Does syntax need to grow on trees? Sources of inductive bias in sequence to sequence networks

Anonymous TACL submission

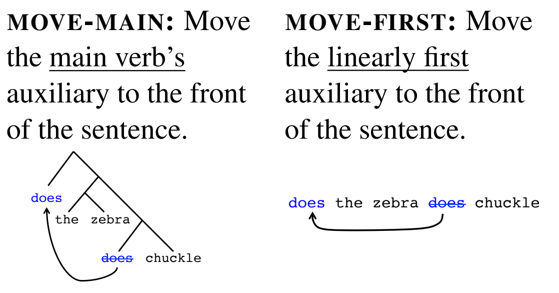
Abstract

Learners that are exposed to the same training data might generalize differently due to differing inductive biases. In neural network models, inductive biases can emerge from two sources: high-level model structure and low-level model details. We investigate which properties qualitatively affect how neural sequence-to-sequence models generalize on two syntactic tasks, English question formation and English tense reinflection. For both tasks, the training set is consistent with a generalization based on hierarchical structure and a generalization based on linear order. We find that all factors investigated can qualitatively affect how a model generalizes, including low-level factors with no clear connection to hierarchical structure. For example, we find that LSTMs and GRUs display qualitatively different inductive biases. However, the only factor with a clearly interpretable contribution to inductive bias across tasks is the high-level structure of the model (i.e., whether it is sequential or tree-based).

1 Introduction

Any finite training set is consistent with multiple generalizations. Therefore, the way that a learner generalizes to unseen examples depends not only on the training data but also on properties of the learner. Suppose a learner is told that a blue triangle is an example of a *blick*. A learner preferring shape-based generalizations would conclude that *blick* means "triangle," while a learner preferring color-based generalizations would conclude that *blick* means "blue object" (Landau et al., 1988). Factors that guide a learner to choose one generalization over another are called **inductive biases**.

What properties of a learner cause it to have a particular inductive bias? We investigate this question with respect to sequence-to-sequence neural networks (Botvinick and Plaut, 2006; Sutskever et al., 2014). The inductive biases of these models are often discussed as arising from their high-level structure; for example, the use of recurrence might create a bias for sequential representations (Battaglia et al., 2018). We investigate not only these high-level properties but also low-level details such as learning rate and hidden layer size. These details are widely known to affect a model's score on its test set, but whether they can qualitatively affect generalization behavior has rarely been studied.


MOVE-MAIN: Move the main verb's auxiliary to the front of the sentence.

MOVE-FIRST: Move the linearly first auxiliary to the front of the sentence.

Figure 1: Two potential rules for English question formation.

As a test case for studying differences in how models generalize, we use the syntactic task of **English question formation**, such as transforming (1a) into (1b):

(1) a. The zebra does chuckle.
    b. Does the zebra chuckle?

Following Chomsky's (1980) empirical claims about the nature of children's linguistic input, we constrain our training set to be consistent with two possible rules illustrated in Figure 1: MOVE-FIRST (a rule based on linear order) and MOVE-MAIN (a rule based on hierarchical syntactic structure). We then evaluate each trained model on examples where the rules make different predictions, such as

(2): given (2a), MOVE-MAIN would generate (2b) while MOVE-FIRST would generate (2c):

(2) a. Your zebras that don't dance do chuckle.
 b. Do your zebras that don't dance chuckle?
 c. Don't your zebras that dance do chuckle?

A model's behavior on such examples illuminates which rule the model is biased toward. This task is discussed further in Section 2.

In addition to the broad question of which factors can affect a model's inductive biases, we also address a second, more specific question: Which factors can impart a bias for hierarchical generalization? This question is important for models of language because hierarchical structure is central to natural language syntax. Because of this fact, such a bias has been argued to underlie language acquisition in humans (Chomsky, 1965), on the assumption that a model with a hierarchical bias would learn syntactic generalizations more readily than a model without such a bias.

To test for hierarchical bias, we combine the question formation experiments with experiments using a second task: **tense reinflection**. For both tasks, our training set is ambiguous between a hierarchical generalization and a linear generalization. If a model only chooses the hierarchical generalization for one task, this preference is likely due to some task-specific factor rather than a general hierarchical bias. On the other hand, a preference for hierarchical generalizations that is consistent across tasks would provide converging evidence that a model has a hierarchical bias.

We find that all factors we tested can qualitatively affect how a model generalizes on the question formation task. These factors include six low-level factors—recurrent unit, attention, hidden layer size, learning rate, and random initialization—as well as one high-level factor, namely whether we use a sequential model or a tree-based model. Even though the low-level factors affected the model's decision between MOVE-MAIN and MOVE-FIRST, only the high-level factor of using a tree-based model can be said to impart a hierarchical bias, since this was the only model type that chose a hierarchical generalization across both of our tasks. Our specific findings include:

- Generalization behavior is profoundly affected by the type of recurrent unit and the type of attention, and also by the *interactions* between these factors.
- LSTMs and GRUs have qualitatively different inductive biases. The difference appears at least partly due to the fact that the values in GRU hidden states are bounded within a particular interval (Weiss et al., 2018).
- For one architecture (GRUs with location-based attention), learning rate and hidden layer size were both significantly positively correlated with hierarchical preference.
- Only a model given the correct tree structure in both the encoder and decoder displayed a robust hierarchical bias across tasks.

Overall, we conclude that low-level factors can qualitatively affect a model's inductive biases, but human-like syntactic generalization may require specific types of high-level structure, at least when learning from text alone.

