TS-FourierLLM: Frozen Frequency-Domain Large Language Blocks for Enhancing Time-Series Modeling

Pengfei Wang, Huanran Zheng, Wenjing Yue, and Xiaoling Wang ✉

School of Computer Science and Technology,
East China Normal University, Shanghai, China
{pfwang,hrzheng,wjyue}@stu.ecnu.edu.cn, xlwang@cs.ecnu.edu.cn

Abstract. Designing effective time-series models is challenging due to factors like complex temporal dependencies, low information density, and scalability constraints. To address these challenges, we propose TS-FourierLLM, a novel approach that integrates a frozen Large Language Models (LLMs) block as a plug-and-play frequency-domain enhancer for time-series modeling. By transferring high-level pre-trained LLM knowledge into the frequency domain, our method bridges the modality gap while avoiding fine-tuning, preserving computational efficiency. The frozen LLM block captures inter-frequency dependencies and enhances global feature representations, complementing time-series encoders. TS-FourierLLM achieves up to a 3% performance improvement over state-of-the-art methods on the benchmark. These results demonstrate the effectiveness of utilizing frozen LLMs as modular and task-agnostic components for advancing time-series modeling.

Keywords: Large Language Models (LLMs) · Frequency-Domain Learning · Transfer learning · Time-Series Modeling

1 Introduction

Multivariate time series classification is a fundamental task with wide-ranging applications in healthcare [21], finance [7], and environmental monitoring [16]. This task involves identifying patterns within sequential data while capturing intricate temporal dependencies and inter-variable relationships. Recent advancements have showcased the transformative potential of multivariate time series analysis. For instance, Neuralink has successfully decoded and classified complex brain signals through electrode implantation[1], enabling patients to control external devices. Similarly, Google's climate model, NeuralGCM, completed a 22-day atmospheric simulation in just 30 seconds[2], highlighting the ability of time series models to analyze large-scale sequential data. These breakthroughs exemplify the practical impact of this field, but they also underline its inherent

[1] https://www.nature.com/articles/d41586-024-02368-8
[2] https://www.nature.com/articles/s41586-024-07744-y

challenges, including high-dimensional modeling, temporal dependencies, and scalability. Despite extensive research efforts, current approaches often rely on task-specific architectures or domain expertise [8, 9, 15, 26], which limits their adaptability to new domains. Furthermore, the scarcity of labeled data presents a significant bottleneck, making it difficult to train robust and accurate classification models. Addressing these challenges calls for developing generalizable, data-efficient frameworks capable of capturing the complex dynamics of multivariate time series.

With the advent of Large Language Models (LLMs), two primary strategies have emerged for their integration into time-series tasks: Single-tower methods and Dual-tower methods, as illustrated in Figure 1. Single-tower methods tokenize time-series data into discrete formats for LLM processing, leveraging their sequence modeling capabilities [3, 27]. However, the inherent *modality mismatch* between time-series data and natural language often leads to performance degradation during fine-tuning [32]. In contrast, Dual-tower methods encode time-series and textual modalities separately, facilitating mutual knowledge transfer in multimodal settings [12, 17, 19]. Despite their effectiveness, these approaches rely on paired textual data or complex alignment strategies, which are scarce in time-series applications where data is typically collected without accompanying textual descriptions. Additionally, a recent article [23] questioning the effectiveness of LLMs in time-series analysis has drawn significant attention. However, its validation is limited to forecasting tasks. Analyzing the source code[3] reveals several issues in the evaluation settings, including the absence of a uniform lookback period and the lack of standardized data splitting. In particular, the incorrect application of the "Drop Last" operation during testing raises concerns about the reliability of the reported results [20]. These findings suggest that the current evaluation may not fully capture the potential of LLMs for time-series modeling, contrasting with the broader consensus that LLMs hold substantial promise in this field [13].

