Filler-gaps that neural networks fail to generalize

Debasmita Bhattacharya and **Marten van Schijndel**
Department of Linguistics
Cornell University
{db758|mv443}@cornell.edu

Abstract

It can be difficult to separate abstract linguistic knowledge in recurrent neural networks (RNNs) from surface heuristics. In this work, we probe for highly abstract syntactic constraints that have been claimed to govern the behavior of filler-gap dependencies across different surface constructions. For models to generalize abstract patterns in expected ways to unseen data, they must share representational features in predictable ways. We use cumulative priming to test for representational overlap between disparate filler-gap constructions in English and find evidence that the models learn a general representation for the existence of filler-gap dependencies. However, we find no evidence that the models learn any of the shared underlying grammatical constraints we tested. Our work raises questions about the degree to which RNN language models learn abstract linguistic representations.

1 Introduction

While sentences appear highly variable on the surface, many syntactic constructions share the same underlying constraints, which determine their acceptability or grammaticality, i.e. the extent to which they are considered "well formed" through adherence to the rules of grammar. One of the strongest pieces of evidence for the existence of these shared underlying constraints are filler-gap constructions such as:

(1) What does Leslie like __?

Filler-gap constructions contain a dependency between an overt filler (*what* in (1)) and a gap site (underlined above). The filler is bound to a referent (e.g., *Robin's painting*) that can fill the gap:

(2) Leslie likes Robin's painting.

There are well-known restrictions (*islands*; Ross, 1967) that prevent certain words from participating in a filler-gap dependency. For example, it isn't possible to form a filler-gap dependency with prenominal (left-branch) noun modifiers:[1]

(3) *Whose does Leslie like __ painting?

Further, very different filler-gap constructions obey shared underlying principles (e.g., *subjacency*; Chomsky, 1973). In this work, we probe recurrent neural network (RNN) language model understanding[2] of these underlying principles in English.

Recent work has claimed that recurrent neural network language models understand filler-gap dependencies (Chowdhury and Zamparelli, 2018; Wilcox et al., 2018, 2019). However, behavioral probing has suggested that this understanding is relatively superficial and doesn't reflect the underlying constraints that govern filler-gap acceptability (Chaves, 2020). An intermediate possibility, which we explore in this paper, is that RNNs do acquire a basic understanding of the underlying constraints but that the learned representations of the constraints are too weak to correctly drive behavior in behavioral probing tasks.

We use cumulative priming (van Schijndel and Linzen, 2018; Prasad et al., 2019) to test for representational overlap between disparate constructions. While we find some evidence that RNNs learn a general representation for the existence of filler-gap dependencies (in keeping with Wilcox et al., 2018, 2019), we find no evidence that RNNs

[1]Throughout this paper we adopt notational conventions from the syntax literature. * indicates an ungrammatical or unacceptable sentence, and ?? indicates a sentence whose acceptability is between wholly acceptable and wholly unacceptable.

[2]We use terms such as 'understand' and 'represent' to refer to the ability of RNN models to process language successfully. We do not mean to imply that models have true, deep understanding of linguistic phenomena as humans do. We simply use these terms for convenience.

learn shared representation of the associated governing constraints (in keeping with Chaves, 2020).

Several recent papers have highlighted ways in which RNN behavior actually reflects shallow surface heuristics (McCoy et al., 2019; Chaves, 2020; Davis and van Schijndel, 2020). Note that the representational overlap we seek in the present study is actually a requirement for appropriate generalization of abstract knowledge to unseen data. This is what differentiates abstract knowledge from the surface heuristics that make RNN behavior fragile to adversarial methods. Therefore, our finding that RNNs fail to learn any shared abstract constraints across filler-gap constructions despite being sensitive to the existence of filler-gap dependencies raises questions about the ability of these models to learn abstract generalizable linguistic patterns.