2 The question formation task

2.1 Background

The classic discussion of English question formation begins with two empirical claims: (i) disambiguating examples such as (2) rarely occur in a child's linguistic input, but (ii) all learners of English nevertheless acquire MOVE-MAIN rather than MOVE-FIRST. Chomsky (1965, 1980) uses these points to argue that humans must have an innate bias toward learning syntactic rules based on hierarchy rather than linear order.

There has been a long debate about these conclusions. Though some have discussed the validity of Chomsky's empirical claims (Crain and Nakayama, 1987; Ambridge et al., 2008; Pullum and Scholz, 2002; Legate and Yang, 2002), most of the debate has been about which mechanisms could explain the preference for MOVE-MAIN. These mechanisms include an assumption of substitutability (Clark and Eyraud, 2007), a bias for simplicity (Perfors et al., 2011), exploitation of statistical patterns (Lewis and Elman, 2001; Reali and Christiansen, 2005), and semantic knowledge (Fitz and Chang, 2017); see Clark and Lappin (2010) for in-depth discussion.

These past works focus on the *content* of the bias that favors MOVE-MAIN (i.e., which types of generalizations the bias supports), but we instead focus on the *source* of this bias (i.e., which factors of the learner give rise to the bias). In *Rethinking Innateness*, Elman et al. (1998) argue that innate

☐ Training set, validation set, test set ▨ Generalization set

	DECL	QUEST
No RC	the newts do see my yak by the zebra . → the newts do see my yak by the zebra .	the newts do see my yak by the zebra . → do the newts see my yak by the zebra ?
RC on object	the newts do see my yak who does fly . → the newts do see my yak who does fly .	the newts do see my yak who does fly . → do the newts see my yak who does fly ?
RC on subject	the newts who don't fly do see my yak . → the newts who don't fly do see my yak .	▨ the newts who don't fly do see my yak . → do the newts who don't fly see my yak ?

Figure 2: The difference between the training set and generalization set. To save space, this table uses some words not present in the vocabulary used to generate the examples. RC stands for "relative clause."

biases in humans must arise from architectural constraints on the neural connections in the brain rather than from constraints stated at the symbolic level, since symbolic constraints are unlikely to be specified in the genome. Here we use artificial neural networks to investigate this claim, testing whether high-level inductive biases can emerge from low-level architectural constraints.

2.2 Framing of the task

Following Frank and Mathis (2007) and McCoy et al. (2018), we train models to take a declarative sentence as input and to either output the same sentence unchanged, or to transform that sentence into a question. The task to be performed is indicated by the final input token, as in (3) and (4):

(3) a. *Input:* your zebra does read . DECL
b. *Output:* your zebra does read .

(4) a. *Input:* your zebra does read . QUEST
b. *Output:* does your zebra read ?

During training, all question-formation examples are consistent with both MOVE-FIRST and MOVE-MAIN, such that there is no direct evidence favoring one rule over the other (see Figure 2).

To assess how models generalize, we evaluate them on a generalization set consisting of examples where MOVE-MAIN and MOVE-FIRST make different predictions due to the presence of a relative clause on the subject (see sentence (2a)).

2.3 Evaluation metrics

We focus on two metrics. The first is **full-sentence accuracy on the test set**. That is, for examples drawn from the same distribution as the training set, does the model get the output exactly right?

For testing generalization to the withheld example type, a natural metric would be full-sentence accuracy on the generalization set. However, in preliminary experiments we found that most models rarely produce the exact output predicted by either MOVE-MAIN or MOVE-FIRST, as they tend to truncate the output, confuse similar words, and make other extraneous errors.

To abstract away from such errors, we use **first-word accuracy on the generalization set**. With both MOVE-FIRST and MOVE-MAIN, the first word of the question is the auxiliary that has been moved from within the sentence, so this word alone is sufficient to differentiate the two rules. For example, in the bottom right cell of Figure 2, MOVE-MAIN predicts having *do* at the start, while MOVE-FIRST predicts *don't*.[1]

2.4 Architecture

We used the sequence-to-sequence architecture in Figure 3 (Botvinick and Plaut, 2006; Sutskever et al., 2014). This model consists of two neural networks: the **encoder** and the **decoder**. The encoder is fed the input sentence one word at a time; after each word, the encoder updates its **hidden state**, a vector representation of the information encountered so far. After the encoder has been fed the entire input, its final hidden state (E_6 in Figure 3) is fed to the decoder, which generates an output sequence one word at a time based on its own hidden state, which is updated after each output word. The weights that the encoder and decoder use to update their hidden states and to generate outputs are learned via gradient descent.[2]

[1] We exclude cases where the two auxiliaries are the same. We also exclude cases where one auxiliary is singular and the other plural so that a model cannot succeed by using heuristics based on the grammatical number of the subject.

[2] We trained all models for 30,000 batches, with a batch size of 5; we continued to train until accuracy on the validation set had not improved for 3,000 batches (with validation

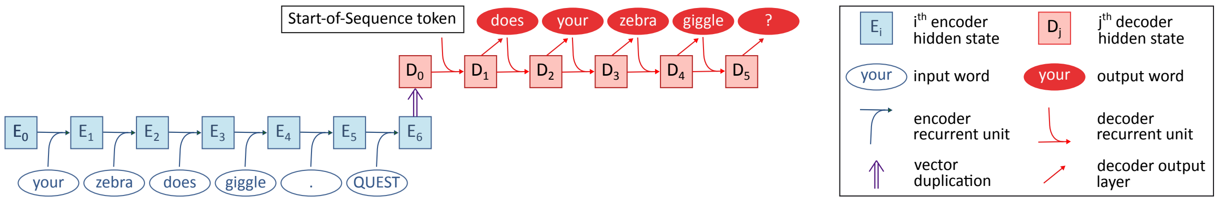


Figure 3: Linear sequence-to-sequence model.

