Sig-Patchformer: A Path Signature Based Transformer for Efficient Time Series Forecasting

Anonymous submission

Abstract

Time series forecasting is essential in finance, healthcare, and climate science, but existing transformer models face high computational costs, large memory use, and long training times when processing long sequences. To address these challenges, we propose Sig-Patchformer, a decoder-only transformer that integrates path signatures and their logarithmic form. The model performs feature engineering by extracting higher-order interactions, provides positional encoding by capturing the order and direction of data, and reduces dimensionality to create compact representations, making it suitable for large-scale forecasting. Unlike traditional transformers, Sig-Patchformer does not rely on additional positional encodings, since log signatures inherently preserve sequence order. This design improves accuracy while significantly lowering training time and memory consumption. Results on five benchmark datasets against multiple baselines demonstrate that Sig-Patchformer achieves both high accuracy and efficiency, with faster training times and lower memory usage, making it a practical solution for large-scale forecasting in big data environments.

Introduction

Time series forecasting, which involves predicting future values from historical sequences, is crucial for decision-making in domains such as financial markets, healthcare systems, and climate monitoring. With the growth of sensing and logging technologies, organizations now collect high-frequency, multivariate data streams, turning forecasting into a big data challenge. This shift demands models capable of processing long contexts with numerous correlated signals, while adhering to strict latency and cost constraints.

Historically, ARIMA, SARIMA, and other machine learning techniques were used for time series prediction. However, with advancements in artificial intelligence, deep neural networks and their variants have been shown to yield superior results. The introduction of the transformer model—designed for processing long sequences—has greatly enhanced time series forecasting by treating time series as sequential inputs, leading to improved prediction accuracy.

Despite the significant progress transformers have made in sequence prediction, they face scaling limitations, especially for long-term forecasting. The core attention mechanism of transformers exhibits quadratic complexity with respect to sequence length, which results in inflated computational costs, higher memory consumption, and longer training times as the input size increases. These issues are compounded by deeper model architectures and larger hidden layer sizes. These limitations make transformers less suitable for real-time forecasting environments, particularly where low latency and minimal storage are essential, such as in Internet-of-Things (IoT) devices. Furthermore, transformers are sensitive to noisy measurements common in real-world data, often requiring intensive regularization or extensive data preprocessing. To address these challenges, we leverage path signatures and their logarithmic form. Path signatures provide compact representations of sequences by summarizing their evolution through ordered iterated integrals. Log signatures, which are the Lie algebra coordinates obtained by taking the logarithm of the path signature, offer a more compact representation that reduces redundancy and captures dependencies between features. They serve as powerful tools for feature extraction while performing dimensionality reduction. Moreover, they effectively capture the position and direction of points, eliminating the need for position embeddings. In this paper, we propose Sig-Patchformer, a novel transformer-based architecture that integrates log signatures to improve accuracy, and reduce training time and memory consumption. Our model divides the input into non-overlapping patches and computes a log signature vector for each patch, capturing important features and positional information. Simultaneously, each patch is processed by a lightweight one-dimensional convolutional neural network (CNN) that projects the input into the same dimension as the log signature output. This step is necessary because path signatures capture direction, not position, and thus help encode where a point is located. The resulting output is passed to a decoder-only causal transformer for forecasting. The main efficiency gains come from two sources. First, patching shortens the sequence length processed by the decoder, directly reducing attention overhead. Second, log signature tokens are compact yet highly expressive, allowing the transformer to operate effectively with fewer layers while maintaining accuracy. From our experiments on multiple benchmark datasets and comparisons with current transformer models, we observe that Sig-Patchformer outperforms these models in most cases, with lower memory consumption and reduced training time.

