

Synthetic Pre-Pre-Training Improves Language Model Robustness to Noisy Pre-Training Data

Xu Guo¹²³ Runyu Peng³ Jian Tong³ Yunhua Zhou³
Haijun Lv³ Zhihui Lu^{†1} Qipeng Guo^{†23}

¹ Fudan University ² Shanghai Innovation Institute

³ Shanghai AI Laboratory [†] Corresponding authors



NeurIPS 2026 • Sydney

guox24@m.fudan.edu.cn

Motivation

Can a structural prior help a model learn useful patterns despite noise?

Pre-training (PT) teaches a language model to predict the next token.

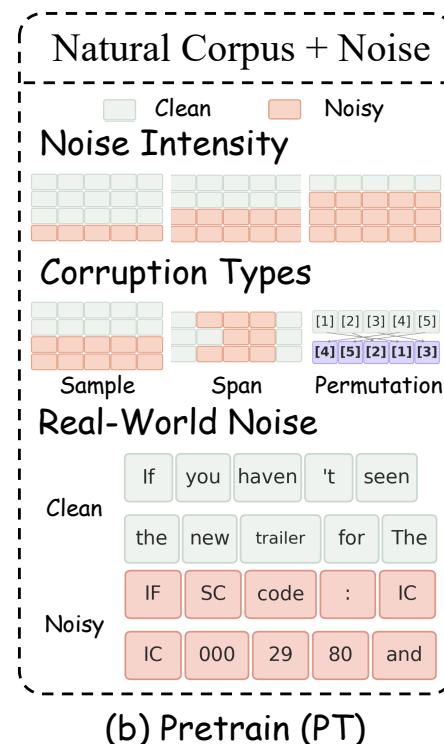
Useful text and noise coexist

Web text includes coherent prose alongside boilerplate, irrelevant fragments, and errors.

Cleaning is imperfect

Too little filtering leaves noise.

Aggressive filtering may remove useful data.

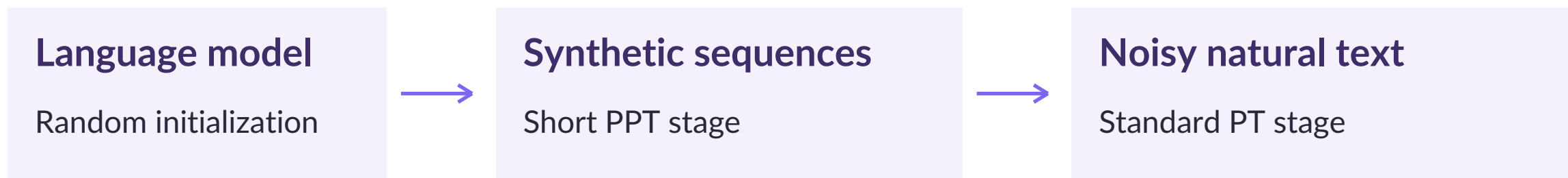


Method

First learn from synthetic sequences, then pre-train on noisy natural text.

Pre-pre-training (PPT): a short next-token prediction stage on synthetic data.

With PPT



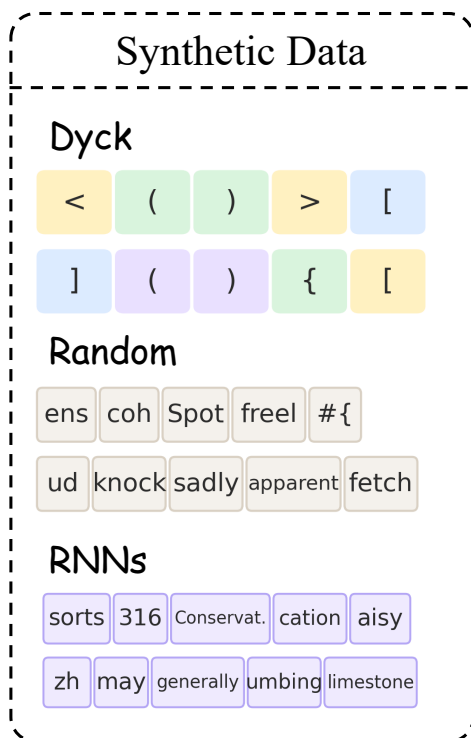
No-PPT baseline

Random initialization

→ Same natural-text PT

Method: Synthetic data sources

All three sources train the same language model with the same PPT budget.



