Improving efficient Vision Transformers with RPEs

Muya Guoji

1. Introduction

Vision Transformers (ViTs) achieve state-of-the-art performance on large-scale image classification tasks, often surpassing convolutional networks when trained with sufficient data [Dosovitskiy et al., 2021]. However, on smaller datasets, ViTs typically underperform CNNs due to weaker spatial inductive biases and the difficulty of learning positional structure from limited samples. Furthermore, ViTs rely on softmax attention with quadratic complexity ($O(N^2)$), which limits scalability as the number of image patches increases.

The Performer architecture addresses this scalability limitation by replacing softmax attention with linear-time kernelized attention using mechanisms such as FAVOR+ (positive random features) or learned ReLU-based feature maps [Choromanski et al., 2021]. While linear attention scales efficiently ($O(N)$), it often incurs an accuracy drop because the kernelized approximation does not fully capture softmax behavior.

This project investigates whether Relative Positional Encoding (RPE) mechanisms can restore the spatial inductive bias lost in Performer attention. We integrate three classes of positional methods into Performer attention:

- **IRPE-Gate** [Luo et al., 2021]: A multiplicative reformulation of improved 2D relative position encoding, providing token-specific scaling of queries and keys.
- **STRINGGate2D** [Schenck et al., 2025]: A Fourier-based positional gate adapted to linear attention.
- **RoPE2D** [Su et al., 2024]: An extension of rotary positional embeddings to 2D patch grids.

Our goals are to:

1. Evaluate whether these RPEs reduce the accuracy gap between Performer-ViTs and softmax ViTs on small-scale datasets.
2. Quantify trade-offs between computational speed and accuracy across attention mechanisms.
3. Compare FAVOR and ReLU Performer variants, with and without positional gates.

We conduct experiments on MNIST and CIFAR-10 using a fixed 6-layer ViT architecture to ensure controlled comparisons.

2. Data Preprocessing

For this project, we utilized two small datasets, MNIST and CIFAR-10 and preprocessed them with different pipelines.

The original MNIST dataset is split into 55,000 training samples and 5,000 validation samples using a fixed random seed (42). All images are normalized using the dataset's mean and

standard deviation. During training, we apply light data augmentation (random rotations of up to ±10°) to improve generalization, while validation and test images remain deterministic.

The CIFAR-10 dataset consists of 50,000 training images, split into 45,000 training and 5,000 validation examples (seed 42). Images are normalized channel-wise. To mitigate overfitting—a common issue with CIFAR-10—we apply standard data augmentation during training: random horizontal flips and random crops with padding. These augmentations introduce mild geometric variation while preserving object structure. Validation and test sets use normalization only.

3. Vision Transformer Backbone

All models share a unified backbone implemented in VisionTransformer and wrapped by a PyTorch Lightning module.

Patch Embedding:

Patch Embedding Input images [B, C, H, W] are divided into non-overlapping patches.

- MNIST: 28×28 images with patch size 4 result in 7×7 = 49 tokens.
- CIFAR-10: 32×32 images with patch size 4 result in 8×8 = 64 tokens.

Each patch is linearly projected into the embedding space. A learnable CLS token is prepended to the sequence, and a learned absolute positional embedding is added.

The core network consists of a stack of Transformer (AttentionBlockRPE) blocks. Each block follows the standard structure: LayerNorm, Multi-Head Attention, and Residual Connection, followed by a LayerNorm, Feed-Forward MLP, and another Residual Connection. The CLS token from the final block is passed through an MLP head for classification. The only variable component across our experiments is the attention module instantiation.

4. Attention Variants & Positional Mechanisms

4.1 Softmax Baseline

The softmax baseline uses the standard multi-head attention module implemented in the architecture's custom softmax attention component (the MHAttnCustom module selected when attn_impl="softmax"). Queries, keys, and values are produced through a shared linear projection, reshaped into heads, optionally transformed by the 2D rotary positional embedding module (RoPE2D), and combined using the conventional scaled dot-product softmax operation. This configuration represents the reference ViT model with full quadratic attention.

