
Relative vs. Absolute Positional Encoding Predicts Fixed-Length Position Generalization in Micro-Transformers

John Lee

Deep Network Understanding Lab
Department of Computer Science
Dickinson College
leejo@dickinson.edu

Abstract

Length generalization for transformer models, training on short sequences and testing on longer ones, has been studied extensively. But a transformer can also fail without any change in sequence length: the shift can be over *where* a task-relevant symbol sits within a fixed length. We study this fixed-length, held-out-position setting on a micro-transformer (under one million parameters), trained from scratch to decide which of two marked symbols in a string comes first. We train with both symbols in the first half of the string and test with both in the second half, and we compare five positional encodings under an otherwise identical model. All five learn the trained positions perfectly, but only the relative encodings (no positional encoding, rotary, and a T5-style relative bias) generalize to the held-out positions, while the absolute encodings (learned and sinusoidal) do not transfer reliably. The relative/absolute distinction, not the individual method, predicts which encodings generalize. Most strikingly, rotary encoding generalizes well here even though it is reported to generalize poorly for length. In our fixed-length setting it behaves like the other relative encodings, but for length generalization it behaves like the absolute ones. This reversal is direct evidence that fixed-length position generalization is a distinct phenomenon from length generalization.

1 Introduction

Transformers often fail to generalize beyond the sequences they were trained on, and much of the work on this failure has focused on *length*: a model trained on short inputs is tested on longer ones, and the test positions are ones the model never saw during training. But length is not the only axis along which the input distribution can shift. Consider a task defined on a fixed-length input, where the answer depends on *where* a special symbol sits. We can train only on examples where the symbol falls in the first half of the input, then test on examples where it falls in the second half, without changing the length. Every position is present in every example; what the model has not seen is the symbol placed at the held-out positions. This is a distribution shift over position within a fixed length, not over length.

This report asks whether a micro-transformer (under one million parameters) generalizes across such a position shift, and whether the choice of positional encoding decides the outcome. The setting is deliberately small. Most work on transformer generalization uses models large enough to require substantial compute. Models this small are comparatively understudied, and they can be trained from scratch on an ordinary CPU. This keeps each experiment cheap, so we can run many seeds and control every position exactly.

We study a relative-order task on a fixed-length string: given one X and one Y, the model decides which comes first. To test generalization we split the data by position, training with both symbols in the first half of the string and testing with both in the second half. All five runs use the same model; only the positional encoding differs. We compare three relative encodings and two absolute ones. The three relative ones are no positional encoding (NoPE) [Kazemnejad et al., 2023], rotary (RoPE) [Su et al., 2021], and a T5-style relative bias (T5) [Raffel et al., 2020]; the two absolute ones are a learned positional encoding [Radford et al., 2018] and a sinusoidal positional encoding [Vaswani et al., 2017].

Our main finding is a separation by family. All five encodings fit the trained positions perfectly, so the task is learnable in every case. On the held-out positions, the three relative encodings generalize, averaging 95.7% to 99.2%. The two absolute encodings do not. Sinusoidal stays below chance, and learned lands near 69% on average but is highly variable across seeds. The distinction that predicts generalization is not the specific method but the family, whether the encoding is relative or absolute. Within the absolute family, the two encodings even fail differently, which we revisit in the analysis.

Our contributions are: (i) a fixed-length, position-split evaluation that isolates position generalization from length generalization; (ii) a five-way comparison of positional encodings under this split on a micro-transformer; and (iii) the finding that the relative/absolute distinction predicts which encodings generalize, and that this setting is distinct from length generalization: most visibly, rotary encoding fails there but succeeds here.

2 Related Work

This report describes the output of an independent research project in the DNU Lab of Dickinson College. By design, DNU Lab projects focus on conducting research rather than surveying existing work. In this section we describe only one piece of related work. This serves as an initial step toward a more complete literature survey.

The closest prior work is Kazemnejad et al. [2023], who compare positional encodings for length generalization in decoder-only transformers. Their study spans ten tasks and models of around 100M parameters. They train on short sequences and test on longer ones, so the test positions never appear during training. They report that NoPE generalizes best, followed by the T5 bias, with the absolute encodings among the worst. We study the same underlying question, but over a different axis. Does the choice of positional encoding decide whether a model generalizes to unseen situations? Our input length is fixed, so every position is seen during training; the shift is over the *placement* of the task-relevant symbols, not over length.