To address these limitations, we propose a novel approach **TS-FourierLLM**, which avoids the need for fine-tuning and paired data by utilizing a frozen LLM block as a plug-and-play enhancer for frequency-domain feature learning in time-series modeling. Unlike conventional LLM applications, our method is independent of language-specific components such as prompts, textual inputs, or outputs, representing a significant departure from conventional usage. Additionally, our approach is versatile, accommodating both pre-trained and non-pre-trained scenarios, offering flexibility while reducing dependency on pre-trained models. Furthermore, we simplify the LLM integration by treating frozen transformer blocks as modular components that bolster existing pipelines without altering their fundamental structure. This innovative design not only challenges the traditional application of LLMs but also redefines their role in time-series analysis.

The key contributions of this work are as follows:

– **LLM-based framework for time-series modeling.** We introduce a residual architecture that integrates a frozen transformer block from pre-trained

[3] https://github.com/BennyTMT/LLMsForTimeSeries

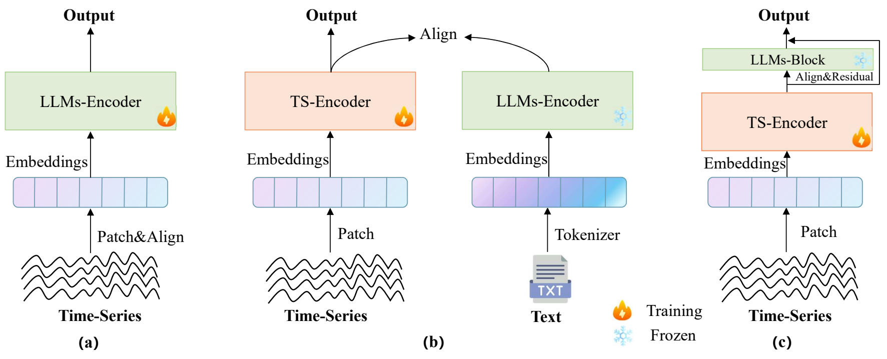

Fig. 1. Comparison of LLMs for time-series analysis frameworks: (a) Single-tower methods, (b) Dual-tower methods, and (c) Our TS-FourierLLM.

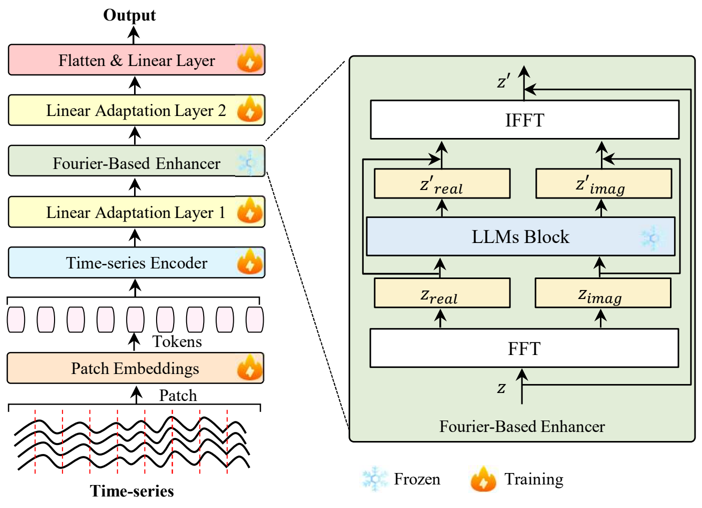

Fig. 2. Overview of the proposed TS-FourierLLM framework.

LLMs as a plug-and-play frequency-domain feature enhancer, effectively handling modality mismatches, temporal dependencies, and inter-frequency interactions.

– **Efficient block-wise LLM transfer.** We show that selectively transferring middle-layer parameters optimally balances low-level pattern extraction and high-level semantics, reducing complexity, mitigating overfitting, and improving frequency-domain refinement.
– **State-of-the-art performance.** Our approach achieves up to 3% higher accuracy than SOTA methods while significantly reducing computational overhead, demonstrating both effectiveness and efficiency.

