Cognition 148 (2016) 47–63

Contents lists available at ScienceDirect

Cognition

journal homepage: www.elsevier.com/locate/COGNIT

Why do some neurons in cortex respond to information in a selective manner? Insights from artificial neural networks

Jeffrey S. Bowers [*], Ivan I. Vankov [1], Markus F. Damian, Colin J. Davis

University of Bristol, United Kingdom

ARTICLE INFO

Article history:
Received 7 May 2014
Revised 26 November 2015
Accepted 12 December 2015
Available online 24 December 2015

Keywords:
Grandmother cells
Localist representations
Distributed representations
Short-term memory
Superposition catastrophe
Generalization

ABSTRACT

Why do some neurons in hippocampus and cortex respond to information in a highly selective manner? It has been hypothesized that neurons in hippocampus encode information in a highly selective manner in order to support fast learning without catastrophic interference, and that neurons in cortex encode information in a highly selective manner in order to co-activate multiple items in short-term memory (STM) without suffering a superposition catastrophe. However, the latter hypothesis is at odds with the widespread view that neural coding in the cortex is highly distributed in order to support generalization. We report a series of simulations that characterize the conditions in which recurrent Parallel Distributed Processing (PDP) models of immediate serial can recall novel words. We found that these models learned localist codes when they succeeded in generalizing to novel words. That is, just as fast learning may explain selective coding in hippocampus, STM and generalization may help explain the existence of selective codes in cortex.

1. Introduction

Gross, Bender, and Roch-Miranda (1969) identified a neuron in inferior temporal (IT) cortex of a macaque monkey that selectively responded to images of hands. The results were so surprising that Gross (1994) later recounted how he was initially nervous to call the neuron a "hand cell", and speculated that this work was largely ignored for a decade because researchers didn't believe the findings. Subsequently there have been dozens of studies reporting similar findings, and the observation that some neurons in cortex and hippocampus respond to high-level perceptual information in a highly selective manner is no longer in doubt (for a detailed review of the neuroscience, see Bowers, 2009).

A related observation is that neural firing in the cortex and hippocampus is highly sparse, meaning that small percentage of neurons in a population of neurons respond to a given input (e.g., Shoham, O'Connor, & Segev, 2006). Selective and sparse coding are distinct concepts, and it is possible for these measure to dissociate (Willmore & Tolhurst, 2001). For example, a small proportion of neurons may fire in response to a given input (the representation is sparse), but at the same time, the active neurons respond to many different inputs (the response is non-selective). This would constitute a sparse distributed code for the input. Alternatively, it is possible that many redundant neurons respond selectively to a given input; this would constitute a selective but non-sparse representation. The failure to distinguish these concepts has led to a number of conceptual confusions (Bowers, 2011; Földiák, 2009). For present purposes, the important point is that various brain systems rely on representations that are both highly selective and sparse.

Given these findings, an important question is why do neural systems code for information in this format? In fact, there may be different reasons for the selective and sparse coding observed in the hippocampus and in the cortex. With regard to the hippocampus, Marr (1971) suggested that information is coded in a highly sparse format in order to support fast learning (as required for episodic memory, for example). The basic insight is that as long as different memories are coded separately from one another, then learning something new will have little impact on previously stored memories. Sparse (or selective) coding is useful in this context because it results in non-overlapping memory representations. By contrast, when memory representations overlap, new learning will often interfere with old memories. Indeed, this is the case with Parallel Distributed Processing (PDP) models that learn highly distributed representations with each input coded as a pattern of activation over many different units (non-sparse coding) and each unit involved in coding many different things (non-selective coding). In these models, rapid learning often leads to rapid forgetting, a phenomenon called catastrophic interference

* Corresponding author.
E-mail address: j.bowers@bristol.ac.uk (J.S. Bowers).
1 New Bulgarian University, Sofia, Bulgaria.

http://dx.doi.org/10.1016/j.cognition.2015.12.009
0010-0277/© 2015 Elsevier B.V. All rights reserved.

(McCloskey & Cohen, 1989), or the stability–plasticity dilemma (Grossberg, 1980).

With regard to the cortex, Bowers, Vankov, Damian, and Davis (2014) argued that selective coding is well suited for co-activating multiple things at the same time, as required for short-term memory (STM). The basic claim is that co-activated distributed patterns are ambiguous in that it is not possible to recover the constituent patterns when they are blended together; the so-called superposition catastrophe (Von der Malsburg, 1986). Bowers et al. (2014) showed that a recurrent PDP model trained to co-activate and recall multiple words at the same time succeeded by learning highly selective (often localist) letter and word codes, and we argued that this may help explain the selective responding of neurons in cortex given that the cortex supports STM in various domains (Cowan, 2001).

However, the latter hypothesis is difficult to reconcile with the common claim that distributed codes are better suited for coding information in the cortex. For example, according to the complementary learning systems (CLS) hypothesis (McClelland, McNaughton, & O'Reilly, 1995) knowledge is coded in a highly distributed format in cortex in order to support various forms of generalization (e.g., identifying an object from a novel viewpoint or reading a novel word), and in a highly sparse and selective manner in the hippocampus for the sake of fast learning (in line with Marr). Indeed, McClelland et al. (1995) claimed that that fast learning and generalization were incompatible functions that require sparse and selective coding on the one hand, and more densely distributed coding, on the other. Consistent with this analysis, PDP models that succeed at generalizing are thought to rely on learned distributed representations (e.g., Plaut, McClelland, Seidenberg, & Patterson, 1996; Seidenberg & McClelland, 1989). Distributed representations support generalization because similar inputs are coded with similar overlapping representations, and as a result, novel inputs overlap with similar pre-existing knowledge.

How can we reconcile the claim that neural representations in cortex are highly selective for the sake of STM (Bowers et al., 2014) with the claim that neural representations in cortex are highly distributed for the sake of generalization (McClelland et al., 1995)? One possibility can quickly be ruled out, namely, that that different sets of representations support STM and generalization. The problem with this solution is that we can generalize and remember multiple things at the same time. For example, not only can we generalize from past reading experience in order to name novel words (e.g., *blap*), but we can also read and remember multiple novel words in a STM task (e.g., read and repeat the nonwords *blap*, *dram*, and *samp*). The same is true with PDP networks. For example, Bowers, Damian, and Davis (2009) found that a PDP model of STM could recall lists of familiar and unfamiliar words at a similar level of performance (see figure P2, p. 997). Some form of representation can obviously generalize and co-activate multiple items at the same time.

One reason to believe localist coding could serve both functions is that there are already a number of localist models that support generalization. For example, in the DRC model of word naming (Coltheart, Rastle, Perry, Langdon, & Ziegler, 2001), novel words are read through the serial application of (hand-wired) localist grapheme–phoneme units. Similar to the distributed models of word naming, novel words in the DRC model overlap with pre-existing knowledge (the localist letter codes overlap), and it is the overlap that supports generalization. This at least raises the possibility that PDP models of immediate serial recall that succeed in generalizing learned localist letter codes.

It is important to emphasize the significance of this issue. A fundamental claim is that PDP models generalize on the basis of distributed representations, and to date, all of the evidence is consistent with this claim. However, this position would have to be substantially modified if PDP models only generalize on the basis of distributed representations in some conditions (when activating one thing at a time), and generalize on the basis of localist representations in other conditions (when co-activating multiple things at the same time). Indeed, given that cortical systems support STM and generalization, it is possible that highly selective codes (if not localist codes) play an important role in human generalization. This conclusion would also help make sense of the selective neural responses reported in cortex, and challenge the complementary learning systems that is predicated on the view that distributed representations are needed in the cortex for generalization. We explore these issues in the following simulations.

Although we are specifically focused on the question of how PDP models generalize when trained to co-activate multiple items at the same time, our findings may well have implications beyond short-term memory tasks. For example, consider again models of word naming (e.g., DRC model). Although the model only name words one-at-a-time, it is widely assumed that the representations that support word naming are involved in other tasks, including speech perception (e.g., Harm & Seidenberg, 2004) and STM (e.g., Page, Madge, Cumming, & Norris, 2007). Accordingly, any constraints we observe regarding the representations that support STM may have implications for other domains, including the identification and naming of single words and objects, etc.

2. Background simulations

The current work is inspired by a PDP model of STM initially developed by Botvinick and Plaut (2006) and subsequent models that further explored the conditions in which recurrent networks succeed and fail in recalling co-activated items. Accordingly, before introducing the current simulations, we briefly review the past findings to set the stage.

Botvinick and Plaut (2006) developed a recurrent PDP model of immediate serial recall that encoded a series of letters presented one-at-a-time at the input layer and reproduced the letters in the same order at the output layer – a classic test of STM. Critically, the model stored lists of letters in STM by superimposing their activations within a hidden layer that included recurrent connections. See Fig. 1. Interestingly, the model reproduced a number of key behavioral phenomena, including some results that are problematic for other theories. However, for present purposes, the key point is that the model was not tested in its ability to generalize. Given that the model was trained to recall lists of letters that were coded in a localist format at the input and output layers, it is not clear how the model could generalize.

Subsequent papers provided additional information about the conditions under which recurrent PDP models of immediate serial recall can generalize to novel items and the types of representations that are learned, but key issues are still unresolved. With regards to generalization, Bowers et al. (2009) introduced a distributed letter-coding scheme to the model (each letter was coded as a pattern of activation across five input and output units) and we trained the model to recall lists of letter taken from a vocabulary of 25 or 26 letters. The model succeeded in recalling lists of familiar letters, but when tested on a single novel letter (a novel pattern of activation across the input units) the model failed most of the time. That is, the model's memory span for novel letters was approximately zero. In response to this work, Botvinick and Plaut (2009) trained another version of the model to recall lists of syllables defined as a pattern of activation across three localist letter units. They trained the model to recall lists of syllables taken from a vocabulary of 999 of possible 1000 syllables. At test, the model did succeed on the 1000th syllable. Similarly, in a follow-up study, Bowers et al. (2009) found that a model could generalize to novel syllables when trained on 500 of a possible 600 syllables.

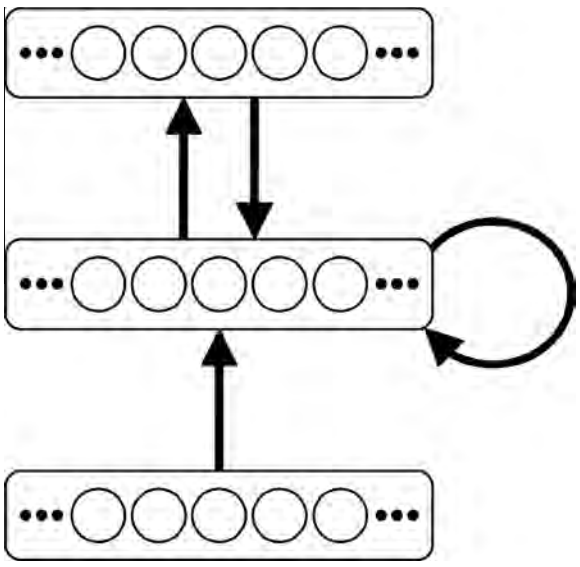

Fig. 1. Diagram of the Botvinick and Plaut (2006) recurrent PDP model of immediate serial recall. The model includes a set of 27 input and output units (one for each letter of the alphabet plus a unit in the input layer that cues recall, and a unit in the output layer that codes the end of a list) plus a set of 200 hidden units. Arrows indicate connections between and within layers.

