

A Scrambled Positional Encoding Is Worse Than None

Srinivas Harish

New York University

srinivas.harish@nyu.edu

Abstract

Rotary positional encoding (RoPE) supplies attention with relative-position information. We ask how language modelling changes when its position indices are distorted while the model architecture and training token stream remain fixed. We compare standard RoPE, a monotone map with irregularly spaced indices, permutations of contiguous blocks, a full fixed permutation, and no positional encoding on WikiText-103. After 10M training tokens, the fixed permutation leaves a 124M-parameter decoder-only model 11.36 perplexity points worse than no encoding, with a 95% paired t interval of [7.20, 15.52] over five initialization seeds. This permutation preserves the set of rotation angles and is identical during training and evaluation. By comparison, the monotone map has a mean absolute offset error of 1.25 index units over token pairs separated by at most eight positions and recovers 93.4% of the standard encoding’s perplexity improvement over no encoding. Shuffling blocks of eight positions recovers 50.5%, and the permuted-versus-no-encoding comparison has the same sign at 29.9M parameters. In a separate five-seed series with a common 20M-token schedule, the final permutation-minus-no-encoding difference is 4.77 points, with a 95% paired t interval of [2.54, 7.00]. These results identify a cost of a particular fixed scrambling under low-budget training, despite its invertible position assignment. Moderate monotone distortion is tolerated in this setting; invertibility alone does not ensure an effective positional representation at the tested budget.

1 Introduction

Language models use positional encodings to distinguish where tokens occur and how far apart they are. Rotary positional encoding, or RoPE, introduces relative offsets through rotations of attention queries and keys (Su et al., 2024). Context-extension methods such as position interpolation

and YaRN modify the position-to-angle map while retaining ordered indices (Chen et al., 2023; Peng et al., 2024). This leaves a question about the precision and organization of the positional information needed during language-model training.

Removing an encoding does not remove every source of position information from a decoder-only transformer. The causal attention mask restricts which tokens can interact, and models with no positional encoding can learn from this structure (Haviv et al., 2022; Kazemnejad et al., 2023). A comparison with that baseline can therefore distinguish a useful positional signal from one that interferes with learning at the same training budget.

We distort the index supplied to RoPE without changing the architecture, initialization, optimizer, or training batches. A fixed random permutation keeps the same set of position indices and rotation angles but changes their assignment to tokens. Because the map remains identical at training and evaluation, this condition tests learning with a consistent scrambled representation. A second condition uses monotone, irregularly spaced indices. Block permutations retain the original offsets within each block and change relationships across blocks.

At 124M parameters and 10M training tokens, the fully permuted condition has 11.36 higher validation perplexity than the no-encoding condition, with a paired 95% t interval of [7.20, 15.52]. The irregular monotone map retains 93.4% of RoPE’s improvement over no encoding, and a block size of eight retains 50.5%. The position permutation is mathematically invertible, but its finite-budget language-modelling performance is worse than omitting the rotations.

We report paired comparisons at two model sizes, quantify how much of RoPE’s benefit survives each index transformation, and measure learning curves against estimated training computation for the permutation and no-encoding conditions.

2 Positional information

Let q_i and k_j be the unrotated query and key at token positions i and j . For a fixed frequency vector ω , RoPE applies a block-diagonal rotation $R(t)$ whose two-dimensional rotation angles are $t\omega$. Its attention score satisfies

$$(R(i)q_i)^\top (R(j)k_j) = q_i^\top R(j-i)k_j. \quad (1)$$

The explicit positional contribution thus depends on the relative offset; the queries and keys also depend on token content and the preceding layers.

Three properties of the index map can be distinguished. Injectivity assigns a distinct index to every position. Monotonicity preserves the ordering of those indices. Uniform spacing makes their differences proportional to token offsets. These properties concern the indices before rotation: sine and cosine are periodic, and their values are not themselves monotone in the position index.