Related Works

Time series forecasting has made significant strides by blending advanced machine learning techniques, especially those combining path signatures and Transformer architectures to handle long sequences more effectively. The signature method, a key tool in this field, captures the unique patterns and shapes of data paths—think of it as a way to summarize a journey through numbers into a compact form. This approach, introduced as a powerful dimension reduction technique for multi-dimensional data, lays a foundation for understanding complex sequences (Chevyrev and Kormilitzin 2016). Building on this, researchers have adapted it for multivariate time series, creating a flexible framework that turns data into paths and extracts signatures. This method has proven its worth in tasks like classification, showing it can work across different domains without needing heavy customization (Morrill et al. 2020).

Transformer models have brought a fresh perspective to long sequence time-series forecasting (LSTF) by tackling issues like slow processing times and the challenge of spotting patterns over long stretches. The Informer model, for instance, introduced a smarter attention mechanism called ProbSparse, which reduces the processing time from O(L^2) to O(L log L) and uses a generative decoder to predict long sequences efficiently. This has outperformed earlier methods on large datasets, making it a go-to for researchers (Zhou et al. 2020). The Autoformer takes this further by breaking down time series into simpler parts and using an Auto-Correlation technique to focus on key sub-series, achieving a remarkable 38% accuracy boost over previous benchmarks (Wu et al. 2021). However, some experts argue that Transformers might not always be the best fit, pointing out that their self-attention, which doesn't care about the order of data, can miss critical time-based clues—sometimes even simple linear models do better (Zeng et al. 2022).

To address these shortcomings, a patch-based approach has gained traction, where time series are split into smaller, manageable chunks or "patches" as input tokens. Imagine treating a time series like a story broken into 64-word segments—researchers have shown this preserves local details, cuts down the attention computation by half, and allows the model to look back further, leading to better long-term predictions and even self-learning capabilities (Nie et al. 2022). A decoder-only foundation model simplifies this further by pretraining on vast data, enabling it to forecast across different lengths without starting from scratch each time (Das et al. 2023). The Pathformer model goes a step further, using multi-scale analysis with adaptive pathways to balance global trends and local details, outperforming top models on real-world data with impressive adaptability (Chen et al. 2024).

Adding external data, or exogenous variables, into the mix has also improved forecasting. The TimeXer model uses patch-wise attention for internal data and cross-attention for external factors, plus a global token to tie it all together, consistently topping benchmarks with scalable results (Wang et al. 2024). Another approach with patch-based MLPs focuses on separating smooth trends from noise and emphasizing interactions between variables, which has boosted multivariate forecasting performance (Tang and Zhang 2024). To handle the computational load of long sequences, a technique called token merging for Transformers and state-space models applies local adjustments with a causal focus, speeding things up by up to 5400

Path signatures have also been woven into Transformer designs for a lighter, continuous approach. The Rough Transformer enhances attention with multi-view signatures, capturing patterns at different scales and frequencies, which beats standard Transformers at a lower cost (Moreno-Pino et al. 2024). The Fredformer counters bias toward dominant data frequencies by learning evenly across all bands, improving accuracy on real-world datasets with a streamlined version (Piao et al. 2024). Some studies even suggest that language model components might be overkill—stripping them away or sticking to basic attention with patching often works just as well (Tan et al. 2024).

Other multi-scale models, like DRFormer, use dynamic tokenizers and position encoding to catch hierarchical patterns, outperforming others on long forecasts (Ding et al. 2024). A knowledge-enhanced Transformer taps into knowledge graphs to understand variable relationships, shining on datasets like Weather and ETT (Kakde et al. 2024). A deeper analysis warns that Transformers can struggle with generalization due to uneven attention learning, especially on inconsistent predictions, suggesting a kernel-based rethink might help (Ke et al. 2025). The Skip-Timeformer explores skip-time interactions for long sequences, raising questions about Transformer fit while offering efficient alternatives (Zhang, Wang, and Zhang 2024).

Beyond forecasting, stable prefix policies in multi-agent reinforcement learning reduce exploration noise, offering stability lessons applicable to time series (Deng, Wang, and Zhang 2024). The PaTH Attention method uses Householder transformations for position encoding, which could enhance time series awareness (Yang et al. 2025). Lastly, Powerformer's weighted causal attention prioritizes key dependencies, setting a new standard in time-series forecasting (Hegazy, Mahoney, and Erichson 2025).

