Connection Science, Vol. 2, Nos 1 & 2, 1990, pp. 53-62.

Syntactic Transformations on Distributed Representations

David J. Chalmers

Center for Research on Concepts and Cognition
Indiana University
Bloomington, Indiana 47405.
E-mail: dave@cogsci.indiana.edu

Abstract

There has been much interest in the possibility of connectionist models whose representations can be endowed with compositional structure, and a variety of such models have been proposed. These models typically use distributed representations that arise from the functional composition of constituent parts. Functional composition and decomposition alone, however, yield only an implementation of classical symbolic theories. This paper explores the possibility of moving beyond implementation by exploiting holistic structure-sensitive operations on distributed representations. An experiment is performed using Pollack's Recursive Auto-Associative Memory. RAAM is used to construct distributed representations of syntactically structured sentences. A feed-forward network is then trained to operate directly on these representations, modeling syntactic transformations of the represented sentences. Successful training and generalization is obtained, demonstrating that the implicit structure present in these representations can be used for a kind of structure-sensitive processing unique to the connectionist domain.

1 Introduction

Since the critique by Fodor and Pylyshyn (1988), connectionists have been investigating possible methods of endowing their representations with some kind of compositional structure, and with operations that are sensitive to this structure. To this end, various models have been devised by Elman (in press), Pollack (1988; in press), and Smolensky (in press), among others. Most of these models have utilized implicit structure in distributed representations, which are constructed and possibly deconstructed via various functional manipulations. This stands in clear contrast to the explicit constituent structure used in "Classical" symbolic models.

Van Gelder (1990) has expressed this distinction by noting that the connectionist implementations of compositional structure have used "functional compositionality," rather than the "concatenative compositionality" that has been used in more traditional AI approaches. Functional compositionality is achieved by the use of (possibly quite complex) *functions* which operate on symbol tokens, and produce a coded representation of a complex compositional structure. In turn, there are further functions which take us back from the compositional representation to the constituent parts. These models differ from the symbolic models in that the compositional representation need not contain explicit physical tokens of the original constituents, but may instead only contain the original information in a very implicit way.

It must be noted, however, that this methodology as it stands is vulnerable to Fodor and Pylyshyn's second-favourite reply to connectionists: "But then it's only an *implementation* of the Classical approach!" If this is all there is to the story, then "functional compositionality" is no more than a simple variation of Classical compositionality. The "composing" functions are simple implementations of Classical composing functions, like the "cons" operation in Lisp. The "extracting" functions are implementations of Classical extraction function, like "car" and "cdr" in Lisp. Describing the model in terms of these operations gives a complete operational description of the system's processing, at a higher level of abstraction. (The main difference might be a slightly degraded performance of the connectionist models, due to imperfect preservation of information over the operations.) As the Classicist might put it: what the representation *looks like* is a mere implementational detail.

But more is offered by the connectionist account. To see this, we should first note that the *only* operations that are available on a Classical compositional representation are those of extraction or further composition. In particular, to do anything with such a representation — to exploit the information that is contained therein — one must first go through the process of extracting the original constituent tokens. Any program that uses a Lisp list, for instance, must at some stage extract the elements of the list with an operation like "car." On the connectionist account, this restriction need not apply. Connectionism offers the opportunity to operate on compositional representations *holistically*, without first proceeding through the step of extraction.

The reason that connectionist models have this ability lies in the fact that connectionist representations are much richer than symbolic representations. *All there is* to a symbolic representation is its compositional structure — beyond this, it is just primitive, atomic components. Connectionist representations may contain compositional

structure, but the compositional structure does not nearly exhaust the properties of the representation. The complex, distributed *microstructure* of these representations contains much information that may be exploited for a variety of purposes.

When a connectionist representation is operated on holistically, it is this microstructure that is being exploited. And these holistic operations represent a whole new range of possible functions that can be used on connectionist representations. There is thus a *functional* distinction between connectionist and Classical models of compositionality, and it is in this functional distinction that the promise of connectionist compositional representation lies. In no sense can these functions be regarded as implementations of Classical operations; they open a whole new area of research.

