Probing Large Language Models for Zero-shot Time Series Understanding

Anonymous ACL submission

Abstract

Recently, large language models (LLMs) have shown promising performance in time series forecasting, including two paradigms: (a.) re-customizing LLMs for supervised forecasting, and (b.) keeping LLMs unchanged for zero-shot forecasting. However, how do large language models understand time series? In this work, we explore the understanding capability of LLMs on time series while maintaining their structure and parameters unchanged in zero-shot forecasting scenarios. Specifically, starting from basic time series patterns, we investigate the forecasting ability of LLMs on basic function series, as well as the impact of diverse periods, amplitudes, and phases on the forecasting for sinusoidal series. Subsequently, to gain deeper insights, we design a series of probing methods to further analyze the understanding of LLMs on time series. Finally, guided by these findings, we propose Frequency Decomposition (Freq-Decomp), a lightweight preprocessing method that enhances LLMs' zero-shot forecasting performance. Experiments across real-world datasets show that LLMs excel at identifying periodic patterns, probing experiments provide insight into the perception of time series information by LLMs' different layers, and Freq-Decomp can yield consistent improvements over prior zero-shot baselines.

1 Introduction

Time series data, as a fundamental data type, is widely present in various fields (Wen et al., 2022, 2023) such as economic forecasting (Niu et al., 2023a; Lerner et al., 2004; Gu et al., 2020), healthcare (Niu et al., 2023b; Chae et al., 2023), and traffic management (Alghamdi et al., 2019; Fang et al., 2024). Analyzing and predicting time series contributes to uncovering real-world insights behind the data, aiding in the analysis of practical problems, and providing decision support. In this context, there have been explorations ranging from statistical analysis to machine learning and deep learning (Gamboa, 2017; Wen et al., 2021). Particularly, recent advancements in models based on Transformer architecture (Liu et al., 2023; Su et al., 2024; Wu et al., 2021; Zhou et al., 2022) and Linear structures (Zeng et al., 2023) have shown promising results. These models can effectively represent meaningful features of time series data for tasks such as prediction.

Recently, large language models (LLMs) (Touvron and et al., 2023; Brown and et al., 2020) have demonstrated remarkable performance in natural language processing (NLP), showcasing impressive zero-shot learning capabilities, in-context learning, and reasoning and planning abilities such as chain-of-thought reasoning (Dong et al., 2023; Schaeffer et al., 2023; Olsson and et al., 2022; Qiu et al., 2023). Subsequently, extensive research has been conducted in fields like computer vision (Dosovitskiy et al., 2021; Kirillov et al., 2023; Oquab and et al., 2023) and speech processing (Latif et al., 2023; Radford et al., 2022). Similarly, the field of time series forecasting has seen numerous studies leveraging LLMs. Based on our observations, we categorize these studies into two paradigms: (a) re-customizing LLMs for supervised forecasting (Zhou et al., 2023; Jin et al., 2023; Liu et al., 2024b,c), and (b) utilizing LLMs unchanged for zero-shot forecasting (Gruver et al., 2023; Mirchandani et al., 2023). For the first paradigm, the approach primarily involves re-customizing LLMs (or parts of their structure) and adding extra prediction heads to meet the requirements of downstream time series tasks. This process requires additional data to supervise the training of the customized language model and the prediction head to achieve the final forecasting performance. The second paradigm does not involve re-customizing the LLMs. Instead, it maintains LLMs' original structure and parameters unchanged and preprocesses the time series to fit the input format of LLMs.

However, LLMs are primarily trained on extensive text data, and how they understand time series and their characteristics in time series forecasting is still not well understood. Therefore, in this paper, we systematically explore the capability of frozen LLMs to understand time series in a zero-shot and training-free settings. Starting with basic function series, we investigate the ability of LLMs to predict series with definite patterns. We begin by testing their ability to model synthetic series composed of basic functions, with a focus on periodic components such as sine and cosine waves. We examine how changes in amplitude, frequency, and phase affect predictive performance. To gain deeper insight, we propose three probing strategies-token perturbation, linear probing, and vocabulary mapping-to analyze how LLMs internally represent time series signals. Our analysis reveals that LLMs are particularly sensitive to periodicity, rely on special tokens (e.g., <s>) for anchoring sequence context, and gradually shift from local to global representations across layers. Based on these observations, we introduce Frequency Decomposition (Freq-Decomp)—a lightweight preprocessing method that decomposes time series into frequency bands via Fourier transforms, allowing LLMs to better model individual components. Empirical results across ten real-world datasets and multiple LLM backbones (e.g., LLaMA2/3, Vicuna, Qwen) show that Freq-Decomp consistently enhances zero-shot forecasting performance, outperforming state-of-the-art baselines without modifying model parameters. Our contributions are threefold:

