Causal Transformers Perform Below Chance on Recursive Nested Constructions, Unlike Humans

Anonymous ACL submission

Abstract

Recursive processing is considered a hallmark of human linguistic abilities. A recent study evaluated recursive processing in recurrent neural language models (RNN-LMs) and showed that such models perform below chance level on embedded dependencies within nested constructions – a prototypical example of recursion in natural language. Here, we study if state-of-the-art Transformer LMs do any better. We test four different Transformer LMs on two different types of nested constructions, which differ in whether the embedded (inner) dependency is short or long range. We find that Transformers achieve near-perfect performance on short-range embedded dependencies, significantly better than previous results reported for RNN-LMs and humans. However, on *long-range* embedded dependencies, Transformers' performance sharply drops below chance level. Remarkably, the addition of only three words to the embedded dependency caused Transformers to fall from near-perfect to below-chance performance. Taken together, our results reveal Transformers' shortcoming when it comes to recursive, structure-based, processing.

1 introduction

One of the fundamental principles of contemporary linguistics states that language processing requires the ability to deal with nested structures. Recursion, a specific type of computation that involves repeatedly applying a function to its own output, is suggested to be at the core of this ability (Hauser et al., 2002). The strongest evidence for recursion in human language processing arises from the tree-like nested structure of sentences in natural language, in which phrases of a particular type (i.e. NPs) can be embedded in other phrases of that same type (Figure 1). Humans, it is argued, are endowed with a unique competence for recursive processing, which allows them to represent and process such nested tree structures (Chomsky, 2000; Hauser et al., 2002; Dehaene et al., 2015).

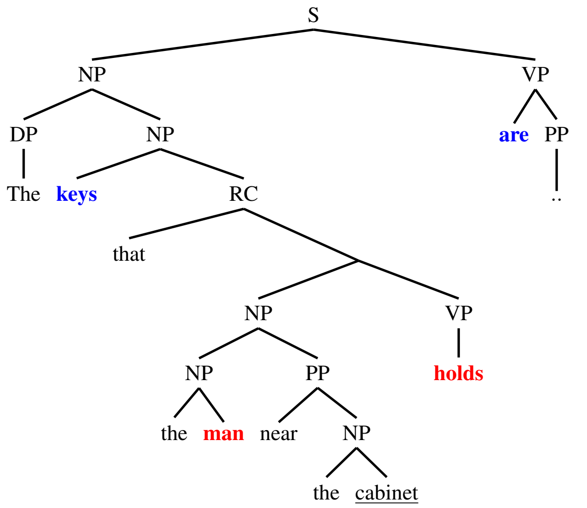


Figure 1: A tree-structure representation of a recursive structure with two long-range dependencies, one nested within the other one.

In recent years, neural language models (NLMs) have shown tremendous advances on a variety of linguistic tasks, such as next-word prediction, translation or semantic inference. Furthermore, evaluations of their syntactic abilities have shown promising results, with similar or even above-human performance on a variety of different tasks (Marvin and Linzen, 2018; Goldberg, 2019; Jumelet et al., 2021; Giulianelli et al., 2018)). However, negative results were recently also presented (Warstadt et al., 2020; Hu et al., 2020). In particular, when it comes to recursive processing, Lakretz et al. (2021b) showed that while recurrent neural network language models (RNN-LMs) perform well on long-range dependencies, such as the relationship between **keys** and **are** in sentences like "The **keys** that the *man* near the cabinet *holds*, **are** red" (Figure 2), they perform below chance on the shorter, embedded dependency (*man-holds*). Humans, instead, perform significantly better on such dependencies, although interestingly, for them too, the shorter inner dependency is more difficult than the long outer one.