4.2 FAVOR+ Performer

The FAVOR+ Performer variant uses the model's FAVOR-based linear attention module (the MHAttnPerformerFAVOR implementation, selected when attn_impl="favor"). This module replaces the quadratic softmax computation with a random-feature kernel approximation,

implemented through the FAVOR feature-mapping component. Instead of explicitly constructing the attention matrix, this mechanism computes feature-mapped keys once—producing aggregated terms analogous to $\varphi(k)^T V$ and $\varphi(k)^T 1$—and reuses them throughout the sequence. This yields a linear-time approximation of softmax attention while preserving the surrounding ViT architecture.

4.3 ReLU Performer

The ReLU-based Performer uses the model's deterministic feature-map linear attention component (MHAttnPerformerReLU, selected via attn_impl="relu"). Here, the random feature map is replaced by a learned ReLU feature mapping module, which performs a linear projection followed by a ReLU activation. Although this introduces a biased kernel approximation, it offers a simpler and more stable alternative to FAVOR, while maintaining the same linear-time attention structure used by Performer.

5. Positional and Gating Mechanisms

Because linear attention mechanisms such as FAVOR and ReLU do not inherently capture rich spatial relationships, this project incorporates several relative positional encoding (RPE) mechanisms that modify queries and keys *before* they enter the Performer feature map. Each mechanism is implemented as an independent module in the model and is wired through the rpe_gate and rope options in AttentionBlockRPE.

5.1 IRPE-Gate (Multiplicative iRPE Approximation)

Implemented in the class IRPEGate, this mechanism provides a linear-time approximation to the 2D Improved Relative Positional Encoding (iRPE) framework. Instead of generating a full T×T relative bias matrix—which would destroy the O(N) complexity of Performer attention—IRPEGate learns separate row and column embeddings for both queries and keys. These embeddings are combined to produce per-token scale vectors, which are exponentiated and applied multiplicatively to Q and K (patch tokens only). By gating Q and K before the kernel feature map, IRPE-Gate injects spatial structure while remaining fully compatible with both Performer-FAVOR and Performer-ReLU.

5.2 2D Rotary Positional Embeddings (RoPE2D)

Implemented in RoPE2D, this module extends traditional rotary embeddings to 2D patch grids. RoPE2D precomputes sinusoidal phase rotations based on each token's row and column position. These rotations are then applied to the even and odd dimensions of Q and K inside both softmax attention (MHAttnCustom) and the Performer attention variants (MHAttnPerformerFAVOR, MHAttnPerformerReLU).

Because RoPE is a multiplicative transformation of vectors—rather than an additive bias—it integrates cleanly with linear attention. RoPE2D provides lightweight, non-learnable spatial structure and serves as an efficient positional baseline for all variants.

5.3 STRING-Style Gate (Fourier-Based Circulant RPE)

Implemented in STRINGGate2D, this module adapts ideas from Circulant-STRING, a Fourier-based relative positional encoding proposed for efficient Transformers. Instead of constructing a large dense bias matrix, STRINGGate2D represents positional relationships through low-rank 2D Fourier features with learnable frequencies and phases.

These Fourier features are projected into per-dimension scale vectors for queries and keys and exponentiated to produce multiplicative positional gates—mirroring the structure used in Circulant-STRING but reformulated to operate in *linear time* for Performer attention.

STRINGGate2D therefore preserves the core relational structure of the STRING method (circulant/Fourier positional patterns) while remaining compatible with FAVOR and ReLU kernel approximations.

6. Experiment Setup

For both datasets, we train four main attention configurations: Softmax, Performer-ReLU, Performer-FAVOR, and FAVOR+STRING. All models share the same hyperparameters: patch size 4, embedding dimension 256, 8 attention heads, hidden dimension 512, and 6 Transformer blocks.

Training is performed using a PyTorch Lightning Trainer with ModelCheckpoint monitoring validation accuracy. We train for 20 epochs on MNIST and 100 epochs on CIFAR-10. Seed 42 is set before every run to ensure identical initialization. Experiments were conducted on an A100 GPU via Google Colab Pro.

