Scaffolded input promotes atomic organization in the recurrent neural network language model

Philip A. Huebner and **Jon A. Willits**
Department of Psychology
University of Illinois at Urbana-Champaign
{huebner3, jwillits}@illinois.edu

Abstract

The recurrent neural network (RNN) language model is a powerful tool for learning arbitrary sequential dependencies in language data. Despite its enormous success in representing lexical sequences, little is known about the quality of the lexical representations that it acquires. In this work, we conjecture that it is straightforward to extract lexical representations (i.e. static word embeddings) from an RNN, but that the amount of semantic information that is encoded is limited when lexical items in the training data provide redundant semantic information. We conceptualize this limitation of the RNN as a failure to learn atomic internal states - states which capture information relevant to single word types without being influenced by redundant information provided by words with which they co-occur. Using a corpus of artificial language, we verify that redundancy in the training data yields non-atomic internal states, and propose a novel method for inducing atomic internal states. We show that 1) our method successfully induces atomic internal organization in controlled experiments, and 2) under more realistic conditions in which the training consists of child-directed language, application of our method improves the performance of lexical representations on a downstream semantic categorization task.

1 Introduction

In order to reproduce the full range of human language behavior, a language system needs to be able to represent sequences of words, while simultaneously being capable of forming semantically organized representations of individual words. To date, researchers have had difficulty specifying a model or system that is good at both tasks. On the one hand, static word embedding models are better at representing individual word meanings (Mikolov et al., 2013), but do not naturally lend themselves to multi-word sequences. On the other hand, language models based on recurrent neural networks (Elman, 1990) or Transformers (Vaswani et al., 2017) are well-suited for dealing with word sequences, but usually perform worse than static word embedding models on word-level tasks (Huebner and Willits, 2018) and are challenging to extract word embeddings from (Vulić et al., 2020).

In this paper, we diagnose why recurrent neural networks (RNNs) learn sub-optimal representations of individual words. In short, we argue that the language modeling objective encodes information relevant to individual words by spreading it across representations of neighboring words when they provide partially redundant information, a phenomenon related to entanglement (Bengio et al., 2013). Although this supports prediction of words in context, this impedes the formation of stand-alone lexical representations that are useful for tasks requiring de-contextualized information about individual words. In this work, we propose a simple strategy for protecting against this kind of context-sensitivity, enabling the learning of representations that, while still good for sequence prediction, are also better at a lexical semantic tasks.

First, we demonstrate the extent of entanglement in RNNs using a set of artificial corpora consisting of words from sets of (artificially constructed) semantic categories. We show that the extent of entanglement between lexical representations learned by RNNs is caused by the availability of partially correlated contextual cues. Next, we demonstrate that pre-training an RNN on a sample of the same artificial grammar *without* correlated cues protects it against entanglement when trained subsequently on input *with* correlated cues. Contrary to expectations based on catastrophic forgetting (McCloskey and Cohen, 1989), biasing the network to start with atomic organization helped the RNN retain that organization when faced with language with redundant information. Lastly, we examined whether our pre-training strategy could be applied to a corpus of natural language. We conclude by discussing implications of this work, both for language models and for theories of human language learning and representation.

1.1 Entangled Representations

Distributed representations are notorious for their susceptibility to entanglement (Bengio et al., 2013). This is especially true in the language domain, stemming from the complexity of linguistic sequences and the result of multiple, co-incident constraints (Chater and Manning, 2006; Jurafsky et al., 2001; Seidenberg and MacDonald, 1999). In order to correctly predict word sequences, a language model needs to simultaneously represent and integrate all of these factors. But the fact that most of these constraints interact, means that the distributed representations learned by prediction models will often integrate this information.

Consider an RNN trying to learn that *dog* is an animal (or, at least, that it clusters with other words that are also animals) by looking at sentences like (a) and (b):

(a) *The black dog jumps.*

(b) *The dog in the house jumps.*

A rational learner could infer that *dog* is more likely to be a an animal on the basis of the observation that 1) *dog* frequently co-occurs with *jumps*, and 2) labels for other animals also frequently co-occur with animate verb forms like *jumps*.

In contrast, the RNN does not have access to the word *dog* in a way that would automatically lead to these inferences. RNNs do not observe *dog* as a standalone item in the input. Rather, concurrently with *dog*, they also model the context of co-incident expressions like *black* and *in the house*, represented as information feeding back from recurrent connections. An RNN that has never before seen the words *black* and *dog* will not encode each word as separately predicting *jumps*. Instead, the RNN would encode a second-order (i.e. conditional) relationship between the sub-sequence *black dog* and *jumps*. The words *black* and *dog* become entangled in the model's representation, and work interactively to predict the upcoming word *jumps*. As a result, a single relationship (that involves all three words) is encoded, rather than two independent pairwise relationships.

The likelihood that a word pair like *black dog* will be represented holistically is proportional to the amount of redundant information each word provides about upcoming words. To the extent that *black* precedes *dog* whenever *dog* precedes *jumps*, *black* and *dog* can be said to provide redundant information about *jumps*.

1.2 Preventing Entanglement by Incentivizing Atomic Organization

In this work we will use the term "atomic" to refer to the internal organization of an RNN that has acquired lexical representations with as little entanglement as possible. The following demonstration is meant to provide an intuitive understanding of this term. Figure 1 shows how hidden state activations of two hypothetical RNNs change while processing the sentence *The black dog in the house jumps*. Each line illustrates the hidden state trajectory along a single hidden layer component (y-axis) over three consecutive time steps (x-axis).

Although both networks have learned to correctly predict the next word *jumps*, as evidenced by reaching the target state marked in blue, they do so by traversing very different internal states. The network illustrated by the dashed line has not arrived at an atomic organization of the language it was trained on, because each state contributes partial information about *jumps*. That is, each word's lexical representation tends to activate similar units (collapsed into the y-axis dimension) that are used by the network to predict the word *jumps*. Because this RNN has learned that the word *black* partially predicts *jumps*, the lexical representation of *dog* need only partially contribute information about the occurrence of *jumps*. A consequence of this is that the representation of *dog* only weakly encodes the category-relevant relationship between *dog* and *jumps*. This is illustrated in the weak incline at the time step at which the RNN is processing *dog*.

The second network has arrived at the same prediction, but using a more atomic organization of its internal states. Specifically, the representation of the word *dog* alone suffices to push the network's hidden layer state towards the target location. Importantly, the word *black* contributes no information to the prediction, which results in the category-relevant relationship being encoded in its entirety at the representation for *dog*.