2 Method

In this section, we introduce the TS-FourierLLM (Fig. 2), a novel approach for time-series classification that uses pretrained LLMs in the frequency domain. The method includes a Time-series Encoder (Section. 2.1) for tokenizing input data, a Fourier-Based Enhancement Module (Section 2.2) for frequency enhancer, and a Task-Specific Classification Head (Section 2.3) for accurate predictions. The overall optimization process is discussed in (Section 2.4) .

2.1 Time-series Encoder

To process time-series data for transformer-based architectures, the input sequence is first tokenized using a patch-based method. Specifically, we follow the PatchTST approach [18], which divides consecutive time steps into non-overlapping patches and converts each patch into a token. This tokenization process captures local temporal features and prepares the data for self-attention mechanisms. Given a multivariate time series $\mathbf{x} \in \mathbb{R}^{T\times D}$, it is segmented into $P = \left\lfloor \frac{T}{L} \right\rfloor + 2$ patches. Each patch $\mathbf{x}_P \in \mathbb{R}^{L\times D}$ represents L consecutive time steps, and the additional terms account for edge handling. These patches are then linearly projected into a compact token representation:

$$\mathbf{F}_M(\mathbf{x}_P) \rightarrow \mathbf{t}_P, \tag{1}$$

where $\mathbf{t}_P \in \mathbb{R}^D$ denotes the embedding vector, D is the dimension of the embedding space. The resulting token sequence $\{\mathbf{t}_P\}_{P=1}^{P} \in \mathbb{R}^{P\times D}$ efficiently captures local temporal structures and edge contexts. These tokens are then fed into a standard transformer-based encoder [24] to model temporal dependencies and implicit inter-variable relationships:

$$\mathbf{F}_{TS}(\mathbf{t}_P) \rightarrow z, \tag{2}$$

where $\mathbf{z}$ represents the refined latent representation of the time series.

2.2 Fourier-based Enhancement Module

The Fourier-based enhancement module $\mathbf{F}_{Fourier\_LM}$ is designed to integrate pretrained LLM blocks into a time-series pipeline, addressing the challenge of capturing both temporal and frequency-domain patterns effectively. The module is embedded between the encoder $\mathbf{F}_E$ and the task-specific decoder $\mathbf{F}_D$, acting as a bridge to enhance feature representations. To address the dimensional mismatch between the encoder output and the LLM input, two lightweight linear mapping layers, $\mathbf{F}_M^b$ and $\mathbf{F}_M^a$, are introduced to transform feature dimensions before and after the module:

$$\mathbf{F}_M^a \cdot \mathbf{F}_{Fourier\_LM} \cdot \mathbf{F}_M^b(z) \rightarrow z'. \tag{3}$$

Here, z represents the feature embeddings from the encoder $\mathbf{F}_E(x)$, and z' is the enhanced embedding passed to the decoder $\mathbf{F}_D$.

The module $\mathbf{F}_{Fourier\_LM}$ integrates Fourier transform and a pretrained LLM block $\mathbf{F}_{LM}$ to enhance frequency-domain features. The process consists of three main steps:

Step 1: Fourier Transform. The input embeddings $z \in \mathbb{R}^{\text{batch} \times n \times \text{dim}}$ are projected into the frequency domain using the Fast Fourier Transform (FFT):

$$z_{\text{freq}} = \text{FFT}(z), \quad z_{\text{real}}, z_{\text{imag}} = \text{Decompose}(z_{\text{freq}}), \tag{4}$$

where z_{real} and z_{imag} are the real and imaginary components, respectively.

Step 2: LLM Enhancement. The concatenated real and imaginary components are passed through the pretrained LLM block $\mathbf{F}_{LM}$, which enhances inter-frequency dependencies:

$$z'_{\text{real}}, z'_{\text{imag}} = \mathbf{F}_{LM}([z_{\text{real}}; z_{\text{imag}}]) + [z_{\text{real}}; z_{\text{imag}}]. \tag{5}$$

The residual connections [10] ensure that original frequency-domain features are preserved while incorporating the enhancements