	🚫	🌎	📖
SRN	0.00	0.93	1.00
GRU	0.88	0.77	1.00
LSTM	0.94	0.98	1.00

(a) Full-sentence accuracy on the test set

	🚫	🌎	📖
SRN	▨	0.43	0.64
GRU	0.01	0.78	0.17
LSTM	0.02	0.05	0.01

(b) First-word accuracy on the generalization set

Figure 4: Results for each combination of recurrent unit and attention type. All numbers are medians over 100 initializations. 🚫 = no attention; 🌎 = location-based attention; 📖 = content-based attention. A grayed-out cell indicates that the architecture scored below 50% on the test set.

2.5 Overview of experiments

Holding the task constant, we varied several fine-grained aspects of the training setup: two aspects of the architecture (namely, the recurrent unit and the type of attention), two types of hyperparameters (namely, hidden size and learning rate), and the random initialization. We also tested a higher-level modification of the architecture, namely the use of tree-based models rather than the sequential structure in Figure 3. Each aspect is discussed in more detail in the following sections.[3]

3 Recurrent unit and attention

The first two model components that we investigated were recurrent unit and attention.

3.1 Recurrent unit

The recurrent unit is the component that updates the hidden state after each word for each encoder and decoder. We experimented with three types of recurrent units: simple recurrent networks (SRNs; Elman, 1990), gated recurrent units (GRUs; Cho et al., 2014), and long short-term memory (LSTM) units (Hochreiter and Schmidhuber, 1997). SRNs and GRUs both use a single vector to represent a model's hidden state, while LSTMs use two vectors (the hidden state and the cell state). The other main difference between these recurrent units is that GRUs and LSTMs both use *gates*, which control what information is retained across time steps, while SRNs do not; GRUs and LSTMs differ in the number and types of gates they use.

3.2 Attention

In the basic model in Figure 3, the final hidden state of the encoder is the decoder's only source of information about the input. To avoid having such a bottleneck, it has become popular to add **attention** (Bahdanau et al., 2015), a feature that enables the decoder to consider all encoder hidden states (E_0 through E_6 in Figure 3) when generating hidden state D_i. The basic model has the only inputs to D_i being D_{i-1} and y_{i-1} (the previous output element); attention adds a third input, c_i, generated as a weighted sum of the encoder's hidden states (E_0 through E_n) using a weight vector α_i whose j^{th} element is denoted by $\alpha_i[j]$:

$$c_i = \sum_j \alpha_i[j] E_j \tag{1}$$

Implementations of attention vary in which inputs determine each weight $\alpha_i[j]$ (Graves et al., 2014; Luong et al., 2015; Chorowski et al., 2015). Attention can be solely *location-based*, where each α_i is determined solely from D_{i-1} (and potentially also y_{i-1}), so that the model chooses *where* to attend without first checking *what* it is attending to. Alternately, attention could be *content-based*, in which case each $\alpha_i[j]$ is determined from both D_{i-1} and E_j, such that the model does consider what it might attend to before attending to it. We test both location-based and content-based attention, and we also test models without attention.

set accuracy measured every 1,000 batches). During training, we used teacher forcing on 50% of training examples.

[3]Our code is available on GitHub. (URL anonymized.)

	Unsquashed	Squashed
GRU	0.99	0.77
LSTM	0.98	0.98

(a) Full-sentence accuracy on the test set

	Unsquashed	Squashed
GRU	0.54	0.78
LSTM	0.05	0.43

(b) First-word accuracy on the generalization set

Figure 5: Effects of squashing. All numbers are medians across 100 initializations. The standard versions of the architectures are the squashed GRU and the unsquashed LSTM.

3.3 Results

We trained models with all nine possible combinations of recurrent unit and attention type, using a hidden size of 256 and a learning rate of 0.001 for all experiments. The results are in Figure 4.

The SRN without attention failed on the test set, mainly because it often confused words that had the same part of speech, a known weakness of SRNs (Frank and Mathis, 2007). Therefore, its generalization set behavior is uninformative. The other architectures performed strongly on the test set ($>$ 50% full-sentence accuracy), so we now consider their generalization set performance. The GRU with location-based attention and the SRN with content-based attention both preferred MOVE-MAIN, while the remaining architectures preferred MOVE-FIRST. These results suggest that both the type of recurrent unit and the type of attention can qualitatively affect a model's inductive biases. Moreover, the *interactions* of these two factors can have drastic effects: with SRNs, content-based attention led to a preference for MOVE-MAIN while location-based attention led to a preference for MOVE-FIRST, while these types of attention had opposite effects with GRUs.

3.4 Discussion: LSTM vs. GRU

One striking result in Figure 4 is that LSTMs and GRUs display qualitative differences, even though the two architectures are often viewed as interchangeable and achieve similar performance in applied tasks (Chung et al., 2014). One difference between LSTMs and GRUs is that a squashing function is applied to the hidden state of a GRU to keep its values within the range $(-1, 1)$, while the cell state of an LSTM is not bounded. Weiss et al. (2018) demonstrate that such squashing leads to a qualitative difference in how these types of models generalize the ability to count. Such squashing may also explain the qualitative differences that we observe: intuitively, we might expect a network that can count to be more likely to choose a linear-order based generalization.