The relationship is threefold. First, our result partly *reproduces* theirs: the relative encodings (NoPE and the T5-style bias) generalize while the absolute encodings fail, as in their setting. Second, our setting *complements* theirs by testing what they cannot. They can only evaluate a sinusoidal absolute encoding, because a learned one has no parameters for positions never seen in training. Our fixed-length setting exposes every position, so we can evaluate a learned absolute encoding directly. Third, and most tellingly, the two settings *diverge* on RoPE. Kazemnejad et al. [2023] report that it generalizes poorly and behaves much like an absolute encoding. In our fixed-length setting, RoPE instead lands firmly on the relative side (95.7% on average, far above either absolute encoding). The same method landing on opposite sides of the relative/absolute divide suggests that length generalization and fixed-length position generalization are not the same phenomenon.

3 Experimental Design

3.1 Task

Each input is a length-20 string made of the digits 0–9 with exactly one X and one Y substituted in, followed by a separator “:” and a single answer token. The label is T if the index of X is smaller than the index of Y, and F otherwise. The other eighteen characters are random digits and do not affect the label. Two examples:

X904Y482520615370751:T (X at 0, Y at 4 → T)
 5337854YX80803953079:F (Y at 7, X at 8 → F)

The model reads the string and the separator and predicts one token. Because the answer is a single token, chance accuracy is 50%, and we keep the two classes balanced so that accuracy is meaningful. The model’s input vocabulary is the ten digits, X, Y, and the separator “:”; the two labels T and F appear only as the output token.

3.2 The position split

The generalization test is created entirely by controlling *where* X and Y are allowed to appear. This report uses the `half` split throughout: training and validation examples place both symbols in the first half of the string (indices 0–9), while the held-out test examples place both in the second half (indices 10–19). The input length is 20 in every pool, so the model sees all 20 positions in every example; the only thing withheld from training is the configuration in which the symbols occupy the second half.

We generate three separate pools rather than slicing one stream. The training and validation pools both use the first-half positions; validation holds unseen examples for monitoring only. The generalization test pool uses the held-out positions and is never seen during training. The pools hold 50,000 training, 5,000 validation, and 2,000 test examples, each balanced 50/50 between T and F.

This design isolates position generalization from length generalization. The input never changes length, so any gap between validation and test accuracy is a failure to transfer a rule from trained positions to held-out positions, not a failure to extrapolate to unseen positions.

3.3 Model and training

All runs use the same decoder-only transformer implemented from scratch, with four layers, four attention heads, and an embedding width of 128, for approximately 0.8M parameters and a causal attention mask. Each example is 22 tokens long, well within the context length of 64. The only independent variable across runs is the positional encoding. Training uses AdamW with learning rate 10^{-3} (decayed to 10^{-4} with a 100-iteration warmup), batch size 64, for 2000 iterations on CPU. The loss is computed only on the final answer token, not on the digits, so the model is scored on the classification decision rather than on modeling the input stream. The code for the model, data generation, and evaluation is available at <https://github.com/JinLeeGG/comp560-JohnLee>.

3.4 The five positional encodings

We compare three relative and two absolute encodings, holding the rest of the architecture fixed:

- **Relative.** *No positional encoding* (NoPE): no position signal is added; order information is available only through the causal attention mask, which lets NoPE learn relative-position patterns [Kazemnejad et al., 2023]. *RoPE* [Su et al., 2021]: query and key vectors are rotated by an angle proportional to their absolute position, so that the attention dot product depends only on the relative offset between tokens. *T5-style relative bias* [Raffel et al., 2020]: a learned scalar bias indexed by the relative distance between query and key is added to the attention score, using the standard T5 bucketing (exact buckets for nearby offsets and log-spaced buckets for far ones; 32 buckets, causal).
- **Absolute.** *Learned* [Radford et al., 2018]: a trainable position embedding is added to the token embedding at each index. *Sinusoidal*: the fixed sine/cosine table of Vaswani et al. [2017] is added instead.

