Experiments on Factorized Spatio-temporal Self-Attention mechanisms and Position Encodings for improved correlation structure and long-dependency modeling

A.Padmaprabhan
External guide's: **Mr.Akash Ganguly** & **Mr.Shivansh Verma**
Physical Research Laboratory, Ahmedabad

June 27, 2025

The broad literature of Vision Transformers have focused on two prominent aspects , with the primary aspect being the reduction of computational costs with local attention and hierarchical encoder stages. This type of attention mechanism is observed in CvT (Convolutional Transformer) Wu et al., 2021, Swin (Shifted-window based Vision Transformer) Liu et al., 2021 and CSwin (Cross shaped Shifted Window based Vision Transformer) Dong et al., 2022. The local window-based self-attention divides the image into distinct subsets and confines self-attention computations within each subset. Therefore, the computational complexity of models using local windows is theoretically linear with the input image spatial resolution. The other aspect of these models are aimed at introducing local inductive biases either through teacher networks as observed in DeiT (Data efficient image Transformers) Touvron et al., 2021 and PvT (Pyramidal Vision Transformers) Wang et al., 2021. The capacity to model trajectory correlations—sequential attention pathways connecting distant spatial positions through intermediate tokens—represents a critical differentiator between global and windowed attention mechanisms.
Positional embeddings are crucial in giving labels, ordering and a distance metric to the timeseries data. Vision transformers such as SwinT and CSwinT use learned RPE's (Relative Positional Embeddings), have demonstrated superiority over sinusoidal APEs, particularly in timeseries tasks. Moreover, encodings based on Legendre Polynomials (PoPE) and rotation matrices, rotary embeddings (RoPE) Su et al., 2023 have improved the correlation structure and ordering of timeseries data. This work contributes to a broader understanding of how how positional embeddings influence the interpretability and effectiveness of transformers in capturing temporal dependencies. This report is a brief overview submitted in partial fulfillment of the requirements for my research internship at the Physical Research Laboratory

Index Terms: Spatio-temporal Time series Transformers, Multi-headed self-attention, Positional encoding

1 Factorized self-attention in FaViT

Vision transformers (or) Video Vision transformers are analogous to timeseries transformers, with the presence of multiple channels evolving across space and time. Hence, the primary focus for attention mechanisms was on ViT/ViViT literature. ViT offers global attention, but is computationally expensive making it less suitable for long time series. Swin Liu et al., 2021 and CSwin Dong et al., 2022 are more efficient but their local window constraints limit contextual awareness. FaViT Qin et al., 2023 factorizes attention to achieve high expressiveness with low costs, learning mix-grained multi head attention.

The authors of FaViT, Haolin Qine et al., proposed a novel factorization self attention mechanism (FaSA,2021) that enjoys both the advantages of local window cost and long-range dependency modeling capability. By factorizing the conventional attention matrix into sparse sub-attention matrices, FaSA captures long-range dependencies while aggregating mixed-grained information at a computational cost equivalent to the local window-based self attention. In comparison to the baseline model Swin-T, the FaViT-B2 significantly improves classification accuracy by 1%.

The experiments on FaViT focused on modifying the Global Average Pooling/ Max-Pooling layer after the dilated sampling over each attention head. Head grouping:

$$X = \left\{ X_i \in \mathbb{R}^{N \times C'} \mid i = 1, 2, \ldots, G \right\}$$

dilation rates across each head group is given by:

$$D_i = \frac{(S_i - 1)}{(M - 1)}$$

where S_i is the group window size of the i^{th} group and M is the window dimension. The dilated samples in each window are fused to obtain sampled features from each window, and then use the aggregation function. The paper uses CIFAR-100 dataset to test the results of FaViT B0 on different aggregation functions.

Table 1
Comparison of different aggregation operators used for cross-window fusion on CIFAR-100.