A fixed permutation preserves injectivity but generally changes both ordering and spacing. A strictly increasing irregular map preserves order while changing spacing. Neither construction changes the causal mask. The experiments compare the resulting inductive biases through validation perplexity, rather than measuring whether internal activations decode the position index.

3 Position-index controls

For a context of length $T = 512$, let ρ map token positions $\{0, \dots, T-1\}$ to the indices used to construct RoPE angles. Each map is drawn once with seed 777 and reused at every training and evaluation step. The same map is used for all initialization seeds.

Preserved quantities. All conditions have the same trainable parameters, optimizer settings and training batches. Every RoPE condition performs the same operations after its rotation table has been precomputed. Full and block permutations preserve the multiset of absolute position angles. They also preserve the multiset of absolute index differences over all distinct pairs, because they relabel a complete set of positions. They change which token pairs receive those differences, including their signs over causal pairs. The monotone condition preserves neither the original index multiset nor the original angle multiset. The no-encoding condition omits the rotations and therefore uses slightly fewer operations.

Offset distortion. For a set S of causal pairs $i > j$, define

$$D(S) = \frac{1}{|S|} \sum_{(i,j) \in S} |\rho(i) - \rho(j) - (i - j)|. \quad (2)$$

We enumerate this quantity over all causal pairs, and separately over pairs with $1 \leq i - j \leq 8$. The latter is the short-range distortion used in the abstract. It is an average absolute error, not a maximum-error guarantee.

Conditions. `rope` uses $\rho(t) = t$. `rope_mono` sorts a random subset of T indices drawn without replacement from $\{0, \dots, 4T-1\}$ and rescales its endpoints to $[0, T-1]$. Its mean absolute causal offset is 169.7, compared with 171.0 for standard RoPE; endpoint rescaling therefore approximately matches the average scale. Its short-range distortion is 1.25 index units.

`rope_blkB` partitions the index sequence into contiguous blocks of size B and permutes the blocks, preserving order inside each block. We use $B \in \{128, 32, 8\}$, with short-range distortions of 4.54, 19.86 and 103.71 units, respectively. Offsets crossing a block boundary can change even when tokens are fewer than B positions apart. `rope_perm` uses a full random permutation, with short-range distortion 168.34 units. `nope` omits RoPE and retains the causal mask.

4 Experimental setup

Model and corpus. The main model is a decoder-only transformer with 12 layers, 12 attention heads, hidden width 768, context length 512, root mean square normalization (RMSNorm) before each sublayer, Gaussian error linear unit (GELU) feed-forward layers, and tied input/output embeddings. It has 123,551,232 trainable parameters, including 84,953,856 outside the token embedding. We refer to it as the 124M model. The second model has six layers, six heads, width 384, and 29,920,512 parameters. Both use the GPT-2 byte-level byte-pair tokenizer with vocabulary size 50,257. WikiText-103 (Merity et al., 2017) supplies 117,920,428 training tokens and 247,289 validation tokens after tokenization. Training samples random contiguous 512-token contexts with replacement; each context resets its positional indices.

Training and evaluation. For the main comparison, each model processes 9,994,240 tokens over 1,220 updates, which we round to 10M tokens. The

effective batch is 8,192 tokens. The AdamW optimizer with decoupled weight decay uses learning rate 6×10^{-4} , 100 warmup updates, cosine decay to 6×10^{-5} , $(\beta_1, \beta_2) = (0.9, 0.95)$, weight decay 0.1, and gradient norm clipping at 1.0. Dropout is zero. Models run in bfloat16 on an RTX 5090. Validation perplexity is the exponential of mean token cross-entropy over 640 sampled contexts, or 327,680 tokens, drawn with replacement from the validation split using a fixed seed. Lower perplexity is better.