Step 3: Reconstruction. The enhanced components are recombined into a complex representation:

$$z'_{\text{freq}} = z'_{\text{real}} + i \cdot z'_{\text{imag}}, \tag{6}$$

and an inverse Fourier Transform is applied:

$$z' = \text{iFFT}(z'_{\text{freq}}) + z. \tag{7}$$

Design Motivation. Frequency decomposition is particularly advantageous for time-series analysis, as it facilitates the identification of periodic and trend patterns, a technique that has demonstrated effectiveness in prior work [31]. The proposed module builds on the following key principles:

- **Frequency-domain representations**: The Fourier Transform decomposes time-series signals into frequency components, where low-frequency features capture global trends and high-frequency features represent local variations.
- **LLM-based enhancement**: Pretrained LLMs, equipped with attention mechanisms, effectively model complex dependencies, enabling joint processing of real and imaginary components in the frequency domain.
- **Structural similarity to language modeling**: The decomposition of signals into frequency bands parallels the tokenization process in language models, where discrete units (tokens) are processed and contextualized.

2.3 Task-Specific Classification Head

The classificaiton head is added on top of the features extracted from the LLMs block. It's specifically designed for time-series classification task, which includes a fully connected layer aggregating all tokens information, followed by activation functions such as softmax for classification. Different from ViT [6], which uses only the [cls] token, all token embeddings are used to capture a global and comprehensive representation.

$$\mathbf{F}_{Task}(z') \rightarrow \hat{y}. \tag{8}$$

2.4 Overall Optimization

During training, the label smoothing technique [22] is used to enhance the robustness of the model, particularly in scenarios with a limited amount of labeled time-series data. By freezing the weights of the pretrained block from the LLM, the model focuses its learning process on the tokenizer, time-series encoder, and task-specific classification head, thus preventing the loss of valuable knowledge and mitigating the risk of overfitting. Mathematically, the cross-entropy loss can be expressed as:

$$L_{CE} = -\sum y \log(\hat{y}), \tag{9}$$

where L_{CE} represents the cross-entropy loss, y denotes the smoothed true label, and $\hat{y}$ signifies the predicted probability for the label.

3 Experiments

In this section, we conduct experiments on ten multivariate time-series datasets, including an overall performance analysis, an ablation analysis to explore the effectiveness of the proposed components, and a deeper investigation into the behavior and capabilities of the TS-FourierLLM.

3.1 Experimental Settings

Datasets. To evaluate the model's capability in high-level representation learning, sequence-level classification tasks are employed. The experimental setup follows state-of-the-art methods, including TimesNet [25] and GPT4TS [32], ensuring consistency in data processing and dataset splits. Multivariate UEA datasets [1] are selected, covering diverse domains such as gesture recognition, action classification, audio recognition, and medical diagnosis. Detailed dataset descriptions are available at website[4].

Experimental Setup. A standard Transformer encoder with 6 layers and 8 heads serves as the time-series encoder, incorporating fixed positional encoding. The model employs 256-dimensional latent vectors across all layers, with a 512-dimensional MLP hidden layer. A patch size of 8 is used for windowing, and the LLM configuration follows GPT4TS [32]. To maintain consistency, we adopt accuracy as the evaluation metric.

3.2 Performance and Analysis

Overall Performance. Table 1 compares classification accuracy across 10 benchmark datasets. Classical methods like ROCKET achieved 72.5%, while MLP-based models performed moderately (67.5%–70.4%). RNNs and CNNs showed competitive results, with LSTNet (71.8%) and TimesNet (73.6%) leading their categories. Transformer models ranged from 70.7% to 73.0%, with PatchTST

[4] https://www.timeseriesclassification.com/dataset.php

Table 1. Comparison of classification accuracy (%) for different models on 10 benchmark datasets. **Red** indicates the best performance, and Blue indicates the second.