Operator	#Param	FLOPs	Top-1 Acc (%)
Pointwise Convolution	3.5 M	0.6 G	68.6
Linear Layer	3.5 M	0.6 G	67.3
Global Average Pooling	3.4 M	0.6 G	68.6
Maximum Pooling (Original)	**3.4 M**	**0.6 G**	**68.9**
ProbSparse Sampling (Top-L^2 keys)	**3.4 M**	**0.6 G**	**70.77**

In ProbSparse attention, which was first introduced by Zhou et al., 2021 in Informer, Kullback–Leibler (KL) divergence is used to quantify the sparsity of the attention distribution produced by each query vector q_i. For a query attending to keys $\{k_j\}_{j=1}^{L_k}$, the attention weight distribution is:

$$P_i(j) = \text{softmax}(q_i k_j^\top)$$

The KL divergence between this distribution and a uniform distribution $U(j) = \frac{1}{L_k}$ is computed as:

$$D_{\text{KL}}(P_i \| U) = \sum_{j=1}^{L_k} P_i(j) \log \left(\frac{P_i(j)}{1/L_k} \right) = \log L_k + \sum_{j=1}^{L_k} P_i(j) \log P_i(j)$$

This provides a measure of how much the attention distribution deviates from uniformity. A higher KL value indicates a sharper distribution and suggests that the query is attending to only a few keys,which attend to selective queries and should be retained for attention computation.

Let L_q be the total number of queries and L_k the number of keys. To reduce the cost of full attention, only the top $u = \log L_q$ queries with the highest KL divergence scores are retained. These selected queries, denoted as $Q_s \in \mathbb{R}^{u \times d}$, are then used for computing attention scores over all keys $K \in \mathbb{R}^{L_k \times d}$.

To further reduce the number of keys, we compute the interaction score between the sampled queries and all keys:

$$S = Q_s K^\top \in \mathbb{R}^{u \times L_k}$$

We then aggregate the scores across queries to obtain a single score per key. This is analogous to average or max pooling over the query dimension. The top L^2 keys are then selected by sorting $\bar{s} \in \mathbb{R}^{L_k}$ and selecting the indices of the largest entries. These top keys can now be used in the final attention computation with all queries, resulting in a sparse yet informative attention pattern.

1.1 Interpretation as Pooling-Based Cross-Window Fusion

This mechanism can be seen as a dynamic, data-driven alternative to fixed operators like average pooling or max pooling. By computing attention via a small set of representative queries and selecting top keys via pooled similarity, this method combines the interpretability of pooling with the adaptability of attention. This exceeds the previous benchmark of using Maximum Pooling aggregator, as ablated by the Qin et al., 2023's FaViT, by 1.87%. Further ablation is being done with the ILSVRC 2012 (ImageNet1K Large Scale Visual Recognition Challenge) dataset, on FaViT B0 and B2 variants to observe the influence of ProbSparsed aggregation in timeseries modelling.

2 Positional Encodings for Improved Correlation Structure

2.1 Momentum based Attention (Tangential space)

The ΔV attention context vectors were introduced as a modification to the standard self-attention mechanism proposed by Vaswani et al.(2017). The core objective of this modification was to enable the model to learn distinct representations for different permutations of the input tokens, effectively addressing the issue of translational invariance, without relying on positional encodings. This idea stems from the perspective centered on token ordering, by encoding the order implicitly through ΔV, the model is encouraged to infer different contextual meanings for different sequences, making the transformer sensitive to token order even in absence of positional embeddings.

$$\Delta x_i = x_i - x_{i-1}$$

For input (x_1, x_2, x_3, x_4):

$$V_1 = x_1 + x_1.x_2(x_2 - x_1) + x_1.x_3(x_3 - x_2) + x_1.x_4(x_4 - x_3)$$

For input (x_2, x_1, x_3, x_4):

$$V_1^{'} = x_2 + x_2.x_1(x_1 - x_2) + x_2.x_3(x_3 - x_1) + x_2.x_4(x_4 - x_3)$$

Table 2
Comparison of ΔV embeddings in attention with ETTh1.

