ECOLOGICAL PSYCHOLOGY, *14*(1–2), 21–51

The Value of Symbolic Computation

Whitney Tabor
Department of Psychology
University of Connecticut

Standard generative linguistic theory, which uses discrete symbolic models of cognition, has some strengths and weaknesses. It is strong on providing a network of outposts that make scientific travel in the jungles of natural language feasible. It is weak in that it currently depends on the elaborate and unformalized use of intuition to develop critical supporting assumptions about each data point. In this regard, it is not in a position to characterize natural language systems in the lawful terms that ecological psychologists strive for. Connectionist learning models offer some help: They define lawful relations between linguistic environments and language systems. But our understanding of them is currently weak, especially when it comes to natural language syntax. Fortunately, symbolic linguistic analysis can help connectionism if the two meet via dynamical systems theory. I discuss a case in point: Insights from linguistic explorations of natural language syntax appear to have identified information structures that are particularly relevant to understanding ecologically appealing but analytically mysterious connectionist learning models.

This article is concerned with the relation between discrete, *symbolic systems* of the sort that have been widely used in linguists' formal analyses of natural languages and *continuous dynamical systems* that many ecological psychologists have found insightful, especially for understanding limb movement and visual perception. It begins by casting the discrete, symbolic linguistic models as unecological in several respects (see also Carello, Turvey, Kugler, & Shaw, 1984). Connectionist (or artificial neural-network) models, a particular type of dynamical system, are offered as a more ecological alternative. But despite their strengths, these connectionist models suffer from a certain opacity, which makes it difficult to understand what they are doing and how to improve their performance. A helpful way of overcoming this opacity is to explore their capabilities using discrete, symbolic models as reference points. The symbolic models are identical in behavior to certain special cases of the

Requests for reprints should be sent to Whitney Tabor, Department of Psychology, U-0020, University of Connecticut, Storrs, CT 06269–1020. E-mail: tabor@uconnvm.uconn.edu

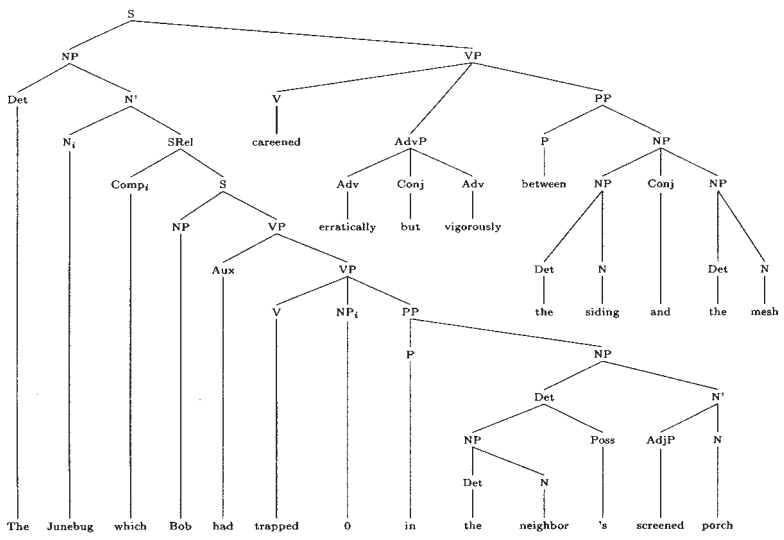
dynamical models and these cases are useful to know about because they are relatively easy to understand. The result points to a general correspondence between regimes of symbolic and dynamical systems and suggests that to understand particularly complex dynamical processes, symbolic insights may be helpful.

By *discrete symbolic computation,* I mean something very like Carello et al.'s (1984) use of the term *discrete-mode computation.* This kind of computation goes hand in hand with the "representationalist" approach to cognition, which Gibson (1979/1986) so soundly rejects, and discrete symbolic theories of psychology often take out "a loan of intelligence" (Dennett, 1978, p. 12) of the sort that many ecological psychologists quite reasonably deplore. The aim of the present article is not to suggest that discrete symbolic computation is accurate or complete as a foundation for psychology, but rather that it provides insights into the structuring of information, and these insights may turn out to be helpful to ecological psychology. Synergy between the perspectives might thus be worth seeking.

I begin with some examples. The formal linguistic theory called *generative linguistics,* working in a discrete and symbolic mode, maps a sentence such as (1) to a representation along the lines of (2).

(1) The June bug, which Bob had trapped in the neighbor's screened porch, careened erratically but vigorously between the siding and the mesh.

(2)



Under the framework proposed by Frege (1892/1952) and laid out by Montague (1970/1974), this representation supports an interpretation along the following lines: The generic meaning of *the* and the generic meaning of *June bug* combine to

form the meaning of *the June bug;* that meaning combines with the meaning of *trapped* in such a way as to convey the notion of the June bug being trapped, rather than doing the trapping, for example; the generic meanings of *the* and *neighbor* also combine to define a meaning for *the neighbor,* and so on. It thus fits into a rather broad coverage and effective theory (*compositional semantics*) of how the literal meanings of sentences can be derived from the generic meanings of their words.

ECOLOGICAL CRITIQUE

Despite its power, there are a number of drawbacks to Analysis 2 if it is taken as a portrayal of the mentation associated with an actual instance of comprehending (or producing) a natural language sentence. Many of these drawbacks can be related to concerns that ecological psychologists have expressed about representationalist approaches to cognition in general. I will discuss three: staticness, context-freeness, and lack of emphasis on lawfulness.

First, *staticness:* Time is not a variable in a phrase-structure diagram. Thus, the phrase-tree viewpoint seems immediately at odds with the ecological emphasis on action: "The ecological approach asserts that the concept of information cannot be developed systematically apart from considerations of activity" (Turvey & Carello, 1981, p. 316). I take the relevant sense of "activity" here to be the activity of interacting with (e.g., comprehending, producing) language as it unfolds over time. It is true that a phrase-structure analysis often provides a map of its utterance across time (usually left-to-right in the diagram corresponds to history-to-future in time). But because such a map is not explicit about what mental states are occurring as the utterance develops through time, theories of parsing have been proposed—that is to say, theories of how the tree structure and its meaning get built over time when an utterance is interpreted or formulated. There are a multitude of possible temporal programs for building tree structures. Much research is dedicated to measuring human interaction with sentences over time in an effort to figure out which one of these programs is correct. But if the temporal information were not factored out of the encoding process in the first place—that is, if the theory of encoding were required to produce something that not only embodied structures but also built them as the input unfolded temporally—then it might well be possible to derive predictions about the time course of processing directly from the mechanism that establishes the encoding, bypassing this nettlesome problem.

A second concern is that the phrase-structure approach gains a great deal of its efficiency from its assumption that the building blocks of tree diagrams (e.g., rules such as **NP → Det N**) are *context-free abstractions over language content.* Ecological psychologists have long objected to context-free mental objects. Indeed, they are cumbersome when the focus is on the ongoing mutual influence between organism and environment. The standardly cited linguistic cases involve *deixis*—linguistic elements whose function is to refer context dependently to entities in the world

(e.g., *I, tonight;* Turvey & Carello, 1981). Abstract phrase-structure objects, such as **NP → Det N,** might seem, at first glance, to be less susceptible to criticism on this account, because they capture patterns that are remarkably stable across instances of the use of a language, and in fact, they provide a helpfully sturdy skeletal structure within which the flexibility of deictic elements can be well modeled (van Eijck & Kamp, 1996). Nevertheless, there are reasons to be skeptical. There are many informational regularities that cut across the independent phrasal units (Charniak, 1993).

For example, in (3),

(3) The chicken which Fred baked was not ready to eat.

the chicken is interpreted as the patient of eating (the one that gets eaten) when the verb *eat* is read. But in (4),

(4) The chicken which Fred fed was not ready to eat.

the chicken is more likely to be interpreted as the agent of eating. The difference between the two chickens is determined by the content of the embedded relative clause, which is phrase-structurally quite remote from the verb *eat.* Somehow, the information has to be transported across the tree so that the parser can select the right role assignments at *eat.* A useful strategy adopted by several linguistic theorists (e.g., Bresnan, 1982; Joshi & Schabes, 1996; Pollard & Sag, 1994) is to employ features that "percolate" from daughter nodes to mother nodes or vice versa and thus carry information to places the context-free rules do not get it to (other theories employ syntactic transformations with similar effect, e.g., Chomsky, 1981). It is tempting to adopt such a strategy in this case by, for example, letting the verb *feed* generate a feature [+alive] specifying that its patient (the thing fed) ought to be alive, by letting [+alive] percolate from its starting point in the relative clause up to *chicken,* by simultaneously letting the verb *eat* specify that its agent should be [+alive] while its patient should be [–alive], by letting the feature from *chicken* percolate down through the main clause to *eat,* and by letting the parser examine the two different ways of linking the subject of *be* with an argument of *eat* in order to choose the one for which the features are consistent. Workable as this approach might be for this case, its plausibility is cast into doubt by (5),

(5) The chicken which Fred last fed just yesterday is now ready to eat.

which has exactly the same verbs and nouns in the same phrase-structural relations to one another but seems to be biased toward the cooked chicken interpretation. The temporal adverbs (*last, just yesterday,* and *now*) are making the difference but it is not clear how that difference could be expressed via modulation of the percolating features. Taking an inspiration from Gibson (1979/1986), one might sug-

gest that perceivers of language do not perceive meaning via context-free syntactic abstractions, but rather they directly perceive meaning. See Tanenhaus, Carlson, and Trueswell (1989) for an experimental development that points toward a similar conclusion.

