Grammar induction pretraining for language modeling in low resource contexts

Xuanda Chen and **Eva Portelance***
Department of Linguistics, McGill University
Mila - Quebec Artificial Intelligence Institute

Abstract

In the context of the BabyLM challenge, we present a language model which uses pretrained embeddings from a grammar induction model as its first layer. We compare it to one of the challenge's baseline models and a minimally different baseline which uses random embeddings. We find that though our model shows improvement over the challenge's baseline, the model with randomly initialized embeddings performs equally well. Our results suggest that it is not the pretrained embeddings which aided performance, but likely our tokenizer and choice of hyperparameters.

1 Introduction

The BabyLM Challenge (Warstadt et al., 2023)'s goal is to develop language models and training pipelines that can learn reasonable linguistic representations for downstream language modeling task using much more constrained datasets. With this goal in mind, we hypothesized that giving models additional information about syntactic structure may help them learn more generalizable representations of language. As part of the strict track of the challenge, we were not allowed to give additional syntactic labels as part of our training data, so instead we propose to first induce a compound probabilistic context-free grammar (compound-PCFG) over the data using a neural grammar induction model (Kim et al., 2019). There are many ways we can then integrate this syntactic information into a language model. Here, we test a simple method: we initialize a language model using the terminal token embeddings of a trained grammar induction model as its embedding layer. We test the effectiveness of this method on the BabyLM strict-small challenge.[1]

*Corresponding author: eva.portelance@mcgill.ca

[1]All code for this project is available in this github repository. The trained models and preprocessed data can be downloaded from this OSF Project repository`this Open Science Framework (OSF) project repository.`

2 Data and preprocessing

In the experiments which follow, we use the 10 Million word BabyLM task dataset (the strict track small dataset) to train our language models. Prior to training, we preprocessed the dataset to remove any blank lines or unecessary formatting punctuation (e.g. '== Title ==' became 'Title'). Additionally, we split paragraphs such that each new line represented a single sentence and removed any sentence that was longer than 40 words.

2.1 Grammar induction data

Since grammar induction algorithms can be quite memory intensive, we use a subset of the 10M BabyLM dataset to train our grammar induction model. We randomly sampled a tenth of the sentences from the corpus, resulting is a smaller grammar induction dataset containing 991,510 words.

2.2 Tokenizer

We trained a custom tokenizer on the 10M BabyLM dataset. To guarantee coverage we created a tokenizer that produces both subwords and word-level tokens. Since previous grammar induction models used word-level tokens, we wanted to maximize the number of word-level tokens and keep subwords and character tokens to only a limited necessary number. We therefore trained a tokenizer using the WordPiece algorithm with a vocabulary size of 10,000 and a maximum alphabet of 72 tokens.

3 Models

3.1 Grammar induction model

We first trained a compound-PCFG grammar (Kim et al., 2019) over our subset of the BabyLM small corpus described above. PCFG embeddings are trained to encode terminal rule information, e.g., reflecting syntactic categories in grammar, which could further improve model's language understanding ability. We used our tokenizer to split

Proceedings of the 27th Conference on Computational Natural Language Learning: Volume 2: The BabyLM Challenge, pages 69–73

Table 1: Overall mean performance on each benchmark

benchmark	baseline	baseline-token	grammar
BLiMP	62.63	**64.78**	64.44
BLiMP suppl.	54.72	54.66	**54.88**
SuperGLUE	63.38	**68.21**	67.93
MSGS	**69.22**	67.45	68.08

sentences into tokens and then induced trees over the corpus[2]. During learning, the model induces embedding representations for the grammar rules and terminals, where the terminals are the token embeddings.[3]

Once the grammar is induced, we extract the token embedding layer of the grammar and use it as the initial embedding layer for an OPT-125m-like [4] language model with a vocabulary size of 10,000 [5]. We then trained this language model on next token prediction using the full BabyLM 10M dataset. The embedding layer is trained with other layers and not frozen during training. We will refer to this model as the *grammar* model in the sections which follow.

3.2 Baseline models

We compare our model results to the OPT-125m baseline model supplied by the BabyLM challenge (*baseline*) and to a baseline OPT-125m language model that we trained using our tokenizer and randomly initialized embeddings (*baseline-token*), thus using a vocabulary size of 10,000 tokens. *Baseline-token* has the exact same hyperparameters as our *grammar* model and only differs in terms of its initial embeddings, here random ones.