(a) Pre-pretrain (PPT)

Random tokens

Independent samples from the vocabulary.

A warm-up with no sequential structure.

Dyck bracket sequences

k-Shuffle Dyck: interleaved balanced bracket streams.

A formal-language source with structured dependencies.

Recurrent neural network (RNN) sequences

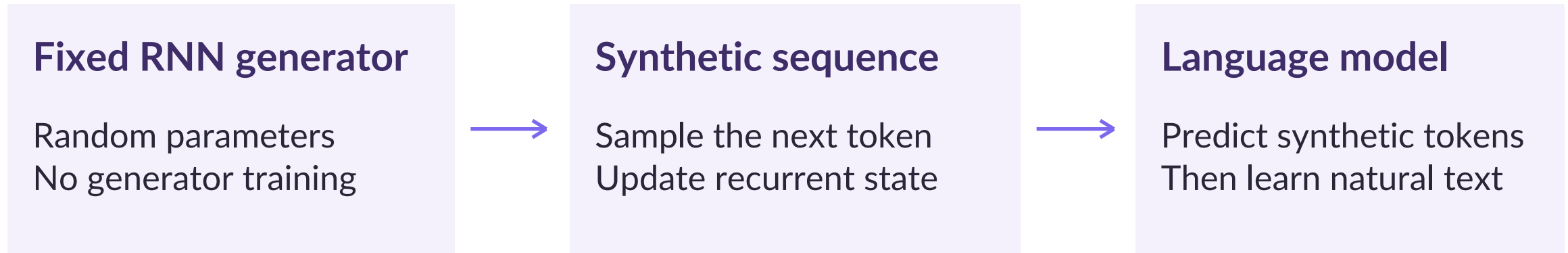
Fixed random generators produce temporal dependencies.

PPT aims to turn this structure into a useful prior.

Method: RNN sequence generation

The RNN generators stay fixed; only the language model is trained.

A recurrent state makes each generated token depend on the sequence history.



Previous token + recurrent state

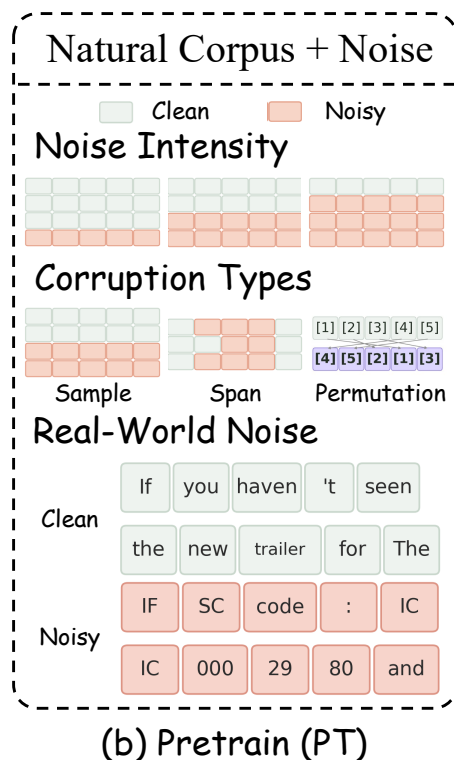
Next token

Dependencies can be learned even when the sequence does not read like natural text.

Experiments: Controlled-noise evaluation

We train on corrupted C4 sequences and evaluate on clean held-out text.

Main experiments use a 160M-parameter Pythia language model and C4 web text.



Training noise is controlled

Replace a fraction of training sequences with random-token sequences.

50% noise means half the sequences are replaced.

Evaluation remains clean

Predict held-out, uncorrupted C4 text.

Lower validation loss means better prediction.