This paper describes an experiment in the use of such holistic operations to model structure-sensitive processes. These operations take place within the context of a Recursive Auto-Associative Memory (RAAM), a very interesting connectionist model of compositional structure devised by Pollack (1988). So far, RAAM has mostly been used to implement the symbolic operations of composition and extraction. This experiment demonstrates that it is also well-suited for the modeling of structure-sensitive operations in a manner which is no mere implementation — something that Fodor and Pylyshyn claimed was impossible. (The future possibility of such processing is anticipated by Pollack (in press), under the name of "Associative Inference.")

The particular structure-sensitive operations that are used here are syntactic transformations on sentences. Specifically, the passivization of sentences is modeled — for instance, the transformation from "John loves Michael" to "Michael is loved by John." The status of such transformations as real cognitive entities is somewhat questionable these days, and no strong commitment is being made here to any particular linguistic paradigm. Rather, transformations are being used as an example of the kind of complex, structured operation in natural language with which connectionist approaches are supposed to have difficulty.

2 The RAAM architecture

The architecture of RAAM is described in detail in Pollack (in press). RAAM is in principle capable of developing distributed representation of fixed-valence tree structures of arbitrary depth. (In this experiment, all trees will have valence 3.) This is achieved via a recursive encoding procedure.

The basic structure of the architecture is shown in Figure 1. This is a simple 3-layer feed-forward network, with 3N units in the input and output layers, and N units in the hidden layer, where N is some positive integer. To encode a given tree, we start with the its terminal elements. Each of these is encoded as a pattern over N units. Starting at the bottom of the tree, these are fed into the network three at a time (where every three inputs correspond to three leaves that descend from a single node of the tree). Assuming the network is functioning correctly, the hidden layer will be activated with a distributed representation of the inputs, compressed to one-third of the original size. This representation can then be used in turn as input for another run of the network, combined with two other representations (which may be terminal or non-terminal). We proceed in this manner, recursively up the tree, until we have a single compressed distributed representation of the entire tree.