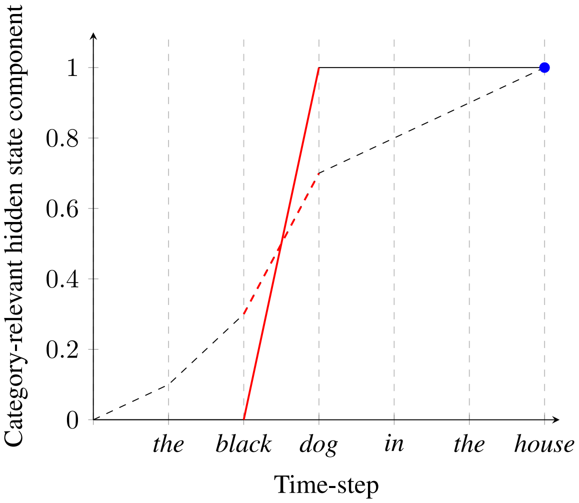

Figure 1: Hidden state trajectories of a hypothetical RNN with atomic internal states (solid line), and with internal states that leak information across time steps (dashed line). The y-axis indicates a composite dimension of all the hidden layer units that encode the semantic relation between *dog* and *jumps*. The red portion of the figure indicates the activation of the lexical representation of *dog* on this composite dimension. The blue dot marks the location in hidden state space that corresponds to the prediction *jumps* at the next time step.

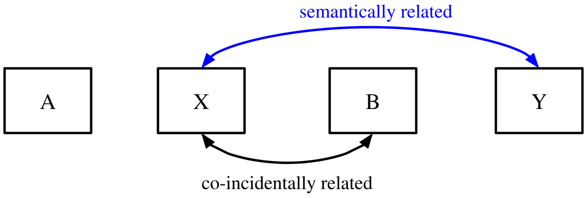

Figure 2: The sequential structure of a sequence in the artificial corpus. The relationship between X and Y is the "true" (category-relevant) relationship that the RNN should use for predicting the upcoming $Y-word$. The co-incidental relationship between X and B is governed by β. When β is large, X and B are more likely predict each other, and as a consequence, B will encode redundant information about the semantic category of the upcoming Y-word.

2 Methods

2.1 Artificial Grammar

To quantify entanglement in RNNs, we trained them on samples from an artificial grammar which - like natural language but in a more controlled manner - had sequential dependencies that were correlated with a set of externally defined semantic categories. Each corpus consisted of 100,000 sequences, each of which consisted of exactly four words (see Fig. 2). The vocabulary was split into four equally sized and disjoint sets, which we refer to as syntactic categories. Each sequence consisted of one word from each syntactic category, occurring exactly in the order "$A\ X\ B\ Y$". Each category was composed of 32 items. We refer to their members as A-words, X-words, B-words, and Y-words, respectively. See Appendix D for examples relating our grammar to English.

Our test of the RNN's ability to learn semantic categories was tested using the X-words. Each X-word belonged to one of four semantic categories, defined by the X-word's relationship to the Y-words, which also belonged to one of four semantic categories. The corpus was designed such that each Y-words had an equal probability of occurring with each of the X-words from the matching semantic category. A-words were sampled randomly and thus did not serve as a reliable cue to semantic category.

The critical manipulation concerned the distribution of the B-words. The B-words varied with regard to their redundancy with the semantic category. In the control condition ($\beta = 0.0$), a corpus was created where the B-words were not redundant with the semantic category, co-occurring with all words from all semantic categories equally. In the fully-redundant condition ($\beta = 1.0$), the B-word was, like the X-word, a perfect cue to the semantic category of the upcoming Y-word. In the partially-redundant condition ($\beta = 0.5$), each B-word behaved identically to the control condition half the time, and identically to the fully-redundant condition the other half of the time.

2.2 RNN Training

We used the RNN in a standard language modeling procedure. At each time step, corresponding to one word in one of our artificial corpora, the RNN was trained to predict the next word in the corpus. The weights were updated after each fourth word (corresponding to the end of each sequence) using stochastic gradient descent with a constant, empirically determined, learning rate. By ensuring that there were always exactly four items in the RNN's memory before updating the weights, our training methodology was equivalent to backpropagation-through-time with gradient truncation applied every four time steps. Implementation details, and hyper-parameters are reported in Appendix A.

2.3 Quantifying Atomicity

To quantify the atomicity of the RNN's internal representations, we evaluated how well the representations performed on a downstream semantic categorization task. In this task, performance was based on how well the organization of X-word representations matched the externally-defined semantic category structure. Given that the task can be solved perfectly by simply tracking the semantic relationship between X and Y-words, performance should be highest when internal organization is most atomic. That is, a network that pays attention only to the relationship between X and Y-words will acquire representations of X-words with all the necessary information to achieve a perfect score. However, if any of the information necessary for categorization is encoded by the representations of the intervening B-words, then the performance of B-words in the semantic categorization should also yield non-zero performance - and potentially reduce that of X-words. The latter scenario is undesirable, as information that could otherwise be represented non-interactively, is nonetheless represented in an entangled manner. Thus, atomic internal organization (atomicity) was operationalized as 1) good categorization of X-words, and 2) bad categorization of B-words. Intuitively, B-words should not contribute to semantic categorization, as they do not provide additional diagnostic information beyond which is already available.

Our measure of categorization performance is based on the highest possible accuracy of correctly deciding whether the representations of two words learned by the RNN[1] belong to the same category. A detailed description of how this was measured is available in Appendix B. Briefly, a balanced accuracy of 0.5 indicates that learned internal states do not reproduce the target category structure better than chance; the maximum 1.0 indicates that learned clusters perfectly reproduce the target category structure.

3 Results

3.1 Experiment 1: Redundancy impedes Atomic Organization

We examined the internal organization an RNN acquires over the course of learning from samples of a simple artificial grammar with known category structure under different input conditions. Specifically, we quantified the extent to which membership in artificially created semantic categories was represented atomically, or via entanglement between multiple representation.

The results are shown in 3. The left panel shows that perfect categorization of X-words was achieved when B-words were not redundant with X-words (red line). Perfect categorization means that X-words had internal hidden states organized into clusters that perfectly corresponded to the target semantic categories. The right panel illustrates categorization performance of B-words that were learned alongside X-words.

As expected, in the no-redundancy condition (red line) where B-words were equally likely to co-occur with all semantic categories, The X-words were perfectly categorized, and the categorization of the B-words was at chance.

In the partial-redundancy condition (blue line), the balanced accuracy of X-words slowly declined after reaching perfect performance and the balanced accuracy of B-words rose. Because the B-words were partially predictive of the Y-words (and thus the semantic category), they had representations that were entangled with the X-word representations. As the model learned about this noisy B-Y relationship, this disrupted the model's representation of the (perfectly predictive) X-Y relationship. This pattern demonstrates a failure to learn atomic internal organisation.

In the full-redundancy condition (black line), the situation was worse. The organization of X-word and B-word representations both failed to perfectly capture the target category structure; the balanced accuracy reaches a plateau near 0.8, and near 0.96, respectively. These non-intuitive results reveal a property of RNNs that is rarely discussed. Because X-words and B-words are *both* perfectly predictive of the upcoming Y-word, a naive observer would predict that their representations should both yield perfect semantic categorization performance. However, because both X and B-words always co-occur, the RNN learns to treat them as a single unit. Consequently, the network spreads the information necessary to predict the upcoming Y-word across both X and B-word representations. All information necessary to predict Y-words (necessary for categorizing X-words) doesn't get stored in the representation of X-words, because some of it can be offloaded to the representation for B-words.