Third, the program of inquiry underlying analysis (2) only weakly supports a lawful treatment of the language–environment system. By a lawful treatment, I mean here a complete and coherent characterization of how the proposed mental state and the proposed environment coevolve at the timescale of moment-to-moment experience. This definition of lawfulness sounds like it could, in principle, accommodate different types of lawfulness from the "specificational" sense that ecological psychologists generally focus on (Turvey & Carello, 1985). For the case at hand, I do not think it does. In the body of this article I use the term *lawfulness* as I have just defined it. In the conclusion, I return to the question of how my use of the term lawfulness is related in its use among Gibsonian psychologists.

For linguistic lawfulness, two levels of completeness need considering. I refer to the first, less complete level as *processing lawfulness* and to the second, more-complete level as *inductive lawfulness*. Processing lawfulness characterizes the relation between a language utterance unfolding through time and the associated mental trajectory of its generator–perceiver. Inductive lawfulness characterizes the relation between an entire linguistic environment (consisting, for example, of all the linguistic experience a person has during childhood) and the associated processing map from utterances in contexts to mental trajectories. Generative linguistic theory has not rejected lawfulness in either of these senses. But there is an issue of how much commitment the methodology of the approach has to seeing the lawfulness through.

The problem is not so much with processing lawfulness. If the individual words can be identified and assigned meanings, then the standard models based on diagrams such as (1) seem to be within range of providing a lawful account of how linguistic utterances specify mental states (up to symbolic ambiguity) and vice versa (Kamp & Reyle, 1993; Montague, 1970/1974; van Eijck & Kamp, 1996). But generative linguistics has shied away from studying inductive lawfulness from an early day when Chomsky (1957) argued that "discovery procedures" (which build grammars from scratch, based on experience) are hard to devise, but "evaluation procedures" (which select among a finite set of clearly distinct options that evolution has made conveniently available) are easier. The idea seems to have been that if enough of the territory of linguistic structure could be mapped out in accurate detail, then the problem of induction could be reduced to making a finite set of binary, or *n*-ary, choices based on the observation of easily recognized "triggers." Indeed a number of proposals along these lines have been put forth (Gibson & Wexler, 1994; Hyams, 1986; Lasnik, 1990). But unifying laws are lacking, and most researchers in the field pay little attention to the predictions of these inductive models, relying instead primarily on intuition to establish most of the structural background (in particular, the tree diagrams) on which models of specific utterances are built.

I suggest that in the case of language, it is crucial to have a full-fledged account of both processing lawfulness and inductive lawfulness. Here, language contrasts, at least in degree, with the domains that are typically studied by ecological psychologists, for example, visually guided locomotion, dynamic touch, wielding, and so forth. Ecological psychologists do not usually object, for example, to attempts to characterize organism-relevant invariants of optic flow without first building a theory of how an organism learns to detect those invariants (if, indeed, learning is required at all in that case). But there is a fairly strong sense in which we understand and can accurately model the nature of the materials (e.g., light, matter) involved in situations where optic flow is important. The materials involved in "syntactic flow" (ostensibly, words; lexical classes, such as noun, verb, adjective, etc.; and phrasal classes, such as noun phrase, verb phrase, sentence, etc.) are more mysterious. In particular, their definitions are highly mutually dependent. Generative theory assumes, for example, that an uttered word *w* counts as belonging to lexical class C and as forming part of an instance of phrase class P, because assuming the existence of C and P, along with the role that *w* plays in the instance at hand, provides optimal generalization of the theory about the set of potential utterances that a native speaker commands. It is with the definition of "optimal generalization" that inductive models are concerned. Because of the considerable interdependence of structure definitions under this notion, a theory of induction is essential to a lawful treatment of language use.

CONNECTIONIST INSIGHT

Connectionist (or artificial neural-network) models (Haykin, 1994; Hertz, Krogh, Palmer, 1991; Rumelhart, McClelland, & PDP Research Group, 1986) of language offer an alternative to the generative linguistics approach that helps address several of these ecological concerns. The next subsection gives a brief introduction to the class of connectionist language models considered here. The following subsection examines the connectionist approach in light of the critiques.

Connectionist Models

Connectionist models are mathematical or computational models of organism behavior, which take their inspiration from neurobiology (O'Reilly & Munakata, 2000). In particular, they consist of networks of nodes and connections that resemble, in a very pared-down way, networks of neurons and axons, synapses, and dendrites. Each node *i* is associated with a number, a_i, called its *activation* (analogous to neural firing rate in real brain tissue); each connection between nodes is unidirectional and is associated with a number, w_{ij}, called its weight (analogous to synaptic strength). By convention, I use the term w_{ij} to denote the weight on the connection from node *j* to node *i*.

Networks undergo two types of change: *activation change,* which happens quickly, is intended as a model of what psychologists typically call "behavior"; *weight change,* which happens slowly, is intended as a model of "learning." Activation change is specified by equation (1), where t indexes time.

$$\begin{aligned} a_i(t) &= f(net_i(t)) \\ net_i(t) &= \sum_j w_{ij} a_j(t-1) \end{aligned} \tag{1}$$

In the present work, the activation function f is a smoothed step function or *sigmoid,* usually $f(x) = \frac{1}{1+e^{-x}}$. Thus, each node computes a weighted sum of the input from nodes that feed into it and becomes strongly activated if the sum is positive and weakly activated if the sum is negative.

The model is immersed in an environment. Weight change is the process by which the network attunes itself to the regularities (invariances) in its environment. In the present case, following the paradigm developed by Elman (1990, 1991, 1995), the model iteratively senses an event corresponding to the perception of a word in a stream of speech and tries to predict the next event, which is assumed to be the perception of the following word. To implement this idea, it is convenient to define one set of nodes within the network as "input" (for sensing the current event) and another as "output" (for predicting the next event), respectively; see Figure 1.[1] To make no prior assumptions about the structure of the data, an unbiased input–output encoding is used: Each word form corresponds to one unit on the input layer and one unit on the output layer. Words from a large sample of language are presented to the model in sequence. At the point of each word presentation, the unit corresponding to the current word is activated on the input layer (all other input and output units have activation 0). Observation of the following word in the language sample provides the basis for adjusting the weights of the model slightly so as to improve its ability to predict the future of its environment. Effective weight change is accomplished by referring to a cost function that computes the discrepancy between the activation pattern the network produces on its output layer and the word that actually occurs next at each point in time. By choosing, as the measure of discrepancy, the Kullback–Leibler divergence, D, between output activation and observed event ($D = \sum_{i \in Outputs} t_i \log(t_i / a_i)$) where t_i is 1 if word j occurred and 0 otherwise), one arrives at

[1] To be able to interpret the outputs as probabilities, I have used the normalized exponential sigmoid for the output units as a group: $a_i = \frac{e^{net_i}}{\sum_{j \in Outputs} e^{net_j}}$.

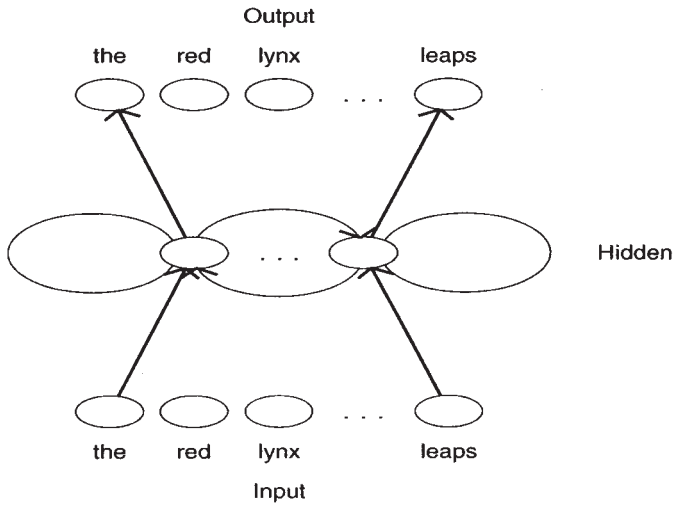


FIGURE 1 The simple recurrent network (Elman, 1990).

a particularly simple and intuitive formula for reducing total cost on the basis of the network's experiences with each event: The change in the weight on the connection from unit j to unit i, Δw_{ij}, is given by $\Delta w_{ij} = (t_i - a_i)a_j$ (Rumelhart, Durbin, Golden, & Chauvin, 1995). This formula is called the *delta rule.* It says, essentially, to change those weights most that come from active nodes (for they are the ones that are creating the current pattern), and change them in the direction that makes the network more strongly expect what just happened in this context to happen again the next time this context is encountered.

As Elman (1990, 1991, 1995), Christiansen (1994), Christiansen and Chater (1999), Rohde and Plaut (1999), Tabor (1994), Tabor, Juliano, and Tanenhaus (1997) have shown, this kind of network can do a reasonable job at learning syntactic and semantic structure from a simple corpus of words approximating patterns that occur in English. When it is exposed to a several-hundred-thousand word corpus of sentences based on a simple vocabulary of 30 lexemes or so, it learns to distribute activation over its output units in probability distributions that fairly accurately characterize the semantic and syntactic constraints inherent in the corpus. After an initial *the,* for example, the trained network distributes activations mostly over adjectives and nouns; if it then gets an adjective such as *happy,* it distributes activation over compatible second adjectives and compatible nouns (e.g., those that refer to sentient creatures). In this sense, the network learns a time-sensitive encoding of the syntactic and semantic structure of the corpus.

Ecological character of connectionism. In what ways does the connectionist model address the previously outlined ecological critique of representationalist treatments?

