Overestimation of Syntactic Representation in Neural Language Models

Jordan Kodner
University of Pennsylvania
Dept. of Linguistics
jkodner@sas.upenn.edu

Nitish Gupta
University of Pennsylvania
Dept. of Computer and Information Science
nitishg@seas.upenn.edu

Abstract

With the advent of powerful neural language models over the last few years, research attention has increasingly focused on what aspects of language they represent that make them so successful. Several testing methodologies have been developed to probe models' syntactic representations. One popular method for determining a model's ability to induce syntactic structure trains a model on strings generated according to a template then tests the model's ability to distinguish such strings from superficially similar ones with different syntax. We illustrate a fundamental problem with this approach by reproducing positive results from a recent paper with two non-syntactic baseline language models: an n-gram model and an LSTM model trained on scrambled inputs.

1 Introduction

In recent years, RNN-based systems have proven excellent at a wide range of NLP tasks, sometimes achieving or even surpassing human performance on popular benchmarks. Their success stems from the complex but hard to interpret, representations that they learn from data. Given that syntax plays a critical role in human language competence, it is natural to ask whether part of what makes these models successful on language tasks is an ability to encode something akin to syntax.

This question pertains to syntax "in the meaningful sense," that is, the latent, hierarchical, largely context-free phrase structure underpinning human language as opposed to superficial or shallow issues of word order (Chomsky, 1957; Marcus, 1984; Everaert et al., 2015; Linzen et al., 2016). Clearly, syntactic information can be explicitly incorporated into neural systems to great effect (e.g., Dyer et al., 2016; Swayamdipta et al., 2018). Less certain is whether such systems induce something akin to hierarchical structure (henceforth, "syntax") on their own when not explicitly taught to do so.

Uncovering what an RNN actually represents is notoriously difficult, and several methods for probing RNNs' linguistic representations have been developed to approach the problem. Most directly, one can extract finite automata (e.g., Weiss et al., 2017) from the network or measure its state as it processes inputs to determine which neurons attend to what features (e.g., Shi et al., 2016; Linzen et al., 2016; Tenney et al., 2019). Alternatively, one can present a task which only a syntactic model should be able to solve, such as grammaticality discrimination or an agreement task, and then infer if a model has syntactic representations based on its behavior (Linzen et al., 2016; Ettinger et al., 2018; Gulordava et al., 2018; Warstadt et al., 2019).

In practice, simple sentences far outnumber the ones that require syntax in any natural corpus, which may obscure evaluation (Linzen et al., 2016). One way around this, referred to here as *template-based probing*, is to either automatically generate sentences with a particular structure or extract just the relevant ones from a much larger corpus. Templates have been used in a wide range of studies, including grammaticality prediction (e.g., Warstadt et al., 2019), long-distance dependency resolution, and agreement prediction tasks (e.g., Gulordava et al., 2018). By focusing on just relevant structures that match a given template rather than the gamut of naturally occurring sentence, template-based probing offers a controlled setting for evaluating specific aspects of a model's representation.

The crux of behavioral evaluation is the assertion that the chosen task effectively distinguishes between a model that forms syntactic representations and one which does not. This must be demonstrated for each task – if a model that does not capture syntax can pass the evaluation, then there is no conclusion to be drawn. However, this step is often omitted (but not always, e.g., Gulordava et al.,

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1757–1762
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

2018; Warstadt et al., 2019). Moreover, template-based generation removes the natural sparse and diverse distribution of sentence types, increasing the chance that a system might pick up on non-syntactic patterns in the data, further increasing the importance of a clear baseline.

This problem is most clearly illustrated with an example. In the following sections, we introduce Prasad et al.'s (2019) novel psycholinguistics-inspired template-based probe of relative clause types, which was taken as evidence in support of syntactic representation in LSTMs. We then pass PvSL's test with two non-syntactic baselines: an n-gram LM which can only capture short-distance word order of concrete types (Section 3), and an LSTM trained on scrambled inputs (Section 4). These baselines show that a combination of collocation and lexical representation can account for PvSL's results, which highlights a critical flaw in that experimental design. Following that, we argue that it is unlikely that LSTMs induce syntactic representations given current evidence and suggest an alternative angle for the question (Section 5).

