Psychological Review
2009, Vol. 116, No. 4, 986–997

0033-295X/09/$12.00 DOI: 10.1037/a0017097

COMMENTS

A Fundamental Limitation of the Conjunctive Codes Learned in PDP Models of Cognition: Comment on Botvinick and Plaut (2006)

Jeffrey S. Bowers and Markus F. Damian
University of Bristol

Colin J. Davis
Royal Holloway, University of London

A central claim shared by most recent models of short-term memory (STM) is that item knowledge is coded independently from order in long-term memory (LTM; e.g., the letter *A* is coded by the same representational unit whether it occurs at the start or end of a sequence). Serial order is computed by dynamically binding these item codes to a separate representation of order. By contrast, Botvinick and Plaut (2006) developed a parallel distributed processing (PDP) model of STM that codes for item-order information conjunctively, such that the same letter in different positions is coded differently in LTM. Their model supports a wide range of memory phenomena, and critically, STM is better for lists that include high, as opposed to low, sequential dependencies (e.g., bigram effects). Models with context-independent item representations do not currently account for sequential effects. However, we show that their PDP model is too sensitive to these effects. A modified version of the model does better but still fails in important respects. The successes and failures can be attributed to a fundamental constraint associated with context-dependent representations. We question the viability of conjunctive coding schemes to support STM and take these findings as problematic for the PDP approach to cognition more generally.

Keywords: symbols, conjunctive coding, connectionism, short-term memory

A number of neural network models of short-term memory (STM) have been developed in recent years (e.g., Botvinick & Plaut, 2006; Brown, Preece, & Hulme, 2000; Burgess & Hitch, 1999; Grossberg & Pearson, 2008; Page & Norris, 1998). Most of these models are concerned with one specific manifestation of STM, namely immediate serial recall, in which participants attempt to repeat a set of items (e.g., letters, numbers, words) in the same order. The average person can report 7 ± 2 items (the so-called magic number 7; Miller, 1956), although STM might actually store 4 ± 1 items (Cowan, 2001). If a person is unable to rehearse the items, the items are quickly lost (Baddeley, 1986).

Jeffrey S. Bowers and Markus F. Damian, University of Bristol, Bristol, England; Colin J. Davis, Department of Psychology, Royal Holloway, University of London, Egham, England.

The programs used to generate study and test lists and the programs used to analyze the outputs of the model are posted at http://psychology.psy.bris.ac.uk/people/jeffbowers.htm. The model itself can be downloaded from Matthew Botvinick's website at http://www.princeton.edu/~matthewb/.

We would like to thank Simon Farrell, Clive Frankish, John Hummel, and Klaus Oberauer for helpful suggestions and comments that helped us formulate some of the key ideas in this article. We would also like to thank Derek Besner, Matthew Botvinick, Max Coltheart, and Stephan Lewandowsky for comments on an earlier version of the article that greatly improved it. Finally, we would like to thank Matthew Botvinick for his advice when we had some difficulties with our simulations.

Correspondence concerning this article should be addressed to Jeffrey S. Bowers, Department of Experimental Psychology, University of Bristol, 12A Prior Road, Clifton, Bristol BS8-ITU, England. E-mail: j.bowers@bris.ac.uk

A key insight that has guided recent theorizing is that STM cannot be based on item-to-item associations. On some earlier models, the sequence *ABCDE* might be stored by developing an association between *A* and *B*, *B* and *C*, *C* and *D*, etc., such that *A* retrieves *B*, which in turn activates *C*, etc. (Lewandowsky & Murdock, 1989; Wickelgren, 1966). Although these so-called chaining models can support immediate serial recall, there is now a general consensus that these mechanisms do not underpin human performance, for a variety of reasons. For example, transpositions are a common type of error (mistakenly recalling the sequence *ABDCE*, with *D* and *C* transposed), whereas, according to a chaining model, transpositions should be rare. For a review of a variety of findings that pose a challenge for chaining models, see Botvinick and Plaut (2006).

The most common response to the deficiencies of chaining models has been to develop models that rely on *context-independent* (in this case, position-independent) item representations in long-term memory (LTM). For example, the LTM representation of the letter *A* is the same when it occurs at the beginning or the end of a study list, and there is no association between *A* and any other letter in a to-be-remembered list. The position of items is coded by transient associations between the items and a separate representation that assigns items a position. In the case of the primacy model (Page & Norris, 1998), position is coded as a decreasing level of activation across items. For example, the sequence *ABCD* is coded with the *A* unit being the most active, *B* second most active, etc., and the sequence *DCBA* would be coded with the same set of letter units but with *D* the most active, *C* the next most active, etc. (see also Grossberg, 1978; Grossberg &

Pearson, 2008). In other models, short-term connection weights (as opposed to activation values) are used to link items to positions (e.g., Brown et al., 2000; Burgess & Hitch, 1999). The reliance on context-independent representations allows these models to overcome the limitations of chaining models. For example, the sequences *ABCD* and *ACBD* are quite similar given that *B*-in-Position-2 and *B*-in-Position-3 are coded with the same *B* unit (and the same two *C* units are used in the two lists), and this similarity leads to transposition confusions in STM.

Recently, Botvinick and Plaut (2006) developed a parallel distributed processing (PDP) model of immediate serial recall that also addresses the limitations of chaining models. The model includes a set of (localist) input and output units and an intervening set of hidden units that map between them (see Figure 1). As can be seen in Figure 1, the hidden units include feedback (recurrent) connections to themselves, and the hidden units are bidirectionally associated with the output layer. The connection weights constitute the LTM of the model, and the activation pattern across the units constitutes the model's STM, with the recurrent connections ensuring that the activation persists in the absence of input.

The key finding reported by Botvinick and Plaut (2006) is that the trained model is able to support STM relying on learned *context-dependent* item representations in LTM. That is, it develops representations that code for items and order conjunctively. For instance, the letter string *ABC* would be coded by coactivating distributed representations of *A*-in-Position-1, *B*-in-Position-2, and *C*-in-Position-3. The model is able to explain a range of STM phenomena, including findings that have posed a problem for chaining models.

At the same time, the model captures another key result in the literature that has proved to be a challenge for models with context-independent representations, namely, the finding that STM is sensitive to background knowledge of sequential structure. For example, strings of letters are better recalled if adjacent items in the list frequently co-occur in English words—the so-called bigram frequency effect (Baddeley, 1964). The reason that the Botvinick and Plaut model is sensitive to sequential structure is that the model learns not only context-dependent representations (e.g., *A*-in-Position-1) but also associations between these representations of items (e.g., *A*-in-Position-1 → *B*-in-Position-2). So, for instance, if *A* and *B* often occur in sequence during training, links are developed that associate *A*-in-Position-1 with *B*-in-Position-2, facilitating the transition between these representations. The complex associations that develop between conjunctive codes over a million training trials ensure that the model becomes sensitive to the sequential dependencies that occurred during training. Nevertheless, the model is not a chaining model: These links code for the entire history of training rather than by item-by-item associations that occur during a specific memory trial, and more importantly, the items themselves (independent of the links) can support recall. That is, even in the absence of any associations between items, the sequence *ABC* could be recalled by virtue of the coactive position-dependent letter units *A*-in-Position-1, *B*-in-Position-2, and *C*-in-Position-3. The learned associations only bias performance.

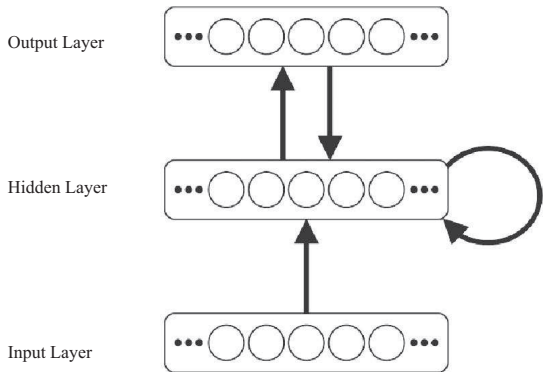

Figure 1. Diagram of the Botvinick and Plaut (2006) recurrent parallel distributed processing model of immediate serial recall. The model includes a set of 27 input and output units (one for each letter of the alphabet plus a unit in the input layer that cues recall and a unit in the output layer that codes end of list) plus a set of 200 hidden units. Arrows indicate connections between and within layers. Adapted from "Short-Term Memory for Serial Order: A Recurrent Neural Network Model," by M. M. Botvinick and D. C. Plaut, 2006, *Psychological Review, 113,* p. 204. Copyright 2006 by the American Psychological Association.