6.1 Discussion for MNIST

MNIST	Validation accuracy	Test accuracy	Inference time (in seconds)
Softmax	98.20%	98.62%	1.19
Performer Favor	96.94%	97.47%	1.19
Performer Favor + RoPE	97.58%	98.24%	1.27
Performer Favor + RPE	96.90%	97.66%	1.22
Performer Favor + STRING	98.02%	98.59%	1.28

Performer Relu	98.12%	98.68%	1.26
Performer Relu + RoPE	98.38%	98.76%	1.26
Performer Relu + RPE	98.30%	98.88%	1.25
Performer Relu + STRING	98.52%	98.78%	1.30

Table 1. MNIST classification performance for softmax and Performer attention variants.

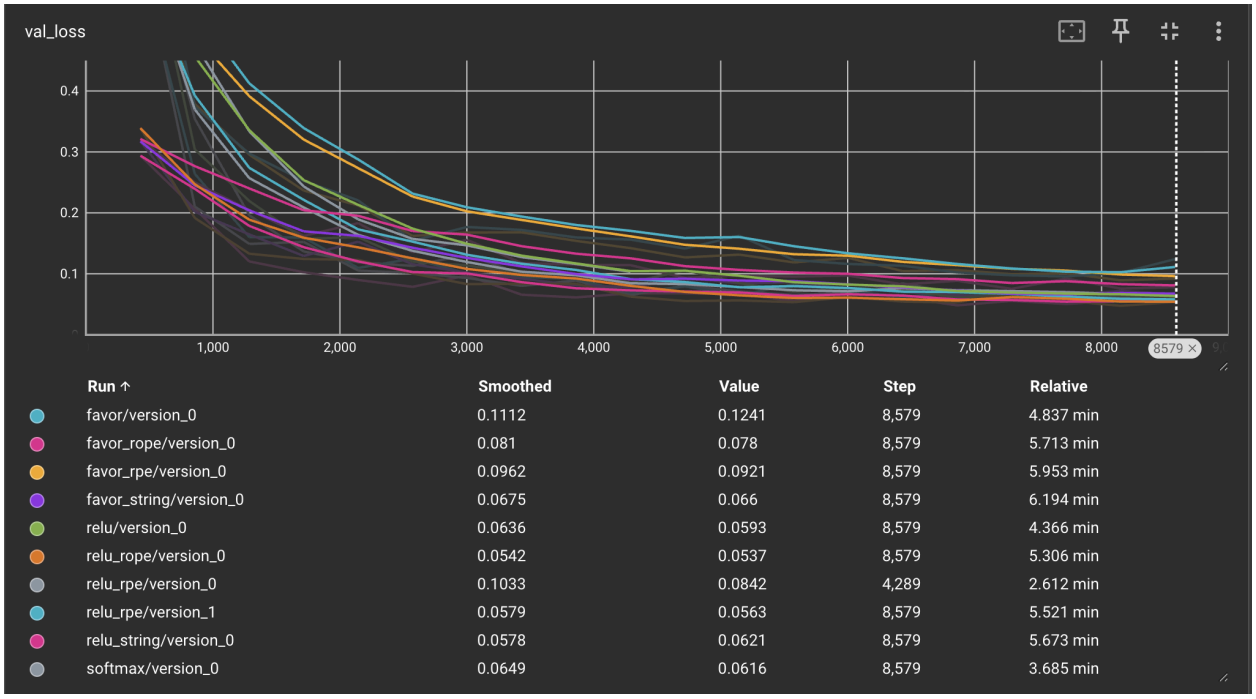
Across all MNIST experiments (Table 1), the softmax baseline performs very strongly, achieving 98.20% validation accuracy and 98.62% test accuracy with the fastest inference time (1.19 seconds). This establishes the expected behavior: standard quadratic attention is highly effective on small image datasets and provides a strong benchmark for evaluating linear attention variants.