- **Periodic Series.** We provide empirical evidence that LLMs are biased toward periodic structures. And experiments with synthetic series show that the recognition ability of LLMs decreases as the number of superimposed frequency components increases, and this decline is more pronounced when the phases also vary.
- **Probing Methods.** We present a systematic probing framework to analyze how LLMs understand time series data. Token Perturbation Probing reveals the mutual impact relationships between different series tokens across different layers. Linear probing reveals the exploration of LLMs on input sequences, as the layers progress, the search range gradually narrows or stabilizes. Vocabulary Mapping Probing elucidate the distribution of time series feature exploration and sequence prediction from the perspective of special tokens and numeric tokens.
- **Frequency Decomposition.** We propose Freq-Decomp, a simple yet effective method to improve zero-shot time series forecasting with frozen LLMs under training-free settings.

2 Related Works

2.1 LLMs for Time Series Analysis

Researchers have begun to focus on utilizing LLMs to address time series problems. This research was facilitated especially with the introduction of PatchTST (Nie et al., 2023). Here, we categorize this research into two paradigms: (a) re-customizing LLMs for supervised forecasting (Zhou et al., 2023; Jin et al., 2023; Liu et al., 2024b,c) and (b) keeping LLMs unchanged for zero-shot forecasting. For the first paradigm, inspired by PatchTST, researchers tokenize the time series by applying patch operations and then re-customize the pre-trained language models (such as GPT-2 (Radford et al., 2019)) by adding a linear prediction head to serve as the time series predictor. Such research typically does not use the entire language model structure but rather selects certain layers to be repurposed as the main structure of the time series predictor. The second paradigm maintains the integrity of the language models, transforming the time series through preprocessing into a format that the language models can accept.

2.2 Pattern Completion with LLMs

Other related work involves using LLMs for pattern recognition and completion. This type of research is not limited to time series data but focuses on how LLMs process data with regular patterns (Tan and Motani, 2023). For example, (Mirchandani et al., 2023) explored the capabilities of LLMs in representing and extrapolating abstract non-linguistic patterns, such as ARC patterns (Chollet, 2019), and discussed their potential applications in controlling robotics. This work deals with data exhibiting definite patterns and employs few-shot learning to make predictions. (Liu et al., 2024a) explored the capability of LLMs to complete dynamic systems governed by principles of physical interest. (Guo et al., 2023) explored the ability of LLMs to learn representations of dynamic system series while implementing in-context learning mechanisms. Such research aims to explore the ability of LLMs to

complete or extrapolate patterns or dynamic systems with relatively definite principles. However, there has been little exploration of LLMs' capabilities in handling patterns or series with uncertain principles.

3 Preliminaries

3.1 Overview

In this section, we first describe the overall setup. Given large language models (LLMs) f_θ (θ represents the parameters of the LLMs), our work aims to explore how they understand and predict time series. Specifically, for a given time series $T = \langle t_1, t_2, \cdots, t_n \rangle$, we tokenize and rescale it following the LLMTime (Gruver et al., 2023) methodology. Then, we can obtain the numerical tokens of the time series $\mathbf{u} = \langle u_1, u_2, \cdots, u_n \rangle$, and input them into the LLMs. Here, n denotes the length of the time series. The LLMs primarily used in this work is LLaMA2 (7B) (Touvron and et al., 2023). Implementation Details are in Appendix A.4.

3.2 Settings

Datasets We primarily use Darts (Herzen et al., 2022) and ETT (Zhou et al., 2021) datasets to test the pattern completion ability of LLMs, which are available to the public.