Pairing and uncertainty. All conditions use initialization seeds 0 through 4 and a separate data generator with seed 1234, so corresponding runs start with identical weights and every run sees the same token batches. Validation uses seed 99. Intervals for perplexity differences are two-sided 95% Student t intervals across the five paired seed differences. They describe initialization variability conditional on the shared data order and position map. We define recovered benefit as

$$100 \frac{\bar{P}_{\text{nope}} - \bar{P}_a}{\bar{P}_{\text{nope}} - \bar{P}_{\text{rope}}}, \quad (3)$$

where $\bar{P}_a$ is mean perplexity for condition a . For this ratio of paired means we use 95% Fieller t intervals, retaining numerator–denominator covariance. These intervals assume approximately normal seed-level observations and are marginal, without a multiple-comparison adjustment.

Computation. With L layers, hidden width d , vocabulary size V , context length T , and D processed tokens, we report the reference dense-matrix training estimate

$$\hat{C}(D) = D [6(12Ld^2 + Vd) + 12LTd]. \quad (4)$$

A multiply and an add each count as one floating-point operation (FLOP). The expression includes forward and backward linear projections, the vocabulary projection, and attention matrix products. RoPE adds $12LdD$ multiply/add operations for query/key rotations and their gradients, less than 0.02% here. Changing the position table adds no per-token operations. This is an arithmetic accounting convention, not a measurement of executed GPU operations. Appendix C derives it and specifies excluded operations. At the original 10M-token budget it gives 7.974 petaFLOPs for the 124M model and 1.935 petaFLOPs for the 29.9M model without rotations; one petaFLOP is 10^{15} FLOPs.

5 Results

Fixed permutation versus no encoding. Table 1 reports the 124M-model comparison after 10M training tokens. The fixed permutation yields mean perplexity 196.51, compared with 185.15 for `nope`. Its paired excess is 11.36 points, with a 95% t interval of [7.20, 15.52]. Every seed has the same sign. This effect occurs with one map reused throughout training and evaluation, with unchanged parameter count and rotation-angle multiset relative to standard RoPE.

Monotone and block distortions. The irregular monotone condition has perplexity 130.41 and retains 93.4% of the standard encoding’s improvement over `nope`. Its short-range offset error averages 1.25 index units. Block sizes 128, 32, and 8 retain 98.0%, 75.3%, and 50.5%, respectively. Within this block family, smaller blocks retain less of RoPE’s benefit. Across all conditions, mean distortion alone does not order performance: block128 retains more benefit than the monotone map despite a larger short-range error of 4.54 units. Figure 1 shows these conditions separately and the validation loss across positions in the context window.

The block8 recovery of 50.5% has a 95% Fieller interval of [47.21, 53.59]%. Preserving offsets within eight-token blocks therefore recovers about half of the benefit under this configuration; it does not preserve every offset of length eight.

Perplexity versus training computation. We train a separate paired series from initialization with a common 20M-token cosine schedule. This series compares `rope_perm` and `nope` at 124M parameters, using the same five initialization seeds, fixed permutation, training stream, and validation contexts as above. Two microbatches of eight contexts accumulate into each 8,192-token update. We evaluate after 305, 610, 1,220, and 2,441 updates, corresponding to approximately 2.5M, 5M, 10M, and 20M processed tokens. The final checkpoint contains 19,996,672 tokens. Its estimated training computation is 15.956 petaFLOPs for the permuted condition. The 10M checkpoint lies midway through this longer learning-rate schedule; it is a separate measurement from the original 10M-schedule endpoint in Table 1.

Figure 2 reports all four checkpoints. The paired permutation-minus-no-encoding differences are -0.50 points at 2.5M training tokens, with a 95% interval of $[-2.28, 1.28]$; 2.95 at 5M [1.06,

Table 1: WikiText-103 validation perplexity after 10M training tokens at 124M parameters, five initialization seeds. Lower perplexity is better. Distortion is mean absolute index-offset error over all causal pairs or over pairs separated by at most eight positions. Recovered benefit is the percentage of the mean `nope`-to-`rope` perplexity gap retained, with a 95% Fieller interval across paired seeds. Every encoded condition uses an injective index map. Dashes indicate that `nope` has no index map. Standard deviation (SD) is across seeds.

