Dependency-Based and Constituency-Based Inductive Biases for BabyLM: Comparison and Analysis

Zhuoxuan Ju
zj153@georgetown.edu

Lanni Bu
lb1437@georgetown.edu

Dagny Whall
dew95@georgetown.edu

Vattana Chan
pc1055@georgetown.edu

Abstract

This paper investigates the effect of syntactic inductive bias on language model learning by pretraining models on structure-only data derived from real natural language. We focus on two major syntactic formalisms—constituency and dependency—and examine their respective influence on downstream performance. To isolate structural information, we remove all lexical content and encode only abstract hierarchical patterns. Our results show that models pretrained on structured data outperform those trained on unstructured random sequences. Among the structured variants, dependency-based encodings yield lower perplexity than constituency-based encodings. Additionally, we compare two types of dependency encoding and find that a simple depth-based representation performs comparably to a more expressive arc-based scheme. Finally, we show that structure extracted from real language leads to better performance than synthetic patterns with similar surface characteristics.

1 Introduction

While the exact role of inductive bias in human and machine language learning is not fully understood, it is hypothesized to be a key factor in guiding learners toward generalizable patterns from sparse input. In recent work, syntactic structures have been used as a powerful source of such bias, guiding models toward representations that align more closely with linguistic principles. Among these structures, dependency and constituency representations stand out as two of the most prominent and theoretically grounded syntactic formalisms in natural language processing (NLP). While both have been shown to be implicitly learnable by modern language model(LMs), their comparative effectiveness as explicit inductive biases during pretraining remains underexplored.

To investigate this question, we examine how different structural biases, specifically, those derived from dependency and constituency structures, affect language model learning when injected during the pretraining stage of LMs. Rather than combining structure with surface-level lexical input, we isolate structure itself by constructing pretraining datasets composed purely of abstract syntactic patterns extracted from natural language. This allows us to ask whether the type of structure a model is exposed to influences its ability to generalize and acquire linguistic competence.

We further extend this inquiry by comparing syntactic patterns derived from real human language with purely synthetic counterparts that share surface-level properties but lack linguistic grounding. This second line of investigation allows us to assess whether structural inductive bias is more effective when grounded in natural language data, or whether abstract structural regularities alone suffice.

Through these two experiments, our goal is to clarify the role of syntactic inductive biases in language model training, and to compare how constituency and dependency structures, both foundational in theoretical linguistics, shape model behavior in practice.

2 Related Work

2.1 Injecting Inductive Bias during Pretraining

The method of injecting inductive bias into LMs by leveraging synthetic data during the pretraining stage has been actively explored in recent years. Papadimitriou and Jurafsky (2023) investigate how inductive biases for recursion and for context-sensitive crossing dependencies differently affect language model behavior. McCoy and Griffiths (2023) propose prior-training, which shares key properties with pretraining, to combine the strong inductive biases of a Bayesian model with the representational flexibility of a neural network. Lindemann et al. (2024a) introduce an inductive bias for finite-state transducers (FSTs) by pretraining a Transformer to predict the output of an FST given an input. Their results also reveal that this method encourages the model's internal dynamics to simulate key aspects of FST behavior. Lindemann et al. (2024b) strengthen the structural inductive bias of LMs via an additional intermediate pretraining step designed to perform syntactic transformations. Hu et al. (2025) take a step further by investigating which features of formal languages impart inductive biases that lead to improved natural language acquisition in LMs.

These studies demonstrate that LMs can acquire inductive biases during the pretraining stage, resulting in different model performance. Building upon this line of work, we focus on injecting inductive bias during pretraining through two of the most prominent syntactic representations, constituency and dependency structures, extracted from real human language.

2.2 Syntactic Structures as Inductive Bias

Natural language syntax is commonly represented through two major types of syntactic structures: dependency structures (Nivre et al., 2016), which capture one-to-one relationships between words, and constituency structures, which represent how words or groups of words combine into larger syntactic units.

Previous work has shown that both dependency and constituency structures can be implicitly acquired by language models. Hewitt and Manning (2019) demonstrate that syntactic dependency trees are implicitly encoded in the geometry of transformer representations, and can be recovered by applying a structural probe consisting of a learned linear projection and a minimum spanning tree algorithm over pairwise L2 distances. Similarly, Arps et al. (2022) show that language models are capable of identifying properties of various constituent types, and that full constituency trees are linearly separable from the models' hidden states with high accuracy.