Compare models at the same natural-text PT budget.

Experiments: Convergence and token efficiency

RNN-PPT reaches the same loss sooner and improves more than Dyck-PPT.

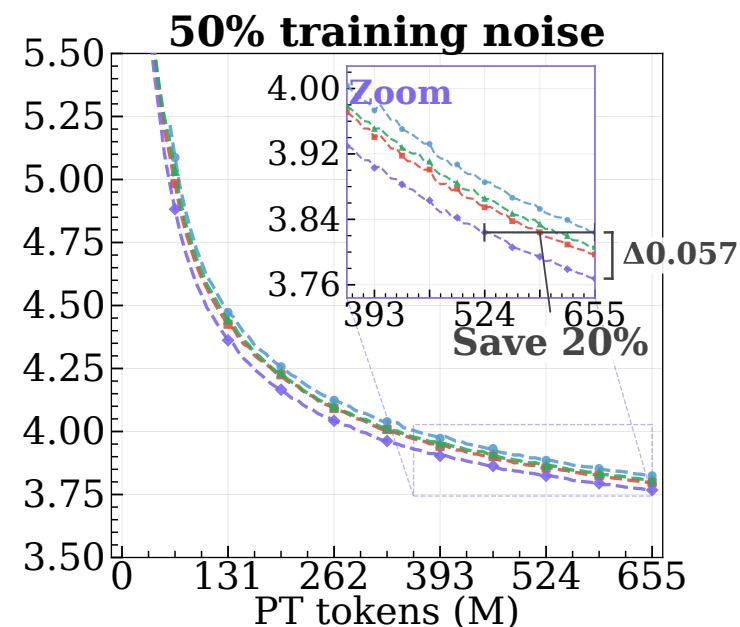
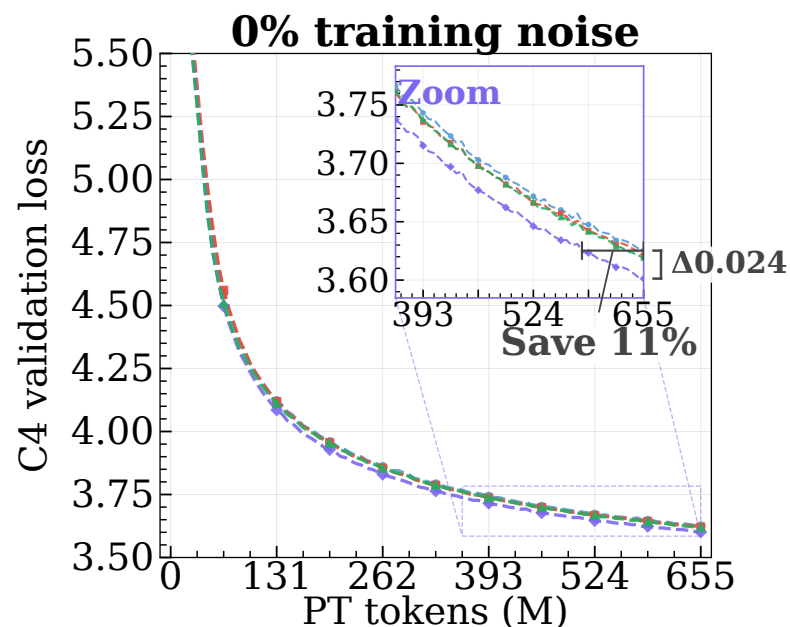
Pythia-160M; clean held-out C4 validation; mean curves over three seeds.

