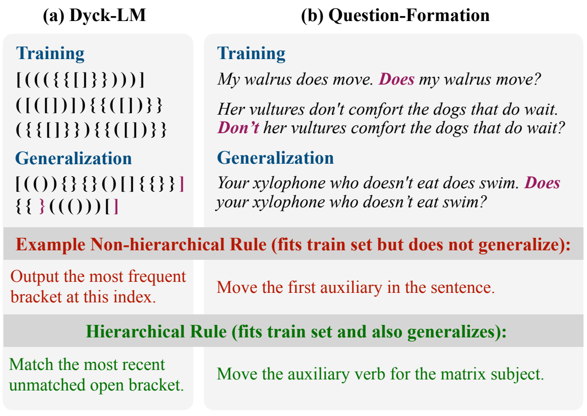
# Grokking of Hierarchical Structure in Vanilla Transformers

**Shikhar Murty**[†] **Pratyusha Sharma**[‡] **Jacob Andreas**[‡] **Christopher D. Manning**[†]
[†]Computer Science Department, Stanford University [‡]MIT CSAIL
{smurty, manning}@cs.stanford.edu, {pratyusha, jda}@mit.edu

## Abstract

For humans, language production and comprehension are sensitive to the hierarchical structure of sentences. In natural language processing, past work has questioned how effectively neural sequence models like transformers capture this hierarchical structure when generalizing to structurally novel inputs. We show that transformer language models can learn to generalize hierarchically after training for extremely long periods—far beyond the point when in-domain accuracy has saturated. We call this phenomenon *structural grokking*. On multiple datasets, structural grokking exhibits inverted U-shaped scaling in model depth: intermediate-depth models generalize better than both very deep and very shallow transformers. When analyzing the relationship between model-internal properties and grokking, we find that optimal depth for grokking can be identified using the tree-structuredness metric of Murty et al. (2023). Overall, our work provides strong evidence that, with extended training, vanilla transformers discover and use hierarchical structure.


**(a) Dyck-LM**

**Training**
[ ( ( ( { { [ ] } } ) ) ) ]
( [ ( [ ] ) ] ) { { ( [ ] ) } }
( { { [ ] } } ) { { ( [ ] ) } }

**Generalization**
[ ( ( ) ) { } { } ( ) [ ] { { } } ]
{ { } ( ( ( ) ) ) [ ]

**(b) Question-Formation**

**Training**
*My walrus does move.* ***Does*** *my walrus move?*

*Her vultures don't comfort the dogs that do wait.* ***Don't*** *her vultures comfort the dogs that do wait?*

**Generalization**
*Your xylophone who doesn't eat does swim.* ***Does*** *your xylophone who doesn't eat swim?*

**Example Non-hierarchical Rule (fits train set but does not generalize):**

Output the most frequent bracket at this index.

Move the first auxiliary in the sentence.

**Hierarchical Rule (fits train set and also generalizes):**

Match the most recent unmatched open bracket.

Move the auxiliary verb for the matrix subject.

Figure 1: Examples from language modeling datasets we use to assess hierarchical generalization in vanilla transformers. These datasets are constructed so that both a non-hierarchical as well as a hierarchical rule can perfectly fit the training set, but only the hierarchical rule generalizes to structurally novel inputs.

## 1 Introduction

Although human language is produced as a linear sequence, it is hierarchically organized. Smaller units compose to form larger constituents. The ability to infer this hierarchical structure underlies our ability to produce and understand new sentences (Chomsky, 1965; Crain and Nakayama, 1987). In this paper, we investigate whether standard neural transformer models (Vaswani et al., 2017) can also generalize hierarchically when trained on language processing tasks (Fig 1). Our main finding is that hierarchical generalization in transformers does occur, but very slowly: performance on structurally novel sentences increases gradually, long after performance on sentences from the training distribution has plateaued. We term this phenomenon *structural grokking*, by analogy to existing findings on simple classification tasks (Power et al., 2022).

On two datasets, we show that structural grokking exhibits inverted U-shaped scaling behavior as a function of model depth: hierarchical generalization improves, then declines, as we train deeper models. Prior work suggests that a number of model-internal properties might track the emergence of hierarchical structure in transformers, including weight norms (Merrill et al., 2021; Liu et al., 2022; Power et al., 2022), attention sparsity (Merrill et al., 2021), and functional tree-structuredness (Murty et al., 2023). We find that functional tree-structuredness is uniquely able to predict structural grokking—while weight norms and attention sparsity increase monotonically in model depth, tree-structuredness is highest for models of the optimal depth for structural grokking.