Given this, both types of structure have been widely used as sources of inductive bias in language modeling (Zhou et al., 2020; Xu et al., 2021; Li et al., 2021), with the goal of improving model performance. In this work, we adopt both constituency and dependency structures as inductive biases to evaluate whether they lead to different effects on model behavior.

3 Methods

The core idea of our research is to inject purely structural data during the pretraining stage to examine whether language models behave differently depending on the type of structure they are exposed to. Specifically, we pretrain models on data containing only structural information, then finetune them on the BabyLM 100M dataset (Choshen et al., 2024).

3.1 Pretraining Data

To extract syntactic structures from natural language, we use the Stanza parser (Bauer et al., 2023) to analyze the Wikipedia dataset (Wikimedia Foundation, 2024) and obtain both constituency and dependency parses.

Constituency Structure. From each constituency parse, we retain only the bracketing information and remove all lexical content. Each matched pair of brackets is replaced with a pair of identical, randomly assigned numeric tokens, resulting in a purely nested structural representation. The conversion procedure is shown in Figure 1.

Dependency Structure. Dependency structure is more complex than constituency structure. Although both are tree-based, dependency parses cannot be encoded using the same bracketing format as constituency trees, as doing so would disrupt word order. Since preserving word order is crucial for our setting, we propose two encoding strategies to extract the structural information from dependency trees:

1. **Arc-based Encoding:** Each dependency relation is represented as a pair of matching numeric tokens inserted around the head and dependent words. Nested pairs reflect projective dependency structures, while crossing pairs indicate non-projective structures. As with the constituency encoding, all lexical content is removed. An illustration of the transformation process from a dependency structure to a purely numeric representation using arc-based encoding is provided in Figure 2.

2. **Depth-based Encoding:** All words at the same depth level in the dependency tree are

assigned the same numeric token. This encoding classifies words by syntactic depth, while omitting information about the relative depth across layers and the specific syntactic dependencies among words. Figure 3 illustrates the encoding process for depth-based representation.

3.2 Implementation

In our experiments, we use the GPT2-Small model as our base model. For pretraining, the dataset consists of 1B tokens with a vocabulary size of 500 tokens. We use a batch size of 512 and train the models for 5,000 steps. In the finetuning stage, we use the BabyLM 100M dataset (Choshen et al., 2024). The batch size is set to 128, and we train totally for 5 epochs.

3.3 Evaluation

Model performance is evaluated using perplexity on the test split of the BabyLM dataset. We also report results on downstream task metrics provided by the 3rd BabyLM Challenge (Charpentier et al., 2025), using models trained for two epochs during finetuning stage.

4 Experiments

4.1 Experiment 1: Dependency vs. Constituency as Inductive Biases

We investigate how constituency and dependency structures, when used as inductive biases, affect the performance of language models. We compare the finetuning performance of models pretrained on data encoding constituency and dependency structures. Additionally, we include a baseline pretraining condition, in which the data consists of randomly generated numeric sequences without any underlying structural information. This baseline is included to demonstrate that structural information in the data leads to better model performance, and to validate the soundness of our experimental setup. The perplexity results are shown in Figure 4, and the downstream task results are presented in Table 1. Models pretrained on structured data outperform those pretrained on random data, confirming the validity of our experimental setup.

Dependency Encoding Outperforms Constituency Encoding. Models pretrained on either of the two dependency encoding strategies achieve lower perplexity than that pretrained on constituency-encoded data. For the arc-based encoding strategy in particular, this phenomenon is consistent with Papadimitriou and Jurafsky (2023), which demonstrate that crossing structures can serve as a stronger inductive bias for language models than nested structures. They also show that introducing a small number of crossing structures into nested structures can lead to better performance than using nested structures alone. Non-projective structures, which naturally occur in dependency trees, are reflected as crossing structures in our arc-based representation and may explain the improved performance.

Different Dependency Encodings Perform Similarly. Models pretrained on the two dependency encoding strategies achieve nearly identical perplexity scores. This result is somewhat unexpected, as the depth-based encoding strategy provides a substantially simpler structural information than the arc-based encoding strategy. In contrast to the arc-based encoding strategy, which captures nested structures that can reflect complex hierarchical relationships, the depth-based encoding assigns tokens to discrete depth levels, distinguishing them by syntactic depth without indicating which levels are deeper or shallower, or whether any syntactic relationships exist between tokens. Despite these differences, both encodings strategies lead to comparable performance.