Regarding the staticness of the generativist representations, the network model brings at least some improvement. At the level of the objects with which it is designed to deal (sentences), the network interacts with those objects in a temporal sequence that is similar to the sequence in which people interact with them in life: one word at a time, starting from the first word, going toward the last word. By embracing this particular dynamical aspect of speech, the network encodes syntactic invariances enmeshed with those "processing" invariants associated with temporal sequencing. Indeed, Christiansen and Chater (1999) have shown that, in empirically prominent ways, the behavior of the network model diverges from the predictions of the simplest generative parsing theory in ways quite similar to the way humans diverge (e.g., both networks and humans struggle with center-embedded structures such as "the dog the cat the rat chased bit died"). Generative models of parsing generally posit an additional mechanism ("memory load"; e.g., E. Gibson, 1998) to account for these facts. The network account, by being more direct, avoids this disjunction.

Although no network of this sort has yet been successfully trained to pick up on facts as subtle as the chicken–dinner vagaries illustrated previously, there is a significant sense in which the network approach opens the door to a treatment of the highly context-sensitive nature of natural language interpretation. Unlike the representationalist account, which begins with context-freeness and tries to back away from it as the data warrant, the network account begins with openness to arbitrary context sensitivity and then tries to mold its attention (though not categorically) to focus just on the most relevant facts in each context. For this reason, learning network models essentially never assign the same encoding to two different perceptions. Their internal codes are real-valued and sensitive to subtle properties of their environment, so the chance of two codes being identical is small (unless "design governs"—see reference to this later in this article). The network models are thus more naturally context sensitive.

Finally, as previously suggested, because it learns an encoding in the service of a simple functional task (predicting the future of its environment), the network introduces a degree of lawfulness that surpasses the lawfulness achieved by the generative program. In network terminology, both activation change and weight change are lawfully related to the properties of the network's environment. These correspond, respectively, to processing lawfulness and inductive lawfulness. Regarding the latter, it is reasonable to say that the network builds a "grammar" by doing the best it can to fit some rather flexible materials in its possession (its "hidden unit manifold") to the structure of the environmental invariants. How, one might ask, is this different from the post hoc inductive models, mentioned previously, which have been constructed on the basis of linguistic structural discoveries? If the discrete parameters of the generative models can have arbitrary form and be great in number, then the difference between symbolic, discrete-parameter models and dynamic, continuous-parameter models may be very subtle and hard to establish empirically. What is different is that the network model effectively derives the parameter list from a single, simple principle—the *learning rule.* Also, by addressing

induction in quantitative terms, the connectionist approach necessitates the development of specific distributional characterizations of the training data, so its analytical emphasis is much more evenly distributed between organism and environment than in the more mindcentric generative approach.[2] It can be said, then, that the network model improves on lawfulness by providing a simple principle for lawful learning, as well as for lawful behavior.

A challenge for the connectionist approach. The foregoing discussion suggests that connectionist language models address several of the ecological complaints about representationalist models of language. But there is, it must be noted, a substantial practical challenge for the connectionist approach. Connectionist models labor hard to learn the kinds of complex temporal dependencies illustrated in Example (1). Their temporal realism introduces a bias that makes the signals coming from short temporal dependencies louder than the signals coming from longer ones; moreover, their generalism with regard to possible dependencies implies that, with even a few tens of vocabulary items and a couple of intervening words, the correlational signals they are trying to detect in the longer, phrasal-dependency cases are shrouded in a haze of noise created by irrelevant potential dependencies, and this haze makes learning very difficult (Servan-Schreiber, Cleeremans, & McClelland, 1991). In other words, the very properties that make the models desirable on ecological grounds seem to fetter them when it comes to handling the complexities of natural language syntax.

Addressing the challenge. What can be done? In this section, I outline a method (described in detail in Tabor, 2000) for encoding languagelike complex dependencies in connectionist networks. The method, called *dynamic automata* or *fractal grammars,* skips over the problem of learning and focuses directly on encoding. Of course, this means it fails to fully meet the lawfulness desideratum identified previously because it does not address induction. It also has a discrete symbolic bias—it is a method of creating idealized, context-free structures whose sentence-parsing abilities are exactly equivalent to the idealized symbol-manipulating grammars that form the backbone of generative linguistics theories. This is useful, I suggest, because it helps establish a set of bearing points in the wide sea of nonlinear systems, which the connectionist networks are capable of embodying (Moore,

[2]It is true that I and many connectionist modelers are quite happy to take it as a goal of research to model "the mentation associated with an actual instance of comprehending." This perspective might seem to be at odds with that of J. Gibson (1979/1986) and many of his successors, who do not seek laws characterizing sequences of mental events. I think this apparent contrast is false. Gibsonians may not explicitly model mental events, but when they seek laws describing, for example, the affordances provided to a deer as opposed to a chipmunk by a downed tree, they are, in effect, characterizing animal-specific mental processes. The difference between the classical symbolic approach on the one hand, and the ecological and connectionist approaches on the other, lies not in whether mental events are accepted or rejected but in the qualities that are ascribed to them.

1998). In a later section (Simulations), I bring the learning algorithm back in and reexamine it with the help of these bearing points. Encouragingly, the learning networks appear to converge on the types of encodings predicted by the dynamical automaton models. Moreover, there is a revealing alignment between symbolic and dynamical computation hierarchies.

Dynamical Automata

A major feature of the syntactic structure of natural language is its nested structure: There are many cases where a phrase occurs inside another phrase, including ones in which the embedded phrase is of the same type as the dominating phrase recall (1). To think clearly about nested sequence structures, it is helpful to design a simple example. Consider a language, called *Language 0,* which has a prototype sentence in it consisting of the words, *a, b, c,* and *d* in sequence. Suppose that after any word of Language 0, it is possible to insert an embedded instance of this prototype sentence. Thus a typical sentence would be something such as *a b a b c d c a a b c d b c d d*. Language 0 can be described by the context-free grammar shown in Table 1.

Table 1 describes a symbolic system for generating Language 0. A symbolic system computes by putting symbols in memory registers and following symbolic rules for manipulating the symbols. By contrast, a connectionist network computes by adjusting the real-valued activations of its nodes. The relation between the form of the symbolic symbols and the information they point to is arbitrary. By contrast, a connectionist network computes in a metric space, where distances between states are defined, and nearby states predict similar behaviors; thus, the form of the network's encoding tends to bear a predictable relation to its content.

How might a connectionist, metric-space computer be designed so it could easily produce all and only the strings of Language 0? To keep track of the temporal dependencies in Language 0, it is necessary to keep track of each point at which an *a b c d* sequence was started but not finished. A symbolic machine uses a *stack* for this purpose—a list of the incomplete embeddings that need to be completed. Suppose this list uses the stack symbol A for an embedding under *a;*

TABLE 1
Grammar 0, Which Defines Language 0

S → A B C D	A → a S	B → b S	C → c S	D → d S
	A → a	B → b	C → c	D → d

Note. To generate a sentence, the grammar-interpreter starts with a rule of the form "S → …" and stores, in a special staging area of its memory, the sequence of symbols to the right of the →. It then attempts to replace each of these symbols following the rules of the grammar under the assumption that the symbol "→" means "can be replaced by." The process terminates whenever it reaches a point where no more replacements can be made. The sequence that lies in the staging area at that point is the generated sentence.

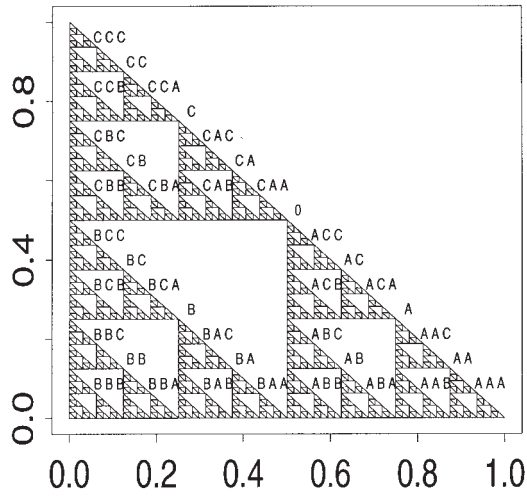


FIGURE 2 A useful way of mapping stack states for Language 0 to neural activation states.

B for an embedding under *b;* and so forth. Then, for Language 0, the set of all possible stack states that can occur in the language is the set of all strings composed of 0 or more *A*s, *B*s, and *C*s (a *D* stack symbol is not needed because *d* always ends a phrase). This set is called {*A, B, C*}*. To keep track of stack states, a metric-space computer needs to map each member of {*A, B, C*}* to a unique point in the metric space.

Figure 2 shows one scheme for doing this in a two-unit neural network. The two axes identify activation values of the two units. The points associated with stack states are points in a fractal set called the *Sierpinski triangle*—note that each stack state is at the midpoint of the hypotenuse of a triangle that is isomorphic to the whole set. Of course, there are (uncountably infinitely) many ways to map the members of {*A, B, C*}* to points in a connected metric space. The proposed way translates naturally into a connectionist encoding because (a) the infinite set of stack states lies in a bounded region—this helps because unit activations are bounded; (b) states that are prominently different in a sensory sense (the state of expecting *b* vs. the state of expecting *c*) are separable from one another by straight-line boundaries or *linearly separable* (Minsky & Papert, 1969)—this helps because sigmoidal activation functions approximate linear separators; and (c) the larger scale shape of the trajectory associated with a particular phrase stays constant across levels of embedding (e.g., *a b c* is always "lower right to lower left to upper left" in Figure 2)—this helps because the structure of the separators can be used to make appropriate distinctions at all levels of embedding, provided the scaling is appropriately normalized. Table 2 shows how to cash these benefits in a connectionist encoding. The essence of the network is a two-element vector, $\vec{z}$, corresponding to a position on the Sierpinski triangle (Barnsley, 1993). When $\vec{z}$ is in the subset of the plane specified in the Compartment column, the possible input

TABLE 2
Dynamical Automaton 0 (DA 0)

Compartment	*Input*	*State Change*
$z_1 > ½$ and $z_2 < ½$	b	$\vec{z} \leftarrow \vec{z} - (½, 0)$
$z_1 < ½$ and $z_2 < ½$	c	$\vec{z} \leftarrow \vec{z} + (0, ½)$
$z_1 < ½$ and $z_2 > ½$	d	$\vec{z} \leftarrow 2(\vec{z} - (0, ½))$
Any	a	$\vec{z} \leftarrow ½\vec{z} + (½, 0)$

words are those shown in the Input column. Given a compartment and a legal input for that compartment, the change in $\vec{z}$ that results from reading the input is shown in the State Change column. If we specify that the network must start with $\vec{z} = (½,½)$, make state changes according to the rules in Table 2 as symbols are read from an input string, and return to $\vec{z} = (½,½)$ (the Final Region) when the last symbol is read, then the computer functions as a recognizer for the language of Grammar 0—that is, the rules will bring it back to the starting point for all and only the sentences of Language 0. To see this intuitively, note that any subsequence of the form *a b c d* invokes the identity map on $\vec{z}$. Thus Dynamical Automaton 0 (DA 0) is equivalent to the nested finite-state machine version of Grammar 0. I refer to Table 2 as a "connectionist encoding" because the formulas translate directly into an artificial neural implementation using standard connectionist devices (Tabor, 2000). Tabor also shows that the method illustrated in this example is sufficiently general that it can handle all nested phrase-structure dependencies.