Informer Modification	MSE/MAE
Vanilla Informer	0.519,0.513
Value embedding, ΔV	0.8696,0.7147
Value, temporal embedding, ΔV	0.914,0.723
All embeddings, ΔV	0.584,0.5967
All embeddings, ΔV and deep MLP	**0.5166,0.524**

Table 3
Comparison of ΔV embeddings in attention with ETTm1.

Informer Modification	MSE/MAE
Vanilla Informer	0.3065,0.3709
Value embedding, ΔV	0.7356,0.5960
Value,temporal embedding, ΔV	0.6265,0.5803
All embeddings, ΔV	**0.2694,0.3525**

which is different from V_1. The ablation was done using Zhou et al., 2021's Informer on ETTh1 and ETTm1.

Similar trends were observed with ETTm1 dataset, proving that the tangential space embeddings only give ordering to the inputs and is far from replacing positional embeddings all together (Table 2, Table 3). But in the last trial, when used with position embeddings, it is observed to improve the accuracy of time series prediction.

2.2 Ablation with Legendre Polynomial based Position Embeddings

At higher dimensions, a near-perfect correlation (>0.999 percent) is observed, even for token positions that are far apart in the sentence in sinusoidal APEs. The paper on Orthogonal Polynomial Based Positional Encoding (PoPE) , introduced a unique class of orthogonal polynomials referred to as Legendre polynomials. Legendre polynomials $P_n(x)$ are special category of orthogonal polynomials. They appear in numerous engineering and physics problems, especially related to spherical harmonics (Dassios, 2012).

$$\mathbf{P}_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} \left(x^2 - 1\right)^n$$

Informer models were tested upon, using these polynomials with varying orders as positional encodings for the inputs.

Table 4
Comparison of Legendre Polynomial embeddings with ETTh1.

Informer Modification	MSE/MAE
Vanilla Informer	0.519,0.513
With PoPE embedding	0.950,0.7031
With PoPE, ΔV	0.8216,0.7031

2.3 Analysis of the cross product terms and positional encoding's in self-attention

$$q_m^\top k_n = \underbrace{x_m^\top W_q^\top W_k x_n}_{\text{token-token}} + \underbrace{x_m^\top W_q^\top W_k p_n}_{\text{token-pos}} + \underbrace{p_m^\top W_q^\top W_k x_n}_{\text{pos-token}} + \underbrace{p_m^\top W_q^\top W_k p_n}_{\text{pos-pos}}$$

The above expansion shows how the word embedding and the positional embedding are projected and queried in the attention module. We can see that there are four terms after the expansion: token-to-token, token-to-position, position-to-token, and position-to-position correlations. To test the contribution of cross product terms (the token-pos and the pos-token), Informer architecture was modified to add position embeddings in each encoder and decoder attention layer. Ke et al., 2021 in their paper on TuPE, proposed the possibility of removal of cross-terms in the attention whereas He et al.(2020) argued that the relative positional information can only be modeled using middle two terms.

Trained Informer with Full Attention (removed ProbSparsing for testing), with the attention pattern being:

$$q_m^\top k_n = (x_m^\top W_q^\top W_k p_n + p_m^\top W_q^\top W_k x_n)$$

Sinusoidal position embeddings were used for the ablation.

Table 5
Analysis of token-pos, pos-token cross terms.

Informer Modification	MSE/MAE
Vanilla Informer	0.519,0.513
Informer with cross terms, full-attention	**0.559,0.547**

This means the cross terms between tokens and positions in self attention are fundamental to how transformers encode and process sequences, allowing them to learn cross-semantic ordering relationships.

Further analysis on other terms shows that input embeddings without positional information performs poorly as expected (Table 6) due to the inherently permutation-invariant nature of multi-head self-attention. In the absence of a positional encoding, the transformer treats the inputs as a 'set' rather than a 'sequence' which is the way of processing in other time-series models such as RNNs, LSTMs etc. This leads to to a degraded performance on tasks requiring order awareness, such as time-series forecasting.