[1] Lexical representations are the input-to-hidden weights corresponding to a particular item in the vocabulary.

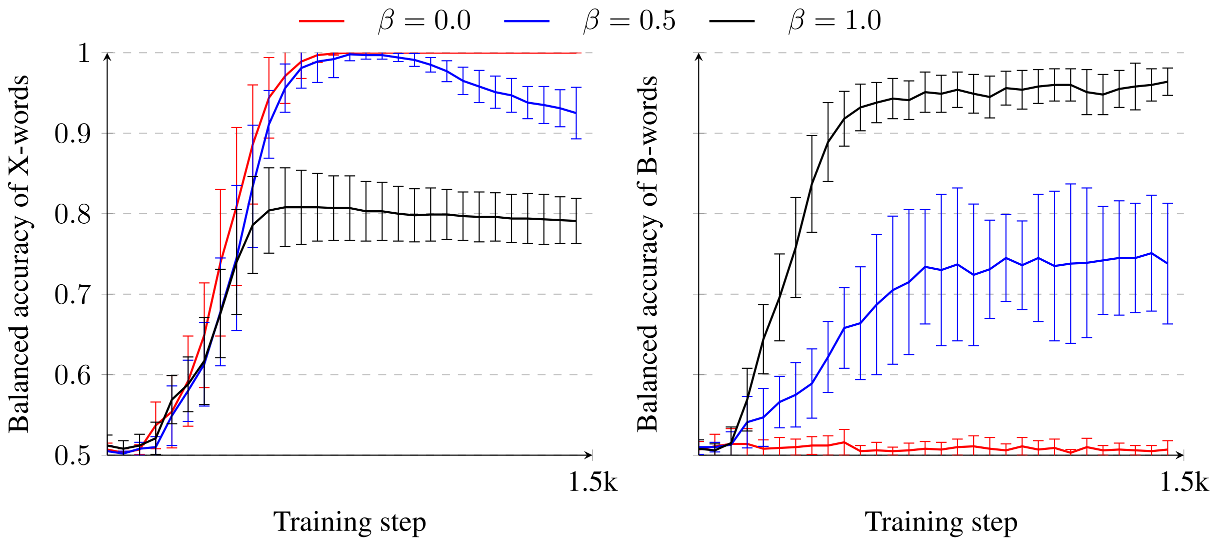

Figure 3: Internal organization of RNNs trained on corpora varying in the amount of redundant information about the target category structure provided by B-words. Each line represents the average performance across 10 RNN simulations, and error bars indicate standard deviations. The balanced accuracy measures the degree to which the internal organization of X-words (left panel) or B-words (right panel) correspond to the externally-defined semantic category structure.

The result is that, despite seemingly "perfect" distributional statistics, perfect categorization is not achieved[2].

This experiment has important implications for the use of sequence prediction language models like RNNs as models of semantic knowledge. Sequences of natural language are rife with redundant, partially correlated cues to semantic categories. The fact that these redundancies lead to non-atomic entangled representations that impair the ability of the model to learn semantic categories could be seen as a serious limitation of these models for learning static word embeddings. In Experiment 2, we test a proposal for helping mitigate this effect.

3.2 Experiment 2: Inducing Atomic Organization

In the following experiment, we evaluated the effectiveness of a novel pre-training strategy for inducing a bias towards more atomic organization of lexical representations. All RNNs were trained on the full-redundancy corpus used in the previous experiment, which we showed resulted in non-atomic internal organization. In contrast to the previous experiment, we also pre-trained the RNNs before exposing them to full-redundancy input.[3]. The experimental manipulation was whether the networks were pre-trained with input from either the no-redundancy, partial-redundancy, or full-redundancy condition. The critical question is whether the models that were pre-trained on input with less redundancy demonstrated an inductive bias towards maintaining a more atomic organization, even after exposure to fully-redundant input. As before, better atomicity should yield a distinct pattern of performance on the downstream semantic categorization task - such that the information relevant to semantic categorization is encoded independently in X-words, rather than in an entangled fashion.

We conceptualized the pre-training phase as an opportunity for the network to acquire a stable processing dynamic in which the contribution of B-words to the internal organization of X-words is ignored. Compared to a randomly initialized network that we know is heavily influenced by redundancy (as observed in 3.1), the parameters of a pre-trained network may thus be less sensitive to the redundant information provided by B-words. Such a network has already minimized language modeling loss, leaving little room to change the established processing dynamic of the network. More specifically, because the relationship between X and Y-

[2] We replicated these results using an LSTM as the algorithm underlying our language model, and observed similar results in all conditions, evidence that our results are not due to the idiosyncrasies of the vanilla RNN architecture, but a property of RNNs in general.

[3] The number of pre-training steps is identical to the number of regular training steps (1,500).

words is identical across both phases of training (pre-training and regular training), the incentive for a pre-trained network to alter the way it processes that relationship should be reduced, since that information has already been encoded.

The results are shown in Figure 4. Recall that all models had the same regular training phase (fully redundant, $\beta = 1.0$), but differed in their pre-training ($\beta = 0.0$, $\beta = 0.5$, $\beta = 1.0$). Thus, the left half of each panel in Figure 4, covering the first 1.5k training steps, resembles the results reported in section 3.1. The second half of each panel reflects the portion of training during which X-words and B-words were fully redundant with Y ($\beta = 1.0$), and which we used to examine the effects of pre-training.

For the models in the no-redundancy pre-training condition (red line), we observed that categorization accuracy for X-words did not remain at 100% after the pre-training phase. While the internal organization was initially extremely atomic, exposure to fully redundant input resulted in a modest re-reorganization of the lexical representation of X-words, such that the best clustering of X-words no longer perfectly corresponded to the semantic category structure. However, the extent of this re-organization *was* mitigated by pre-training. Categorization of X-words that had partially redundant input (blue line, left panel) finished training with higher accuracy than the models with fully redundant training (0.84 vs. 0.78), and models pre-trained in the no-redundancy condition (red line, left panel) finished training with markedly higher accuracy than the models in the full-redundancy pre-training condition (0.88 vs. 0.78). Even more striking is the effect of pre-training on the representations of B-words. Pre-training in the no-redundancy condition (red line, right panel) completely prevented the models from learning that the B-words were perfectly correlated with Y-words in the second half of training. Put differently, the RNN "learned to ignore" the co-incidental relationship between X, B and Y-words, even though B-words were equally predictive of Y-words and X-words. These results demonstrate the durable impact of pre-training with low-redundancy input, and its utility for inducing atomic internal organization when exposed to input that strongly promotes entangled lexical representations.

3.3 Experiment 3: Semantic Categories in Natural Language

The purpose of the final two experiments was to extend our findings to a corpus of natural language. Specifically, we wanted to answer two questions: First, how long does atomic internal organization last in a real-world language modeling scenario where the training data is larger and consists of more complex dependencies [4]. While the previous experiment has shown that atomic internal organization is temporarily stable, is this also true for networks trained under more realistic conditions using naturalistic rather than artificial input? Our second question concerns the utility of atomic internal organization. Can induction of atomic internal organization actually help language models learn better (less entangled) lexical representations in a real-world setting?