Model/Dataset		EC	FD	HW	HB	JV	PF	SCP1	SCP2	SAD	UW	**Avg.** ↑
Classical Methods	DTW [2]	32.3	52.9	28.6	71.7	94.9	71.1	77.7	53.9	96.3	90.3	66.97
	XGBoost [4]	43.7	63.3	15.8	73.2	86.5	98.3	84.6	48.9	69.6	75.9	65.98
	ROCKET [5]	45.2	64.7	58.8	75.6	96.2	75.1	90.8	53.3	71.2	94.4	72.53
MLP	LightTS [29]	29.7	67.5	26.1	75.1	96.2	88.4	89.8	51.1	100	80.3	70.42
	DLinear [28]	32.6	68	27	75.1	96.2	75.1	87.3	50.5	81.4	82.1	67.53
RNN	LSTM [11]	32.3	57.7	15.2	72.2	79.7	39.9	68.9	46.6	31.9	41.2	48.56
	LSTNet [15]	39.9	65.7	25.8	77.1	98.1	86.7	84	52.8	100	87.8	71.79
	LSSL [9]	31.1	66.7	24.6	72.7	98.4	86.1	90.8	52.2	100	85.9	70.85
CNN	TCN [8]	28.9	52.8	53.3	75.6	98.9	68.8	84.6	55.6	95.6	88.4	70.25
	TimesNet [25]	35.7	68.6	32.1	78	98.4	89.6	91.8	57.2	99	85.3	73.57
Transformers	Transformer [24]	32.7	67.3	32	76.1	98.7	82.1	92.2	53.9	98.4	85.6	71.9
	Reformer [14]	31.9	68.6	27.4	77.1	97.8	82.7	90.4	56.7	97	85.6	71.52
	Informer [30]	31.6	67	32.8	80.5	98.9	81.5	90.1	53.3	100	85.6	72.13
	FEDformer [31]	31.2	66	28	73.7	98.4	80.9	88.7	54.4	100	85.3	70.66
	PatchTST [18]	34.2	69.2	32.7	77.2	98.6	87.9	93.2	59.4	99.2	88.1	72.1
LLMs	GPT4TS [32]	34.2	69.2	32.7	77.2	98.6	87.9	93.2	59.4	99.2	88.1	73.97
	Time-LLM [18]	30.0	50.2	6.7	72.1	23.9	23.1	63.8	50.0	22.7	33.4	37.6
	S2IP [19]	28.6	66.5	25.8	72.6	93.8	80.41	85.3	57.2	96.6	82.1	68.9
	CALF [17]	25.8	68.1	26.8	75.6	95.1	78.2	89.4	53.8	98.1	86.5	69.78
	Ours	52.9	68.9	23.8	78	99.2	93.1	93.2	62.8	99.2	90.6	**75.3**

reaching 72.1%. LLM-based approaches excelled, with GPT4TS achieving 74.0% and our TS-FourierLLM attaining the highest accuracy of 75.3%, demonstrating its effectiveness in time-series classification. Notably, Time-LLM struggled with multivariate tasks due to its reliance on full LLM architectures, leading to overfitting.

Ablation Analysis. The ablation study evaluates key components of TS-FourierLLM in multivariate time-series classification. The baseline model with only the time-series encoder achieved 72.1% accuracy. Adding a Randomly Initialized LLM Transformer provided a slight boost to 72.3%, indicating limited impact without pretraining. The Pretrained Tiny Transformer Transfer significantly improved accuracy to 74.4%, highlighting the benefits of pretrained LLM knowledge. Introducing the FFT & iFFT module achieved 73.1%, capturing essential frequency-domain features. The full TS-FourierLLM framework, integrating all components, reached 75.3%, demonstrating their synergistic effect. These results underscore the importance of each component in enhancing classification performance.

Analyzing the Impact of LLM Enhancement in the Fourier Domain. As shown in Figure 3, integrating LLM-based enhancement in the Fourier domain improves the model's ability to capture frequency-domain features. A symmetric, bell-shaped weight distribution indicates effective frequency enhancement, leading to better performance. Skewed distributions, however, suggest overemphasis

Table 2. Ablation study.