Botvinick and Plaut (2006) took the model's sensitivity to sequential structure as strong evidence that STM relies on conjunctive item–position representations, contrary to the assumption in many alternative accounts. However, we challenge this claim in the present commentary. One of the standard criticisms of the PDP framework is that context-dependent (conjunctive) codes in LTM limit generalization in a variety of cognitive and perceptual domains (e.g., Bowers, 2002; Davis, 1999; Fodor & Pylyshyn, 1988; Marcus, 1998; Pinker & Prince, 1988). This suggests that the Botvinick and Plaut model may be too sensitive to the sequential structure of the training lists and fail to recall various types of untrained sequences. In line with this analysis, we show that the model is constrained in ways that alternative models of STM (and humans) are not. We take these findings to challenge Botvinick and Plaut's model of STM and the PDP approach in general.

The rest of this commentary is organized as follows. First, we set the stage by outlining the types of novel letter sequences that might prove difficult for the model to recall (given its reliance on context-dependent item representations). Second, we report a series of seven simulations that test the original Botvinick and Plaut model as well as a modified (fully distributed) version of the model on these sequences. We show that the original model fails on lists that humans should find trivial. The modified model does better, but it nevertheless fails to recall sequences that humans could recall (and that are within the capacities of alternative models). Finally, we consider the implications of these findings for PDP theories of STM and cognition more generally.

Generalization and Context-Dependent Representations

A core claim of the PDP approach is that all knowledge is coded in a context-dependent manner (cf. McClelland, Rumelhart, & The PDP Research Group, 1986). According to critics of this approach, networks that rely on context-dependent representations cannot support various forms of generalization that humans routinely perform. For instance, Davis (1999; see also Davis & Bowers, 2004, 2006) highlighted the limitations of models of word identification and naming that code for letter identity and letter position using context-dependent letter codes (e.g., *A*-in-Position-1, *B*-in-

Position-2). This includes all PDP theories as well as many localist ones (e.g., Coltheart, Rastle, Perry, Langdon, & Ziegler, 2001; McClelland & Rumelhart, 1981). The problem is that the models fail to identify familiar words in unfamiliar contexts (e.g., *POLE* in *CATPOLE*) because the familiar words are coded in a novel way (e.g., the letters *P*-in-Position-5, *O*-in-Position-6, *L*-in-Position-7, and *E*-in-Position-8 have not been associated with the familiar word *POLE*). Similar constraints are claimed to apply to PDP models designed to generalize to new morphological forms (Pinker & Prince, 1988), semantic and syntactic relations (Fodor & Pylyshyn, 1988; Marcus, Vijayan, Bandi Rao, & Vishton, 1999), sequential behavior (Cooper & Shallice, 2006), analogies (Hummel & Holyoak, 1997), etc.

In contrast to the longstanding debate regarding the extent to which PDP models can successfully accommodate human generalization in the domains of language, thought, and behavior, this issue has not been given much consideration in the case of STM. Botvinick and Plaut (2006) did show that their model can recall untrained sequences. For example, in their first simulation, the model was trained on approximately 1 million random letter sequences (of various lengths) and was then tested on random strings of six letters. The test sequences were almost always novel (99.3% of the time), and the model succeeded approximately 50% of the time, which is similar to human performance. Nevertheless, all these novel sequences included letters that were repeatedly trained in all positions (e.g., *A* was presented in Position 1, as well as Position 2, etc., multiple times), and all bigrams were repeatedly trained in all positions. Indeed, given 1 million training trials with list length stepping up from one to nine items, each possible bigram will be presented approximately 6,150 times and each possible trigram approximately 200 times.

A critical question, then, is whether Botvinick and Plaut's model of STM can support recall for lists that contain familiar items in novel positions or novel bigrams of familiar items. It seems unlikely that human performance is constrained in this way. For example, if you are taught the new word *BLAP*, then it is self-evident that you can recall it when it is presented in Position 1, 2, or 3 in a list (even if it was never trained in Position 2 or 3). Similarly, read the following list of words and then try to recall them in sequence: *TEDDY COGNITION BLUEBERRY SYNAGOGUE MATTRESS STIMULI*. If you succeed in recalling all (or most) of the words, you have accomplished this despite the fact that you have never seen any of the adjacent word pairs before. Indeed, as of August 10, 2009, these word pairs never co-occur in the entire search space of Google. Type "*TEDDY COGNITION*" (in quotation marks), and you get zero hits.

The main question we addressed in the following simulations was whether PDP models of STM can recall familiar letters when they are presented in novel positions within a list or in novel bigram contexts.

Testing the Original Botvinick and Plaut Model of STM (Simulations 1–3)

Simulation 1

As a first step, we attempted a replication of Simulation 1 from the Botvinick and Plaut (2006) article. This model includes a set of 27 input and output units (one for each letter of the alphabet plus a unit in the input layer that cues recall and a unit in the output layer that codes end of list) plus a set of 200 hidden units. We followed the same training regime and assessed performance on six-item lists when the to-be-remembered items were randomly selected (without replacement). We then tested the model after 1 million training trials, a level of training that led the model to closely mirror human performance in terms of overall accuracy. Furthermore, to determine whether the similar overall performance of the model and human behavior reflects an intrinsic memory capacity of the model, we continued training for 5 million trials. Performance of the model was assessed at intervals of 1 million training trials.

As is clear from Figure 2, we replicated Botvinick and Plaut's (2006) Simulation 1, with performance approximately 40% for six-item lists following 1 million training trials. Performance continued to improve with training, so that after 5 million trials, recall accuracy was approximately 100% for six-item lists and approximately 60% for nine-item lists. Thus, the close match between human memory capacity and the overall performance of the model reported by Botvinick and Plaut should not be given too much weight. Although it is impressive that the model can support the human performance, it could also simulate any level of performance between zero and perfect accuracy.

Simulation 2

In our first test of the model's ability to generalize more broadly, we tested it on sequences in which some letters were excluded from specific positions. We trained the Botvinick and Plaut model in the same way as above except that we ensured that four letters never occurred in specific positions, namely, *B*-in-Position-1, *D*-in-Position-2, *G*-in-Position-3, and *J*-in-Position-4. When a random list was generated in which one (or more) of these letters occurred in these positions, we simply eliminated that list and generated another sequence. At recall we tested the model on lists of 1,000 six-letter sequences when zero (baseline), one, two, three, or all four of the critical untrained letter–position combinations were included in the list. We varied which specific letters were

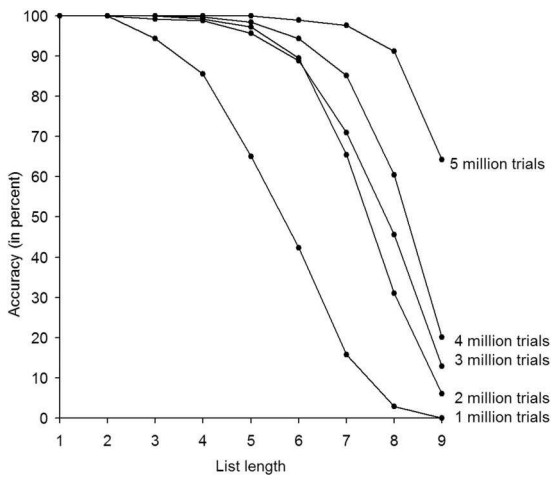

Figure 2. Performance of the Botvinick and Plaut (2006) model on letter lists of various lengths after various amounts of training.

excluded in different lists. For instance, when a single untrained letter was presented at test, we included 1,000 trials in which *B*-in-Position-1 was tested, 1,000 trials in which *D*-in-Position-2 was presented, etc. Similarly, when two critical untrained letters were presented at test, we included 1,000 trials in which *B*-in-Position-1 and *D*-in-Position-2 were tested, 1,000 trials in which *D*-in-Position-2 and *G*-in-Position-3 were tested, etc. Other than this restriction, the sequence of the six letters in the lists was random at test. This ensured that the model was trained on all bigrams of letters. That is, *BD*, *DG*, and *GJ* were studied but not in Positions 1–2, 2–3, and 3–4, respectively.

The results (averaging across the specific test letters that were included) can be seen in Figure 3. As is clear from this figure, the model's performance was severely impaired on lists that included the untrained letter–position combinations. Indeed, when the model was tested on lists that contained all four of the untrained critical letter–position combinations, the model achieved less than 1% correct.