Together, these advancements highlight the potential of combining efficient patching with path-signature representations in Transformers, setting the stage for our proposed SigPatchformer, which aims to tackle the challenges of long-term forecasting with a humanized, practical approach.

Methodology

Path Signature

A path signature can be defined as a mathematical representation of sequential data by encoding it as iterated integrals that capture order and interactions between features. For a path $X : [0,1] \to \mathbb{R}^d$, the truncated signature up to depth m is defined as

$$S(X) = \left(1, \int dX, \int dX^{\otimes 2}, \ldots, \int dX^{\otimes m}\right), \qquad (1)$$

where d is the input dimension, and the higher-order iterated integrals encode increasingly complex interactions of increments in the path.

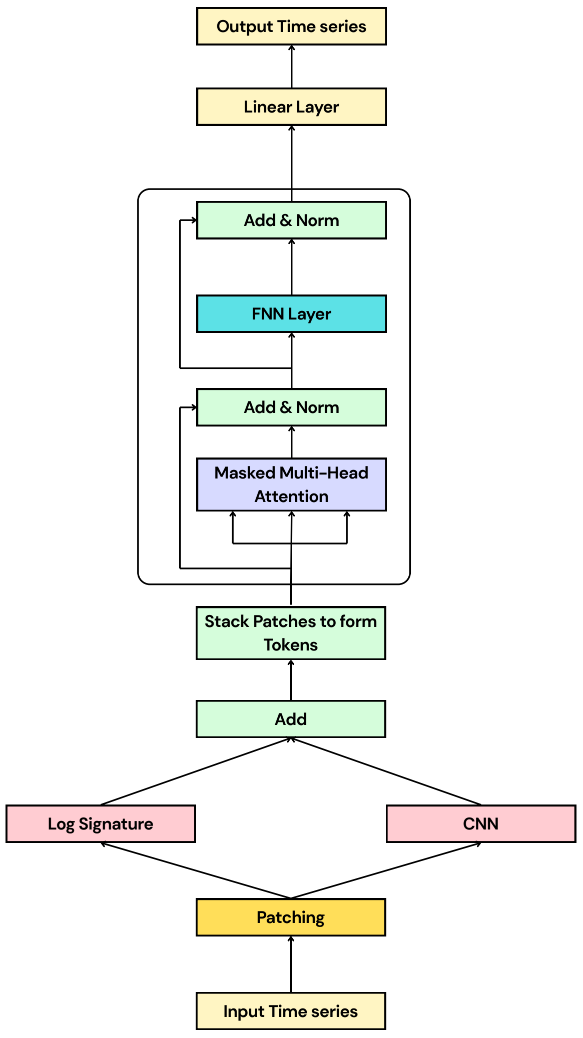

Figure 1: Overview of the transformer-based forecasting model.

To reduce the dimensionality of this representation while preserving structural information, we employ the log-signature, which maps the signature into a Linear algebra,

$$\log S(X) = \sum_{k=1}^{m} \frac{(-1)^{k+1}}{k} (S(X) - 1)^k, \tag{2}$$

yielding a compact vector $\zeta(X) \in \mathbb{R}^k$, where k depends on the truncation depth m and input dimension d.

Log-signatures offer properties that make them suitable replacements for standard positional and embedding mechanisms in transformers. First, positional information is encoded directly through ordered iterated integrals, eliminating the need for added positional embeddings. For example, the first-level term

$$\int dX = \sum_{t=1}^{p} (x_t - x_{t-1}) \tag{3}$$

represents cumulative increments over time, preserving order in the sequence.

Second, log-signatures provide feature engineering by incorporating higher-order interactions of the sequence through the polynomial terms $(S(X) - 1)^k$, which summarize correlations across multiple time steps in a structured manner.

