (1980). Artificial intelligence and Piagetian theory. In M. Boden (Ed.), *Minds and mechanisms: Philosophical psychology and computational models* (pp. 236–261). Ithaca, NY: Cornell University Press.

Boden, M.A. (1982). Is equilibration important? A view from artificial intelligence. *British Journal of Psychology, 73,* 65–173.

Boden, M.A. (1989). *Computer models of mind,* Cambridge, England: Cambridge University Press.

Buckingham, D., & Shultz, T.R. (1994). A connectionist model of the development of velocity, time, and distance concepts. *Proceedings of the sixteenth annual Conference of the Cognitive Science Society* (pp. 72–77). Hillsdale, NJ: Erlbaum.

Buckingham, D., & Shultz, T.R. (1995). *Computational power and realistic cognitive development* (Tech. Rep. No. 1995). Montréal: McGill University.

Case, R. (1985). *Intellectual development: Birth to adulthood.* New York: Academic.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems, 2,* 303–314.

Drescher, G.L. (1991). *Made-up minds. A constructivist approach to artificial intelligence.* Cambridge, MA: MIT Press.

Elkind, D. (1964). Discrimination, seriation, and numeration of size and dimensional differences in young children: Piaget replication study VI. *Journal of Genetic Psychology, 104,* 276–296.

Elman, J. (1993). Learning and development in neural networks: The importance of starting small. *Cognition, 48,* 71–99.

Elman, J.L., Bates, E.A., Karmiloff-Smith, A., Johnson, M.H., Parisi, D., & Plunkett, K. (1996). *Rethinking innateness: Connectionism in a developmental framework.* Cambridge, MA: MIT Press.

Fahlman, S.E. (1988). Faster-learning variations on back-propagation: An empirical study. In D.S. Touretzky, G.E. Hinton, & T.J. Sejnowski (Eds.). *Proceeding of the 1988 Connectionist Models Summer School* (pp. 38–51). Los Altos, CA: Morgan Kaufmann.

Fahlman, S.E., & Lebiere, C. (1990). The cascade-correlation learning architecture. In D.S. Touretzky (Eds.), *Advances in neural information processing systems 2* (pp. 524–532). Los Altos, CA: Morgan Kaufmann.

Ferretti, R.P., & Butterfield, E.C. (1986). Are children's rule assessment classifications invariant across instances of problem types? *Child Development, 57,* 1419–1428.

Fischer, K.W. (1980). A theory of cognitive development: The control and construction of hierarchies of skills. *Psychological Review, 87,* 477–531.

Fischer, K.W., & Bidell, T. (1991). Constraining nativist inferences about cognitive capacities. In S. Carey & R. Gelman (Eds.). *The epigenesis of the mind* (pp. 199–235). Hillsdale, NJ: Erlbaum.

Flavell, J.H. (1971). Stage-related properties of cognitive development. *Cognitive Psychology, 2,* 421–453.

Fodor, J. (1975). *The language of thought.* Cambridge, MA: Harvard University Press.

Fodor, J. (1980). Fixation of belief and concept acquisition. In M. Piatelli-Palmarini (Ed.), *Language and learning: The debate between Chomsky and Piaget* (pp. 143–149). Cambridge, MA: Harvard University Press.

Gelman, R. (1991). Epigenetic foundations of knowledge structures: Initial and transcendent constructions. In S. Carey & R. Gelman (Eds.), *The epigenesis of the mind* (pp. 293–322). Hillsdale, NJ: Erlbaum.

Grossberg, S. (1982). *Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control.* Dordrecht: Reidel.

Hertz, J., Krogh, A., & Palmer, R.G. (1991). *Introduction to the theory of neural computation.* Addison Wesley.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks, 2,* 359–366.

Karmiloff-Smith, A. (1992). *Beyond modularity.* Cambridge, MA: MIT Press.

Karmiloff-Smith, A. (1993). Self-organization and cognitive change. In M. Johnson (Ed.), *Brain development and cognition: A reader* (pp. 592–618). Oxford: Blackwell.

Keil, F.C. (1990). Constraints on constraints: Surveying the epigenetic landscape. *Cognitive Science, 14,* 135–168.