Lyapunov Analysis

Having used representational conceptions to design dynamical systems (neural networks) for processing phrase-structure languages, I wanted to see whether self-organizing (learning) neural networks were, in fact, creating similar encodings when faced with unbounded nesting languages. But because the networks need to work with a few more than two dimensions (otherwise learning success is a long shot), it is not easy to compare dynamical automata to trained neural-symbol processors simply by "looking at" their hidden unit encodings. A tool is needed.

Lyapunov characteristic exponents (Abarbanel, 1996; Oseledec, 1968) are useful for characterizing the complexity of dynamical processes (processes such as walking, seeing, haptically exploring, or, in this case, understanding and producing language). Many dynamical systems have attractors, or states, which the system tends toward over time. Attractors can be single, static-system states (such as the hanging-straight-down state of a pendulum). They can also be dynamic—for example, an easily sustained rhythmic gait in a walking organism. I suggest that familiar phrasal sequences (e.g., determiner-adjective-noun, noun_phrase-verb-noun_phrase, etc.) are associated with trajectories on a dynamic attractor that underlies language processing. There are two main classes of dynamic attractors: *attractive limit cycles*—bounded dynamic sequences that repeat—and *chaotic attractors*—these are bound-

ed like limit cycles but move around so wildly (and unpredictably) that they never repeat. Lyapunov exponents measure the average rate of contraction (or, equivalently, divergence) of system states near the attractor of a dynamical system. In deterministic dynamical systems, Lyapunov exponents can be used to classify an attractor as repeating or chaotic: The maximal Lyapunov exponent on a limit cycle is negative; the maximal exponent on a chaotic attractor is positive. Thus, by measuring Lyapunov exponents, one can discover qualitative distinctions between dynamical systems. I suspect that natural languages have complexity such as that of chaotic attractors (and thus positive Lyapunov exponents). One of the main points of this article is to suggest that linguistic theory's grammars are associated with dynamical systems that have Lyapuonv exponents equal to 0. This puts them right on the border between repeating processes and unpredictable ones. Though linguistic processes probably do not live right on the border, my suspicion is that they live near it, because they show a lot of similarity to the cases on the border. Thus, understanding how the cases on the border work may be a helpful step toward understanding how natural languages work.

The standard definition of Lyapunov exponents applies only to deterministic dynamical systems. The Appendix describes an extension to symbol-driven stochastic dynamical systems, which I used in discussing the following analyses. Under this extended definition, dynamical automata that process context-free languages have Lyapunov exponents equal to 0.

The next section reports measurements of Lyapunov exponents of self-organizing (learning) connectionist networks trained to generate languages defined by symbolic processes. I tested the hypothesis that the learning networks trained on phrasal-embedding languages would exhibit 0 Lyapunov exponents.

Zero Lyapunov exponents in connectionist networks trained on context-free languages would be of interest for several reasons. First, they would indicate that the self-organizing connectionist models develop the same fractal organization as the preprogrammed dynamical automata, and thus that insight into the easily understood dynamical automata can be extended to poorly understood connectionist models. Second, they would indicate a correspondence between an important dividing line in the realm of dynamical systems (zero Lyapunov exponents) and an important dividing line in the realm of symbolic computers (infinite-state Turing machines, which lie between finite-state machines and nonrecursive devices). Third, the "edge of chaos" (where Lyapunov exponents are 0) has been identified as important for living systems based on rather different considerations. For example, Alexander and Globus (1996) argue that brains have a recursive cellular organization that puts their dynamics on the edge of chaos, and Kauffman (1993) argues that optimal biological evolution occurs when systems are on the edge of chaos. The identification of a pervasive property of natural language syntax as chaos proximal would suggest that language theory might be incorporated into biology in a helpful new way and that linguistic analysis offers tools for understanding complex phenomena that might have application in other sciences.

SIMULATIONS

Which stack-based languages should be used to test the neural networks? Linguistic research has identified several patterns in natural languages that motivate the use of various kinds of stack memories. In this section, I introduce the training data by describing some of that research.

Linguistic–Computational Guidance

The syntactic patterns of natural language come in dizzying variety. Chomsky (1957) attempted to shine a beam on this feature of human-generated languages by describing a series of increasingly powerful computing mechanisms, now referred to as the *Chomsky hierarchy* (Table 3). About this series of devices, Chomsky posed the question: Which is the least powerful device that can encode all the order-of-morpheme patterns of each human language in the world?

A *finite-state language* is a set of element sequences that can be generated by a computer, called a *finite-state machine,* which only occupies a finite number of states. Each *context-free language* can be generated by a finite-state machine manipulating an unbounded pushdown stack (i.e., a first-in, last-out memory). Each *context-sensitive language* can be generated by a finite-state machine manipulating a memory that can be accessed in any order, provided the amount of memory needed by the device grows at most linearly with the length of the output. Each *Turing language* can be generated by a finite-state machine coupled with an any-access-order memory of unbounded (though still finite) size (Hopcroft & Ullman, 1979).

This hierarchy provides a kind of crude map, which is useful for navigating among the syntactic patterns that occur in natural languages. I concur with Moore (1998) that the Chomsky hierarchy is an imperfectly designed tool, and it will be useful to replace it with a better apparatus. I will discuss part of the case for this claim below, but the hierarchy is not altogether useless, and it serves well for getting started.

Grammars

Counting. Chomsky (1957) argued for the need for (at least) stack-based computation (above finite state on the hierarchy) on the basis of constructions such as (6), where S_i corresponds to some declarative sentence.

TABLE 3
The Formal Language Classes
of the Chomsky (1957) Hierarchy

Finite state	Finite number of states
Context free	Pushdown stack
Context sensitive	Linear bounded tape
Turing machine	Unbounded tape

(6) a. If S_1 then S_2.
b. If it is the case that if S_1 then S_2 then S_3.

A less formalized example reveals that the structure, (6b) can be part of a normal-sounding sentence (7).

(7) If it is true that if we leave without telling Bunnie where we are going then she will ransack the apartment when she returns, then I'd say let's wait until she gets back before we head out.

Simplification and generalization of (6), with substitution of shorter symbols for longer yields (8).

(8) If separator if separator if … then S_1 then … then S_k where the number of ifs = the number of thens = k.

Assuming the content of the S_is is arbitrary (a case of the simplifying context-freeness assumption), the embedding structure of this form is isomorphic to the embedding structure of the language $\{a^n b^n : n \in \{1, 2, 3, \ldots\}\}$ (i.e., the set of strings consisting of 1 or more *a*s followed by the same number of *b*s). As Rodriguez (2001) noted, a symbolic stack for generating this language functions simply as a counter—it counts the number of *a*s and remembers the count in order to determine how many *b*s to generate. This "counting language" is, in some sense, the simplest non-finite-state language. I chose it as the basis for the first network simulation. To design a generator with an embedding distribution similar to that of natural language, I used the probabilized version of $a^n b^n$ shown in Row Sl of Table 4. In this language, sentences of embedding level i occur half as often as sentences of embedding level $i - 1$.

Palindromic. Typically, languages do not repeat the same elements over and over in the same sentence (as in the language $a^n b^n$ just discussed). But on an abstract level, they use the same patterns over and over again, sometimes in rather elaborately nested configurations. For example, the transitive clause pattern (subj-verb-object) appears repeatedly in (9).

(9) The realization$_{1\text{-subj}}$ that someone$_{2\text{-Subj/3-Obj}}$ she$_{3\text{-Subj}}$ had known$_{3\text{-Verb}}$ well but had not seen$_{3\text{-Verb}}$ for over twenty years was about to walk$_{2\text{-Verb}}$ into the room filled$_{1\text{-Verb}}$ Tia$_{1\text{-obj}}$ with a kind of delicious dread.

Many subject–noun phrases can combine felicitously with a limited class of verbs. Simplifying considerably, we can approximate the dependencies with a list of preferred sequences (10).

TABLE 4
Infinite State Symbol-Generating Grammars

Grammar Number	*Name*	*Definition*
S1	Counting	**S** → S1 p **S1** → a (S1) b
S2	Palindromic	**S** → S1 p **S1** → S11/S12 **S11** → a (S1) b **S12** → (S1) y
S3	Interleaving	**S** → W_i**w**$_i$**p** **W**$_i$ → {n1, n2}* **w**$_i$ **is hom (W**$_i$**)**[a]

Note. A constituent in parentheses is present in half the instances and absent in the other half. Two or more constituents with slashes (/) between them split the instances equally among them. In Language S3, the probability of generating a string of length 2k + 1 was $1/2^k$. The symbol *p* is an end-of-sentence marker.