4.2 Experiment 2: Natural vs. Synthetic Nested Structures

We explore whether structural patterns derived from natural language yield different performance compared to purely synthetic structures with similar surface characteristics. We introduce an additional pretraining data composed of randomly generated nested structures. During synthetic nested pretraining data generation, each position in the sequence is assigned a fixed probability of being an "opening" token, while non-opening positions attempt to serve as "closing" tokens, matching with the most recent unmatched opening token to form a valid nested pair. We compare the performance of the model pretrained on this synthetic nested data with the model pretrained on constituency-encoded data, which also exhibits a purely nested structure but is extracted from real human language.

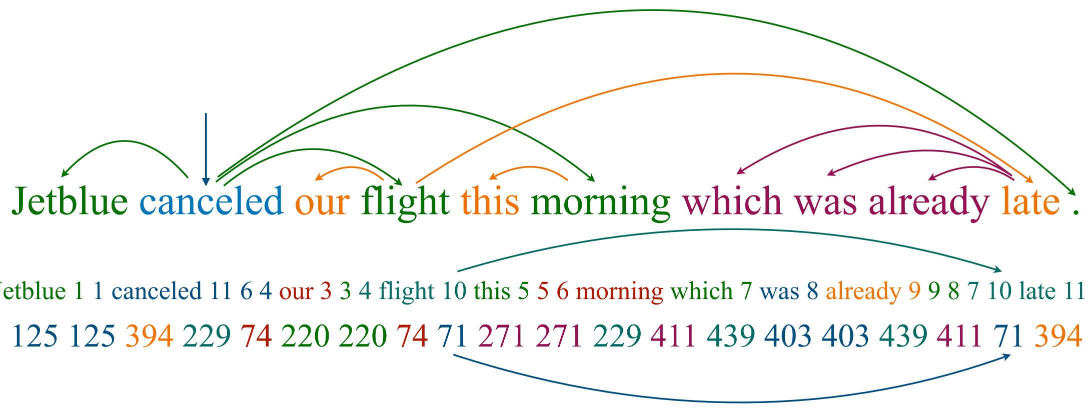
The results, also shown in Figure 4 and Table 1, reveal that even though both datasets share a purely nested structure, there are notable differences in perplexity score, suggesting that structural informa-

(ROOT (S (NP (PRP She)) (VP (VBD gave) (NP (PRP me)) (NP (DT the) (NN book))) (. .)))

(((( )) (( ) (( )) (( ) ( ))) ( )))

96 14 33 15 15 33 71 56 56 6 82 82 6 40 38 38 370 370 40 71 153 153 14 96

Figure 1: Conversion of a constituency parse into a structure-only representation. The top line shows the original parse tree with bracketed syntactic categories. In the middle, each bracket is retained while lexical content is removed. In the bottom line, each matching bracket pair is replaced with the same randomly assigned numeric token, resulting in a purely nested structural sequence.



Jetblue canceled our flight this morning which was already late .
Jetblue 1 1 canceled 11 6 4 our 3 3 4 flight 10 this 5 5 6 morning which 7 was 8 already 9 9 8 7 10 late 11 .
125 125 394 229 74 220 220 74 71 271 271 229 411 439 403 403 439 411 71 394

Figure 2: Visualization of the transformation from a dependency structure to its arc-based encoding. The top line shows the original sentence with arcs indicating head-dependent relations. In the middle line, each dependency relation is converted by inserting a matching pair of numeric tokens around the head and dependent words to represent the existence of the relation. In the bottom line, lexical content is removed, and each token pair is replaced with a randomly assigned numeric value. The arcs in the lower two lines highlight crossing structures, which result from non-projective dependencies in the original sentence.