Although the latter simulations demonstrate that PDP models can overcome the superposition catastrophe and generalize in some conditions it is unclear whether distributed or localist codes supported successful performance. Either outcome is plausible given that Botvinick and Plaut (2006) provided evidence that their original model recalled lists of familiar letters relying on distributed code, whereas Bowers et al. (2014) showed that a recurrent PDP model trained to code multiple words at the same time (without regards to order) learned localist letter and word codes. However, neither of these models was trained in a way that could support generalization, so these findings do not reveal the nature of the learned representations that support STM and generalization.

3. Current studies

We report a series of simulations on a PDP model of immediate serial recall and carry out a set of analyses on the hidden units in order provide insights into (a) the training conditions in which these models succeed and fail to recall familiar and novel words, and (b) the representations that support success and failure.

With regard to (a) the previous models of immediate serial recall that succeeded and failed with novel words (and novel letters) differed in a number of ways, but perhaps the most salient difference was the size of the training set (the models that succeeded were trained on a larger vocabulary of items). Accordingly, the size of the training set was varied in the simulations reported below. We found that vocabulary set size is indeed the critical variable. In order to address (b) we systematically analyzed the activations of the hidden units in response to the words and nonwords and carried out "lesion" studies to assess the impact of removing single hidden units on performance. Both sets of analyses showed that the model learned localist codes when it succeeded (sparseness was not the critical factor). In order to further test this hypothesis we predicted previous PDP models of immediate serial recall that failed to generalize (e.g., Botvinick & Plaut, 2006; Bowers et al., 2009) relied on leaned distributed codes where as previous models that succeeded in generalizing (e.g., Botvinick & Plaut, 2009) relied on learned localist codes (despite the fact that the models differed in various parameters and training conditions). These predictions were confirmed.

It is important to emphasize that our primary goal is not to make claims about how recurrent PDP models of immediate serial recall work (although this is interesting in its own right). Rather, we want to provide some insights into whether highly selective or distributed representations are better suited for supporting two core functions of cortex, namely, STM and generalization. For this purpose PDP models are useful because the learned representations are said to be emergent rather than "stipulated" by the modeler (Plaut & McClelland, 2000). Accordingly, if PDP models learn localist codes when they succeed it suggests that there are computational advantages of these representations under these conditions.

3.1. Simulation 1(a and b): training on lists of words

The purpose of these simulations was to assess generalization as a function of vocabulary size. We developed a model that has similar computational resources as the original Botvinick and Plaut (2006) model, but adapted the model for our purposes. The model included 30 input phoneme units, 200 hidden units, and 30 output phoneme units, and the trained words were coded through the co-activation of three phoneme codes: The first 10 input (and output) units coded for the onsets of words, the next 10 items for the vowels, and the final 10 units for codas, and each word was coded as one active onset, nucleus, and coda unit. Specifically, the 30 input units coded for the following phonemes: (b, c, d, f, g, h, j, k, l, m) (a, e, i, o, u, y, aa, ea, ou, oo) (n, p, q, r, s, t, v, w, x, z). Given that each word was defined as the co-activation of one onset, one nucleus, and one code unit, the total number of possible words was 1000 (10 onsets $\times$ 10 vowels $\times$ 10 codas). There was also one more input unit that coded for the end of a list, as well as a corresponding output unit.

On each trial the network was presented with a random sequence of words from the training vocabulary (without replacement). We followed the general training procedure of Botvinick and Plaut (2006): Each training cycle began with a single word, with length increasing by one until a simulation-specific maximum length was reached (in Simulations 1 the maximum list length was nine words). Following presentation of a list of maximum length, the list length returned to one and the cycle repeated. At test we assessed recall for lists of familiar and novel words, varying from list length one to nine. The output of the model was determined by comparing the pattern of activation at the output layer with the 1000 patterns that defined all possible words. The model was said to recall the word (either a trained or a non-trained word) that was composed of the three most active units at the output.

The network was trained using backpropagation through time (Werbos, 1990). The learning rate was fixed to 0.0005 and no momentum was used. The standard sigmoid activation function was used in both the hidden and the output layer. The gain of the activation function was set to 1. The error at the output layer was computed using the cross entropy function.[2]

The key difference between Simulation 1a and Simulation 1b was the size of the training vocabulary, with a small and large vocabulary of 30 and 300 words used, respectively. The vocabularies were randomly selected from the possible 1000 words, but we ensured that all the letters and pairs of letters (bigrams) occurred at similar rates in the training vocabulary. At test the familiar words were again composed of a random selection from the vocabulary (without replacement), and the novel words were composed

[2] We used a sigmoid as activation function for the out layer because multiple active units (three units were activated for each word, one for each letter), whereas Botvinick and Plaut (2006) used the softmax output function. Softmax is used when there is just one active unit within a layer; this was not appropriate for our simulations.

of a random selection of words from the non-trained set. For example, when trained on the small vocabulary of 30 words, the novel words were randomly selected from the 970 untrained words ($10 \times 10 \times 10$ possible words, minus 30).

We trained the network in the two vocabulary conditions until it successfully recalled six familiar words at 50% (roughly human performance). This required 3 million trials for the small vocabulary and 15 million for the large vocabulary. Fig. 2 shows the performance on the familiar and unfamiliar words in Simulations 1a and 1b as a function of list length. Consistent with past results (Botvinick & Plaut, 2009; Bowers et al., 2009) the model succeeded in recalling lists of familiar words of various lengths regardless of the training size vocabulary whereas performance on novel words was catastrophically bad when trained on the smallest vocabulary, and excellent when trained on a larger vocabulary. Given that the two models were identical apart from their training, the current results support the hypothesis that vocabulary size is the critical variable that accounts for the mixed generalization results in previous studies.

3.2. Simulation 2 (a and b): training words one-at-a-time

The above simulations confirm that PDP models of immediate serial recall can indeed generalize to novel items when trained on a large vocabulary of items. Below we explore how the model succeeded by assessing the nature of the learned representations. But first we explore why the model failed to generalize when trained on a smaller vocabulary (Simulation 1a).

One possibility is that the small vocabulary presented at training did not sample broadly enough from the possible 1000 words, and as a consequence, the model did not learn the statistical structure of the overall vocabulary that would allow the model to generalize to novel items. There are many demonstrations of PDP models failing to generalize due to the inadequate sampling of the training space. Another possibility, however, is that the sampling size was adequate but the task of co-activating multiple words at the same time restricted generalization in the small vocabulary condition.

Why might co-activating multiple words restrict generalization? Consider Botvinick and Plaut's (2006) explanation of how their model solved the task. They noted that blends of co-active distributed patterns can be ambiguous (the superposition catastrophe) and argued that their model learned to cope with ambiguities by learning a bias to recall the most likely sequence of items given its training history. This was thought to reduce the ambiguity to an extent sufficient to allow distributed representations to support STM at a level commensurate with human performance.

However, Botvinick and Plaut did not consider the downside of this solution, namely, that such a bias works against recalling novel items. That is, novel words are not the most likely output given the training history, and accordingly, the model might be expected to lexicalize, producing an incorrect familiar word most consistent with a given blend. Although human STM is better for familiar compared to novel words (Jefferies, Frankish, & Lambon Ralph, 2006), we nevertheless have no difficulty in repeating a few nonwords, such as "blip-blap". The difficulty in coding multiple novel things with distributed representations was highlighted by Bowers (2002):

> Perhaps most problematic, blends are not necessarily the product of combining pre-trained patterns. Imagine the situation in which two words are co-active in a distributed phonological system. Although the blend pattern may be more similar to the two constituent words compared to any other trained word, the pattern is not more similar to many possible items (or possible blends). The blend pattern might have been produced by combining two nonwords, for example, although this possibility cannot be recovered from the blend. But we can co-encode two novel items: e.g., phonologically, as BLIP–BLAP in short-term memory… blend patterns in distributed systems are deeply ambiguous (p. 431).

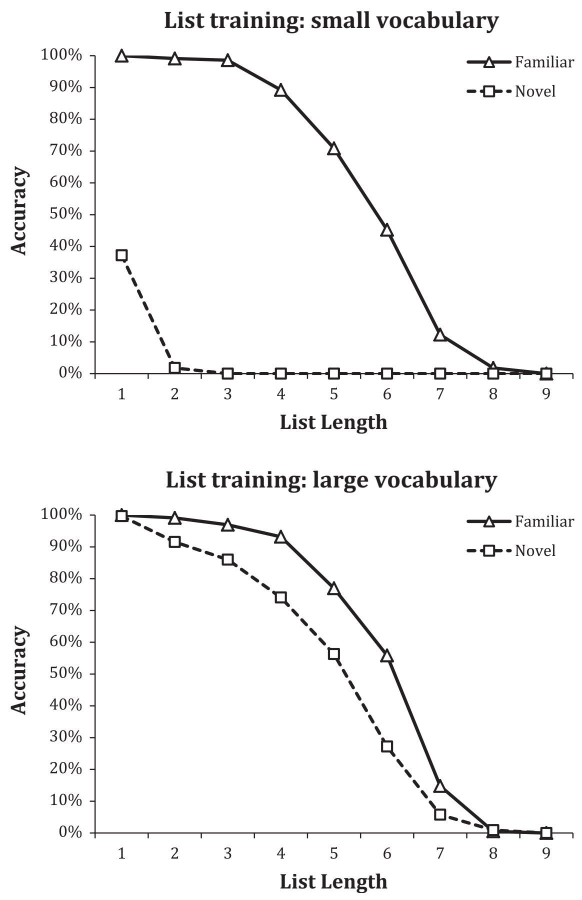

Fig. 2. Performance of the network on lists of familiar and novel words (from list length 1–9) when trained on a small vocabulary of 30 words and a large vocabulary of 300 words.

To reiterate, there are two contrasting predictions as to why the PDP model trained with a small vocabulary (Simulation 1a) failed to generalize: either the model was not trained on a large enough vocabulary of words in order to extract the necessary statistical regularities of the inputs, or the model failed to generalize because it learned a lexical bias in order to recall lists of familiar words in the face of ambiguous blend patterns (consistent with the analysis of Botvinick & Plaut, 2006). That is, generalization was compromised in an attempt to overcome the superposition catastrophe.

In order to contrast these two hypotheses we trained the same network with vocabulary sizes of 30 (Simulation 2a) and 300 (Simulation 2b) when words were presented one-at-a-time. That is, the model was presented with the same set of words, but the model did not face the superposition catastrophe. If the small vocabulary was responsible for the poor generalization then generalization should continue to be poor. By contrast, if the superposition catastrophe played a role in restricting generalization then the model trained on a small vocabulary of items one-at-a-time should be able to generalize.

We first trained the networks in Simulation 2a and Simulation 2b until they performed at 100% accuracy on familiar words, which took 20,000 and 30,000 trials, respectively. This led to 59% and 98% correct performance on nonwords in the small and large vocabularies, respectively. We then extended training in both simulations to one million trials (still much less training than in Simulation 1) and again found that the model did quite well with single novel words in both vocabulary conditions as well (81% and 100%, respectively). Given that the same set of words were presented in the small vocabulary conditions in Simulation 1a and Simulation 2a, it appears that the poor generalization in Simulation 1a can be attributed, at least in part, to learning to cope with the superposition catastrophe with distributed representations (perhaps by means of learning a lexical bias, as suggested by Botvinick and Plaut (2006)).

4. Analyses of the hidden layer

The above simulations suggest that the model trained on a small vocabulary learned a lexical bias in order to co-activate multiple words whereas the model trained on a large vocabulary learned to co-activate multiple words without a lexical bias (allowing it to generalize). That is, the models appear to cope with the superposition constraint in two different ways.