Our results challenge findings from prior work (Mueller et al., 2022; Petty and Frank, 2021) claiming that ordinary transformers completely fail on the tests of hierarchical generalization that we study. We attribute these failures to early stopping based on in-domain validation performance, which significantly underestimates hierarchical generalization due to structural grokking. On the datasets where this prior work reports generalization accuracies below 20%, *simply by training for longer*, mean accuracy across random seeds reaches 80%, and several seeds achieve near-perfect generalization performance. Past findings are also partially explained by U-shaped scaling: this work uses models that are too shallow (Mueller et al., 2022; Petty and Frank, 2021) or too deep (Mueller et al., 2022). Our results align with past findings on the role of extended training in other language processing problems (Csordás et al., 2021; Hoffmann et al., 2022).

## 2 Background

**Transformers** Given a sequence of tokens $w_{\leq i} = w_1, w_2, \ldots, w_i$, where each token is drawn from a fixed vocabulary $V$, an $L$-layer transformer language model (LM) $f_\theta^L$ outputs a distribution over the next token $w_{i+1} \in V$, $f_\theta^L(w_{\leq i}) \in \mathbb{R}^{|V|}$. A key part of the architecture is a sequence of $L$ *self-attention* layers, where layer $p$ computes contextual vectors of token $k$ as a non-linear parametric function of a convex combination of contextual vectors of tokens $w_{\leq k}$ from the previous layer, where coefficients $\mathbf{a}_k^p \in \mathbb{R}^k$ are known as the *attention distribution*. The LM weights are learned by maximizing the log probability of the correct continuation $w_{k+1}$, given prefix $w_{\leq k}$.

**Hierarchical structure in transformers** While unsupervised pre-training of transformers has led to state-of-the-art transfer learning results across NLP, the architecture itself has been claimed to lack human-like inductive biases toward hierarchical structure (Tran et al., 2018; Hahn, 2020; Petty and Frank, 2021; Mueller et al., 2022). We revisit these claims in this work.

To understand whether a given model has a bias for acquiring hierarchical structure, we follow McCoy et al. (2020) and evaluate generalization in models trained on ambiguous tasks in which training data is consistent with both a "hierarchical rule" as well as a "non-hierarchical rule" (Fig 1). To test if the hierarchical rule has been acquired, we test generalization on a separate out-of-distribution test set, constructed such that only learners that have acquired the hierarchical rule are successful.

**Grokking** Power et al. (2022) identify the phenomenon of *grokking* on small algorithmic datasets where they find that test performance improves long after training performance has saturated. We hypothesize a similar *structural grokking*, where the model groks hierarchical structure long after in-domain validation performance has saturated, and consequently, hierarchical generalization can continue to improve with extended training.

## 3 Experiments

**Datasets** Since our goal is to understand hierarchical generalization in transformers, we use two datasets from (McCoy et al., 2020) and additionally evaluate on a simple bracket-tracking task. For *Dyck*, models are trained to predict next tokens in strings drawn from $\text{Dyck}_{20,10}$, the language of well-nested brackets with 20 types and max nesting depth of 10. We evaluate generalization to structurally unobserved strings in $\text{Dyck}_{20,10}$ (see Fig 1 for examples and Appendix A for details). For the McCoy et al. (2020) datasets, in *Question-Formation*, models must convert English sentences into questions and, in *Tense-Inflection*, models must map from sentences and tense markers to appropriately re-inflected sentences. We evaluate generalization on the out-of-distribution test set from McCoy et al. (2020).

**Model** We train transformer LMs with {2, 4, 6, 8, 10} layers (see Appendix B for more details). For each depth, we train models with 10 random seeds for 300k (400k for Dyck) steps. Given the input sentence (or prefix in the case of Dyck) we decode greedily from the model at test time. For Dyck, we report the accuracy of generating the correct closing bracket type by ranking among closing brackets, given an input prefix from the language. As done in prior work (McCoy et al., 2020; Petty and Frank, 2021; Mueller et al., 2022), for Question-Formation, we report first word accuracy of the decoded question, and for Tense-Inflection, we report the fraction of test inputs for which the target verb is correctly inflected.