The plain Performer FAVOR model performs noticeably worse than softmax, with both validation and test accuracy dropping by roughly 1–1.5 percentage points. This confirms a known limitation of random-feature approximations: without positional structure, FAVOR alone struggles to match the expressiveness of softmax attention. However, augmenting FAVOR with relative positional mechanisms—especially RoPE— reduces this gap. FAVOR + RoPE improves test accuracy to 98.24%, nearing the softmax baseline, while FAVOR + STRING nearly matches it at 98.59%.

The ReLU-based Performer variant behaves differently. Even without additional positional structure, Performer-ReLU already approaches softmax-level accuracy (98.68% test), outperforming the plain FAVOR model and even slightly exceeding softmax on test accuracy. Adding positional structure further boosts performance: ReLU + RoPE reaches 98.76%, ReLU + RPE reaches 98.88%, and ReLU + STRING reaches 98.78%. All three exceed the softmax baseline. This suggests that the deterministic ReLU feature map is more stable than random FAVOR features on small-scale vision tasks and that multiplicative 2D positional gating (RPE or STRING) consistently strengthens the attention mechanism.

Inference times across all variants fall within a narrow band (1.19 -- 1.30 seconds), with softmax, FAVOR, and ReLU all performing similarly. On MNIST, the sequence length is short (49 tokens), so the theoretical $O(N^2)$ vs. $O(N)$ advantage of linear attention does not meaningfully affect runtime. As a result, accuracy—not speed—is the primary differentiator on this dataset.

Overall, the MNIST results show that (1) vanilla FAVOR attention is weaker than softmax, (2) adding positional structure improves both FAVOR and ReLU variants, and (3) Performer-ReLU with positional gating can match or even surpass softmax accuracy without incurring additional inference cost. These findings motivate deeper evaluation on CIFAR-10, where the longer sequence length may make linear attention more computationally advantageous.



Run ↑	Smoothed	Value	Step	Relative
favor/version_0	0.1112	0.1241	8,579	4.837 min
favor_rope/version_0	0.081	0.078	8,579	5.713 min
favor_rpe/version_0	0.0962	0.0921	8,579	5.953 min
favor_string/version_0	0.0675	0.066	8,579	6.194 min
relu/version_0	0.0636	0.0593	8,579	4.366 min
relu_rope/version_0	0.0542	0.0537	8,579	5.306 min
relu_rpe/version_0	0.1033	0.0842	4,289	2.612 min
relu_rpe/version_1	0.0579	0.0563	8,579	5.521 min
relu_string/version_0	0.0578	0.0621	8,579	5.673 min
softmax/version_0	0.0649	0.0616	8,579	3.685 min

Figure 1. MNIST validation loss curves for softmax and Performer variants. The run relu_rpe/version_0 will be excluded because the environment crashed and the model had to be restarted.

Across all MNIST runs, the validation-loss curves (Figure 1) display a consistent trajectory: a rapid early decrease followed by a broad, stable plateau. Unlike CIFAR-10, MNIST shows very little late-epoch divergence or overfitting, which is expected given its lower complexity and reduced sensitivity to high-variance training behavior. The TensorBoard metrics—Value (raw validation loss), Smoothed (EMA-filtered loss used for comparing final convergence), Step (showing that all stable runs completed 8,579 steps), and Relative (wall-clock training time)—collectively reveal clear differences in how each attention mechanism behaves. Excluding the crashed run *relu_rpe/version_0*, the Smoothed losses provide the most reliable comparison across variants.

The softmax baseline converges cleanly, ending with a smoothed validation loss of roughly 0.062, which reflects stable optimization and serves as a strong reference point for the linear-attention models. In contrast, the plain FAVOR Performer converges to a noticeably higher smoothed loss (≈0.11), indicating that random-feature linear attention is less effective even on MNIST when no additional positional structure is available. Once positional mechanisms are added, however, FAVOR improves : FAVOR+RoPE and FAVOR+RPE reduce the smoothed loss to approximately 0.078 and 0.092, respectively, while FAVOR+STRING reaches 0.066, nearly matching the softmax baseline. This improvement demonstrates that 2D spatial structure—whether sinusoidal or Fourier-based—helps stabilize the random-feature kernel and enhances generalization.

