Can Transformers Learn SCFGs? An Analysis of Pretraining on Synthetic Texts

Anonymous ACL submission

Abstract

We explore the ability of Transformers to infer Synchronous Context-Free Grammars (SCFGs), i.e. to learn a particular grammar just from example strings. Two experiments were conducted. The first experiment explored Transformers' capacity to translate between synthetic languages corresponding to the source and target side of an SCFG grammar. The second experiment sought for a Transformer configuration which would be capable of SCFG parsing, i.e. identifying the ability to recognize licensed SCFG pairs of strings based on only positive and negative training examples. With a sufficiently large model, Transformers proved capable to learn this task to a high accuracy (96.70%) even for very long inputs, longer than any training items. Experiments show limitations and variability that leave parts of the problem open to further research.

1 Introduction

Transformers (Vaswani et al., 2017) are, in principle, incapable of handling unbounded context-free languages due to their fixed-size architecture (Hahn, 2020). Yet, they excel in many natural language processing tasks, including machine translation, where they achieve near-human quality (Popel et al., 2020), as well as image (Dosovitskiy et al., 2020) and speech (Dong et al., 2018) processing. Translation, however, often requires modeling complex structural alignments between source and target languages, which, for some language pairs, can be described by synchronous context-free grammars (SCFGs) (Chiang, 2006). SCFGs impose a "doubly bracketing" structure—parallel hierarchical dependencies in both languages—making them particularly challenging for Transformers, as their theoretical limitations suggest a need for model size to scale with input length (Hahn, 2020). While these theoretical constraints are well-established, their practical implications remain underexplored. How large must a Transformer be to process SCFGs of varying complexity? How many examples, and of what diversity, are needed to learn such grammars? This paper studies the empirical behavior of Transformers on synthetic SCFGs to estimate these practical limits and investigate their learnability.

We focus on two tasks: **translation**, where Transformers must induce and apply SCFG rules to map between synchronized languages, and **parsing**, where they determine whether a string pair conforms to the grammar. Our experiments aim to assess whether Transformers can generalize from examples— To this end, we:

- **Design SCFG templates** that emphasize key properties, such as nested productions and depth of recursion, while maintaining simplicity (e.g., limited vocabulary size).
- **Pre-train two Transformer architectures**:
  1. **Encoder-Decoder** (Seq2Seq) for translation between synchronized languages.
  2. **Encoder-Only** for parsing, predicting whether sentence pairs conform to the SCFG's rules, even for longer inputs than those seen during training. This tests generalization beyond the training set.

2 Synchronous Context-Free Grammars

Synchronous Context-Free Grammars (SCFGs) extend standard Context-Free Grammars (CFGs) by generating pairs of related strings simultaneously (Chiang, 2006). SCFGs consist of production rules with dual right-hand sides (rhs), referred to as the "source rhs" and "target rhs". Each production synchronously generates the same non-terminals on both sides, though their positions may differ, allowing flexibility in the order between source and target outputs. Non-terminals in SCFGs are linked via indices—numbers that connect each source non-terminal to its corresponding target non-terminal—enabling synchronous generation. This structure fa-

cilitates the alignment of syntactic structures across languages, making SCFGs particularly suited for translation tasks.

2.1 Experimental Grammar Design

The constructed grammar for this study incorporates the following features:

- **Recursion depth**: Enables complex hierarchical structures.
- **Limited vocabulary**: Limited terminal symbols (vocabulary of the two languages).
- **Binary form**: Pre-binarized production rules to ensure compatibility with the CYK-like SCFG parser, which requires each grammar to be in Chomsky Normal Form. This restriction allows efficient parsing algorithms, at the expense of restricted translation power (Gildea and Satta, 2016).
- **No ϵ-productions**: they would introduce a further level of complexity.
- **Ambiguity**: A single token on the source right-hand side may correspond to multiple possible translations on the target right-hand side.

The grammars used for the generation of pairs are reported in Appendix A. A Synchronous CYK Parser, also called Bi-Text Parser (Chiang, 2006), was implemented from scratch to check translations (for the Section 3 experiment) and to filter datasets (for both experiments).

See more details about the grammars G and G_{14} and the "anti-grammars" G', G_{rand}, and $G_{rand_{14}}$ and the parser in Appendix A.

3 First Experiment: Machine Translation

This experiment evaluates the performance of a sequence-to-sequence Transformer model for machine translation, implemented using the `fairseq` (Ott et al., 2019), a PyTorch-based framework developed by Meta. The objective was to train a Transformer on paired sentences and assess its ability to generate accurate target sentences from source sentences during testing. Specifically, we evaluated whether the generated translations are licensed by the specified SCFG (in this experiment: G) when paired with the source sentence—this is verified using the parser. The model follows the standard Transformer encoder-decoder architecture as implemented in `fairseq`, without architectural modifications beyond the configuration choices detailed below.

While promising results would demonstrate the transformer's translation capabilities, they may not fully reveal its generalization ability. To address this, a second and more complex experiment is described in Section 4.