2 Prasad, van Schijndel, & Linzen 2019

Prasad et al. (PvSL; 2019) leverage an analogy from psycholinguistic *syntactic priming* to test whether an LSTM is able to distinguish between sentences with different syntactic structures. When human subjects are *primed* by receiving an example of some input, their expectation of receiving similar subsequent input will temporarily increase relative to their expectation of other inputs. This can be used to test questions about syntax because once one is primed with sentences with a specific structure, subsequent sentences with shared structure will tend to show decreased surprisal responses relative to those with different structures.

PvSL observe that this procedure may be applied to neural networks as well. Since a model's surprisal upon receiving some input decreases as it receives subsequent similar inputs, one could cumulatively "prime" a model by adapting it toward a certain class of input (van Schijndel and Linzen, 2018). As the reasoning goes, if the model can be primed for a particular syntactic structure, that implies that it is able to recognize that structure and therefore has learned a representation for it.

This paradigm is used to assess an LSTM's ability to distinguish between five superficially similar but structurally distinct sentences types: those containing an unreduced object relative clause (RC), reduced object RC, unreduced passive subject RC, unreduced passive subject RC, and active subject RC, as well as two types matched for lexical content: passive subj./obj. RC-matched coordination sentences and active subj. RC-matched coordination. (1-2) present an example object RC and subject RC sentence to illustrate the structures.[1] These are distinguished syntactically by the origin of their subjects. In the first case, the subject of the sentence, 'the cake,' is also the object of the relative clause (position indicated by underscore), but in the second case, the sentence subject, 'the baker,' is also the subject of the relative clause.

(1) unreduced obj. RC: *The cake$_t$ [that the baker baked _$_t$] impressed the customers.*

(2) unreduced subj. RC: *The baker$_t$ [that _$_t$ baked the cake] impressed the customers.*

As PvSL note, if a model were able to track the position of the implicit syntactic origin, it would be able to distinguish these sentence types, so one would expect the model to exhibit a greater adaptation effect (greater decrease in surprisal) when primed and tested on the same sentence type than if primed on one type and tested on the other.

2.1 Main Experiment

PvSL populated templates to generate five sets of 20 adaptation and 50 test sentences for each sentence type with lexical items chosen to minimize lexical overlap between corresponding adaptation and test sets. Modifiers were optionally inserted in order to vary surface word order somewhat, and generated sentences were constrained to be *felicitous*, that is, they all made plausible semantic sense.

They trained 75 LSTM language models (van Schijndel and Linzen, 2018) on five splits of the WikiText-103 corpus. Average surprisal was computed for each model for each test set, then each model was adapted to ("primed for") each sentence type. They were then retested on the same test sets. The difference between pre- and post-adaptation surprisal ("adaptation effect") for each adaptation sentence type/test type pair was recorded, and adaptation effects were averaged across all models for each sentence type.

They establish a consistently and significantly stronger adaptation effect for same-type adaptation and test runs than different-type runs (PvSL

[1] More examples can be found in PvSL §4.1.

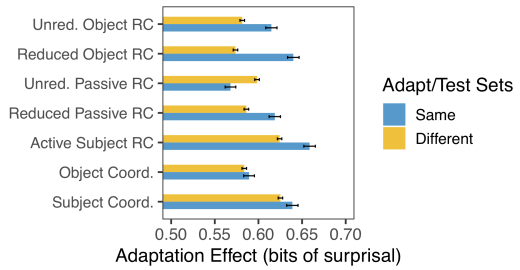

Figure 1: Average same-type vs. different-type adaptation effects for n-gram models. All differences are statistically significant except for object coordination.

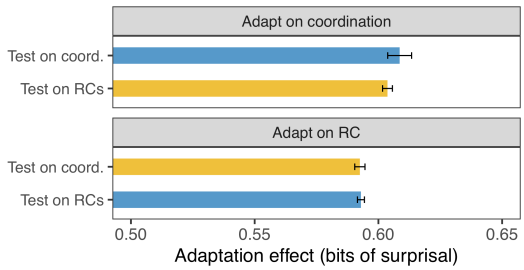

Figure 2: Average RC vs. coordination adaptation effects for n-gram models. Adapt on coord. is significant

§5.2), a stronger effect for RCs tested on models adapted for RCs rather than coordination sentences and vice-versa (PvSL §5.3), and for runs matched for passive voice over mismatched runs and for runs matched for reduction over mismatched runs (PvSL §5.4). Altogether, this is consistent with their hypothesis that the LSTM LMs are capturing abstract syntactic properties of their inputs.