Third, compared to raw transformer embeddings that scale linearly with dimension, log-signatures achieve dimensionality reduction since many redundant tensor components in $S(X)$ are eliminated by the logarithmic mapping, yielding a compact representation $\zeta(X) \in \mathbb{R}^k$ with $k \ll \dim(S(X))$.

Proposed Transformer

The Figure 1 gives an overview of the proposed transformer. We extend a decoder-only causal transformer with two preprocessing steps, log signature calculation and CNN computation. These steps although computationally expensive at first decrease the input size whilst capturing correlations between features enhancing accuracy and decreasing training time in the long run.

Consider a multivariate time series

$$X = \{x_1, x_2, \ldots, x_T\}, \quad x_t \in \mathbb{R}^d, \tag{4}$$

where T is the sequence length and d is the number of features at each time step. To expose local structure, the sequence is partitioned into non-overlapping patches of length p,

$$P_i = [x_{(i-1)p+1}, \ldots, x_{ip}] \in \mathbb{R}^{p \times d}, \quad i = 1, \ldots, N, \tag{5}$$

where P_i is the i-th patch, p is the patch length, and $N = T/p$ is the total number of patches.

For each patch, we compute its log-signature representation,

$$\zeta_i = \log S(P_i) \in \mathbb{R}^k, \tag{6}$$

where ζ_i encodes the algebraic features of the patch. In parallel, the same patch is processed by a convolutional encoder,

$$h_i = \mathrm{CNN}(P_i) \in \mathbb{R}^k, \tag{7}$$

where h_i extracts local temporal dependencies and both embeddings are designed to have the same dimension k. The two representations are then fused by addition and stacked into a sequence of tokens,

$$z_i = \zeta_i + h_i \in \mathbb{R}^k \tag{8}$$

where z_i is the token corresponding to the i-th patch. The tokens are then stacked to form Z, the full token matrix for the sequence shown below.

$$Z = [z_1, z_2, \ldots, z_N] \in \mathbb{R}^{N \times k} \tag{9}$$

The sequence Z is passed into a decoder-only causal transformer, which applies masked multi-head self-attention. Attention ensures each token incorporates information from its context, while masking enforces autoregressiveness by preventing future positions from being used. The masked scaled dot-product attention is defined as

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right) V, \tag{10}$$

Table 1: Summary of implemented datasets.

Dataset	Number of Parameters	Frequency	Number of Timestamps
Weather	21	10 mins	52,696
ETTm1	7	15 mins	69,680
ETTm2	7	15 mins	69,680
ETTh1	7	1 hour	17,420
ETTh2	7	1 hour	17,420

where $Q, K, V \in \mathbb{R}^{N\times d_k}$ are the query, key, and value projections of the input Z, d_k is the head dimension, and $M \in \mathbb{R}^{N\times N}$ is the causal mask with $M_{ij} = 0$ if $j \leq i$ and $M_{ij} = -\infty$ if $j > i$. This ensures that the attention at position i depends only on positions up to i, preserving causality in the forecasting task.

The outputs from all H heads are concatenated, resulting in a dimension of $H \cdot d_k$, and then projected back to the model dimension d_{model}. This result is combined with the residual connection from the input and normalized, after which it is passed through a position-wise feed-forward network. The feed-forward network typically consists of two linear transformations separated by a non-linear activation such as ReLU or GELU. It first expands the representation to a higher-dimensional space of size d_{ff} and then projects it back to d_{model}, enabling the model to capture complex non-linear relationships in the time series. A second residual connection and normalization follow, yielding the updated hidden representation. By stacking L such decoder blocks, the model integrates information across different temporal scales, producing the final sequence representation H.

$$H = Z^{(L)} \in \mathbb{R}^{N\times d_{\text{model}}}, \tag{11}$$

The output of the decoder H is passed through a linear projection layer that maps it to the original feature dimension d, producing the forecasted sequence $\widehat{Y} \in \mathbb{R}^{N\times d}$. The last m rows of $\widehat{Y}$ correspond to the predictions for the next m time steps.