3.5 Seeds and reporting

We run ten seeds per encoding. Each seed regenerates the data split, so a seed varies the training and test examples as well as the model initialization and batch order; the reported spread therefore reflects data-sampling variance, not only initialization noise. We report accuracy on the in-distribution validation pool (trained positions) and on the generalization test pool (held-out positions), as the mean over seeds. We also give the per-seed breakdown, so the family separation can be compared against seed variation.

Table 1: Accuracy on the half split, mean $\pm$ standard deviation over ten seeds (each seed regenerates the data split). Validation is on trained positions (0–9); held-out test is on positions 10–19. Chance is 50%. All five encodings learn the task; the relative family generalizes while the absolute family does not.

Encoding	Family	Val (trained pos.)	Held-out (unseen pos.)
NoPE (causal mask)	relative	100%	99.2 $\pm$ 1.7
T5 relative bias	relative	100%	99.2 $\pm$ 2.1
RoPE	relative	100%	95.7 $\pm$ 6.8
Learned	absolute	100%	68.6 $\pm$ 18.1
Sinusoidal	absolute	100%	31.9 $\pm$ 6.9

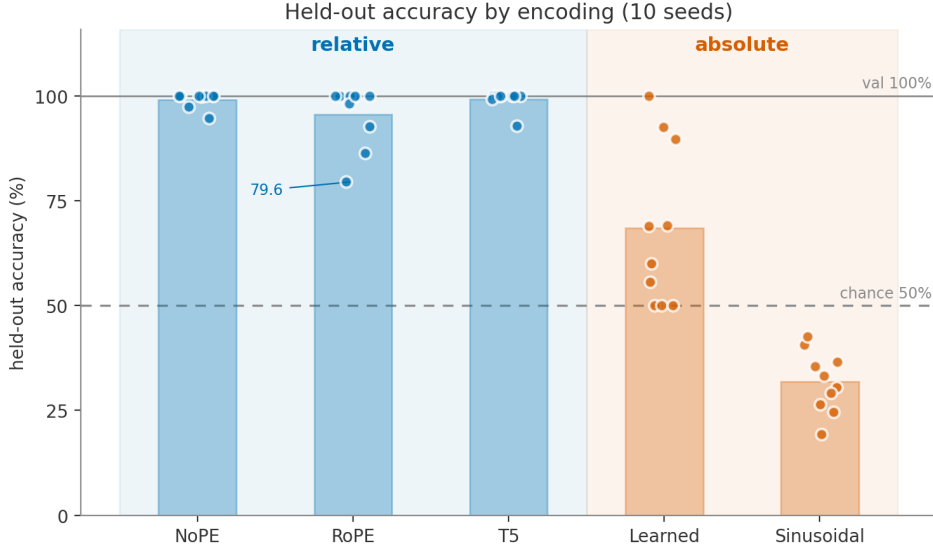


Figure 1: Held-out accuracy by encoding on the half split. Bars are the mean over ten seeds; each dot is one seed (a box plot is misleading at $n = 10$). The relative family sits near the val=100% line; learned straddles chance with a wide, bimodal spread, so its mean bar (68.6%) is a value no single seed produced; sinusoidal sits below chance. RoPE’s one low seed (79.6%) is labeled. Because every encoding reaches 100% validation accuracy, each gap below it is a generalization failure, not a learning failure.

4 Results

Table 1 reports validation and held-out accuracy for each encoding, as the mean and standard deviation over ten seeds. Every encoding reaches 100% validation accuracy, so all five learn the task on the trained positions. The encodings separate by family on the held-out positions: the three relative encodings average between 95.7% and 99.2%, while the two absolute encodings average 68.6% (learned) and 31.9% (sinusoidal), both well below the relative family. The relative/absolute gap (roughly 27 to 67 percentage points) is far larger than the standard error of any mean, so the separation is robust. Sinusoidal in particular sits well below chance rather than merely failing to transfer, a stronger failure we take up in Section 5.2.

Figure 1 plots held-out accuracy per encoding with the individual seeds overlaid, and shows that the family gap holds on every seed while the encodings differ in stability. Among the relative encodings RoPE is the least stable, dropping to 79.6% on one seed while NoPE and the T5 bias never fall below 92.9%. The two absolute encodings fail differently: sinusoidal is below chance on every seed, whereas learned is bimodal, ranging from 50% up to 100%, so its mean is a value no single seed produced. Full per-seed numbers are in Appendix A.