- **Darts** is a collection of 8 univariate real-world time series datasets (e.g., airlines, etc.)
- **ETT** datasets originate from the electricity industry and record load and oil temperature variation data. Here, we primarily utilize the ETTh1 and ETTm1 datasets.

Metrics We utilize mean absolute error (MAE) and mean square error (MSE) to serve as metrics in alignment with the majority of existing works (Jin et al., 2023; Zhou et al., 2023; Cao et al., 2024).

4 Time Series Pattern Analysis

4.1 Basic Function Series

Method Firstly, to explore the ability of LLMs to understand time series, we start with basic function series (e.g., linear, polynomial, cosine function, etc.). Specifically, we select a series of basic functions $\mathcal{G} = \{g_1(x), g_2(x), \cdots, g_m(x)\}$, m denotes the number of basic functions. Then, we randomly sample their parameters based on the types of these basic functions to generate a set of function instances,

$$\mathcal{G}_{inst} = \{g_{ij}(x) \mid i \in [1, m\ j \in [1, \tau]\}, \quad (1)$$

where τ denotes the number of instances for each basic function $g_i(x)$. For each instance, we generate a numerical sequence of length 200, evenly spaced within the range of 1 to 200, to serve as the values for the independent variable x. We then input these x values into the generated function instances $g_{ij}(x)$ to obtain the corresponding $y = g_{ij}(x)$ values. From these y values, we construct the corresponding basic function series. These series are used as time series inputs for the LLMs to perform predictions.

Table 1: Samples were randomly generated for each function based on different **Basic** function types.

Genre	Vicuna		Qwen	
	MAE ↓	MSE ↓	MAE ↓	MSE ↓
Sine	**7.7298e-04**	**8.5829e-07**	**7.7298e-04**	**8.5829e-07**
Cosine	**7.5509e-04**	**9.3791e-07**	**7.6130e-04**	**8.9099e-07**
Absolute	8.0011e-01	2.0237e+00	8.3408e-01	2.1060e+00
Linear	6.4938e-01	1.5325e+00	6.7482e-01	1.5941e+00
Logarithm	3.2064e-02	1.3905e-03	2.3618e-02	8.4406e-04
Polynomial	5.5825e-01	8.2777e-01	6.3369e-01	1.0176e+00
Reciprocal	8.7832e-01	2.0281e+00	8.7093e-01	2.0235e+00
ReLU	5.0175e-01	1.0431e+00	5.1793e-01	1.0844e+00

Results The results are presented in Figure 1 (a.) (based on LLaMA2), showing that LLMs perform best on series derived from periodic functions (sine and cosine). Additionally, the Table 10 presents the outcomes of different basic function series on the Vicuna and Qwen. It can be observed that, compared to other function sequences, LLMs are particularly more adept at predicting periodic functions (sine and cosine) series. This suggests that LLMs excel at recognizing repetitive patterns in input series and mimicking them in the outputs.

4.2 Periodic Function Series

Method Based on the observation, we further analyze the periodic function series. Specifically, we explore three parameters of periodic function series: amplitude ($\mathcal{A}$), frequency ($\mathcal{F}$), and phase ($\mathcal{P}$). For sine periodic functions $g^{sin}(x) = \mathcal{A}\sin(\mathcal{F}x + \mathcal{P})$, we vary the values of amplitude ($\mathcal{A}$), frequency ($\mathcal{F}$), and phase ($\mathcal{P}$) separately to obtain sine function instances,

$$\mathcal{G}^{sin}_{inst,\{\mathcal{A},\mathcal{F},\mathcal{P}\}} = \{g^{sin}_j(x) \mid j \in [1, \tau]\}. \quad (2)$$

Here, the subscript $\{\mathcal{A}, \mathcal{F}, \mathcal{P}\}$ denotes the parameters we randomly sample. We only randomly sam-