3.1 Dataset Preparation

The dataset is derived from grammar G and includes only positive examples. It consists of approximately 107,000 source sentences in the training set, with approximately 100,000 additional sentences split between validation (40%) and test (60%) sets. Each source sentence is paired with a corresponding target sentence, aligned line-by-line in separate source (.src) and target (.tgt) files to comply with `fairseq` framework requirements. The dataset excludes sentences longer than 14 tokens in the test set to account for the $O(n^6)$ time complexity of the synchronous parser used for verification. Tokenization is performed without Byte-Pair Encoding, as the synthetic language tokens do not require substring tokenization. The `fairseq-preprocess` phase automatically constructs the model's vocabulary during preprocessing.

3.2 Model Configurations and Training

Five model configurations are tested, created using the `fairseq` framework. Each model consists of an encoder and decoder, with the first three configurations featuring 1, 3, and 6 layers respectively. The final two configurations, referred to as "6-layer-v1" and "6-layer-v2", also have 6 layers but with reduced embedding and feed-forward network dimensions to maintain comparable parameter counts to the 1-layer and 3-layer models.

All models share consistent hyperparameters during training:

- **Embedding dimension**: 512
- **Attention heads**: 8
- **Dropout**: 0.3
- **Optimizer**: Adam with learning rate 1×10^{-3} and inverse square root scheduling with 4000-step warmup
- **Loss**: Label-smoothed cross-entropy with smoothing factor 0.1
- **Embedding sharing**: Input and output embeddings shared

During inference, the `fairseq-generate` command is used with a batch size of 128 and beam search width of 5. Performance is evaluated using BLEU score and valid parse ratio, computed by parsing generated sentences against grammar G

using the synchronous parser. These metrics reveal whether the model produces grammatically valid translations according to the synthetic language definition.

3.3 Results

Table 1 summarizes the best-performing checkpoint for each configuration, selected based on the number of valid parses on a fixed subset of 1000 test sentences.

Config.	Ckpt	BLEU	Valid	Diff.
1-layer	11	0.4227	63.6%	68.7%
3-layer	15	0.4327	**73.3%**	**72.4%**
6-layer	13	0.3408	39.6%	55.6%
6-layer-v1	12	0.4016	59.9%	68.3%
6-layer-v2	13	0.3733	52.9%	59.6%

Table 1: Translation results across the five Seq2Seq configurations (best checkpoint per model, evaluated on 1000 test sentences). "Diff." is the fraction of valid parses that differ from the gold target sentence.

Other graphs are included in Appendix B.1 showing validation performance over training time for each configuration.

4 Second Experiment: Acceptor

This experiment investigates whether a Transformer can classify a pair of input sentences as licensed by an SCFG (Chiang, 2006). The Acceptor Transformer is an Encoder-Only Transformer designed for binary classification. It takes as input a pair of sentences from source and target languages and outputs a special token: "[Y]" (accepted) if the pair conforms to the target grammar, or "[n]" (rejected) otherwise. Successful classification of sentence pairs as within or outside the SCFG's language would indicate the Transformer's ability to generalize beyond memorization of training data. Such capability would suggest an understanding of the grammar's complex structure, characterized by recursion and potential ambiguity. We trained the Acceptor Transformer on pairs of sentences on length up to 14 tokens, then evaluated its performance on pairs of lengths up to 100 tokens, to assess Transformer's ability to generalize. This experiment is more powerful than translation alone as evidence of grammar learning, since it directly tests whether the model recognizes the grammar's boundaries, especially on fully unfamiliar sentence pairs (longer than training ones).

4.1 Dataset Preparation

We construct balanced datasets of labeled sentence pairs (w, w'), with 50% accepted examples ([Y]) and 50% rejected examples ([n]). For each dataset, we generate 200k positive pairs from the relevant grammar and 200k negative pairs, validate labels with a deterministic Sync-CYK parser, and split the resulting data into train/validation/test (56/22/22). Training and validation contain only short pairs (length 2–14), while the test set contains longer pairs (length 15–100) to directly probe length generalization. We consider three dataset variants: G**+rand** (negatives are random strings over the same vocabulary and length distribution as G outputs), G+G_6 (negatives are generated by the perturbed grammar G_6, producing hard near-miss examples with the same vocabulary), and G_{14} + **rand**$_{14}$ (same construction as G+rand but using the more complex grammar G_{14}).

4.2 Model Setup and Training

We use an Encoder-Only Transformer as a binary classifier over concatenated sentence pairs, using a unified vocabulary and special tokens <CLS>, <SEP>, and <LABEL> to structure the input and predict "[Y]"/"[n]". We perform an extensive hyperparameter sweep over attention heads (8, 16, 32), layers (1, 2, 3, 5), model dimension (32, 64, 128, 256, 384, 512), and training duration (1, 3, 5, 8, 12, 20 epochs): 432 possible combinations, of which ∼200 were evaluated (from *275k* up to *13M* parameters; largest tested: layers=5, dim=512, heads=32). Each configuration is trained separately on each dataset with cross-entropy loss, Adam (lr = 1e-3) with cosine-annealing scheduling, batch size 256, and gradient clipping (norm 1.0); varying the epoch budget corresponds to distinct training runs rather than checkpoints from a single long run.