3.4 Experimental Setup

Our experimental setup was similar to that described in section 3.2, except that we used the AO-CHILDES corpus (Huebner and Willits, 2021) in place of the previously used non-zero redundancy artificial language corpora. AO-CHILDES contains approximately 5M words of transcribed child-directed speech, recorded as part of a large collection of language development studies during structured in-lab activities and in-home recordings. The corpus was built from raw transcripts available in the American-English section of the CHILDES database (MacWhinney, 2000) and retrieved from *childes-db* (Sanchez et al., 2019). Details are available in Appendix C.

As input to the semantic categorization task, we used the 720 probe words described by Huebner and Willits (2018) in place of the X-words used previously. These probe words frequently occur in AO-CHILDES, are all nouns[5], and each belong to one of 30 semantic categories (e.g. MAMMAL, FRUIT). We used the input-to-hidden weight vector for each of the 720 probe words as their learned representations, and computed the average balanced

[4] There are many ways in which dependencies in natural language can be more complex than in our artificial language. For example, we only modeled redundancy and partial redundancy, but lexical dependencies can also be synergistic. This occurs when a combination of two words provides more information about upcoming words than when each is considered independently of the other. Additionally, dependencies may span longer distances in both linear or hierarchical distance.

[5] We excluded probes in the NUMBER category which are more often used as adjectives rather than nouns.

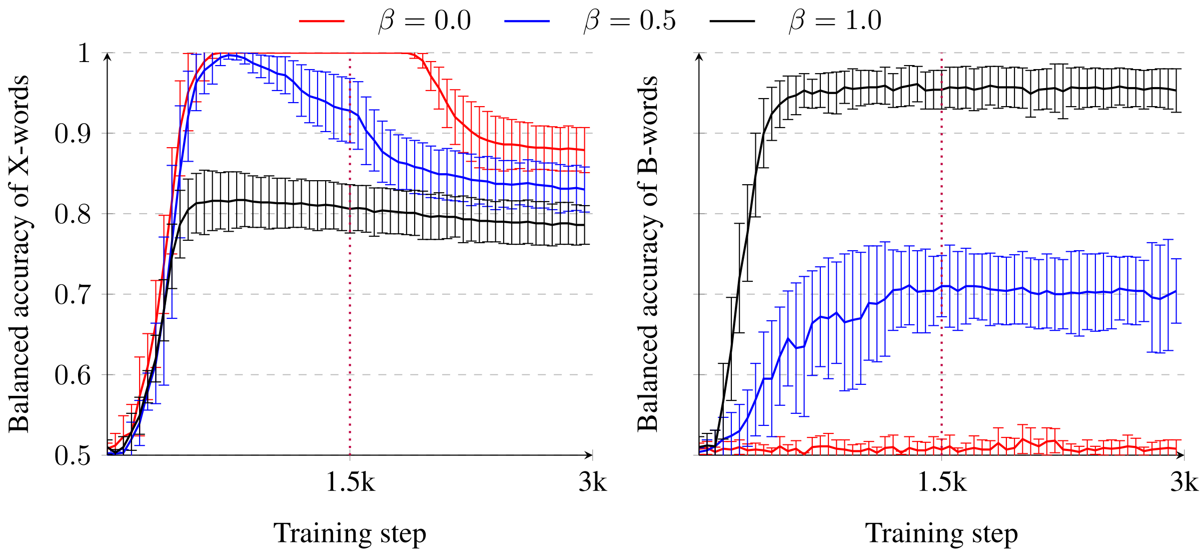

Figure 4: Internal organization of RNNs trained on corpora varying in the amount of redundant information about the target category structure provided by B-words during the pre-training phase only (first half of training). Each line represents the average performance across 10 RNN simulations, error bars indicate standard deviations, and the dotted vertical line marks the end of pre-training. The balanced accuracy measures the degree to which the internal organization of X-words (left panel) or B-words (right panel) correspond to the externally-defined semantic category structure.

accuracy as described in section 2.3. The balanced accuracy indicates the extent to which internal states corresponding to probe words could be clustered to mimic the externally-defined semantic category structure.

Like any sample of natural language with rich semantics, AO-CHILDES contains a high degree of semantic redundancy between noun contexts[6] (similar to the relationship between X-words, B-words, and Y-words in our non-zero redundancy corpora). Non-zero redundancy is a straightforward consequence of English grammar (and that of many other languages) because it allows the same noun to enter into more than one dependency. There are many different kinds of such dependencies, each of which can provide semantic information about the same noun: The adjective (e.g *The mean cat* ...), the verb (e.g *The cat jumped* ...), and any modifying relative clause (e.g *The cat that bit the dog* ...).

3.5 Experiment 3a: Is Atomicity Stable?

We trained 10 RNNs in each of two conditions. In the baseline condition, the networks were pre-trained with 37K steps on 7-token windows sampled[7] from AO-CHILDES. Conversely, in the atomicity-bias condition, networks were pre-trained on sequences of the form X_iY_i where X-words are probe words and Y-words are nonsense items whose distributions are perfectly diagnostic of a probe's target semantic category. Because sequences consisted only of probes and a single category-diagnostic neighbor without the possibility of redundancy with other items, the acquired lexical representations are atomic. The reason we opted for 37K steps is that this was the minimum number of steps needed for the models in the atomicity-bias condition to achieve perfect semantic categorization. In the standard-training phase, networks in the baseline condition simply continued training on AO-CHILDES sequences. In the atomicity-bias condition, the learned lexical representations of probe words were transferred to otherwise randomly initialized RNNs, which were then trained on the same sequences of AO-CHILDES as models in the baseline condition.

The results are shown in the left panel of Figure 5. Networks in the atomicity-bias condition (red line) are able to preserve their initially strong semantic category knowledge over the course of millions of training steps, with only slow degrada-

[6] We verified this offline using the interaction information, an information-theoretic tool which can be used to quantify the extent to which three or more variables share information.

[7] We sampled without replacement. Once all windows are exhausted, we begin a new epoch, and repeat the process.

tion that appears linear in the number of training steps. Furthermore, at the end of training, performance in the atomicity-bias condition is still well above that of networks trained on AO-CHILDES only. This finding demonstrates the long-lasting stability of atomic internal organisation in the face of complex lexical interactions present in naturalistic child-directed input.

3.6 Experiment 3b: Is Atomicity Useful?

Based on our results from section 3.1, we predicted that atomicity induction could be used to improve performance on a downstream lexical task. We tested this idea by comparing the lexical semantic knowledge acquired by RNNs trained on a AO-CHILDES with and without atomic internal organisation.