Inspection of the erroneous responses revealed that the model frequently made anticipatory errors at recall. For example, when tested on lists with *B*-in-Position-1, the model recalled the second letter in the sequence in the first position 57.4% of the time. That is, the model's response was captured by trained sequences that were similar to the presented sequence. In the test sequence *BJRSCQ*, *J* is output as *J*-in-Position-1 because *J* in the first two positions is coded in a similar way in the hidden layer of the model (Botvinick & Plaut, 2006, argued that this similarity plays a critical role in inducing transposition errors in their model and human performance), and *J*-in-Position-1 has been trained.

It is also interesting to note that the model often perseverated after making an anticipation. For example, when the model was tested on lists with *B*-in-Position-1, anticipations were followed by perseverations 92.9% of the time. If the model was presented with the sequence *BCQRTF*, it was quite likely to respond *CCQRTF* (note that the perseverative response, the second *C*, is correct). Perseverations are rare in human errors (Henson, 1996), and Botvinick and Plaut (2006) highlighted the finding that their model captured this constraint on errors. But clearly this is not the case when the model is tested on sequences of untrained bigrams.[1]

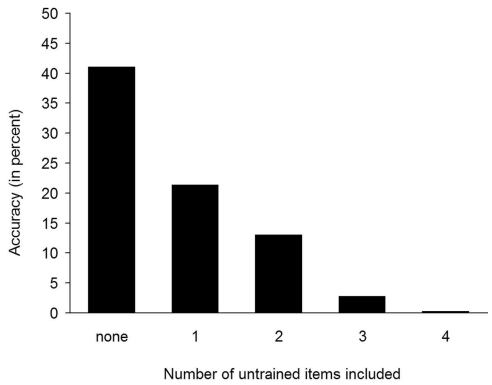

Figure 3. Performance of the Botvinick and Plaut (2006) model when tested on sequences that included zero (baseline), one, two, three, or four letters in a given position that were not included at training. The baseline condition assessed the model's memory when tested with trained sequences; the latter conditions assessed memory on untrained sequences. The untrained letter–position combinations were *B*-in-Position-1, *D*-in-Position-2, *G*-in-Position-3, and *J*-in-Position-4.

Simulations 3a–3c

In Simulations 3a–3c we trained the Botvinick and Plaut model in the same way as above except that we eliminated a small set of bigrams (from one to three) from the training lists. Specifically, we eliminated the bigram *D–G* in Simulation 3a, the bigrams *D–G* and *G–J* in Simulation 3b, and the bigrams *B–D*, *D–G*, and *G–J* in Simulation 3c. When these bigram sequences were randomly generated in a list for training, we simply eliminated that list and generated another sequence. At recall we tested the model on 1,000 pseudorandom lists of six letters in one of two conditions: (a) None of the lists included the untrained bigrams, or (b) all the lists included the untrained bigrams.

Not surprisingly, performance in Condition A was similar in Simulations 3a–3c; as in all cases, the test lists were restricted to sequences of trained bigrams.[2] By contrast, performance was much poorer for sequences in Condition B. With one untrained bigram, performance dropped by approximately 25%, and with three untrained bigrams performance was less than 1% (see Figure 4).

It should be emphasized that one of the key findings that Botvinick and Plaut (2006) presented in support of their approach is that their PDP model shows the appropriate sensitivity to bigram frequencies (in accordance with human performance), whereas models relying on context-independent representations do not unless additional assumptions are built in. However, it turns out that this putative advantage is actually a flaw. That is, the Botvinick and Plaut model is too reliant on sequential dependencies. Human performance is sensitive to, but transparently not reliant on, sequential dependencies; STM extends to word sequences that have never been observed, as in the *TEDDY COGNITION* illustration above.

Once again, it is interesting to note the type of errors the model made. In many cases the model's response indicates that it was captured by a trained sequence at the expense of the untrained test sequence, which often manifested itself as an anticipation (30.9% of the time). For example, when the sequence *DGJ* was excluded during training, it was nevertheless the case that the sequence *DJ* was trained. As a consequence, the model often responded with the

[1] A surprising result is that the performance of the model was sensitive not only to the number of untrained letters that it was tested on but also to which untrained letters. Performance was always poorest when the test sequences included an untrained letter in the first position of the list. Indeed, when the model was presented with the single untrained item *B*-in-Position-1 at test, performance was at 5.3%. We are unclear why the model fails most dramatically for untrained items at the start of the list.

[2] For the simulations contrasting unconstrained with constrained learning environments, such as Simulations 3a–3c, we initially had difficulty in matching baseline performance when the model was given different training histories. The difficulty arose because the model exhibits a cyclical pattern of performance, with overall performance improving and declining quite dramatically within only a few cycles. We therefore trained the various versions of the model up to 1 million trials and then continued training in steps of 10 cycles until baseline performance matched as closely as possible across conditions (generally within 1%–2%; see Figures 4 and 6). It is unclear to us why performance of the model is cyclical.

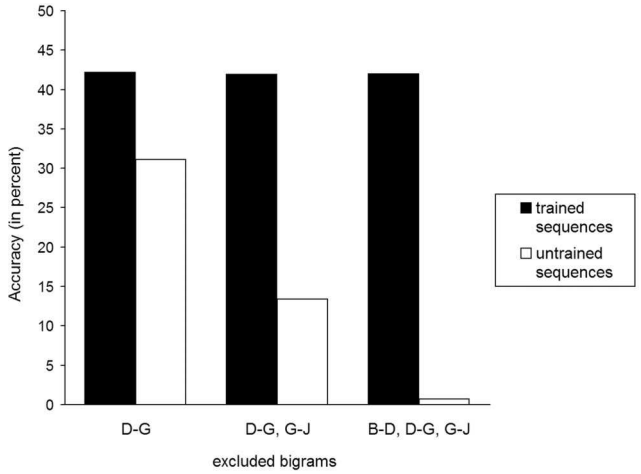

Figure 4. Performance of the Botvinick and Plaut (2006) model when training excluded one, two, and three bigrams. Performance was assessed on lists that either included the untrained bigrams or excluded the untrained bigrams. The untrained bigrams were *B–D*, *D–G*, and *G–J*.

sequence *DJ* when presented with *DGJ*. This is similar to the types of errors that Cooper and Shallice (2006) reported when they tested a PDP recurrent model of sequential behavior (Botvinick & Plaut, 2004). As in our analysis, the errors of this recurrent network were determined by the training history, with correct performance strongly biased toward trained sequences.[3]

Testing a Fully Distributed Botvinick and Plaut Model (Simulations 4–7)

Although the Botvinick and Plaut model is challenged by Simulations 2 and 3, it is possible that its failure is a by-product of the choice of input coding scheme for the model. In all their simulations, Botvinick and Plaut included a set of localist units in the input and output layers (localist letter units in their Simulations 1–3, localist pseudoword units in their Simulation 4), with distributed representations limited to the hidden layer of their model. This feature of their model was not considered critical, and Botvinick and Plaut (2006) made it clear that the localist codes were included only for convenience. However, there is some reason to think that the localist coding schemes played a role in the model's failures. In Simulation 2, the exclusion of sequences with *B* at the start ensured that input and output *Unit* 2 (that codes for *B*) was never activated at the beginning of a sequence. Similarly, the exclusion of the sequence *BD* in Simulation 3 ensured that the corresponding input–output Units 2 and 4 were never activated in sequence. These restrictions on the activation of the input–output units may have played a key role in constraining the model's performance.

These generalization problems might be overcome if the model included distributed input and output units. Consider a case in which the letters are coded as a distributed pattern of activation across a collection of input and output units. In this situation, the exclusion during training of *B* at the start of the list, or *B* followed by *D*, does not eliminate the activation on Node 2 at the beginning of the list or the activation of Node 2 followed by Node 4. For instance, if *B* is coded by the pattern {0, 1, 1, 0} over four units, *D* as {0, 0, 1, 1}, and *F* as {1, 1, 0, 0}, and *B* never occurs first during training, it is nevertheless the case that Node 2 is activated at the beginning of the list whenever *F* is presented (the second unit is active in the distributed pattern for *F*). Similarly, if *D* never follows *B* at training, it is nevertheless the case that Node 4 is activated following Node 2, by virtue of *D* following *F*. That is, the inclusion of distributed representations at the input and output layers ensures that all input and output nodes are activated in all positions and sequences, even when a letter in a given position or a specific bigram sequence was untrained. It seems possible, then, that a fully distributed Botvinick and Plaut model would generalize more broadly.