For clarity, we use the following shorthand for hyperparameters in all tables:

- **d**: embedding dimension
- **h**: number of attention heads
- **l**: number of layers
- **e**: number of training epochs

4.3 Performance Extremes and In-Depth Analysis

We noticed how poor configurations lead to 3 distinct behaviors: (1) the model just predicts one class and so performs 50%, (2) the model guesses randomly and so performs around 50%, or (3) the

model overfits the training set, achieving high training accuracy but ∼50-60% test accuracy. An example of such a poor configuration is shown in Table 2.

Metric	G+G_6	G+**random**
Training accuracy	97.38%	97.32%
Validation accuracy	86.23%	91.48%
Test accuracy	49.86%	50.16%
'Y' labels	1.16%	0.37%
'n' labels	98.57%	99.95%

Table 2: Performance results for a shallow and narrow configuration d=64, h=16, l=3, e=5.

Metric	G+G_6	G+**random**
Training accuracy	98.12%	99.12%
Validation accuracy	97.74%	98.75%
Test accuracy	84.73%	96.70%
'Y' labels	87.84%	99.74%
'n' labels	81.62%	93.67%

Table 3: Performance results for the best configuration d=384, h=16, l=3, e=12.

To summarize the trends discussed in the in-depth configuration study, Table 4 reports test accuracies for the highlighted model settings across the three datasets (G+rand, G+G_6, and G_{14} + rand_{14}).

Config.	G+**rand**	G+G_6	G_{14}+$\textbf{rand}_{14}$
d=256, h=16, l=3, e=12	93.1%	89.3%	71.1%
d=128, h=16, l=5, e=12	95.4%	95.1%	64.1%
d=384, h=16, l=3, e=12	84.7%	**96.7%**	67.9%
d=256, h=16, l=3, e=12	94.3%	81.8%	53.1%

Table 4: Test accuracy ranges for the configurations analyzed in the performance-extremes and in-depth model analysis.

4.4 Future Work

Several directions merit future investigation:

Exploration of Theoretically Validated Grammars: Investigate SCFGs with stronger theoretical properties and increased complexity in terms of terminals and production rules, moving toward structures more representative of natural language phenomena.

Improved Loss Functions: Develop loss functions that leverage the Synchronous CYK Parser to recognize valid translations beyond single gold targets, accounting for language ambiguity. Reinforcement Learning approaches could help models learn structural constraints more effectively.

Length Generalization Analysis: Systematically explore the factors influencing the "breaking point" where generalization performance degrades. Plotting accuracy against sequence length across configurations could reveal patterns predictive of generalization bounds.

Scaling Laws: Conduct extensive experiments to derive scaling laws quantifying relationships between model parameters (attention heads, layers, dimensionality), dataset size, and task performance across diverse grammars.

5 Conclusions

We empirically examined the learnability of certain SCFGs by Transformers. The sequence-to-sequence Transformer demonstrated the ability to learn the general structure of the grammar, achieving valid parse ratios up to 73.3%, despite the vast number of possible recursive syntactic structures. More significantly, the Encoder-Only Transformer learned to identify sequences licensed by the tested SCFGs with accuracy up to 96.70%, opening to a practical alternative to standard parsing. Additionally, analysis of length generalization (see **??**) shows that trained models can generalize to sequences up to five times longer than the maximum training length, reaching sequence lengths of 70 tokens when trained on data of length up to 14.

Limitations

This study has several limitations. The grammars tested use a limited set of terminals (far fewer than natural languages) and a small number of production rules (20–30, contrasting with thousands required for natural language). While these grammars exhibit high recursion, their simplicity might be limiting generalizability. Natural languages have more complex alignments, such as non-1:1 token alignments and ϵ-productions, not captured here. Additionally, the deterministic Synchronous CYK Parser has $O(n^6)$ complexity, severely limiting dataset size and sentence length. Generating longer sentences (e.g., 200 tokens) or scaling datasets to millions of examples would require prohibitive computational resources. Model performance exhibits variability on test data, particularly for length generalization. While some configurations generalize to sequences five times longer than training data, others degrade significantly beyond 2.5 times the maximum training length. The specific factors determining this breaking point require further investigation.

References

David Chiang. 2006. An introduction to synchronous grammars. Technical report, University of Notre Dame.

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszok, and Neil Houlsby. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*. Presented at ICLR 2021.

Daniel Gildea and Giorgio Satta. 2016. Synchronous context-free grammars and optimal parsing strategies. *Computational Linguistics*, 42(2):207–243.

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

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Frederic P. Miller, Agnes F. Vandome, and John McBrewster. 2009. *Levenshtein Distance: Information theory, Computer science, String (computer science), String metric, Damerau?Levenshtein distance, Spell checker, Hamming distance*. Alpha Press.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Martin Popel, Marketa Tomkova, Jakub Tomek, Łukasz Kaiser, Jakob Uszok, Ondřej Bojar, and Zdeněk Žabokrtské. 2020. Human translation quality is achievable with deep learning. *Nature Communications*, 11(1):4296.