condition	offset distortion		perplexity		recovered benefit (%)	
	all	short-range	mean	SD	estimate	95% interval
<code>rope</code>	0.00	0.00	126.55	0.35	100.0	reference
<code>rope_blk128</code>	192.38	4.54	127.73	0.45	98.0	[97.33, 98.62]
<code>rope_blk32</code>	248.99	19.86	141.01	0.65	75.3	[73.19, 77.25]
<code>rope_blk8</code>	232.12	103.71	155.53	0.87	50.5	[47.21, 53.59]
<code>rope_mono</code>	2.68	1.25	130.41	0.26	93.4	[92.51, 94.29]
<code>rope_perm</code>	246.80	168.34	196.51	1.95	-19.4	[-27.74, -11.77]
<code>nope</code>	–	–	185.15	2.46	0.0	reference

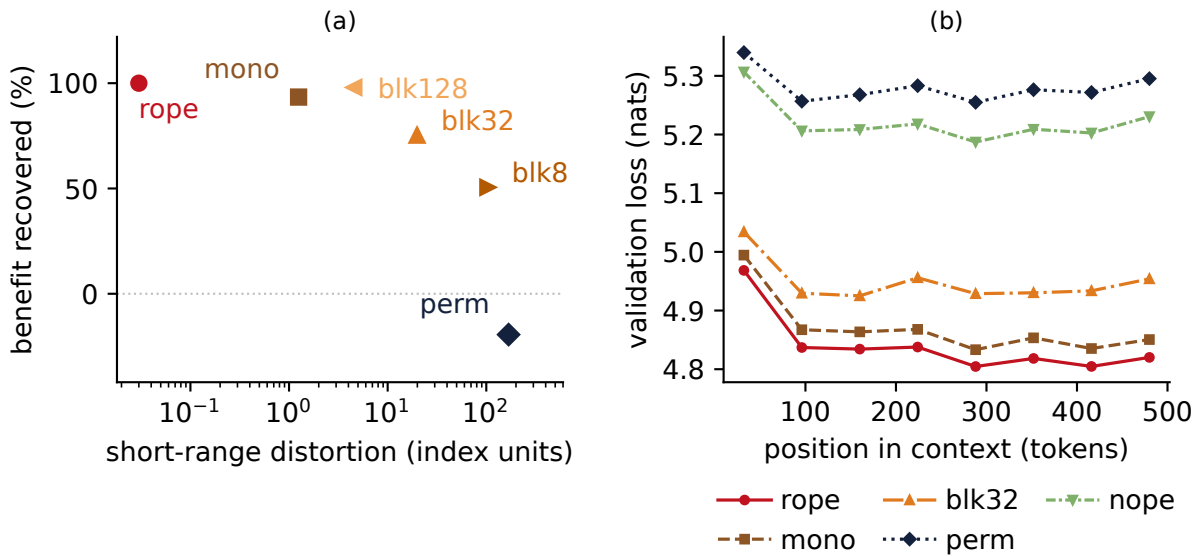


Figure 1: Positional distortion and validation loss for the 124M model after 10M training tokens. (a) Mean recovered benefit over five seeds versus short-range mean absolute offset error, on a logarithmic horizontal axis. Standard RoPE’s zero distortion is displayed at 0.03 for visibility. Each point is one fixed map. The points do not define a monotone relation across map families. (b) Mean cross-entropy in eight consecutive 64-position bins of the context window, using the same five seeds and validation sample. Lower loss is better.

4.85]; 11.81 at 10M [8.98, 14.63]; and 4.77 at 20M [2.54, 7.00]. At the final checkpoint, mean perplexity is 114.52 with the permutation and 109.75 without positional encoding. The earliest interval includes zero. A positive difference appears by 5M tokens, grows at 10M, and narrows by 20M while remaining positive in every seed. The difference therefore depends on training stage within this common schedule.