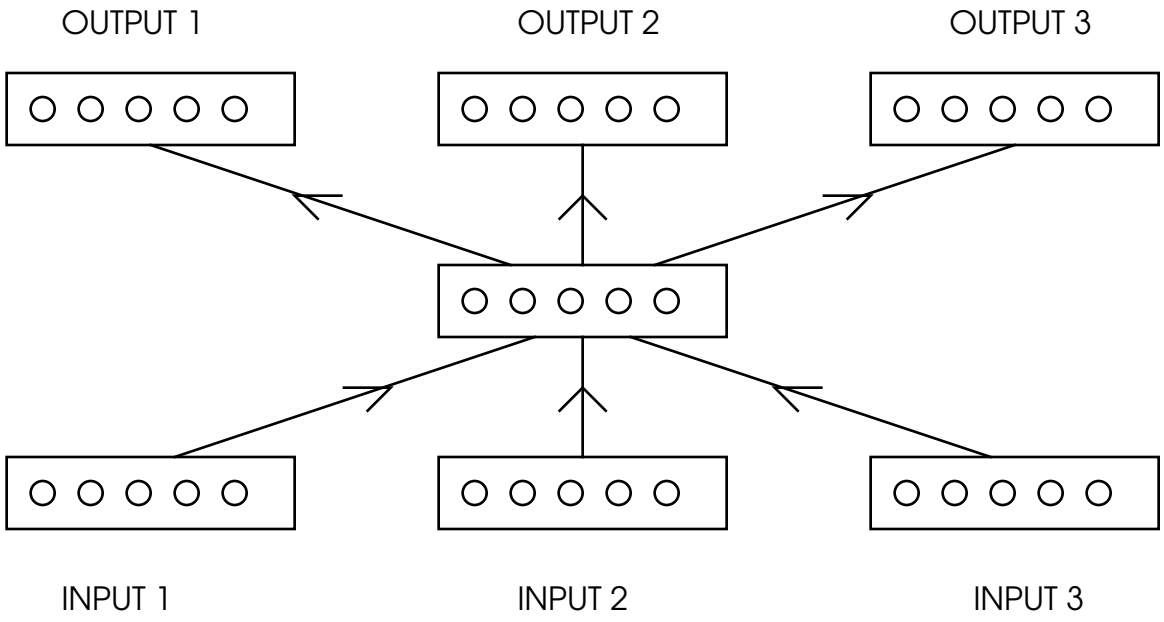

Figure 1. The basis of the RAAM Architecture

Of course, the network has to be set up to encode the representations in a useful way. This is done by first training the network to *auto-associate* the desired inputs, using the familiar backpropagation procedure. The appropriate sets of three terminal representations are given as inputs, and the network is trained to reproduce the same representations in the output layer. The hidden layer is thus forced to develop a compressed representation of the original inputs. At the same time, the hidden layers from these training cycles are extracted and used in turn as inputs to train the network to auto-associate on higher-order structures, and so on. (As the network learns, the patterns in the hidden layers change, so we get a kind of "moving target" learning.)

If this training process is carried through to completion, not only will we have a reliable encoding process but also a *decoding* process into the bargain. To decode a compressed representation of a given tree, we need only feed it in directly to the hidden layer of the network, and propagate activation to the output layer. If the output layer consists of representations of terminal symbols, we are done; otherwise, we take any non-terminal representations and repeat the process on these, recursively, until we bottom out with terminal representations everywhere. The only difficulty lies in the fact that we need a decision procedure to decide whether a given representation is a terminal element, or whether it is a non-terminal representation that needs to be further decoded. Such procedures can be devised quite simply, but finding an optimal decision procedure may require some thought.

3 The implementation of syntactic structures in RAAM

To enable the desired syntactically sensitive operations on sentences, we need a representation of their syntactic structures. This is done in a simple manner, representing sentences as valence-3 trees as shown in Figure 2. In this small-scale experiment, all sentences used have one of the two basic structures shown in Figure 2. Note that dummy

objects ("NIL") are used to fill in the spaces where only two of the three leaves descending from a node are required.

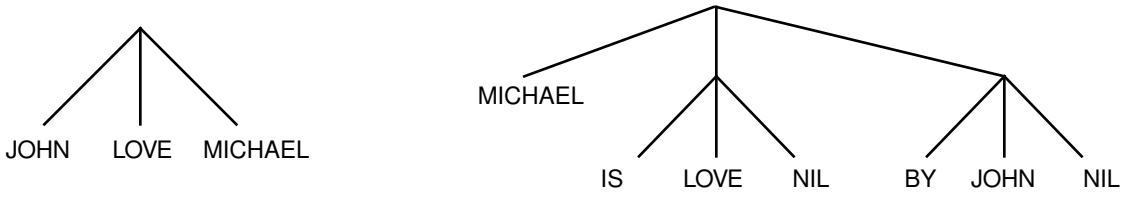

Figure 2. Examples of sentences to be represented.

Before training the RAAM, a terminal representation for the words has to be chosen. A simple localist representation is used here. Terminal representations consist of 13 units in all. The first 6 units represent "part of speech". The parts of speech used here are, somewhat arbitrarily: noun, proper noun, verb, adjective, auxiliary verb and preposition. (As it happens, only "proper noun", "verb", "auxiliary verb" and "preposition" are needed in this experiment.) Only one of these units is activated in a given word. The next 5 units represent the particular word. Again, only one of these units can be on, so there is a maximum vocabulary of 5 words for every part of speech, or 30 words in all. The final 2 units are spare units. These are never used for input representations, but provide valuable extra space to be used by the distributed representations formed by the RAAM.