Ehud Reiter. 2018. A structured review of the validity of bleu. *Computational Linguistics*, 44(3):393–401.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30.

A SCFGs definition

The grammar, denoted as G, is used to generate the positive sentence pairs. Two additional grammars, denoted as G' and G_{rand} – called "anti-grammars" –, are employed to create negative examples by generating incorrect target sentences for given source sentences. The grammar G_{14} was later defined, and is a variant of G with more non-terminal rules, used to generate longer sentences for testing generalization capabilities. Its negative examples are generated using $G_{rand_{14}}$.

G_{rand} and $G_{rand_{14}}$ are not explicitly defined here, as they are generated through a random process that ensures they do not conform to the structure of G and G_{14} respectively, while maintaining the same terminal vocabulary and similar statistical properties. The explicit definitions of G, G', and G_{14} are as follows:

1. G: (manually created)

```
S -> A{1} B{2} // B{2} A{1}
A -> A{1} B{2} // A{1} B{2}
A -> C{1} F{2} // C{1} F{2}
B -> B{1} F{2} // F{2} B{1}
B -> D{1} A{2} // D{1} A{2}
C -> C{1} D{2} // D{2} C{1}
C -> F{1} B{2} // B{2} F{1}
D -> F{1} A{2} // F{1} A{2}
D -> D{1} C{2} // D{1} C{2}
F -> D{1} B{2} // B{2} D{1}
F -> F{1} C{2} // C{2} F{1}
A -> a // e
A -> b // f
A -> a // g
B -> b // f
B -> a // e
C -> c // g
C -> d // f
D -> d // h
D -> c // g
F -> c // g
F -> a // h
```

2. G' ("anti-grammar"):

```
S -> A{1} B{2} // B{2} A{1}
A -> F{1} E{2} // F{1} E{2}
A -> C{1} F{2} // C{1} F{2}
B -> A{1} F{2} // F{2} A{1}
B -> D{1} A{2} // D{1} A{2}
C -> D{1} D{2} // D{2} D{1}
C -> B{1} B{2} // B{2} B{1}
D -> F{1} A{2} // F{1} A{2}
```

```
D -> S{1} C{2} // S{1} C{2}
F -> D{1} B{2} // B{2} D{1}
F -> F{1} C{2} // C{2} F{1}
E -> D{1} C{2} // D{1} C{2}
E -> F{1} A{2} // A{2} F{1}
A -> d // e
A -> c // f
A -> b // g
B -> a // f
B -> c // e
C -> d // g
C -> d // h
D -> c // h
D -> c // g
F -> a // g
F -> b // e
E -> b // h
```

3. G_{14}: (generated by a Grammar Generator)

```
S -> N{1} Y{2} // N{1} Y{2}
Y -> A{1} R{2} // R{2} A{1}
N -> D{1} N{2} // N{2} D{1}
R -> S{1} A{2} // S{1} A{2}
D -> N{1} R{2} // N{1} R{2}
N -> D{1} R{2} // R{2} D{1}
D -> R{1} N{2} // N{2} R{1}
R -> Y{1} D{2} // Y{1} D{2}
N -> A{1} N{2} // N{2} A{1}
A -> Y{1} N{2} // Y{1} N{2}
N -> R{1} A{2} // R{1} A{2}
A -> R{1} R{2} // R{1} R{2}
Y -> R{1} Y{2} // Y{2} R{1}
D -> A{1} R{2} // R{2} A{1}
Y -> k // m
R -> k // s
A -> d // m
Y -> d // t
D -> l // p
D -> i // s
A -> i // v
Y -> k // s
A -> h // u
Y -> l // v
```

B Seq2Seq Experiment Details

B.1 Translation: analyses by sentence length

This section provides a length-based view of translation behavior for the best checkpoint of each Seq2Seq configuration. We report (i) the accuracy of predicting the target length (Figure 1), (ii) the percentage of outputs that yield a valid synchronous parse under grammar G when paired with their sources (Figure 2), and (iii) the proportion of those valid outputs that nevertheless differ from the single gold target (Figure 3), reflecting ambiguity in the grammar and the evaluation setup.

Metrics and evaluation. To complement the length-based plots, we also measure similarity between model outputs and gold targets at the string level. We use (a) *token differences*, the number of position-wise token mismatches between prediction and target (a Hamming-style count), and (b) *Levenshtein distance*, the minimum number of single-token insertions, deletions, or substitutions needed to transform the prediction into the target (Miller et al., 2009). Figure 4 illustrates these metrics for representative best checkpoints from the 1-layer and 6-layer configurations.