Simulation 4

We first assessed whether the modified version of the Botvinick and Plaut model with distributed input representations would succeed when training and test lists of letters were generated randomly. The architecture and processing assumptions of the model were unchanged, and we simply modified the nature of the inputs and outputs to the model. In particular, instead of representing each letter through the activation of one unit, we represented a letter over five units. That is, in addition to coding *A* with Unit 1, *B* with Unit 2, etc., we assigned each letter four additional units; for example, the letter *A* was coded by the Units 1, 3, 12, 20, and 22. The consequence of this coding was that each letter was uniquely defined as a pattern of activation over five units, with each unit being involved in the coding of five letters. Each letter was randomly assigned five units, with the restriction that each unit was involved in coding five letters.

We followed the same training regime as in Simulation 1, and accuracy at recall was determined by comparing the output of the model with the distributed representations of all 26 letters and selecting the letter with the highest cosine similarity. This is similar to selecting the most active letter in a localist coding scheme. After 1 million trials, the model was correct on 51.5% of six-item lists. This performance is roughly comparable to the localist model under the same conditions.

Simulation 5

In Simulation 5 we replicated Simulation 2 with the distributed input and output coding schemes. The training phase was the same

[3] The fact that errors were often the product of the model anticipating letters that were trained raises the possibility that performance would improve if we removed the bigrams that incorrectly captured the model's response. That is, if we removed the bigram *D–J* in the above example, then it should be less likely that the model would incorrectly respond *DJG* to the input *DGJ*. Out of interest, we trained the model by removing the first-order bigrams (such as *D–G* and *G–J*), as above, but also second-, third-, and fourth-order bigrams (as relevant). For instance, if the sequence *BDFGJ* was excluded from training, we eliminated not only *B–D*, *D–F*, *F–G*, and *G–J* but also the second-order bigrams *B–F*, *D–G*, and *F–J*; the third-order bigrams *B–G* and *D–J*; and the fourth-order bigram *B–J*. We did this across all positions in the list. It is interesting to note that the model's performance did improve somewhat but was still far from satisfactory. Its performance was 20.0%, 10.7%, and 3.1% when two, three, and four bigrams (plus their corresponding higher order bigrams) were removed, respectively.

as in Simulation 2. That is, we excluded the letter *B*-in-Position-1, *D*-in-Position-2, *G*-in-Position-3, and *J*-in-Position-4 during training and then tested the model on 1,000 six-letter lists when zero (baseline), one, two, three, or all four of the critical letters were included. As before, we also varied the specific letters that were included at test and averaged performance across items.

The results can be seen in Figure 5. Clearly, the model did much better in all conditions. Indeed, when tested on sequences that included all four untrained letters, the model achieved 44.7% correct, compared with less than 1% for the localist model.

Simulations 6a–6c

Simulations 6a–6c were a replication of Simulations 3a–3c except that distributed input and output letter codes were used. As before, the model was trained on sequences that had either one, two, or three bigrams removed (Simulations 6a, 6b, and 6c, respectively). Once again, we tested the model on 1,000 six-letter lists in one of two conditions: Either none of the lists included the untrained sequences, or all the lists included the untrained sequences. Accuracy at recall was determined in the same way as in Simulation 5.

As can be seen in Figure 6, the model performed well in both conditions. That is, unlike the localist model tested in Simulation 3, the distributed model generalizes to untrained bigram sequences. Thus, it is clear that a PDP model can support memory for lists of letters even when they contain letters in untrained positions (Simulation 5) or sequences of untrained bigrams (Simulation 6). Accordingly, the problem with the Botvinick and Plaut model might appear to be superficial—it is simply that localist coding schemes were employed at the input and output layers, and this restricted generalization.

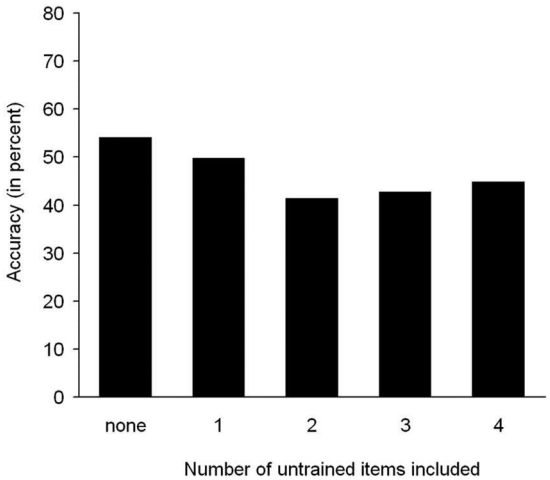

Figure 5. Performance of the modified Botvinick and Plaut (2006) model with distributed representations when tested on sequences that included zero (baseline), one, two, three, or four letters in a given position that were not included at training. The baseline condition assessed the model's memory when tested on trained sequences; the latter conditions assessed memory on untrained sequences. The untrained letter–position combinations were *B*-in-Position-1, *D*-in-Position-2, *G*-in-Position-3, and *J*-in-Position-4.

Simulation 7

In the final simulation we tested the modified model on a more extreme form of generalization. We trained the model on a set of letters that were free to occur in any position and in any order (as in Simulation 4) and then introduced a new target letter that was trained only in Position 1. At test the model was tested with sequences that included the target letter in initial or noninitial positions. This is analogous to Simulation 5 in which we tested the model on lists that contained letters in untrained positions, but here the model is being asked to generalize more extensively. That is, the model is familiar with the target letter only in Position 1, and it is being asked to generalize to all other positions. In Simulation 5 the model succeeded in recalling a target letter in an untrained position (e.g., recalling *B*-in-Position-1), but this was after it was trained on the letter in all other positions.

We first trained the model on 25 of the 26 distributed letter patterns used above for 1 million trials (we excluded the letter *R*, which corresponds to Units 9, 11, 13, 14, and 25). After training, STM performance on six-letter lists composed of these letters was 54.7%, similar to the model's performance when trained on all 26 letters in Simulation 5. We then trained the model on all 26 letters but allowed the target letter *R* to appear only in Position 1. That is, lists were randomly generated, and we eliminated any lists in which *R* appeared in Positions 2–6. We assessed the model's performance on lists that contained *R* in Positions 1–6 before any additional training (before the model was trained with *R*) and following 100,000, 200,000, 300,000 and 1 million additional training trials. For each test position of *R*, the model was presented with 1,000 lists of six letters, with *R* fixed in one position and the letters in the other position randomly.

The results are presented in Figure 7. Before any additional training, the model performed near floor on lists that contained *R* in any position (i.e., the model was unable to recall lists that contained a novel letter), and with additional training, performance selectively improved in Position 1. Following an additional 300,000 training trials, the model performed 50.2% (similar to performance on lists that did not contain *R* following the initial training), and following 1 million trials, the model correctly recalled 97.6% of the lists with *R* in Position 1. By contrast, the model catastrophically failed on lists that contained *R* in all other positions regardless of the amount of training, ranging from 0.3% to 3.4%.

To provide a more sensitive test of any learning across positions, we tested the model trained for an extra 1 million trials on 1,000 two-letter lists in one of two conditions: Either the letter *R* was presented first followed by a random letter, or a random letter was presented first followed by the letter *R*. Not surprisingly, performance on two-item lists that started with *R* was perfect (100%). By contrast, accuracy on lists that ended with *R* was only 12.6% (with all the errors on the letter *R*). Clearly, generalization is highly constrained even with the distributed input coding scheme of the modified model. By contrast, it is clear that humans can readily recall the sequences described in Simulation 7. That is, if you learn a new letter name (e.g., *ree*), or a new word (e.g., *BLAP*), or a Zulu click, or a cough that is presented at the start of a list, you can recall that novel item in Position 2. One does not need to conduct an experiment to see whether a participant who recalls the sequence *ree-B* can also succeed on the sequence *B-ree*.

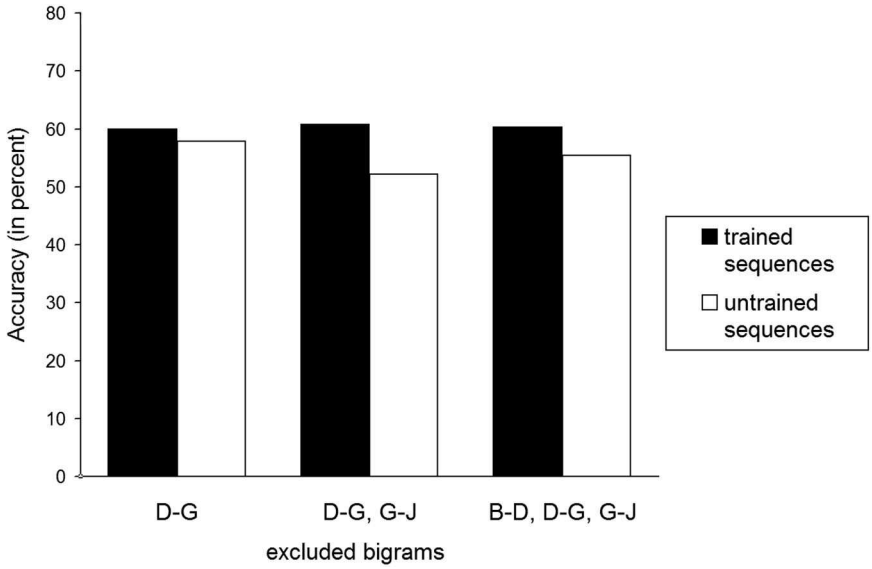

Figure 6. Performance of the modified Botvinick and Plaut (2006) model when training excluded one, two, and three bigrams. Performance was assessed on lists that either included the untrained bigrams or excluded the untrained bigrams. The untrained bigrams were *B–D*, *D–G*, and *G–J*.