2 Background

There are a number of different kinds of filler-gap constructions whose behavior is governed by a variety of different underlying constraints (see Table 1). Previous work has probed model understanding of filler-gap dependencies by testing model performance on individual construction types rather than the underlying constraints that might govern them (Chowdhury and Zamparelli, 2018; Wilcox et al., 2018, 2019, c.f. Chaves 2020). These studies have followed the logic of the subject-verb agreement probing literature (e.g., Linzen et al., 2016): a model that understands filler-gap dependencies should assign greater probability to grammatical filler-gap dependencies than to ungrammatical filler-gap dependencies. However, certain properties are shared across a variety of filler-gap constructions, giving rise to the hypothesis in the syntax literature that there are shared constraints that underlie multiple different filler-gap constructions. Chaves (2020) identifies failure cases where neural language models incorrectly rank sentences containing acceptable and unacceptable filler-gap dependencies, possibly because they have overlearned the individual filler-gap constructions without understanding the broader underlying constraints. In this paper, we test whether models understand four underlying filler-gap constraints that have been widely studied in the syntax literature. Specifically, we test whether multiple constructions that are governed by a single constraint share any representational features that are not present in other constructions. If such selective representational overlap exists, it could be an indication that the models do understand the underlying constraints but that their representation is too weak to correctly rank acceptable/unacceptable sentence pairs.

Cumulative priming has been introduced as a method for probing the linguistic representations encoded in RNNs (van Schijndel and Linzen, 2018; Prasad et al., 2019; Lepori et al., 2020).[3] This approach involves fine-tuning pretrained models for a single epoch with a small amount of additional training data. The pretrained model acts as a filter on the linguistic features learned during fine-tuning. More salient features will be affected by the fine-tuning to a larger degree than less salient features. By measuring the responsiveness of the model to linguistic input before and after priming (fine-tuning), researchers can identify which linguistic features are salient for the pretrained model. Importantly, if one construction primes a different construction, there is representational overlap between the two constructions within the model. Therefore, this approach provides a direct method of separating abstract linguistic knowledge from surface heuristics.

3 Constructions

We analyze eight types of syntactic constructions involving filler-gap dependencies in English. Syntacticians often refer to these dependencies as extractions, which alludes to the linguistic theory that filler phrases originate at the gap site before being *extracted* to the filler site during language production.

3.1 Adjunct islands

An adjunct island is formed from an adjunct clause, out of which wh-extraction is not possible. Adjunct clauses are introduced by *because*, *if*, and *when*, as well as by relative clauses.

(4) a. She ate her hat because they announced the plan.

b. ***What** did she eat her hat because they announced __?

[3]For example, Prasad et al. (2019) use cumulative priming to show that RNN language models cluster multiple different kinds of relative clauses together in representation space (suggesting an abstract relative clause representation), and that simple syntactic features (passivity, relative clause reduction) are similarly clustered. However, the syntactic abstractions probed by Prasad et al. (2019) are still tied to surface cues.

Construction	Examples	L	S	E	D
Adjunct island	*What did you go home because you needed to do __?	-		-	
Wh- island	*Whom did Susan ask why Sam was waiting for __?		-		(+)
Subject island	*Who is that __ went home likely?	-	-	-	
Left branch island	*Whose does Susan like __ account?		-		(+)
Coordinate structure island	*What did Sam eat __ and broccoli?			+	
Complex NP island	*What did you hear the claim that Fred solved __?		-		
Object extraction	Who is it probable that Bill likes __?	+		+	(+)
Non-bridge verb island	*How did she whisper that he had died __?	?	?		

Table 1: Island constructions (rows) and the associated underlying constraints (L-marking, Subjacency, and the Empty Category Principle) that govern their behavior. We only examine constraints that are hypothesized to apply to multiple of our construction types. For constraints that are thought to be particularly influential of a construction, we denote the influence with +/-. For example, subject island extractions are unacceptable because they violate L-marking, while object extractions are acceptable because they adhere to L-marking. Discourse-Linking (D-linking) is an optional shared feature that can make some unacceptable constructions more acceptable.

3.2 Wh-islands

A wh-island is created by an embedded sentence which is introduced by a wh-word. Extraction out of a wh-island results in an unacceptable sentence.

(5) a. Sam wonders who solved the problem.
b. ***What** does Sam wonder who solved __?

3.3 Subject islands

A subject island is formed from a subject clause or a subject phrase, out of which wh-movement is not possible.