### 3.1 Main Results

**Transformers exhibit structural grokking** We first present results obtained with the best model depth on all datasets in Fig 2. We find clear evidence of structural grokking: Across datasets, generalization improves many training steps after in-distribution accuracy has saturated, sometimes approaching perfect accuracy.

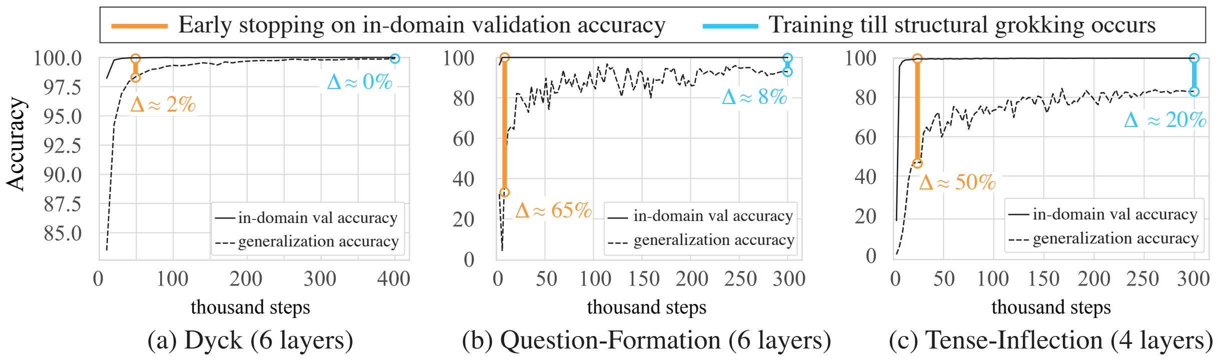


Figure 2: Average accuracy across 10 random seeds on the in-domain val set (solid) and generalization set (dashed) for all datasets. Generalization performance improves even after in-domain accuracies have saturated, showing *structural grokking*. We highlight with orange and blue lines the gap between in-domain and generalization accuracies at the point of early stopping based on the in-domain val set performance vs. at the end of training, noting that prior work stops training at the orange line. Stopping training prior to structural grokking can result in a vast underestimation of generalization performance.

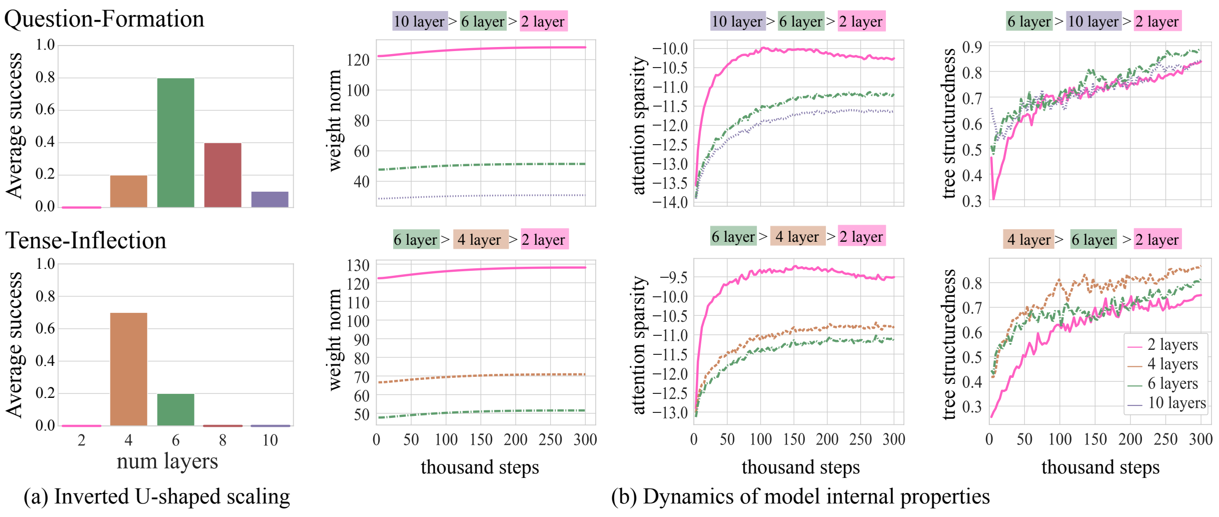


Figure 3: (a) Inverted U-shaped laws for grokking: On Question-Formation (top) and Tense-Inflection (bottom), we find that both very small and very deep models either fail to exhibit structural grokking or do so infrequently, compared to an in-between optimal model depth. (b) While weight norms and attention sparsity increase for all models and do not differentiate between different sizes, tree-structuredness is highest for the optimal model depth.