--- No-PPT baseline

--- Random-PPT

--- Dyck-PPT

--- RNN-PPT

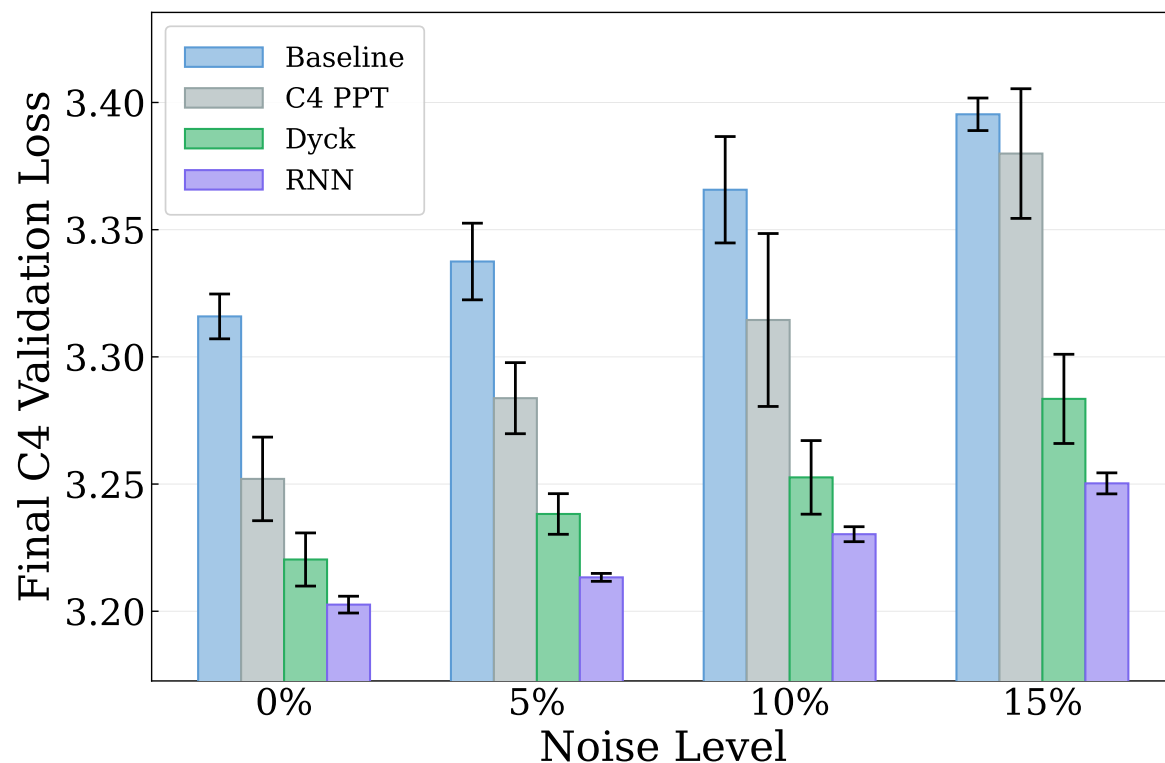


Random-token PPT shows no benefit.

Experiments: Larger model: Pythia-1B

RNN-PPT has the lowest final loss across all tested 1B noise levels.

C4-text warm-up uses the same synthetic-stage token budget as RNN-PPT.



Bars: final clean C4 validation loss; mean $\pm$ standard deviation, three seeds.

The source still matters

RNN-PPT beats Dyck-PPT and an equally long warm-up on clean C4 text.

Across noise levels

RNN-PPT has the lowest final validation loss at all four tested noise levels.

Experiments: Chinchilla-style training budget

RNN-PPT retains an advantage at the Chinchilla-style optimal TPP.

Pythia-160M; extended PT: 49K steps, 3.21B tokens, ≈ 20 tokens per parameter (TPP).

Final clean C4 validation loss; mean $\pm$ standard deviation, three seeds.