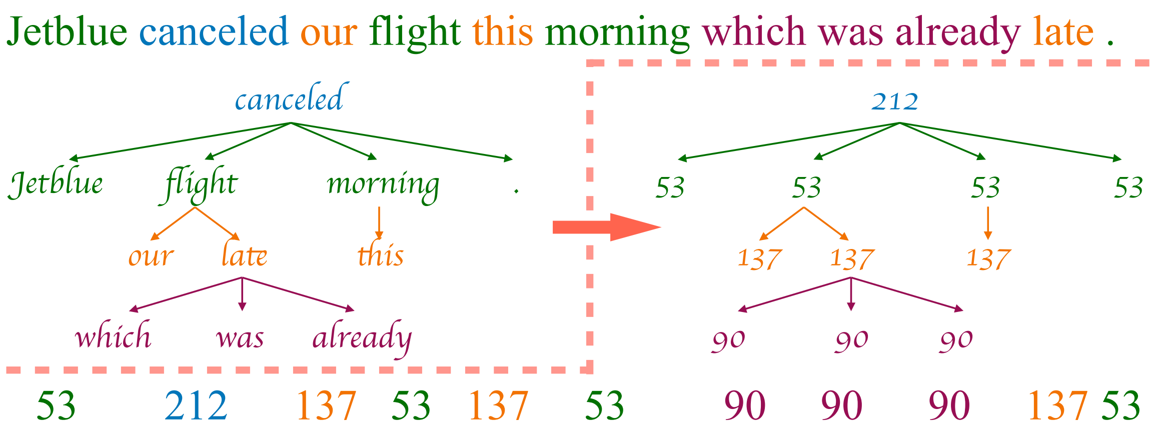


Figure 3: Visualization of the transformation from a dependency structure to its depth-based encoding. The left side shows the original dependency tree, where each word is placed at its corresponding syntactic depth. On the right, all words at the same depth level are assigned the same numeric token, replacing lexical content with abstract group identifiers. This encoding captures hierarchical grouping while omitting explicit head-dependent relations and relative depth between layers.

Model	BLiMP ↑	BLiMP-S ↑	GLUE ↑	EWOK ↑	Eye Tracking ↑	Self-paced Reading ↑	Entity Tracking ↑	WUGs ↑
Experiment 1: Dependency vs. Constituency as Inductive Biases								
Random	66.47	53.99	50.95	49.69	0.95	0.08	13.58	65.50
Constituency	70.26	57.76	51.51	50.12	0.83	0.02	14.37	37.00
Arc	70.66	60.38	51.59	50.40	0.83	0.08	26.26	44.00
Depth	69.36	60.10	51.31	50.92	1.22	0.04	26.22	49.00
Experiment 2: Natural vs. Synthetic Nested Structures								
Nested	69.04	58.63	51.22	49.74	0.61	0.08	15.40	39.00

Table 1: Performance on Downstream Tasks for Different Structural Pretraining Data.

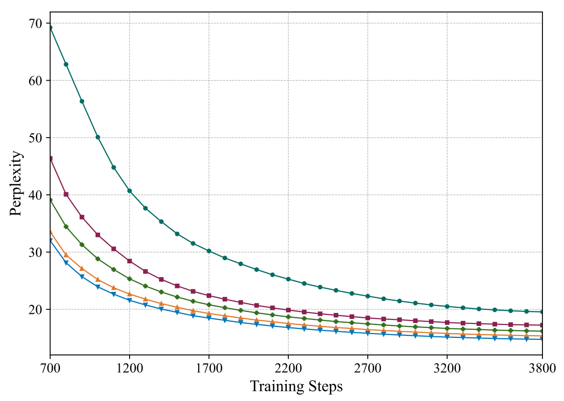


Figure 4: Perplexity over Training Steps for Different Structural Pretraining Data

tion drawn from real language are not equivalent to synthetic data, even if they appear to share similar surface structures.

5 Limitations

During our experiment, we only partially captured the hierarchical properties of dependency trees. While our arc-based and depth-based encodings introduce localized structural cues, such as nested brackets and depth levels. They do not fully preserve the global hierarchical organization of the tree, such as the relative depth between dependents and full head-dependent paths. We leave the development of more complete hierarchical encodings as part of future work.

Although our approach sheds light on how inductive biases shape learnability in artificial systems, we recognize a gap between the learning environment of our models and that of human language learners. In particular, the amount of data used for fine-tuning (e.g., BabyLM 100M) greatly exceeds the quantity and nature of linguistic input available to children, which tends to be sparse, socially grounded, and interaction-based. This discrepancy raises important questions about the cognitive plausibility of our setup. As a next step, we suggest exploring whether similar learning outcomes and convergence patterns can be observed when models are fine-tuned on more constrained and developmentally realistic datasets (e.g., BabyLM 10M). Nevertheless, we caution against directly extrapolating our findings to human cognition, as the computational mechanisms underlying language acquisition in neural models remain fundamentally different from those of human learners. Our results are best interpreted as computational analogies that inform, rather than confirm, theories of human language learning.