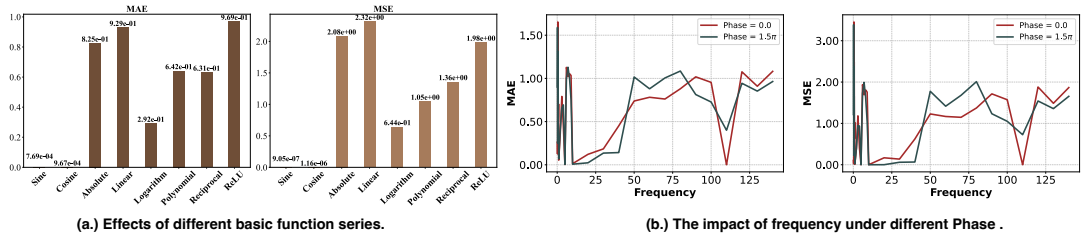


Figure 1: (a.) The effects of different basic function series. (b.) The impact of frequency under phase 0.0 and 1.5π.

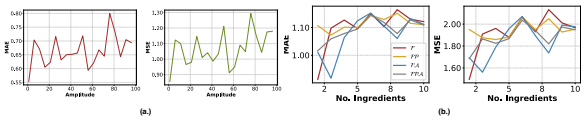


Figure 2: (a.) The impact of amplitude under phase 0.0. (b.) The synthetic impact of the quantities of frequency and amplitude ingredients under phase 0.0.

ple one parameter while keeping the others constant, thus obtaining these three sets of instances $\mathcal{G}^{sin}_{inst,\{\mathcal{A}\}}$, $\mathcal{G}^{sin}_{inst,\{\mathcal{F}\}}$, and $\mathcal{G}^{sin}_{inst,\{\mathcal{P}\}}$. Similarly, we continue to use x from 4.1 as the independent variable and input the numerical series $y = g^{sin}(x)$ obtained from periodic functions into LLMs for prediction. We utilize LLMs to predict the obtained function series.

Results Figure 1 (b.) illustrates the impact of frequency variations on LLMs' predictive performance for sine function series with different phases. In these experiments, the length of the input series is fixed at 200. When the frequency is very low, the fixed series length results in less than one or half a cycle of the input series. During these instances, LLMs do not receive sufficient series pattern information, leading to significantly reduced prediction accuracy. As the frequency increases, LLMs acquire enough information from the input series, resulting in noticeable improvements in prediction performance. However, with continued frequency increases, the prediction performance, as measured by MAE and MSE, declines to some extent. This indicates that LLMs have a limit in their capacity to capture series patterns, making it difficult to accurately recognize patterns in high-frequency series. Furthermore, variations in phase have minimal impact on LLMs' ability to predict periodic function series.

Figure 2 (a.) shows the effect of increasing amplitude values on LLMs' predictive capabilities. Here, we fixed the phase at 0.0 and the frequency at 10. Although the prediction performance series exhibit fluctuations, there is a clear trend of decreasing predictive ability as the amplitude increases. This indicates that while the overall series pattern remains unchanged, the large numerical values in the input series negatively impact LLMs' understanding of the time series.

4.3 Synthetic Series

Method Previously, we analyzed individual periodic function series. In this section, we combine multiple Sinusoidal series with varying amplitude ($\mathcal{A}$), frequency ($\mathcal{F}$), and phase ($\mathcal{P}$) to create the more complex synthetic series. We adjust the number of ingredients η in the synthetic series and restrict the parameters of the ingredients used for synthesis, denoted as $\mathcal{F}$/$\mathcal{A}$/$\mathcal{P}$ for frequency, amplitude, and phase, respectively. Thus, we can obtain a series of synthetic function instance sets,

$$\mathcal{G}^{sin}_{inst,\Omega} = \{g^{sin,\eta}_j(x) \mid j \in [1,\tau]\}. \quad (3)$$

The subscript Ω indicates $\{\mathcal{F}, \mathcal{FA}, \mathcal{FP}, \mathcal{FPA}\}$, the adjusted parameters for each instance set, allowing simultaneous adjustment of two or more parameters. And $g^{sin,\eta}$ indicates that the function is composed of the sum of η functions g^{sin}. We use the same method in Section 4.1 to obtain the series and input them into LLMs for prediction.