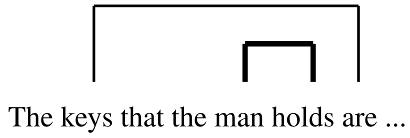
The study by Lakretz et al. illustrates how investigations of neural networks can inspire experiments about human language processing. However, their study focuses on only a single architecture, an RNN-LM with LSTM units (Hochreiter and Schmidhuber, 1997), which is currently outperformed on many fronts by the newer *Transformer* models (Vaswani et al., 2017). In this short paper, our main question is therefore whether Transformer models do any better when it comes to processing recursive constructions. We then further explore similarities and differences in performance patterns of RNN and Trasformer language models.

Our main results show that when tested on nested constructions with a short-range embedded dependency, Transformers outperform RNN-LM across all conditions, with error rates close to zero. However, when the embedded dependency is long-range, their performance dramatically drops to below chance, similarly to the case of RNNs. The mere addition of a short prepositional phrase ('near the cabinet' in the example shown in Figure 1) to the embedded dependency causes model performance to drop from near perfect to below chance level. Thus, contrary to what might be expected based on their much improved performance and the fact that they are trained on substantially more data, Transformer models share RNNs' shortcoming when it comes to recursive, structure-sensitive, processing.

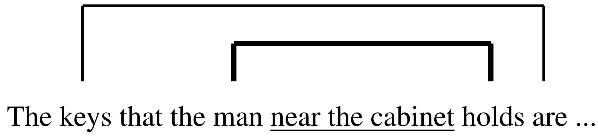
Last, all models made more errors when trying to carry a noun in the singular across dependencies which involved a plural noun, than in the converse situation. Interestingly, this bias towards greater interference by plural than by singular is opposite to that reported in Italian RNN-LMs (Lakretz et al., 2021b), and is akin to the Markedness Effect reported for humans.

2 Related Work

In psycholinguistics, grammatical agreement became a standard method to probe online syntactic processing in humans (Bock and Miller, 1991; Franck et al., 2002), since it is ruled by hierarchical structures rather than by the linear order of words in a sentence. More recently, it has also become a standard way to probe grammatical generalization in NLMs (Linzen et al., 2016; Bernardy and Lappin, 2017; Giulianelli et al., 2018; Gulordava et al., 2018; Jumelet et al., 2019; Kersten et al., 2021; Lakretz et al., 2019; Sinha et al., 2021), pointing to both similarities and differences between human and model error patterns.