To test whether squashing increases a model's preference for MOVE-MAIN, we created a modified LSTM that includes squashing in the calculation of its cell state. The equations governing the weight update of a standard LSTM are as follows:

$$i_t = \sigma(W_i * [h_{t-1}, w_t] + b_i) \tag{2}$$
$$f_t = \sigma(W_f * [h_{t-1}, w_t] + b_f) \tag{3}$$
$$g_t = \tanh(W_g * [h_{t-1}, w_t] + b_g) \tag{4}$$
$$o_t = \sigma(W_o * [h_{t-1}, w_t] + b_o) \tag{5}$$
$$c_t = f_t * c_{t-1} + i_t * g_t \tag{6}$$
$$h_t = o_t * \tanh(c_t) \tag{7}$$

To create a new LSTM whose cell state exhibits squashing, like the hidden state of the GRU, we modified the LSTM cell state update in (6) to (8), where the new coefficients now add to 1:

$$c_t = \frac{f_t}{f_t + i_t} * c_{t-1} + \frac{i_t}{f_t + i_t} * g_t \tag{8}$$

Similarly, we also created a modified GRU that did not have the squashing usually present in GRUs. The standard GRU equations are:

$$r_t = \sigma(W_r[h_{t-1}, w_t] + b_r) \tag{9}$$
$$z_t = \sigma(W_z[h_{t-1}, w_t] + b_z) \tag{10}$$
$$\tilde{h} = \tanh(W_x[r_t * h_{t-1}, w_t] + b_x) \tag{11}$$
$$h_t = z_t * h_{t-1} + (1 - z_t) * \tilde{h} \tag{12}$$

The GRU's squashing comes from the fact that its update gate z merges the functionality of the input gate i and forget gate f of the LSTM (cf. equations 6 and 12). As a result, the input and forget weights are tied together in the GRU but not the LSTM. To make a non-squashed GRU, we therefore add an input gate i to the GRU:

$$i_t = \sigma(W_i[h_{t-1}, w_t] + b_i) \tag{13}$$

We then changed the hidden state update in Equation 12 so that z functions solely as a forget gate:

	32	64	128	256	512	1024	2048
SRN 🚫							
SRN 🌐		0.37	0.45	0.38	0.45	0.21	0.17
SRN 📖		0.55	0.63	0.61	0.56	0.60	0.54
GRU 🚫			0.03	0.03	0.00	0.00	0.00
GRU 🌐		0.45	0.44	0.75	0.91	0.95	0.97
GRU 📖	0.32	0.27	0.28	0.35	0.20	0.27	0.22
LSTM 🚫			0.02	0.02	0.01	0.00	0.01
LSTM 🌐		0.09	0.03	0.02	0.27	0.17	0.13
LSTM 📖		0.04	0.02	0.02	0.01	0.00	

(a) Hidden size results.

	5e-5	1e-4	5e-4	1e-3	5e-3	1e-2	5e-2
SRN 🚫							
SRN 🌐	0.40	0.44	0.44	0.38	0.39		
SRN 📖			0.66	0.61			
GRU 🚫			0.05	0.03	0.00	0.00	
GRU 🌐		0.51	0.53	0.75	0.90	0.75	
GRU 📖			0.00	0.35	0.10	0.42	
LSTM 🚫			0.03	0.02	0.00	0.01	0.03
LSTM 🌐			0.05	0.02	0.32	0.35	0.24
LSTM 📖			0.02	0.02	0.00	0.01	0.14

(b) Learning rate results.

Figure 6: Hyperparameter experiment results: median first-word accuracy on the generalization set over 10 random initializations. In (a), the learning rate was always 0.001; in (b), the hidden size was always 256. Cells that are grayed out indicate that the median full-sentence accuracy on the test set was below 0.5. 🚫 = no attention; 🌐 = location-based attention; 📖 = content-based attention.

$$h_t = z_t * h_{t-1} + i_t * \tilde{h} \qquad (14)$$

Models with squashing chose MOVE-MAIN more often than models without squashing (Figure 5), suggesting that such squashing is one factor that causes GRUs to behave differently than LSTMs.

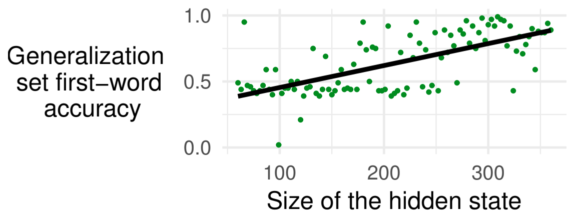


Figure 7: Effect of hidden state size for a GRU with location-based attention.

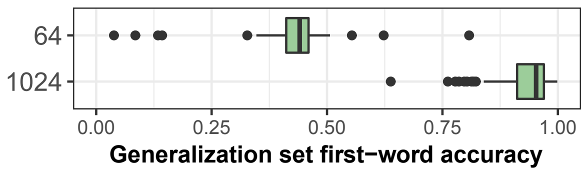


Figure 8: Hidden size 64 vs. 1024 for 100 GRUs with location-based attention.