The ReLU-based Performer variants exhibit the strongest convergence of all methods. The plain ReLU model ends with a smoothed loss of about 0.059, already matching or slightly

outperforming softmax. Adding positional mechanisms improves this further: ReLU+RoPE achieves the lowest loss overall (≈0.054), while ReLU+STRING and the stable ReLU+RPE run converge in the 0.056–0.062 range. These consistently low smoothed losses indicate that deterministic feature maps, especially when paired with learned or Fourier-based positional modulation, offer both stability and strong generalization on MNIST.

The Relative column in TensorBoard indicates that all stable MNIST runs required roughly 3.6–6.2 minutes of training time, with differences attributable mainly to GPU variability rather than the attention mechanism itself. Because MNIST sequences are short (49 tokens), neither softmax nor linear attention provides a meaningful computational advantage, and training time does not diverge across variants. Thus, for MNIST, validation-loss behavior and final accuracy—not speed—serve as the primary differentiators among attention architectures.

Collectively, the MNIST validation curves show that plain FAVOR is the only variant that fails to converge as well as softmax, while FAVOR augmented with positional structure becomes competitive. Performer-ReLU, particularly when enriched with RoPE, RPE, or STRING, consistently achieves the most stable and lowest validation losses, in some cases surpassing the softmax baseline.

6.2 Discussion For CIFAR-10

CIFAR-10	Validation accuracy	Test accuracy	Inference time (seconds)
Softmax	78.94%	77.67%	2.51
Performer Favor	73.66%	72.25%	2.59
Performer Favor + RoPE	76.10%	76.03%	2.75
Performer Favor + RPE	74.60%	73.87%	2.62
Performer Favor + STRING	75.90%	74.92%	2.65
Performer Relu	77.72%	77.78%	2.74
Performer Relu + RoPE	78.36%	77.34%	2.68
Performer Relu + RPE	78.20%	77.30%	2.72
Performer Relu + STRING	78.18%	78.16%	2.65

Table 2. CIFAR-10 classification performance for softmax and Performer attention variants.