(a) Short-Nested

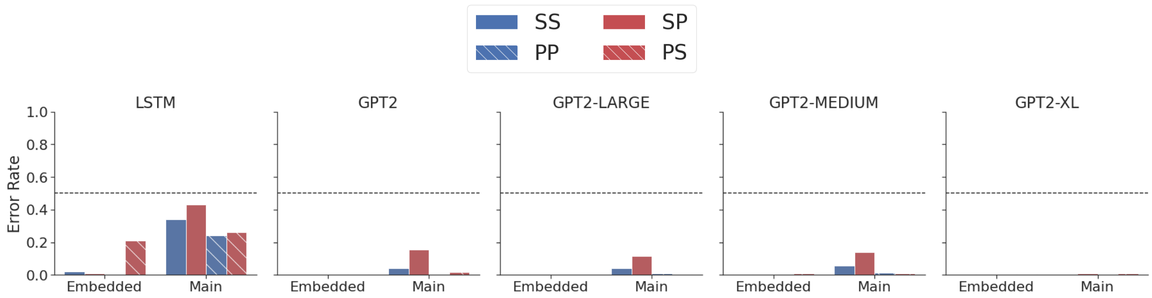


(b) Long-Nested

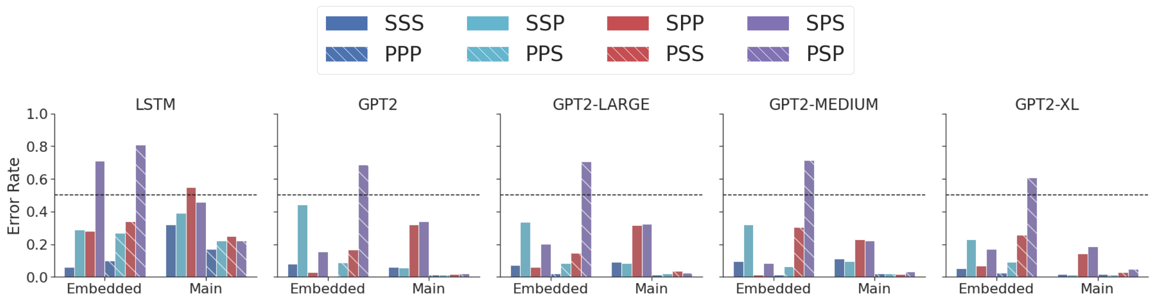
Figure 2: Experimental Design: the two number-agreement tasks – *Short-Nested* and *Long-Nested*. In Short-Nested, the embedded dependency is short-range (in bold); in Long-Nested, it is long-range, through the insertion of a three-word prepositional phrase.

Lakretz et al. (2019) showed that RNN-LMs trained on a large corpus with English sentences develop a number-propagation mechanism for long-range dependencies. The core circuit of this mechanism was found to be extremely sparse, comprising of only a very small number of units. This sparsity of the mechanism suggests that models are not able to process two long-distance dependencies simultaneously, and indeed, this was later confirmed in simulations (Lakretz et al., 2021b). Inspired by this finding, Lakretz et al. (2021b) conducted a following experiment with humans, which showed that they, too, make more errors on nested long-range dependencies. However, contrary to LMs, their performance was above chance on these constructions. This finding suggests that human recursive processing remains significantly better than that of RNN-LMs.

Recursive processing of nested constructions in RNN-LMs was also studied using artificial grammars (Cleeremans et al., 1989; Servan-Schreiber et al., 1991; Gers and Schmidhuber, 2001; Christiansen and Chater, 1999; Hewitt et al., 2020). Recently, Suzgun et al. (2019) showed that memory-augmented RNNs can capture recursive regularities of Dyck languages (also known as "bracket languages"). However, when tested on a simple extension of these languages, RNN-LMs failed to generalize to unseen data with a greater nest-



(a) Short-Nested.



(b) Long-Nested.

Figure 3: Error rates on nested constructions for all models, for both the main and embedded agreements. Conditions are marked by the value of the grammatical number of all nouns in the sentence. For example, condition SP means that the first noun is singular and the second is plural. While error-rates are near zero for Short-Nested, they are worse than chance-level for one of the incongruent conditions of Long-Nested, consistently across all models. In this condition (PSP), grammatical agreement is with respect to the second noun, which is singular.

ing depth (Lakretz et al., 2021a). Specifically, the models failed also in cases in which the training data contained deep structures, up to five levels of nesting. This suggests that the poor recursive processing of RNN-LMs is not merely due to shallow nesting depth in natural data, which is typically not more than two (Karlsson, 2007).

Taken together, previous work suggests that RNN-LMs struggle to capture recursive regularities in either natural or artificial data. Inspired by this line of work, we focus here on Transformer LMs: do they show different patterns when it comes to processing recursive structures? Do they better approximate human ability for recursion?

3 Experimental Setup

We largely follow the experimental setup of Lakretz et al. (2021b), but consider a different language (English instead of Italian) and a different set of models.

Data We consider two number-agreement tasks (*NA-tasks*): *Short-Nested* and *Long-Nested*. Both tasks contain two subject-verb dependencies; they differ in terms of whether the embedded dependency is *short-* or *long-range*. In *short-nested*, the subject and verb in the nested dependency are adjacent (Figure 2a). They are embedded in a sentence by inserting an object-relative clause to modify the subject of a different sentence. The *Long-Nested* task (Figure 2b) uses the same constructions, except that an additional three-word prepositional phrase ("near the cabinet") is added in the embedded dependency.[1]

Models We run experiments with all causal transformer-based NLMs that are currently compatible with the BigBench framework, available from HuggingFace.[2] Specifically, we include four GPT-2 models that differed in size: GPT2, GPT2-Medium, GPT2-Large and GPT-XL (Radford et al., 2019). In addition, as a baseline, we conduct an experiment with an English LSTM-LM, which was studied in numerous work in the past (Gulordava et al., 2018).