**Early stopping considered harmful** Next, we compare generalization accuracy obtained by early stopping on in-domain validation accuracy (as done in Petty and Frank (2021); Mueller et al. (2022)) to longer training runs (Fig 2). Early stopping leads to vastly underestimating generalization. For instance, average generalization goes up from <40%, <50% to <90%, <80% on Question-Formation and Tense-Inflection, respectively.

**Inverted U-shaped scaling** On Question-Formation and Tense-Inflection, we train models of increasing depths from 2 to 10 layers. For each depth, we report the fraction of seeds (out of 10) where generalization accuracy eventually crosses 80%, in Fig 3a. We find an inverted U-shaped scaling behavior—very shallow and very deep models are unsuccessful, while most seeds generalize in models of intermediate depth. This may also explain why prior work that either used very shallow models (1–3-layer transformers in Petty and Frank (2021); Mueller et al. (2022)) or very deep models (12-layer transformers in Mueller et al. (2022)) failed to generalize well.

## 4 Analysis

Given that structural grokking occurs only in a subset of model architectures, can we identify when it has happened (or predict when it will occur)? Several model-internal properties have been claimed to relate to either grokking or emergent hierarchical structure in transformers.

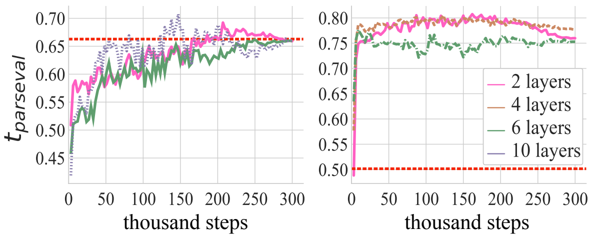
**Weight Norms** Recent work (Power et al., 2022; Liu et al., 2022) identifies the $L_2$ norm of parameter weights as an important quantity for grokking. For instance, Power et al. (2022) find weight decay to improve grokking speed and Liu et al. (2022) identify a "goldilocks zone" in weight norm space where grokking occurs. More generally, norm growth over the course of training has been studied as a key factor in neural network generalization (Soudry et al., 2018).

**Attention Sparsity** Merrill et al. (2021) prove that norm growth in transformers leads to attention saturation, an important property for emergent linguistic structure (Merrill et al., 2022). As a proxy for attention sparsity of $f_\theta^L$, we compute the negative mean entropy of all distributions $\{\mathbf{a}_k^p\}$.

**Tree-structuredness** McCoy et al. (2020) show that tree-structured encoders such as Tai et al. (2015) show near perfect hierarchical generalization. While transformers are relatively unconstrained, recent evidence suggests that, when trained on language data, they implictly implement (approximately) tree-structured computations. In particular, the *tree projection* method of Murty et al. (2023) precisely characterizes the extent to which a transformer's internal computation on an input can be approximated with a tree-structured neural encoding, providing a tree-structuredness score ($t_{\text{score}}$) for any transformer, and a binary tree that best approximates its computation on an input string (see Appendix C for details). To evaluate whether these trees correspond to human notions of syntax, we additionally compare recovered trees to gold-standard ones ($t_{\text{parseval}}$, Black et al., 1991).

### 4.1 Results

We characterize the *dynamics* of weight norms (normalized by number of layers to compare different model depths), attention sparsity, and tree-structuredness, by computing these quantities every 3k gradient updates for Question-Formation and Tense-Inflection. For data-dependent properties such as attention sparsity and tree-structuredness, we sample 10k examples from the training data. We plot these quantities for the smallest model, the largest model for which at least one run shows successful grokking, and for the optimal model depth, in Fig 3b.



(a) Question-Formation (b) Tense-Inflection

Figure 4: While some models fail to generalize hierarchically, all models are effective at learning computations whose closest tree structures progressively evolve towards ground truth syntax, matching or outperforming a right branching baseline (in dashed red).