There is a caveat. The way in which we induce atomic internal organisation in the previous experiment requires the networks be provided with perfect knowledge of the target task. For a fair comparison, we require a method that can induce atomicity without providing more information about the target task than is warranted. Therefore, instead of using a pre-training strategy to induce atomicity, we induced atomic integral organisation by pre-training a network exclusively on AO-CHILDES, and then randomly re-initialized all parameters except for learned lexical representations of probe words. This strategy was designed to induce atomicity by removing any learned interactions between probe words and the contexts in which they occurred. That is, after re-initialization, a network cannot use its previous knowledge about the contexts in which probes occur to predict semantic features of probes - its only knowledge is that encoded in the learned lexical representations of probe words.

The results are shown in the right panel of Figure 5. The semantic categorisation accuracy of RNNs in the atomicity-bias condition (red line) initially drops below the baseline (blue line) and then increases rapidly and finally surpasses the baseline (average balanced accuracy is 65.71 ± 0.17 [8] in the atomicity-bias condition and 66.29 ± 0.17 in the baseline condition). The initial dip occurs because the randomly re-initialized networks must re-learn all parameters not responsible for representing probe words, including all other lexical representations.

3.7 Next-Word Prediction

We considered the possibility that RNNs in the atomicity-bias condition might also perform better with regards to next-word prediction. To test this, we computed the perplexity of fully trained networks on a 100K word subset of AO-CHILDES not seen during training. We did not find a significant difference between the two conditions in Experiment 3a or 3b. The average perplexity in experiment 3a is 69.58 ± 0.84 vs. 69.39 ± 0.58, and 69.39 ± 0.58 vs. 70.18 ± 2.71, in the baseline and atomicity-bias condition, respectively.

4 Discussion

Language users represent both the meanings of sentences and individual words (Jacobs et al., 2016; Ambridge et al., 2015). In this work, we examined whether a recurrent neural network (RNN) language model can do the same. Because next-word prediction requires extreme sensitivity to lexical context, we hypothesized the RNN acquires lexical representations that are not suitable for downstream tasks in which lexical representations must stand on their own - without being supported by the contexts in which they occurred during training.

In section 3.1, we compared the internal organization of RNN language models trained on samples from an artificial grammar, and showed that non-atomic lexical representations are acquired by models trained on input in which neighboring items provide redundant information about an upcoming word. Instead, the RNN learned to offload information relevant to one set of words to the representations of neighboring words - resulting in entanglement between the two words' representations. This entanglement makes the RNN good at next-word prediction, but lowers the effectiveness of its lexical representations for tasks that operate on single words. Next, we argued that an inductive constraint might protect lexical representations from irrelevant (i.e. redundant) information provided by neighboring words. We reasoned that by establishing processing dynamics during pre-training on input without redundant information, the RNN could no longer minimize the language modeling objective by predicting redundant items, and would therefore "learn to ignore" the redundant information provided by neighboring items.

In section 3.2, we observed that our pre-training strategy could induce a long-lasting inductive bias for acquiring lexical representations that are robust

[8] mean ± margin of error, with $\alpha = 0.05$

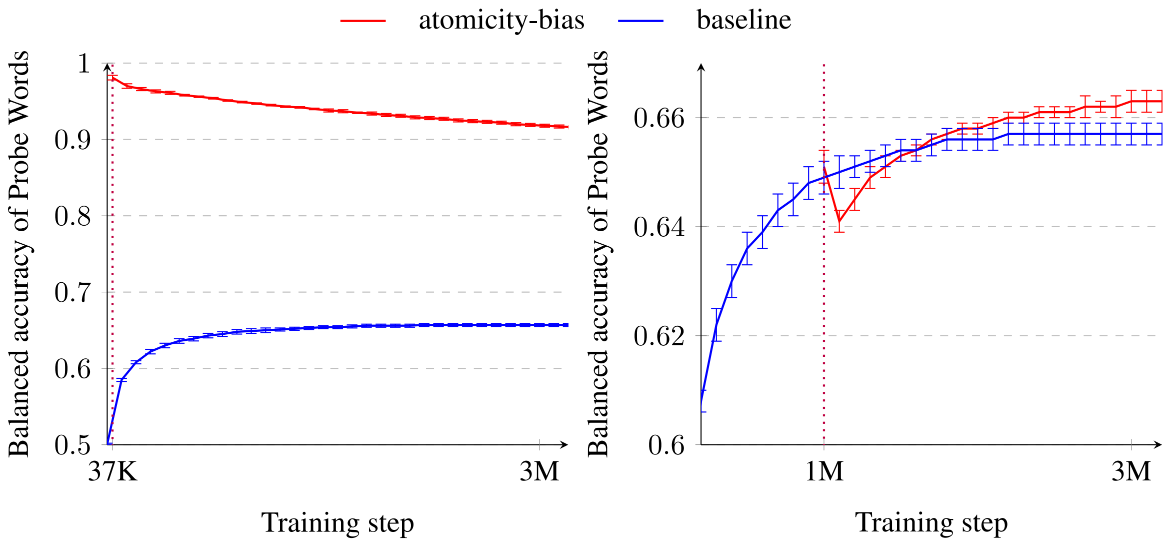

Figure 5: Semantic categorization accuracy, in units of balanced accuracy (y-axis), across training time (x-axis). Each line represents the average performance across 10 RNN simulations, and error bars indicate 95% confidence-intervals. **Left Panel:** Results of Experiment 3a. Two groups of 10 RNNs were trained on approximately 3M steps of child-directed language randomly sampled from AO-CHILDES, but with different initial knowledge and experience. In the atomicity-bias condition, the RNNs were first trained with 37K steps on simple artificial 2-item sequences (of the form $X_i Y_i$) to induce perfect knowledge of the target semantic category structure and with perfect atomicity. In the baseline condition, the RNNs were first trained with 37K steps on sequences of child-directed input - which neither provides perfect knowledge of the target task nor strong atomicity. **Right Panel:** Results of Experiment 3b. RNNs were first trained for 1M steps (13 epochs) on AO-CHILDES, and then separated into one of two groups (time of separation is indicated by ······). In the atomicity-bias condition (——), all the parameters of an RNN were randomly re-initialized except for lexical representations of probe words. In the baseline condition (——), training on AO-CHILDES continued uninterrupted for another 2M steps (27 epochs).

against redundant information. Strikingly, we observed that once an atomic internal organization is acquired, it is relatively stable. In section 3.3, we extended this analysis to RNNs trained on a corpus of natural language. Not only were we able to show induction of atomic internal organisation is feasible, and long lasting, but that it can yield a noticeable improvement on a downstream lexical task.

To conclude, our demonstrations reveal a fundamental trade-off between sequence-level and word-level learning dynamics; while greater utilisation of contextual information is useful for modeling sequences, individual lexical items become increasingly reliant on information provided elsewhere in the system and are thus less useful as input to downstream lexical tasks. Does this mean we are better off using different models for sequence versus lexical tasks? This is unsatisfying from the perspective of a unified account of language acquisition. An alternative is to extend models operating on sequences with inductive biases that encourage more atomic organization. Our demonstration in section 3.3 showed that induction of an atomicity-bias is a promising first step in that direction.