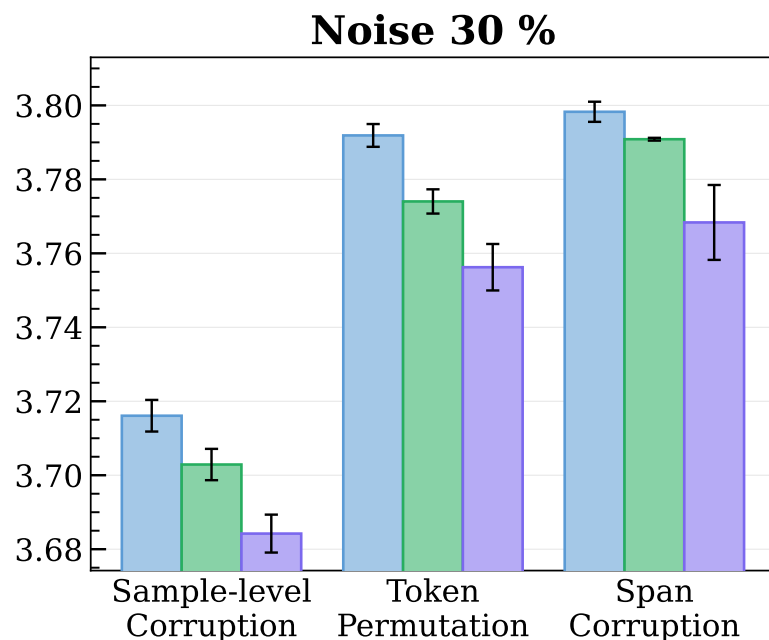
PT noise	No-PPT	RNN-PPT	Loss reduction
0%	3.3935 $\pm$ 0.0010	3.3769 $\pm$ 0.0019	0.0166 $\pm$ 0.0025
10%	3.4153 $\pm$ 0.0008	3.3987 $\pm$ 0.0039	0.0167 $\pm$ 0.0046
30%	3.4556 $\pm$ 0.0033	3.4397 $\pm$ 0.0018	0.0159 $\pm$ 0.0023
50%	3.5168 $\pm$ 0.0021	3.4954 $\pm$ 0.0007	0.0214 $\pm$ 0.0026

Experiments: Generalization across corruption types

RNN-PPT improves loss under all three tested corruption types.

Pythia-160M; 30% training noise; clean C4 validation; mean $\pm$ SD, three seeds.

— No-PPT — Dyck-PPT — RNN-PPT



Y-axis: final clean C4 validation loss.

Sample-level corruption

Replace entire training sequences with random vocabulary tokens.

Token permutation

Shuffle token order within selected local windows.

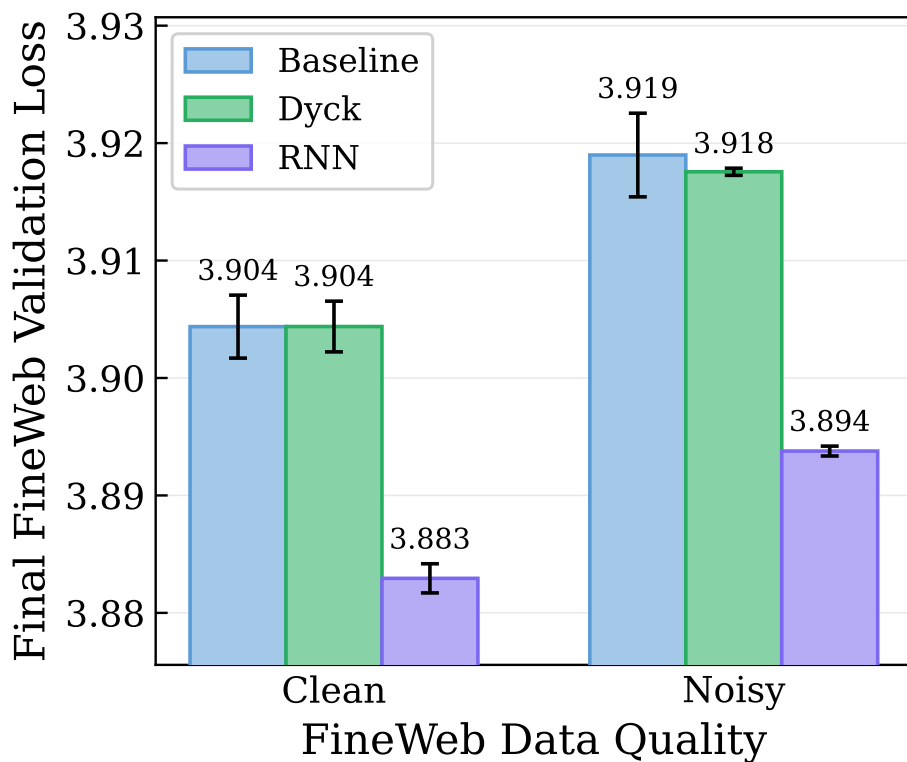
Span corruption

Replace contiguous token spans with random vocabulary tokens.