**Optimal models are most tree-structured** Weight norms and attention sparsity grow for all model settings in both datasets. However, these properties by themselves are unable to predict that both shallow and deep models fail—shallow models learn the sparsest solutions as well as solutions with largest weight norms, but never generalize hierarchically. As noted by Murty et al. (2023), $t_{\text{score}}$ improves over time for all models, indicating increased tree-structuredness over time. For both datasets, the "optimal" model learns the most tree-structured solution compared to both deep and shallow models. Liu et al. (2022) note that, on algorithmic tasks, grokking "coincides with the emergence of structure in embeddings". Similarly, for language tasks, we find that structural grokking coincides with the emergence of tree structured internal computations.

**Transformers are surprisingly effective at structure induction** From the dynamics of $t_{\text{parseval}}$ in Fig 4, we note that all models, *regardless of whether they generalize or not*, learn structures that are close to ground truth syntax, sometimes outperforming a right-branching baseline. McCoy et al. (2020) note that tree-structured encoders only generalize when structured according to correct parse trees. Here, we find that all transformers learn correct tree structures, but only the ones that are the most tree-structured generalize best.

## 5 Conclusion

This work shows that transformers are capable of exhibiting *structure-sensitive* "hierarchical generalization" via a grokking mechanism. Their overall learning behavior gradually shifts from memorization (high in-domain accuracy, poor out-of-domain accuracy) to generalization (high in-domain and out-of-domain accuracy). While we show such behavior on relatively small datasets with small models, we believe these results may have broader implications, as training for longer has been shown to help even for web-scale language modeling (Hoffmann et al., 2022) and on compositional generalization tasks (Csordás et al., 2021). Structural grokking happens most often at "medium-sized" model depths, and both very shallow and very deep models fail to exhibit it. While properties previously connected with linguistic generalization in transformers such as weight norms and attention sparsity do not differentiate good architectures from bad ones, functional tree-structuredness of the transformer can well predict the optimal model depth. While there are clear limitations to the transformer architecture (such as the inability to implement unbounded recursion), our results show that it may have stronger inductive biases than previously believed: With sufficient training, transformers can represent hierarchical sentence structure and use this structure to generalize correctly.

## 6 Reproducibility

All code and data for these experiments is available at `https://github.com/MurtyShikhar/structural-grokking.git`.

## 7 Acknowledgements

SM was funded by a gift from Apple Inc. CM is a fellow in the CIFAR Learning in Machines and Brains program. We thank John Hewitt, Belinda Li, Rishi Bommasani and members of the Stanford NLP group for feedback on the paper.

## Limitations

Our work has the following limitations. First, we only evaluate generalization on datasets based on English language. Second, we show structural grokking on three datasets, and while we believe this to be a general phenomenon, we leave investigating similar behavior on other datasets for future work. Next, we also do not study the effect of training data size on structural grokking, and do not investigate whether transformers learn to grok hierarchical structure in low data regimes. Finally, all datasets here are based on context-free grammars, either similar to or taken directly from prior work, and we believe constructing similar generalization benchmarks on real language data is a good avenue for future work.

## References

E. Black, S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Speech and Natural Language: Proceedings of a Workshop Held at Pacific Grove, California, February 19-22, 1991*.

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

Stephen Crain and Mineharu Nakayama. 1987. Structure dependence in grammar formation. *Language*, 63(3):522–543.

Róbert Csordás, Kazuki Irie, and Juergen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 619–634, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Hahn. 2020. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. In *Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS 2022)*.

Ziming Liu, Eric J Michaud, and Max Tegmark. 2022. Omnigrok: Grokking beyond algorithmic data. In *The Eleventh International Conference on Learning Representations*.

R. Thomas McCoy, Robert Frank, and Tal Linzen. 2020. Does syntax need to grow on trees? Sources of hierarchical inductive bias in sequence-to-sequence networks. *Transactions of the Association for Computational Linguistics*.

William Merrill, Vivek Ramanujan, Yoav Goldberg, Roy Schwartz, and Noah A. Smith. 2021. Effects of parameter norm growth during transformer training: Inductive bias from gradient descent. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1766–1781, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

William Merrill, Ashish Sabharwal, and Noah A. Smith. 2022. Saturated transformers are constant-depth threshold circuits. *Transactions of the Association for Computational Linguistics*, 10:843–856.

Aaron Mueller, Robert Frank, Tal Linzen, Luheng Wang, and Sebastian Schuster. 2022. Coloring the blank slate: Pre-training imparts a hierarchical inductive bias to sequence-to-sequence models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1352–1368, Dublin, Ireland. Association for Computational Linguistics.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D Manning. 2023. Characterizing intrinsic compositionality in transformers with tree projections. In *The Eleventh International Conference on Learning Representations*.