Datasets

The proposed architecture is implemented on 5 publicly available time series datasets widely considered as benchmark datasets, ETTh1, ETTh2, ETTm1, ETTm2 and Weather. A summary of the datasets is given in Table 1.

The Electric Transformer Temperature (ETT) dataset (Zhou et al. 2021) comprises measurements from two regions in China from 2016 to 2018. It includes Oil Temperature (OT) and six load-related features, HUFL High Useful Load, HULL High Useless Load, MUFL Middle Useful Load, MULL Middle Useless Load, LUFL Low Useful Load, and LULL Low Useless Load. The data are provided at two temporal resolutions, hourly ETTh and 15 minute intervals ETTm. The ETT datasets are split in a 6 to 2 to 2 ratio, where 60% of the data is used for training, 20% for validation and 20% for testing. The Weather dataset (Max Planck Institute for Biogeochemistry 2025) is a public resource curated by the Max Planck Institute for Biogeochemistry, Germany. It contains 21 meteorological variables recorded every 10 minutes since 2008. Following prior long sequence time series forecasting work, we use the 2020 subset. The dataset is split in a 7 to 1 to 2 ratio, where 70% is used for training, 10% for validation and 20% for testing.

Baselines

We compare our transformer against five state of the art and efficient transformers, namely the Vanilla Transformer, Autoformer, Informer, Crossformer and Reformer. These models have been chosen for their efficiency in memory usage, training time and accuracy. We briefly summarize each baseline below.

- Vanilla Transformer (Vaswani et al. 2017) which uses the canonical multi head self attention and position wise feed forward network
- Autoformer (Wu et al. 2021) which introduces auto correlation based attention and a series trend decomposition module
- Informer (Zhou et al. 2020) which adopts ProbSparse self attention with a distilling operation that keeps the most informative keys
- Crossformer (Zhang and Yan 2023) which performs cross dimensional block attention to model inter series and intra series dependencies
- Reformer (Kitaev, Kaiser, and Levskaya 2020) which replaces dense attention with locality sensitive hashing and uses reversible layers for low memory training

Experiment Setup

All experiments are conducted on the Intel Unnati Server and implemented in PyTorch. The lookback window L is 336 and we evaluate horizons H in braces 96 192 336 720. Batch size is 32 and training uses automatic mixed precision. The optimizer is AdamW with learning rate 1e-3 and weight decay 1e-2. Training runs for at most 100 epochs with early stopping on validation mean squared error with patience 10 and we report the best validation checkpoint. Each configuration is trained with three seeds 42 43 44 and the best results and reported. The proposed transformer, SigPatchformer uses non overlapping patches with p in braces 16 24 32 48. For each patch we compute a log signature of order m = 2 producing a k dimensional vector. In parallel a 1D CNN encodes the patch using Conv1d with kernel size 3 and stride 1 followed by GELU then a pointwise Conv1d and temporal pooling to obtain a vector in R to the power k. We fuse the two vectors by element wise addition to form a token then linearly map tokens to the model width. The decoder is causal with L equal to 4 layers H equal to 8 heads model width 512 feed forward width 2048 and dropout 0.1. No positional embeddings are added since the log signature encodes order. Baselines are configured with their standard components and parameter counts are matched within ten% of ours to ensure fair capacity. Efficiency measurements are conducted on the ETTh1 dataset and include average time per epoch computed as average over the last five epochs after excluding the first epoch and VRAM consumption of models is measured in MegaBytes MB. We report both accuracy metrics Mean Squared Error MSE and Mean Absolute Error MAE.