Second model size. At 29.9M parameters (6 layers, 6 heads, width 384), the four evaluated conditions retain the same perplexity ordering (Table 2). The permuted condition has 3.49 higher perplexity than `nope`, with a paired 95% t interval of [1.99,

5.00]. The block32 condition recovers 77.4% of RoPE’s benefit, compared with 75.3% at 124M parameters. Both sizes receive the same number of training tokens, giving different token-to-parameter ratios and different computation budgets.

Greedy generation. We continue 64 held-out prefixes of 128 tokens with 256 generated tokens each, using separately retrained seed-0 models and greedy decoding. Table 3 reports repetition within these continuations. The permuted condition has seq-rep-4 0.874 against 0.865 for `nope`, while the monotone condition repeats less than standard RoPE at this seed despite its higher perplexity. All four models have high repetition under this low

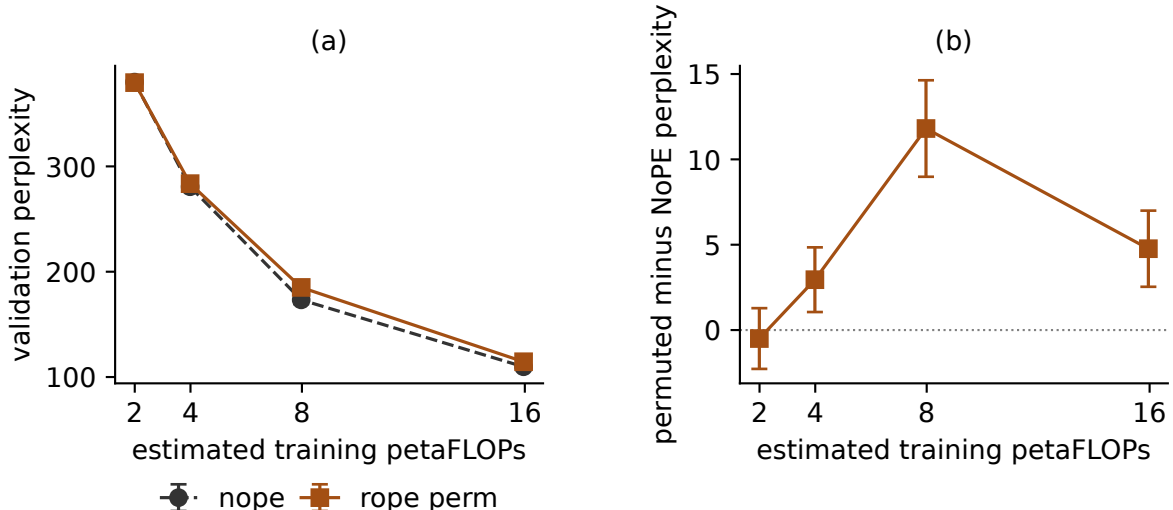


Figure 2: The separate 20M-token training series, evaluated at four checkpoints with five paired initialization seeds. (a) Mean validation perplexity for the fixed permutation and no positional encoding (NoPE). (b) Their paired perplexity difference. Bars are marginal 95% Student t intervals across seeds, without a multiple-comparison adjustment. The horizontal axis uses the dense arithmetic estimate in Equation 4; it is not measured GPU FLOPs.

Table 2: Second size: 29.9M parameters, 6 layers, 6 heads, width 384, 10M training tokens, 5 seeds. Validation perplexity, lower better; *recovered* is the share of the nope-to-rope gap retained, higher better. SD is the sample standard deviation. Four conditions were evaluated at this size.

condition	mean	SD	recovered (%)
rope	159.90	0.41	100.0
rope_blk32	176.09	0.97	77.4
rope_perm	234.98	1.73	-4.9
nope	231.48	1.58	0.0

training budget. This single-seed comparison describes generation behaviour alongside the replicated perplexity result.