One interesting possibility is that the both models learned distributed representations but developed different strategies for coping with the blend patterns. That is, the lexical bias strategy in the small vocabulary condition was replaced by an alternative strategy in the large vocabulary condition that allowed generalization to novel items relying on distributed representations. This would show that the superposition catastrophe is more easily overcome than previously presumed (e.g., Bowers, 2002; Bowers et al., 2014; Von der Malsburg, 1986). It would also support the widespread view that generalization in PDP models is supported by distributed representations.

Another possibility, however, is that the model trained on a small and large vocabulary learned distributed and highly selective (or even localist) representations, respectively. On this latter hypothesis, overlapping distributed patterns are only moderately ambiguous when the model was trained on a small vocabulary, and the model could recover the constituent word patterns by adopting a lexical bias. By contrast, when the model was trained on a large vocabulary of items, the blends became more ambiguous, and the model was forced to learned selective (largely non-overlapping) representations in order to avoid the superposition constraint. We assess these two possibilities next.

4.1. Single-unit recording studies

In order to gain insight into why we observed contrasting results in the two vocabulary conditions we examined the activation of individual hidden units – much like the single-cell recording studies carried out in neuroscience. Following training in all of the above simulations we recorded the activation of all 200 hidden units in response to all possible words (1000 in all) presented one-at-a-time, and displayed the results using a graphical method introduced by Berkeley, Dawson, Medler, Schopflocher, and Hornsby (1995). In this method, a separate scatter plot for each hidden unit is created, and each point in a scatter plot corresponds to a unit's activation in response to a single input (in this case, a word). All the relevant inputs can then be presented to the network, and the response of each unit is recorded. Level of unit activation is coded along the *x*-axis, and an arbitrary value is assigned to each point on the *y*-axis in order to prevent points from overlapping (in case two different inputs drive a given unit to the same level). This effectively provides a single-cell (or in the case, a single unit) recording for each hidden unit in response to a large set of inputs.

In order to facilitate the presentation of these recordings, we adapted this procedure in a number of ways for our purposes. In the first set of scatter plots, we sorted the words according to whether they were familiar or novel, with familiar words presented in dark crosses, and novel words presented in light crosses. Familiar and novel words were blocked along the *y*-axis, and within each block, familiar and novel words were organized alphabetically. In Fig. 3a and b we plot the activation of each hidden unit as a function of vocabulary size when words were trained one-at-a-time (Simulations 2a and b), and in Fig. 4a and b we plot the activation of each hidden unit as a function of vocabulary size when words were trained on lists (Simulations 1a and b).

The pattern of results is strikingly different across the four training conditions. When the model was trained on words one-at-a-time there is no pattern in any of the plots, and accordingly, no straightforward way to interpret the output of a given unit. That is, the hidden units code for the words in a distributed manner, regardless of the vocabulary size. By contrast, when the model was trained on lists of words, training set size had a clear impact on plots. That is, when trained on a small vocabulary there was again no discernible pattern (so again, the model was relying on distributed codes), but when trained on a large vocabulary, many hidden units showed a clear "banding" pattern of activation, with familiar and novel words taking on one of two distinct activation levels (see Fig. 4b). Furthermore, it is straightforward to interpret many of the bands given that all the familiar and unfamiliar words in a given band often contained a specific letter. For instance, consider hidden unit 18 that included one band of activation near 0, and another band of activation near 1. All the words that activated this unit contained the letter "K", whereas all words and words that did not contain a "K" failed to activate this unit. That is, this unit appears to be a localist detector for the letter 'K' (see Fig. 5).

To more formally assess the extent to which the network developed localist letter detectors we developed a "selectivity" metric that measured the extent to which a given hidden unit selectively responded to a given letter (this metric was also used in Bowers et al. (2014)). The selectivity of a hidden unit was computed as the minimal difference in activation between words that contained a given letter and words that did not contain this letter. These selectivity values can vary from +1 (when all words that contain a given letter drive the hidden unit to an activation of +1 and all words that do not contain the letter do not, i.e., $1 - 0 = 1$) to -1 (when all words that contain a given letter do not activate the unit at all, and all other words drive the unit to an activation of +1, i.e., $0 - 1 = -1$). In Fig. 6a–d we display the selectivity plots for the network trained on small and large vocabularies when words were presented one-at-a-time and in lists. We labeled a unit with a letter and its selectivity value when the selectivity was above .1. As is clear from these plots the model only learned selective units when trained on lists of words taken from a large vocabulary. In this condition the model learned selective codes for 23 of the 30 letters at a criterion of .1, with 33 selective codes altogether (10 of the selective codes coded a letter redundantly).

Interestingly, four of selective units had negative selectivity scores, meaning that the unit was less active in response to a given letter. We will call these OFF as opposed to ON units. At the .5 selectivity criterion there were selective codes for 14 of the 30 letters, with no redundant coding, and no OFF units.[3]

These scatter and selectivity plots help make sense of why generalization to novel words was so poor when the model was trained on lists of words taken from a small vocabulary and why

[3] In order to insure that the distribution of localist codes across the four training conditions was not due to some idiosyncratic feature of the above simulations we replicated all the simulations five times over. The only condition in which localist codes emerged was when the model was trained on lists of words taken from a large vocabulary. The average number of selective codes was 12.4 (SD = 2.61) with a selectivity criterion of .5, and 23.8 (SD = 3.35) with a selectivity criterion of .1.

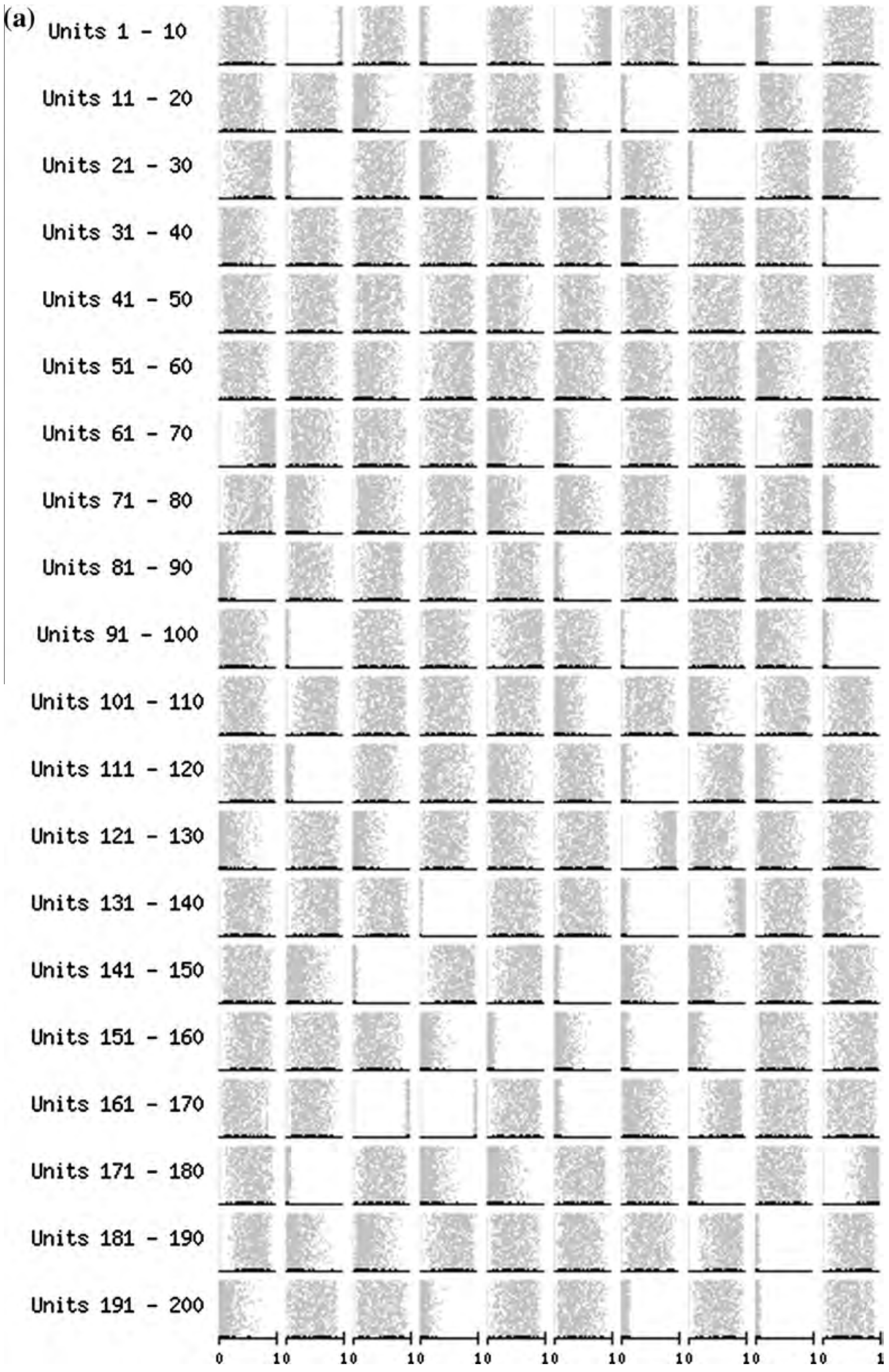

Fig. 3. Scatter plots of the 200 hidden units when the network was trained on words one-at-a time taken from (a) a small vocabulary of 30 words, (b) a large vocabulary of 300 words. Within each scatter plot, each dot represents the unit's response to a particular word. Dark dots refer to familiar words; light dots refer to novel words.

the model did so well when trained on lists of words taken from a large vocabulary. In the former case, the model succeeded to recall lists of familiar words using distributed representations, and under this condition, the model appears to have adopted a lexical bias in order to minimize the level of ambiguity associated with blends, much as Botvinick and Plaut (2006) claimed. This restricted generalization. By contrast, in the latter condition, the model abandoned distributed representations and succeeded on lists of familiar words using learned localist codes. Localist codes are not ambiguous (they avoid the superposition constraint) and accordingly the pressure to learn a lexical bias was not longer operative, and the model could generalize.