[a]hom (W_i) replaces n1 with v1 and replaces n2 with v2.

(10) N_1 V_1
N_2 V_2

The nesting then creates patterns such as (11).

(11) N_2 N_1 V_1 V_2
N_2 N_2 V_2 V_2
N_1 N_2 N_2 V_2 V_2 V_1
etc.

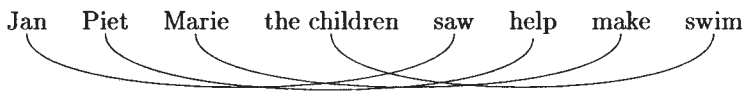
which, taken all ways, define the language WW′, where each W is a sequence of one or more N_1s and N_2s, and each W′is the corresponding reversed sequence of V_1s and V_2s (see Row S2 of Table 4). This language is a type of *palindrome language.* In the realm of symbolic devices, a pushdown stack is the minimal device that is needed to keep track of the dependencies in this language. For the second simulation, I used a pushdown automaton to generate strings from the palindrome language. The probability of embedding was again equal to .5 wherever embedding was possible.

Interleaving. Harman (1963), perhaps inspired by the ubiquity of nested dependencies in English and other languages, promoted the thesis that a restrictive theory of grammar should use only (context-free) phrase-structure grammars for syntactic representation (see also Pullum & Gazdar, 1982). Claims such as his prompted linguists to search the languages of the world for patterns that could not be handled by context-free rules. In fact, several cases turned up, most of them involving structures along the lines of the Dutch sentence (12)

(Huybregts, 1976; example taken from Bresnan, Kaplan, Peters, & Zaenen, 1982; see also Savitch, 1987).

(12) … dat Jan Piet Marie de kinderen zag helpen laten zwemmen
… that Jan Piet Marie the children saw help make swim
… that Jan saw Piet help Marie make the children swim.

The subject–verb correlations in this sentence have the interleaved structure shown in (13).

(13) 

Again, approximating the informational dependencies with constraints on the sequencing of types, we consider the language WW′ where W is a sequence of nouns chosen from $\{N_1N_2\}$ and W′ is the corresponding sequence of verbs with the verbs in the same order as the nouns. I refer to this language as an *interleaving language.* A simple interleaving language with two types of Ns and two types of Vs is listed on Row S3 of Table 4.

The existence of crossed-serial dependencies in Dutch and other languages was originally interpreted as evidence that natural languages lie higher on the Chomsky hierarchy than the context-free level. Following Moore (1998), I suggest a different interpretation: The Chomsky hierarchy is an imperfect framework. The simplest symbolic device that can keep track of the dependencies in an interleaving language is a queue automaton, which uses a first-in, first-out stack. The difference between pushdown stacks and queues is not a computational power difference but a difference in the kind of computational device involved. The fact that natural languages exhibit both kinds of structure is evidence that the theory of language needs to cut in at a more fundamental level. One response (not insightful) would be to list both stacks and queues in the array of devices available for the learners of natural languages to choose from when they set up a grammar. A better response, I suggest, is to define a more general computational framework and to let structure, in the sense of stacks and queues, develop emergently, in response to experience with the data. The following results indicate that the class of connectionist networks studied here constitutes such a framework.

Control Cases

The dynamical automaton hypothesis predicts that if a neural network learns any language that can be efficiently characterized with a stack mechanism (either

pushdown or queuelike), then the maximal Lyapunov exponent of the induced stochastic process should approximate 0. Because a learning network only converges on a perfect stack emulator in the limit (and is also limited by the precision of its implementation), measurements will produce values near 0, but not exactly 0. Thus, to test the dynamical automaton hypothesis, it is useful to construct a set of control cases against which quantitative comparisons can be made. When should the maximal exponent not be 0? If the learning algorithm functions conservatively in the sense that it does not build an emergent device more complex than it needs to for the task at hand, then all finite-state processes should lead to negative maximal exponents. Likewise, memoryless infinite-state processes should lead to negative maximal exponents. I examined a variety of such control cases.

Finite, memoryless. Within the finite-state languages, there is an even smaller class of languages that Chomsky did not include as a separate entry in his hierarchy—the *finite languages*. The sentences of a finite language can be listed in a finite-length list. For the first control case, I trained a network on the finite language consisting of the single sentence *a b c* (repeated over and over again throughout the training process; see Language Cl of Table 5).

Finite with memory. Crudely speaking, the difference between the finite-state and infinite-state languages on the Chomsky hierarchy is the inclusion of memory. But the memory of a stack or tape is a special kind of memory because its size is unbounded. There are many finite languages and finite-state languages that require the use of a memory too, a finite-length memory. One may wonder whether the presence of any correlational structure that requires the use of memory will induce a zero Lyapunov exponent. To address this question, I included a control net-

TABLE 5
Finite-State Symbol Generating Grammars

C1	Finite, memoryless	**S** → a b c
C2	Finite with memory	**S** → a b a c
C3	Infinite-state, memoryless	**S** → a (P1/P2) p
		P1 → a1 (P11/P12)
		P2 → a2 (P21/P22)
		P11 → a11 (P111/P112)
		P12 → a12 (P121/P122)
		P21 → a21 (P211/P212)
		P22 → a22 (P221/P222)
		…
C4	Finite-state with memory	**S** → (NP) v (NP) p
		NP → n (NP)

Note. A constituent in parentheses is present in half the instances and absent in the other half. Two or more constituents with slashes (/) between them split the instances equally among them. The symbol *p* is an end-of-sentence marker.

work that was trained on the language consisting of the sentence *a b a c* (Language C2 in Table 5). In order to distinguish between the *b* and *c* events, the processor needs to remember the event preceding each *a* while simultaneously attending to the current symbol—a simple, finite-memory task.

Infinite-state, memoryless. It is also possible to define an infinite-state language that requires no memory if one employs an infinite alphabet of symbols. Language C3 in Table 5 is such a language. This language has the structure of a set of lineages read randomly off an infinite-depth taxonomic tree starting at the root, with truncation probability always equal to .5. This language provides a valuable comparison to the palindrome language and the interleaving language because in all three cases, there is a hyperbolic (Zipf's law) relation between each state's frequency and its frequency rank (Zipf, 1949). One might, a priori, expect such 1/f structure in the frequency distribution to be associated with a chaotic or edge-of-chaos dynamical process. But because memory is not involved, the dynamical automaton hypothesis predicts limit-cycle dynamics and hence a negative maximal Lyapunov exponent.

Finite-state with memory. Early in the history of psycholinguistics, it was noted that probabilistic finite-state machines (also called finite-state Markov models) can provide good approximations of natural languages (Osgood & Sebeok, 1954). As control Language C4 (Table 5), I included the output of one such device, inspired by English compound noun and clause structure. This language contains infinite-length sentences and requires information to be stored over unbounded time, but it only requires a finite number of states.

For all four of these control cases, the dynamical automaton hypothesis predicts limit-cycle dynamics and, hence, negative maximal Lyapunov exponents in the network solution.

Results

The simulations were run with the simulator Lens (Rohde, 2001). Each network was trained for 3,000,000 pattern presentations with learning rate 0.0001. The simulations used "Doug's momentum" (value 0.9), a method of avoiding overly radical adjustment of weights when the cost function is steep. The root mean squared error (RMSE) at the end of training for each type of network (computed with respect to the grammar-derived probabilities on an appropriate test corpus of 200 random sentences in each case) are shown in the column labeled RMSE in Table 6. Root mean squared error is the standard error measure reported in neural-network studies—it gives an approximate sense of how the network performed quantitatively with respect to the cost function it was trying to minimize, but it does not give a very good sense of the qualitative character of the results in cases such as the ones at hand where correctness on specific structures is important. To

address qualitative performance, a particular word-to-word transition was defined as "correctly processed" if the vector of network activations on that transition was closer to the correct grammar-derived probability vector than it was to any other grammar-derived probability vector (Tabor & Tanenhaus, 1999). At the end of training, each network that processed embedded structures processed at least 69% of the transitions in doubly embedded sentences correctly, 88% of the transitions in singly embedded sentences correctly, and 98% of the transitions in matrix sentences correctly in a sample of 200 sentences, although performance was much better on the simplest case, $a^n b^n$ (> 99.3% of all transitions correct down to 6 levels of embedding). Each network that processed data from the taxonomic grammar processed all sentences at least 4 words long correctly in a sample of 200 random sentences. Each finite-state-trained network processed all transitions correctly in a sample of 200 random sentences. The superior results for matrix sentences and finite-state grammars may reflect the finite-state bias of the random-initial state of each network noted by Christiansen and Chater (1999) and Tiňo, Čerňanský, and Beňušková (2001). I note below, however, that at least in the case where the network did well on an infinite-state language ($a^n b^n$), the network cum learning algorithm may have an infinite-state bias.

Table 6 shows the average maximal Lyapunov exponent for each type of network. An analysis of variance (ANOVA) with language environment as random factor indicated that the maximal exponents were less negative for the set of stack-trained networks than for the set of non-stack-trained networks, $F(1, 5) = 15.66$, $p = .011$. This result is encouragingly consistent with the dynamical automaton hypothesis. A second analysis showed that the maximal exponents for just the stack-trained networks were significantly less than 0. This result was expected because the networks appear to approximate infinite-state computation by building progressively more complex limit-cycle machines over the course of training. The limit cycles lead one to expect negative exponents; only in the limit of infinite training should the maximal exponent actually equal 0.