Although the results are impressive, there are potential issues with their suggested interpretation. Namely, there may still be sufficient superficial word order information to achieve the effect despite the addition of optional modifiers (e.g., if unreduced object RCs often contain the bigram "that the," but unreduced subject RCs never do). Also, the felicity constraint means that the lexical items that appear in each sentence type should pattern together in the training data (i.e., verbs that are more likely to appear in object RCs are likely to pattern similarly in other constructions too). We test both possibilities in the following sections.

3 N-Gram Model

We begin by training an n-gram language model (through 4-grams) with Knesser-Ney smoothing (Ney et al., 1994) with the NLTK toolkit to determine whether it could be primed to distinguish PvSL's sentence types. An n-gram LM can only learn surface collocations and so cannot capture (hierarchical) syntax, so if it produces a significant differential adaptation effect, then the experiment is not able to discriminate between models which capture syntax from those which do not.

Adaptation and testing were carried out with PvSL's adaptation and test sets, and LM training was modified slightly to address n-gram models' characteristics. They have no recency bias, unlike RNNs, which diminishes the impact of adaptation. As such, 20 smaller models were trained on disjoint subsets of WikiText-2 rather than the full-sized WikiText-103 subsets.

Plotting and statistical analysis were carried out with PvSL's code[2]. Figure 1 shows the average adaptation effect observed when the models are adapted and tested on the same sentence type or different sentence types. Importantly, the same-type adaptation effect is greater than the different-type effect for six of seven sentence types (unreduced passive RC is reversed). Although the adaptation effect is uniformly weaker than observed for PvSL's LSTM LMs, there is a statistically significant difference between the same-type and different-type effects for six of seven sentence types.

Figure 2 compares the adaptation effect over RCs compared to coordination sentences. The n-gram models show a significantly greater same-type adaptation effect for coordination but not for RCs. A small but significant increase in voice- and reduction-matched adaptation over unmatched combinations was found (matched-passive matched reduction: 0.610, matched-passive mismatched-reduction: 0.594, mismatched-passive matched-reduction: 0.575, mismatched-passive mismatched-reduction: 0.572).

4 Scrambled-Input Model

Next, the same van Schijndel and Linzen (2018) trained LSTM LMs which PvSL employed were adapted on altered versions of their adaptation sets in which the word order of each sentence was scrambled to destroy the sentence's syntax while retaining its lexical content, then tested on the original non-scrambled test sets. Even though PvSL minimize the amount of lexical overlap in the adaptation and test sets, it may be the case that the models pick up on lexical similarities because of the felicity constraint which was imposed on them.

[2]https://github.com/grushaprasad/RNN-Priming, with minor aesthetic changes to plots

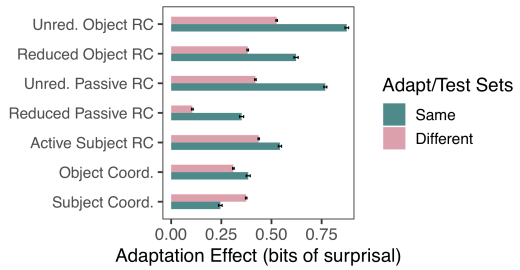

Figure 3: Average same-type vs. different-type adaptation effects for scrambled LSTM models. All differences are significant.

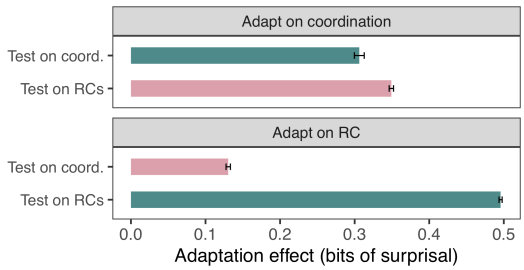

Figure 4: Average RC vs. coord. adaptation effects for scrambled LSTM models. Differences are significant.

Scrambling was random on a sentence-by-sentence basis. Results were averaged across all the adaptation sets and models (as they were in PvSL), so the effect of any individual accidentally grammatical scramble was diminished.