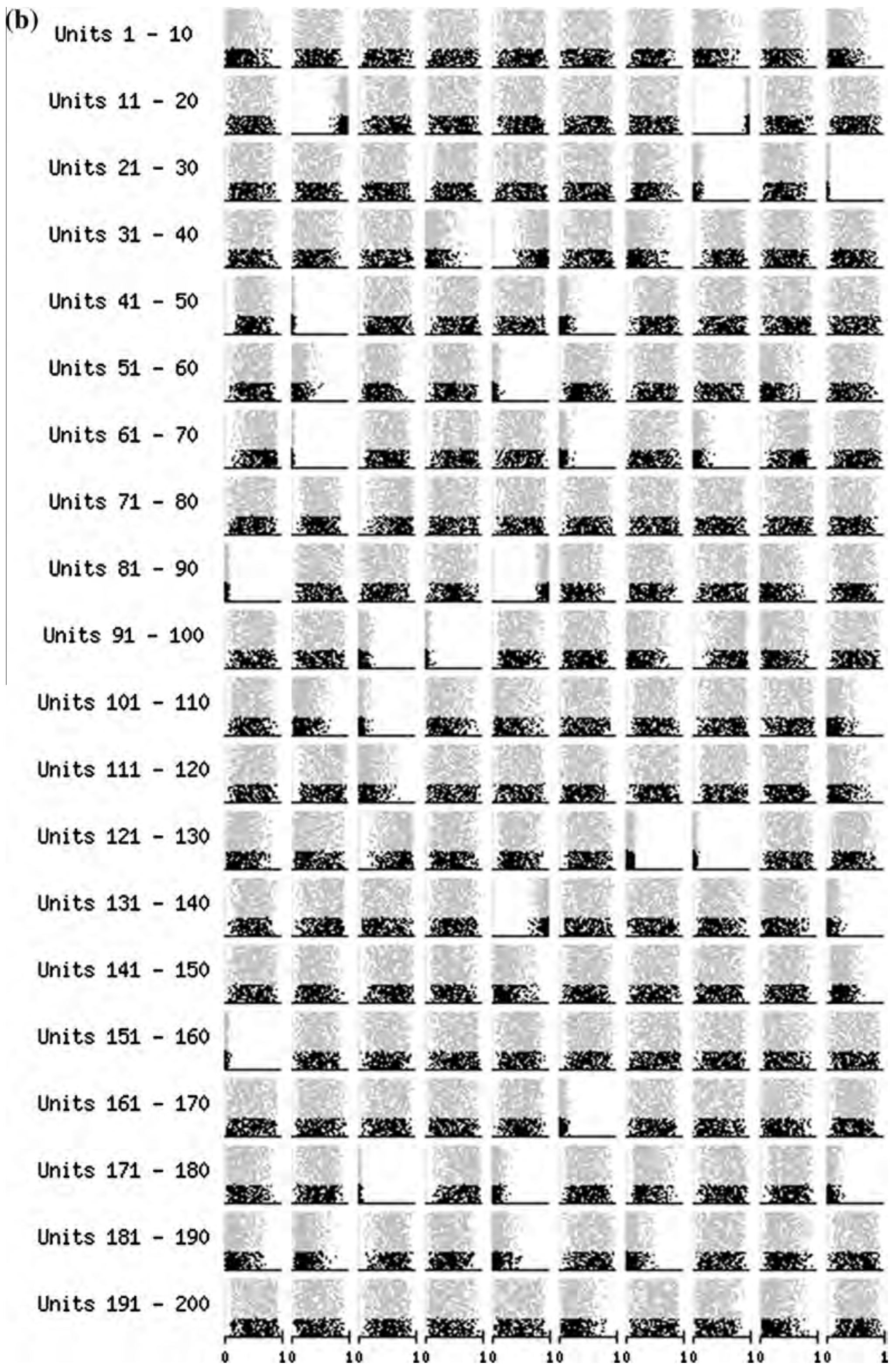

Fig. 3 (*continued*)

Why did the model learn localist codes when trained on a large vocabulary? Our hypothesis is that when the model was trained on the large vocabulary the blends of distributed patterns were too ambiguous to resolve with a lexical bias. That is, a given blend pattern could be produced by many possible combinations of familiar words, and as a consequence, a bias to retrieve the most likely set of words was no longer a successful strategy. Accordingly, the model adopted the only solution it could; namely, it learned localist letter codes. On this analysis, the model learned localist representations in response to the superposition constraint rather than any pressure to generalize. But a side effect of learning localist codes is that the model no longer relied on a lexical bias in order to

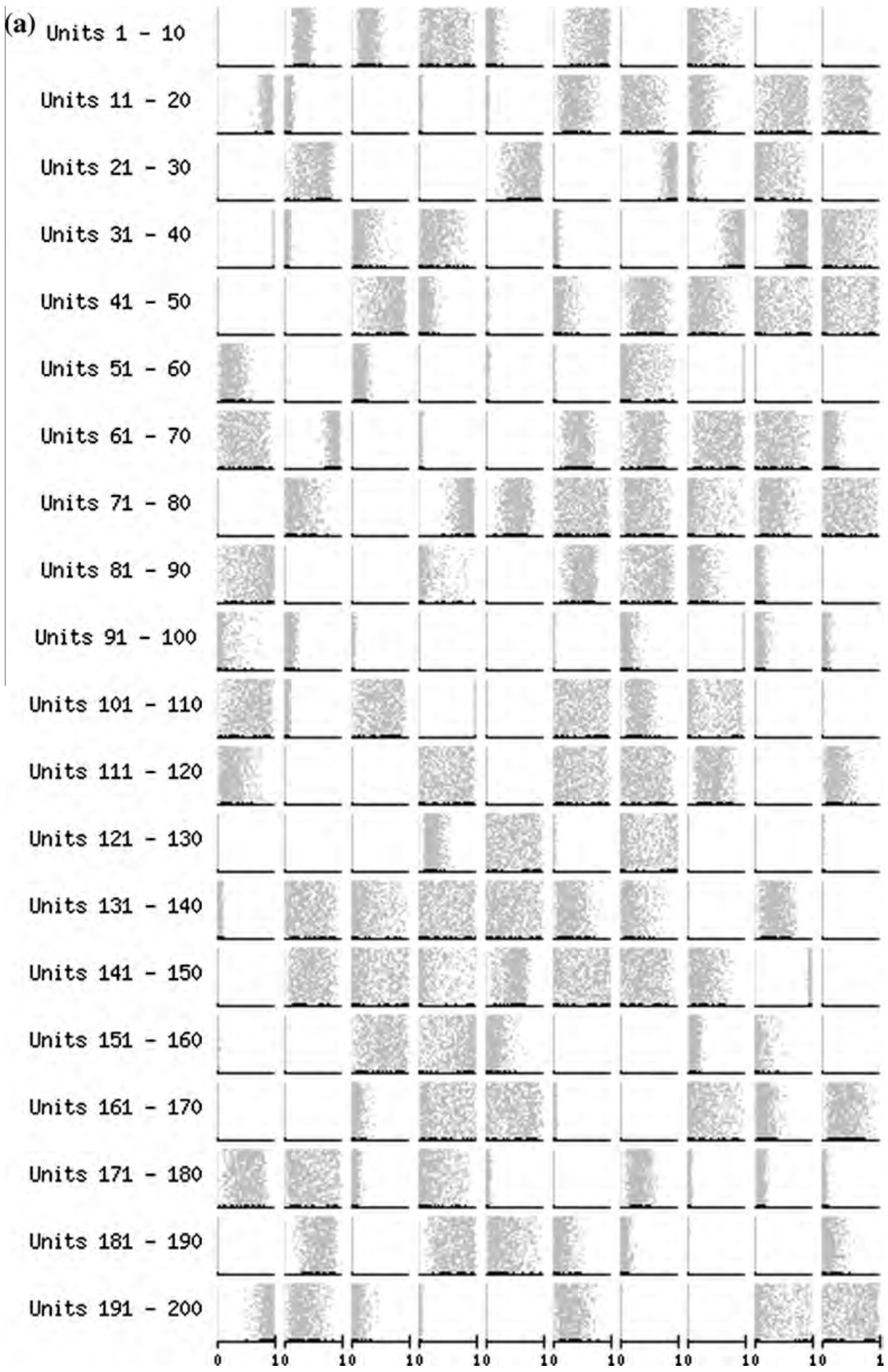

Fig. 4. Scatter plots of the 200 hidden units when the network was trained on lists of words taken from (a) a small vocabulary of 30 words, (b) a large vocabulary of 300 words.

disambiguate co-activated distributed patterns, which in turn facilitated generalization. This suggests that localist codes are not only better at supporting the co-activation of familiar words, but in addition, localist codes are better at supporting generalization in this context.

4.2. Lesion studies

In order to better understand the functional role of the highly selective hidden units we carried out a systematic set of lesion studies on the network trained on lists of items taken from the

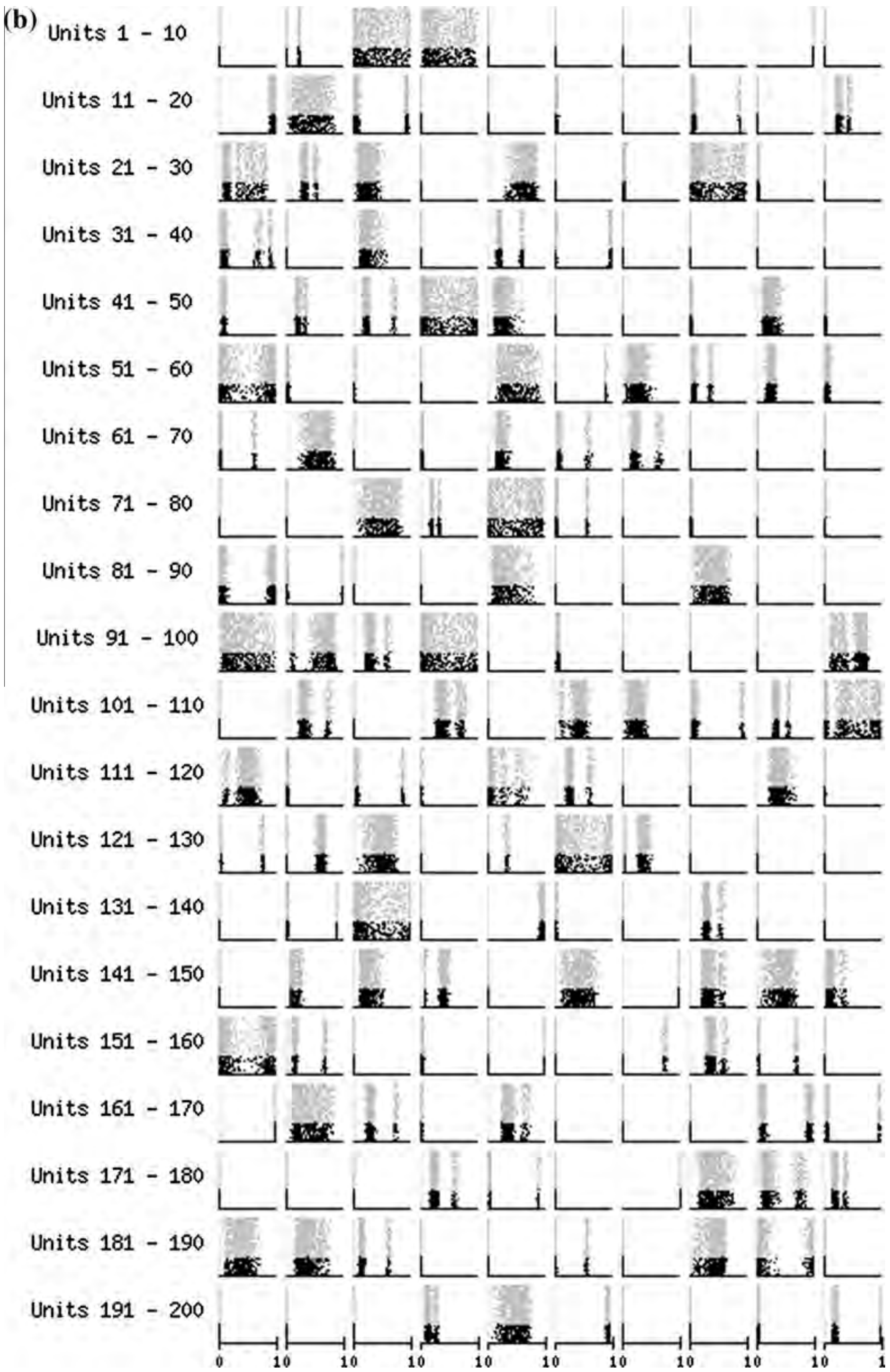

Fig. 4 (*continued*)

large vocabulary. We first lesioned each hidden unit that had a selectivity score above .1 and assessed the model's performance on all 1000 words (300 words and 700 novel words) when presented one-at-a-time during the recall phase (analogous to presenting an image in a single-cell recording study). We excluded errors that were due to failure to produce the end-of-list symbol, and we calculated the proportion of times the network performed as expected on the assumption that the selective units were necessary for the recall of the item (e.g., a selective unit for the letter A is necessary for recalling all words that contain the letter A). Note, the assumption that a given unit is necessary for recall is not a core claim of localist theories of perception nor "grandmother cell" theories of the brain. Indeed, on any biologically plausible grandmother theory there would need to be multiple redundant codes