6 Discussion

The matched low-budget comparison distinguishes information preserved by an index assignment from information that is useful to the trained model. A permutation of 512 positions has an inverse, yet using that permutation in RoPE gives higher perplexity than omitting RoPE in both tested model sizes. Invertibility of the assignment therefore does not guarantee equivalent learning behaviour under the tested optimizer and token budget.

Moderate monotone distortion retains most of the standard encoding’s benefit. The block conditions show that globally non-monotone maps can also retain substantial benefit when they preserve

Table 3: Degeneration under greedy decoding: 64 held-out prefixes of 128 tokens, 256 new tokens, one seed. *seq-rep-4* is the share of generated 4-grams repeating an earlier one; lower is better. *distinct-3* is the share of unique 3-grams; higher is better.

condition	seq-rep-4	distinct-3
rope	0.800	0.170
rope_mono	0.740	0.214
nope	0.865	0.110
rope_perm	0.874	0.100

local relationships. In particular, block128 performs better than the monotone condition despite larger mean short-range distortion. The arrangement of preserved offsets carries information that this scalar summary does not capture.

For an architecture using a different positional mechanism, these results motivate a comparison against both an ordered reference and a no-encoding baseline at matched training budgets. The experiment measures one concrete failure of fixed scrambling in RoPE. Recurrent states, decay kernels and other encoding families require their own comparisons before the same behaviour can be attributed to them.

7 Related work

Models without positional encodings. Haviv et al. (2022) study how decoder-only language models infer position information from the causal at-

tention mask. [Kazemnejad et al. \(2023\)](#) compare positional encodings for length generalization and demonstrate the effectiveness of no positional encoding in their evaluated settings. These findings motivate our `nope` baseline.

Randomized positions. [Ruoss et al. \(2023\)](#) sample and sort indices from a larger range during training to improve length generalization on algorithmic tasks. Their sorted construction is related to our monotone irregular map, although our map is fixed and rescaled to the original context endpoints. Their Appendix B.1 also evaluates unsorted randomized sinusoidal encodings in an approximately 249k-parameter encoder-only transformer, finding poor performance on several order-sensitive tasks. Those maps are resampled during training. We instead compare a fixed permutation of RoPE indices with no encoding in causal language models, using matched token streams and initialization seeds. This addresses the finite-budget effect of a stable scrambling in a different architecture and task setting. [Mehta et al. \(2026\)](#) extend randomized positional sampling to YaRN under a length curriculum; our fixed-window experiment does not evaluate context extension.

Context extension. Position interpolation ([Chen et al., 2023](#)) and YaRN ([Peng et al., 2024](#)) adjust the positional representation to extend context windows. Their effectiveness motivates studying how sensitive language models are to positional distortions, but their training settings and transformations differ from the fixed-map controls evaluated here.

8 Conclusion

At a low training budget, a fixed non-monotone scrambling of RoPE increases language-model perplexity relative to no positional encoding, while the tested moderate monotone distortion retains most of standard RoPE’s benefit. The scrambled condition is worse by 11.36 perplexity points at 124M parameters and by 3.49 points at 29.9M parameters, over five initialization seeds at each size after 10M training tokens. In the separate 20M-token series, the gap reaches 11.81 points at the 10M checkpoint and falls to 4.77 points at 20M, with a paired 95% interval of [2.54, 7.00] at the final checkpoint. Because the permutation remains fixed throughout training and evaluation, the result demonstrates a finite-budget cost of a consistent, invertible position assignment. The positional representation’s orga-

nization affects learning even when the assigned indices themselves lose no information.

Limitations

The experiments use WikiText-103 with about 0.08 processed tokens per parameter for the original 124M runs, 0.33 for the 29.9M runs, and 0.16 for the extended 124M runs. These budgets do not separate eventual learnability of the permuted representation from its learning speed, or establish behaviour at convergence. Each positional condition uses one map draw, so the results describe these sampled maps rather than an average over permutations or more extreme reorderings. The monotone condition is measured only at 124M parameters, and other positional encoding families and context lengths are outside the experimental scope. Generation uses one initialization seed and greedy decoding on 64 prefixes, so its repetition differences are descriptive.