Kingma, J. (1984). The influence of task variation on seriation research: Adding irrelevant cues to the stimulus materials. *Journal of General Psychology, 144,* 241–253.

Klahr, D., & Wallace, J.G. (1976). *Cognitive development: An information processing view,* Hillsdale, NJ: Erlbaum.

Langley, P. (1987). A general theory of discrimination learning. In D. Klahr, P. Langley, & R. Neches (Eds.), *Production system models of learning and development* (pp. 99–161). Cambridge, MA: MIT Press.

Lewandowsky, S. (1993). The rewards and hazards of computer simulations. *Psychological Science, 4,* 236–243.

Mareschal, D., Plunkett, K., & Harris, P. (1995). Developing object permanence: A connectionist model. In J.D. Moore & J.E. Lehman (Eds.), *Proceedings of the seventeeth annual Conference of the Cognitive Science Society* (pp. 170–175). Hillsdale, NJ: Erlbaum.

Mareschal, D., & Shultz, T.R. (1993). A connectionist model of the development of seriation. *Proceedings of the fifteenth annual Conference of the Cognitive Science Society* (pp. 676–681). Hillsdale, NJ: Erlbaum.

Mareschal, D., & Shultz, T.R. (1996). *Development of children's seriation: A connectionist approach.* Manuscript submitted for publication.

McClelland, J.L. (1989). Parallel distributed processing: Implications for cognition and development. In R.G.M. Morris (Ed.), *Parallel distributed processing: Implications for psychology and neurobiology* (pp. 8–45). Oxford: Oxford University Press.

McClelland, J.L. (1995). A connectionist perspective on knowledge and development. In T.J. Simon & G.S. Halford (Eds.), *Developing cognitive competence: New approaches to process modeling* (pp. 157–204). Hillsdale, NJ: Erlbaum.

Mehler, J., & Dupoux, E. (1994). *What infants know. The new cognitive science of early development.* Oxford: Blackwell.

Molenaar, P.C.M. (1986). On the impossibility of acquiring more powerful structures: A neglected alternative. *Human Development, 29,* 245–251.

Newell, A. (1990). *Unified theories of cognition.* Cambridge, MA: Harvard University Press.

Oshima-Takane, Y. (1988). Children learn from speech not addressed to them: The case of personal pronouns. *Journal of Child Language, 15,* 95–108.

Papert, S. (1963). Intelligence chez l'enfant et chez le robot [Intelligence in the child and the robot] In L. Apostel, J. Grize, S. Papert, & J. Piaget. La filiation des structures. *Etudes D'Epistemologie Génétiques, 15,* 131–194.

Papert, S. (1980). The role of artificial intelligence in psychology. In M. Piatelli-Palmarini (Ed.), *Language and learning: The debate between Chomsky and Piaget* (pp. 89–106). Cambridge, MA: Harvard University Press.

Piaget, J. (1965). *The child's concept of number.* New York: Norton Library.

Piaget, J. (1977). *The development of thought: Equilibration of cognitive structures.* Oxford: Blackwell.

Piatelli-Palmarini, M. (Ed.). (1980). *Language and learning: The debate between Chomsky and Piaget.* Cambridge, MA: Harvard University Press.

Plunkett, K., Kamiloff-Smith, A., Bates, E., Elman, J.L., & Johnson, M.H. (in press). Connectionism and developmental theory. *Annual Review of the Journal of Child Psychology and Psychiatry.*

Plunkett, K., & Sinha, C. (1992). Connectionism and developmental theory. *British Journal of Developmental Psychology, 10,* 209–254.

Plunkett, K., Sinha, C., Moller, M.F., & Strandsby, O. (1992). Symbol grounding or the emergence of symbols? Vocabulary growth in children and a connectionist net. *Connection Science, 4,* 293–312.

Quartz, S.R. (1993). Neural networks, nativism, and the plausibility of constructivism. *Cognition, 48,* 223–242.

Quartz, S.R., & Sejnowski, T.J. (in press). The neural basis of cognitive development: A constructivist manifesto. *Behavioral and Brain Sciences.*

Rumelhart, D.E., & McClelland, J.L. (1986). *Parallel distributed processing: Exploration in the microstructure of cognition: Vol. 1. Foundations.* Cambridge, MA: MIT Press.

