SAMformer: Unlocking the Potential of Transformers in Time Series Forecasting

Ambroise Odonnat
Huawei Noah's Ark Lab

Criteo AI Lab

July 15, 2024

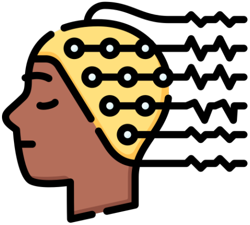
Outline

Outline

1. Introduction
2. Failure of Transformers
3. SAMformer
4. Experiments
5. Take Home Message

Time Series Data

In many applications, data are gathered sequentially.



Multivariate Long-Term Forecasting

Goal $\rightarrow$ Analysing time series data to predict future trends.

- Forecast of ECG recording to predict cardiac arrhythmia,
- Electricity consumption forecasting to match future demand,
- Predicting stock market prices.

Challenges

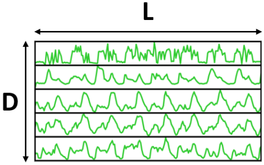
1. Long-term temporal dependencies,
2. Highly correlated features.

Multivariate Long-Term Forecasting

D-dimensional time series of length $L \to$ predict next H values.

- Training set of N observations $(\{\mathbf{X}^{(i)}\}_{i=0}^{N}, \{\mathbf{Y}^{(i)}\}_{i=0}^{N})$,
- Find predictor $f_{\boldsymbol{\omega}} \colon \mathbb{R}^{D \times L} \to \mathbb{R}^{D \times H}$ that minimizes the MSE

$$\mathcal{L}_{\text{train}}(\boldsymbol{\omega}) = \frac{1}{ND} \sum_{i=0}^{N} \|\mathbf{Y}^{(i)} - f_{\boldsymbol{\omega}}(\mathbf{X}^{(i)})\|_{\mathrm{F}}^{2}.$$

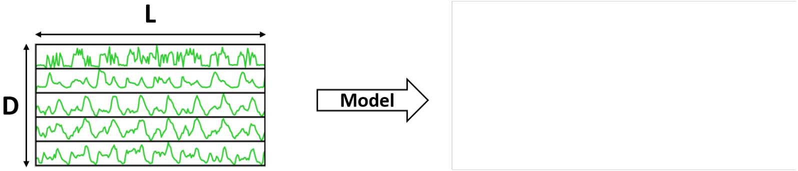


Multivariate Long-Term Forecasting

D-dimensional time series of length $L \to$ predict next H values.

- Training set of N observations $(\{\mathbf{X}^{(i)}\}_{i=0}^{N}, \{\mathbf{Y}^{(i)}\}_{i=0}^{N})$,
- Find predictor $f_{\boldsymbol{\omega}} \colon \mathbb{R}^{D \times L} \to \mathbb{R}^{D \times H}$ that minimizes the MSE

$$\mathcal{L}_{\text{train}}(\boldsymbol{\omega}) = \frac{1}{ND} \sum_{i=0}^{N} \|\mathbf{Y}^{(i)} - f_{\boldsymbol{\omega}}(\mathbf{X}^{(i)})\|_{\mathrm{F}}^{2}.$$

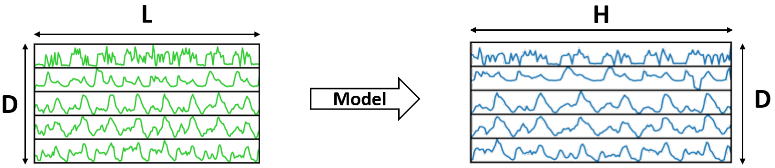


Multivariate Long-Term Forecasting

D-dimensional time series of length $L \to$ predict next H values.

- Training set of N observations $(\{\mathbf{X}^{(i)}\}_{i=0}^{N}, \{\mathbf{Y}^{(i)}\}_{i=0}^{N})$,
- Find predictor $f_{\boldsymbol{\omega}} \colon \mathbb{R}^{D \times L} \to \mathbb{R}^{D \times H}$ that minimizes the MSE

$$\mathcal{L}_{\text{train}}(\boldsymbol{\omega}) = \frac{1}{ND} \sum_{i=0}^{N} \|\mathbf{Y}^{(i)} - f_{\boldsymbol{\omega}}(\mathbf{X}^{(i)})\|_{\text{F}}^2.$$



Standard methods:

- AR models (ARIMA)
- Seasonal naive

Deep learning methods

- RNN, CNN
- Transformer-based models

Outline

Transformers for Time Series Forecasting

Motivation

- Transformers tailored to deal with sequential data,
- Impressive results in NLP and Computer Vision.