Experiments: Naturally noisy FineWeb data

RNN-PPT helps on naturally occurring web noise, beyond injected corruption.

We pre-train on cleaner and noisier subsets of FineWeb, a real web-text corpus.



Natural quality differences

A fixed reference model separates cleaner and noisier documents. Human audits verify the contrast.

Matched training budget

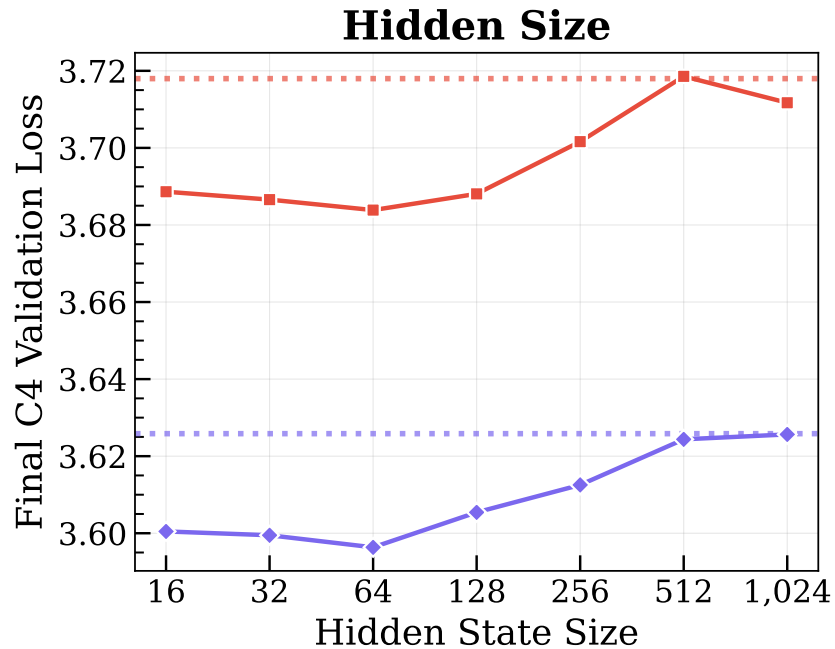
Train separately on both subsets with the same PT token budget. Compare final FineWeb validation loss.

Ablation study: RNN generator size

Moderate generator sizes transfer best; very large generators lose most of the benefit.

RNN hidden size is the dimension of the generator's internal recurrent memory.

Purple: clean PT. Red: 30% noise. Dotted lines: No-PPT baselines.



Moderate size transfers best

Hidden size 64 is strongest in the displayed 30% noise setting.

Very large generators lose most of the transfer benefit.

Analysis: Local-statistics source controls

The results support a role for learnable dependencies beyond local n-grams.

Sample new sequences from unigram, bigram, or trigram models fitted to RNN tokens.

Final C4 loss reduction versus No-PPT; mean over three seeds. Positive is better.

PPT source	0% noise	10%	30%	50%
Unigram matched	0.006	0.006	0.008	0.014
Bigram matched	-0.002	0.002	-0.011	-0.001
Trigram matched	0.002	0.003	0.004	0.005
RNN subset	0.026	0.029	0.036	0.052

The RNN-subset row is the matched source used to construct these local-statistics controls.

Analysis: Noise self-modeling probe

Independently corrupt held-out C4 text and record which token positions are noisy.

Noise self-modeling: noisy positions attend to other noisy positions.

$$r_{\text{noise}} = \frac{1}{|Q_{\text{noise}}|} \sum_{q \in Q_{\text{noise}}} \sum_{k \in K_{\text{noise}}(q)} a_{qk}$$

Q: noisy queries; K(q): noisy context positions; a: attention weight from q to k.