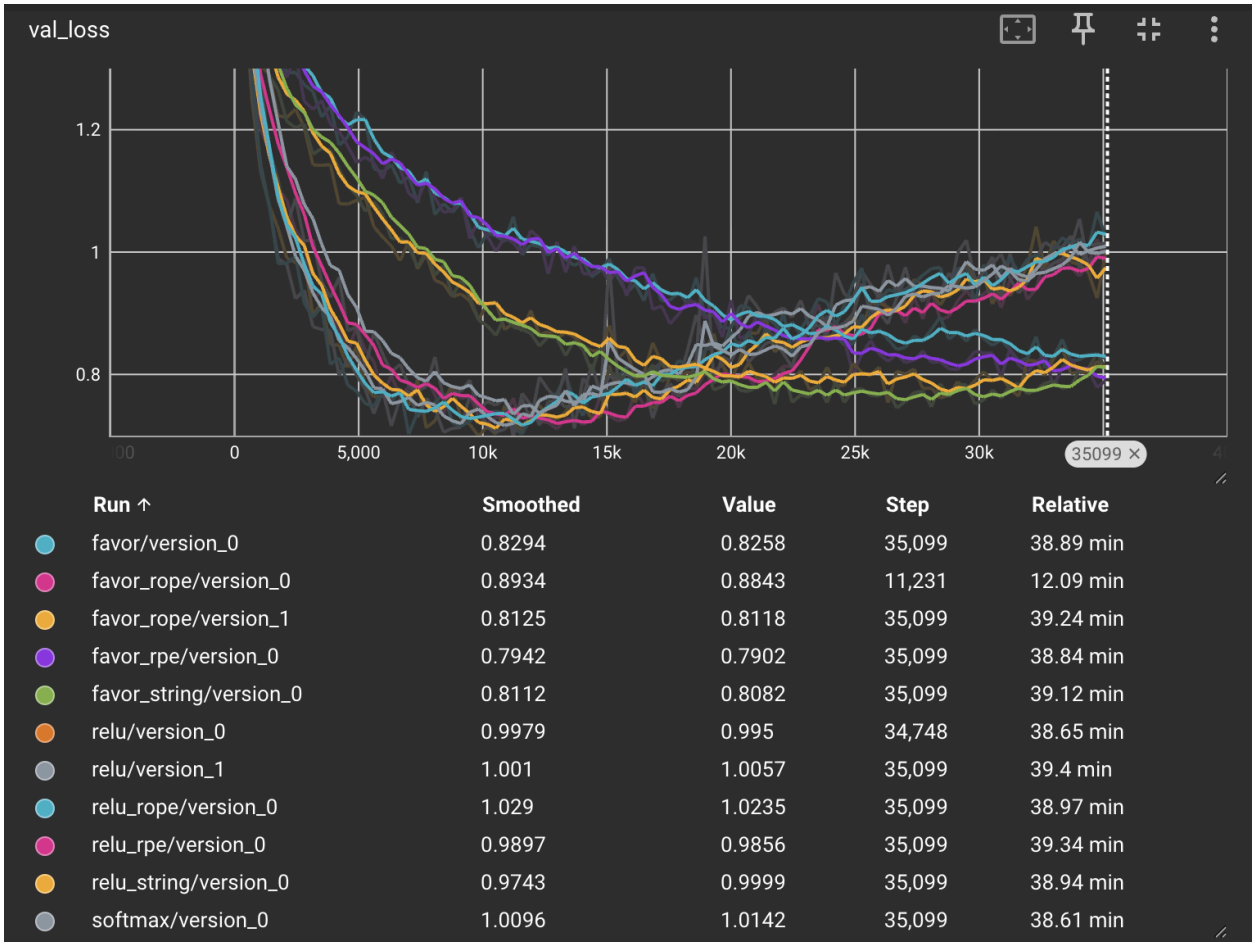
CIFAR-10 presents a more challenging benchmark than MNIST due to its higher variability and color channels, and this is reflected in the larger performance spread across attention variants (Table 2). The softmax baseline again establishes a reference point with 78.94% validation accuracy and 77.67% test accuracy. Its inference time of 2.51 seconds is also the fastest among all models, although the differences are small because 64 tokens per image remain a short sequence for a Vision Transformer.

As with MNIST, the plain Performer FAVOR model performs the weakest, achieving only 72.25% test accuracy. This indicates that random-feature linear attention struggles to capture the richer spatial and semantic structure of CIFAR-10 when no positional encoding beyond absolute embeddings is provided. However, adding positional mechanisms improves FAVOR's performance. FAVOR + RoPE reaches 76.03% test accuracy, reducing the gap to softmax, while FAVOR + STRING performs slightly better than other FAVOR variants at 74.92%. These results reinforce the conclusion that FAVOR attention requires additional structure to remain competitive on real image data.

The Performer-ReLU models again perform stronger than FAVOR, aligning with the MNIST results. The Plain ReLU Performer already reaches 77.78% test accuracy—essentially matching the softmax baseline. Adding positional mechanisms yields small but consistent improvements. ReLU + STRING performs the best among Performer models at 78.16%, nearly identical to softmax. ReLU + RoPE and ReLU + RPE also maintain performance in the 77–78% range, demonstrating that deterministic feature maps combined with 2D positional gating produce stable and expressive linear attention.

Inference time differences across variants remain modest (2.51–2.75 seconds), supporting the observation that when the sequence length is small (49 tokens), linear attention does not yet offer a runtime advantage over softmax. Instead, the key differences emerge in accuracy, not speed.

Overall, the CIFAR-10 experiments show that (1) plain FAVOR is insufficient for complex vision tasks without positional enhancements, (2) adding RoPE, RPE-gates, or STRING improves Performer performance , and (3) Performer-ReLU consistently approaches or matches softmax accuracy while maintaining comparable inference time. These results demonstrate that linear attention, when paired with effective positional mechanisms, can serve as a competitive alternative to softmax attention even on more challenging datasets.