Main challenges

1. Quadratic computation of self-attention,
2. Complex long-term dependencies.

Main challenges

1. Quadratic computation of self-attention,
2. Complex long-term dependencies.

Transformers for Time Series Forecasting

Main challenges

❶ Quadratic complexity of self-attention

- Sparse attention: LogTrans [5], Informer [13]
- Modified attention: Pyraformer [6]

❷ Complex long-term dependencies

- Decomposition scheme: Autoformer [10], Pyraformer [6]
- Fourier domain: FEDformer [14]

It leads to a wide range of Anything-formers with heavy and complex implementation and many parameters.

Failure of Transformers

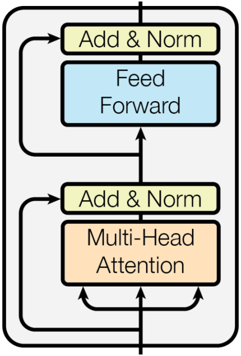
[11] showed that linear models outperform SOTA Anything-former.

Outline

1. Introduction
2. Failure of Transformers
3. SAMformer
4. Experiments
5. Take Home Message

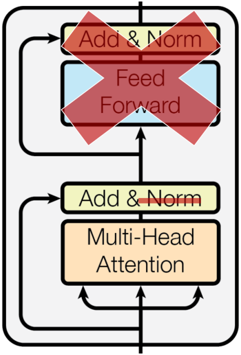
Starting Point: Linear Regression

- Generate toy data according to $\mathbf{Y} = \mathbf{X}\mathbf{W}_{\text{toy}} + \varepsilon$,
- Designing the simplest `Transformer` possible.



Starting Point: Linear Regression

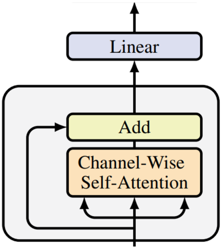
- Generate toy data according to $\mathbf{Y} = \mathbf{X}\mathbf{W}_{\text{toy}} + \varepsilon$,
- Designing the simplest `Transformer` possible.



- Generate toy data according to $\mathbf{Y} = \mathbf{X}\mathbf{W}_{\text{toy}} + \varepsilon$,
- Designing the simplest `Transformer` possible.

$$\mathbf{A}(\mathbf{X}) = \texttt{softmax}\left(\frac{\mathbf{X}\mathbf{W}_{\mathbf{Q}}\mathbf{W}_{\mathbf{K}}^{\top}\mathbf{X}^{\top}}{\sqrt{d_{\mathrm{m}}}}\right)$$

$$f(\mathbf{X}) = [\mathbf{X} + \mathbf{A}(\mathbf{X})\mathbf{X}\mathbf{W}_V\mathbf{W}_O]\mathbf{W}$$

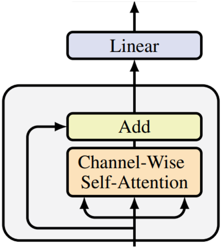


Starting Point: Linear Regression

- Generate toy data according to $\mathbf{Y} = \mathbf{X}\mathbf{W}_{\text{toy}} + \varepsilon$,
- Designing the simplest `Transformer` possible.

$$\mathbf{A}(\mathbf{X}) = \texttt{softmax}\left(\frac{\mathbf{X}\mathbf{W}_{\mathbf{Q}}\mathbf{W}_{\mathbf{K}}^{\top}\mathbf{X}^{\top}}{\sqrt{d_{\mathrm{m}}}}\right)$$

$$f(\mathbf{X}) = [\mathbf{X} + \mathbf{A}(\mathbf{X})\mathbf{X}\mathbf{W}_V\mathbf{W}_O]\mathbf{W}$$



Theorem (Ilbert, O., Feofanov et al.)

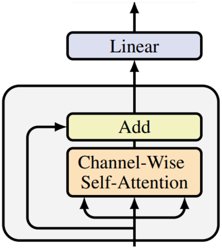
Given fixed attention weights $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V, \mathbf{W}_O$, there exists an infinity of optimal $\mathbf{W}$ reaching the oracle, i.e., $f(\mathbf{X}) = \mathbf{X}\mathbf{W}_{\text{toy}}$.