4 Results

Results for the *baseline* model were taken directly from the BabyLM evaluation pipeline project page (`github.com/babylm/evaluation-pipeline`). For the *baseline-token* and *grammar* models, these were trained for 3 epochs and tested on validation accuracy every 100,000 sentences; we report the best models found during training based on next token prediction on the validation dataset.

We tested all models on the BabyLM evaluation tasks, which included the Benchmark of Linguistic Minimal Pairs (BLiMP) (Warstadt et al., 2020a), a custom supplementary set of BLiMP-like tasks, 'Super' benchmark for General Language Understanding Evaluation (SuperGLUE) (Wang et al., 2019), and the Mixed Signals Generalization Set evaluation (MSGS) (Warstadt et al., 2020b). Results are reported in Table 1. The complete performance results by individual task are presented in Tables 4-7 in Appendix C.

The *baseline-token* and *grammar* models generally do better than the *baseline* on all benchmarks except MSGS, where they perform slightly worse. Overall, the gains in performance are small, though the *baseline-token* and *grammar* do seem to do quite a lot better on the SuperGLUE benchmark than the *baseline* in particular. Importantly, we do not find that the *grammar* model performs better than the *baseline-token* model, suggesting the the addition of our pretrained embeddings did not help the model perform better on the evaluation pipeline.

5 Discussion

Though our *grammar* model did do better overall than the BabyLM OPT-125m *baseline*, when we compared it to our *baseline-token* model, we did not find that initializing the model with pretrained grammar induction embeddings helped perfomance overall. Instead, it may be our tokenizer and choice of hyperparameters which helped improve performance between the *baseline* and *baseline-token/grammar* models.

Simply using the terminal embedding layer of a grammar induction model to initialize a language model is not be the most effective way to encode syntactic information into the model. In future work, we would like to consider other methods for combining these two types of models, like enriching the training set with copies of induced constituents or more complex architectural modification to condition recurrent states with rule embeddings representing the syntactic rules applied to generate a sub-string at each state.

[2] See Appendix D for example induced parses

[3] Hyperparameters for the grammar induction model are reported in Appendix A.

[4] We refer to these models as as OPT-125M-like since they minimally vary from this baseline, however since their vocabulary size is 10,000, they in fact have 94M parameters.

[5] 10,000 was the original vocabulary size used in Kim et al. (2019). Since we did not do hyperparameter search over the grammar induction model, we followed their ideal settings.

References

Yoon Kim, Chris Dyer, and Alexander Rush. 2019. Compound probabilistic context-free grammars for grammar induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational*

Linguistics, pages 2369–2385, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Warstadt, Leshem Choshen, Ryan Cotterell, Tal Linzen, Aaron Mueller, Ethan Wilcox, Williams Adina, and Chengxu Zhuang. 2023. Findings of the BabyLM Challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge*. Association for Computational Linguistics (ACL).

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020a. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020b. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.

A Hyperparameters for Grammar Induction

Table 2: Hyper-parameter setting for grammar induction

parameters	values
latent dimension	64
number of preterminal states	60
number of nonterminal states	30
symbol embedding dimension	256
hidden dim for variational LSTM	768
word embedding dim	768
sentence max length	40
vocab size	10000
number of epochs	15
batch size	5
learning rate	1e-4
random seed	1213

B Hyperparameters for OPT language models

Table 3: Hyper-parameter setting for language modeling

parameters	values
embedding size	10000
number of epochs	3
batch size	20
learning rate	1e-4
warm-up steps	2000
gradient clipping threshold	3
max grad norm for gradient clipping	1.0
random seed	527

Table 4: Compute resources for language modeling

parameters	values
Device	A100
Memory	32G of GPU memory
Training time	12 hours

C Complete evaluation results

Table 5: BLiMP accuracy scores

task	baseline	baseline-token	grammar
Anaphor agr.	63.8	68.6	**69.7**
Arg. structure	**70.6**	65.7	63.4
Binding	67.1	66.5	**67.6**
Control/Raising	**66.5**	62.2	60.4
Det.-Noun agr.	**78.5**	77.8	77.2
Ellipsis	**62**	49.3	51.6
Filler-Gap	**63.8**	62.1	63
Irregular forms	67.5	**81.4**	81.2
Island effects	**48.6**	48.5	47.9
NPI licensing	46.7	**56.3**	55
Quantifiers	59.6	**71.4**	68.9
Subject-verb agr.	56.9	**67.5**	67.4
Overall mean	62.63	**64.78**	64.44