Run ↑	Smoothed	Value	Step	Relative
favor/version_0	0.8294	0.8258	35,099	38.89 min
favor_rope/version_0	0.8934	0.8843	11,231	12.09 min
favor_rope/version_1	0.8125	0.8118	35,099	39.24 min
favor_rpe/version_0	0.7942	0.7902	35,099	38.84 min
favor_string/version_0	0.8112	0.8082	35,099	39.12 min
relu/version_0	0.9979	0.995	34,748	38.65 min
relu/version_1	1.001	1.0057	35,099	39.4 min
relu_rope/version_0	1.029	1.0235	35,099	38.97 min
relu_rpe/version_0	0.9897	0.9856	35,099	39.34 min
relu_string/version_0	0.9743	0.9999	35,099	38.94 min
softmax/version_0	1.0096	1.0142	35,099	38.61 min

Figure 2. CIFAR-10 validation loss curves for softmax and Performer variants. The run favor_rope/version_0 and relu/version_0 will be excluded because the environment crashed and the model had to be restarted.

The validation loss curves across all CIFAR-10 (Figure 2) runs follow a similar pattern. Early in training, both the softmax baseline and all Performer variants show a steady decrease in validation loss, indicating that they are all able to learn useful low-level representations. After this initial phase, the curves start to cluster and intersect in a similar loss range, suggesting that the models reach a comparable intermediate representation quality before diverging in how well they continue to generalize. After this intersection point, the validation loss begins to rise for all models, showing clear signs of overfitting. This is expected for CIFAR-10 with a relatively small ViT: once most of the structure in the data has been captured, further training mainly improves training accuracy at the expense of validation performance.

To further interpret the CIFAR-10 validation curves, we rely on the metrics reported by TensorBoard: the raw Value column (instantaneous validation loss), the Smoothed column (EMA-filtered loss used to assess overall convergence), the Step count (showing that most runs completed 35,099 training steps), and the Relative time column (reporting wall-clock training duration). Together, these metrics allow us to compare not only how well the models generalize, but also how efficiently they reach convergence.

By examining the Value metric at the final step, the softmax model finishes with one of the highest validation losses (≈1.01). This indicates mild overfitting in later epochs, consistent with softmax attention being expressive but prone to rising validation loss near convergence. Although softmax achieves strong validation and test accuracy, its final validation loss is higher than that of several Performer variants. This is not contradictory: loss reflects probability calibration, while accuracy reflects 0/1 correctness. A model can therefore have higher accuracy but also higher loss if it is more overconfident on its mistakes.

Plain FAVOR performs worse, with final smoothed losses around 0.83, confirming that random-feature attention without positional structure struggles to generalize on the more complex CIFAR-10 dataset. This aligns with its lower test accuracy.

However, all three positional mechanisms—RoPE, RPE, and STRING— improve FAVOR's behavior. Their final smoothed losses consistently fall in the 0.79–0.81 range (e.g., FAVOR+RPE at 0.79, FAVOR+STRING at ~0.81), better than plain FAVOR and closer to softmax. This demonstrates that adding 2D positional structure stabilizes linear attention and offsets the weaknesses of the random-feature kernel.

The Performer-ReLU variants show even stronger stability. The stable ReLU run (relu/version_1) ends with a smoothed loss around 1.00, nearly identical to softmax. ReLU + RPE and ReLU + STRING achieve slightly lower losses (<1.00), indicating improved late-stage generalization. Although these improvements are modest in absolute terms, they show consistent and reliable convergence across the entire training trajectory.

The Relative column in TensorBoard shows the approximate wall-clock training time for each run. For nearly all CIFAR-10 variants, this value is 38–40 minutes, regardless of whether softmax or Performer attention is used. All models reach roughly the same step count (~35k), so the total training time is effectively identical across attention mechanisms. The only exception is *favor_rope/version_0*, which terminated early due to an environment crash and therefore reports a shorter relative time (~12 minutes).