4 Hyperparameters

We now vary two hyperparameters: the hidden state size and the learning rate. We do not have hypotheses about how the learning rate would be relevant for this task, but there are plausible reasons why hidden size might matter: a smaller hidden size might force a model to learn more compact representations, which would lead to a bias for generalizations that can be described efficiently. For a language that can be most compactly described in hierarchical terms, such a bias would favor MOVE-MAIN (Perfors et al., 2011). On the other hand, the sequential structure of our model is closer to the linear structure underlying MOVE-FIRST than the tree structure underlying MOVE-MAIN. Thus, a more compressed hidden state might bias a sequential model toward MOVE-FIRST by giving it less capacity to develop a representation that deviates from its own structure.

4.1 Hidden state size

For many architectures, hidden size has little effect (Figure 6a). However, for GRUs with location-based attention, larger hidden sizes are associated with greater preferences for MOVE-MAIN.

We ran two follow-up experiments to test this effect. First, we trained 100 GRUs with location-based attention with different hidden sizes uniformly spaced in the range [60, 357] (Figure 7). We found a significant positive correlation between hidden size and generalization set first-word accuracy (Spearman's correlation coefficient = 0.62; $p < 10^{-11}$). Moreover, this correlation cannot be attributed to overall stronger performance, since test set full-sentence accuracy was negatively correlated with hidden size (Spearman's correlation coefficient = -0.56).

Second, we ran 100 random initializations each for GRUs with location-based attention with hidden sizes 64 and 1024 to see if the effects of these hidden sizes persist with a greater sample size than the sample size of 10 in Figure 6a. As shown in Figure 8, there was a consistent differ-

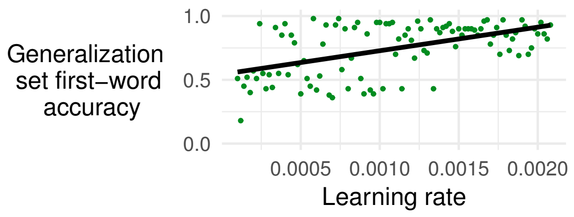


Figure 9: Effect of learning rate for a GRU with location-based attention.

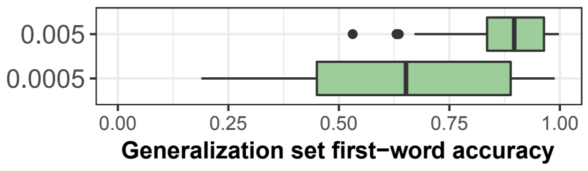


Figure 10: Learning rate 0.005 vs. 0.0005 for 100 GRUs with location-based attention.

ence between these two hidden sizes, with 1024 strongly preferring MOVE-MAIN and 64 not showing a strong preference for either generalization.

4.2 Learning rate

For many models, the learning rate did not noticeably affect the model's generalization behavior (Figure 6b). However, for the GRU with location-based attention, higher learning rates were associated with greater preferences for MOVE-MAIN.

We ran two follow-up experiments to test this effect. First, we trained 100 models with different learning rates uniformly spaced in the interval [0.0001, 0.00208] (Figure 9). Learning rate and generalization set first-word accuracy are significantly positively correlated (Spearman's correlation coefficient = 0.42; $p < 10^{-4}$). Learning rate is also positively correlated with test set full-sentence accuracy; however, a likelihood ratio test shows that learning rate significantly predicts generalization set first-word accuracy beyond the effects of test set full-sentence accuracy ($p < 10^{-4}$).

Second, we ran 100 random initializations each for learning rates 0.0005 and 0.005 to see if their differences persist with a larger sample size than in Figure 6b. The higher learning rate did tend to have a greater preference for MOVE-MAIN than the lower learning rate (Figure 10).

5 Random initialization

Figure 11 shows that the random initialization of a model can qualitatively affect how that model generalizes even if it has little effect on the performance on the test set (for a similar finding, see Weber et al. (2018)).

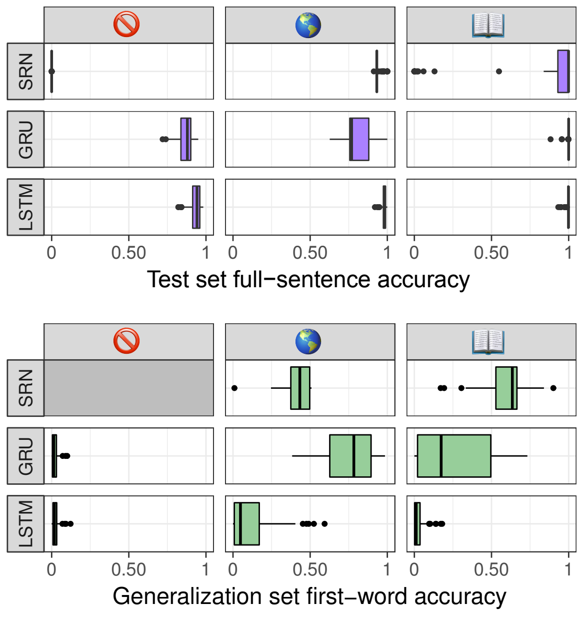


Figure 11: Variation in performance based on random initialization. One cell in the generalization plot is grayed out because of test set performance below 0.50. 🚫 = no attention; 🌏 = location-based attention; 📖 = content-based attention.

6 Tree models

So far we have tested whether properties that are not interpretably related to the task nevertheless affect how a model generalizes. We now turn to a related but opposite question: when a model's design is meant to give it a task-relevant inductive bias, does this design succeed at giving the model this bias? The question formation task hinges upon hierarchical structure, so we investigate several models designed to use hierarchical structure to see whether they exhibit a hierarchical bias.