Starting Point: Linear Regression

- Generate toy data according to $\mathbf{Y} = \mathbf{X}\mathbf{W}_{\text{toy}} + \varepsilon$,
- Designing the simplest `Transformer` possible

$$\mathbf{A}(\mathbf{X}) = \texttt{softmax}\left(\frac{\mathbf{X}\mathbf{W}_{\mathbf{Q}}\mathbf{W}_{\mathbf{K}}^{\top}\mathbf{X}^{\top}}{\sqrt{d_{\text{m}}}}\right)$$

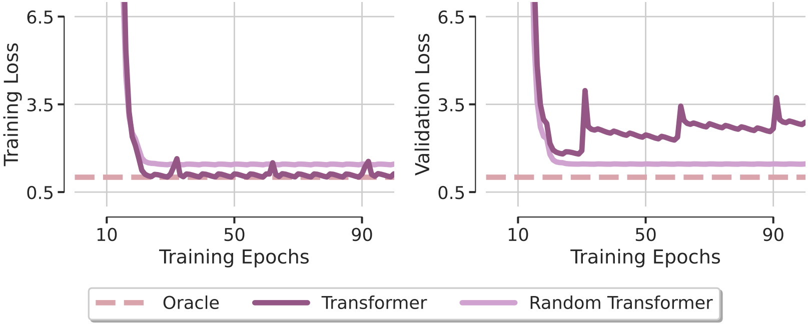
$$f(\mathbf{X}) = [\mathbf{X} + \mathbf{A}(\mathbf{X})\mathbf{X}\mathbf{W}_V\mathbf{W}_O]\mathbf{W}$$



In theory, our simplistic Transformer can be optimal. Is this the case in practice?

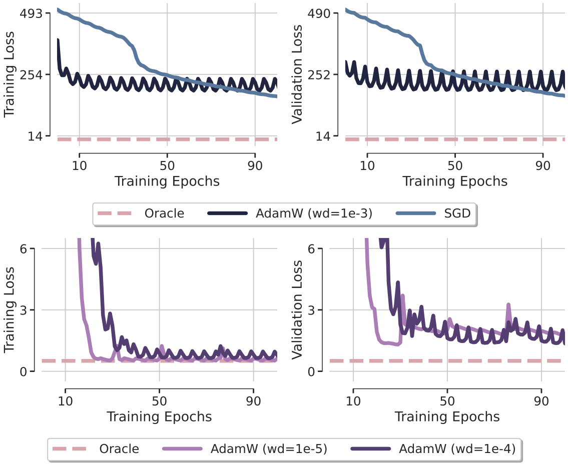
Poor Generalization

- Oracle: optimal solution,
- `Transformer` with $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V, \mathbf{W}_O, \mathbf{W}$ trainable,
- `Random Transformer`: only $\mathbf{W}$ is trainable.



Despite its simplicity, Transformer overfits a lot. Fixing the attention weight improves generalization.

Similar Behaviour with other Optimizers

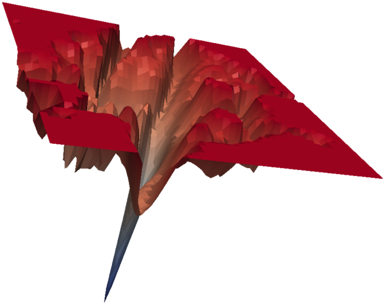


Poor generalization of Transformer with SGD, Adam, and AdamW.

Trainability Issues due to the Attention

Hypothesis from NLP and Computer Vision

- Transformers have **sharp loss landscape** [2]
  - Convergence to sharp minima $\rightarrow$ poor generalization,
  - Computed as $\lambda_{\max}$, the largest singular value of the Hessian.

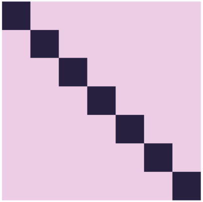


Trainability Issues due to the Attention

Hypothesis from NLP and Computer Vision

- Transformers have **sharp loss landscape** [2],
- Attention suffers from **entropy collapse** [12].
  - Entropy = average entropy of the rows,
  - It causes training instability,
  - [12] $\rightarrow$ entropy collapse and sharpness appear in tandem.

$\sim$ Uniform $\rightarrow$ high entropy.