(6) a. That he has met Julia Roberts is unlikely.
b. ***Who** is that __ has met Julia Roberts unlikely?

(7) a. The rumour about Susan was circulating.
b. ***Whom** was the rumour about __ circulating?

3.4 Left branch islands

Left branch islands consist of noun phrases with modifiers, such as possessive determiners and attributive adjectives, that appear on a left branch under the noun. These preceding modifiers of a noun cannot be extracted from a noun phrase.

(8) a. Eric likes John's boat.
b. ***Whose** does Eric like __ boat?

(9) a. You failed the difficult test.
b. ***How difficult** did you fail the __ test?

3.5 Coordinate structure islands

Coordinate structure islands allow extraction out of a conjunct of a coordinate structure only if the extraction affects all the conjuncts of the coordinate structure equally, i.e. if the extraction occurs across the board.

(10) a. They ordered [tiramisu] and [espresso].
b. ***What** did they order [tiramisu] and __?
c. ***What** did they order __ and [espresso]?

(11) a. Alicia [gave a guitar to me] and [loaned a trumpet to you].
b. **What** did Alicia [give __ to me] and [loan __ to you]?

3.6 Complex noun phrase islands

Complex noun phrase islands ban extraction from the clausal complement of a noun, and from a relative clause modifying a noun.

(12) a. You heard the rumour that Bill speaks a Balkan language.
b. ***What** did you hear the rumour that Bill speaks __?

(13) a. They hired someone who cleans a dirty surface.
b. ***What dirty surface** did they hire someone who cleans __?

3.7 Object extraction

Object extraction is when a filler-gap dependency involves an object clause or phrase. In contrast to subject islands, object extraction produces acceptable sentences.

(14) a. She told me that her mother is a teacher.
 b. **Her mother**, she told me __, is a teacher.

(15) a. It is important to invite Will to our party.
 b. **Who** is it important to invite __ to our party?

3.8 Non-bridge verb islands

Non-bridge verb islands ban extraction out of that-clause verb complements when the matrix verb is a non-bridge verb. Non-bridge verbs include manner-of-speaking verbs, such as whisper or shout.

(16) a. She thinks that he died in his sleep.
 b. **How** does she think that he died __?

(17) a. She whispered that he had died in his sleep.
 b. ***How** did she whisper that he had died __?

The unacceptability of non-bridge verb islands hinges on the frequency of the verb (Liu et al., 2019). Therefore, the degree to which different constraints govern its behavior is hotly debated. As such, we do not use this construction when probing for underlying contraint knowledge. However, we do include grammatical variants of this construction (16-b) as acceptable stimuli in our general filler-gap analysis (Section 7.2).

4 Constraints

We analyze four underlying syntactic principles that have been hypothesized to govern the behavior of the above constructions.

4.1 Subjacency

Subjacency (Chomsky, 1973) is defined in terms of the notion of extraction mentioned in the previous section. Syntacticians theorize that a filler is iteratively extracted from its gap site to particular possible landing sites during language production. The subjacency principle states that extraction is only permitted if all landing site positions that intervene between the filler and the gap are unfilled during the extraction process. If the possible structural positions are unavailable because they are filled with another lexical item, extraction is blocked and the resulting filler-gap dependency is deemed ungrammatical. The effect of the subjacency constraint can be observed in the above examples of Wh-islands, subject islands, left branch islands, and complex noun phrase islands.

4.2 Empty Category Principle

The Empty Category Principle (ECP; Kayne, 1980; Chomsky, 1981) is a syntactic constraint that requires a gap be properly governed. To be properly governed, gaps must be identifiable as empty positions in the surface structure of a sentence, which allows a tree structure to "remember" what has happened at earlier stages of a sentence's derivation. Adherence to this constraint makes extraction of a wh-word from a subject or adjunct position ungrammatical, while extraction from an object position or from a coordinate structure island is grammatical.