Model Variant	**Accuracy (%)**
Only Time-series Encoder (Baseline)	72.1
+ LLMs Transformer (Random Initialization)	72.3
+ Pretrain Tiny Transformer Transfer	74.4
+ FFT & iFFT	73.1
TS-FourierLLM (Ours)	**75.3**

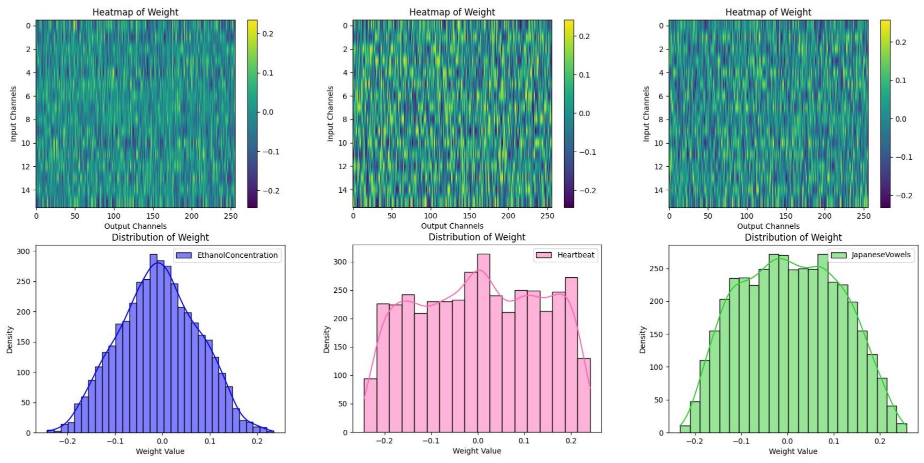

Fig. 3. Analyzing the Impact of LLM Enhancement in the Fourier Domain.

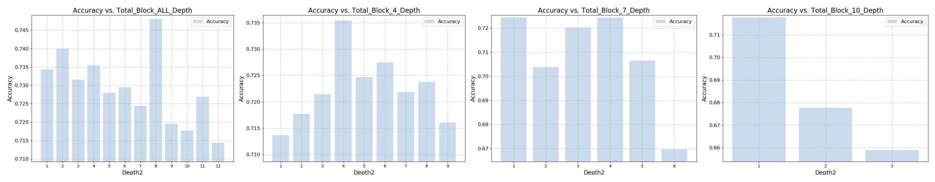

Fig. 4. Impact of LLM Transformer Block Variations on Time-Series Classification.

or neglect of certain frequency bands, reducing accuracy. These findings highlight the importance of LLM-based enhancement in refining frequency-domain features and boosting the predictive accuracy of the our framework.

Impact of LLM Transformer Block Variations on Time-Series Classification. The impact of LLM Transformer block variations on time-series classification is analyzed in Figure 4. Mid-level GPT-2 blocks strike the best balance between detail and semantics, suggesting an optimal depth for capturing both fine-grained and high-level features. Increasing the number of blocks slightly reduces performance, likely due to overfitting on limited datasets, highlighting the challenge of adapting general-purpose LLMs to time-series tasks without sufficient fine-tuning.

4 Conclusion

This paper introduces TS-FourierLLM, a framework for time-series modeling that integrates a pre-trained LLM's transformer block as a frequency-domain enhancer. It addresses challenges like modality mismatches and complex temporal dependencies, without requiring fine-tuning. Evaluations on benchmarks show it outperforms state-of-the-art methods with up to 3% accuracy improvement. Ablation studies reveal that transferring middle-layer parameters helps balance pattern extraction and semantic representation, enhancing model performance and reducing overfitting. This approach sets a new direction for applying LLMs in time-series analysis.

Acknowledgements. This work was supported by NSFC grant (No. 62136002 and 62477014), Ministry of Education Research Joint Fund Project (8091B042239), and Shanghai Trusted Industry Internet Software Collaborative Innovation Center.