6.1 Tree model that learns to parse

The first hierarchical model that we test is the Ordered Neurons LSTM (ON-LSTM; Shen et al., 2018). This model is not given the tree structure of each sentence as part of its input but must instead learn to implicitly parse its input and to use that parse to perform the task. This implicit tree structure is created by imposing a stack-like constraint on the updates to the values in the cell state vector of an LSTM: the degree to which the i^{th} value is updated must always be less than or equal to the degree to which the j^{th} value is updated for

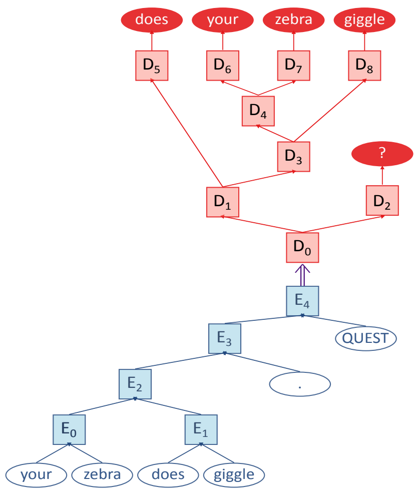


Figure 12: RNN with a tree-based encoder and tree-based decoder.

all $j \leq i$. This hierarchy of cell-state values adds an implicit tree structure to the model, where each level in the tree is defined by a soft depth in the cell state to which that level extends.

We find that the ON-LSTM achieves a test set full-sentence accuracy of 0.93 but a generalization set first-word accuracy of 0.05, showing a strong preference for MOVE-FIRST over MOVE-MAIN, contrary to what one would expect from a model with a hierarchical inductive bias.

6.2 Tree models given the correct parse

The ON-LSTM results show that hierarchical structure alone is not sufficient to induce a bias for MOVE-MAIN, suggesting that constraints on *which* trees the model uses may also be necessary.

To assess whether providing the correct parse trees is sufficient to bias a model toward MOVE-MAIN, we tested tree-RNNs that were explicitly fed the correct parse tree. A model's encoder and/or its decoder can be tree-based; Figure 12 shows a model where both the encoder and decoder are tree-based. For the tree-based encoder, we use the Tree-GRU from Chen et al. (2017). This model composes together the vector representations for each pair of sister nodes in the tree to generate a vector representing their parent node. It performs this composition bottom-up, starting with the word embeddings at the leaf nodes and ending with a single vector representing the root node (E_4 in Figure 12), which acts as the encoding of the input sentence. For the tree-based decoder, we use a model based on the Tree-LSTM decoder from (Chen et al., 2018), but using a GRU instead of an LSTM, for consistency with the tree encoder. This tree decoder is the mirror image of the tree encoder: starting with the vector representation of the root note (D_0 in Figure 12), it takes the vector representation of a parent node and outputs two vectors, one for the left child and one for the right child, until it reaches a leaf node, where it outputs a word. We test models with a tree-based encoder and linear decoder, a linear encoder and tree-based decoder, or a tree-based encoder and tree-based decoder, all without attention.

The results for these models are in Figure 13, along with the previous results of the fully linear GRU without attention for comparison. The model with a tree-based encoder and linear decoder preferred MOVE-FIRST, like the fully linear model. Only the models with a tree-based decoder preferred MOVE-MAIN, consistent with the finding of McCoy et al. (2019) that it is the decoder that determines an encoder-decoder model's representations. However, the model with a linear encoder and a tree decoder failed on the test set (as measured by full sentence accuracy), so the only model that both succeeded on the test set and showed a bias toward a MOVE-MAIN generalization was the fully tree-based model (Tree/Tree).[4] The behavior of this Tree/Tree model was striking in another way as well: Its generalization set *full-sentence* accuracy was 69%, while all other models—even those that achieved high *first-word* accuracy on the generalization set—had close to 0% generalization set full-sentence accuracy. These results show that an architecture designed to have a certain inductive bias might, but will not necessarily, display the intended bias.

7 The reinflection task

We have shown that several models reliably preferred MOVE-MAIN over MOVE-FIRST. However, this behavior alone does not necessarily mean that these models have a hierarchical bias, because a preference for MOVE-MAIN might arise not from a hierarchical bias but rather from some task-specific factors such as the prevalence of certain n-grams (Kam et al., 2008; Berwick et al., 2011). A true hierarchical bias would lead a model to adopt hierarchical generalizations across training

[4]We do not have an explanation for the failure of the linear/tree model on the test set; most of its errors involved confusion among words that had the same part of speech (e.g. generating *my* instead of *your*).

Model	Full-sentence test acc.	First-word gen. acc.
Linear/Linear	0.88	0.03
Linear/Tree	0.00	0.90
Tree/Linear	0.96	0.13
Tree/Tree	0.96	0.99

Figure 13: Results with tree-based models (medians over 100 random initializations). Model names indicate encoder/decoder; e.g., Linear/Tree has a linear GRU encoder and a tree-GRU decoder.

tasks; by contrast, we hypothesize that other factors (such as a bias for focusing on n-gram statistics) will be more sensitive to details of the task and will thus be unlikely to consistently produce hierarchical preferences. To test the robustness of the hierarchical preferences of our models, we introduce a second task, **tense reinflection**.

7.1 Reinflection task

The reinflection task uses English subject-verb agreement to illuminate a model's syntactic generalizations (Linzen et al., 2016). The model is fed a past-tense English sentence as input. It must then either output that sentence unchanged or transformed to the present tense, with the final word of the input indicating the task to be performed:

(5) a. *Input:* my zebra swam . PAST
 b. *Output:* my zebra swam .