Discussion

Botvinick and Plaut (2006) developed a recurrent PDP model that supports STM through learned conjunctive letter–position codes. For example, the letter string *ABC* would be coded by coactivating distributed representations of *A*-in-Position-1, *B*-in-Position-2, and *C*-in-Position-3. The model is able to accommodate a wide range of findings that proved difficult for chaining models of STM and, critically, is sensitive to background knowledge of sequential structure, such as the bigram effect observed in humans (Baddeley, 1964). These latter findings pose a challenge to current alternative models of STM that rely on context-independent item representations in LTM (e.g., Page & Norris, 1998).

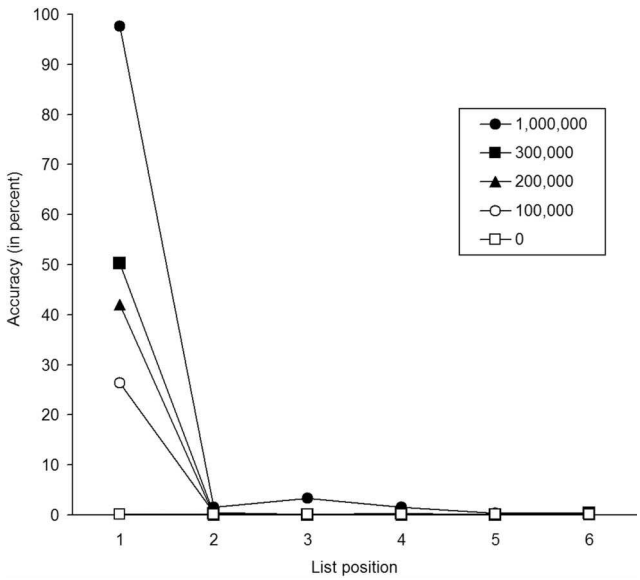

Figure 7. Performance of the modified Botvinick and Plaut (2006) model on lists of six letters that contained the letter *R* in various positions (from 1 to 6). The model was initially trained for 1 million trials on lists that excluded the letter *R* and was then provided additional training (ranging from 0 to 1 million extra training trials) in which *R* was free to occur in Position 1 but in no other position.

However, we have shown that the original Botvinick and Plaut model is excessively sensitive to background sequential structure. That is, the model catastrophically failed when tested on letter sequences composed of letters in untrained positions (Simulation 2) and when tested on letter sequences composed of untrained bigrams (Simulation 3). By contrast, these sequences would pose no problem for humans.

At the same time, we have shown that a modified version of the Botvinick and Plaut model that includes a distributed, as opposed to a localist, input letter coding scheme succeeded in these two conditions (Simulations 5–6). Nevertheless, when we tested the model on a more extreme form of generalization, the model catastrophically failed (Simulation 7). That is, when the model was trained with the letter *R* only in the first position of a list, it was not able to recall sequences that contained the letter *R* in other positions. These types of strings would appear to be well within the capacities of human memory (and alternative models of STM, as discussed below). For example, if you learn a new letter named *ree*, then you can recall that letter in any position within a list (limited only by your memory span)—there is no need to be trained on the letter *ree* in all positions.

Although Simulations 2, 3, and 7 identify some serious restrictions in the performance of the original and modified Botvinick and Plaut model, perhaps some other PDP model with a different input–output coding scheme would succeed. Indeed, we have shown that the input coding scheme greatly impacts on the model's ability to recall unfamiliar sequences. Although we cannot rule out this possibility, we are skeptical. Whatever input coding scheme is

adopted, as long as serial recall relies on conjunctive item–position coding within the hidden layer, it seems likely that the modified model would suffer similar limitations and for the same reason; namely, there are no learned units that code for *R*-in-Position-6 when it has been trained only in *R*-in-Position-1. Still, it is always possible that another PDP model of STM that learned a different set of context-dependent codes would succeed in the current contexts. One thing is clear: If PDP models of STM are to provide a viable account of human STM performance, it will be important to show that they can generalize more broadly.

Learning Poses Another Problem for Models of STM That Rely on Context-Dependent Representations

Although we have focused on the claim that generalization is limited in PDP models of STM that rely on context-dependent representations, there is another possible limitation of this approach with regard to learning. The original and modified Botvinick and Plaut models were able to learn a set of 156 letter–position conjunctive codes (26 letters × 6 positions) given the appropriate training. Nevertheless, it took the models approximately 1 million training trials to match human performance in terms of overall accuracy. Given this, consider the difficulty of recalling sequences of words. To recall the list *DOG*, *RAIN*, *MOTHER*, and *FUN*, the model must have a conjunctive representation of *DOG*-in-Position-1, *RAIN*-in-Position-2, *MOTHER*-in-Position-3, and *FUN*-in-Position-4. Because it is possible to recall these words in any order, the model needs separate conjunctive representations for each word (indeed all words) in all positions, such as *DOG*-in-Position-2, *DOG*-in-Position-3, etc. It seems unparsimonious to include separate conjunctive representations for each word in each position in a list. But more importantly, given the amount of training required to learn 156 conjunctive codes for letters, it is unclear whether it is feasible to learn approximately 600,000 conjunctive codes for words (assuming 100,000 words in a person's vocabulary × 6 positions).

One tempting solution would be to argue that memory for lists of words is supported by conjunctive codes of letters (or perhaps familiar syllables), in which letters (or syllables) are coded by both their position within a word (e.g., some form of slot coding typical of PDP models) and their position within a list. This would minimize the number of conjunctive codes that need to be learned to a manageable number. For example, to remember lists of up to six words that vary from one to eight letters, the model would need to learn only 1,248 conjunctive representations (26 letters × 8 possible letter positions within a word × 6 possible words in a list). For instance, the sequence *DOG CAT* could be coded by *D*-in-Position-1-of-a-word-at-beginning-of-list, *O*-in-Position-2-of-a-word-at-beginning-of-list, *G*-in-Position-3-of-a-word-at-beginning-of-list (together that code for *DOG* at the start of a list), *C*-in-Position-1-of-a-word-in-second-position-in-a-list, *A*-in-Position-2-of-a-word-in-second-position-in-a-list, and *T*-in-Position-3-of-a-word-in-second-position-in-a-list (together that code for *CAT* in Position 2 of a list).

Although this later solution might allow a PDP model to learn a set of conjunctive codes that would support STM for a list of words, it would not provide a good account of human performance, for at least three reasons. First, a standard finding in the behavioral literature is that STM is better for words than for nonwords. For instance, Jefferies, Frankish, and Lambon Ralph (2006) found that participants recalled 79% of consonant–vowel–consonant (CVC) words presented in a list of five words, compared with 31% of CVC nonwords presented in a list of five nonwords. This shows that STM relies on lexical knowledge. Any attempt to model STM for words on the basis of conjunctive letter codes (ignoring word knowledge) seems unlikely to succeed. Second, memory for lists of words is relatively insensitive to word length. For instance, Hulme, Suprenant, Bireta, Stuart, and Neath (2004) compared STM for lists of words that were either one or five syllables in length. When long and short words were intermixed, they were recalled at the same level. When long and short words were presented in separate blocks, the short items enjoyed a slight advantage (~12%), but given that five times as many phonemes need to be recalled in the case of the long words, it is clear that STM is not the product of recalling a series of phonemes (or syllables). Rather, memory for words is based on recalling a sequence of word representations. Third, STM is sensitive not only to background knowledge of the sequential structure of letters but also to the sequential structure of words (e.g., Miller & Selfridge, 1951); these dependencies reflect lexical, not sublexical, knowledge. Thus, if PDP models of STM are to be extended to support STM for words, they are going to have to learn a lot of conjunctive codes.