[1] All data sets are available in the BigBench collaborative benchmark https://github.com/google/BIG-bench/tree/main/bigbench/benchmark_tasks/subject_verb_agreement

[2] https://huggingface.co/transformers/

Model evaluation Following previous work, we evaluated model performance on agreement by comparing the output probabilities for the correct (e.g., 'are') vs. wrong ('is') verb form. For both tasks, we evaluated model performance on agreement for both the embedded and the inner verb, and separately for each task condition (see SM).

4 Results

4.1 Short-Nested task

In Figure 3a, we show model performance on the Short-Nested task for all models. Overall, the English LSTM made more errors on the main (outer) dependency compared to the embedded (inner) one, with more than 20% errors, across all four conditions. In contrast, Transformers, and in particular GPT2-XL, achieved close to perfect performance across all conditions, on both the embedded and main dependency. For GPT2, GPT2-Medium and Large, the longer main dependency was, however, overall more difficult than the embedded one, but with no more than 20% errors in the incongruent conditions (SP and PS; Table S2).

Interestingly, consistently across all models, both Transformers and the LSTM model made more errors on conditions in which the agreement was with respect to singular, compared to plural.

4.2 Long-Nested task

In Figure 3b, we further show the performance of all models for the Long-Nested task. Overall, all models made more errors across all conditions compared to Short-Nested, but with the same tendency of making more errors on dependencies with respect to singular compared to plural. The most striking difference between the two tasks was the performance of the models on the embedded dependency. In particular, for Transformers, their error rate was close to zero in Short-Nested, but dropped to below-chance on one of the incongruent conditions (PSP) in Long-Nested. Similarly, For the LSTM, this was the case for both incongruent cases (PSP and SPS).

In contrast to the embedded dependency, all models performed above chance on the main, longer, dependency. This shows that for Long-Nested, the length of the dependency affected model performance less than the presence of recursive embedding.

5 Discussion

In this study, we evaluated the recursive abilities of Transformer LMs on two number-agreement tasks that were previously shown to be exceptionally challenging for LSTM language models. Our experiments showed that, overall, Transformers outperformed LSTM-LMs, and in particular, achieved close to perfect performance on short embedded dependencies. However, similarly to LSTM-LMs, the addition of only a short prepositional phrase to the embedded dependency caused model performance to sharply drop to below chance level.

Furthermore, we found that all models showed a bias towards plural and therefore err more when the subject of a verb is in the singular. A similar bias was previously observed in Italian LSTM models (Lakretz et al., 2021b), however, in the opposite direction, with more errors on plural dependencies. We hypothesize that this difference might be due to marking of the verb form, given that in English, the marked form of the verb is singular, whereas in Italian, it is plural. Related biases were previous reported for humans in both languages, a phenomenon known as the Markedness Effect (Bock and Miller, 1991; Vigliocco et al., 1995). The relation between emerging biases in NLMs and humans is an interesting topic for future work.

In LSTM-LMs, the poor performance was predicted by the underlying neural mechanism for grammatical agreement identified in the models (Lakretz et al., 2019, 2021b). The fact that Transformer models perform similarly poorly on these constructions, and on the same dependency (inner), raises interesting questions. Do transformers use syntactic-processing strategies akin to those emerged in RNN-LMs? And what does that tell us about the data that those models are trained on and about the potential processes that humans may use to process such constructions (Lakretz et al., 2020)?