TABLE 6
RMSE and Maximal Lyapunov Exponents
for the Net, Organized by Training Grammar

Label	*Grammar*	*RMSE*	*Standard (RMSE)*	*Stack Memory?*	*Maximal Exponent*	*SD*
S1	Counting	0.05	0.02	Yes	–0.32	0.10
S2	Palindromic	0.15	0.01	Yes	–0.31	0.20
S3	Interleaving	0.14	0.01	Yes	–0.22	0.09
C1	Finite	0.00014	0.0000039	No	–1.88	0.21
C2	Finite with memory	0.00029	0.000091	No	–1.44	0.24
C3	Infinite-state, memoryless	0.03	0.01	No	–1.74	0.19
C4	Finite-state with memory	0.013	0.003	No	–0.75	0.23

Note. RMSE = root mean squared error.

Structuralism and Causality

A related result sheds some new light on the discussion about the relation between discrete (or "symbolic") and dynamical computation. Carello et al. (1984) argued against a dualist treatment of these types (e.g., as advocated by Pattee, 1982) and stated that under their "strategy of elaborating continuous dynamics, the so-called discrete mode would be relieved of an explanatory role and relegated to the status of just one way (out of several or many ways) that a complex system might behave" (p. 237).

"One way out of several" is an accurate description of the status of the discrete mode in the models at hand. The implementation of stacklike computation in the recurrent network, a dynamical computer, depends on the creation of a precise balance in the timing of the model's habitation of expansive and contractive regions of the hidden unit manifold. Tabor (2000) showed that in one parameterizable dynamical automaton, the cases of context-free computation (one kind of discrete-mode computation) are rare atolls in a wide sea of non-context-free behavior. Countability considerations indicate that these non-context-free behaviors must consist mostly of super Turing processes, that is, they are outside the realm of the discrete mode.

When Carello et al. (1984) talked of relieving the discrete mode of its "explanatory role," I interpret them as rejecting the structuralist habit of discovering a pattern in a domain and describing a characterization of the pattern as an explanation. One wants to know why the pattern is there in a more ultimate sense. The present work attempts to probe more ultimate causes by examining language learning (inductive lawfulness). But one particular behavior of the learning mechanism at hand adds an interesting twist to the debate about the explanatory relevance of symbolic models: Tabor (2001) asked, How do networks such as those described above generalize when they are trained on just the most frequent sentences of one of the infinite-state probabilistic languages examined above? This question is a way of asking how the network goes beyond its input. The answer is that it shows a clear bias toward the infinite-state process from which the finite approximations were derived, even to the point of distorting its approximation of situations it has reliable experience with.

In Tabor (2001), I trained the network on the output of a series of finite-state grammars derived from the counting grammar mentioned previously. The first grammar just had the sentence *a b;* the second had, in addition, *a a b b;* the third added *a a a b b b,* and so forth. The probabilities of the sentences were the same as they were for the infinite-state counting language described above, except that the most deeply embedded sentence had twice the probability of its counterpart in the counting language and there was no possibility of continuing on after the last *a.*

I trained networks such as those described above on this succession of grammars (200,000 words presented from each). I compared the performance of the network after each stage of training to the performance of a variable-length Markov model (VLMM; Ron, Singer, & Tishby, 1996) trained on the same data. VLMMs work on

a simple principle: Predict the future by finding the longest identical past to the past of the current state; if there are multiple longest identical pasts, take their mean as the current prediction.[3] VLMMs are among the best finite-state methods of approximating natural language phenomena (Guyon & Pereira, 1995). Thus, comparing the network's performance to that of the VLMM provided a way of asking how the network tended to diverge from optimal finite-state behavior.

I tested the network and the VLMM on their predictions about new sentences with greater levels of embedding than those included in each training grammar. I used the infinite-state process as a standard. For all novel sentences, the network diverged from the VLMM in the direction of the infinite-state process: Its error with respect to the behavior of the infinite-state process was substantially lower than the error of the VLMM with respect to the same process (Figure 3). In fact, it even showed an infinite-statelike response to the most deeply embedded observed sentence, even though the training data provided a statistically reliable signal indicating finiteness—Figure 3. Tabor (2001) takes this result as suggesting that the infinite-state process may be a kind of attractor for the neural network: It tends to gravitate to it, even when the input only approximates it.

The possibility that the simple, context-free grammar $a^n b^n$ is an attractor of the learning process has particular relevance to the question of causes. Imagine a flock of neural networks, such as these, providing the training signals for "younger" neural networks that are trying to learn the "language of their community." If a particular grammar is an attractor in this sense, then it is plausible that the community would migrate toward that grammar across the generations. Such a model is, to be sure, rather unreal, because it is disembodied: There is no world that the language refers to; thus, for example, the language does not have a communicative function with respect to such a world. But it is not implausible that if extra linguistic reference were brought into the picture and it was useful to use context-free (or interleaved) embedded structures to describe the world at hand, then a lineage of network talkers such as these, with stacklike mental attractors, might well converge on one or another of the attractors over time. In this sense, the current model looks helpful for probing causes.

Thus, the simple infinite-state systems that symbolic theories have identified as structurally relevant may also turn out to be relevant to a theory of the causes of the nature of language organization. To be sure, just identifying these states as important, as discrete-mode theorizing has done, is not adequate for understanding the causality involved. But it may be a helpful first step. The present study suggests that learning neural-network models may be able to take this development a step further.

---

[3]In natural language modeling work using real corpora, exhaustive exploration of identical paths is not feasible and so approximation methods are needed. In such contexts, VLMMs are thus associated with a specific strategy for employing long histories only where they help (Guyon & Pereira, 1995; Ron et al., 1996). Because the present study involves simpler, artificial languages, maximal matching contexts and their associated probabilities can be computed accurately in all cases, so no truncation is used.

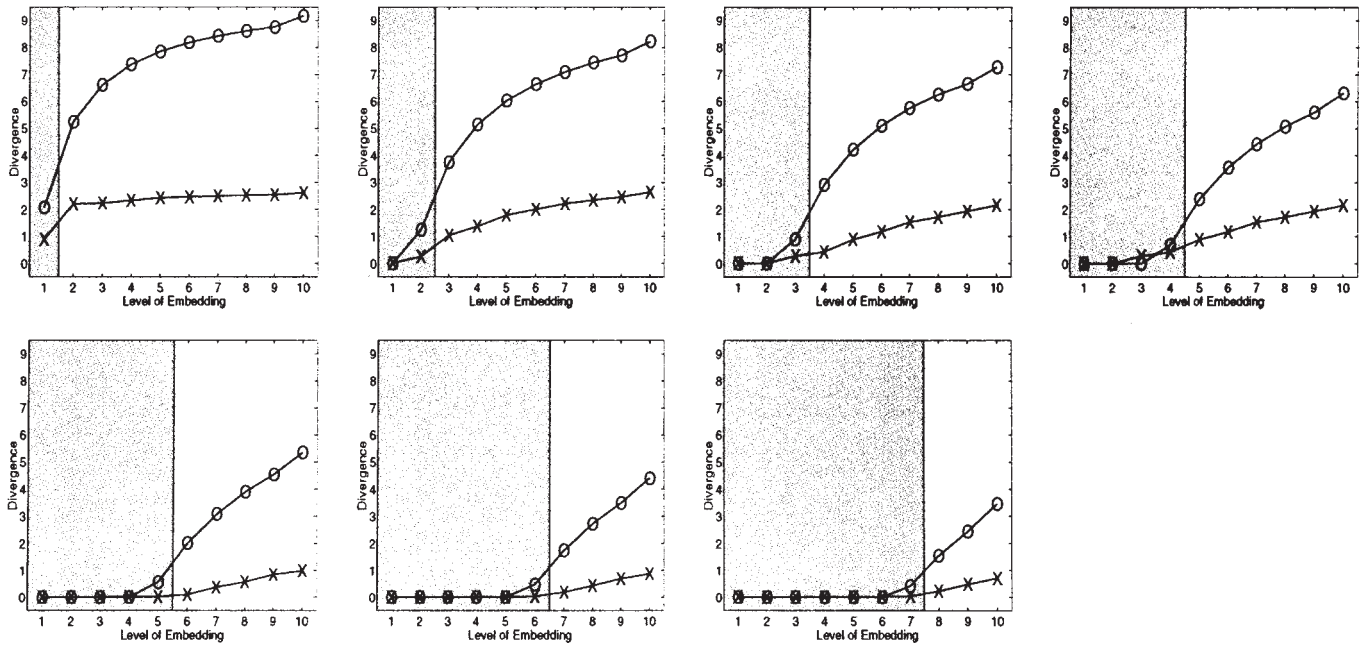


FIGURE 3 Generalization behavior of the $a^n b^n$ model when it is trained on successively longer strings. Subgraph *i* corresponds to training grammar with sentences with embedding level 1 … *i* for each *i*. Each subgraph is stratified into 10 levels of embedding (*x* axis). The curves marked *0* show the average Kullback–Leibler divergence per word between the VLMM and the infinite-state process. The curves marked X show the average divergence per word between the network and the infinite-state process. The shaded regions indicate the depths of embedding on which the network and VLMM were trained.

CONCLUSIONS

In a nutshell, this article has developed the following argument: Generative linguistic models of sentence-level structure in natural languages have several properties that ecological psychologists have rightly criticized in representationalist models in general: staticness, context-freeness, and lack of a lawful basis. Connectionist learning models offer an alternative that is more dynamic, desirably context sensitive, and explicit about the laws relating the organism's state to its environment. But it has not been clear how the connectionist learning devices can handle the complex temporal patterns that characterize natural language syntax. For this problem, representationalist mechanisms are manifestly useful. In earlier work (Tabor, 2000), I described a method (dynamical automata–fractal grammars) of translating the principles of their successes into the encoding framework in which connectionist models operate. The present study tested connectionist learning of complex, languagelike processes to see if they shared an important feature with the dynamical automata for such languages—zero Lyapunov exponents, or balance between habitation of contractive and expansive regions. Indeed, the results suggest that the learning connectionist models converge on representation with the same kind of fractal organization as the corresponding dynamical automata. Moreover, under similar learning conditions, Tabor (2001) found evidence suggesting that the infinite-state fractal computers are attractors of the learning system.