Table 2: Forecasting Performance (MSE and MAE) Across Datasets and Horizons

Dataset	Horizon	Proposed		Transformer		Autoformer		Informer		Crossformer		Reformer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.386**	**0.411**	1.123	0.848	0.649	0.542	1.188	0.860	0.472	0.493	0.837	0.728
	192	**0.527**	**0.496**	1.172	0.858	0.582	0.544	1.157	0.822	0.532	0.519	0.923	0.766
	336	**0.570**	**0.534**	1.270	0.912	0.550	0.519	1.218	0.856	0.698	0.602	1.097	0.835
	720	0.670	0.566	1.375	0.966	**0.557**	**0.537**	1.191	0.857	0.847	0.713	1.257	0.889
ETTh2	96	**0.573**	**0.485**	2.153	1.268	1.373	1.421	1.882	1.099	0.884	0.690	2.626	1.317
	192	**0.956**	**0.897**	2.745	1.444	2.298	1.452	2.537	1.323	2.835	1.381	3.120	2.979
	336	**1.897**	**1.103**	2.620	1.408	2.385	1.346	2.804	1.438	2.339	1.241	4.028	1.688
	720	2.746	1.904	2.661	1.345	**2.449**	**1.478**	3.054	1.492	4.387	2.810	5.381	2.015
ETTm1	96	0.465	0.492	0.831	0.680	0.664	0.539	0.839	0.684	**0.444**	**0.467**	0.538	0.528
	192	**0.481**	**0.506**	0.982	0.773	0.682	**0.547**	0.903	0.734	0.522	0.524	0.658	0.592
	336	**0.599**	**0.660**	1.079	0.846	0.686	0.605	1.220	0.877	0.649	0.592	0.898	0.721
	720	0.856	0.745	0.975	0.750	0.759	0.597	1.031	0.762	**0.818**	**0.672**	1.102	0.841
ETTm2	96	0.298	0.312	1.056	0.844	**0.269**	**0.343**	0.515	0.568	0.350	0.421	0.658	0.619
	192	0.354	0.376	1.309	0.908	**0.302**	**0.364**	0.764	0.697	0.643	0.596	1.078	0.827
	336	0.546	0.528	1.661	1.086	**0.332**	**0.380**	1.117	0.858	2.051	1.774	1.549	0.972
	720	0.592	0.635	2.265	1.243	**0.414**	**0.428**	1.968	1.172	3.788	2.439	2.631	1.242
Weather	96	**0.203**	**0.176**	0.354	0.405	0.289	0.357	0.316	0.360	0.356	0.407	0.319	0.361
	192	**0.267**	**0.251**	0.419	0.434	0.312	0.364	0.467	0.471	0.314	0.370	0.325	0.371
	336	0.337	0.304	0.583	0.543	0.349	0.387	0.788	0.644	**0.324**	**0.294**	0.339	0.377
	720	0.381	0.366	0.916	0.705	0.393	0.412	1.301	0.860	0.755	0.885	**0.367**	**0.392**

Results

Forecasting Performance

Table 2 presents the MSE and MAE results for all models across the five datasets at different forecasting horizons. The Proposed model consistently achieves the lowest MSE and MAE across most datasets and horizons, demonstrating superior forecasting accuracy. For instance, on the ETTh1 dataset at horizon 96, the Proposed model achieves an MSE of 0.386 and an MAE of 0.411, significantly outperforming the Vanilla Transformer (MSE: 1.123, MAE: 0.848) and Informer (MSE: 1.188, MAE: 0.860). The model shows particularly strong performance on the Weather dataset, with an MSE of 0.203 and an MAE of 0.176 at horizon 96, compared to the next best model, Autoformer (MSE: 0.269, MAE: 0.343). Even at longer horizons, such as 720 time steps, the Proposed model maintains competitive performance, with an MSE of 0.381 and an MAE of 0.366 on the Weather dataset, outperforming all baselines.

We can observe from the given results that replacing positional encoding with the path signatures did not lead to a drop in performance. Additionally, its ability to capture relations between variables despite decreasing the input size did not hinder performance. When compared with the vanilla transformer, the Sig-Patchformer achieved significant increase in accuracy, with exception of one scenario on the ETTh2 dataset with horizon 720. This increase in performance can be attributed to the ability of log signatures in enhancing accuracy.