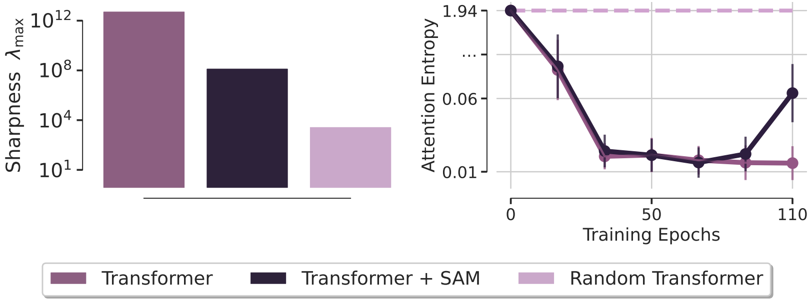


Diagonal $\rightarrow$ low entropy.

Trainability Issues due to the Attention

Hypothesis from NLP and Computer Vision

- Transformers have **sharp loss landscape** [2],
- Attention suffers from **entropy collapse** [12].



Training the attention induces an entropy collapse and a sharp loss landscape.

Existing Solutions

❶ **σReparam** [12]

Replace each weight matrix $\mathbf{W}$ by

$$\widehat{\mathbf{W}} = \frac{\gamma}{\|\mathbf{W}\|_2}\mathbf{W}, \text{ with } \gamma \in \mathbb{R} \text{ learnable },$$

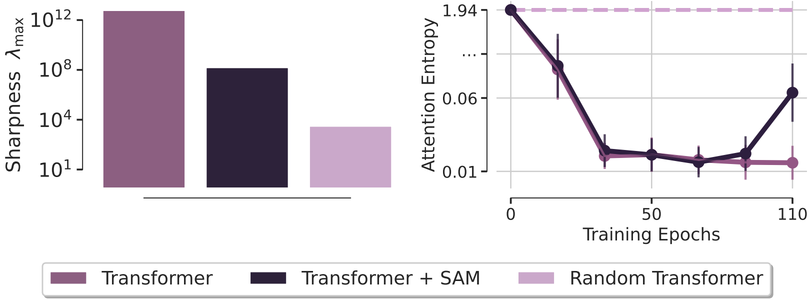
❷ **Sharpness-Aware Minimization (SAM)** [3]

Replace the training loss $\mathcal{L}_{\text{train}}$ by

$$\mathcal{L}_{\text{train}}^{\text{SAM}}(\boldsymbol{\omega}) = \max_{\|\boldsymbol{\varepsilon}\|<\rho} \mathcal{L}_{\text{train}}(\boldsymbol{\omega}+\boldsymbol{\varepsilon}) \approx \mathcal{L}_{\text{train}}\left(\boldsymbol{\omega} + \rho \cdot \frac{\nabla \mathcal{L}_{\text{train}}(\boldsymbol{\omega})}{\|\nabla \mathcal{L}_{\text{train}}(\boldsymbol{\omega})\|_2}\right).$$

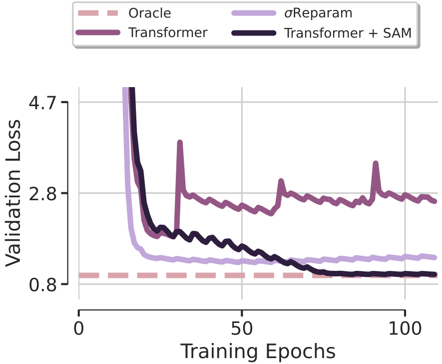
σReparam doesn't solve the problem, but SAM does.

Contrary to NLP and Computer Vision, entropy collapse seems benign in time series forecasting while sharpness is harmful.



σReparam doesn't solve the problem, but SAM does.

Contrary to NLP and Computer Vision, entropy collapse seems benign in time series forecasting while sharpness is harmful.



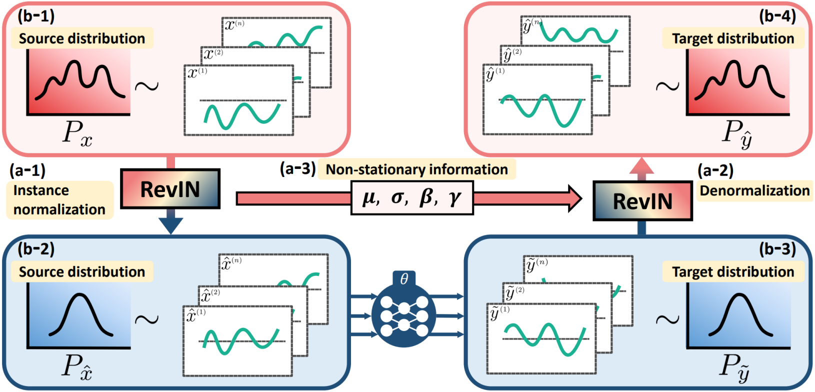
σReparam helps but is not sufficient while using SAM leads to the optimal solution (oracle).