Results The results, shown in Figure 2 (b.), indicate that as the number of synthetic ingredients increases, the complexity of the composite prediction rises, and the predictive performance of LLMs declines. This suggests that LLMs have a significantly reduced capacity to understand synthetic series with varying amplitude, frequency, and phase. Additionally, from the perspective of variable adjustments, when comparing $\mathcal{FP}$ and $\mathcal{FA}$ to $\mathcal{F}$, even though the impact of phase on individual series is minimal, the synthetic series with different phases present a greater challenge to LLMs than those with different amplitudes. This results in a more significant decline in LLMs' performance on series synthesized by varying phases.

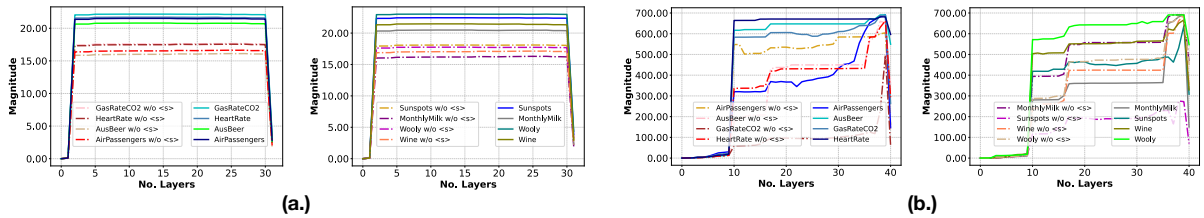


Figure 3: Layer-wise Analysis. The variation in the average token impact values across different layers. {Datasets} w/o <*s*> denotes the input series without <*s*>. The result of (a.) is from LLaMA2 (7B), and (b.) is from Qwen.

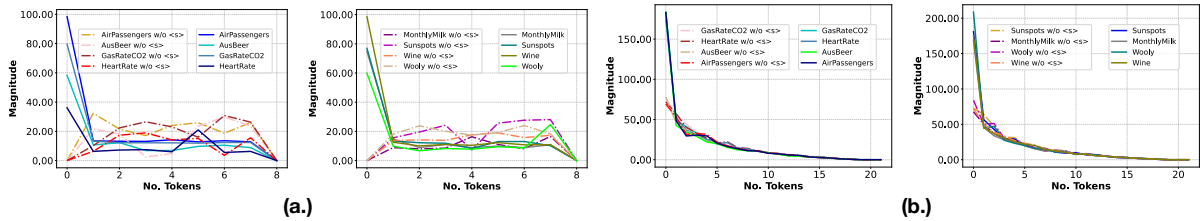


Figure 4: Sequence-wise Analysis. The variation in the average token impact values across token series (time series) on the darts dataset. The subfigure (a.) is from LLaMA3 (8B), and (b.) is from LLaMA2 (7B).

5 Token Perturbation Probing

5.1 Method

In this section, we employ a Perturbation Token Probing method to investigate the extent of mutual impact among time series tokens within each layer when utilizing LLMs for zero-shot time series forecasting. Given a series of numeric tokens $\mathbf{u} = \langle u_1, u_2, \cdots, u_n \rangle$ of the input time series T, LLMs can map each u_i into a contextualized representation $\mathbf{h}_\theta(\mathbf{u})_i$, where θ represents the LLM's parameters. Following (Wu et al., 2020), we can derive an impact function $\phi(x_i, x_j)$ to capture the impact of an arbitrary token u_j on token u_i,

$$\phi(u_i, u_j) = d\left(\mathbf{h}_\theta(\mathbf{u}\backslash\{u_i\})_i, \mathbf{h}_\theta(\mathbf{u}\backslash\{u_i, u_j\})_i\right). \tag{4}$$

We add a special token to the LLMs, replacing a token u_i in the input series. Then, we input this modified series $\mathbf{u}\backslash\{u_i\}$ into the LLMs to obtain the representation $\mathbf{h}_\theta(\mathbf{u}\backslash\{u_i\})_i$ for this token u_i. Next, we further replace an arbitrary token u_j with this special token, and use the obtained representation as the new representation $\mathbf{h}_\theta(\mathbf{u}\backslash\{u_i, u_j\})_i$ for u_i. The above $d(\cdot, \cdot)$ represents the Euclidean distance metric. Through this method, we can obtain the value of the impact function $\phi(u_i, u_j)$. By repeating the perturbation process iteratively, we can obtain an impact matrix $\mathcal{E} \in \mathbb{R}^{n \times n}$. Based on the impact matrix, we conducted analysis, the results are as follows.