Across baselines the Vanilla Transformer generally trails the field with a single notable edge on ETTh2 at horizon 720 where it has the lowest MAE among baselines. Autoformer is the most consistent winner dominating ETTm2 across all horizons and leading ETTh1 at 336 and 720 as well as Weather at 96 and 192. Crossformer excels on short to mid horizons and minute level data winning ETTh1 at 96 and 192 most of ETTm1 except 720 and Weather at 336. Reformer rarely leads but is best on Weather at 720 for both MSE and MAE. Informer records no overall wins and typically lags Autoformer and Crossformer though it attains the best MAE on ETTh2 at 192

Table 3: Average training time per epoch (seconds) on ETTh1 dataset.

Model	**Training Time (s)**
Proposed	**38**
Transformer	153
Autoformer	107
Crossformer	115
Informer	49
Reformer	63

Training Efficiency

Table 3 reports average training time per epoch on ETTh1 and shows Sig-Patchformer is fastest at 38 seconds per epoch and other models follow with Transformer at 153 seconds, Autoformer at 107, Crossformer at 115, Informer at 49 and Reformer at 63. This corresponds to about 75% less time than Transformer, 22% less than Informer and 40% less than Reformer. The speed gains arise from patching which reduces the number of tokens and from the log signature computation that performs dimensionality reduction. Despite Informer and Reformer using sub quadratic attention the shorter token sequence makes Sig Patchformer faster in

Table 4: Memory consumption (MB) on ETTh1 dataset.

Model	Memory Consumption (MB)
Proposed	**924**
Transformer	3271
Autoformer	2615
Crossformer	2429
Informer	1598
Reformer	<u>1248</u>

this setup. Faster epochs enable larger batch sizes or longer look back windows on the same GPU which is valuable for big data workloads.

Memory Consumption

Table 4 presents the memory consumption in megabytes for each model on the ETTh1 dataset. The Proposed model is the most memory efficient, consuming 924 MB, compared to the Vanilla Transformer's 3271 MB. Reformer and Informer are relatively memory efficient, with consumptions of 1248 MB and 1598 MB respectively, while Autoformer and Crossformer consume 2615 MB and 2429 MB respectively. This corresponds to about 65% less memory than Autoformer and about 62% less than Crossformer. In sum, patching with log signature tokens and a lean causal decoder delivers faster training and much lower memory than the baselines, making the approach practical for large scale forecasting.

Conclusion

In this paper, we presented Sig-Patchformer, an efficient transformer-based model for time series forecasting that leverages path signatures, logarithmic transformations, and patching to overcome the limitations of traditional transformers, such as high computational overhead and memory demands. By combining log-signatures for compact feature extraction and positional encoding with convolutional processing for local dependencies, our approach enables accurate long-term predictions with reduced sequence lengths and model complexity. Empirical results on benchmark datasets, including ETTh1, ETTh2, ETTm1, ETTm2, and Weather, show that Sig-Patchformer consistently achieves superior forecasting performance in terms of MSE and MAE compared to leading baselines, while demonstrating significant efficiency gains—training in just 38 seconds per epoch and consuming only 924 MB of memory on ETTh1, outperforming the next best models by substantial margins. These findings highlight the potential of integrating rough path theory with modern neural architectures to create practical, scalable solutions for real-world time series applications. Future work could explore extensions to handling irregularly sampled data, incorporating exogenous variables more dynamically, or deploying the model in edge computing environments for low-latency forecasting.

References

Chen, P.; Zhang, Y.; Cheng, Y.; Shu, Y.; Wang, Y.; Wen, Q.; Yang, B.; and Guo, C. 2024. Pathformer: Multi-scale Transformers with Adaptive Pathways for Time Series Forecasting. *arXiv preprint arXiv:2402.05956*.

Chevyrev, I.; and Kormilitzin, A. 2016. A Primer on the Signature Method in Machine Learning. *arXiv preprint arXiv:1603.03788*.

Das, A.; Kong, W.; Sen, R.; and Zhou, Y. 2023. A Decoder-only Foundation Model for Time-Series Forecasting. *arXiv preprint arXiv:2310.10688*.