Figure 3 shows the average differential adaptation effects on these scrambled annotation runs. The same-type adaptation effect is significantly greater than different-type for six of seven sentence types (except subject coord.), and the largest relative difference is seen for unreduced passive RCs, the only type for which the n-gram models produced a reverse effect. Overall, the adaptation effect is an order of magnitude larger than for the n-gram models' but still smaller than PvSL's.

Figure 4 shows differential adaptation effects for RC and coordination sentences. A backward effect is observed for sentences adapted on coordination, but a large positive effect is found for those adapted on RC sentences. This is the complement of what was found for n-gram models. A significant positive difference was found between sentence types matched and unmatched in passives and reduction (matched-passive matched reduction: 0.65, matched-passive mismatched-reduction: 0.53, mismatched-passive matched-reduction: 0.53, mismatched-passive mismatched-reduction: 0.43).

5 Discussion

These results call into question the van Schijndel and Linzen (2018) and Prasad et al. (2019) syntactic priming paradigm's ability to distinguish models which represent syntax from those which rely on shallow phenomena by achieving a positive result with two non-syntactic baseline models. First, success in the priming paradigm is measured by whether or not adaptation reduces surprisal, but not by how much, so even though both baseline models tested here reduce surprisal by less than PvSL's models on average, they still pass the success criterion. To put it another way, PvSL report quantitative results but do not actually establish what would constitute a meaningful effect size. Even though the effect sizes of both our baseline replications were smaller, PvSL could have reported the results from our baseline models instead of their actual model and drawn the same conclusions.

Second, the fact that our surface word order n-gram model and lexical similarity-only scrambled LSTM LMs also show surprisal effects draws into question the basic claim that only a syntactic model would respond to adaptation: it is our hypothesis that the combined effect of word order and lexical similarity are what drive the LSTM models' larger effect. This is upheld, especially when it is noted that the adaptation effects of both baselines complement each other. Both alternative sources of information are well known in the community and have been tested in the past (Bernardy and Lappin, 2017; Gulordava et al., 2018). This reiterates the need for proper baseline testing in computational linguistics and for informative evaluations.

This highlights a more general problem with template-based probing, namely, that the unnatural lack of sentence diversity imposed by the templates imposes unintended regularity for models to latch onto. Given the well-known observation that neural models will "take the easy way out" given the presence of this unintended surface information (Jia and Liang, 2017; Naik et al., 2018; Sennhauser and Berwick, 2018), and other work suggesting that LSTMs do not necessarily induce syntactic structure (Gupta and Lewis, 2018; McCoy et al., 2018; Warstadt et al., 2019), one must take successes in template-based probing studies with a grain of salt. The evaluation of non-syntactic baselines is an easy-to-implement way to combat the tendency of these behavioral probes to overestimate language models' abilities.

To improve the priming paradigm in particular, one would need to establish a success metric that discriminates between baselines and alter the experimental setup to mitigate information side channels. One possibility would be to include infelicitous "colorless green ideas" sentences with grammatical syntax (cf. Gulordava et al., 2018), which might decrease the lexical similarity problem. Removing the issue altogether could require enforcing completely lexically disjoint training, adaptation, and test sets, but we cannot reasonably expect a model to function when it has no generalizations to work with, and demanding lexically distinct sets (including function words) greatly limits the set of phenomena that could be studied.

5.1 An Alternative Approach

As a more radical alternative, we suggest extending behavioral analysis into "consequence-based" analysis. The two have similar reasoning: from an engineering perspective, a family of models that is capable of inducing syntax is useful because it may be expected to improve performance on downstream tasks. Marcus (1984) discusses in a theory-independent way which kinds of sentences a model capturing syntax should be able to parse but a "no-explicit-syntax" model (in the modern context, probably a baseline RNN) should not (cf. Chomsky, 1957; Rimell et al., 2009; Nivre et al., 2010; Bender et al., 2011; Everaert et al., 2015). It follows then that no-explicit- and explicit-syntax models should exhibit quantitatively different behavior on tasks that require parsing such sentences. A model that solves problems that *only* one capable of inducing syntactic structure can solve may as well have induced syntactic structure from a practical standpoint.