5.2 Results

5.2.1 Layer-wise Analysis

Figure 3 illustrates the Layer-wise Analysis for Token Perturbation Probing, which indicates the variation in average impact values across different layers. Specifically, we obtain impact matrices through Perturbation Token Probing, average them by layer, and record the mean values. We also compare the impact of the presence or absence of the <*s*> token in the input time series on the impact matrices. It can be observed that the average impact values are relatively low in the early layers (e.g., 0-1 layers in LLaMA2 (7B)) and the final layer (e.g., 31 layer in LLaMA2 (7B)), while they are higher in the intermediate layers. We suppose that the low average impact values in the initial layers indicate the model is still in the exploratory phase, and the interactions between tokens are not very pronounced. In contrast, the significant changes in the average impact values in the final layer might suggest that the model is preparing for generation and prediction, hence modifying the impact matrix accordingly. When the <*s*> token is removed, the average impact values in the intermediate layers decrease. This is due to, without the <*s*> token, the values in the impact matrix are more dispersed (the presence of <*s*> concentrates more impact), leading to a reduction in average impact values.

5.2.2 Sequence-wise Analysis

In addition, we also measure the variation in average impact values as the time series tokens varia-

tions, which is a sequence-wise analysis for Token Perturbation Probing. Specifically, we average the impact values along the layer dimension and for a specific token dimension to obtain the average impact of each time series token. The results (Figure 4) show that in the presence of the <*s*> token (special token in LLMs), more time series tokens tend to focus on the information from <*s*>, making it the most influential. Without the <*s*> token, the impact values are more dispersed.

6 Linear Probing

6.1 Method

We design linear probing (Alain and Bengio, 2018) experiments to further explore how LLMs understand time series. Linear probing allows us to test whether intermediate layer outputs (representations of time series tokens) $\mathbf{h}(u_i)^\ell$ (ℓ denotes the layer) contains quantities of interest. Here, we trained a small probe regressor φ to probe the effect of using these intermediate representations $\mathbf{h}(u_i)^\ell$ directly for time series prediction. Specifically, we use a two-layer multilayer perceptron (MLP) as the probe regressor.

$$\varphi(\mathbf{h}) = \mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{h} + \mathbf{b}_1) + \mathbf{b}_2, \qquad (5)$$

where $\mathbf{W}_2$, $\mathbf{W}_1$, $\mathbf{b}_1$ and $\mathbf{b}_2$ denotes learnable parameters, and we abbreviate $\mathbf{h}(u_i)^\ell$ as $\mathbf{h}$. Then, we compare the probe regressor's predictions with the actual data and perform a statistical analysis on the computed Mean Squared Error (MSE).

6.2 Results

LLMs may malfunction when the length of patterns exceeds a certain threshold. To give a quantitative analysis, we design a linear probing method (Alain and Bengio, 2017; Guo et al., 2023) to test each time series token at every layer. Linear probing linearly regress quantities of interest (tokens x_i) on each intermediate layer output of token x_i (h_i^l), where l denotes the layers and i denotes the time series tokens. We evaluate the predictive performance using linear probing, with MSE as the metric. Based on the test results, we investigate the position of the time series token with the smallest MSE in each layer, as shown in Figure 5 and 6. Linear probing reveals that the initial layers of LLMs tend to perform broad searches for tokens with the smallest MSE over the entire sequence length. As the layers progress, the search range gradually narrows or stabilizes, indicating more localized or refined token selection in later layers.

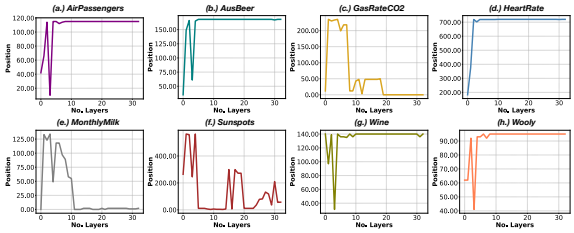


Figure 5: The results of linear probing on LLaMA2.