References

1. Bagnall, A., Dau, H.A., Lines, J., et al.: The uea multivariate time series classification archive, 2018. arXiv preprint arXiv:1811.00075 (2018)
2. Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: KDD (1994)
3. Cao, D., Jia, F., Arik, S.O., et al.: Tempo: Prompt-based generative pre-trained transformer for time series forecasting. In: ICLR (2024)
4. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: SIGKDD. pp. 785–794 (2016)
5. Dempster, A., Petitjean, F., Webb, G.I.: Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. Data Mining and Knowledge Discovery **34**(5), 1454–1495 (2020)
6. Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2020)
7. Foumani, N.M., Tan, C.W., Webb, G.I., Salehi, M.: Improving position encoding of transformers for multivariate time series classification. Data Mining and Knowledge Discovery **38**(1), 22–48 (2024)
8. Franceschi, J.Y., Dieuleveut, A., Jaggi, M.: Unsupervised scalable representation learning for multivariate time series. In: NeurIPS (2019)
9. Gu, A., Goel, K., Ré, C.: Efficiently modeling long sequences with structured state spaces. In: ICLR (2022)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)

11. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
12. Jia, F., Wang, K., Zheng, Y., et al.: Gpt4mts: Prompt-based large language model for multimodal time-series forecasting. In: AAAI (2024)
13. Jiang, Y., Pan, Z., Zhang, X., et al.: Empowering time series analysis with large language models: A survey. In: IJCAI (2024)
14. Kitaev, N., Kaiser, Ł., Levskaya, A.: Reformer: The efficient transformer. In: ICLR (2020)
15. Lai, G., Chang, W.C., Yang, Y., et al.: Modeling long-and short-term temporal patterns with deep neural networks. In: ACM SIGIR (2018)
16. Lee, Z., Lindgren, T., Papapetrou, P.: Z-time: efficient and effective interpretable multivariate time series classification. Data mining and knowledge discovery **38**(1), 206–236 (2024)
17. Liu, P., Guo, H., Dai, T., et al.: Calf: Aligning llms for time series forecasting via cross-modal fine-tuning. AAAI (2025)
18. Nie, Y., Nguyen, N.H., Sinthong, P., et al.: A time series is worth 64 words: Long-term forecasting with transformers. In: ICLR (2023)
19. Pan, Z., Jiang, Y., Garg, S., et al.: s^2ip-llm: Semantic space informed prompt learning with llm for time series forecasting. ICML (2024)
20. Qiu, X., Hu, J., Zhou, L., et al.: Tfb: Towards comprehensive and fair benchmarking of time series forecasting methods. PVLDB (2024)
21. Schäfer, P., Leser, U.: Weasel 2.0: a random dilated dictionary transform for fast, accurate and memory constrained time series classification. Machine Learning **112**(12), 4763–4788 (2023)
22. Szegedy, C., Vanhoucke, V., Ioffe, S., et al.: Rethinking the inception architecture for computer vision. In: CVPR. pp. 2818–2826 (2016)
23. Tan, M., Merrill, M.A., et al.: Are language models actually useful for time series forecasting? In: NeurIPS (2024)
24. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: NeurIPS (2017)
25. Wu, H., Hu, T., Liu, Y., et al.: Timesnet: Temporal 2d-variation modeling for general time series analysis. ICLR (2023)
26. Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. NeurIPS (2021)
27. Xue, H., Salim, F.D.: Promptcast: A new prompt-based learning paradigm for time series forecasting. IEEE Transactions on Knowledge and Data Engineering (2023)
28. Zeng, A., Chen, M., Zhang, L., Xu, Q.: Are transformers effective for time series forecasting? In: AAAI (2023)
29. Zhang, T., Zhang, Y., Cao, W., et al.: Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. arXiv preprint arXiv:2207.01186 (2022)
30. Zhou, H., Zhang, S., Peng, J., et al.: Informer: Beyond efficient transformer for long sequence time-series forecasting. In: AAAI (2021)
31. Zhou, T., Ma, Z., Wen, Q., et al.: Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In: ICML (2022)
32. Zhou, T., Niu, P., Wang, X., et al.: One fits all: Power general time series analysis by pretrained lm. NeurIPS (2023)