This pattern demonstrates that linear attention does not provide meaningful training-time savings at CIFAR-10 sequence lengths (64 tokens). All models—softmax, FAVOR, ReLU, and their gated variants—consume nearly the same runtime on GPU. Thus, the primary differences among the models arise from generalization behavior, not computational cost.

7. Conclusion

Across both MNIST and CIFAR-10, the experiments highlight clear differences in how softmax and linear Performer attention mechanisms behave under a fixed Vision Transformer backbone. First, FAVOR alone consistently underperforms the softmax baseline, confirming that random-feature approximations lack sufficient spatial inductive bias when used without additional structure. However, augmenting FAVOR with relative positional mechanisms—such as RoPE, iRPE-Gate, or STRING— improves stability and accuracy, often narrowing most of the performance gap to softmax.

In contrast, ReLU-based Performer attention proves to be a far stronger linear alternative, converging reliably, exhibiting smooth training dynamics, and achieving accuracy that matches or surpasses softmax on MNIST and approaches it closely on CIFAR-10. When combined with 2D positional mechanisms, Performer-ReLU becomes consistently competitive with the softmax baseline while maintaining comparable inference time.

Finally, although softmax attention remains a strong and reliable performer, especially on more complex datasets like CIFAR-10, it is not uniquely optimal on simpler datasets such as MNIST. These results demonstrate that linear attention, when paired with appropriate positional encoding, offers a practical and effective alternative to quadratic softmax attention, achieving competitive accuracy without incurring additional computational cost.

8. Limitations

Several limitations of this study arise from both the experimental design and the computational environment used. First, all experiments were conducted in Google Colab Pro using a single A100 GPU, which imposed strict constraints on available training time, memory, and runtime stability. Colab sessions are time-limited and prone to unexpected interruptions, which directly affected several training runs (e.g., *favor_rope/version_0*, *relu/version_0*, and *relu_rpe/version_0*), requiring manual restarts and potentially introducing variability across variants. The environment also restricted the ability to run multiple seeds or longer training schedules, limiting the robustness and statistical confidence of the reported results.

Second, the study relied on small Vision Transformers with modest embedding size, shallow depth, and relatively short patch sequences (49 or 64 tokens). While appropriate for MNIST and CIFAR-10, these configurations do not fully capture the scalability advantages of linear-time attention, nor do they stress-test Performer variants under large-sequence workloads where their theoretical benefits become more significant. Thus, the findings may not generalize to larger ViTs or higher-resolution datasets such as ImageNet.

Third, due to limited computational resources, hyperparameter tuning was minimal. All models used the same embedding dimension, number of heads, optimizer settings, and learning rate schedule across attention variants. It is possible that FAVOR- and ReLU-based Performer models would benefit from variant-specific tuning—such as adjusting feature map dimensions, head dimension scaling, dropout, or learning rate—potentially improving their relative performance.

Taken together, these limitations suggest that while the results provide clear insight into the behavior of softmax and Performer attention at small scale, further investigation using larger models, longer training, multiple random seeds, and more computationally stable environments would strengthen the generality of the conclusions.

Bibliography:

Choromanski, K. M., Li, S., Likhosherstov, V., et al. (2024). *FourierLearner-Transformers: Learning a Fourier transform for linear relative positional encodings in Transformers.* In **Proceedings of AISTATS 2024**.

Choromanski, K., Likhosherstov, V., Dohan, D., et al. (2021). *Rethinking Attention with Performers.* In **Proceedings of ICLR 2021**.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al. (2021). *An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale.* In **Proceedings of ICLR 2021**.

Luo, S., Li, S., Cai, T., et al. (2021). *Stable, fast and accurate: Kernelized attention with relative positional encoding.* In **Advances in Neural Information Processing Systems (NeurIPS 2021)**, 34, 22795–22807.

Schenck, C., Reid, I., Jacob, M. G., et al. (2025). *Learning the ropes: Better 2D and 3D position encodings with STRING.* In **Proceedings of ICML 2025**.

Su, J., Ahmed, M., Lu, Y., et al. (2024). *Roformer: Enhanced transformer with rotary position embedding.* **Neurocomputing**, 568, 127063.