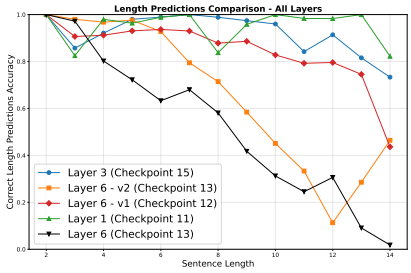


Figure 1: Accuracy of correct length predictions per sentence length for the best checkpoint of each configuration.

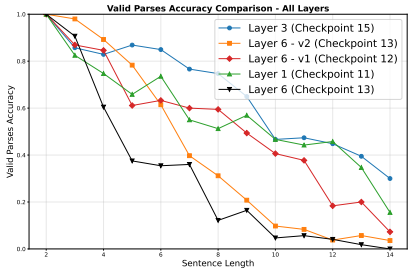


Figure 2: Percentage of valid parses per sentence length for the best checkpoint of each configuration.

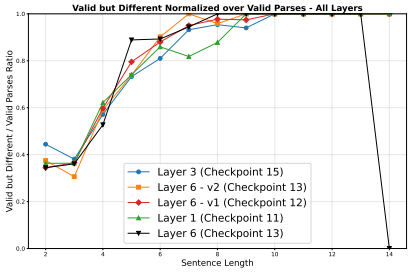
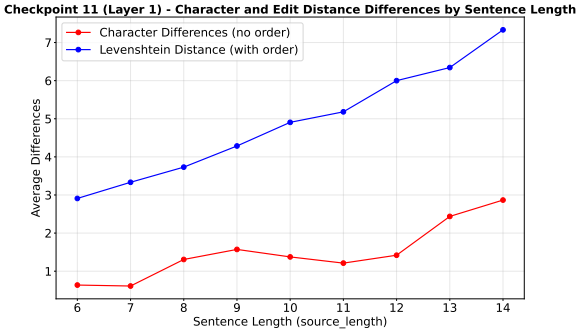
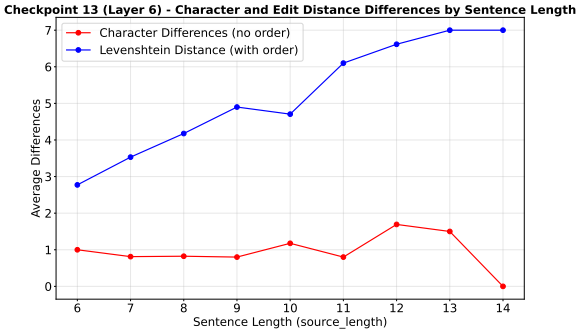


Figure 3: Valid parses different from the gold target per sentence length, normalized over the total number of valid parses (best checkpoint per configuration).



(a) Checkpoint 11 (1-layer model).



(b) Checkpoint 13 (6-layer model).

Figure 4: Token-difference and edit-distance curves (prediction vs. gold target) for representative best checkpoints.

Conclusion. Across configurations, length prediction can remain relatively stable even when grammatical validity degrades with sentence length (Figures 1 and 2). Many invalid outputs are near-misses that differ from valid translations by small local reorderings or substitutions, which can keep string-overlap metrics relatively high despite violating the SCFG constraints. Conversely, a substantial fraction of outputs are valid parses but differ from the single gold target (Figure 3), so BLEU-style overlap should be interpreted cautiously in this setting (Reiter, 2018).

C Dataset pairs

Dataset pairs follow this structure:

`<CLS> (src) <SEP> (tgt) <LABEL> ([Y]/[n])`

Where (src) is a string $\in$ the source language described by G and (tgt) is a string $\in$ the target language described by G. <CLS>, <SEP>, <LABEL> are special tokens and "[Y]" or "[n]" represent the final labels: accepted or rejected.

D Acceptor Experiment Details

D.1 Distribution of the sentences

The distribution of the sentences, sorted by their length, is reported in Figure 5:

- **Blue bins**: G+rand dataset.
- **Red bins**: G+G_6 dataset.
- **Green bins**: G_{14} + rand$_{14}$ dataset.

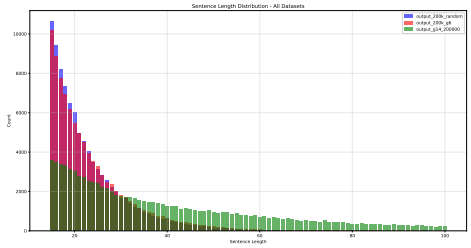


Figure 5: Distribution of sentences by length for positive and negative examples combined. 50% are positive and 50% are negative.

D.2 Dataset Construction Details

The experiment utilizes datasets with 50% positive and 50% negative examples. The datasets are designed to evaluate the model's ability to generalize under different conditions:

- **Training on Short Sentences**: Sentences of length 2–14 (approximately 214,000 examples for training after parsing).
- **Validation on Similar Data**: Short sentences (length 2–14, matching training distribution) to evaluate performance on familiar sentence lengths.
- **Testing on Complex Data**: Longer, entirely unseen sentences (length 15–100) to assess generalization to more complex structures.