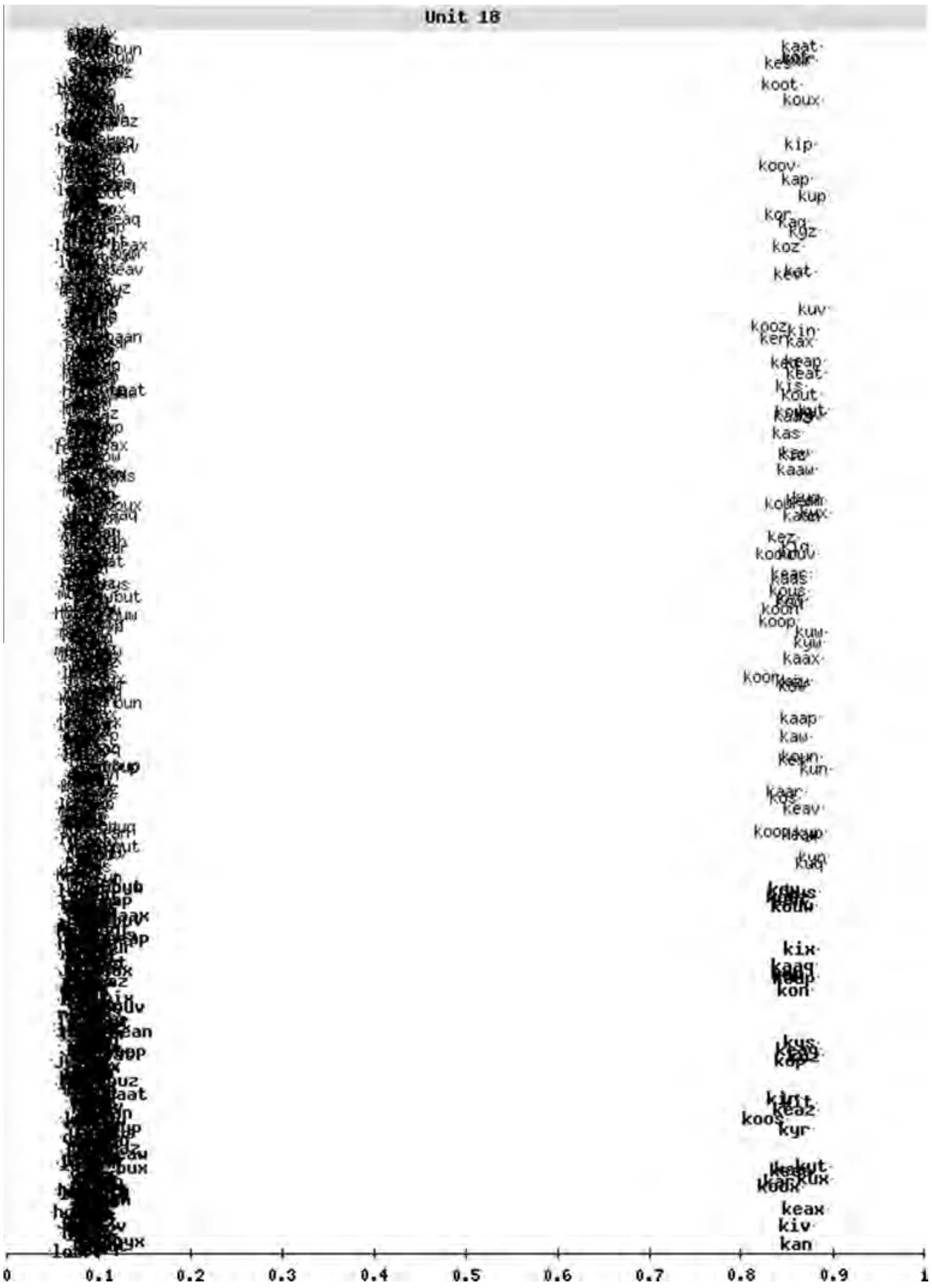

Fig. 5. Labeled scatter plot of hidden unit 18 following training on lists of words taken from a large vocabulary. In this plot the word names themselves are displayed, with familiar words depicted in bold.

of a given thing and accordingly, lesioning a single neuron would have little impact on perception (e.g., Barlow, 1985; Bowers, 2009; Gross, 2002; Page, 2000). Nevertheless, in the current simplified context, it is interesting to assess the extent to which single units are not only selective, but also necessary for performance. This is not something we examined in our previous work (Bowers et al., 2014).

The predictions are clear for the ON units, and less so for the OFF units. For ON units, the unit appears to code for a specific letter by being on, and accordingly, lesioning an ON unit should lead to a failure in recalling words that contain the corresponding letter (predicted failures) and should not impair performance on words that do not contain that letter (predicted successes). By contrast, for the OFF units, it is possible to interpret the unit in two different ways. On the one hand, the unit might be seen as coding for a specific unit by being off, in which case the predictions are the same; that is, lesioning an OFF unit should also lead to a failure of recalling words that contain the letter (predicted failures) and should not impair performance on words that do not contain the letter (predicted success). On the other hand, an OFF cell might be interpreted as part of a highly distributed code that represents all letters apart from a given letter. For instance, Unit 122 in Fig. 6d might be an anything-but-O unit. In which case, lesioning an OFF unit should not impair performance on the corresponding

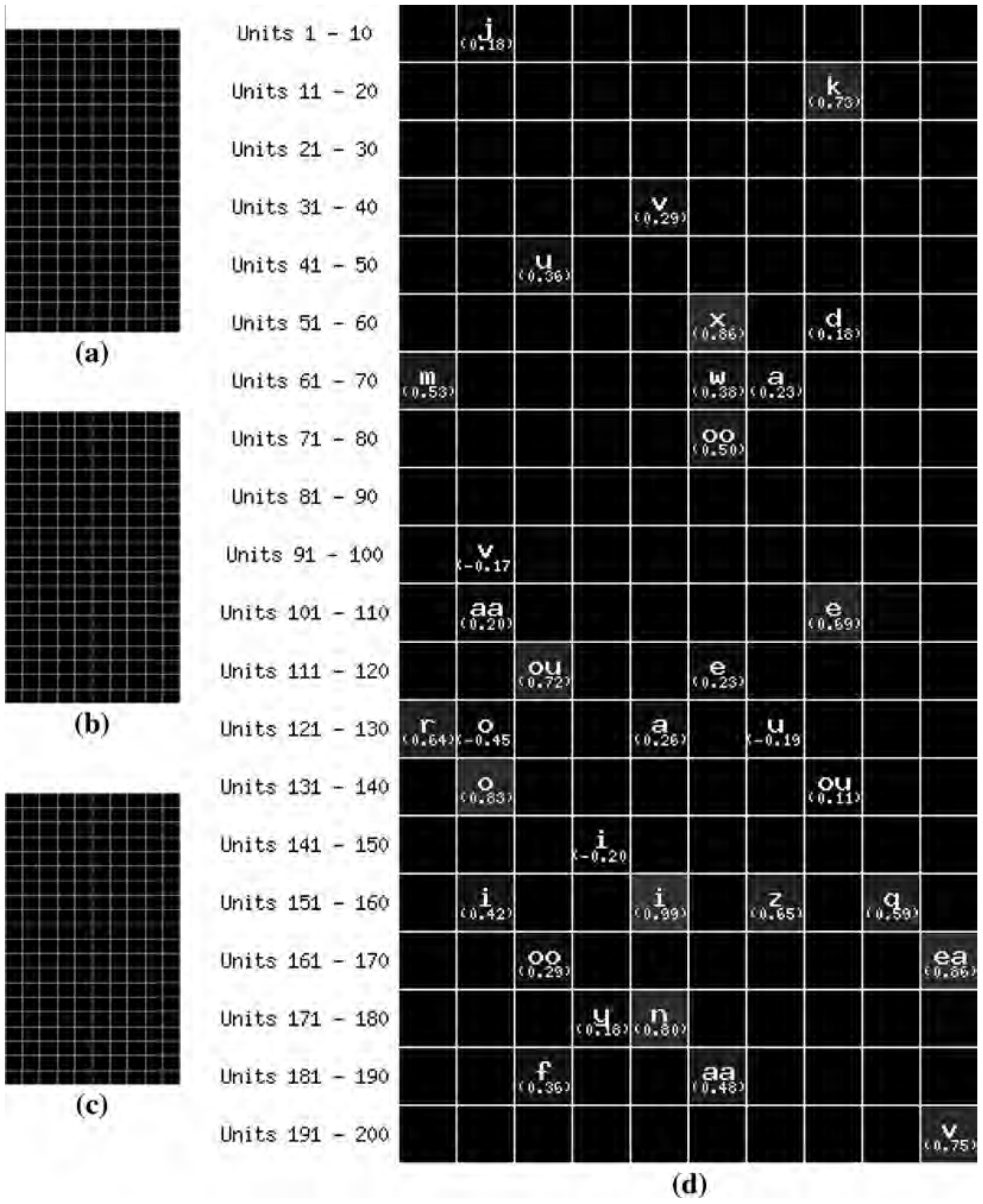

Fig. 6. Selectivity plot for the network trained on words one-at-a-time taken from the small (a) and large (b) vocabularies and trained on lists of words taken from the small (c) and large vocabularies (d). Each hidden unit is coded by a square (10 per row), and gray squares indicating selectivity scores above .5. Units that taken on selectivity values greater than |.1| are labeled with the letter they selectively respond to, and the precise selectivity value is presented in brackets. Units only became selective when trained on lists of words taken from a large vocabulary.

(predicted success) and fail on all other letters (predicted failures). Table 1 presents the proportion of predicted failures and successes for the selective units depicted in Fig. 6 on the assumption that OFF units are part of a distributed code ("anything-but" units) that do not uniquely represent a specific word.

In almost all cases the predicted successes were confirmed. For the ON cells, lesions had no impact on performance of words that did not contain the letter. Or to put it another way, the baseline chance of failing on words that do not contain the selective letter unit is close to 0%. Similarly, for the four OFF cells, lesions had no impact on performance for words that did contain the letter. However, this result in itself is not so critical, as the same result might be predicted by a fully distributed coding scheme, given that lesions are thought to produce "graceful degradation". As such, lesioning one of 200 hidden units might be expected to have little impact on performance.

The critical question, then, is whether the model failed as predicted following a lesion of a single hidden unit. Here the predictions were far from perfect, but nevertheless, it is striking how frequently these lesions impaired performance as predicted. First consider the ON cells. In some cases the predictions were perfect (e.g., following a lesion to unit 152 that was an ON cell selectively coding for the letter "i" the model failed 100% of the time with words that contained the letter "i"), and in eight cases lesions had no impact (performance was perfect following a lesion to a selective unit). Interesting, in four of these case the model had

Table 1
Performance of the network following lesions to the selective units (found in Fig. 6).

Unit no.	Local code	Selectivity	Predicted successes (%)	Predicted failures (%)
67	a	0.23	99	27
125	a	0.26	99	9
102	aa	0.20	99	8
186	aa	0.48	100	0
58	d	0.18	99	19
108	e	0.69	100	35
116	e	0.23	99	9
170	ea	0.86	99	16
183	f	0.36	99	81
144	i	−0.20	99	4
152	i	0.42	99	100
155	i	0.99	99	0
2	j	0.18	100	0
18	k	0.73	99	33
61	m	0.53	99	28
175	n	0.80	99	0
122	o	−0.45	100	100
132	o	0.83	100	1
76	oo	0.50	100	9
163	oo	0.29	99	48
113	ou	0.72	99	29
138	ou	0.11	99	1
159	q	0.59	100	34
121	r	0.64	99	45
43	u	0.36	99	55
127	u	−0.19	100	6
35	v	0.29	100	0
92	v	−0.17	100	97
200	v	0.75	99	0
66	w	0.38	100	74
56	x	0.86	99	0
174	y	0.18	100	69
157	z	0.65	100	0

learnt redundant localist letter codes. Overall the model failed 25% of the time as predicted following a lesion of 1 unit out of 200 units. This is very different than the near 0% chance of failing on words that did not contain the selective letter following lesions.