4.3 L-marking

L-Marking (Chomsky, 1986) is a process that defines the types of categories that act as barriers to movement, including extraction. A category is L-marked if and only if it gets its theta role from a lexical head. A theta role specifies the number and type of arguments that are syntactically required by a particular verb. For example, direct objects in English receive theta roles from the main verb, while adjuncts and subjects do not. Movement is grammatical only when it occurs out of an L-marked phrase, as in object extraction but not in adjunct or subject islands.

4.4 D-linking

Discourse-linking or D-linking (Pesetsky, 1987) indicates that there is a pre-existing contextual relationship between a filler and its associated noun phrase (e.g., *which man*). D-linked phrases contrast with non-discourse linked interrogative pronouns such as *who*, which do not necessarily imply familiar discourse entities. Left branch island extractions and wh-island extractions become more acceptable with a D-linked wh-phrase (Pesetsky, 1987; Atkinson et al., 2015). For example:

(18) a. ??**Which book** did Will ask why John read __?

b. ***What** did Will ask why John read __?

Since D-linking is an optional feature that can be added to filler-gap constructions, it isn't something that can be violated, per se. Therefore, in our analyses to probe for D-linking knowledge we only test constructions that adhere to D-linking (18-a).

5 Models

We focused in this work on recurrent neural language models with long short-term memory units (LSTMs; Hochreiter and Schmidhuber, 1997). We analyzed five of the highest performing models released by van Schijndel et al. (2019),[4] who showed that these models perform comparably to state-of-the-art transformers GPT and BERT on many simple syntactic agreement tasks. The models are 2-layer LSTMs with 400 hidden units per layer, each with a unique random initialization, trained on 80 million training tokens of English Wikipedia data. Analyzing multiple similar models with different random seeds helps ensure that our results are more representative of a class of models rather than simply revealing how a single exceptional model behaves (e.g., BERT; Devlin et al., 2019). This is very important given the speed with which new individual models supplant each other in the literature. Our results are averaged across all five of our models.

6 Method

We generated 40 sentences per construction type, partitioned into a prime set (15 sentences) and a test set (25 sentences). Sentences are available in the supplementary materials.

We measured model performance as the average surprisal (negative log-probability; Shannon, 1948; Hale, 2001) experienced by a model M when processing each word w_i of each sentence s_j in a set S:

$$\text{perf}(M, S) = -\frac{1}{|S|}\sum_{j=1}^{|S|}\frac{1}{|s_j|}\sum_{i=1}^{|s_j|}\log \text{P}_M(w_i \mid w_{0..i-1}) \quad (1)$$

Priming is achieved by giving the model a single training epoch on the set of priming stimuli S_P. This process produces a modified model M'_P whose performance on a test set S_T differs from the original model in a way that gives insight into the representations of the original model. One way to think about this is that pretraining provides a certain kind of model initialization. Priming the model moves the model representations along gradients which are characterized by an interaction of the priming stimuli and the initial pre-trained model state. If a set of priming stimuli has a consistent set of features, the initial state's sensitivity to that set of features can be probed with those stimuli. Following Prasad et al. (2019), we denote this raw effect of priming (also known as adaptation) as:

$$\mathbb{A}(S_T \mid S_P, M) = \text{perf}(M, S_T) - \text{perf}(M'_P, S_T) \quad (2)$$

We are actually interested in the interaction between the original model and the priming set, but the above measure also includes the interaction between the original model and the test set. Less expected (more surprising) test constructions can produce larger measures of priming simply because the original model has more room for improvement (Prasad et al., 2019). Therefore, we used linear regression to predict the size of the priming effect using the original model's performance on the test set:

$$\mathbb{A}(S_T \mid S_P, M) \sim \beta_0 + \beta_1\text{perf}(M, S_T) + \epsilon \quad (3)$$

To obtain a more appropriate *adaptation effect* (AE) for analysis, we subtracted out the predicted linear relation between the original model's test performance and the size of the final priming effect:

$$AE(S_T \mid S_P, M) = \mathbb{A}(S_T \mid S_P, M) - \beta_1\text{perf}(M, S_T) \quad (4)$$

This measure of priming more directly reflects the interaction of the original model with the priming set, normalizing the adaptation effect by each model and prime construction, and producing a comparable measure to that studied by Prasad et al. (2019). Across all analyses, greater values of the adaptation effect indicate greater similarity between adaptation and test structures.