Table 6: BLiMP-Supplement accuracy scores

task	baseline	baseline-token	grammar
Hypernym	50	52.3	**53.3**
QA congr. (easy)	54.7	**57.8**	45.3
QA congr. (tricky)	31.5	**41.8**	40
Subj.-aux. inversion	80.3	67.5	**82.9**
Turn taking	**57.1**	53.9	52.9
Overall mean	54.72	54.66	**54.88**

Table 7: Super(GLUE) accuracy scores

task	baseline	baseline-token	grammar
CoLA	64.6	**69.6**	68.8
SST-2	81.9	**85**	83.3
MRPC (F1)	72.5	**76.1**	73.7
QQP (F1)	60.4	78.9	**79.1**
MNLI	57.6	**66.4**	65.9
MNLI-mm	60	66	**67.8**
QNLI	61.5	**66.5**	**66.5**
RTE	60	**52.5**	**52.5**
BoolQ	63.3	**67.6**	65.8
MultiRC	55.2	60.2	**62.3**
WSC	60.2	**61.5**	**61.5**
Overall mean	63.38	**68.21**	67.93

Table 8: MSGS accuracy scores

task	baseline	baseline-token	grammar
contr.-raising/lex. cat.	66.5	66.7	**68.9**
contr.-raising/rel. tok. pos.	67	**67.2**	**67.2**
main verb/lex. cat.	66.5	**66.8**	66.6
main verb/rel. tok. pos.	67.6	**66.8**	**66.8**
synt. cat./lex. cat.	**80.2**	69	71.3
synt. cat./rel. pos.	67.5	**68.2**	67.7
Overall mean	**69.22**	67.45	68.08

D Example trees from grammar induction model

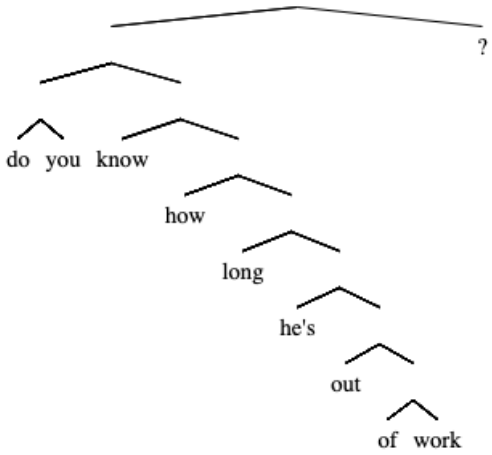

Figure 1: Induced tree for “do you know how long he’s out of work?"

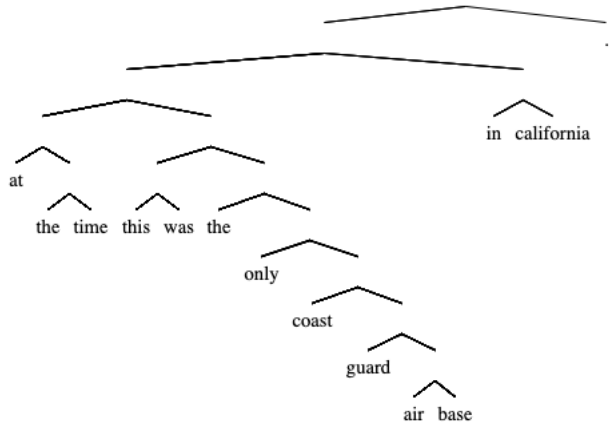

Figure 2: Induced tree for “at the time this was the only coast guard air base in california."

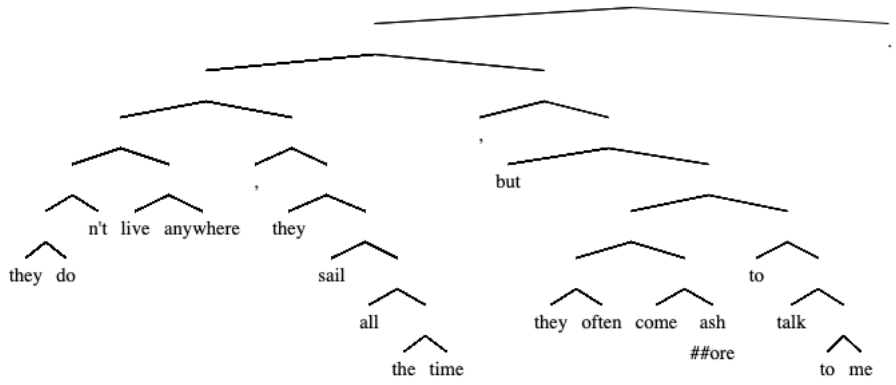

Figure 3: Induced tree for “they don’t live anywhere, they sail all the time, but they often come ashore to talk to me."