What is measured?

For each noisy query, sum attention on noisy context.

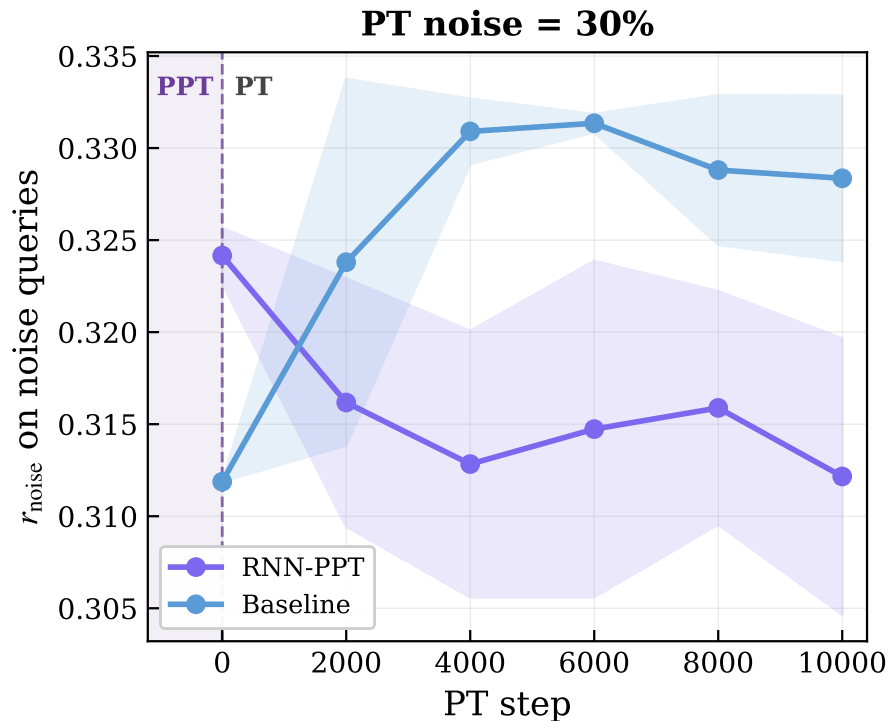
Average across noisy queries, then across layers and heads.

Lower values mean weaker noise self-modeling.

Analysis: Attention dynamics during pre-training

PPT changes the later learning trajectory toward less noise self-modeling.

Compare RNN-PPT and No-PPT at successive checkpoints; PT training noise is 30%.



A gap forms during PT

PPT alone does not suppress this attention immediately.

The gap forms during noisy PT and persists over training.

Mean $\pm$ standard deviation over three seeds; independent held-out C4 probe.

Analysis: Attention intervention at a fixed checkpoint

Pythia-160M; 10K PT steps at 30% training noise; model weights and input stay fixed.

Probe input

Replace 50% of test tokens with noise.
Record the corruption mask.

Prediction target

Loss is computed at unreplaced tokens.
This measures prediction of clean text
from the corrupted input.

Noisy query → noisy context attention

Layers 9–11

Scale post-softmax attention weights.

→ **Multiply by 0.5 or 2**

Then renormalize each attention row

Analysis: Effect of the attention intervention

Halving improves prediction; doubling worsens it across all three methods.

Clean-token prediction loss change relative to the same unmodified checkpoint.

Loss change = after intervention – before intervention.

Negative means better prediction.

Route scale	No-PPT	k-Shuffle Dyck	RNN-PPT
0.5×	−0.0460±0.0081	−0.0112±0.0028	−0.0299±0.0072
2×	+0.0488±0.0107	+0.0121±0.0029	+0.0317±0.0082

Mean ± standard deviation, five seeds; 50% noise in the fixed C4 probe.

The intervention tests this route's effect; it does not apportion the full PPT training gain.

Conclusion

A short synthetic stage improves later learning from noisy natural text.

Code and synthetic data are available for reproducing the training pipeline.

Learnable synthetic structure

RNN-PPT lowers loss and reaches a matched loss with fewer natural-text training tokens.