However, currently, the neural mechanisms underlying syntactic processing in transformers are poorly understood (Belinkov and Glass, 2019). Our findings of below-chance performance by transformer models calls for a further investigation in *how* these models achieve their earlier found successes on syntactic related tasks, and why they generalise so poorly on constructions which only minimally differ (a single three-word prepositional phrase) from the constructions they process well.

References

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Jean-Philippe Bernardy and Shalom Lappin. 2017. Using deep neural networks to learn syntactic agreement. *Linguistic Issues in Language Technology*, 15(2):1–15.

Kathryn Bock and Carol Miller. 1991. Broken agreement. *Cognitive Psychology*, 23(1):45–93.

Noam Chomsky. 2000. Minimalist inquiries: The framework. In Roger Martin, David Michaels, Juan Uriagereka, and Samuel Keyser, editors, *Step by step: Essays on minimalist syntax in honor of Howard Lasnik*, pages 89–155. MIT Press, Cambridge, MA.

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

Axel Cleeremans, David Servan-Schreiber, and James L McClelland. 1989. Finite state automata and simple recurrent networks. *Neural computation*, 1(3):372–381.

Stanislas Dehaene, Florent Meyniel, Catherine Wacongne, Liping Wang, and Christophe Pallier. 2015. The neural representation of sequences: From transition probabilities to algebraic patterns and linguistic trees. *Neuron*, 88(1):2–19.

Julie Franck, Gabriella Vigliocco, and Janet Nicol. 2002. Subject-verb agreement errors in french and english: The role of syntactic hierarchy. *Language and cognitive processes*, 17(4):371–404.

Felix Gers and Jürgen Schmidhuber. 2001. LSTM recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, 12(6):1333–1340.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. 2018. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 240–248, Brussels, Belgium. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of NAACL*, pages 1195–1205, New Orleans, LA.

Marc Hauser, Noam Chomsky, and Tecumseh Fitch. 2002. The faculty of language: What is it, who has it, and how did it evolve? *Science*, 298(5598):1569–1579.

John Hewitt, Michael Hahn, Surya Ganguli, Percy Liang, and Christopher D Manning. 2020. Rnns can generate bounded hierarchical languages with optimal memory. *arXiv preprint arXiv:2010.07515*.

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

Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger P Levy. 2020. A systematic assessment of syntactic generalization in neural language models. *arXiv preprint arXiv:2005.03692*.

Jaap Jumelet, Milica Denic, Jakub Szymanik, Dieuwke Hupkes, and Shane Steinert-Threlkeld. 2021. Language models use monotonicity to assess NPI licensing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4958–4969, Online. Association for Computational Linguistics.

Jaap Jumelet, Willem Zuidema, and Dieuwke Hupkes. 2019. Analysing neural language models: Contextual decomposition reveals default reasoning in number and gender assignment. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 1–11, Hong Kong, China. Association for Computational Linguistics.

Fred Karlsson. 2007. Constraints on multiple center-embedding of clauses. *Journal of Linguistics*, 43(2):365–392.

Tom Kersten, Hugh Mee Wong, Jaap Jumelet, and Dieuwke Hupkes. 2021. Attention vs non-attention for a shapley-based explanation method. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 129–139, Online. Association for Computational Linguistics.

Yair Lakretz, Stanislas Dehaene, and Jean-Rémi King. 2020. What limits our capacity to process nested long-range dependencies in sentence comprehension? *Entropy*, 22(4):446.

Yair Lakretz, Théo Desbordes, Jean-Rémi King, Benoît Crabbé, Maxime Oquab, and Stanislas Dehaene. 2021a. Can rnns learn recursive nested subject-verb agreements? *arXiv preprint arXiv:2101.02258*.

Yair Lakretz, Dieuwke Hupkes, Alessandra Vergallito, Marco Marelli, Marco Baroni, and Stanislas Dehaene. 2021b. Mechanisms for handling nested dependencies in neural-network language models and humans. *Cognition*, page 104699.