Recently, Kodner and Gupta (2020) have raised concerns about the efficacy of this technique in

[4] https://zenodo.org/record/3559340

probing model representations. Specifically, they showed that non-syntactic models can produce qualitative patterns that appear to mimic the effects of syntactic priming. However, their results demonstrate that while the qualitative patterns may be similar, syntactic priming produces much larger effects in syntactic models compared with non-syntactic models. Therefore, their results should not be taken as an indictment of this methodology but simply that reasonable baselines must be employed when using this probing method.

7 Probing for Filler-Gaps

7.1 Null-Prime Baseline

Adaptation effects are difficult to interpret on their own. A positive effect indicates that priming produced more accurate predictions in a given class, while a negative effect indicates the opposite, but the magnitude of the effect is tricky to interpret. Following Prasad et al. (2019), for each analysis we define a class of interest and then compare within-class adaptation effects to cross-class adaptation effects. However, as Kodner and Gupta (2020) point out, spurious correlations may be introduced during stimulus creation/selection. Therefore, we also compare to null-primed models, which use the same original models but we prime them on prime sets whose sentences are shuffled at the word level (as Example (19-b)):

(19) a. ***What** does Sam wonder who solved __?
b. *Sam wonder What does solved who?

These shuffled null-prime stimuli do not contain any filler-gap dependencies, and so the adaptation effect from null-priming must represent phenomena in which we are not interested (e.g., lexical priming). Our effect-of-interest (filler-gap knowledge) is therefore reflected in any adaptation effect in excess of the null-prime effect.

7.2 Priming Filler-Gap Existence

First we ask whether the models have learned to represent the overall existence of a filler-gap dependency. To test this, we partition our stimuli into wholly acceptable constructions (involving object extraction, bridge verb extraction, and instances of left branch extraction and coordinate structure extraction) and wholly unacceptable constructions (the remaining constructions). We then test whether

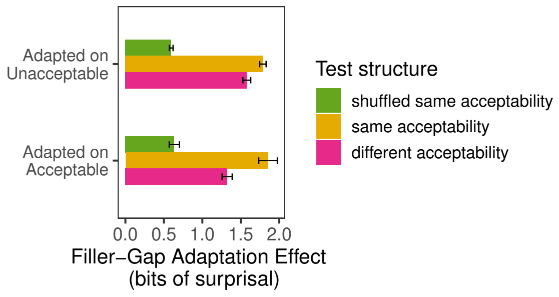

Figure 1: Adaptation effect produced by the existence of a filler-gap dependency. The green bars (top) represent priming a construction type from shuffled instances of similar acceptability. The gold bars (middle) represent the grammatical priming grammatical and ungrammatical priming ungrammatical effect. The pink bars (bottom) represent the acceptable priming unacceptable and vice versa effects.

grammatical constructions can prime ungrammatical constructions and vice versa.

Although at the sentence level, ungrammatical constructions do not have a resolvable filler-gap dependency, that unacceptability only manifests at or near the end of the sentence. From the perspective of our unidirectional models, both sets of sentences initially require the retention of an apparent “filler.” We therefore hypothesize that if the models understand filler-gap, both of these sets should initially contain a shared unidirectional representation of filler-gap existence.

We find that grammatical constructions prime ungrammatical constructions beyond the baseline shuffled adaptation effect (Figure 1). In other words, fine-tuning on grammatical items teaches the models how to process ungrammatical items with apparent filler-gap dependencies. We also find that ungrammatical items prime grammatical items in a similar fashion. Since the single unifying feature present in both grammatical and ungrammatical constructions is the presence of an apparent filler-gap dependency, these results suggest that the models contain a representation of filler-gap existence that is shared across constructions.

8 Probing for Filler-Gap Constraints

We next consider constraint-specific priming in order to determine whether the aforementioned abstraction of filler-gap dependency has shared substructure that would indicate understanding of the filler-gap constraints described in Section 4.