Can Models Relying on Context-Independent Representations Account for Sequential Effects in STM?

Although the current article is primarily concerned with highlighting a limitation of PDP models of STM, it is worth considering whether Botvinick and Plaut's (2006) main criticism of alternative models is justified. That is, they claimed that models that rely on context-independent representations do not naturally account for the fact that STM is sensitive to background knowledge of sequential structure, such as bigram effects. Nevertheless, they highlighted one possible mechanism by which future models of this sort might account for the effects, citing the work of Lee and Estes (1981). That is, if STM stores a fixed number of (context-independent) items, then memory performance would be improved if the items themselves were organized into larger chunks. For instance, if a model can store four chunks in STM, and the model includes the sequence *GO* as a context-independent item (in that *GO* is coded the same way regardless of its position or context), then memory would be better for the sequence *GOXYZ* compared with *VWXYZ*. Both sequences involve five letters, but in the former case only four chunks need be recalled. The key point for present purposes is that a bigram frequency effect would be predicted if the model has learned some chunks of frequent bigrams.

Although the relevant learning mechanisms have not yet been implemented in alternative models, there is good evidence that chunking does indeed impact on STM span. For example, Cowan, Chen, and Rouder (2004) tested the original Miller (1956) claim that STM has a fixed capacity of memory chunks. In this study participants completed a training task in which a series of letters were presented either one at a time or in pairs (a condition in which participants might learn a new chunk), and then they completed an immediate serial recall task on lists that contained the letter pairs. The participants who had been trained on the letter pairs performed better than those who had been trained on the same letters presented one at a time, and critically, an analysis of the performance revealed that this improvement was attributable to recalling the letter pairs as a chunk. That is, memory for all participants was estimated to be about 3.5 chunks, but memory was better for participants who had learned the new chunks.

Thus there is every reason to assume that models of STM that rely on context-independent representations can be developed in a principled way in order to account for bigram effects (as well as other effects of background knowledge of sequential dependences).

Relating the Current Findings to the More General Debate Regarding PDP Versus Symbolic Models of Cognition

The failure of both the original and modified Botvinick and Plaut models to recall all the relevant sorts of test lists highlights a generalization constraint associated with context-dependent codes. It is exactly this sort of restriction that has led many authors to develop *symbolic* models of cognition that include context-independent representations of items in LTM and a dynamic process of assigning a role to these items. Models of this sort have been developed in a variety of domains, including word recognition (e.g., Davis, 1999), morphology (e.g., Pinker & Ullman, 2002; Prasada & Pinker, 1993), object recognition (Hummel & Biederman, 1992), conceptual structure (e.g., Fodor & Pylyshyn, 1998), and analogical reasoning (e.g., Hummel & Holyoak, 1997). It is important to emphasize that although symbolic models are inconsistent with PDP models, they do not constitute a rejection of neural networks in general. Indeed, a wide range of neural networks can operate on the basis of context-independent representations (cf. Bowers, 2002; Davis, 1999; Hummel & Holyoak, 1997). Pinker and Prince (1988) coined the terms *eliminative connectionism* to describe neural networks that eliminate symbols (i.e., models that reject context-independent item representations and a process of dynamically binding these items to a role) and *implementational connectionism* to describe neural networks that rely on symbols (i.e., models that rely on exactly these item representations and processes).

In this context, it is clear that neural network models of STM that rely on context-independent item representations (e.g., *A*), and a process of dynamically assigning the item a role in a to-be-remembered list (e.g., temporally assign *A* to the start of the list), should be described as symbolic (e.g., Brown et al., 2000; Burgess & Hitch, 1999; Farrell & Lewandowsky, 2002; Grossberg & Pearson, 2008; Page & Norris, 1998). Although some of these authors might question the use of the term *symbolic* when applied to their models, we would argue that these models not only satisfy the definition of *symbolic* that is commonly adopted in the domains of language and semantics (e.g., Pinker & Prince, 1988) but, more importantly, behave like symbolic models with regard to their ability to generalize.

For instance, consider the Page and Norris (1988) model that codes for letters in LTM in a context-independent fashion and dynamically codes order by the level of activation of the letter codes (such that the sequence *ABC* is coded with *A* the most active, *B* the second most active, and *C* the third most active). Here it does not matter whether a given letter was learned only in Position 1. As long as the model has an LTM representation of a letter, it is free to generalize to all positions, as the process of producing an activation gradient over a set of context-independent letter units is blind to which positions letters have been studied. Indeed, the model will be able to recall any possible sequences of familiar items (within its memory span), and the same generalization capacities will extend to all models that rely on context-independent representations (Brown et al., 2000; Burgess & Hitch, 1999; Farrell & Lewandowsky, 2002; Grossberg & Pearson, 2008). It is precisely the capacity of symbolic models to generalize that has led to their development in so many cognitive domains.

Part of the reason for the continuing debate regarding the relative merits of symbolic versus PDP models relates to disagreement about the operating characteristics of human cognitive and perceptual systems. On the one hand, advocates of symbolic models often highlight the generative capacities of the human mind (e.g., Fodor & Pylyshyn, 1988; Hummel & Holyoak, 1997; Pinker & Prince, 1988), and on the other hand, advocates of PDP theories often highlight the limitations of both PDP models and human minds in terms of generalization, with the limitations of the model taken as a virtue (e.g., Elman, 1998; McClelland, McNaughton, & O'Reilly, 1995; Munakata & O'Reilly, 2003). For instance, when considering the merits of symbolic and connectionist models in the domain of language, Thomas and McClelland (2008) wrote:

> Again, however, the characterization of human cognition in [symbolic, syntactically driven] terms is highly controversial; close scrutiny of relevant aspects of language—the ground on which the dispute has largely been focused—lends support to the view that the systematicity assumed by proponents of symbolic approaches is overstated and that the actual characteristics of language are well matched to the characteristics of connectionist systems. (p. 27)

As long as it is possible to disagree about human capacities in a given domain, there is little chance of resolve the modeling debate.

In our view, an advantage of the current analysis is that we have identified a restriction in generalization in a domain in which the capacities of the system should be less controversial. That is, whatever one's view regarding the systematicity of language or thought, there should be little disagreement that a person who learns to recall a letter in the initial position of a list is also capable of recalling that letter at the end of a list. The fact that a PDP model fails in this predictable way should make the computational limitations of conjunctive coding all the more salient. We take these restrictions as evidence for the role of symbols in cognition more generally.

References

Baddeley, A. D. (1964). Immediate memory and the "perception" of letter sequences. *Quarterly Journal of Experimental Psychology, 16,* 364–367.

Baddeley, A. D. (1986). *Working memory.* Oxford, England: Oxford University Press.

Botvinick, M. M., & Plaut, D. C. (2004). Doing without schema hierarchies: A recurrent connectionist approach to normal and impaired routine sequential action. *Psychological Review, 111,* 395–429.

Botvinick, M. M., & Plaut, D. C. (2006). Short-term memory for serial order: A recurrent neural network model. *Psychological Review, 113,* 201–233.

Bowers, J. S. (2002). Challenging the widespread assumption that connectionism and distributed representations go hand-in-hand. *Cognitive Psychology, 45,* 413–445.

Brown, G. D. A., Preece, T., & Hulme, C. (2000). Oscillator-based memory for serial order. *Psychological Review, 107,* 127–181.

Burgess, N., & Hitch, G. J. (1999). Memory for serial order: A network model of the phonological loop and its timing. *Psychological Review, 106,* 551–581.

Coltheart, M., Rastle, K., Perry, C., Langdon, R., & Ziegler, J. (2001). DRC: A dual route cascaded model of visual word recognition and reading aloud. *Psychological Review, 108,* 204–256.

Cooper, R. C., & Shallice, T. (2006). Hierarchical schemas and goals in the control of sequential behavior. *Psychological Review, 113,* 887–916.