Similarly, with regards to the OFF cells, the predicted failures were far from perfect but revealing. Two units acted just as predicted, such that the model failed on all words that did not contain the letter. For example, after lesioning unit 92 the model failed on 97% of the words that did not contain the letter "v" (that is, most words). The predicted failures for the other two OFF units did not materialize, and the model succeeded for the vast majority of words. Still, the overall pattern of results suggests that the OFF cells tended to code for all the letters but one; that is, "anything-but" units. Accordingly, it is probably best not to consider these four OFF units localist as they do not appear to be representing a specific letter.

Why was the correspondence between lesions and performance less than perfect? Part of the answer may be that the model often learned redundant codes for letters (10 redundant units above the .1 selectivity metric), and these redundant codes may have supported performance following a lesion to a specific unit on some occasions. For example consider the success of the model on words containing the letter "i" following a lesion to unit 155 that selectively coded for "i". Perhaps the reason the model continued to succeed on this item was that unit 152 also coded the letter "i". It should be noted, however, that this was not always the case. For example, unit 56 was the only unit that selectively responded to the letter "x" and nevertheless the model succeeded in retrieving words containing the letter "x" following the lesion of this unit. Accordingly, the model is not always relying on localist units to perform, and it seems likely that the model has learned some sort of distributed code for the letter "x" in addition to the localist × unit. Nevertheless, the bottom line is that the model learned localist codes for most letters (defined as units that selectively responded to a given letter), and in the many cases, the selectivity measures have predictable functional consequences following lesions to single units. Again, the level of selective impairment we observed is greater than what would be predicted by any biologically plausible "grandmother cell" theory (no advocate of grandmother cells imagines that removing a single neuron from cortex would result in the selective loss of a word). These lesion studies highlight how important the learned localist representations are to the model's performance.

4.3. Selectivity vs. sparseness

As noted in the introduction, representations in neural networks (and brains) can be characterized along two separate dimensions, namely, selectivity and sparseness. Selectivity is associated with the interpretability of single units. On one extreme of this continuum, a given unit or neuron responds highly selectively things (e.g., specific faces, objects, and words) such that the output of that unit can be interpreted unambiguously. Units of this sort are sometimes called localist units or "grandmother cells". On the other extreme, each unit responds to a wide range of quite different things, such that it is impossible to unambiguously interpret the output of a given unit. This selectivity dimension is conceptually distinct from sparseness, which refers to the proportion of units in a network that respond to a given input. On one extreme, a single unit responds to an input (what we might call ultimate-sparse coding), and at the other extreme, a high proportion of units are active at any point in time (a dense representation). In general sparseness and selectivity are correlated, but it is possible for these measures to dissociate (Földiák, 2009).

The analyses reported above characterized the selectivity of hidden units, and it is also worth considering the role (if any) that sparseness played in supporting generalization and STM. In order to assess the potential role of sparseness to the solution we modified the scatter plots so that each plot corresponds to a word (rather than a hidden unit), and each cross in each plot corresponds to the activation of a hidden unit (with 200 crosses). Once again, activation is plotted on the x-axis, and along the y-axis we plotted the hidden units in sequence (e.g., 1, 2, 3, etc.) so that points did not overlap. This provides a depiction of how many (and which) hidden units are active in response to a given word, and to what extent.

In Fig. 7a we include the scatter plots for the 30 trained words when the model was trained on lists of words taken from the small vocabulary, and in Fig. 7b we included scatter plots for the same 30 words when the model was trained on lists of words taken from the large vocabulary. Note, we only included the plots of these 30 words even when the model was trained on the large vocabulary because of space restrictions (it is not practical to include plots for all 300 trained words), but the same patterns occur for the non-displayed words as well.

The most noticeable feature of the sparseness plots is that there is no qualitative change (unlike the striking difference in the selectivity plots). There was a small reduction in the percentage of hidden units active above .5 in response to a given word when the model was trained on the large compared to small vocabulary (20% and 14%, respectively), and this can be attributed to the fact that many of the selective ON units were not highly activated in response to most of the words (other than the words that contained a specific letter). That is, the learned selectivity of the ON units is the likely cause of the small reduction in the sparseness measures in the large vocabulary condition (and indeed, as noted above, sparseness and selectivity often do go together). But it is clear from these figures that it is the selectivity rather than the

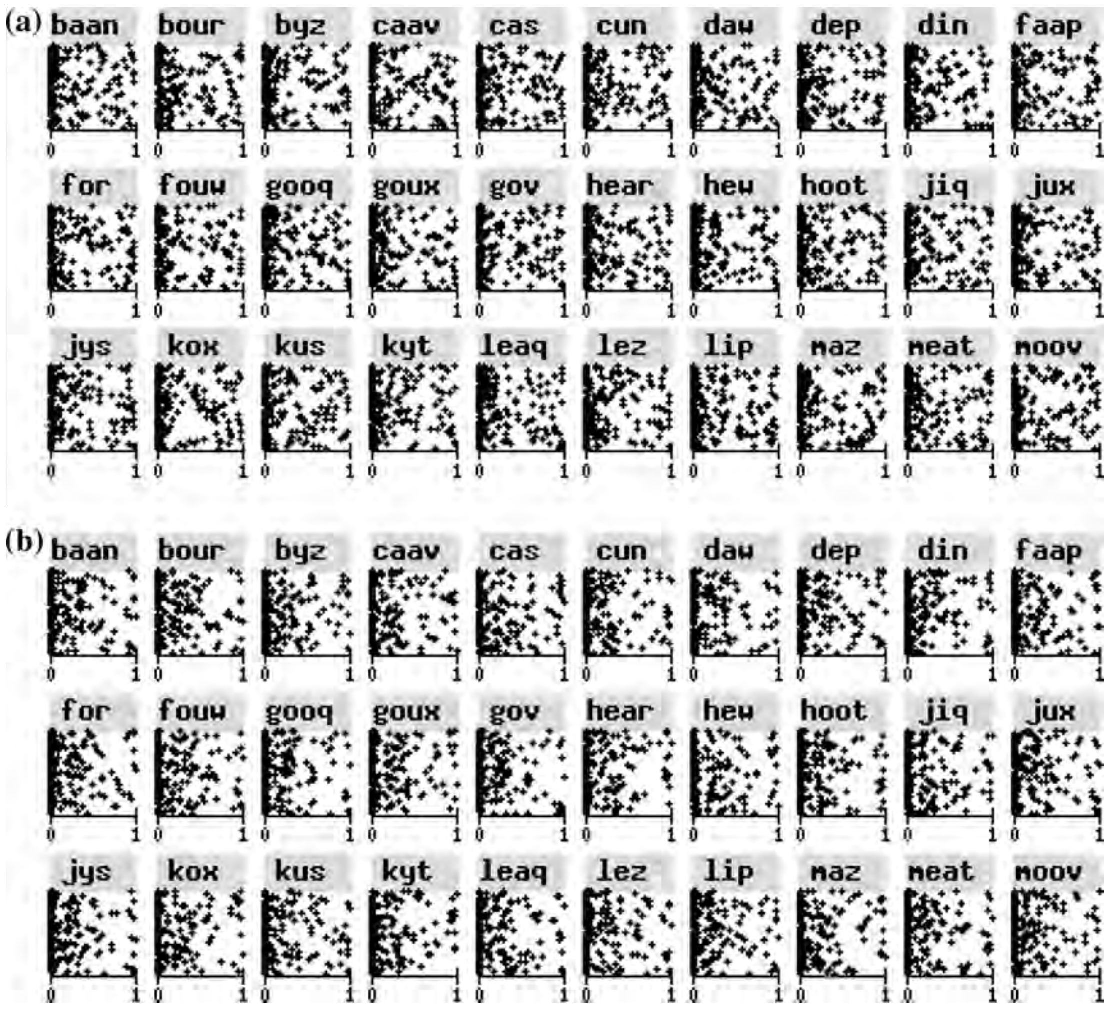

Fig. 7. Sparseness plots of the network trained on lists of words taken from the (a) small and (b) large vocabularies. Each plot refers to a word rather than a hidden unit and only the plots of the words from the small vocabulary are presented. Each cross refers to a hidden unit, with level of activation along the *x*-axis (from 0 to 1) and hidden unit number (1–200) organized along the *y*-axis (200 crosses per plot).

sparseness that is associated with the model's success, a conclusion that is further supported by the lesion analyses.

5. Simulation 3 (a–d): making predictions

Based on the simulations above we hypothesize that localist or highly selective representations are better than distributed representations in coping with the superposition catastrophe. When a model only needs to co-activate items taken from a small vocabulary (when the superposition only leads to modest ambiguities) then distributed codes can partially cope by adopting a lexical bias, a strategy that compromises generalization. But this strategy fails when a model needs to co-activate many items take from a larger vocabulary (when the superposition leads to substantial ambiguities). In the latter case, the model learns localist codes, and these codes support generalization.

If this is correct, and the results do not reflect some idiosyncratic feature of the above simulations, then we should be able to predict the types of representations learned in previous published models that failed in generalizing (Bowers et al., 2009, Simulation 7) or succeeded in generalizing (Botvinick & Plaut, 2009; Bowers et al., 2009, Simulation P2). That is, despite the different parameters of the models (number of hidden units, specific training conditions, etc.), the former model should have learned few if any selective units whereas the latter models should have learned many selective units.

In order to test these predictions we replicated these simulations and then carried out the single unit recordings as above. In Simulation 3a we replicated the Bowers et al. (2009) Simulation 7 simulation that failed to generalize. This model was trained to recall lists of letters (up to 9 letters) taken from a vocabulary of 25 letters. Each letter was coded as a random pattern of five active units across 26 input units, and after training, the model recalled lists of 6 letters at over 50% accuracy. Critically, however, performance on lists that contained a novel letter (an untrained pattern of five active units) in any position was near floor. After re-running this simulation we carried out single-unit recordings in the hidden layer. No selective units at the .5 selectivity criterion were found. This is consistent with our prediction that generalization requires localist codes.