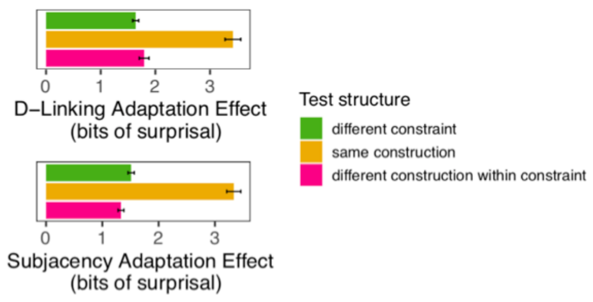

Figure 2: Adaptation effect due to D-linking adherence and Subjacency violation. Green bars (top) represent priming of a construction with constructions from other constraints (indicating general filler-gap priming). Gold bars (middle) represent priming of each construction with other constructions of the same type. Pink bars (bottom) represent priming of each construction with other constructions governed by the same constraint.

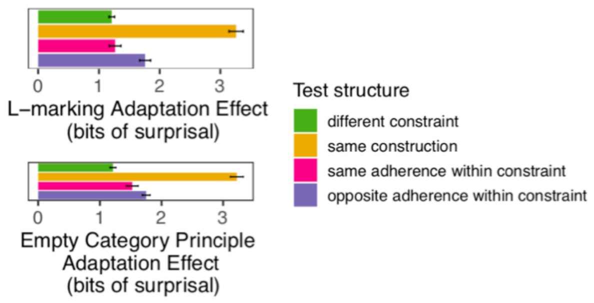

Figure 3: Adaptation effect due to L-marking and ECP. Some constructions adhered to these constraints and others violated them. Green bars (top) represent priming of a construction with constructions from other constraints (indicating general filler-gap priming). Gold bars (second from top) represent priming of each construction with other constructions of the same type. Pink bars (third from top) represent adherence priming adherence and violation priming violation. Purple bars (bottom) represent adherence priming violation and vice versa.

8.1 Filler-Gap Existence Baseline

In the previous section, we found that RNNs represent the existence of filler-gap dependencies similarly across construction types. We are therefore interested in whether the models systematically differentiate between filler-gap constructions that are governed by different constraints. If so, we would expect the shared representation of a constraint to produce greater priming within a constraint than across constraints. Therefore, rather than using shuffled sentences, our lower-bound baseline in this section consists of the adaptation effect from priming on sentences that do not share a constraint (green bars). We also use an upper-bound baseline adaptation effect from when models are tested and primed with the same syntactic construction (though the actual sentences differed; gold bars).

8.2 Priming Filler-Gap Constraints

Our results are presented in Figures 2 and 3. We divided our constructions based on those that consistently adhere to or violate particular constraints. If an abstract filler-gap constraint is learned by the model, then constraint adherence should prime adherence in the test set and constraint violation should prime violation in the test set (pink bars). Further, true understanding of a constraint would mean that constraint adherence would prime subsequent adherence more than subsequent violation and vice versa (purple bars reflect adherence priming violation and vice versa).

For subjacency, simple existence of filler-gap primed the model significantly more than other constructions involving subjacency.[5] Adherence to D-linking primed subsequent adherence to D-linking significantly more than simple filler-gap existence did, suggesting that perhaps the models do have an abstract representation of D-linking (but see Section 8.3).

Constructions involving L-marking and ECP produced significantly more priming in the mismatched adherence condition (adherence priming violation; violation priming adherence) than the matched adherence condition (adherence priming adherence; violation priming violation), suggesting that these constraints aren't learned by RNN language models. In fact, matched adherence in L-marking was not significantly different than priming with unrelated filler-gap sentences. Matched adherence in ECP did produce significantly greater priming than filler-gap existence, but since the priming effect was even greater for mismatched adherence, we can conclude that the model did not have an abstract representation of ECP that modulates filler-gap acceptability in a predictable way.

8.3 D-Linking Modulation

Our priming results in the previous subsection suggested that D-linked stimuli prime subsequent D-linking more than simple filler-gap existence does. However, D-linking also has an explicit surface

[5]Significance was determined by two-sample t-tests. See Supplementary Materials for details.