Good Practices: RevIN

RevIN [4] $\rightarrow$ Reduce distribution shift between input and target.

- Multivariate time series $\mathbf{X} \in \mathbb{R}^{D \times L}$, learnable γ, β.
- Normalize each feature $\mathbf{X}_i \leftarrow \tilde{\mathbf{X}}_i = \frac{\mathbf{X}_i - \mu}{\sigma} \leftarrow \gamma \tilde{\mathbf{X}}_i + \beta$,
- Apply model on multivariate time series $\tilde{\mathbf{Y}} = f(\tilde{\mathbf{X}})$,
- Denormalize each feature $\tilde{\mathbf{Y}}_i \leftarrow \hat{\mathbf{Y}}_i = \frac{\tilde{\mathbf{Y}}_i - \beta}{\gamma} \leftarrow \sigma \hat{\mathbf{Y}}_i + \mu$.

RevIN [4] $\rightarrow$ Reduce distribution shift between input and target.

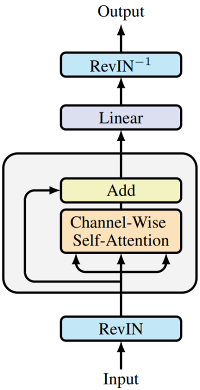


SAMformer: SAM & Channel-Wise Attention

- Input $\mathbf{X} \in \mathbb{R}^{D \times L}$, output $f(\mathbf{X}) \in \mathbb{R}^{D \times H}$,
- Reduce distribution shift with **RevIN** [4],
- **Channel-wise attention** $\mathbf{A}(\mathbf{X}) \in \mathbb{R}^{D \times D}$,
- **Smooth** loss landscape with SAM [3].

$$\mathbf{A}(\mathbf{X}) = \mathtt{softmax}\left(\frac{\mathbf{X}\mathbf{W}_{\mathbf{Q}}\mathbf{W}_{\mathbf{K}}^{\top}\mathbf{X}^{\top}}{\sqrt{d_{\mathrm{m}}}}\right)$$

$$f(\mathbf{X}) = [\mathbf{X} + \mathbf{A}(\mathbf{X})\mathbf{X}\mathbf{W}_V\mathbf{W}_O]\mathbf{W}$$



SAMformer is a shallow transformer trained with SAM.
→ One head, one encoder, 15 lines of code!

Outline

Baselines & Datasets

All-MLP model (2023): TSMixer [1].

Transformers (2021-2022): FEDformer [14], Autoformer [10].

Recent Transformers (2023-2024): iTransformer [7], PatchTST [8].

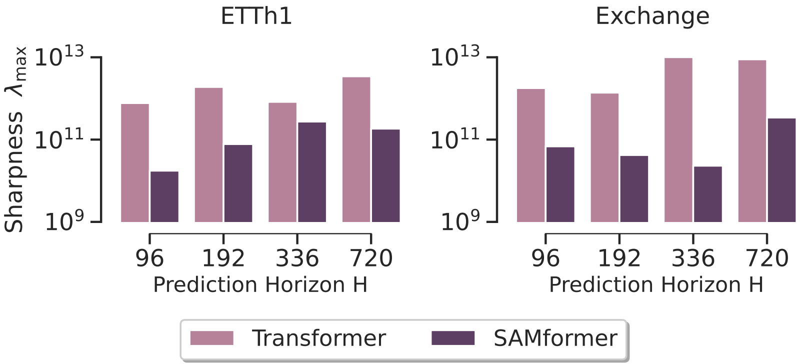
Dataset	ETTh1/ETTh2	ETTm1/ETTm2	Electricity	Exchange	Traffic	Weather
# features	7	7	321	8	862	21
# time steps	17420	69680	26304	7588	17544	52696
Granularity	1 hour	15 minutes	1 hour	1 day	1 hour	10 minutes