The representations of all words used in the experiment are shown in Table 1. The dummy word ("NIL" in Figure 2) is represented by having all 13 units unactivated. It should be noted that no attempt is made to represent verb tense or subject-verb agreement. The variation of verb form has been investigated separately by Rumelhart and McClelland (1986), among others.

WORD	REPRESENTATION		
JOHN	0 1 0 0 0 0	1 0 0 0 0	0 0
MICHAEL	0 1 0 0 0 0	0 1 0 0 0	0 0
HELEN	0 1 0 0 0 0	0 0 1 0 0	0 0
DIANE	0 1 0 0 0 0	0 0 0 1 0	0 0
CHRIS	0 1 0 0 0 0	0 0 0 0 1	0 0
LOVE	0 0 1 0 0 0	1 0 0 0 0	0 0
HIT	0 0 1 0 0 0	0 1 0 0 0	0 0
BETRAY	0 0 1 0 0 0	0 0 1 0 0	0 0
KILL	0 0 1 0 0 0	0 0 0 1 0	0 0
HUG	0 0 1 0 0 0	0 0 0 0 1	0 0
IS	0 0 0 0 1 0	1 0 0 0 0	0 0
BY	0 0 0 0 0 1	1 0 0 0 0	0 0
NIL	0 0 0 0 0 0	0 0 0 0 0	0 0

Table 1. Representations of words used in the experiment.

Altogether, there are 125 possible sentences of each of the two forms shown in

Figure 2 (as there are five possibilities for each of the three variable parts of the sentences). A sample of 40 sentences of the active form was randomly generated. These sentences, along with the 40 corresponding passivized sentences, were used as a training corpus. A further (non-overlapping) sample of 40 more sentences of each type was generated, to be used as a test of generalization abilities.

3.1 Training the RAAM

The initial corpus of 80 sentences, 40 of each type, was used to train the RAAM in the manner specified in Section 2. There were 160 cycles of the network per epoch, one for each of the original sentences and three for each of the passivized sentences (corresponding to the three internal nodes of the tree). The initial learning rate was 0.1; this was lowered to 0.025 by the end of the training procedure. A momentum rate of 0.9 was used.

The network was trained for 6400 epochs, by which time all but 20 output units (out of 6240 in all) had an error of less than 0.05. The maximum error on any output unit was 0.12.

3.2 Testing the RAAM

To use the decoding ability of the RAAM, we first need an appropriate test for terminal elements. In this experiment, a representation was deemed non-terminal if more than 2 units had an activation greater than 0.15 and less than 0.85, or if either of the two spare units had activation greater than 0.5, or if more than one speech part unit had activation greater than 0.5. To prevent the possibility of infinite descent, a maximum depth of 3 was imposed on the decoded trees. (This is one greater than the maximum depth of trees actually used in the experiment.)

If a representation was deemed terminal, then a simple procedure was used to find the associated word. If no speech part unit had activation greater than 0.5, then the word was deemed the "dummy" word; else the speech part was determined by the unit with maximum activation. The associated specific word was obtained by taking the "word" unit with maximum activation.

As an initial test, the 80 sentences in the training corpus were recursively encoded, and then decoded using the above method. Unsurprisingly, all 80 were decoded back to the correct original sentence.

As a test of generalization, the 80 new sentences from the testing corpus were encoded and decoded in the same fashion. All but 13 of these sentences decoded back to the original sentence. Of the 13 mistakes, only one was decoded to an incorrect sentence *structure*; the other 12 all had one incorrect word within a correct sentence structure.

A generalization rate of over 80% seems to be remarkably good. This appears to indicate that the RAAM network is very appropriate for encoding complex sentence structures, although of course more work is needed to investigate the results of encoding many different sentence types. Detailed analysis of how the sentences are internally represented has not yet been undertaken.