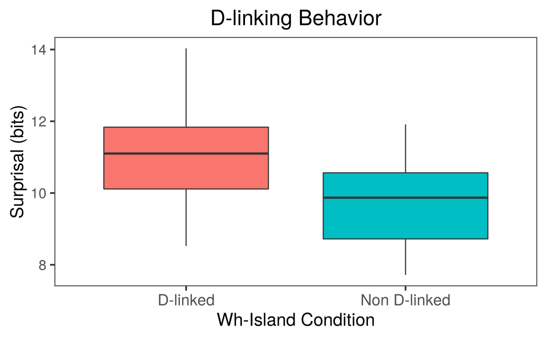

Figure 4: Pretrained model surprisal of D-linked wh-extractions (left) and non-D-linked wh-extractions (right). If the models learned to use D-linking to modulate the acceptability of filler-gap constructions, the surprisal of D-linked constructions should be lower.

cue, sentence-initial 'which', that could produce representational clustering even without an abstract linguistic concept of the constraint. Therefore, in this section, we use a behavioral probe to determine whether the models actually encode the underlying constraint.

As noted above, D-linking increases the acceptability of a filler-gap dependency.

(20) a. ??**Which book** did Will ask why John read __?

b. ***What** did Will ask why John read __?

At the filler, D-linking provides semantic clues to help predict the gap site (e.g., books will be bought or read but not eaten), and at the gap site D-linking greatly reduces the set of possible referents and eases retrieval of the correct referent both for syntactic attachment of the filler and for comprehension of the sentence (e.g., John did not read a sign or a scroll). We therefore expect that a model that understands D-linking will find D-linked sentences easier to process, similar to humans. We compared model performance (average surprisal; Equation 1) on each set of constructions. If a model uses D-linking in a human-like way, the D-linked sentences (like (20-a)) should produce better performance (be less surprising) than the non-D-linked sentences (like (20-b)).

In contrast to humans, RNNs find D-linked sentences more surprising than non-D-linked ones (Figure 4). In other words, the models prefer the less acceptable filler-gap constructions. These results suggest that the underlying D-linking feature is not learned even in a construction-specific way, let alone in a way that is shared across multiple constructions.

Based on the findings in the current and preceding sections, we conclude that RNN language models do not go beyond representing the existence of filler-gap dependencies to representing any of the shared underlying constraints we studied here.

9 Discussion

Our results support previous behavioral findings that recurrent neural language models acquire some abstract concept of "filler-gap dependency" (Chowdhury and Zamparelli, 2018; Wilcox et al., 2018, 2019), but go farther than those findings by indicating that this concept is representationally shared across different constructions. However, our findings also indicate that RNNs do not encode any of the syntactic filler-gap constraints we studied.

Both grammatical and ungrammatical sentences involving apparent filler-gap dependencies cause models to anticipate the existence of subsequent filler-gap dependencies. However, in two of four cases, the baseline adaptation effect from priming on unrelated filler-gap constructions was comparable to or greater than that from priming on other constructions governed by the same constraint. Constructions that violated ECP and L-marking were represented similarly to constructions that adhered to those same constraints. And models assigned lower probability to sentences involving D-linking than sentences without D-linking, which is the opposite of human results.

Overall, our results provide robust evidence that Chaves (2020) was correct that recurrent neural language models do not fully understand filler-gap constructions. While it is entirely possible that the constraints analyzed in this work do not actually govern filler-gap dependencies (i.e. these particular syntactic theories may be incorrect), we chose four constraints that have been very widely studied by syntacticians precisely because of their broad coverage. Therefore, even if the underlying syntactic theories are incorrect, it is conceivable that RNNs could induce these constraints as plausible abstractions to aid in filler-gap processing. It is therefore striking that RNNs learn none of them.

One might wonder whether our priming sets simply needed to be larger to observe the desired priming effects. Our priming sets consisted of 15 items, which is comparable to the number of priming stimuli used by Prasad et al. (2019) and Kodner and Gupta (2020). Since the constraints we study involve fewer surface cues, they could require more priming data to produce noticable effects. However, our non-baseline adaptation effects were around 1.5 bits, which is much larger than the 0.5-1 bit priming effects seen in those previous studies. Since we are already seeing large priming effects with these constructions, it seems unlikely that increasing the amount of priming data would produce qualitatively different effects.