(6) a. *Input:* my zebra swam . PRESENT
 b. *Output:* my zebra swims .

Because the past tense in English does not inflect for number (e.g., the past tense of *swim* is *swam* whether the subject is singular or plural), the model must determine from context whether each verb should be singular or plural. Example (6) is consistent with two salient rules for determining which aspects of the context are relevant:

(7) AGREE-SUBJECT: Each verb should agree with its hierarchically-determined subject.

(8) AGREE-RECENT: Each verb should agree with the linearly most recent noun.

Though these rules make the same prediction for (6), they make different predictions for other examples, such as (9a), for which AGREE-SUBJECT predicts (9b) while AGREE-RECENT predicts (9c):

Model	Full-sentence test acc.	Main-verb gen. acc.
SRN 🚫	0.00	
SRN 🌍	1.00	0.00
SRN 📖	1.00	0.00
GRU 🚫	0.92	0.01
GRU 🌍	0.80	0.00
GRU 📖	1.00	0.00
LSTM 🚫	0.95	0.02
LSTM 🌍	0.99	0.00
LSTM 📖	1.00	0.00
ON-LSTM 🚫	0.95	0.03
Tree/Tree 🚫	0.96	0.75

Figure 14: Reinflection results (medians over 100 initializations). 🚫 = no attention; 🌍 = location-based attention; 📖 = content-based attention.

(9) a. my zebra by the yaks swam . PRESENT
 b. my **zebra** by the yaks **swims** .
 c. my zebra by the **yaks swim** .

Similar to the setup for the question-formation experiments, we trained models on examples for which AGREE-SUBJECT and AGREE-RECENT made the same predictions and evaluated the trained models on examples where the rules make different predictions. We ran this experiment with all 9 linear models ([SRN, GRU, LSTM] x [no attention, location-based attention, content-based attention]), the ON-LSTM, and the model with a tree-based encoder and tree-based decoder that were provided the correct parse trees. We evaluated these models on the full-sentence accuracy on the test set and also main-verb accuracy for the generalization set (that is, the proportion of generalization set examples for which the main verb was correctly predicted, such as choosing *swims* rather than *swim* in the output for (9a)).

All sequential models, even the ones that preferred MOVE-MAIN in the question formation experiments, overwhelmingly chose AGREE-RECENT for this reinflection task (Figure 14), consistent with the results of a similar experiment done by Ravfogel et al. (2019). The ON-LSTM also displayed a strong preference for AGREE-RECENT. By contrast, the fully tree-based model preferred the hierarchical generalization AGREE-SUBJECT. Thus, although the question-formation experiments showed qualitative differences in linear models' inductive biases, this experiment shows that those differences cannot be described as some linear models having a general (if implicit) hierarchical bias, since the reinflection task shows that is not the case. What the relevant bias for these models *is* remains unclear; all that we claim to show is that it is not a hierarchical bias. Overall, the model with both a tree-based encoder and a tree-based decoder is the only model we tested that can plausibly be said to have a generic hierarchical bias, as it is the only one that behaved consistently with such a bias across both tasks.

8 Discussion

We have found that all factors we tested can qualitatively affect a model's inductive biases but that a hierarchical bias—which has been argued to underlie children's acquisition of syntax—only arose in a model with explicit hierarchical structure.

8.1 Relation to *Rethinking Innateness*

Our experiments were motivated by the book *Rethinking Innateness* (Elman et al., 1998) which argued that humans' inductive biases must arise from constraints on the wiring patterns of the brain. Our results support two conclusions from this book. First, those authors argued that "Dramatic effects can be produced by small changes" (p. 359), which we also found in the experiments showing that seemingly insignificant aspects of training qualitatively affect the ways models generalize. Second, they also argued that "What appear to be single events or behaviors may have a multiplicity of underlying causes" (p. 359); in our case, we found that a model's generalization behavior results from some combination of factors that interact in hard-to-interpret ways; e.g., we observed that changing the type of attention had different effects in SRNs than in GRUs.

The dramatic effects of these low-level factors offer some support for the claim that humans' inductive biases can arise from fine-grained architectural constraints in the brain. However, this support is only partial, since our only model that robustly displayed a hierarchical bias did not derive this bias from low-level architectural properties but rather from the explicit encoding of linguistic structure.

8.2 Relation to the poverty of the stimulus

In the context of the poverty of the stimulus debate, the importance of tree structure in our experiments suggests that the biases displayed by humans when acquiring English question formation must specifically make reference to hierarchical structure, as opposed to having a hierarchical preference emerge from more general principles. Moreover, since the only hierarchical model that succeeded was one that took the correct parse trees as input, our results suggest that the bias is not only about hierarchical structure but also includes biases governing which specific trees a model will learn. It is implausible that the correct parse trees would be innately available to children as they were for our models; however, it is possible that, by the time children begin to acquire question formation, they have already learned the basic syntax of their language and that this knowledge helps guide their acquisition of question formation.

An important caveat for extending our conclusions to humans is that our training language is not representative of the data available to children, as it omits all semantic and prosodic information as well as many syntactic constructions. It is possible that, with data closer to a child's input, more general inductive biases might succeed.

8.3 Practical takeaways

Our results leave room for three possible approaches to imparting a model with a desired inductive bias. First, one could search the space of hyperparameters and random initializations to find a setting that leads to the desired generalization. However, this may be ineffective: We conducted a thorough search of this space for our sequential (non-tree-based) models but did not find a setting that led to hierarchical generalization across tasks.