SOTA Performance

Dataset	**SAMformer** -	iTransformer 2024	PatchTST 2023	TSMixer 2023	FEDformer 2022	Autoformer 2021
ETTh1	**0.410**	0.454	0.469	0.437	0.440	0.496
ETTh2	**0.344**	0.383	0.387	0.357	0.437	0.450
ETTm1	**0.373**	0.407	0.387	0.385	0.448	0.588
ETTm2	**0.269**	0.288	0.281	0.289	0.305	0.327
Traffic	**0.425**	0.428	0.481	0.620	0.610	0.628
Weather	0.260	**0.258**	0.259	0.267	0.309	0.338
Overall improvement		**6.58%**	**8.79%**	**13.2%**	**22.5%**	**35.9%**

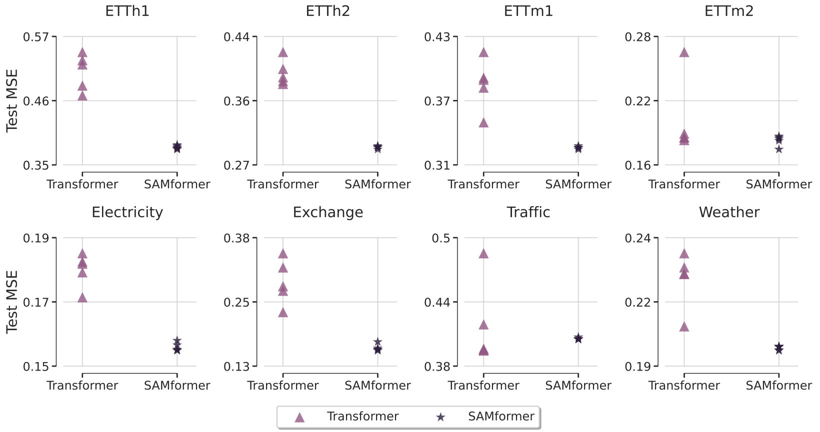
SAMformer outperforms all baselines while having significantly fewer parameters.

Smoother Loss Landscape



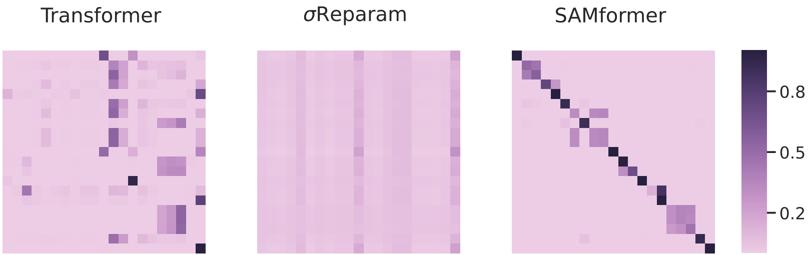
SAM provides a smoother loss landscape ...

Better Generalization



... leading to better generalization and robustness.

Better Signal Propagation

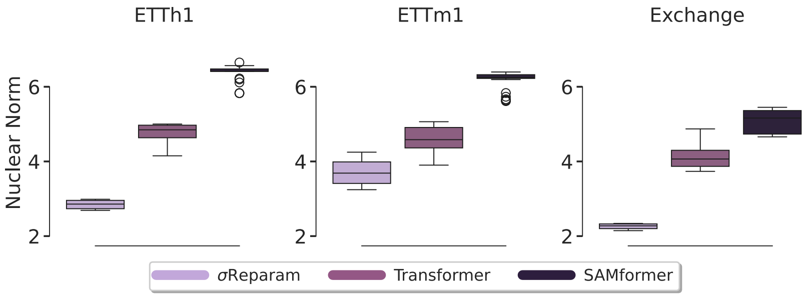


Channel-wise attention improves the propagation of the signal with self-feature correlations as in ViTs.

Theorem (Ilbert, O., Feofanov et al.)

Applying σReparam [12] leads to **attention rank collapse**.

$$\|\mathbf{X}\mathbf{W}_Q\mathbf{W}_K^\top\mathbf{X}^\top\|_* \quad \leq \quad \underbrace{\|\mathbf{W}_Q\mathbf{W}_K^\top\|_2}_{\text{goes to } 0 \text{ with } \sigma\text{Reparam}} \quad \|\mathbf{X}\|_{\mathrm{F}}^2.$$



Strong Competitor to MOIRAI

- MOIRAI [9]: foundation model trained on **27B samples**,
- Nb. params: small (**14M**), base (**91M**) and large (**314M**).