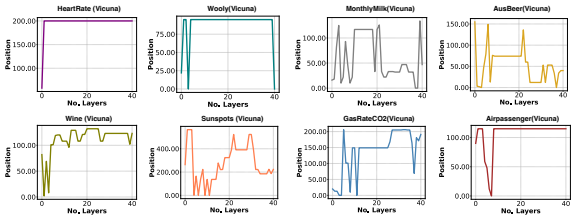


Figure 6: The results of linear probing on Vicuna.

7 Vocabulary Mapping Probing

7.1 Method

For the numeric tokens $\mathbf{u} = \langle u_1, u_2, \cdots, u_n \rangle$ of the input time series T, LLMs can provide us with a series of hidden state $\mathbf{H} = \{\mathbf{h}(u_1)^\ell, \mathbf{h}(u_2)^\ell, \mathbf{h}(u_n)^\ell\}$ at each intermediate layer ℓ. Following (Dar et al., 2023), we adopt a vocabulary mapping probing method to map these hidden states $\mathbf{H}$ into the vocabulary embedding space, allowing us to explore the representations of these time series tokens. Specifically, for a given numeric token $\mathbf{h}(u_i)^\ell \in \mathbb{R}^d$ in the time series, we map it to the vocabulary embedding space using the embedding matrix $\mathbf{E} \in \mathbb{R}^{V \times d}$,

$$\tilde{\mathbf{h}}(u_i)^\ell = \mathbf{E}\mathbf{h}(u_i)^\ell, \qquad (6)$$

where d denotes the hidden size of LLMs, and V denotes the vocabulary size. We select the top k vocabulary items with the highest logits in $\tilde{\mathbf{h}}(u_i)^\ell$ as the return results of the vocabulary probe. Thus, we can obtain the vocabulary mapping of each time series token $\mathbf{h}(u_i)^\ell$ at each layer ℓ.

7.2 Results

7.2.1 Numeric Mapping

We selected the representations of time series tokens from each layer and mapped them to the corresponding vocabulary tokens using the vocabulary mapping probe. The mapped tokens are ranked

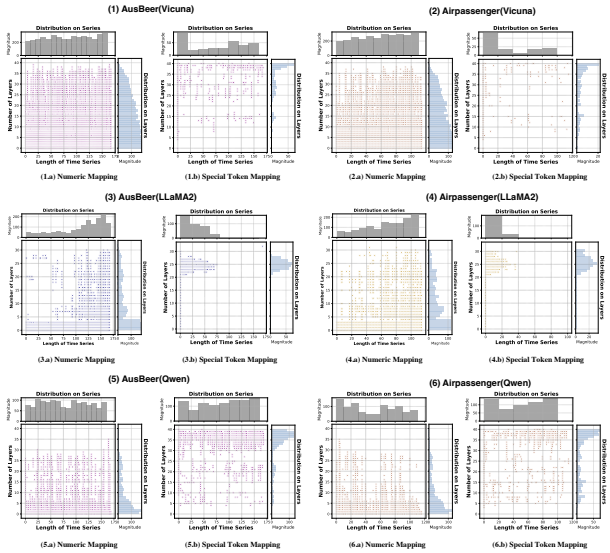


Figure 7: The results of Vocabulary Probing Probing for Vicuna, LLaMa2, and Qwen.

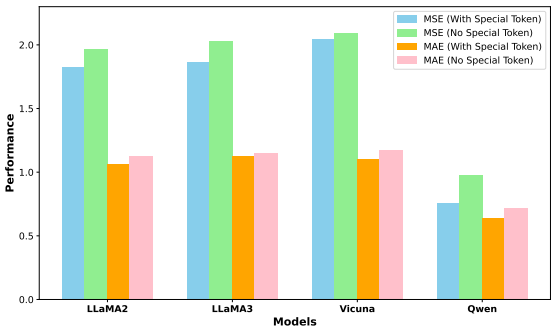


Figure 8: Performance with and without special tokens.

Table 2: Comparison results on baselines.