The pos-pos term, captures pure spatial correlations, independent of the content, often underappreciated. It reflects how purely positional alignment contributes to attention (Table 6). It term encodes structural priors, that closer positions should attend more, and farther positions attention span should decay in a patricular manner.

Table 6
Analysis of pos-pos, token-token terms.

Informer Modification	MSE/MAE
Informer without position embedding	0.8820,0.7249
Informer with pos-pos terms, full-attention	0.5846,0.5742

3 Tools and Frameworks, Training Strategies

PyTorch 2.0.1 + cu118, T4 GPU (ETTh1 and ETTm1), Vikram-1000 HPC: A100 GPU, GPU-long nodes (CIFAR-100, ImageNet1K).

Training with ETTh1 and ETTm1: Multivariate-to-Multivariate prediction, sequence length=64, label length for decoder=48, prediction length= 24 for ETTh1 (1 day) and 96 (1.5 hours) for ETTm1, d_{model} = 512, batch size=32, learning rate= 10^{-4}, encoders=2, decoders=1, ProbSparse attention (except Cross product analysis), embed= timeF, distillation= True, 6 epochs with T4 GPU.

Training with CIFAR-100, ImageNet1K: Used AdamW optimizer with 300 epochs including initial 10 warm-up epochs and the final 10 cool-down epochs (for fair comparison with original FaViT). A cosine decay learning rate scheduler is applied, reducing the learning rate by a factor of 10 every 30 epochs, beginning from a base learning rate of 0.001. Weight decay is set to 0.05, and input images are resized to 224×224 (for CIFAR 100, it was 32x32 originally).Data augmentations and regularization methods align with those employed in FaViT. Used a mini-batch size of 128 samples and leveraged the computational power of NVIDIA A100 through HPC server.

4 Acknowledgements

I would like to express my sincere gratitude to Mr. Akash Ganguly and Mr. Shivansh Verma at the Physical Research Laboratory (PRL) for their guidance and continuous support throughout my research tenure. I am also thankful for giving me access to high-performance computing (HPC) resources, which played a critical role in facilitating the ablation.

References

Dong, Xiaoyi et al. (2022). *CSWin Transformer: A General Vision Transformer Backbone with Cross-Shaped Windows*. arXiv: 2107.00652 [cs.CV]. URL: https://arxiv.org/abs/2107.00652.

Ke, Guolin, Di He, and Tie-Yan Liu (2021). *Rethinking Positional Encoding in Language Pre-training*. arXiv: 2006.15595 [cs.CL]. URL: https://arxiv.org/abs/2006.15595.

Liu, Ze et al. (2021). *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. arXiv: 2103.14030 [cs.CV]. URL: https://arxiv.org/abs/2103.14030.

Qin, Haolin et al. (2023). *Factorization Vision Transformer: Modeling Long Range Dependency with Local Window Cost*. arXiv: 2312.08614 [cs.CV]. URL: https://arxiv.org/abs/2312.08614.

Su, Jianlin et al. (2023). *RoFormer: Enhanced Transformer with Rotary Position Embedding*. arXiv: 2104.09864 [cs.CL]. URL: https://arxiv.org/abs/2104.09864.

Touvron, Hugo et al. (2021). *Training data-efficient image transformers distillation through attention*. arXiv: 2012.12877 [cs.CV]. URL: https://arxiv.org/abs/2012.12877.

Wang, Wenhai et al. (2021). *Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions*. arXiv: 2102.12122 [cs.CV]. URL: https://arxiv.org/abs/2102.12122.

Wu, Haiping et al. (2021). *CvT: Introducing Convolutions to Vision Transformers*. arXiv: 2103.15808 [cs.CV]. URL: https://arxiv.org/abs/2103.15808.

Zhou, Haoyi et al. (2021). *Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting*. arXiv: 2012.07436 [cs.LG]. URL: https://arxiv.org/abs/2012.07436.