Paper

NeurIPS official page

Robustness beyond one setting

Gains persist at 1B scale, ≈ 20 TPP, and on naturally noisy FineWeb text.



Code

Training and analysis

A different learning trajectory

RNN-PPT weakens noise self-modeling during PT; reducing this attention route helps prediction.

guox24@m.fudan.edu.cn



Data

Synthetic sequences

Supplementary: Training budgets and evaluation

Budget definitions distinguish synthetic PPT tokens from natural-text PT tokens.

Setting	Pythia-160M	Pythia-1B
PPT steps	500	1,000
Synthetic PPT tokens	32.8M	65.5M
Natural-text PT steps	10,000	25,000
Tokens per step	65,536	65,536

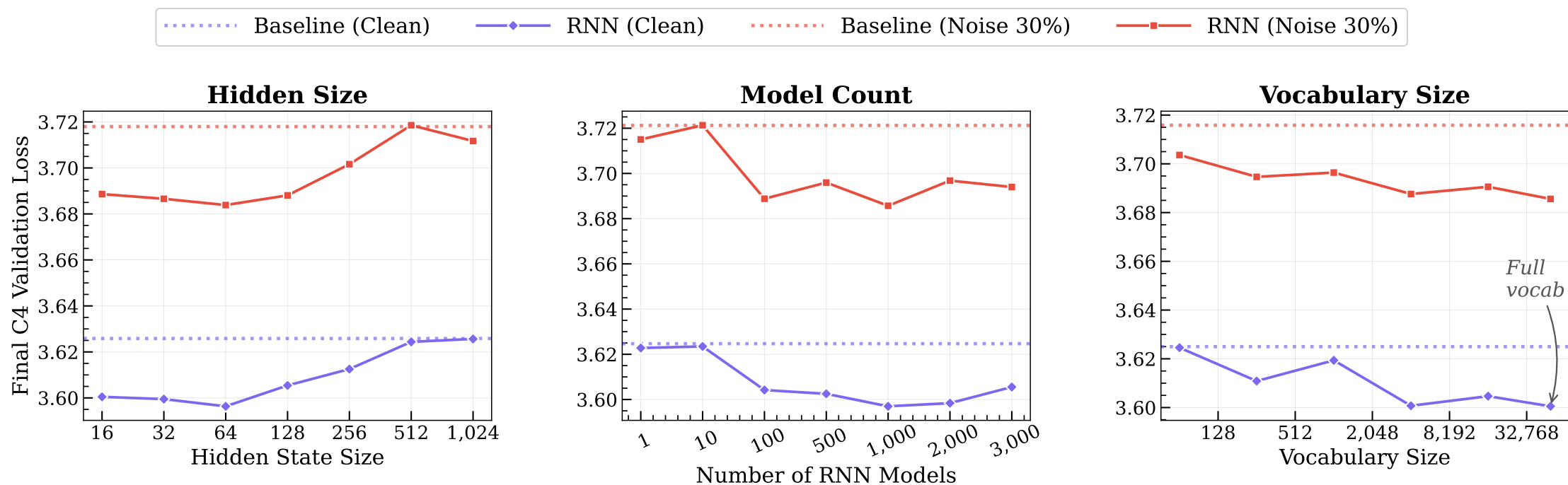
Chinchilla follow-up: 160M, 49K PT steps, about 3.21B natural-text tokens.

Matched-loss savings count PT tokens; extra synthetic tokens are stated separately.

Supplementary: Generator diversity and token support

Transfer favors learnable patterns with diverse generators and broad token support.

Original design sweeps at clean and 30% PT noise; dotted lines are No-PPT baselines.



Default source: 1,000 fixed random RNNs; hidden size 64; full vocabulary; temperature 1.

Supplementary: Head-level attention changes

The strongest reductions are localized in later layers rather than uniform across heads.

Head-level change: RNN-PPT minus No-PPT; blue means weaker noise self-modeling.

Same held-out C4 probe with 50% corrupted token positions across all conditions.