Deng, Y.; Wang, Z.; and Zhang, Y. 2024. Improving Multi-agent Reinforcement Learning with Stable Prefix Policy. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI-24)*, 49–57.

Ding, R.; Chen, Y.; Lan, Y.-T.; and Zhang, W. 2024. DR-Former: Multi-Scale Transformer Utilizing Diverse Receptive Fields for Long Time-Series Forecasting. 446–456.

Hegazy, K.; Mahoney, M. W.; and Erichson, N. B. 2025. Powerformer: A Transformer with Weighted Causal Attention for Time-series Forecasting. *arXiv preprint arXiv:2502.06151*.

Kakde, S.; Mitra, R.; Mandal, J.; and Tiwari, M. 2024. Knowledge-enhanced Transformer for Multivariate Long Sequence Time-series Forecasting.

Ke, Y.; Liang, Y.; Shi, Z.; Song, Z.; and Yang, C. 2025. Curse of Attention: A Kernel-Based Perspective for Why Transformers Fail to Generalize on Time Series Forecasting and Beyond. In Chen, B.; Liu, S.; Pilanci, M.; Su, W.; Sulam, J.; Wang, Y.; and Zhu, Z., eds., *Conference on Parsimony and Learning*, volume 280 of *Proceedings of Machine Learning Research*, 675–738. PMLR.

Kitaev, N.; Kaiser, L.; and Levskaya, A. 2020. Reformer: The Efficient Transformer. *arXiv preprint arXiv:2001.04451*.

Max Planck Institute for Biogeochemistry. 2025. Weather dataset: Max Planck Institute for Biogeochemistry, Germany. [Online]. Available: https://www.bgc-jena.mpg.de/wetter/.

Moreno-Pino, F.; Arroyo, ; Waldon, H.; Dong, X.; and Cartea, 2024. Rough Transformers: Lightweight Continuous-Time Sequence Modelling with Path Signatures.

Morrill, J.; Fermanian, A.; Kidger, P.; and Lyons, T. 2020. A Generalised Signature Method for Multivariate Time Series Feature Extraction. *arXiv preprint arXiv:2006.00873*.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2022. A Time Series is Worth 64 Words: Long-Term Forecasting with Transformers. *arXiv preprint arXiv:2211.14730*.

Piao, X.; Chen, Z.; Murayama, T.; Matsubara, Y.; and Sakurai, Y. 2024. Fredformer: Frequency Debiased Transformer for Time Series Forecasting.

Tan, M.; Merrill, M.; Gupta, V.; Althoff, T.; and Hartvigsen, T. 2024. Are Language Models Actually Useful for Time Series Forecasting?

Tang, P.; and Zhang, W. 2024. Unlocking the Power of Patch: Patch-Based MLP for Long-Term Time Series Forecasting. *arXiv preprint arXiv:2405.13575*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is All You Need. In *Advances in Neural Information Processing Systems*.

Wang, Y.; Wu, H.; Dong, J.; Qin, G.; Zhang, H.; Liu, Y.; Qiu, Y.; Wang, J.; and Long, M. 2024. TimeXer: Empowering Transformers for Time Series Forecasting with Exogenous Variables. *arXiv preprint arXiv:2402.19072*.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. *arXiv preprint arXiv:2106.13008*.

Yang, S.; Shen, Y.; Wen, K.; Tan, S.; Mishra, M.; Ren, L.; Panda, R.; and Kim, Y. 2025. PaTH Attention: Position Encoding via Accumulating Householder Transformations. *ArXiv*, abs/2505.16381.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2022. Are Transformers Effective for Time Series Forecasting? *arXiv preprint arXiv:2205.13504*.

Zhang, W.; Wang, H.; and Zhang, F. 2024. Skip-Timeformer: Skip-Time Interaction Transformer for Long Sequence Time-Series Forecasting. 5499–5507.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2020. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *arXiv preprint arXiv:2012.07436*.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 11106–11115.