Dataset	Full-shot	Zero-shot		
	SAMformer	$\mathtt{MOIRAI}_{\mathtt{Small}}$	$\mathtt{MOIRAI}_{\mathtt{Base}}$	$\mathtt{MOIRAI}_{\mathtt{Large}}$
ETTh1	<u>0.410</u>	**0.400**	0.434	0.510
ETTh2	<u>0.344</u>	**0.341**	0.345	0.354
ETTm1	**0.373**	0.448	<u>0.381</u>	0.390
ETTm2	**0.269**	0.300	<u>0.272</u>	0.276
Electricity	**0.181**	0.233	<u>0.188</u>	<u>0.188</u>
Weather	0.260	<u>0.242</u>	**0.238**	0.259
Overall MSE improvement		6.9%	1.1%	7.6%

SAMformer outperforms MOIRAI while having significantly fewer parameters!

Outline

Easier, Better, Faster, Stronger

Findings

- Transformer failure $\rightarrow$ trainability issues of the attention,
- In time series forecasting, entropy collapse is benign,
- But sharpness prevents good generalization.

Proposal

- **SAMformer**: RevIN + channel-wise attention + SAM,
- SOTA and lightest model,
- Strong competitor to MOIRAI [9].

To Know More

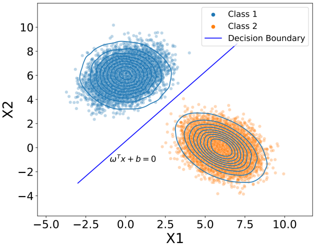
This work has been accepted as an **Oral at ICML 2024, Vienna**. You may find the links to the paper and the code below. To know more about my research, check my website: ambroiseodt.github.io and feel free to contact me.

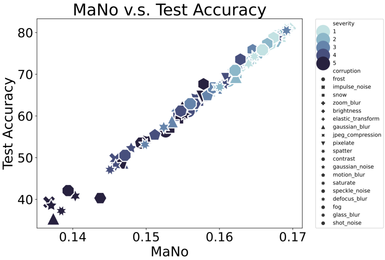
- ⋆ Paper: https://arxiv.org/pdf/2402.10198
- ⋆ Code: https://github.com/romilbert/samformer

Acknowledgements

This project was led by Romain Ilbert and myself with our co-authors Vasilii Feofanov, Aladin Virmaux, Giuseppe Paolo, Themis Palpanas, and Ievgen Redko.

MANO: Exploiting Matrix Norm for Unsupervised Accuracy Estimation Under Distribution Shifts





https://arxiv.org/pdf/2405.18979

Thanks for your attention !

References I

[1] Chen, S.-A., Li, C.-L., Arik, S. O., Yoder, N. C., and Pfister, T. (2023). TSMixer: An all-MLP architecture for time series forecasting. *Transactions on Machine Learning Research*.

[2] Chen, X., Hsieh, C.-J., and Gong, B. (2022). When vision transformers outperform resnets without pre-training or strong data augmentations. In *International Conference on Learning Representations*.

[3] Foret, P., Kleiner, A., Mobahi, H., and Neyshabur, B. (2021). Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*.

[4] Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.-H., and Choo, J. (2021). Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*.

References II

[5] Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

[6] Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., and Dustdar, S. (2022). Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations*.

[7] Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., and Long, M. (2024). itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*.

References III

[8] Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. (2023). A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*.

[9] Woo, G., Liu, C., Kumar, A., Xiong, C., Savarese, S., and Sahoo, D. (2024). Unified training of universal time series forecasting transformers.

[10] Wu, H., Xu, J., Wang, J., and Long, M. (2021). Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*.

[11] Zeng, A., Chen, M., Zhang, L., and Xu, Q. (2023). Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[12] Zhai, S., Likhomanenko, T., Littwin, E., Busbridge, D., Ramapuram, J., Zhang, Y., Gu, J., and Susskind, J. M. (2023). Stabilizing transformer training by preventing attention entropy collapse. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 40770–40803. PMLR.

[13] Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. (2021). Informer: Beyond efficient transformer for long sequence time-series forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, volume 35, pages 11106–11115. AAAI Press.

[14] Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., and Jin, R. (2022). FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proc. 39th International Conference on Machine Learning (ICML 2022)*.