Per-position accuracy by encoding, averaged over 10 seeds

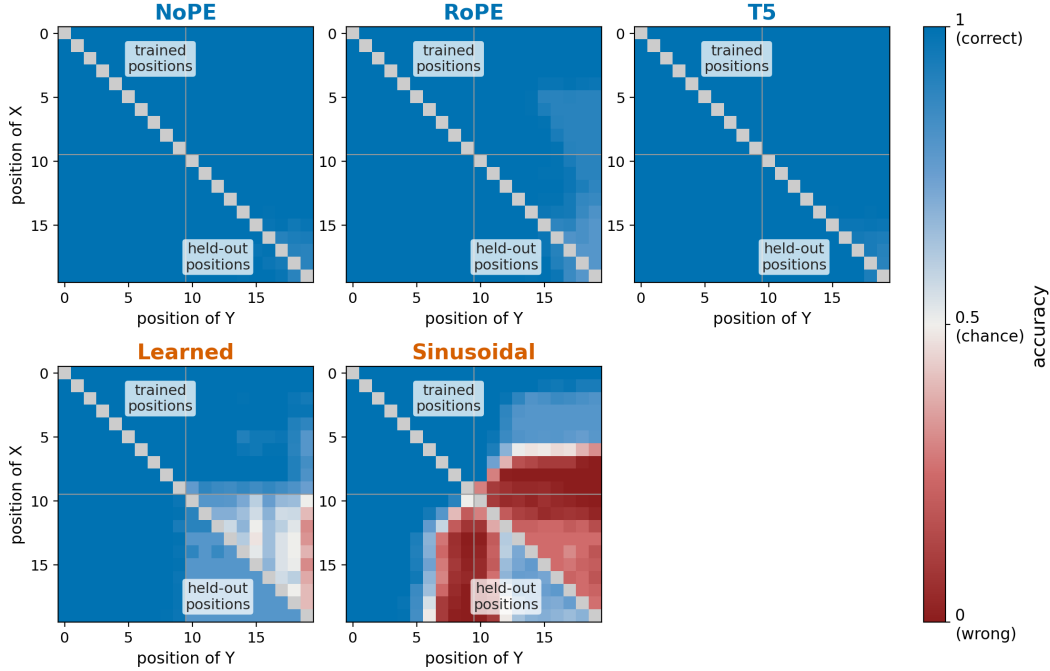


Figure 2: Per-position accuracy for each encoding (ten-seed mean); the vertical axis is X’s position and the horizontal is Y’s. The diagonal ($x = y$) is excluded because the two symbols never share a cell. Blue = correct, gray = chance, red = below chance. Each cell is one ordered position pair, so the grid covers every pair of positions. The top-left block has both symbols in 0–9 (trained) and the bottom-right both in 10–19 (held-out); the two off-diagonal blocks, with one symbol in each half, are shown only for context. In sinusoidal’s held-out block, the split into a correct and an incorrect triangle reflects the model largely predicting a single label: it gives the same answer regardless of position, so it is correct only where that answer happens to be right, not because it reads the positions.

Figure 2 breaks accuracy down by the positions of X and Y. The relative encodings are correct across the whole grid; the absolute encodings are correct in the trained block (top-left) but fail in the held-out block (bottom-right), where learned is uneven and sinusoidal falls well below chance.

5 Analysis

5.1 Why the relative encodings generalize

The relative encodings express the answer through the offset between positions rather than through the positions themselves. NoPE has order information only through the causal mask, which reflects whether one token precedes another; RoPE makes the attention dot product a function of the offset between two tokens; the T5-style bias is indexed directly by relative distance. A rule stated in terms of relative offset (is X before or after Y?) is the same rule in the first half and the second half of the string, so a model that learns it on the trained positions applies it unchanged on the held-out positions. The absolute encodings tie the computation to the specific indices seen in training (0–9). On the held-out indices (10–19) those features do not carry the meaning the model learned, so the model misfires.