Yair Lakretz, Germán Kruszewski, Theo Desbordes, Dieuwke Hupkes, Stanislas Dehaene, and Marco Baroni. 2019. The emergence of number and syntax units in LSTM language models. In *Proceedings of NAACL*, pages 11–20, Minneapolis, MN.

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.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

David Servan-Schreiber, Axel Cleeremans, and James L McClelland. 1991. Graded state machines: The representation of temporal contingencies in simple recurrent networks. *Machine Learning*, 7(2-3):161–193.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. *CoRR*, abs/2104.06644.

Mirac Suzgun, Sebastian Gehrmann, Yonatan Belinkov, and Stuart M Shieber. 2019. Memory-augmented recurrent neural networks can learn generalized dyck languages. *arXiv preprint arXiv:1911.03329*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Gabriella Vigliocco, Brian Butterworth, and Carlo Semenza. 1995. Constructing subject-verb agreement in speech: The role of semantic and morphological factors. *Journal of Memory and Language*, 34(2):186–215.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: A benchmark of linguistic minimal pairs for English. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 409–410, New York, New York. Association for Computational Linguistics.

Supplementary Materials

1 Number-Agreement Tasks

		Number-Agreement tasks
Short-Nested		$\mathtt{NP}_a$ $\mathtt{that}$ $\mathtt{NP}_b$ $\mathtt{V}_b$ $\mathtt{V}_a$
	SS	The **key** that the *man holds* **is**
	SP	The **key** that the *men hold* **is**
	PS	The **keys** that the *men holds* **are**
	PP	The **keys** that the *man holds* **are**
Long-Nested		$\mathtt{NP}_a$ $\mathtt{that}$ $\mathtt{NP}_b$ $\mathtt{P}$ $\mathtt{NP}_c$ $\mathtt{V}_b$ $\mathtt{V}_a$
	SSS	The **key** that the *man* near the cabinet *holds* **is**
	SSP	The **key** that the *man* near the cabinets *holds* **is**
	SPS	The **key** that the *men* near the cabinet *hold* **is**
	SPP	The **key** that the *men* near the cabinets *hold* **is**
	PSS	The **keys** that the *man* near the cabinet *holds* **are**
	PSP	The **keys** that the *man* near the cabinets *holds* **are**
	PPS	The **keys** that the *men* near the cabinet *hold* **are**
	PPP	The **keys** that the *men* near the cabinet *hold* **are**

S 1: **The Short- and Long-Nested Number-Agreement tasks.** The first column denotes the name of the task, the second shows the conditions for each task, the third shows the sentence template, where `NP` is used as an abbreviation of `Det N`. The indices a, b mark the subject-verb dependencies in the templates. For example, in *Long-Nested*, there are three nouns and two verbs, the indices a and b indicate that the last verb $\mathtt{V}_a$ is syntactically dependent on the first noun phrase $\mathtt{NP}_a$, whereas the penultimate verb $\mathtt{V}_b$ instead should match the features of the second noun phrase $\mathtt{NP}_b$. Below each template, and example for each condition is given. Bold and italic face highlight the dependencies marked by the indices in the templates. For each agreement task, we systematically vary the *number* of all nouns in the template, resulting in four different conditions (SS, SP, PS and PP) for the *Short-Nested* and eight different conditions (SSS, SSP, SPS, SPP, PSS, PSP, PPS and PPP) for Long-Nested.