We selected four common, well-studied filler-gap influences from the syntax literature and tested whether RNNs shared representational features across filler-gap constructions in a way that would suggest they had learned those constraints. While we did find evidence that RNNs encode some general representation of the existence of filler-gap dependencies, we found no evidence for more abstract underlying shared constraints. That is, while we find that RNN language models can learn abstract representations that are shared across constructions, our work raises questions about the depth of such abstractions.

Acknowledgements

We would like to thank Miloje Despić for feedback on the theoretical syntax discussion in this work. We would also like to thank the C.Psyd lab at Cornell for feedback on earlier versions of this work.

References

Emily Atkinson, Aaron Apple, Kyle Rawlins, and Akira Omaki. 2015. Similarity of wh-phrases and acceptability variation in wh-islands. *Frontiers in Psychology*, 6(2048).

Rui Chaves. 2020. What don't RNN language models learn about filler-gap dependencies? In *Proceedings of the Society for Computation in Linguistics (SCiL) 2020*, pages 20–30.

Noam Chomsky. 1973. Conditions on transformations. In S. Anderson and P. Kiparsky, editors, *A Festschrift for Morris Halle*, pages 232–286. Holt, Rinehart & Winston, New York.

Noam Chomsky. 1981. *Lectures on Binding and Government: The Pisa Lectures*. Foris Publications.

Noam Chomsky. 1986. *Barriers*. MIT Press.

Shammur Absar Chowdhury and Roberto Zamparelli. 2018. RNN simulations of grammaticality judgments on long-distance dependencies. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 133–144, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Forrest Davis and Marten van Schijndel. 2020. Recurrent neural network language models always learn English-like relative clause attachment. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1979–1990, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

John Hale. 2001. A probabilistic Earley parser as a psycholinguistic model. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

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

Richard Kayne. 1980. ECP extensions. *Linguistic Inquiry*.

Jordan Kodner and Nitish Gupta. 2020. Overestimation of syntactic representation in neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1757–1762, Online. Association for Computational Linguistics.

Michael Lepori, Tal Linzen, and R. Thomas McCoy. 2020. Representations of syntax [MASK] useful: Effects of constituency and dependency structure in recursive LSTMs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3306–3316, Online. Association for Computational Linguistics.

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.

Yingtong Liu, Rachel Ryskin, Richard Futrell, and Edward Gibson. 2019. Factive and manner-of-speaking islands are an artifact of nonlinearity in the acceptability judgment task. In *Proceedings of the 32nd Annual CUNY Conference on Human Sentence Processing (CUNY 2019)*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

David Pesetsky. 1987. Wh-in-situ: movement and unselective binding. *The representation of (in) definiteness*, pages 98–129.

Grusha Prasad, Marten van Schijndel, and Tal Linzen. 2019. Using priming to uncover the organization of syntactic representations in neural language models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 66–76, Hong Kong, China. Association for Computational Linguistics.

John Ross. 1967. *Constraints on variables in syntax*. Ph.D. thesis, Massachusetts Institute of Technology.

Marten van Schijndel and Tal Linzen. 2018. A neural model of adaptation in reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4704–4710, Brussels, Belgium. Association for Computational Linguistics.

Marten van Schijndel, Aaron Mueller, and Tal Linzen. 2019. Quantity doesn't buy quality syntax with neural language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5831–5837, Hong Kong, China. Association for Computational Linguistics.

Claude E. Shannon. 1948. The mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423.

Ethan Wilcox, Roger Levy, and Richard Futrell. 2019. What syntactic structures block dependencies in RNN language models? In *Proceedings of the 41st Annual Meeting of the Cognitive Science Society (CogSci)*, pages 1199–1205.

Ethan Wilcox, Roger Levy, Takashi Morita, and Richard Futrell. 2018. What do RNN language models learn about filler–gap dependencies? In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 211–221, Brussels, Belgium. Association for Computational Linguistics.