Two different dataset combinations are tested:

Dataset G + G_6

The first dataset pairs positive examples from grammar G with negative examples from a modified variant grammar G_6, designed to be structurally

similar to G but with deliberate deviations. The two grammars share the same terminal vocabulary and approximately 30 total productions, with all non-terminals capable of producing both unary and binary productions. The G_6 grammar introduces subtle changes such as terminal symbol substitutions and rule reorderings. Sentences are processed through the synchronous parser to remove any pairs that could belong to both grammars (false ambiguities), ensuring a clean binary classification task. This dataset tests the model's ability to distinguish a highly plausible grammar with subtle deviations from valid constructions.

Dataset G + random

The second dataset pairs positive examples from grammar G with randomly generated negative examples. The random generator uses the same terminal vocabulary as G and mimics its terminal selection probabilities, but places terminals completely randomly with no structured tree derivation. All strings are verified through synchronous parsing to remove rare false positives (which occur with exponentially decreasing probability for longer strings). This dataset provides a more robust evaluation of the model's ability to learn the grammar's structural constraints, rather than simply rejecting noise. A similar dataset was constructed using the more complex grammar G_{14}, which features more productions, terminals, and recursion depth.

Dataset G_{14} + random$_{14}$

An additional dataset is created using the more complex grammar G_{14}, which has increased productions, terminals, and recursion depth. This dataset pairs positive examples from G_{14} with randomly generated negative examples, following the same principles as the G + random dataset. This allows for testing the model's ability to learn and generalize from a more complex grammatical structure.

E Acceptor Variation and Random Analyses

The Acceptor experiments exhibit non-trivial sensitivity to both *configuration variation* (hyperparameter choices) and *randomness* (e.g., random weight initialization and data-order effects). To assess robustness, we re-trained selected configurations with multiple random seeds and report the resulting accuracy-by-length curves. This helps distinguish consistent trends from seed-specific outcomes, especially for length generalization beyond the training regime.

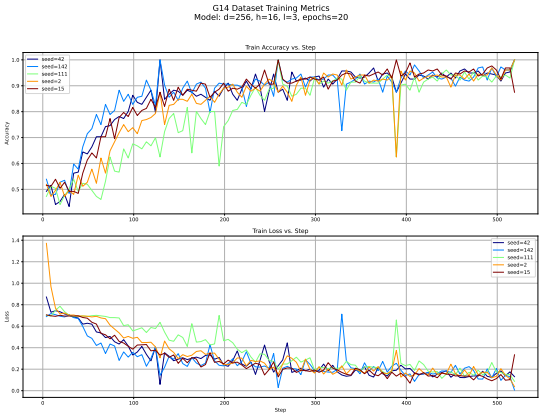
Seed-induced variability is typically limited on the training set and often modest on validation, but it can become substantially larger on the test set. In our analyses, different random seeds sometimes lead to markedly different length generalization profiles (e.g., one seed remaining accurate up to substantially longer sequences than another), particularly on datasets containing random noise (e.g., G+rand).

To complement the seed-based variance study, we also employ bootstrap resampling to quantify uncertainty in test-set metrics and assess whether observed differences between two models are statistically meaningful. Following Koehn (Koehn, 2004), we resample test examples with replacement over 10,000 iterations using the models' stored predictions and compute metric distributions (accuracy, precision, recall, and F1), along with 95% confidence intervals and two-tailed p-values for metric differences. This analysis supports that some performance gaps are non-random, while also highlighting that bootstrap uncertainty does not capture variance due to changing random seeds.

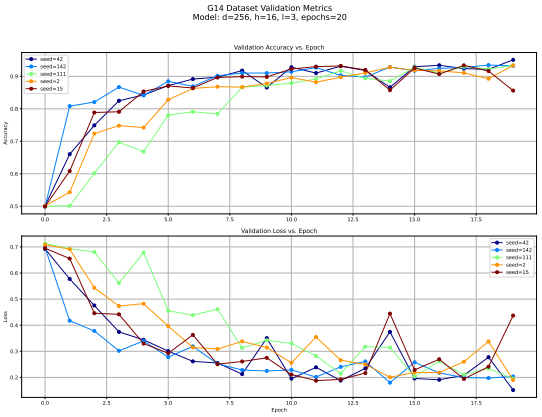
E.1 Learning Curves

We complement the accuracy-by-length analyses with learning curves (train/validation accuracy and loss) for a fixed, larger configuration trained under multiple random seeds. Figure 6 shows that training dynamics are generally stable across seeds: learning progresses similarly and the main peaks and inflection points tend to occur at comparable steps. The most challenging dataset, $G_{14} + \text{rand}_{14}$, exhibits consistently higher variability in training trajectories, reflecting its increased grammatical complexity.