4 The Transformation Network

So far, what has been described amounts merely to a connectionist *implementation* of Classical symbol structures. To get beyond an implementation, we must use operations other than just composition and extraction.

Here, we will model the process of syntactic transformation, by operating *directly* on the compressed distributed representations of sentences, and without passing through any stages of composition or extraction. This kind of operation is not present in Classical models.

The transformation we choose to model here is that of passivization, though many others could also be modeled. To do this, we take a sentence of a similar form to "John loves Michael," and the corresponding passivized sentence of the form "Michael is loved by John." Using the RAAM as trained above, we encode these using the tree structure of Figure 2, yielding a distributed representation for each sentence. We wish to train a network to transform directly from one distributed representation to the other.

The network that we will use is another simple 3-layer feedforward network (of course, it is not recursive this time), as shown in Figure 3. The input and output layers both consist of 13 units, the size of the compressed representations developed by the RAAM. The number of units in the hidden layer varied in these experiments from 13 to 29.

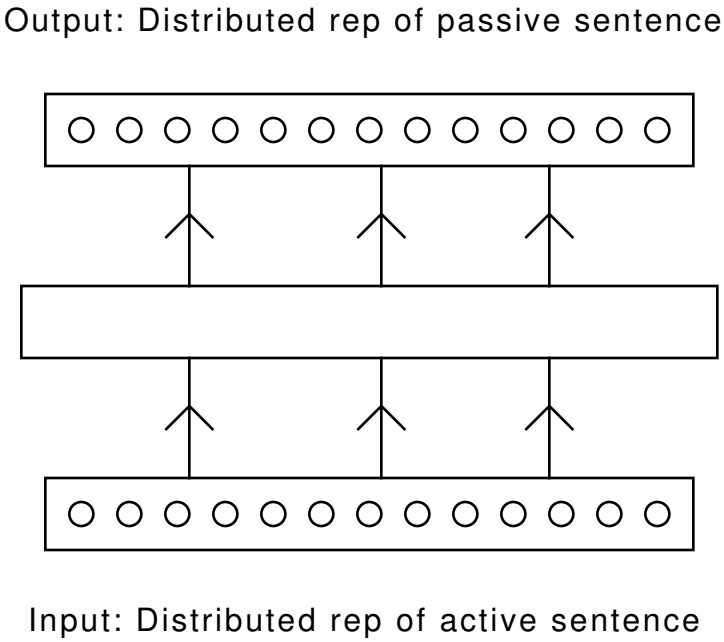

Figure 3. The Transformation network.

4.1 Training the Transformation Network

The same training corpus was used: 40 sentences of "active" form and the 40 corresponding passive sentences. This time, however, these were arranged as 40

input/output pairs. On every training cycle, input to the Transformation network was the encoded RAAM representation of an active sentence; the desired output of the network was the encoded RAAM representation of the corresponding passivized sentence.

In the main set of experiments, 13 units were used in the hidden layer. The learning rate was 0.1, and the momentum rate 0.9. The network was trained for 1500 epochs, consisting of 40 cycles each. At the end of this time, only one output unit (out of 1560 per epoch) had an error greater than 0.05. The maximum error was 0.067.

4.2 Testing the Transformation Network

As an initial test, the 40 "active" sentences from the training corpus were encoded using RAAM, and fed to the Transformation network, yielding a new distributed representation. These representations were decoded using RAAM (with the usual tests). All 40 of these decoded to the correct passive sentence.

As a test of generalization, 40 new input/output pairs of sentences were used (the same 40 as in the testing corpus from Part 3). Again, RAAM-encoded distributed representations of the "active" sentences were fed to the Transformation Network, yielding new distributed representations, which could be decoded to sentences. Out of these 40, 26 decoded to the correct passivized sentence; of the remaining 14, one had incorrect sentence structure, and the other 13 had a single incorrect word.

This 65% generalization rate was better than expected. It indicates that the network has developed a high degree of sensitivity to the structure encoded implicitly in the distributed representations.