Consequence-based analysis would be implemented over naturalistic data rather than templates by embedding it in higher level tasks like question answering to mitigate the unnaturalness problem and demonstrate a model's practical utility. The possibility of side-channel information is already known in relation to these higher-level tasks (e.g., Poliak et al., 2018; Geva et al., 2019), and various challenge data sets have been constructed to mitigate it in different ways (Levesque et al., 2011; Chao et al., 2017; Dua et al., 2019; Lin et al., 2019; Dasigi et al., 2019). Uniting these with a collection of hard sentence types (e.g., Marvin and Linzen, 2018; Warstadt et al., 2019) in something like a syntax-focused QA challenge set would provide new insights into which families of models capture the practical benefits of true hierarchical syntactic representation.

Acknowledgments

We are particularly grateful to Marten van Schijndel for sharing the van Schijndel and Linzen (2018) model checkpoints with us. We also thank Mitch Marcus, Charles Yang, and Ryan Budnick for their comments and suggestions. This work was funded by an NDSEG fellowship awarded to the first author by the ARO, in addition to funding by the ONR under Contract No. N00014-19-1-2620, and by sponsorship from the LwLL DARPA program under Contract No. FA8750-19-2-0201. (The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.)

References

Emily M Bender, Dan Flickinger, Stephan Oepen, and Yi Zhang. 2011. Parser evaluation over local and non-local deep dependencies in a large corpus. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 397–408. Association for Computational Linguistics.

Jean-Philippe Bernardy and Shalom Lappin. 2017. Using deep neural networks to learn syntactic agreement. *LiLT (Linguistic Issues in Language Technology)*, 15.

Wei-Lun Chao, Hexiang Hu, and Fei Sha. 2017. Being negative but constructively: Lessons learnt from creating better visual question answering datasets. *arXiv preprint arXiv:1704.07121*.

Noam Chomsky. 1957. *Syntactic Structures*. Moulton & Co.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *EMNLP/IJCNLP*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *NAACL-HLT*.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209.

Allyson Ettinger, Ahmed Elgohary, Colin Phillips, and Philip Resnik. 2018. Assessing composition in sentence vector representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1790–1801.

M.B.H. Everaert, Marinus Huybregts, Noam Chomsky, Robert Berwick, and Johan Bolhuis. 2015. Structures, not strings: Linguistics as part of the cognitive sciences. *Trends in Cognitive Sciences*, xx.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *EMNLP/IJCNLP*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205.

Nitish Gupta and Mike Lewis. 2018. Neural compositional denotational semantics for question answering. In *EMNLP/IJCNLP*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *EMNLP/IJCNLP*.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2011. The winograd schema challenge. In *KR*.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning over paragraph effects in situations. In *MRQA*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *TACL*.

Mitchell P Marcus. 1984. Some inadequate theories of human language processing. In Carroll J. Bever, T. and L. Miller, editors, *Talking Minds: The Study of Language in the Cognitive Sciences*, chapter 9, pages 253–279. MIT Press, Cambridge, MA.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. *arXiv preprint arXiv:1808.09031*.

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. *arXiv preprint arXiv:1802.09091*.

Aakanksha Naik, Abhilasha Ravichander, Norman M. Sadeh, Carolyn Penstein Rosé, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *COLING*.

Hermann Ney, Ute Essen, and Reinhard Kneser. 1994. On structuring probabilistic dependences in stochastic language modelling. *Computer Speech & Language*.

Joakim Nivre, Laura Rimell, Ryan McDonald, and Carlos Gomez-Rodriguez. 2010. Evaluation of dependency parsers on unbounded dependencies. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 833–841. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In **SEM@NAACL-HLT*.

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.

Laura Rimell, Stephen Clark, and Mark Steedman. 2009. Unbounded dependency recovery for parser evaluation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 813–821. Association for Computational Linguistics.

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.

Luzi Sennhauser and Robert Berwick. 2018. Evaluating the ability of lstms to learn context-free grammars. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 115–124.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural mt learn source syntax? In *EMNLP/IJCNLP*, pages 1526–1534.

Swabha Swayamdipta, Sam Thomson, Kenton Lee, Luke Zettlemoyer, Chris Dyer, and Noah A Smith. 2018. Syntactic scaffolds for semantic structures. In *EMNLP/IJCNLP*, pages 3772–3782.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *ACL*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2019. BLiMP: A benchmark of linguistic minimal pairs for English.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2017. Extracting automata from recurrent neural networks using queries and counterexamples. *arXiv preprint arXiv:1711.09576*.