Russell, B. (1961). *History of western philosophy.* London: Routledge.

Rutkowska, J.C. (1987). Computational models and developmental psychology. In J. Rutkowska & C. Cook (Eds.), *Computers, cognition, and development.* London: Wiley.

Rutkowska, J.C. (1993). *The computational infant: Looking for developmental cognitive science.* London: Harvester Wheatsheaf.

Schmidt, W.C., & Ling, C.X. (1996). A decision-tree model of balance scale development. *Machine Learning,* 1–30.

Schmidt, W.C., & Schultz, T.R. (1991). *A replication and extension of McClelland's balance scale model* (Tech. Rep. 91–10–18). Montréal: McGill University.

Scruton, R. (1982). *Kant.* Oxford: Oxford University Press.

Shultz, T.R. (1991). Simulating stages of human cognitive development with connectionist models. In L. Birnbaum & G. Collins (Eds.), *Machine learning: Proceedings of the eighth International Workshop* (pp. 105–109). San Mateo, CA: Morgan Kaufmann.

Shultz, T.R., Buckingham, D., & Oshima-Takane, Y. (1994). A connectionist model of the learning of personal pronouns in English. In S.J. Hanson, T. Petsche, M. Kearns, & R.L. Rivest (Eds.), *Computational learning theory and natural learning systems: Vol. 2. Intersection between theory and experiment* (pp. 347–362). Cambridge, MA: MIT Press.

Shultz, T.R., Mareschal, D., & Schmidt, W.C. (1994). Modeling cognitive development on balance scale phenomena. *Machine Learning, 16,* 57–86.

Shultz, T.R., & Schmidt, W.C. (1991). A cascade-correlation model of balance scale phenomena. *Proceedings of the thirteenth annual Conference of the Cognitive Science Society* (pp. 635–640). Hillsdale, NJ: Erlbaum.

Shultz, T.R., Schmidt, W.C., Buckingham, D., & Mareschal, D. (1995). Modeling cognitive development with a generative connectionist algorithm. In T.J. Simon & G.S. Halford (Eds.), *Developing cognitive competence: New approaches to process modeling* (pp. 205–261). Hillsdale, NJ: Erlbaum.

Siegler, R.S. (1976). Three aspects of cognitive development. *Cognitive Psychology, 8,* 481–520.

Siegler, R.S. (1981). Developmental sequences between and within concepts. *Monographs of the Society for Research in Child Development, 46* (Whole No. 189).

Siegler, R.S. (1986). *Children's thinking.* Englewood Cliffs, NJ: Prentice-Hall.

Simon, T.J., & Halford, G.S. (Eds.). (1995). *Developing cognitive competence: New approaches to process modeling.* Hillsdale, NJ: Erlbaum.

Spelke, E. (1994) Initial knowledge: Six suggestions. *Cognition, 50,* 431–445.

Sternberg, R.S. (Ed.). (1981). *Mechanisms of cognitive development.* New York: Freeman.

Takane, Y., Oshima-Takane, Y., & Shultz, T.R. (1995). Network analyses: The case of first and second person pronouns. *Proceedings of the 1995 IEEE International Conference on Systems, Man and Cybernetics* (pp. 3594–3599).

Valiant, L.G. (1984a). Deductive learning. *Philosophical Transactions of the Royal Society of London, series A. 312,* 441–446.

Valiant, L.G. (1984b). A theory of the learnable. *Communications of the ACM, 27,* 1134–1142.

van Geert, P. (1991). A dynamic systems model of cognitive and language growth. *Psychological Review, 98,* 3–53.

Wilkening, F. (1981). Integrating velocity, time, and distance information: A developmental study. *Cognitive Psychology, 13,* 231–247.

Young, R. (1976). *Seriation by children: An artificial intelligence analysis of a Piagetian task.* Basel: Birkhauser.

Zelazo, P.D., & Shultz, T.R. (1989). Concepts of potency and resistance in causal prediction. *Child Developmental, 60,* 1307–1315.