Cowan, N. (2001). The magical number 4 in short-term memory: A

reconsideration of mental storage capacity. *Behavioral and Brain Sciences, 24,* 87–114.
Cowan, N., Chen, Z., & Rouder, J. N. (2004). Constant capacity in an immediate serial-recall task: A logical sequel to Miller (1956). *Psychological Science, 15,* 634–640.
Davis, C. J. (1999). *The self-organising lexical acquisition and recognition (SOLAR) model of visual word recognition* (Unpublished doctoral dissertation). University of New South Wales, Sydney, Australia.
Davis, C. J., & Bowers, J. S. (2004). What do letter migration errors reveal about letter position coding in visual word recognition? *Journal of Experimental Psychology: Human Perception and Performance, 30,* 923–941.
Davis, C. J., & Bowers, J. S. (2006). Contrasting five different theories of letter position coding: Evidence from orthographic similarity effects. *Journal of Experimental Psychology: Human Perception and Performance, 32,* 535–557.
Elman, J. (1998). Generalization, simple recurrent networks, and the emergence of structure. In M. A. Gernsbacher & S. Derry (Eds.), *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society* (p. 6). Mahwah, NJ: Erlbaum.
Farrell, S., & Lewandowsky, S. (2002). An endogenous model of ordering in serial recall. *Psychonomic Bulletin & Review, 9,* 59–79.
Fodor, J. A., & Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition, 28,* 3–71.
Grossberg, S. (1978). Behavioral contrast in short-term memory: Serial binary memory models or parallel continuous memory models? *Journal of Mathematical Psychology, 17,* 199–219.
Grossberg, S. (1980). How does the brain build a cognitive code? *Psychological Review, 87,* 1–51.
Grossberg, S., & Pearson, L. (2008). Laminar cortical dynamics of cognitive and motor working memory, sequence learning and performance: Toward a unified theory of how the cerebral cortex works. *Psychological Review, 115,* 677–732.
Henson, R. N. A. (1996). *Short-term memory for serial order* (Unpublished doctoral dissertation). University of Cambridge, Cambridge, England.
Hulme, C., Suprenant, A. M., Bireta, T. J., Stuart, G., & Neath, I. (2004). Abolishing the word-length effect. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 30, 1,* 98–106.
Hummel, J. E., & Biederman, I. (1992). Dynamic binding in a neural network for shape recognition. *Psychological Review, 99,* 480–517.
Hummel, J. E., & Holyoak, K. J. (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review, 104,* 427–466.
Jefferies, E., Frankish, C. R., & Lambon Ralph, M. A. (2006). Lexical and semantic binding in verbal short-term memory. *Journal of Memory and Language, 54,* 81–98.
Lee, C. L., & Estes, W. K. (1981). Item and order information in short-term memory: Evidence for multilevel perturbation processes. *Journal of Experimental Psychology: Human Learning and Memory, 7,* 149–169.
Lewandowsky, S., & Murdock, B. B. (1989). Memory for serial order. *Psychological Review, 96,* 25–57.
Marcus, G. F. (1998). Rethinking eliminative connectionism. *Cognitive Psychology, 37,* 243–282.
Marcus, G. F., Vijayan, S., Bandi Rao, S., & Vishton, P. M. (1999, January 1). Rule learning in seven-month-old infants. *Science, 283,* 77–80.
McClelland, J. L., McNaughton, B. L., & O'Reilly, R. C. (1995). Why there are complementary learning-systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review, 102,* 419–457.
McClelland, J. L., & Rumelhart, D. E. (1981). An interactive activation model of context effects in letter perception: I. An account of basic findings. *Psychological Review, 88,* 375–407.
McClelland, J. L., Rumelhart, D. E., & The PDP Research Group. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Vol. 2. Psychological and biological models*. Cambridge, MA: MIT Press.
Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63,* 81–97.
Miller, G. A., & Selfridge, J. A. (1951). Verbal context and the recall of meaningful material. *American Journal of Psychology, 63,* 176–185.
Munakata, Y., & O'Reilly, R. C. (2003). Developmental and computational neuroscience approaches to cognition: The case of generalization. *Cognitive Studies, 10,* 76–92.
Page, M. P. A., & Norris, D. (1998). The primacy model: A new model of immediate serial recall. *Psychological Review, 105,* 761–781.
Pinker, S., & Prince, A. (1988). On language and connectionism: Analysis of a parallel distributed processing model of language-acquisition. *Cognition, 28,* 73–193.
Pinker, S., & Ullman, M. (2002). The past and future of the past tense. *Trends in Cognitive Science, 6,* 456–463.
Prasada, S., & Pinker, S. (1993). Generalisation of regular and irregular morphological patterns. *Language and Cognitive Processes, 8,* 1–56.
Thomas, M. S. C., & McClelland, J. L. (2008). Connectionist models of cognition. In R. Sun (Ed.), *The Cambridge handbook of computational psychology* (pp. 23–58). New York, NY: Cambridge University Press.
Wickelgren, W. A. (1966). Phonemic similarity and interference in short-term memory for single letters. *Journal of Experimental Psychology, 71,* 396–404.

Received June 1, 2007
Revision received February 2, 2009
Accepted February 4, 2009 ■

Postscript: More Problems With Botvinick and Plaut's (2006) PDP Model of Short-Term Memory

Jeffrey S. Bowers and Markus F. Damian
University of Bristol

Colin J. Davis
Royal Holloway, University of London

In our commentary we demonstrated that Botvinick and Plaut's (2006) model of immediate serial recall catastrophically fails when familiar letters are tested in untrained positions within a list (Simulation 2), and a modified version of their model with a distributed letter coding scheme also fails to recall familiar (and novel) letters when tested in untrained positions (Simulation 7). That is, short-term memory (STM) did not generalize to all possible test sequences. We argued that these failures reflect a fundamental limitation of the conjunctive coding schemes used in parallel distributed processing (PDP) models of cognition. Indeed, these constraints have inspired *symbolic* models of cognition that rely on context-independent representations of items in long-term memory (LTM; e.g., a representation for the letter *A*, unspecified by position within a list) and a dynamic (short-term) process of binding these items to a given role (e.g., a dynamic process of binding the letter *A* to a given position) in order to generalize more broadly.

Botvinick and Plaut (2006) rejected these claims and reported a simulation in which a new version of their model recalls familiar

(and novel) items in novel positions. However, it is important to note the conditions in which this model succeeded. It included 30 input–output units, with the first 10 units coding for the onset, the next 10 units the vowel, and the final 10 units the coda. Each syllable was defined by activating one onset, one vowel, and one coda unit, and the model was trained on 999 out of a possible 1,000 ($10 \times 10 \times 10$) syllables. Their critical finding was that the model could recall the untrained item without difficulty (in all positions). What Botvinick and Plaut did not emphasize, however, was that the model was trained on all the letters in all positions of the list. So, in principle, the model could recall novel syllables (and familiar syllables in untrained positions) by recalling familiar phonemes in trained positions. For example, if the untrained syllable was *SAM*, then the model could recall SAM in Position 1 of a list by learning and activating the following trained conjunctive codes: *S*-onset-in-list-Position-1, *A*-vowel-in-list-Position-1, and *M*-coda-in-list-Position-1. Indeed, that is what the model has done.

To further highlight the generalization constraints associated with these learned conjunctive codes, we ran two new simulations. First, we developed a modified model in which the first 10 units were reserved for onsets, the next six for vowels, and the final 10 for codas (resulting in $10 \times 6 \times 10$ or 600 possible syllables). We trained the model for 3 million trials on lists of up to nine syllables taken from a random set of 300 syllables but excluded 32 syllables that included the phoneme *R* in the coda position (henceforth *R*-syllables; *R* represented by input and output Unit 17). We then trained the model for another 2 million trials, during which *R*-syllables were allowed to appear in Position 1 but not in other positions. This constitutes a general replication of the procedure we reported in our Simulation 7 but using a similar representational structure as Botvinick and Plaut's new simulation. At test, the model was presented with 1,000 lists of six syllables (taken from the vocabulary of 300) that all contained one random *R*-syllable in list Positions 1–6. As can be seen in Figure P1, when the model had not been trained on *R*-syllables, it catastrophically failed on these lists. During the additional training with the *R*-syllables, the model slowly developed a position-specific knowledge of these items: Performance improved for the *R*-syllables in first position, but the model continued to catastrophically fail when these syllables were presented in other positions. This pattern of performance is just as we reported in Simulation 7. More strikingly, in a second simulation, we trained the model on the 300 syllables with no restrictions except that the *R*-syllables were not permitted to occur in list Position 1. After 3 million training trials, the model could recall lists of six syllables that contained one *R*-syllable as long as it did not occur in Position 1. That is, when the model was tested on 1,000 lists of six syllables, its recall performance was 2.5%, 49.4%, 46.9%, 45.8%, 47.0%, and 45.5% when the *R*-syllable occurred in Positions 1–6, respectively. So, learning to recall *CAR* in list Positions 2–6 did not allow the model to recall *CAR* in Position 1. Botvinick and Plaut endorse our claim that anyone who can recall the sequence *ree-B* should also succeed in recalling the sequence *B-ree*. But as we have demonstrated here, PDP models do not exhibit the same position invariance.