These results suggest a potentially insightful way of aligning discrete and dynamical computation. Limit cycles correspond to finite-state devices. Edge-of-chaos processes correspond to stack-based mechanisms. The fact that a Turing machine can be built with three stacks suggests that all recursively enumerable computations may lie on the edge of chaos in the current sense. I speculate that chaotic processes correspond to so-called super-Turing computation (Siegelmann, 1999). The results thus locate one of the core entities of discrete computation (stack mechanisms) at the heart of dynamical computation as well. This insight shows promise of helping us figure out how to understand connectionist learning of the complex dependencies of natural language syntax. For example, one of the first insightful analyses of connectionist learning of complex languages, Rodriguez (2001), succeeds by training networks and then building similar dynamical automata. If, as Tabor (2001) suggests, the stack-based computers are attractors of the dynamical learning mechanism, then they may also play a central role in understanding what causes languages to be organized into constituent structures.

The value, then, of symbolic, representationalist objects, is that they provide important bearing points for exploring complex dynamical learning systems, a psychologically appealing class of models. They are not by themselves, as the orthodox theories hold, good models of human mental states. The problem is that if one restricts one's ontological prospect to just the bearing points, then one is helpless when it comes to navigation in the surrounding space. Such navigation is very

helpful for addressing inductive lawfulness, as the connectionist models demonstrate. But having some bearing points for the navigation is also a good thing.

In the introduction, I characterized a lawful theory, for present purposes, as one that provides a complete and coherent account of how mental states and environments coevolve at the timescale of moment-to-moment experience. When lawfulness is the topic, ecological psychologists are inclined to emphasize "specificational" lawfulness—illustrated, for example, by the relation between a physical situation and the optic flow pattern that it produces (Turvey & Carello, 1985). Specificational lawfulness contrasts with "indicational" lawfulness, which characterizes, among other things, the relation between a linguistic symbol and the meaning to which it is conventionally linked. If one assumes that linguistic objects (by which I mean actually occurring phonemes, morphemes, phrases, etc.) are to be treated as symbols and are thus of a fundamentally different ilk from other objects in the world (e.g., sidewalks, doors, and chairs), then it would appear that indicational lawfulness is fundamentally different from specificational lawfulness. Indeed, linguistic understanding seems, on initial observation, to require a different kind of computation from the kind that works well for detecting kinematically relevant invariants in rays of light. But what the connectionist models discussed here suggest is that linguistic understanding may not operate so differently after all. At the basic level, it may depend on the same necessity of establishing invariants that support effective action. It may require the same enfolded organization of metric-space computation, rather than the disconnected memory structure of Turing machines. The suggested similarity is supported by the fact that connectionist models operating on very similar principles to the ones described here have been used successfully to characterize nonlinguistic visual, olfactory, and auditory domains that appear to involve specification in the classical Gibsonian sense (Arbib, 1995). Convention seems strange in comparison to something from classical physics (e.g., gravity) because of the prominent role of arbitrariness. But it seems less strange in comparison to biological variety (Millikan, 2001), and there are good reasons to believe that biological variety is founded on a specificational relation between each organism's environment and its nature (Thompson, 1917/1992). The prominence of linguistic arbitrariness may reflect the fact that we live on such intimate terms with linguistic conventions. The shift of the focus from language processing to language induction makes the conventional features less distracting because they become small bits of material that form the substance of a larger, law-governed whole.

I conclude with four ideas:

- It might be helpful to study connectionist models of phenomena, such as haptic perception, dynamic touch, locomotion, and so forth, in order to situate these in the computational (as well as the dynamical) framework outlined here.
- Representationalist entities might play a useful role in the ecological study of nonlinguistic perception if they were viewed as conceptual bearing points rather

than as sufficient models of the phenomena. For example, perhaps there are learned schemata of limb movement that could so serve.

- The notion that symbolic objects might be attractors of mental dynamics suggests a new species of observer's paradox: Why are representationalist models so appealing to many researchers? Perhaps the conceptions they identify are familiar because our mental states are likely to spend time near them. Why is it hard to examine the bigger picture, even when we can model it? Perhaps it is because the nonsymbolic states are mental transients that we experience only fleetingly.
- Symbolic objects as attractors do, I believe, help probe the question of causality, but they also raise an interesting question about the proper approach to explanation from an ecological perspective: Are the stable states there because they afford us a selective advantage in the world (e.g., fractal language is useful for talking about the recursive structure of nature)? Or are they there because our minds and the world are traveling in a kind of abstract, information-theoretic reality, and the observed topologies are the (objective) nature of that reality? The notion of universal "laws of information" favors the latter view.

ACKNOWLEDGMENTS

This research was partly supported by University of Connecticut Research Foundation Grant FRS 444038. Thanks to Chaopeng Zhou for discussions and help with running the simulations. Many thanks to Carol Fowler, Claire Michaels, William Mace, and Guy van Orden for helpful feedback on earlier drafts of this article.

REFERENCES

Abarbanel, H. D. I. (1996). *Analysis of observed chaotic data.* New York: Springer-Verlag.

Alexander, D. M., & Globus, G. G. (1996). Edge-of-chaos dynamics in recursively organized neural systems. In E. MacCormac & M. I. Stamenov (Eds.), *Fractals of brain, fractals of mind: In search of a symmetry bond* (pp. 31–73). Amsterdam: Benjamins.

Arbib, M. A. (1995). *The handbook of brain theory and neural networks.* Cambridge, MA: MIT Press.

Barnsley, M. (1993). *Fractals everywhere* (2nd ed.). Boston: Academic.

Bresnan, J. (1982). *The mental representation of grammatical relations.* Cambridge, MA: MIT Press.

Bresnan, J., Kaplan, R. M., Peters, S., & Zaenen, A. (1982). Cross-serial dependencies in Dutch. *Linguistic Inquiry, 13,* 613–635.

Carello, C., Turvey, M. T., Kugler, P. N., & Shaw, R. E. (1984). Inadequacies of the computer metaphor. In M. S. Gazzaniga (Ed.), *Handbook of cognitive neuroscience* (pp. 229–248). New York: Plenum.

Charniak, E. (1993). *Statistical language learning.* Cambridge, MA: MIT Press.

Chomsky, N. (1957). *Syntactic structures.* The Hague, The Netherlands: Mouton.

Chomsky, N. (1981). *Lectures on government and binding.* Dordrecht, The Netherlands: Foris.

Christiansen, M. H. (1994). *Connectionism, learning, and linguistic structure.* Unpublished doctoral dissertation, Department of Linguistics, University of Edinburgh, Scotland.

Christiansen, M. H., & Chater, N. (1999). Toward a connectionist model of recursion in human linguistic performance. *Cognitive Science, 23,* 157–205.

Crutchfield, J. P. (1994). The calculi of emergence: Computation, dynamics, and induction. *Physica D, 75,* 11–54.

Dennett, D. C. (1978). *Brainstorms: Philosophical essays on mind and psychology.* Cambridge, MA: MIT Press.

Elman, J. (1990). Finding structure in time. *Cognitive Science, 74,* 179–211.

Elman, J. (1991). Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning, 7,* 195–225.

Elman, J. (1995). Language as a dynamical system. In R. Port & T. van Gelder (Eds.), *Mind as motion: Explorations in the dynamics of cognition.* Cambridge, MA: MIT Press.

Frege, C. (1952). On sense and reference. In P. T. Geach & M. Black (Eds.), *The philosophical writings of Gottlob Frege* (pp. 56–78). Oxford, England: Basil Blackwell. (Original work published 1892)

Gibson, E. (1998). Linguistic complexity: Locality of syntactic dependencies. *Cognition, 68,* 1–76.

Gibson, E., & Wexler, K. (1994). Triggers. *Linguistic Inquiry, 25,* 407–454.

Gibson, J. J. (1986). *The ecological approach to visual perception.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc. (Original work published 1979)

Guyon, I., & Pereira, F. (1995). Design of a linguistic postprocessor using variable memory length Markov models. In *International Conference on Document Analysis and Recognition* (pp. 454–457). Montreal, Quebec, Canada: IEEE Computer Society Press. (Available at www.clopinet.com/isabelle/Papers/index.html)

Harman, C. H. (1963). Generative grammars without transformational rules. *Language, 39,* 597–616.

Haykin, S. S. (1994). *Neural networks: A comprehensive foundation.* New York: Macmillan.

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

Hopcroft, J. E., & Ullman, J. D. (1979). *Introduction to automata theory, languages, and computation.* Menlo Park, CA: Addison-Wesley.

Huybregts, M. A. C. (1976). Overlapping dependencies in Dutch. *Utrecht Working Papers in Linguistics, 1,* 24–65.

Hyams, N. M. (1986). *Language acquisition and the theory of parameters.* Dordrecht, The Netherlands: Reidel.

Joshi, A. K., & Schabes, Y. (1996). Tree-adjoining grammars. In C. Rosenbeerg & A. Salomaa (Eds.), *Handbook of formal languages* (Vol. 3, pp. 69–123). New York: Springer-Verlag.

Kamp, H., & Reyle, U. (1993). *From discourse to logic: Introduction to model-theoretic semantics of natural language, formal logic, and discourse representation theory.* Dordrecht, The Netherlands: Kluwer Academic.