Jackson Petty and Robert Frank. 2021. Transformers generalize linearly. *arXiv preprint arXiv:2109.12036*.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. 2022. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. 2018. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Ke M Tran, Arianna Bisazza, and Christof Monz. 2018. The importance of being recurrent for modeling hierarchical structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4731–4736.

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*, volume 30.

| Dataset | Train | In-Domain Val | Generalization |
|---|---|---|---|
| Dyck | 200000 | 20000 | 20000 |
| Question-Formation | 100000 | 1000 | 10000 |
| Tense-Inflection | 100000 | 1000 | 10000 |

Table 1: Statistics for all datasets used in this work.

## A Dataset Details

All statistics are in Table 1. For Question-Formation and Tense-Inflection, we use splits as given in McCoy et al. (2020) with no additional preprocessing. We give details of Dyck below.

**Dyck Details** We construct our Dyck dataset by sampling 200k strings from $\text{Dyck}_{20,10}$, the language of well-nested brackets with 20 different bracket types and nesting depth atmost 10. For each string, we define its structure as a binary vector of 0s and 1s. For instance the structure of "(([]))" is "11110000". To construct a generalization set, we sample strings with unobserved structures i.e. strings whose 0-1 structure does not match the structure of any of the training strings. Since the objective at test time is to measure closing bracket accuracy, we only rank model probability among all closing brackets, and only evaluate on prefixes where the opening bracket is atleast 10 tokens away from its corresponding closing bracket.

## B Model Details

We use a transformer language model with the following hyperparameters:

- Number of attention heads = 4
- Hidden dimensionality = 512
- Tied input and output matrices as done in Press and Wolf (2017)

Next, we use the following hyperpameters for optimization:

- AdamW ($\beta_1$: 0.9, $\beta_2$: 0.999, $\epsilon$: 1e-7), with learning rates in $\{$1e-4, 5e-5, 1e-5$\}$, noting that 1e-4 works best for all experiments. We use a linear warmup scheduler warming up from 0 to the final learning rate over 10k gradient steps.
- We clip gradients to have a max $L_2$ norm of 10.
- We use a batch size of 8.

## C Functional Tree-Structuredness

Tree Projections (TP; Murty et al. (2023)) measure how well computations performed by a given transformer $f$ can be approximated with tree-structured encoders. To do this, TP solves the following optimization problem:

$$\phi_{\text{proj}}, T_{\text{proj}} \triangleq \arg\min_{\phi, T} \mathcal{L}(f, g_\phi, T), \tag{1}$$

where $g_\phi$ is the class of tree structured encoders that processes sentence $S$ according to bottom-up trees $T(S)$, and $\mathcal{L}$ is a distance function between vector outputs of $f$ and $g_\phi$ on spans from the binary tree $T$. TP minimizes Equation 1 approximately, and recovers an approximate $\widehat{T}_{\text{proj}}$. The tree score over a dataset $\mathcal{D}$ is defined as

$$t_{\text{score}} \triangleq \frac{\sum_{S \in \mathcal{D}} \mathbb{E}_T \text{SCI}(S, T) - \text{SCI}(S, \widehat{T}_{\text{proj}}(S))}{|\mathcal{D}|}, \tag{2}$$

where SCI (span contextual invariance) is the distance between contextual and context-free vector representations of all spans $p$ in $T$ (for more details, see Murty et al. (2023)). In particular, SCI score for a sentence $S$ structured according to $T(S)$ is