Finally, it is possible that the lack of atomicity is a limitation unique to recurrent or other networks that force sequential order during processing of word sequences. Work by Vulić et al. (2020) suggests that word embeddings extracted from non-recurrent Transformer-based language models, which represent word order in the parameters of the model rather than being strictly enforced as a result of sequential processing, outperform static word embeddings on a variety of lexical tasks. Therefore, an important next step is to quantify and compare atomicity of language models that vary in how strictly word order is enforced during processing (e.g. recurrence vs. self-attention).

Acknowledgments

We wish to thank the anonymous reviewers for insightful commentary and helpful suggestions, and in particular for providing the inspiration that led to Experiment 3b.

References

Gerry T M Altmann and Yuki Kamide. 1999. Incremental interpretation at verbs: restricting the domain of subsequent reference.

Ben Ambridge, Evan Kidd, Caroline F Rowland, and Anna L Theakston. 2015. The ubiquity of frequency effects in first language acquisition. *J. Child Lang.*, 42(2):239–273.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

Nick Chater and Christopher D Manning. 2006. Probabilistic models of language processing and acquisition. *Trends in cognitive sciences*, 10(7):335–344.

Morten H Christiansen and Nick Chater. 1999. Toward a connectionist model of recursion in human linguistic performance. *Cogn. Sci.*, 23(2):157–205.

Gary S Dell and Franklin Chang. 2014. The p-chain: Relating sentence production and its disorders to comprehension and acquisition. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1634):20120394.

Jeffrey L Elman. 1990. Finding structure in time. *Cogn. Sci.*, 14(2):179–211.

Jeffrey L Elman. 1991. Distributed representations, simple recurrent networks, and grammatical structure. *Mach. Learn.*, 7(2):195–225.

Kara D Federmeier. 2007. Thinking ahead: the role and roots of prediction in language comprehension. *Psychophysiology*, 44(4):491–505.

Betty Hart and Todd R Risley. 1995. *Meaningful differences in the everyday experience of young American children.* Paul H Brookes Publishing.

S Hochreiter and J Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Ryan J Hubbard, Joost Rommers, Cassandra L Jacobs, and Kara D Federmeier. 2019. Downstream behavioral and electrophysiological consequences of word prediction on recognition memory. *Front. Hum. Neurosci.*, 13:291.

Philip A Huebner and Jon A Willits. 2018. Structured semantic knowledge can emerge automatically from predicting word sequences in child-directed speech. *Frontiers in Psychology*, 9:133.

Philip A Huebner and Jon A Willits. 2021. Using lexical context to discover the noun category: Younger children have it easier. In *Psychology of learning and motivation*, volume 75. Elsevier.

Cassandra L Jacobs, Gary S Dell, Aaron S Benjamin, and Colin Bannard. 2016. Part and whole linguistic experience affect recognition memory for multiword sequences. *Journal of Memory and Language*, 87:38–58.

Daniel Jurafsky, Alan Bell, Michelle Gregory, and William D Raymond. 2001. Probabilistic relations between words: Evidence from reduction in lexical production. *Typological studies in language*, 45:229–254.

Tal Linzen and T Florian Jaeger. 2016. Uncertainty and expectation in sentence processing: Evidence from subcategorization distributions. *Cogn. Sci.*, 40(6):1382–1411.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for analyzing talk. transcription format and programs*, volume 1. Psychology Press.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

D L Rohde and D C Plaut. 1999. Language acquisition in the absence of explicit negative evidence: how important is starting small? *Cognition*, 72(1):67–109.

Alessandro Sanchez, Stephan C Meylan, Mika Braginsky, Kyle E MacDonald, Daniel Yurovsky, and Michael C Frank. 2019. childes-db: A flexible and reproducible interface to the child language data exchange system. *Behavior research methods*, 51(4):1928–1941.

Mark S Seidenberg and Maryellen C MacDonald. 1999. A probabilistic constraints approach to language acquisition and processing. *Cognitive science*, 23(4):569–588.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Edinburgh neural machine translation systems for wmt 16. *arXiv preprint arXiv:1606.02891.*

W Tabor, C Juliano, and M K Tanenhaus. 1997. Parsing in a dynamical system: An attractor-based account of the interaction of lexical and structural constraints in sentence processing. *Lang. Cogn. Process.*

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS*.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 7222–7240, Online. Association for Computational Linguistics.

A Implementation Details and Reproducibility

Code for training RNNs on our artificial corpora is available at https://github.com/phueb/Entropic.

A.1 The RNN

Learning in the RNN is based on minimizing the error between the predicted and actual next item in the corpus, given the model's encoding of previous items. Its memory of previously seen items is encoded into a fixed-size low dimensional vector typically referred to as the hidden layer. The output of the hidden layer h_t of an RNN can be defined as

$$h_t = f(h_{t-1}, x_t), \tag{1}$$

where h_{t-1}, is the value of the hidden layer at time $t-1$, x_t is the input feature vector and $f(.)$ is a nonlinear function. The weights of the network that connect the input to The Hidden layer, the hidden to Hidden layer, and the hidden to the Output layer, are randomly initialized and updated using Stochastic gradient descent or some variant thereof.

In theory, the RNN is capable of maintaining information about items located an arbitrary number of steps into the past, but due to the instability of the gradient across time steps, this is extremely improbable in practice. To overcome the difficulty of learning longer-distance dependencies, numerous extensions of the RNN have been proposed. For example, the Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) uses gating units to control the flow of information into and out of the hidden state. Because the hidden state in the LSTM is never squashed by a non-linearity, this allows for more efficient gradient propagation across time steps. Due to the widespread use of LSTMs, we repeat each of our experiments using an LSTM in place of the standard RNN.

In language acquisition research, the RNN has been primarily used to examine what kind of linguistic dependencies can be captured by the network and under what circumstances (e.g. size of training data, distance between dependent items), and whether the resultant knowledge generalizes to unseen examples. Early studies using artificial grammars or pseudo-English showed that the network learns to predict the next word probability distribution reasonably well, even in situations requiring it to learn long-distance dependencies across potentially uninformative sub-sequences, such as embedded clauses (Elman, 1990, 1991). In addition to evaluating the network's success at the next word prediction task, the network's hidden layer activations have been studied, and showed that activation patterns over hidden units capture the syntactic and semantic structure built into a training corpus (Elman, 1990, 1991; Christiansen and Chater, 1999; Rohde and Plaut, 1999; Tabor et al., 1997; Huebner and Willits, 2018). For these reasons and others, the RNN has become the model of choice for researchers who consider predictive processing as fundamental to both language comprehension and production (Altmann and Kamide, 1999; Federmeier, 2007; Dell and Chang, 2014; Hubbard et al., 2019; Linzen and Jaeger, 2016).

A.2 Hyper-parameters

Instead of tuning our hyperparameters on the language modeling objective, we chose hyperparameters that resulted in best performance on the downstream categorization task directly. Because we are interested primarily in performance on this task, our tuning strategy eliminated any bias that would have resulted from optimizing the language modeling objective. Our hyperparameters for experiments 1 and 2 are shown in 1, and hyperparameters used in section 3.3 are shown in 2.

hyperparameter	vanilla RNN	LSTM
window size	7	7
hidden layers	1	1
hidden units	64	64
learning rate	0.4	1.0
optimizer	SGD	SGD
batch size	64	64
steps	3K	3K
non-linearity	*tanh*	*tanh*
initialization	uniform	uniform

Table 1: Hyper-parameters used in Experiment 1 and 2.