4.3 Variations

The performance of the Transformation network may be even better than the 65% generalization rate indicates. This is because a large number of the errors may be due to decoding mistakes by the RAAM network, rather than mistakes in transforming the distributed representations. This phenomenon arises because we are testing the Transformation networks on sentences that were not in the training corpus for the RAAM network.

To test this hypothesis, the Transformation network was retrained, this time with a training corpus consisting of 20 active/passive pairs from the original training corpus of the RAAM network, and 20 active/passive pairs from the original testing corpus. This training proceeded as in Part 4.1.

The results of this training were then tested on the 20 other active/passive pairs from the original RAAM training corpus, which were not in the training corpus for the Transformation network. Generalization on these 20 sentences was perfect — all 20 transformed distributed representations decoded to the correct passive form of the sentences.

When the 20 active/passive pairs that were in the Transformation training corpus but not in the RAAM training corpus were tested, 12 out of 20 decoded correctly. Exactly the same results were achieved for the final 20 active/passive pairs, which were in the training corpus for neither network.

This seems to confirm the hypothesis that most of the original generalization errors were due to incorrect generalization by the RAAM decoding process, rather than by the Transformation process. The generalization ability of the Transformation network itself appears to be remarkably good.

To eliminate the possibility of RAAM generalization error, a new experiment was run, retraining the RAAM network on all 250 possible sentences (125 active and 125 passive). The Transformation network was then trained on a randomly selected set of 75 of the 125 possible active/passive pairs. Generalization was tested on the other 50 active/passive pairs. All 50 of the transformed distributed representations decoded to the correct passivized sentence, giving a generalization rate of 100%.

Finally, the reverse transformation, from passive to active, was modeled in a similar way. (This was partially to see whether a network could be sensitive to the more intricate structure present in the composition of the passive sentences.) A new Transformation network was trained on the same 75 active/passive pairs, this time taking the representations of the passive sentences as input and being trained to produce representations of the active sentences as output. When tested on the other 50 active/passive pairs, the generalization rate was again 100%.

5 Relationship with existing work

Precursors to this work include the connectionist implementations of *production systems*, including those of Touretzky and Hinton (1988) and of Dolan and Smolensky (1989). Like the current model, these operate on complex structured items (usually triplets of the form (A,B,C)), associating them with new structures, whilst operating at the level of the distributed representation (constituents of the original structures are never explicitly extracted). Unlike the model under discussion, however, these make no attempt at modeling *systematicity*. These models associate certain specific triplets with other specific triplets — (A,B,C) might be associated with (N,P,G) — in a way that is not systematically dependent on the structure present in the triplets. Apart from the use of a limited notion of variable binding, no attempt is made at modeling truly structure-sensitive operations, or at capturing any kind of generalization.

Also worth mentioning is an earlier connectionist model of syntactic transformations by Touretzky (1986). This used an architecture based upon the Boltzmann machine to transform parse trees of active sentences into those of the corresponding passive sentences. It operated by first extracting the atomic constituents of the original tree (using implementations of the "car" and "cdr" operators) and then recombining them into the new tree. Unlike the model described in this paper, therefore, this was a direct implementation of a Classical algorithm.

6 Discussion

The above experiments have established that the distributed representations formed by RAAM are well-suited for structure-sensitive operations. Not only is compositional structure *encoded* implicitly in a pattern of activation, but this implicit structure can be *utilized* by the familiar connectionist devices of feed-forward/backpropagation in a meaningful way. Such a conclusion is by no means obvious *a priori* — it might well have turned out that the the structure was "buried too deeply" to be directly used, and that all useful processing would have to proceed first through the step of extraction. That this turns out not to be the case bodes well for the connectionist exploration of natural language processes. It suggests that connectionism might be able to offer some ideas that are in an important sense, *new*.