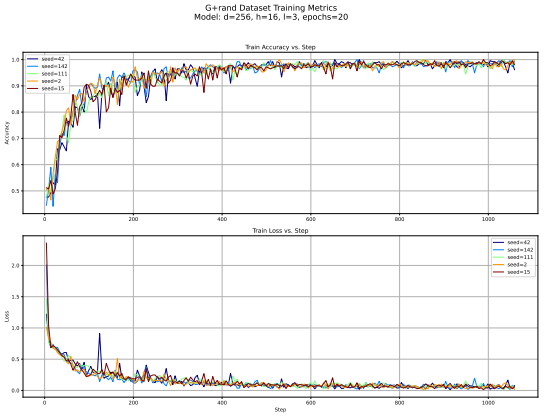
Validation curves in Figure 7 are broadly consistent across seeds, but seed effects become more visible than in training, particularly on datasets that include random noise (e.g., G+rand). This pattern anticipates the larger seed-induced variance observed on test sets: small differences in initialization and training order can translate into noticeably different generalization behavior even when the training fit appears similar.



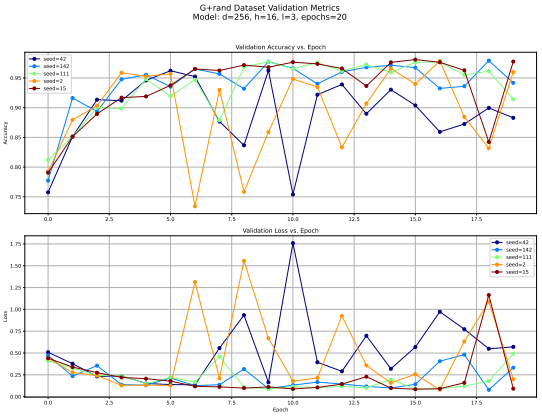
(a) Training metrics for $G_{14} + \text{rand}_{14}$ dataset.



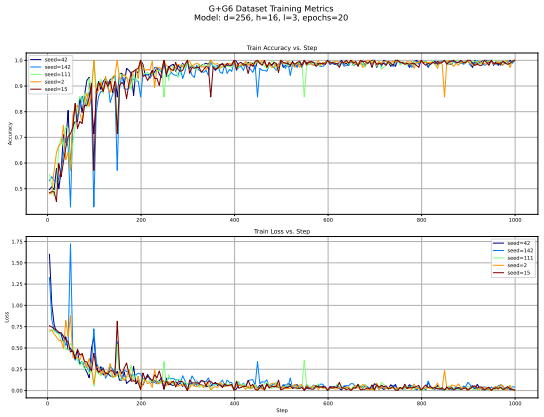
(a) Validation metrics for $G_{14} + \text{rand}_{14}$ dataset.



(b) Training metrics for G+rand dataset.

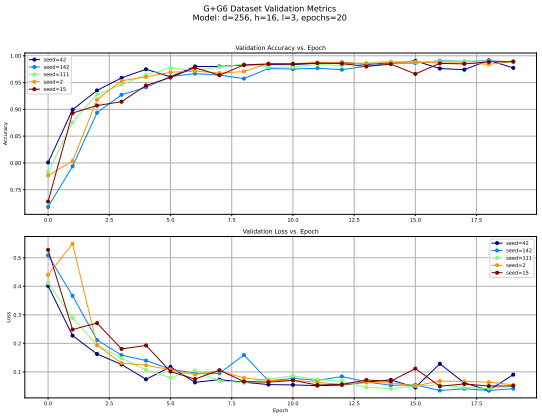


(b) Validation metrics for G+rand dataset.



(c) Training metrics for G+G_6 dataset.

Figure 6: Training learning curves across $G_{14} + \text{rand}_{14}$, G+rand, and G+G_6 datasets.



(c) Validation metrics for G+G_6 dataset.

Figure 7: Validation learning curves across $G_{14} + \text{rand}_{14}$, G+rand, and G+G_6 datasets.

E.2 Acceptor: length-based performance analyses

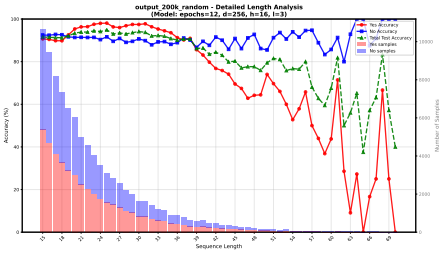
Plot legend (for all accuracy-by-length figures below):

- **X-axis**: sequence length
- **Left Y-axis**: accuracy (%)
- **Right Y-axis**: number of samples
- **Red line**: positive-class accuracy (%)

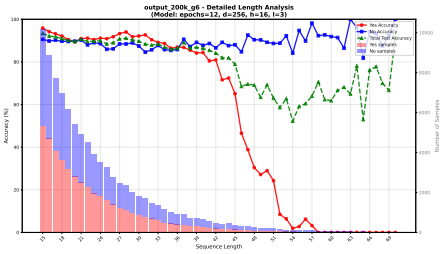
- **Red bars**: number of positive samples
- **Blue line**: negative-class accuracy (%)
- **Blue bars**: number of negative samples
- **Green line**: average accuracy (%)

Brief synthesis of the in-depth model analysis (selected configurations):