2 Detailed Results for all Models

Model	NA-Task	Dependency	Condition	Error Rate
LSTM	Short-Nested	Main	SS	0.34
LSTM	Short-Nested	Main	SP	0.43
LSTM	Short-Nested	Main	PS	0.26
LSTM	Short-Nested	Main	PP	0.24
LSTM	Short-Nested	Embedded	SS	0.02
LSTM	Short-Nested	Embedded	SP	0.01
LSTM	Short-Nested	Embedded	PS	0.21
LSTM	Short-Nested	Embedded	PP	0.00
LSTM	Long-Nested	Main	SSS	0.32
LSTM	Long-Nested	Main	SSP	0.39
LSTM	Long-Nested	Main	SPS	0.46
LSTM	Long-Nested	Main	SPP	0.55
LSTM	Long-Nested	Main	PSS	0.25
LSTM	Long-Nested	Main	PSP	0.22
LSTM	Long-Nested	Main	PPS	0.22
LSTM	Long-Nested	Main	PPP	0.17
LSTM	Long-Nested	Embedded	SSS	0.06
LSTM	Long-Nested	Embedded	SSP	0.29
LSTM	Long-Nested	Embedded	SPS	0.71
LSTM	Long-Nested	Embedded	SPP	0.28
LSTM	Long-Nested	Embedded	PSS	0.34
LSTM	Long-Nested	Embedded	PSP	0.81
LSTM	Long-Nested	Embedded	PPS	0.27
LSTM	Long-Nested	Embedded	PPP	0.10

Model	NA-Task	Dependency	Condition	Error Rate
GPT2	Short-Nested	Main	SS	0.04
GPT2	Short-Nested	Main	SP	0.16
GPT2	Short-Nested	Main	PS	0.02
GPT2	Short-Nested	Main	PP	0.00
GPT2	Short-Nested	Embedded	SS	0.00
GPT2	Short-Nested	Embedded	SP	0.00
GPT2	Short-Nested	Embedded	PS	0.00
GPT2	Short-Nested	Embedded	PP	0.00
GPT2	Long-Nested	Main	SSS	0.06
GPT2	Long-Nested	Main	SSP	0.05
GPT2	Long-Nested	Main	SPS	0.34
GPT2	Long-Nested	Main	SPP	0.32
GPT2	Long-Nested	Main	PSS	0.02
GPT2	Long-Nested	Main	PSP	0.02
GPT2	Long-Nested	Main	PPS	0.01
GPT2	Long-Nested	Main	PPP	0.01
GPT2	Long-Nested	Embedded	SSS	0.08
GPT2	Long-Nested	Embedded	SSP	0.44
GPT2	Long-Nested	Embedded	SPS	0.16
GPT2	Long-Nested	Embedded	SPP	0.03
GPT2	Long-Nested	Embedded	PSS	0.17
GPT2	Long-Nested	Embedded	PSP	0.69
GPT2	Long-Nested	Embedded	PPS	0.09
GPT2	Long-Nested	Embedded	PPP	0.00

Model	NA-Task	Dependency	Condition	Error Rate
GPT2-MEDIUM	Short-Nested	Main	SS	0.06
GPT2-MEDIUM	Short-Nested	Main	SP	0.14
GPT2-MEDIUM	Short-Nested	Main	PS	0.01
GPT2-MEDIUM	Short-Nested	Main	PP	0.01
GPT2-MEDIUM	Short-Nested	Embedded	SS	0.01
GPT2-MEDIUM	Short-Nested	Embedded	SP	0.00
GPT2-MEDIUM	Short-Nested	Embedded	PS	0.01
GPT2-MEDIUM	Short-Nested	Embedded	PP	0.00
GPT2-MEDIUM	Long-Nested	Main	SSS	0.11
GPT2-MEDIUM	Long-Nested	Main	SSP	0.10
GPT2-MEDIUM	Long-Nested	Main	SPS	0.22
GPT2-MEDIUM	Long-Nested	Main	SPP	0.23
GPT2-MEDIUM	Long-Nested	Main	PSS	0.02
GPT2-MEDIUM	Long-Nested	Main	PSP	0.03
GPT2-MEDIUM	Long-Nested	Main	PPS	0.02
GPT2-MEDIUM	Long-Nested	Main	PPP	0.02
GPT2-MEDIUM	Long-Nested	Embedded	SSS	0.10
GPT2-MEDIUM	Long-Nested	Embedded	SSP	0.32
GPT2-MEDIUM	Long-Nested	Embedded	SPS	0.08
GPT2-MEDIUM	Long-Nested	Embedded	SPP	0.01
GPT2-MEDIUM	Long-Nested	Embedded	PSS	0.30
GPT2-MEDIUM	Long-Nested	Embedded	PSP	0.71
GPT2-MEDIUM	Long-Nested	Embedded	PPS	0.06
GPT2-MEDIUM	Long-Nested	Embedded	PPP	0.01