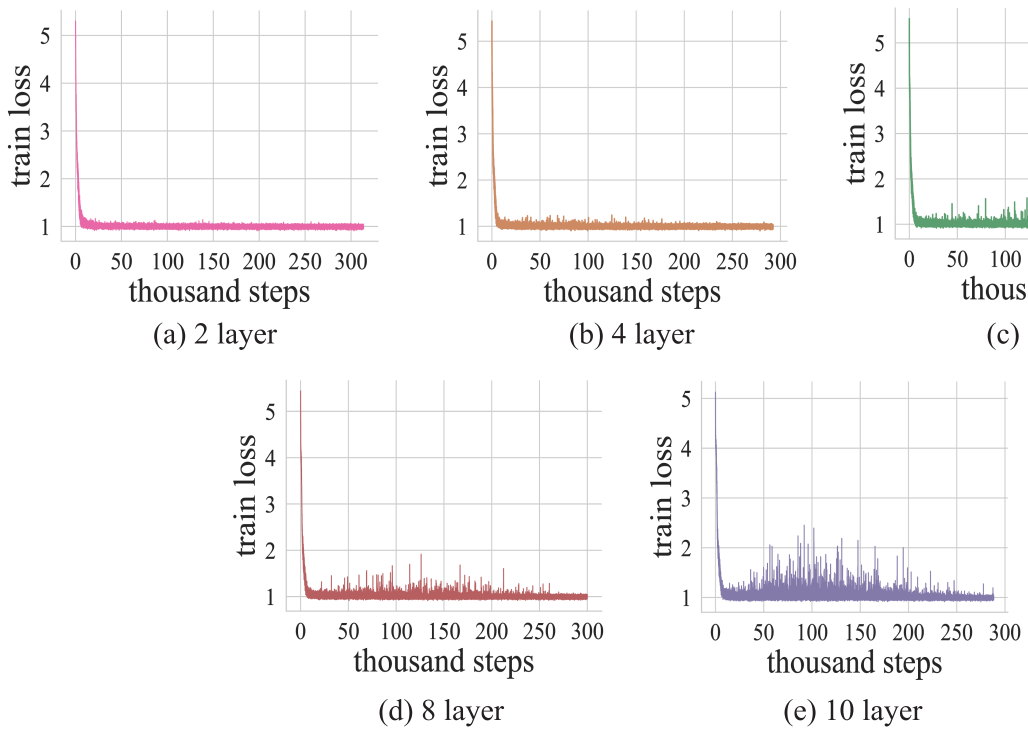
$$\text{SCI}(S, T) \triangleq \sum_{s \in T} d(\boldsymbol{v}_p^S, \tilde{\boldsymbol{v}}_p) \tag{3}$$

for some suitably chosen distance function $d$ (here, cosine similarity). To measure the bracketing F1 score (PARSEVAL; Black et al. (1991)) of the induced tree projection of the transformer $\widehat{T}_{\text{proj}}$ against ground truth gold syntax trees, $T_g$, when available, Murty et al. (2023) define

$$t_{\text{parseval}} \triangleq \text{PARSEVAL}(\widehat{T}_{\text{proj}}, T_g, \mathcal{D}). \tag{4}$$

## D Training Loss Curves

We explore the hypothesis that syntactic grokking is simply a result of the training loss continuing to decrease, even after in-domain validation performance has saturated in Fig 5. We note that training losses generally saturate before in-domain validation performance saturates (also noted in Power et al. (2022)). Next, we also find that all models, regardless of whether they grok or not, eventually get to comparable training losses. We conclude that the inverted U-shaped trend is not an artifact of poorly optimized models.



(a) 2 layer (b) 4 layer (c) 6 layer (d) 8 layer (e) 10 layer

Figure 5: We plot average training loss for all model depths on the Question-Formation dataset. We note that (1) grokking happens even though training losses fully flat line around 10k gradient steps, and that (2) the inverse U-shaped scaling is not a result of poor optimization of small / large models since all models eventually have the same stable training loss as the optimal model depth.

## ACL 2023 Responsible NLP Checklist

**A For every submission:**

☑ A1. Did you describe the limitations of your work?
*Section-6*

☒ A2. Did you discuss any potential risks of your work?
*Left blank.*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*Yes (in Section-1 and Abstract)*

☒ A4. Have you used AI writing assistants when working on this paper?
*Left blank.*

**B ☑ Did you use or create scientific artifacts?**

*We trained several transformer models, the checkpoints of which will be available upon de-anonymization*

☑ B1. Did you cite the creators of artifacts you used?
*We cite all prior work whose datasets we use in Sections-2 and 3*

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*Not applicable. Left blank.*

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
*Not applicable. Left blank.*

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
*Not applicable. Left blank.*

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*Not applicable. Left blank.*

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
*Left blank.*

**C ☑ Did you run computational experiments?**

*Left blank.*

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
*Left blank.*

*The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.*

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
*Left blank.*

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
*Left blank.*

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
*Not applicable. Left blank.*

**D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?**
*Left blank.*

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
*Not applicable. Left blank.*

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
*Not applicable. Left blank.*

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
*Not applicable. Left blank.*

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*Not applicable. Left blank.*

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*Not applicable. Left blank.*