Kauffman, S. (1993). *The origins of order: Self-organization and selection in evolution.* Oxford, England: Oxford University Press.

Lasnik, H. (1990). *Essays on restrictiveness and learnability.* Dordrecht, The Netherlands: Kluwer Academic.

Millikan, R. G. (2001). *Purposes and cross-purposes: On the evolution of languages and language.* Unpublished manuscript, Department of Philosophy, University of Connecticut.

Minsky, M., & Papert, S. (1969). *Perceptrons.* Cambridge, MA: MIT Press.

Montague, R. (1974). English as a formal language. In R. H. Thomason (Ed.), *Formal philosophy: Selected papers of Richard Montague* (pp. 108–221). New Haven, CT: Yale University Press. (Original work published 1970)

Moore, C. (1998). Dynamical recognizers: Real-time language recognition by analog computers. *Theoretical Computer Science, 201,* 99–136.

O'Reilly, R. C., & Munakata, Y. (2000). *Computational explorations in cognitive neuroscience.* Cambridge, MA: MIT Press.

Oseledec, V. I. (1968). A multiplicative ergodic theorem: Lyapunov characteristic numbers for dynamical systems. *Trudy Moskovskogo Matematicheskogo Obshchestva, 19,* 197.

Osgood, C., & Sebeok, T. (1954). Psycholinguistics: A survey of theory and research problems. *Journal of Abnormal and Social Psychology, 49*(4, Pt. 2), 1–203.

Pattee, H. H. (1982). The need for complementarity in models of cognitive behaviors: Response to Carol Fowler and Michael Turvey. In W. Weimer & D. Palermo (Eds.), *Cognition and the symbolic processes* (Vol. 2). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Pollard, C., & Sag, I. A. (1994). *Head-driven phrase structure grammar.* Chicago: University of Chicago Press.

Pullum, G. K., & Gazdar, G. (1982). Natural languages and context free languages. *Linguistics and Philosophy, 4,* 471–504.

Rodriguez, P. (2001). Simple recurrent networks learn context-free and context-sensitive languages by counting. *Neural Computation, 13,* 2093–2118.

Rohde, D. (2001). Lens, the light, efficient network simulator [Computer software]. Retrieved from http://www.cs.cmu.edu/dr/Lens/

Rohde, D., & Plaut, D. (1999). Language acquisition in the absence of explicit negative evidence: How important is starting small? *Journal of Memory and Language, 72,* 67–109.

Ron, D., Singer, Y., & Tishby, N. (1996). The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning, 25,* 117–149.

Rumelhart, D., Durbin, R., Golden, R., & Chauvin, Y. (1995). Backpropagation: The basic theory. In Y. Chauvin & D. Rumelhart (Eds.), *Backpropagation: Theory, architectures, and applications* (pp. 1–34). Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

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

Savitch, W. J. (Ed.). (1987). *The formal complexity of natural language.* Norwell, MA: Kluwer.

Servan-Schreiber, D., Cleeremans, A., & McClelland, J. L. (1991). Graded state machines: The representation of temporal contingencies in simple recurrent networks. *Machine Learning, 7,* 161–193.

Siegelmann, H. T. (1999). *Neural networks and analog computation: Beyond the Turing limit.* Boston: Birkhäuser.

Tabor, W. (1994). Syntactic innovation: A connectionist model. *Dissertation Abstracts International, 55*(01), 3178A.

Tabor, W. (2000). Fractal encoding of context-free grammars in connectionist networks. *Expert Systems: The International Journal of Knowledge Engineering and Neural Networks, 17,* 41–56.

Tabor, W. (2001). *Lyapunov characteristic exponents of discrete stochastic neural networks.* Unpublished manuscript, University of Connecticut.

Tabor, W., Juliano, C., & Tanenhaus, M. (1997). Parsing in a dynamical system: An attractor-based account of the interaction of lexical and structural constraints in sentence processing. *Language and Cognitive Processes, 12,* 211–271.

Tabor, W., & Tanenhaus, M. K. (1999). Dynamical models of sentence processing. *Cognitive Science, 23,* 491–515.

Tanenhaus, M., Carlson, G., & Trueswell, J. (1989). The role of thematic structures in interpretation and parsing. *Language and Cognitive Processes, 4,* S1211–S1234.

Thompson, D. W. (1992). *On growth and form.* New York: Dover. (Original work published 1917)

Tiňo, P., Čerňanský, M., & Beňušková, L. (2001). *Markovian architectural bias of recurrent neural networks.* Unpublished manuscript, Neural Computing Research Group, Aston University, Birmingham, England.

Turvey, M. T., & Carello, C. (1981). Cognition: The view from ecological realism. *Cognition, 10,* 313–321.

Turvey, M. T., & Carello, C. (1985). The equation of information and meaning from the perspectives of situation semantics and Gibson's ecological realism. *Linguistics and Philosophy, 8,* 81–90.

van Eijck, J., & Kamp, H. (1996). Representing discourse in context. In J. van Benthem & A. T. Meulen (Eds.), *Handbook of logic and linguistics* (pp. 179–237). Oxford, England: Elsevier.

Von Bremmen, H., Udwadia, F. E., & Proskurowski, W. (1997). An efficient method for the computation of Lyapunov numbers in dynamical systems. *Physica D, 110,* 1–16.

Wolf, A., Swift, J., Swinney, H., & Vastano, J. (1985). Determining Lyapunov exponents from a time series. *Physica D, 16,* 285–317.

Zipf, G. K. (1949). *Human behavior and the principle of least effort: An introduction to human ecology.* Cambridge, MA: Addison-Wesley.

APPENDIX: LYAPUNOV EXPONENTS FOR SYMBOL-DRIVEN STOCHASTIC DYNAMICAL SYSTEMS

Let

$$\vec{x}_{t+1} = f(\vec{x}_{t+1}) \tag{A1}$$

be a discrete, deterministic dynamical system with n dimensional state, $\vec{x}$. The Lyapunov exponents, λ_i for $i = 1, \ldots, n$, of the trajectory starting at $\vec{x}$ are the logarithms of the eigenvalues of the matrix

$$OSL(\vec{x}) = \lim_{t\to\infty} (Df_t)^T (Df_t))^{\frac{1}{2t}}(\vec{x}) \tag{A2}$$

where T denotes transpose, $Df(\vec{x})$ is the Jacobian of f at $\vec{x}$ and

$$Df(\vec{x}) = D[f_t \circ f_{t-1} \circ \ldots \circ f(\vec{x})] = Df(\vec{x}_t) \cdot Df(\vec{x}_{t-1}) \cdot \ldots \cdot Df(\vec{x}) \tag{A3}$$

For $\vec{x}$ in the basin of a single attractor, the values of the eigenvalues are essentially independent of the choice of $\vec{x}$ so we may speak of the Lyapunov exponents of the attractor. From another perspective, the ith eigenvalue measures the average rate of growth of the ith principle axis of infinitesimal ellipses surrounding points on the attractor (Wolf, Swift, Swinney, & Vastano, 1985). The sum of the Lyapunov exponents indicates the global stability of the system: The sum must be negative for the system to have Lyapunov stability. If all the exponents are negative, then the system is a limit cycle and visits only finitely many points. If at least one exponent is positive (and the sum is negative), then the system is chaotic. The case in which the greatest Lyapunov exponent is 0 in a discrete system is a special case that can yield complex behavior (famously for the logistic map at the "edge of chaos"; Crutchfield, 1994).

Definition (A2) can be extended to symbol-driven stochastic dynamical systems (such as the neural networks discussed previously) as follows: Because of the nondeterminicity, we have to let the definition of eigenvalues depend not only on

the initial condition but also on the specific random sequence of inputs, S, which the autonomously functioning network chooses:

$$OSL(\vec{x}, S) = \lim_{t \to \infty} (Df_t)^T (Df_t))^{\frac{1}{2t}} (\vec{x}) \qquad (A4)$$

In Tabor (2001), I provide simulation evidence that this particular extension of the definition of Lyapunov exponents to stochastic systems is useful in the sense that the logarithms of the eigenvalues of this matrix are constants for almost all $\vec{x}$ and corresponding sequences S. In other words, the generalized Lyapunov exponents defined by (A4) and the standard Lyapunov exponents defined for deterministic systems are invariants with respect to the same types of sets. This suggests that they provide a stable characterization of the stochastic system's expansion and contraction behavior. Encouraged by such results, I use these measures here as a way of characterizing and comparing various stochastic dynamical systems (dynamical automata and learning neural networks). I conjecture also that the Lyapunov exponents of the stochastic dynamical systems provide analogous classificatory information about dynamical regimes of the stochastic systems, as the Lyapunov exponents of the deterministic systems provide for deterministic systems For example, negative exponents indicate limit cycles; a negative sum of exponents indicates a dissipative system; positive exponents in a dissipative system indicate chaos, and so forth (see Abarbanel, 1996).

In the case of pushdown dynamical automata such as DA 0 above, the characteristic exponents can be calculated exactly. If DA 0 starts at $\vec{z} = (½,½)$, then every trajectory produces a string with an equal number of *a*s, *b*s, *c*s, and *d*s. The Jacobian for the *b* and *c* transitions is I, the identity matrix. For the *a* transitions, it is $\begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$, and for the *d* transitions, it is $\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$. Therefore, the Oseledec matrix is I in the limit and both Lyapunov exponents are 0. A similar analysis applies anytime a fractal set is used to keep track of a stack memory, where well-formedness maps to an empty stack. In the general case, there must be at least one dimension of fractal scaling, and that dimension will produce a zero Lyapunov exponent. The Lyapunov values reported in the main body of the paper were calculated using the algorithm of Von Bremmen, Udwadia, and Proskurowski (1997).