The RNN used in experiment 3.3 used a custom sub-word vocabulary based on Byte-Pair Encoding introduced by Sennrich et al. (2016). This eliminates the need for "UNK" symbols, and is therefore a more cognitively plausible representation of the

hyperparameter	vanilla RNN	LSTM
window size	7	7
hidden layers	1	1
hidden units	512	512
learning rate	0.01	0.01
optimizer	AdaGrad	AdaGrad
batch size	64	64
steps	3M	3M
non-linearity	*tanh*	*tanh*
initialization	uniform	uniform

Table 2: Hyper-parameters used in Experiment 3.

input[9].

B Computation of the Balanced Accuracy

The model's judgements are based on a similarity matrix S obtained by computing all pairwise similarities[10] between either A-word, X-word, or B-word representations. To obtain the model's learned Lexical representations for one set of words, we retrieved the vector that connects the input unit corresponding to a single word with the hidden layer. Each similarity in matrix S was used to make a "same vs. different" judgment within a signal detection framework, tested at multiple similarity thresholds (r = 0.0 to 1.0 with step size 0.001) to determine the threshold for maximum accuracy. If two words with indices i and j belong to the same category, and if $S_{ij} > r$, a hit is recorded, whereas if $S_{ij} < r$, a miss is recorded. On the other hand, if the two words do not belong to the same category, either a correct rejection or false alarm is recorded, depending on whether $S_{ij} < r$ or $S_{ij} > r$. At each threshold, we computed the balanced accuracy by taking the average of sensitivity and specificity. The measure of interest is the balanced accuracy at the similarity threshold which yielded the highest value. We used this process to compute a balanced accuracy score for each model, at each evaluation time point. Chance-level performance on this task would produce a balanced accuracy of 0.5. We evaluated the balanced accuracy for X-words, and B-words after every 50 training steps during training to obtain a complete picture of the learning trajectory. The atomicity of the learned internal organisation of the RNN can then be determined by inspecting the extent to which the balanced accuracy is high for X-words but low for B-words.

The balanced accuracy is appropriate because it eliminates bias due to the unbalanced distribution of correct "same" and "different" judgements - the vast majority of X-word pairs do not belong to the same category. Because the balanced accuracy is the average between the sensitivity and the specificity, it measures the average accuracy obtained from both the minority and majority classes. This quantity reduces to the traditional accuracy if a classification accuracy is identical for either classes. But, if the high value of the traditional accuracy is due to taking advantage of the distribution of the majority class, then the balanced accuracy will decrease compared to the traditional accuracy.

It is worth noting that the categorization task (computation of balanced accuracy) does not involve predicting category labels. There are no category labels, and the RNN therefore does not explicitly learn to map members of the same category to their label. In order to successfully reconstruct the target category structure, the RNN must acquire lexical representations such that their similarity is higher for same-category members than members that belong to different target categories. In this sense, the categorization judgement is entirely similarity-based. Importantly, the similarity structure of the learned internal representations is an indicator of the overall organisation of the representational landscape learned by the model.

C The AO-CHILDES corpus

The AO-CHILDES corpus was created as follows: First, we first obtained all transcripts in the CHILDES database that involve children 0 to 6 years of age from American English speaking households and excluded those for which no age information was available[11]. After removal of non-adult speech, we obtained 3,251 transcripts containing 272,250 unique word types, and 5,245,298 total word tokens. Considering that a typical American child receives approximately 6.5-11.0 million words per year (Hart and Risley, 1995), the corpus

[9] We trained a custom Byte-Pair vocabulary of size 8192 using the Python package *tokenizers*

[10] As a measure of similarity between two vectors, we used the cosine of the angle between them.

[11] Transcripts were obtained from childes-db.stanford.edu on Dec 1, 2017 and processed using code available at `https://github.com/UIUCLearningLanguageLab/AOCHILDES`

represents approximately 8–14% of lexical input of the average 6-year-old child.

The transcribed corpus was tokenized by splitting on spaces and contractions, and sentence-boundary punctuation (periods, exclamation marks, and question marks) was left in the corpus as individual tokens. This was intended to serve as a very crude way for representing the pauses and prosody that tend to accompany utterance boundaries. We did not perform any morphological parsing in order to leave intact as many naturalistic properties of the corpus as possible. In the original corpus, the transcripts were ordered by the age of the target child (AO is short for age-ordered). In this work, however, we did not take advantage of this ordering, because we trained the RNN on the full corpus with each new epoch.

D Relationship to Natural Language

The artificial grammar used in Experiments 3.1 and 3.2 may at first appearance bear little relation to natural language. However, the artificial grammar has an intuitive relation to English sentences, and which is worth sharing to readers who prefer specific examples to make this relationship explicit. Recall that our grammar produces strings of the form $A_i\ X_j\ B_k\ Y_l$. We are now in a position to imagine pseudo-English examples for such a sequence. Given that each set strictly obeys the positional rules, we can think of them roughly as part-of-speech or phrasal categories (e.g. VP), and their items as representing members thereof. For example, during corpus creation, we conceptualized X-words as English nouns, and Y-words as English verb phrases that express actions selectively associated with the entities referred to by the nouns. Items in A and B may be thought of as pre-nominal expressions, and relative clauses, respectively. Thus, we can think of a sequence in our corpus as declarative constructions like (b-d):

(a) $a_i\ x_i\ b_i\ y_i$

(b) *[The] doll [over there] [looks funny]*

(c) *[The red] doll [over there] [looks funny]*

(d) *[The red] doll [in your hand] [looks funny]*

We use the subscript i to index the i-th sequence in the corpus, and brackets to join multiple English words into a single item. Because we are interested in learning lexical representations for items in X, we require that X-words be thought of as individual words. All other items in the vocabulary can be thought of as any linguistic expression (e.g. inflectional marker, morpheme, lexeme, phrase). For example, B-words could take the form of a plural marker, like the English s. However, we only mention this for the purpose of providing the reader a tool for connecting our ideas to natural language sentences. It must be kept in mind that our hypothesis makes no commitment to the kinds of expressions the items in our corpus represent - save for X-words. This makes our findings applicable to researchers using different tokenization methods (and researchers studying sequential relationships outside of language).

E Additional Findings

E.1 A-words

Note that our grammar ($A_i\ X_j\ B_k\ Y_l$) includes A-words, which occur at the beginning of each sequence. In the same way in which we manipulated the redundancy of B-words, we also investigated the effects of manipulating the redundancy of A-words - by controlling the parameter α in the same way we did with β. The important difference between B-words and A-words is that A-words occur before, rather than after, X-words.