6 Conclusion

In this work, we examined the impact of syntactic inductive biases on language model learning by pretraining on structure-only data derived from constituency and dependency representations. Our experiments show that models trained with structured data consistently outperform those trained on unstructured baselines, confirming the value of syntactic information even in the absence of lexical content. Furthermore, we found that dependency-based encodings lead to lower perplexity than constituency-based encodings, suggesting that dependency structures may provide a more effective inductive signal under our experimental setup. Interestingly, we also observed that a simplified depth-based encoding performs on par with a more expressive arc-based dependency representation. Finally, our comparison between real and synthetic nested structures indicates that syntactic patterns grounded in natural language offer advantages beyond surface-level regularity. These results contribute to a deeper understanding of how different forms of inductive bias influence language model behavior.

References

David Arps, Younes Samih, Laura Kallmeyer, and Hassan Sajjad. 2022. Probing for constituency structure in neural language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6738–6757, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

John Bauer, Chloé Kiddon, Eric Yeh, Alex Shan, and Christopher D. Manning. 2023. Semgrex and ssurgeon, searching and manipulating dependency graphs. In *Proceedings of the 21st International Workshop on Treebanks and Linguistic Theories (TLT, GURT/SyntaxFest 2023)*, pages 67–73, Washington, D.C. Association for Computational Linguistics.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Hu, Jaap Jumelet, Tal Linzen, Jing Liu, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Wilcox, and Adina Williams. 2025. Babylm turns 3: Call for papers for the 2025 babylm workshop. *Preprint*, arXiv:2502.10645.

Leshem Choshen, Ryan Cotterell, Michael Y. Hu, Tal Linzen, Aaron Mueller, Candace Ross, Alex Warstadt, Ethan Wilcox, Adina Williams, and Chengxu Zhuang. 2024. [call for papers] the 2nd babylm challenge: Sample-efficient pretraining on a developmentally plausible corpus. *Preprint*, arXiv:2404.06214.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. 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 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Michael Y. Hu, Jackson Petty, Chuan Shi, William Merrill, and Tal Linzen. 2025. Between circuits and chomsky: Pre-pretraining on formal languages imparts linguistic biases. *Preprint*, arXiv:2502.19249.

Zhongli Li, Qingyu Zhou, Chao Li, Ke Xu, and Yunbo Cao. 2021. Improving BERT with syntax-aware local attention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 645–653, Online. Association for Computational Linguistics.

Matthias Lindemann, Alexander Koller, and Ivan Titov. 2024a. SIP: Injecting a structural inductive bias into a Seq2Seq model by simulation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6570–6587, Bangkok, Thailand. Association for Computational Linguistics.

Matthias Lindemann, Alexander Koller, and Ivan Titov. 2024b. Strengthening structural inductive biases by pre-training to perform syntactic transformations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11558–11573, Miami, Florida, USA. Association for Computational Linguistics.

R. Thomas McCoy and Thomas L. Griffiths. 2023. Modeling rapid language learning by distilling bayesian priors into artificial neural networks. *Preprint*, arXiv:2305.14701.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Isabel Papadimitriou and Dan Jurafsky. 2023. Injecting structural hints: Using language models to study inductive biases in language learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8402–8413, Singapore. Association for Computational Linguistics.

Wikimedia Foundation. 2024. Wikimedia downloads. `https://dumps.wikimedia.org`. Accessed: 2025-05-17.

Zenan Xu, Daya Guo, Duyu Tang, Qinliang Su, Linjun Shou, Ming Gong, Wanjun Zhong, Xiaojun Quan, Daxin Jiang, and Nan Duan. 2021. Syntax-enhanced pre-trained model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5412–5422, Online. Association for Computational Linguistics.

Junru Zhou, Zhuosheng Zhang, Hai Zhao, and Shuailiang Zhang. 2020. LIMIT-BERT : Linguistics informed multi-task BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4450–4461, Online. Association for Computational Linguistics.