These findings suggest that our new model succeeded (to the extent that it did) by relying on learned phoneme–position conjunctive codes. To test this more directly, we trained it on a random sample of 500 of the possible 600 syllables for 4 million trials and then tested it on 1,000 lists of syllables composed of

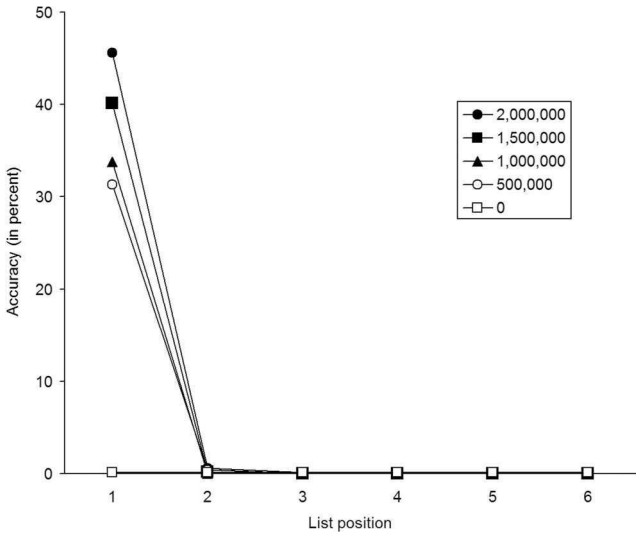

Figure P1. Performance of our modified model when it was first trained for 3 million trials on lists of syllables that excluded the phoneme *R* (*R*-syllables) and then trained another 2 million trials when the *R*-syllables were free to occur in Position 1 but not in other positions. Performance was assessed on 1,000 lists of six syllables that all contained one *R*-syllable in various positions (1–6) after various levels of training: immediately after the 3 million trials in which the *R*-syllables were untrained and following an additional 500,000, 1 million, 1.5 million, and 2 million training trials in which the *R*-syllables were free to occur in Position 1.

familiar or unfamiliar syllables that varied in length. If lists of syllables are recalled on the basis of phoneme–position conjunctive codes (e.g., the syllable *SAM* at the start of the list is coded by coactivating the long-term representations for *S*-onset-in-list-Position-1, *A*-vowel-in-list-Position-1, and *M*-coda-in-list-Position-1), then the familiarity of the syllables should be irrelevant. This is indeed the case, as depicted in Figure P2. By contrast, lexical representations play a key role in supporting human STM, as revealed by a robust advantage of words over nonwords (e.g., Jefferies, Frankish, & Lambon Ralph, 2006). Another failure of the model follows directly from this. STM is sensitive to background knowledge of sequential dependencies, and this extends to the sequential dependencies between lexical items, or newly trained syllables (e.g., Botvinick & Bylsma, 2005). Indeed, the original Botvinick and Plaut model trained on 26 letters captured these sequential effects, and this was considered a key advantage of the model compared with others. But these sequential effects are lost in the modified model given that memory performance is based on remembering sequences of phonemes. In short, when a modified Botvinick and Plaut model is trained on a larger vocabulary (e.g., 100s syllables rather than 26 letters), it suffers from both under- and overgeneralization. That is, the model cannot recall familiar (or novel) syllables that include familiar phonemes in untrained positions, but as long as this constraint is avoided (by ensuring that the training extends to all phonemes in all positions), it recalls novel syllables just as well as familiar ones and untrained sequences of syllables just as well as trained sequences.

Two additional points merit brief discussion. First, Botvinick and Plaut (2006) claimed that single-cell recording data lend support to their view that STM is mediated by context-dependent

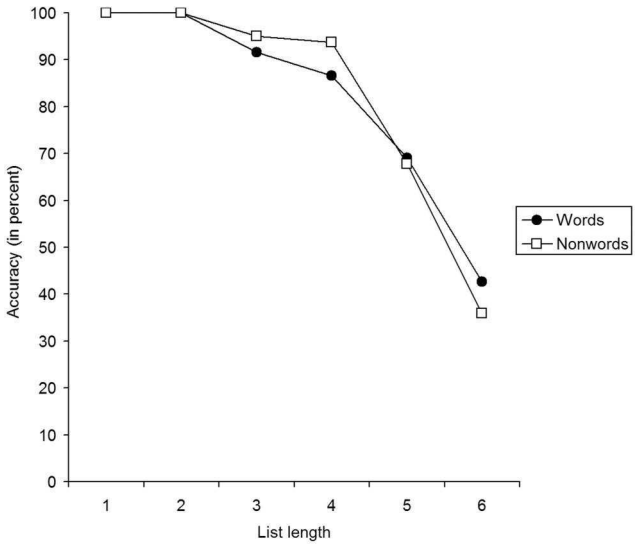

Figure P2. Performance of our modified model when it was trained for 4 million trials on random lists of syllables taken from a vocabulary of 500 of the possible 600 syllables. Performance was assessed on 1,000 lists of syllables taken from the trained (word) and untrained (nonword) sets, with list length varying from one to six syllables.

representations. But they failed to mention the evidence for context-independent representations. For example, they cited Ninokura, Mushiaske, and Tanji (2004), who reported that 30% of the relevant neurons in the lateral prefrontal cortex were selective to both position and identity (conjunctive cells). It is perhaps worth mentioning that 44% of the task-relevant neurons in this study were sensitive to list position irrespective of object identity, and 26% responded to object identity irrespective of list position (context-independent cells). Similar findings have been reported elsewhere (e.g., Averbeck, Chafee, Crowe, & Georgopoulos, 2002; Inoue & Mikami, 2006). Second, we think that Botvinick and Plaut mischaracterized Page and Norris's (1998) primacy model of STM, and they appear to have a misunderstanding regarding the representations employed in PDP and symbolic models. They claimed that the primacy model relies on conjunctive representations of items and order. But the model includes LTM representations of items that are coded independently of order, and the order of an item in a list is dynamically coded by the relative activation of the items representations. The fact that the given letter (e.g., *R*) is coded with the same unit regardless of its list position allows the model to generalize more broadly than PDP models that do rely on conjunctive representations. Indeed, all symbolic models of cognition include a process that dynamically assigns items a role, where the role could specify the position of a letter within a word (e.g., Davis, 1999), an attachment relation between object parts (e.g., Hummel & Biederman, 1992), or, in the present case, the order of items in a to-be-remembered list (e.g., Page & Norris, 1998). By contrast, Botvinick and Plaut adopted a modeling approach that binds items to roles statically, through conjunctive codes in LTM (e.g., where *R*-in-Position-1 and *R*-in-Position-2 are coded differently). By relying on a version of back-propagation, they "stipulated" that their model would learn conjunctive (context-dependent) long-term representations. The consequences are just as we predicted (see also Bowers & Davis, in press). The ball is now in their court to show that the many limitations of their model can be addressed without appealing to symbolic (context-independent) representations in LTM.

References

Averbeck, B. B., Chafee, M. V., Crowe, D. A., & Georgopoulos, A. P. (2002). Parallel processing of serial movements in prefrontal cortex. *Proceedings of the National Academy of Sciences, USA, 99,* 13172–13177.

Botvinick, M., & Bylsma, L. M. (2005). Regularization in short-term memory for serial order. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 31,* 351–358.

Botvinick, M. M., & Plaut, D. C. (2006). Short-term memory for serial order: A recurrent neural network model. *Psychological Review, 113,* 201–233.

Bowers, J. S., & Davis, C. J. (in press). Learning representations of wordforms with recurrent networks: Comment on Sibley, Kello, Plaut, and Elman (2008). *Cognitive Science.*

Davis, C. J. (1999). *The self-organising lexical acquisition and recognition (SOLAR) model of visual word recognition* (Unpublished doctoral dissertation). University of New South Wales, Sydney, Australia.

Hummel, J. E., & Biederman, I. (1992). Dynamic binding in a neural network for shape recognition. *Psychological Review, 99,* 480–517.

Inoue, M., & Mikami, A. (2006). Prefrontal activity during serial probe reproduction task: Encoding, mnemonic, and retrieval processes. *Journal of Neurophysiology, 95,* 1008–1041.

Jefferies, E., Frankish, C. R., & Lambon Ralph, M. A. (2006). Lexical and semantic binding in verbal short-term memory. *Journal of Memory and Language, 54,* 81–98.

Ninokura, Y., Mushiaske, H., & Tanji, J. (2004). Integration of temporal order and object information in the monkey lateral prefrontal cortex. *Journal of Neurophysiology, 91,* 555–560.

Page, M. P. A., & Norris, D. (1998). The primacy model: A new model of immediate serial recall. *Psychological Review, 105,* 761–781.