These results about the processing of implicit structure form a direct counterexample to a recent argument by Fodor and McLaughlin (1990). Arguing against connectionist compositional representations, Fodor and McLaughlin claim that to support structure-sensitive processing, representations of compositional structure must contain explicit tokens of the original constituent parts. If a representation of "John loves Michael" is not a concatenation of tokens of "John", "loves", and "Michael", it is argued, then later processing cannot be sensitive to the compositional structure that is represented. The results presented here show that this conclusion is false. In the distributed representations formed by RAAM, there is no such explicit tokening of the original words. (A detailed analysis of the representations is yet to be performed, and would be a valuable next step. But even on a cursory analysis, it is plain that the RAAM representations do not have the kind of concatenative structure that Fodor and McLaughlin require.) Nevertheless, the representations support systematic processing. *Explicit* constituent structure is not needed for systematicity; implicit structure is enough.

These experiments fall squarely under the rubric of *limitivist symbol-processing*. This connectionist system is certainly not doing away with the idea of symbols altogether, and there are approximate law-like regularities to its performance. But it is impossible to describe the operation of this system, at any level of functional abstraction, as an implementation of a pure symbolic process. The compositional representations used here are not operated on *reductionistically*, by splitting into their constituent parts and then processing. Rather, the operations are direct and *holistic*.

The experiments outlined above demonstrate the possibility of such holistic structure-sensitive operations, but they by no means exploit the full potential of these. After all, syntactic transformations are handled very easily by pure symbolic systems. The real potential of holistic operations arises because of a special property of connectionist representations: they can carry their own content, or at least part of it. In Hinton's (1988) terms, such representations are *reduced descriptions* of the original objects, that go beyond the atomic, content-free representations that are the foundation of symbolic models. The information carried in patterns of activation can be used not only for composition and extraction, but also for semantically meaningful operations. It would be very interesting to see connectionist processes that are structure-sensitive while simultaneously utilizing the kind of content-dependent pattern association for which connectionist networks are renowned. This is the promise that is laid open by implicitly-structured connectionist representations, and it is a promise that could lead to radically new ideas about natural language processing.

References

Dolan, C. P., and Smolensky, P. (1989). Tensor Product Production System: a modular architecture and representation. *Connection Science*, 1: 53-68.

Elman, J. L. (in press). Structured representations and connectionist models. In Gerald Altmann (ed.), *Computational and Psycholinguistic Approaches to Speech Processing*. New York: Academic Press.

Fodor, J. A., and McLaughlin, B. P. (1990). Connectionism and the problem of systematicity: Why Smolensky's solution doesn't work. *Cognition*, 35: 183-204.

Fodor, J. A., and Pylyshyn, Z. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28: 3-71.

Hinton, G. E. (1988). Representing part-whole hierarchies in connectionist networks. In *Proceedings of the Tenth Annual Conference of the Cognitive Science Society*. Montreal, Canada, pp. 48-54.

Pollack, J. B. (1988). Recursive auto-associative memory: Devising compositional distributed representations. In *Proceedings of the Tenth Annual Conference of the Cognitive Science Society*. Montreal, Canada, pp. 33-39.

Pollack, J. B. (in press). Recursive distributed representations. *Artificial Intelligence*.

Rumelhart, D. E., and McClelland, J. L. (1986). On learning the past tense of English verbs. In Rumelhart and McClelland (eds.), *Parallel Distributed Processing*, Vol. 2, pp. 216-271. Cambridge, MA: MIT Press.

Smolensky, P. (in press). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*.

Touretzky, D. S. (1986). Representing and transforming recursive objects in a neural network, or "Trees *do* grow on Boltzmann machines." In *Proceedings of the 1986 IEEE Conference on Systems, Man and Cybernetics*. Atlanta, GA.

Touretzky, D. S. and Hinton, G. E. (1988). A distributed connectionist production system. *Cognitive Science*, 12: 423-466.

Van Gelder, T. (1990). Compositionality: A connectionist variation on a Classical theme. *Cognitive Science*, 14.