In Simulation 3b we replicated the Botvinick and Plaut (2009) model that was able to generalize. This model included an input and output layer with three sets of 10 units, representing the onset, nucleus, and coda, respectively, and a hidden layer with 75 units. A syllable was represented by activating one unit from each of these groups, and the model was trained on 999 of the 1000 possible syllables on lists ranging from length one to three. Botvinick and Plaut showed that this model was able to generalize to lists containing the non-studied syllable. Again we re-ran this simulation and carried out single-unit recordings in the hidden layer. In this case we found 34 units selective at a .1 selectivity criterion and 18 at the .5 selectivity criterion, as can be seen in Fig. 8. This is again

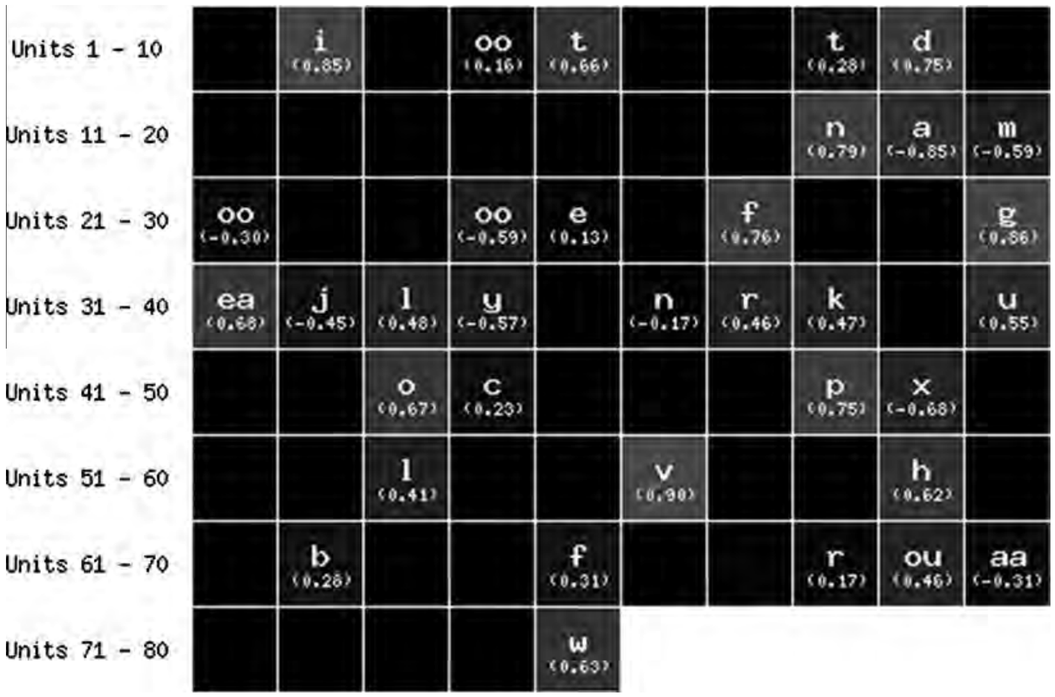

Fig. 8. Selectivity plot of 80 hidden units when we replicated Botvinick and Plaut's (2009) simulation in which the model was trained on lists of syllables taken from a large vocabulary. Each hidden unit is coded by a square (10 per row), and gray squares indicating selectivity scores above .5. Units that taken on selectivity values above .1 are labeled with the letter they selectively respond to, and the precise selectivity value is presented in brackets.

consistent with our prediction that localist coding is associated with successful generalization.

In Simulation 3b we replicated Bowers et al. (2009) Simulation P2 that included 10 units reserved for onsets, the next six for vowels, and the final 10 units for codas (resulting in 10 × 6 × 10 or 600 possible syllables), as well as 200 hidden units. The model was trained on 500 syllables on lists ranging from length one to nine and was able to recall lists of familiar and unfamiliar syllables at a similar rate. After training the model learned 24 codes at the .1 selectivity criterion and 12 local codes at the .5 selectivity criterion.[4]

In addition, if our analysis is correct, localist codes were learned in response to the superposition constraint rather than the overall difficulty of learning the task. However, given the findings reported thus far, the latter hypothesis cannot be ruled out: The models that learned localist codes were not only trained under conditions in which the superposition was the most ambiguous, but in addition, the number of training trials needed to support good performance was much greater than in the other conditions. Accordingly, it is possible that the additional training led to more selective coding rather than the superposition catastrophe per se.

In order to assess the impact of learning difficulty we trained the model to recall single items taken from the large vocabulary but made the task of recalling items more difficult by reducing the model's resources. That is, we reduced the number of the hidden units from 200 to 100, 50, 25, and finally 10 (Simulation 3d). Learning this task should become progressively more difficult, but the superposition constraint never arises (given only one item is retrieved at a time).

The network was trained in the same way as in Simulation 2b, and training was stopped when the network performance reached 100% on the familiar words. As can be seen in Table 2, the task did indeed become much more difficult with fewer hidden units, but no localist codes developed in any of the conditions (there was just one selective unit at .1 level in the 100 hidden units condition). This suggests that the difficulty of learning the task (and the associated increase in training) was not responsible for the development of localist codes. It is also worth noting Bowers et al. (2014) found localist codes in a "pure superposition" condition in which multiple items were co-activated and recalled without regard to order. Together, these findings suggest that it is the superposition constraint per se. that is responsible for the localist coding, rather than the amount of training or the requirement to code for items in the correct order.

Table 2
Number of units with a selectivity scores above .5 when model was trained on words one-at-a-time as a function of number of hidden units.

Number of hidden units	Number of training trials	Performance on novel words (%)	Number of selective codes at .5
100	100,000	99.76	0
50	200,000	96.70	0
25	700,000	88.28	0
10	10,000,000	83.94	0

6. General discussion

The principal result of the simulations is that a PDP model of immediate serial recall only succeeded in recalling lists of familiar and novel words when it learned localist letter codes. This provides an important extension to Bowers et al. (2014) who also highlighted the emergence of localist codes in response to the superposition catastrophe, but did not consider whether these selective codes could support generalization. It is commonly assumed that PDP models generalize on the basis of learned distributed representations, but clearly, PDP models can learn localist codes, and these representations support generalization in some contexts.

[4] We also analyzed the hidden units from the Botvinick and Plaut (2006) model that was trained on a small vocabulary of letters and that could not generalize. Consistent with our analysis, it did not learn any localist unit at the .5 selectivity criterion.

Our claim that the superposition catastrophe provides a computational pressure to learn selective codes in the cortex complements the long-standing claim that catastrophic interference provides a computational pressure to learn sparse and selective codes in hippocampus (e.g., Marr, 1971). It is often hypothesized that selective coding is restricted to the hippocampus (e.g., McClelland et al., 1995), but the neuroscience suggests otherwise (Bowers, 2009), and the current simulations provide a potential explanation as to why selective coding is observed in cortex as well. That is, various perceptual and cognitive systems in cortex support both STM and generalization, and selective coding schemes are best suited for this.

As noted in the introduction, our findings may have implications for how knowledge is coded outside STM tasks. It is widely assumed that common representations support STM and the perception of single letters, words, objects, and faces (e.g., Page et al., 2007); indeed, we are not aware of any theories according to which the representations of words, objects, faces are coded in one format (e.g., localist) for the sake of STM and another format (e.g., distributed) for the sake of perception. If indeed common representations support both tasks, then our STM results provide strong constraints on the types of representations that support the perception of single items. That is, the representations for perception may be localist because these same representations need to be co-active with other items in STM. This is indeed what we found in our simulations. Although the model was trained to encode multiple words at the same time, our single-unit recordings and lesion studies were carried out when the model was presented with words (and novel words) one-at-a-time.

It is important to emphasize that there are undoubtedly multiple constraints determining the types of representations that are best suited for a given task. We have found that STM provides a strong pressure to learn localist representations in artificial neural networks, and it will be important to better characterize additional pressures that may also play a role explaining the selective codes that have been observed in cortex. Indeed, our findings are consistent with a range of additional computational (e.g., Masquelier, Guyonneau, & Thorpe, 2009; Page, 2000) and biological (e.g., Lennie, 2003) advantages of selective (and sparse) codes that might also help explain the development of selective and sparse coding in cortex.

6.1. A summary and explanation of the main findings

The overall pattern of our simulation results and analyses are quite complex, so we review and explain the findings in some detail, and then consider the wider implications.

The network performance of the models and hidden unit analyses highlight two limitations with distributed representations when co-activating multiple items in STM: One limitation associated with novel words, another with familiar words. First, consider the set of results obtained with novel words. When the network was trained to recall lists of words taken from a small vocabulary (Simulation 1a) the model learned distributed representations, and under these conditions, the model catastrophically failed at recalling novel words. Importantly, this failure could not entirely be attributed to the small training set, as the model was much better at recalling novel words when trained on the same set of words one-at-a-time rather than in lists (Simulation 2a). That is, when the network learned distributed representations it could either generalize to novel words (this was found after training the model on words one-at-a-time) or recall multiple familiar words (this was found after training on lists of words taken from a small vocabulary), but not both. This attribute of distributed representations might be called the "generalization-superposition trade-off".

The underlying cause for this trade-off was in fact identified by Botvinick and Plaut (2006) who first showed that a PDP model of immediate serial recall could recall lists of familiar letters based on co-activated and superimposed distributed representations. They noted that a blend of co-activated distributed letters is indeed ambiguous, but argued that these blends can sometimes be decomposed into the correct set of letters by adopting a bias to recall the most likely sequence of letters given its training history. However, Botvinick and Plaut did not consider the downside of this solution, namely, that the bias works against recalling novel items. That is, the bias that allows the model to recall lists of familiar words works against generalizing to novel words.

The second manifestation of the superposition catastrophe was observed with familiar words. That is, our network learned localist representations for letters when trained to recall lists of words taken from a large vocabulary (Simulations 1b). Our interpretation of this result is that superimposed distributed representations, even when combined with a lexical bias, were no longer sufficient to support good performance with familiar words. The problem was there were too many possible sequences of the familiar words that would produce a given blend, and as a result, a lexical bias was no longer helpful. Under these conditions, the models gave up on fully distributed coding (the source of the problem), and instead, started learning localist representations. The behavioral manifestation of this selective responding was dramatic: Performance went from catastrophic failure with novel words to striking success (with performance on the novel words approaching performance with the familiar ones). A key reason why generalization improved with the emergence of localist letter codes is that the lexical bias solution to the superposition catastrophe was no longer operative, and as a consequence, the model was no longer biased against outputting novel patterns. In sum, the model learned localist representations in response to a greater superposition constraint, and this in turn facilitated generalization.

Finally, we have also shown that the sparseness of the learned representations played little or no role in solving the superposition catastrophe. Indeed, the model learned dense representations in all training conditions. This result takes on added importance given that some researchers have argued against localist or "grandmother cell" coding in cortex or hippocampus on the basis that too many neurons fire in response to a given input (e.g., Waydo, Kraskov, Quian Quiroga, Fried, & Koch, 2006). Our simulations show the flaw in this logic. We found localist codes for letters, and at the same time, 14% of the units were active above .5 in response to a given input (in vivo it is estimated that less than 1% of neurons are active at any one time; Lennie, 2003). So again we found localist (grandmother cell) codes in the context of a single input activating many units (dense coding scheme). For more discussion of the contrast between sparseness on one hand and selectivity on the other, see Földiák (2009) and Bowers (2011).

6.2. Are there some conditions in which distributed representations can support immediate serial recall and generalization?

Although we have found that a family of closely related recurrent PDP models of immediate serial recall relied on localist letter codes in order to generalize to novel words, our claim is not that distributed codes can never support successful performance. In fact there are existing models of immediate serial recall that do just this. But in all cases the models succeed by avoiding the problem of superposition rather than actually solving the constraint.

For instance, in some models of immediate serial recall, only one item is activated at a time (avoiding the superposition problem altogether). This is achieved by relying on weight-based (as opposed to activation based) methods for encoding and maintaining serial order information. For example, in the SOB model of

Farrell and Lewandowsky (2002) each to-be-remembered item is encoded in the connection weights of the network, with greater weights associated with items presented at the start of the list. At recall the network settles into a sequence of different active states as a function of the size of the weights. Critical for the present purposes, at no point are multiple different words co-activated. Similarly, a number of connectionist models of STM include separate representations of items (e.g., the word DOG) and order (e.g., position 1, position 2, etc.), and items are assigned an order within a list by changing the connection weights between item and order representations (e.g., Brown, Preece, & Hulme, 2000; Burgess & Hitch, 1999). Although these so-called "context models" differ in various ways, the key claim is that the units coding for order are recalled in sequence, and these representations in turn retrieve the items. Again, this weight-based solution does not involve co-activating multiple items at the same time, and accordingly, it is possible for these networks to encode a sequence of items (including novel items) using distributed representations without suffering from the superposition catastrophe.