Model	NA-Task	Dependency	Condition	Error Rate
GPT2-LARGE	Short-Nested	Main	SS	0.04
GPT2-LARGE	Short-Nested	Main	SP	0.11
GPT2-LARGE	Short-Nested	Main	PS	0.01
GPT2-LARGE	Short-Nested	Main	PP	0.01
GPT2-LARGE	Short-Nested	Embedded	SS	0.00
GPT2-LARGE	Short-Nested	Embedded	SP	0.01
GPT2-LARGE	Short-Nested	Embedded	PS	0.00
GPT2-LARGE	Short-Nested	Embedded	PP	0.00
GPT2-LARGE	Long-Nested	Main	SSS	0.09
GPT2-LARGE	Long-Nested	Main	SSP	0.09
GPT2-LARGE	Long-Nested	Main	SPS	0.32
GPT2-LARGE	Long-Nested	Main	SPP	0.32
GPT2-LARGE	Long-Nested	Main	PSS	0.04
GPT2-LARGE	Long-Nested	Main	PSP	0.03
GPT2-LARGE	Long-Nested	Main	PPS	0.02
GPT2-LARGE	Long-Nested	Main	PPP	0.01
GPT2-LARGE	Long-Nested	Embedded	SSS	0.07
GPT2-LARGE	Long-Nested	Embedded	SSP	0.34
GPT2-LARGE	Long-Nested	Embedded	SPS	0.20
GPT2-LARGE	Long-Nested	Embedded	SPP	0.06
GPT2-LARGE	Long-Nested	Embedded	PSS	0.15
GPT2-LARGE	Long-Nested	Embedded	PSP	0.71
GPT2-LARGE	Long-Nested	Embedded	PPS	0.08
GPT2-LARGE	Long-Nested	Embedded	PPP	0.02

Model	NA-Task	Dependency	Condition	Error Rate
GPT2-XL	Short-Nested	Main	SS	0.00
GPT2-XL	Short-Nested	Main	SP	0.01
GPT2-XL	Short-Nested	Main	PS	0.01
GPT2-XL	Short-Nested	Main	PP	0.01
GPT2-XL	Short-Nested	Embedded	SS	0.00
GPT2-XL	Short-Nested	Embedded	SP	0.01
GPT2-XL	Short-Nested	Embedded	PS	0.00
GPT2-XL	Short-Nested	Embedded	PP	0.00
GPT2-XL	Long-Nested	Main	SSS	0.02
GPT2-XL	Long-Nested	Main	SSP	0.01
GPT2-XL	Long-Nested	Main	SPS	0.19
GPT2-XL	Long-Nested	Main	SPP	0.14
GPT2-XL	Long-Nested	Main	PSS	0.03
GPT2-XL	Long-Nested	Main	PSP	0.05
GPT2-XL	Long-Nested	Main	PPS	0.01
GPT2-XL	Long-Nested	Main	PPP	0.02
GPT2-XL	Long-Nested	Embedded	SSS	0.05
GPT2-XL	Long-Nested	Embedded	SSP	0.23
GPT2-XL	Long-Nested	Embedded	SPS	0.17
GPT2-XL	Long-Nested	Embedded	SPP	0.07
GPT2-XL	Long-Nested	Embedded	PSS	0.26
GPT2-XL	Long-Nested	Embedded	PSP	0.61
GPT2-XL	Long-Nested	Embedded	PPS	0.09
GPT2-XL	Long-Nested	Embedded	PPP	0.03