In Figure 6 we demonstrate the results of RNN language modeling simulations for corpora differing in the value of α. The right panel shows that perfect categorization (balanced accuracy reaches 1.0) of X-word representations is achieved when A-words are not redundant, or partially redundant (red and blue lines), but not fully redundant with X-words (black line). Perfect categorisation means that X-word representations are organized into clusters that perfectly corresponds to their target categories. The left panel illustrates categorization performance of A-word representations that were learned alongside X-word representations. At the end of training, in both the no-redundancy and partial-redundancy conditions, the balanced accuracy for X-words is at ceiling and the balanced accuracy for A-words is close to chance (balanced accuracy of 0.5). Thus, can say that the networks trained in these two conditions have acquired highly atomic lexical representations. However, in the full-redundancy condition (black line), in which A-words are fully redundant with X-words, the target category structure is only partially captured in the organization of X-word representations. This failure is accompanied by strong, but imperfect, categorization performance of A-word representations, shown in the left panel. This pattern of results exemplifies low atomicity: Neither the representations of A-words nor of X-words alone has captured the target category structure perfectly; instead, the target category structure must be encoded in the *interaction* between the two representations[12].

In Figure 7 we demonstrate the results of RNN pre-training on corpora differing in the value of α. For reference, the transition from pre-training to regular training occurs at training step 1,500, which we marked by a vertical line in the figure. Strikingly, the performance attained at the end of pre-training is nearly perfectly maintained throughout the subsequent 1,500 steps of training in the full-redundancy condition. Although training directly on the full-redundancy corpus results in non-atomic internal organisation (poor balanced accuracy for both A-words and X-words), networks pre-trained in either the no-redundancy or partial-redundancy condition were able to maintain their nearly perfect atomic organisation (high and low balanced accuracy, for X-words and A-words, respectively). In alignment with our predictions, these results demonstrate that pre-training can be effectively leveraged for inducing robustness against redundancy patterns that promote non-atomic internal organisation.

[12] We verified this claim by computing the balanced accuracy for composite representations generated by inputting A X sequences to the RNN. Categorization using these composite representations was at ceiling.

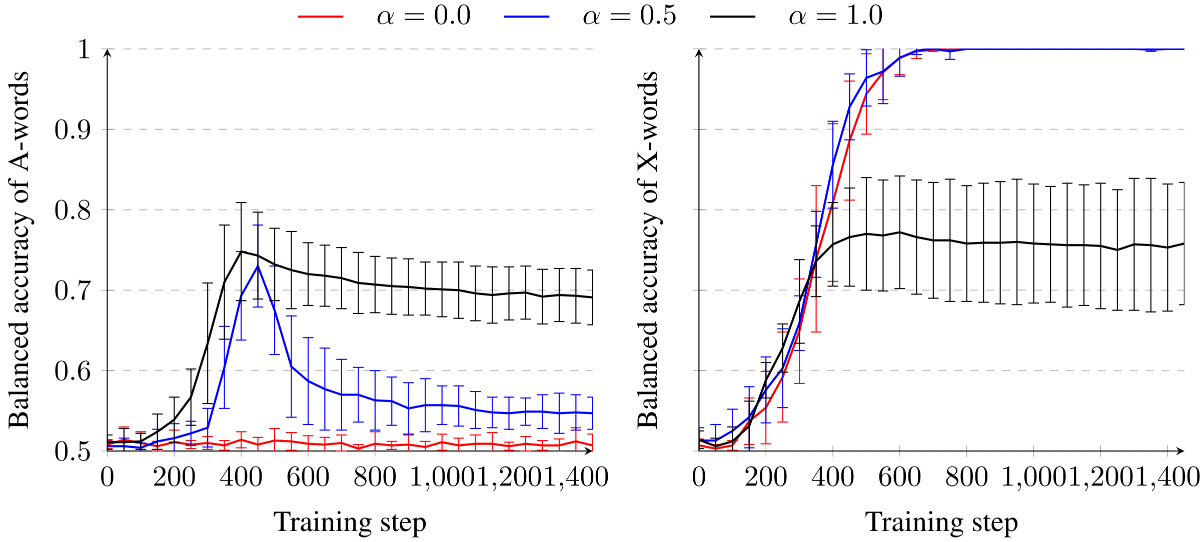

Figure 6: Internal organization of RNNs trained on corpora varying in the amount of redundant information about the target category structure provided by B-words during the pre-training phase only (first half of training). Each line represents the average performance across 10 RNN simulations, error bars indicate standard deviations, and the dotted vertical line marks the end of pre-training. The balanced accuracy measures the degree to which the internal organization of A-words (left panel) or X-words (right panel) correspond to the externally-defined semantic category structure.

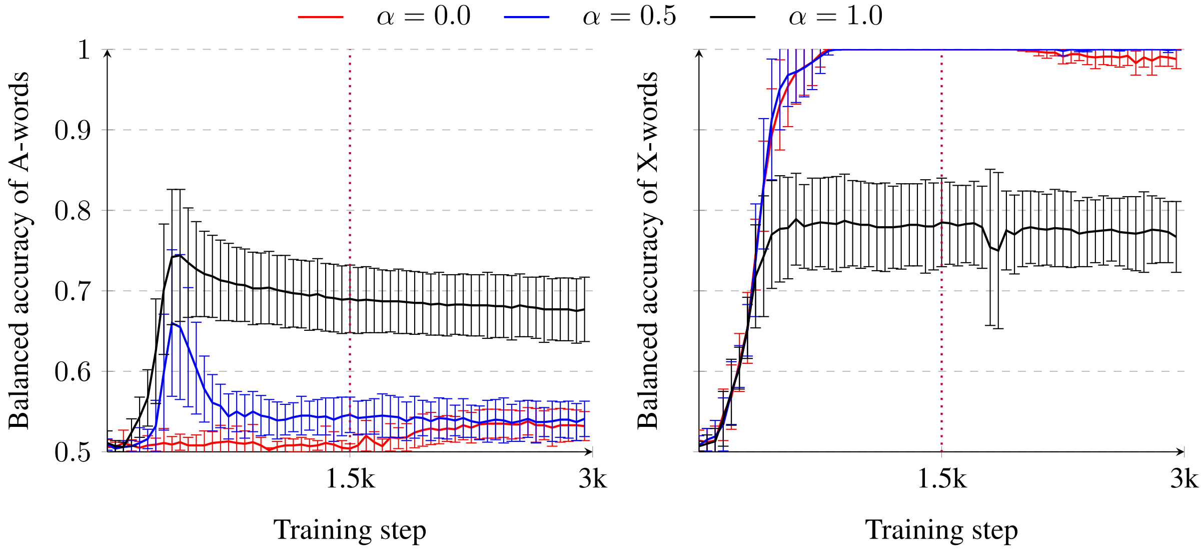

Figure 7: Internal organization of RNNs trained on corpora varying in the amount of redundant information about the target category structure provided by A-words during the pre-training phase only (first half of training). Each line represents the average performance across 10 RNN simulations, error bars indicate standard deviations, and the dotted vertical line marks the end of pre-training. The balanced accuracy measures the degree to which the internal organization of A-words (left panel) or X-words (right panel) correspond to the externally-defined semantic category structure.