It may also be possible to use activation-based networks to avoid the superposition catastrophe although no model of this sort has been applied to immediate serial recall tasks. For example, it has been argued that temporal synchrony is an important mechanism for supporting STM (e.g., Hummel & Holyoak, 1997), and this might potentially provide a mechanism that could support immediate serial recall. To illustrate, imagine a situation in which JOHN is coded as one distributed pattern of activation and PAUL is coded as another. If a network can learn to oscillate between the representations of PAUL and JOHN, with PAUL activated briefly followed by JOHN, followed by PAUL, etc., then the oscillating patterns can be used as an unambiguous representation of JOHN and PAUL in STM, and that this in turn could be adapted for purposes of the immediate serial recall test. But again, this model is avoiding the superposition catastrophe by ensuring that only one distributed representation is active at a given point in time.

Is it possible to overcome the superposition constraint in a model that codes for multiple items at the same time over the same set of units? For example, would it be possible to adapt the current PDP model so that it succeeds recalling lists of familiar and unfamiliar words relying on distributed as with localist codes? We cannot rule out this possibility, and future work may identify conditions in which PDP models do indeed co-activate multiple familiar and unfamiliar items. But it is striking that we took PDP models that are assumed to learn distributed representations and showed, as predicted on the basis of the superposition constraint (Von der Malsburg, 1986), that they succeeded only when they learned localist codes. It is often claimed that a key advantage of PDP models is that the learned representations are emergent rather than "stipulated" by the modeler (Plaut & McClelland, 2000), and the fact that previous PDP models (trained on items one-at-a-time) learned distributed codes was thought to provide evidence for computational advantages of distributed coding. In the same way, the emergence of localist coding in our models suggest localist coding is an efficient solution to the task demands we imposed. It is important to emphasize that we are not using these simulation results as evidence for the existence of highly selective neural codes in cortex (the evidence comes from neuroscience). Rather, our findings provide some insight into why these neurons exist.

6.3. Implications for the complementary learning systems (CLS) hypothesis

A fundamental claim of the CLS hypothesis is that information in the hippocampus is coded with highly selective neurons whereas information in the cortex is coded in a highly distributed manner. In the initial version of this hypothesis (McClelland et al., 1995), the claim was that fast learning and generalization are incompatible functions, with selective coding in hippocampus required for fast learning, and distributed coding in cortex required for generalization. On this hypothesis, the selective codes in hippocampus are used to teach the distributed codes in cortex in a slow process of consolidation. More recently, CLS hypothesis has been modified in response to two sets of findings. First, in response to data showing that newly acquired memories in the hippocampus can support generalization (Eichenbaum, 2004), Kumaran and McClelland (2012) argued that the sparse and selective codes in the hippocampus can support both fast learning and generalization. Indeed, they showed that generalization with sparse coding was possible in the hippocampus on the basis of recurrent connections. Second, in response to data showing fast learning in cortex (Tse et al., 2007, 2011), McClelland (2013) argues that fast learning is possible in cortex using distributed codes as long as newly acquired knowledge is consistent with previous knowledge. As a consequence of these modifications of the CLS hypothesis, the fundamental contrast is no longer between slow learning in the cortex that supports generalization and fast learning in the hippocampus that supports episodic memory, but rather, learning in the cortex and hippocampus that is more versus less dependent on prior-knowledge, respectively. Nevertheless, one of the fundamental claims of the original theory is still thought to hold, namely, that information in cortex is coded in a highly distributed format.

This claim also faces challenges both on empirical and computational grounds. With regards to the data, there is good evidence for sparse and selective coding in the cortex (cf., Bowers, 2009). Indeed, the selective responses of neurons in inferotemporal cortex identified by Logothetis, Pauls, and Poggio (1995) are every bit as selective as anything observed in the hippocampus. With regards to the computational considerations, the current simulations provide evidence that distributed representations are poorly suited for solving the superposition catastrophe, and that even PDP models learn localist codes when they succeed in the joint task of co-activating multiple items and generalizing (functions supported by the cortex).

We do not want to overstate our case. It may well be that the level of selectivity and sparseness varies in the hippocampus and cortex, and indeed, we are arguing that there are different computational pressures for learning selective codes in these different regions. It may also be the case that updated versions of the complementary learning systems hypothesis may be advanced in which cortex learns some combination of localist and distributed coding (as we observe in our simulations). But if selective codes in cortex support STM and generalization, then the underlying motivation and claims associated with CLS hypothesis needs to change. That is, the hippocampus may be teaching the cortex over the course of a few hours (perhaps during sleep; e.g., Dumay & Gaskell, 2007), but the hippocampus needs to teach the cortex highly selective codes that are suitable for solving the superposition catastrophe and generalization (at the same time).

7. Conclusion

The current simulations highlight a computational pressure to learn highly selective representations in cortex. Just as sparse and selective representations are better at learning quickly without suffering catastrophic interference (McClelland et al., 1995), highly selective representations are better at the dual tasks of coding multiple things at the same time and generalizing. Indeed, contrary to the standard view, PDP models that learned localist codes were much better at generalizing in this context. Given that various cortical system need to code for multiple things at the same time (e.g., Cowan, 2001), we would suggest that our findings help explain one of the key finding from 50 years of neurophysiology, namely, that information in cortex is coded in a highly selective manner.

Author note

This research was supported by Leverhulme Grant RJ5538, awarded to Jeffrey S. Bowers.

References

Barlow, H. B. (1985). The 12th Bartlett Memorial Lecture: The role of single neurons in the psychology of perception. *Quarterly Journal of Experimental Psychology: Section A. Human Experimental Psychology, 37*, 121–145.

Berkeley, I. S. N., Dawson, M. R. W., Medler, D. A., Schopflocher, D. P., & Hornsby, L. (1995). Density plots of hidden unit activations reveal interpretable bands. *Connection Science, 7*, 167–186.

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

Botvinick, M. M., & Plaut, D. C. (2009). Empirical and computational support for context-dependent representations of serial order: Reply to Bowers, Damian, and Davis (2009). *Psychological Review, 116*, 998–1001.

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

Bowers, J. S. (2009). On the biological plausibility of grandmother cells: Implications for neural network theories in psychology and neuroscience. *Psychological Review, 116*, 220–251.

Bowers, J. S. (2011). What is a grandmother cell? And how would you know if you found one? *Connection Science, 2*, 91–95.

Bowers, J. S., Damian, M. F., & Davis, C. J. (2009). A fundamental limitation of the conjunctive codes learned in PDP models of cognition: Comments on Botvinick and Plaut (2006*). Psychological Review, 116*, 986–995.

Bowers, J. S., Vankov, I. I., Damian, M. F., & Davis, C. J. (2014). Neural networks learn highly selective representations in order to overcome the superposition catastrophe. *Psychological Review, 121*, 248–261.

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.

Cowan, N. (2001). The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences, 24*, 87–114.

Dumay, N., & Gaskell, M. G. (2007). Sleep-associated changes in the mental representation of spoken words. *Psychological Science, 18*(1), 35–39.

Eichenbaum, H. (2004). Hippocampus: Cognitive processes and neural representations that underlie declarative memory. *Neuron, 44*, 109–120.

Farrell, S., & Lewandowsky, S. (2002). An endogenous distributed model of ordering in serial recall. *Psychonomic Bulletin & Review, 9*(1), 59–79.

Földiák, P. (2009). Neural coding: Non-localist but explicit and conceptual. *Current Biology, 9*, R904–R906.

Gross, C. G. (1994). How inferior temporal cortex became a visual area. *Cerebral Cortex, 5*, 455–469.

Gross, C. G. (2002). The genealogy of the "grandmother cell". *The Neuroscientist, 8*, 512–518.

Gross, C. G., Bender, D. B., & Roch-Miranda, C. E. (1969). Visual receptive fields of neurons in inferotemporal cortex of monkey. *Science, 166*, 1303–1306. December 5.

Grossberg, S. (1980). How does a brain build a cognitive code? *Psychological Review, 87*, 1–51.

Harm, M. W., & Seidenberg, M. S. (2004). Computing the meanings of words in reading: Cooperative division of labor between visual and phonological processes. *Psychological Review, 111*(3), 662.

Hummel, J. E., & Holyoak, K. J. (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review, 104*(3), 427.

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.

Kumaran, D., & McClelland, J. L. (2012). Generalization through the recurrent interaction of episodic memories: A model of the hippocampal system. *Psychological Review, 119*, 573–616.

Lennie, P. (2003). The cost of cortical computation. *Current Biology, 13*, 493–497.

Logothetis, N. K., Pauls, J., & Poggio, T. (1995). Shape representation in the inferior temporal cortex of monkeys. *Current Biology, 5*, 552–563.

Marr, D. (1971). Simple memory: A theory for archicortex. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 23–81.

Masquelier, T., Guyonneau, R., & Thorpe, S. J. (2009). Competitive STDP-based spike pattern learning. *Neural Computation, 21*, 1259–1276.

McClelland, J. L. (2013). Incorporating rapid neocortical learning of new schema-consistent information into complementary learning systems theory. *Journal of Experimental Psychology: General, 142*(4), 1190–1210.

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.

McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks. The sequential learning problem. In G. H. Bower (Ed.), *The psychology of learning and motivation*. New York: Academic Press.

Page, M. P. A. (2000). Connectionist modeling in psychology: A localist manifesto. *Behavioral and Brain Sciences, 23*, 443–512.

Page, M. P., Madge, A., Cumming, N., & Norris, D. G. (2007). Speech errors and the phonological similarity effect in short-term memory: Evidence suggesting a common locus. *Journal of Memory and Language, 56*(1), 49–64.

Plaut, D. C., & McClelland, J. L. (2000). Stipulating versus discovering representations. *Behavioral and Brain Sciences, 23*, 489–491.

Plaut, D. C., McClelland, J. L., Seidenberg, M. S., & Patterson, K. (1996). Understanding normal and impaired word reading: Computational principles in quasi-regular domains. *Psychological review, 103*(1), 56–115.

Seidenberg, M. S., & McClelland, J. L. (1989). A distributed, developmental model of word recognition and naming. *Psychological Review, 96*(4), 523–568.

Shoham, S., O'Connor, D. H., & Segev, R. (2006). How silent is the brain: Is there a "dark matter" problem in neuroscience? *Journal of Comparative Physiology: A. Neuroethology Sensory Neural and Behavioral Physiology, 192*, 777–784.

Tse, D., Langston, R. F., Kakeyama, M., Bethus, I., Spooner, P. A., Wood, E. R., ... Morris, R. G. M. (2007). Schemas and memory consolidation. *Science, 316*, 76–82. http://dx.doi.org/10.1126/science.1135935 (April 6).

Tse, D., Takeuchi, T., Kakeyama, M., Kajii, Y., Okuno, H., Tohyama, C., ... Morris, R. G. M. (2011). Schema-dependent gene activation and memory encoding in neocortex. *Science, 333*, 891–895. http://dx.doi.org/10.1126/science.1205274 (August 12).

Von der Malsburg, C. (1986). Am I thinking assemblies? In G. Palm & A. Aertsen (Eds.), *Brain theory*. Berlin: Springer.

Waydo, S., Kraskov, A., Quian Quiroga, R., Fried, I., & Koch, C. (2006). Sparse representation in the human medial temporal lobe. *Journal of Neuroscience, 26*, 10232–10234.

Werbos, P. J. (1990). Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE, 78*(10), 1550–1560.

Willmore, B., & Tolhurst, D. J. (2001). Characterizing the sparseness of neural codes. *Network: Computation in Neural Systems, 12*, 255–270.