A second option is to use more interpretable architectures which explicitly incorporate the desired inductive bias. Our results suggest that this approach is more viable than hyperparameter search, as this approach did yield models that reliably generalized hierarchically. However, even this approach only worked with a very strong version of this bias (specifically, having the correct trees provided in both the encoder and decoder).

A final option, which we did not test, would be to change the training regimen by either adding a pre-training task or by using multi-task learning (Caruana, 1997; Collobert and Weston, 2008; Enguehard et al., 2017), where the additional task is designed to impart the desired bias. We leave such experiments as a direction for future work.

References

Ben Ambridge, Caroline F. Rowland, and Julian M. Pine. 2008. Is structure dependence an innate constraint? New experimental evidence from children's complex-question production. *Cognitive Science*, 32(1):222–255.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.

Robert C. Berwick, Paul Pietroski, Beracah Yankama, and Noam Chomsky. 2011. Poverty of the stimulus revisited. *Cognitive Science*, 35(7):1207–1242.

Matthew M. Botvinick and David C. Plaut. 2006. Short-term memory for serial order: A recurrent neural network model. *Psychological Review*, 113(2):201.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017. Improved neural machine translation with a syntax-aware encoder and decoder. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1936–1945. Association for Computational Linguistics.

Xinyun Chen, Chang Liu, and Dawn Song. 2018. Tree-to-tree neural networks for program translation. *arXiv preprint arXiv:1802.03691*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of EMNLP*.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Noam Chomsky. 1980. Rules and representations. *Behavioral and Brain Sciences*, 3(1):1–15.

Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, pages 577–585. MIT Press.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NeurIPS Deep Learning and Representation Learning Workshop*.

Alexander Clark and Rémi Eyraud. 2007. Polynomial identification in the limit of substitutable context-free languages. *Journal of Machine Learning Research*, 8(Aug):1725–1745.

Alexander Clark and Shalom Lappin. 2010. *Linguistic Nativism and the Poverty of the Stimulus*. John Wiley & Sons.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167. ACM.

Stephen Crain and Mineharu Nakayama. 1987. Structure dependence in grammar formation. *Language*, pages 522–543.

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

Jeffrey L. Elman, Elizabeth A. Bates, Mark H. Johnson, Annette Karmiloff-Smith, Domenico Parisi, and Kim Plunkett. 1998. *Rethinking innateness: A connectionist perspective on development*. MIT press.

Émile Enguehard, Yoav Goldberg, and Tal Linzen. 2017. Exploring the syntactic abilities of RNNs with multi-task learning. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 3–14, Vancouver, Canada. Association for Computational Linguistics.

Hartmut Fitz and Franklin Chang. 2017. Meaningful questions: The acquisition of auxiliary inversion in a connectionist model of sentence production. *Cognition*, 166:225–250.

Robert Frank and Donald Mathis. 2007. Transformational networks. In *Proceedings of the Workshop on Psychocomputational Models of Human Language Acquisition*. Cognitive Science Society.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Xuân-Nga Cao Kam, Iglika Stoyneshka, Lidiya Tornyova, Janet D. Fodor, and William G Sakas. 2008. Bigrams and the richness of the stimulus. *Cognitive Science*, 32(4):771–787.

Barbara Landau, Linda B. Smith, and Susan S. Jones. 1988. The importance of shape in early lexical learning. *Cognitive Development*, 3(3):299–321.

Julie Anne Legate and Charles D. Yang. 2002. Empirical re-assessment of stimulus poverty arguments. *The Linguistic Review*, 18(1-2):151–162.

John D. Lewis and Jeffrey L. Elman. 2001. Learnability and the statistical structure of language: Poverty of stimulus arguments revisited. In *Proceedings of the 26th Annual Conference on Language Development*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

R. Thomas McCoy, Robert Frank, and Tal Linzen. 2018. Revisiting the poverty of the stimulus: Hierarchical generalization without a hierarchical bias in recurrent neural networks. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society*, pages 2093–2098, Madison, WI.

R. Thomas McCoy, Tal Linzen, Ewan Dunbar, and Paul Smolensky. 2019. RNNs implicitly implement tensor-product representations. In *International Conference on Learning Representations*.

Amy Perfors, Joshua B. Tenenbaum, and Terry Regier. 2011. The learnability of abstract syntactic principles. *Cognition*, 118(3):306–338.

Geoffrey K. Pullum and Barbara C. Scholz. 2002. Empirical assessment of stimulus poverty arguments. *The Linguistic Review*, 18(1-2):9–50.

Shauli Ravfogel, Yoav Goldberg, and Tal Linzen. 2019. Studying the inductive biases of RNNs with synthetic variations of natural languages. *arXiv preprint arXiv:1903.06400*.

Florencia Reali and Morten H. Christiansen. 2005. Uncovering the richness of the stimulus: Structure dependence and indirect statistical evidence. *Cognitive Science*, 29(6):1007–1028.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2018. Ordered neurons: Integrating tree structures into recurrent neural networks. *arXiv preprint arXiv:1810.09536*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

Noah Weber, Leena Shekhar, and Niranjan Balasubramanian. 2018. The fine line between linguistic generalization and failure in seq2seq-attention models. In *Proceedings of the Workshop on Generalization in the Age of Deep Learning*, pages 24–27. Association for Computational Linguistics.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. On the practical computational power of finite precision rnns for language recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Vol-*

ume 2: Short Papers), pages 740–745. Association for Computational Linguistics.