Generative AI Disclosure

Generative AI tools were used for editorial and implementation assistance. All research design, experiments, analysis, and claims are the authors’ own, and the authors take full responsibility for the manuscript.

References

- Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. [Extending context window of large language models via positional interpolation](#). *arXiv preprint arXiv:2306.15595*.
- Adi Haviv, Ori Ram, Ofir Press, Peter Izsak, and Omer Levy. 2022. [Transformer language models without positional encodings still learn positional information](#). In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1382–1390. Association for Computational Linguistics.
- Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2023. [The impact of positional encoding on length generalization in transformers](#). In *Advances in Neural Information Processing Systems*.
- Manas Mehta, Fangcong Yin, and Greg Durrett. 2026. [Randomized YaRN improves length generalization for long-context reasoning](#). *arXiv preprint arXiv:2606.23687*.
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. [Pointer sentinel mixture models](#). In *International Conference on Learning Representations*.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. [YaRN: Efficient context window extension of large language models](#). In *International Conference on Learning Representations*.

Anian Ruoss, Grégoire Delétang, Tim Genewein, Jordi Grau-Moya, Róbert Csordás, Mehdi Bannani, Shane Legg, and Joel Veness. 2023. [Randomized positional encodings boost length generalization of transformers](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1889–1903. Association for Computational Linguistics.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. [RoFormer: Enhanced transformer with rotary position embedding](#). *Neurocomputing*, 568:127063.

A Per-seed results

Table 4 gives validation perplexity for each initialization seed at 124M parameters. Seeds are paired across conditions. The training token stream, validation sample, and position-map draw are common to all seeds.

Table 4: Validation perplexity, every condition at every seed, 124M parameters. Conditions share the token stream, so columns are paired.

condition	s0	s1	s2	s3	s4
rope	126.01	126.65	126.87	126.42	126.80
rope_blk128	127.15	128.23	127.72	127.45	128.09
rope_blk32	140.02	141.23	140.73	141.59	141.50
rope_blk8	155.28	156.95	155.36	154.59	155.48
rope_mono	130.55	130.67	130.36	129.99	130.50
rope_perm	197.65	193.37	195.89	198.17	197.46
nope	186.43	185.47	186.71	186.31	180.82

B Distortion enumeration

For a position map ρ , the global distortion averages over all $i > j$ in a 512-token window. The short-range distortion averages over $1 \leq i - j \leq 8$. Both enumerate the index map directly, with no sampling. The monotone map has mean absolute errors of 2.68 units globally and 1.25 units over short-range pairs. The numerical mean does not bound individual errors.

C Training computation

Each layer has $4d^2$ attention-projection weights and $8d^2$ feed-forward weights. The tied vocabulary projection contributes Vd weights that participate in a matrix multiplication at every token. Forward multiplication costs approximately two operations

per weight per token, and its input and weight gradients cost four, giving the factor six in Eq. 4. The two attention products each cost $2T^2d$ per layer in forward, and together with their backward products contribute $12LTd$ per token under square-matrix accounting. Enumerating only the causal triangle replaces this attention term with $6L(T + 1)d$. The latter convention gives 7.691 and 1.865 petaFLOPs for the 124M and 29.9M no-encoding models at 9,994,240 training tokens.

Our reference count excludes normalization, GELU, attention softmax, cross-entropy, optimizer updates, gradient clipping, embedding lookup and its scatter gradient, residual additions, unary sign changes, validation and preprocessing. It also excludes kernel padding and recomputation inside fused attention, so it is neither a device FLOP measurement nor a strict bound on one. Rotation tables are precomputed. For each pair of coordinates, each query or key rotation uses four multiplications and two additions in forward and again in backward, producing the $12Ld$ increment per token.