- **d=256, h=16, l=3, e=12** (Figures 8 and 9): strong length generalization on G+rand and G+G_6 up to roughly 2.5–3$\times$ the training maximum, with a marked breaking point around length $\sim$40; on G_{14} + rand$_{14}$ performance appears increasingly dominated by class bias at longer lengths (high acceptance accuracy while rejection accuracy drops), and aggregate accuracy can hide failures on rare long sequences. For the next configurations, G_{14} + rand$_{14}$ will not be shown as it is always very close to random performance.
- **d=128, h=16, l=5, e=12** (Figure 10): best average accuracy on the G-based datasets, with a later breaking point on G+rand (around length $\sim$55) and comparatively stable behavior across lengths on G+G_6; this configuration tends to achieve high accuracy on both acceptance and rejection for these two datasets.
- **d=256, h=16, l=3, e=20** (Figure 11): stable accuracy across lengths on G+rand, but markedly worse and length-sensitive behavior on G+G_6, illustrating that extending training can improve one dataset while not transferring reliably to harder near-miss negatives.
- **d=384, h=16, l=3, e=12** (Figure 12): best-performing configuration on the G+G_6 dataset, reaching 96.70% test accuracy; on G + rand it shows a steep initial drop in positive-class accuracy that later recovers, achieving very high raw accuracy without a clear breaking point up to length 70.

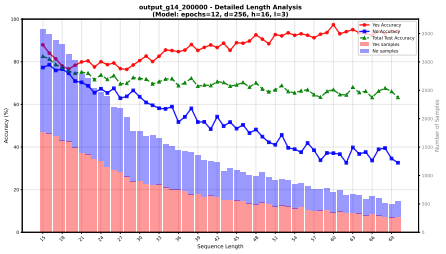


(a) G+rand test set.



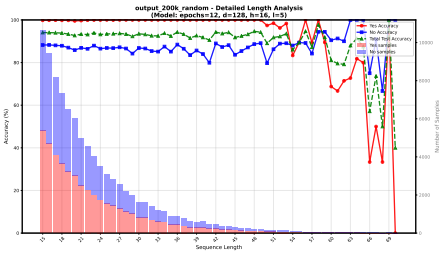
(b) G+G6 test set.

Figure 8: Accuracy-by-length analysis for d=256, h=16, l=3, e=12 on G+rand and G+G6 test sets.

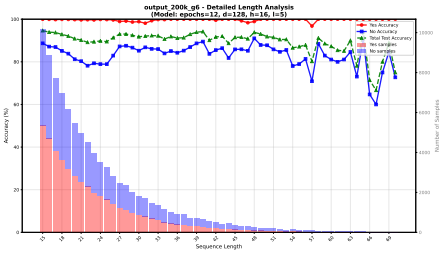


(a) G14+rand14 test set.

Figure 9: Accuracy-by-length analysis for d=256, h=16, l=3, e=12 on G14+rand14 test set.

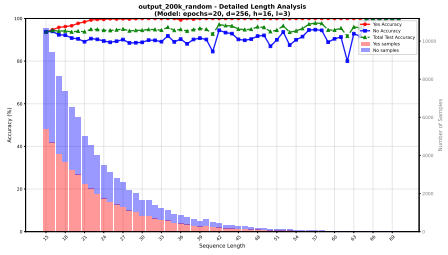


(a) G+rand test set.

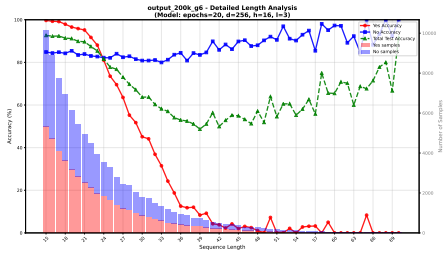


(b) G+G6 test set.

Figure 10: Accuracy-by-length analysis for d=128, h=16, l=5, e=12 on G+rand and G+G6 test sets.

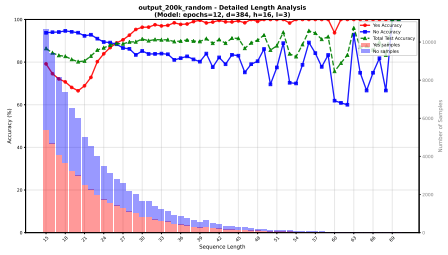


(a) G+rand test set.

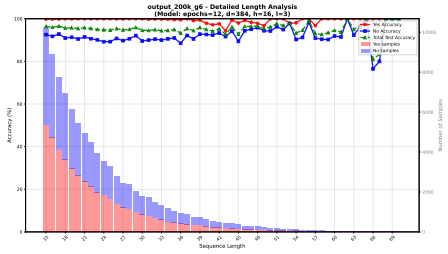


(b) G+G6 test set.

Figure 11: Accuracy-by-length analysis for d=256, h=16, l=3, e=20 on G+rand and G+G6 test sets.



(a) G+rand test set.



(b) G+G6 test set.

Figure 12: Accuracy-by-length analysis for d=384, h=16, l=3, e=12 on G+rand and G+G6 test sets.