5.2 Two ways for an absolute encoding to fail

The two absolute encodings fail in different ways, and the difference points to a mechanism. Sinusoidal fails robustly: it is below chance on every seed (31.9 ± 6.9), because its position vectors are fixed by a formula and never adapt. The vectors for the held-out positions produce a definite signal, but the rule the model attached to that signal in the first half is wrong in the second half, so the model is confidently incorrect rather than merely guessing. Learned fails on average but not reliably: it varies from pure chance to full generalization across seeds (68.6 ± 18.1 , up to 100% on one split). We attribute this to how the two encodings treat the held-out positions during training. In our fixed-length inputs every position is occupied in every example, so the learned position embeddings for indices 10–19 do receive gradient updates, from the filler digits that sit there, even though X and Y never do. Whether those embeddings happen to develop features usable for the order computation is init- and data-dependent, which produces the large seed variance. Sinusoidal’s held-out vectors, being fixed, never get this chance, which is why it fails consistently. This asymmetry is invisible in a length-generalization setting, where the held-out positions do not exist during training at all.

5.3 Limitations

Several limits should temper how far this result is read. The study covers a single task, a single split, and a single fixed length, so we cannot yet say the relative/absolute line holds across position-dependent tasks in general. The ten seeds each regenerate the data split, so the reported spread reflects both model-initialization and data-sampling variance; still, ten seeds pin down the family separation far better than they pin down the learned encoding’s mean, whose large variance would need more seeds to estimate precisely. We therefore treat learned’s occasional full generalization as a qualitative observation, not a quantified rate. Finally, the mechanism we propose (that learned’s held-out embeddings are trained by the filler tokens while sinusoidal’s are fixed) is consistent with the accuracy pattern but is not directly verified; confirming it is a question for mechanistic analysis rather than accuracy numbers.

6 Conclusion

On a fixed-length, position-split task, a micro-transformer generalizes to held-out positions when its positional encoding represents position relatively and fails when the encoding represents position absolutely. All five encodings we tested learn the trained positions perfectly; the relative encodings (NoPE, RoPE, and a T5-style bias) transfer to held-out positions at 95.7% to 99.2% on average, while the absolute encodings do not transfer, sinusoidal failing robustly below chance and learned failing on average but with high variance. The family, not the individual method, predicts the outcome. This mirrors the relative-over-absolute ordering reported for length generalization, but the two settings diverge, most visibly on RoPE, which suggests that generalizing over position at fixed length is a distinct phenomenon worth studying on its own.

Future work includes tightening the estimate of the absolute encodings’ variance with more seeds, testing other position splits and tasks, shrinking the model toward the smallest one that still solves the task, and using mechanistic analysis to test the held-out-embedding explanation directly.

References

- Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.19466.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. Technical report, OpenAI, 2018.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.
- Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

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

Use of Artificial Intelligence

An AI assistant (Claude) was used substantially in building the infrastructure for this project and in preparing this report. It helped write the code (the positional-encoding engine, the data-preparation and evaluation scripts, and the plotting code), ran the ten-seed sweep, organized the experiment logs, produced the figures, and drafted the text of this report.

The human contributions were the direction and the judgment. The author chose which experiment to report, framed the research questions, and read the cited work personally to verify the Related Work. The author reviewed every claim, number, and figure against the experiment logs, revised the draft sentence by sentence, and takes full responsibility for the accuracy and originality of the final report.

A Per-seed held-out accuracy

Table A1 gives the held-out accuracy of every encoding on each of the ten seeds, the per-seed data behind Figure 1 and the means in Table 1.

Table A1: Held-out accuracy by seed on the half split (each seed’s own data split). The relative family stays high on every seed; sinusoidal is consistently below chance; learned is highly variable (50% to 100%).

Seed	NoPE	RoPE	T5	Learned	Sinusoidal
1337	100.0	99.7	100.0	69.0	30.6
1338	97.5	92.8	100.0	50.0	26.4
1339	100.0	100.0	100.0	89.7	36.6
1340	100.0	100.0	100.0	69.1	40.6
1341	94.7	79.6	100.0	100.0	33.2
1342	100.0	100.0	92.9	50.0	35.4
1343	100.0	86.5	100.0	92.7	29.1
1344	100.0	98.3	99.3	60.0	42.7
1345	100.0	100.0	100.0	50.0	24.6
1346	100.0	100.0	100.0	55.6	19.2
Mean	99.2	95.7	99.2	68.6	31.9