Model	AusBeer		AirPassengers		GasRateCO2		HeartRate	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Promptcast	1.2594	2.0353	0.7218	0.8821	0.8370	1.2809	1.1883	2.3561
Onefitall	1.2458	2.0408	1.0410	1.6880	1.0876	1.8837	1.0798	1.7404
Tempo	1.0847	1.8905	0.8882	1.1335	1.1283	1.9082	1.2239	2.2388
Time-LLM	1.1224	1.8743	1.0515	1.7423	1.0568	1.7433	1.1997	2.1329
LLMTime	0.9513	1.6420	0.9028	1.3850	1.2649	2.6860	1.2618	2.6131
Freq-Decomp	**0.6504**	**0.6295**	**0.7125**	**0.8028**	**0.6739**	**0.7108**	**0.8408**	**1.4376**
Model	**MonthlyMilk**		**Sunspots**		**Wine**		**Wooly**	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Promptcast	1.6009	3.1964	1.2402	2.3650	1.0717	1.9892	1.1377	2.0421
Onefitall	1.0826	1.7906	1.1737	2.0982	0.9565	1.2061	0.9711	1.3966
Tempo	1.1789	1.9440	1.1445	2.0355	0.8706	1.2373	1.0449	1.7496
Time-LLM	1.0372	1.5591	1.0506	1.7276	0.9170	1.4674	1.2998	2.5337
LLMTime	1.1724	1.9495	1.1403	1.9716	1.0875	1.6939	1.0561	1.7531
Freq-Decomp	**0.9272**	**1.1805**	**0.9634**	**1.3721**	**0.8024**	**1.0458**	**0.8781**	**1.2198**

top-k, and in this section, we analyze the first token to determine if it is numeric. The statistical results are shown in subfigure (#.a) in Figure 7, where # represent the case index in thise Figure. In the initial layers of LLMs (e.g., layers 0-3 in LLaMA2), nearly all time series tokens are mapped to numeric tokens. This indicates that the early layers primarily focus on observing local information (with time series inputs encoded as numeric values). The proportion of numeric tokens decreases in the later layers, suggesting that LLMs begin to explore global patterns beyond local numerical inputs.

7.2.2 Special Token Mapping

Simultaneously, we analyze the distributions of tokens mapped to the special token (e.g., <*s*> token in LLaMA2) through the vocabulary mapping probing. We select the top three mapped tokens, and if any of these tokens are <*s*>, we record this in Figure 7 (#.b) subfigures. In terms of layer depth, tokens mapped to <*s*> are primarily located in the later layers of LLMs. Combined with the results of the numeric mapping, the <*s*> mapping results complement the numeric mapping results. The areas where numeric mapping is sparse correspond to where <*s*> mapping occurs. Therefore, it can be inferred that the later layers of LLMs integrate global information and specific numerical information for time series prediction.

7.2.3 Special Token Influence

In Figure 8, we compare the performance of different LLMs with and without special tokens (e.g. <s> in LLaMA2). We statistic MSE and MAE on Darts dataset. The performance with special tokens of different LLMs consistently outperforms those without special tokens. It is indicate that using special tokens in zero-shot time series forecasting can help the performance enhancement.

8 Frequency Decomposition for Zero-shot Time Series Forecasting

Probing experiments are conducive to exploring the changes in LLMs during time series analysis. Building on these exploration, we propose the Freq-Decomp method to enhance LLMs' zero-shot forecasting capabilities. More detailed experiments and analysis are shown in Appendix A.3, F, and G.

8.1 Method

According previous observations, we found that LLMs excel at processing periodic series, but an excessive synthesis of series of different periods also impair the performance of LLMs (Section 4). Meanwhile, LLMs may perform better with simpler series (Section 6), and benefit from the use of special tokens (Section 5 and 7). Thus, Freq-Decomp method introduces a preprocessing method to decompose the original time series into frequency bands before using LLMs. This enables LLMs to handle the time seires patterns they are proficient

in, leveraging their strengths in processing time series. Concretely, we transform the initial input time series $T = \langle t_1, t_2, \cdots, t_n \rangle$ into the frequency domain using the Fast Fourier Transform (FFT) operation $\mathscr{F}$,

$$\Omega = \mathscr{F}(T), \tag{7}$$

where the frequency domain representation of T is $\Omega = \langle \omega_1, \omega_2, \cdots, \omega_n \rangle$. Then, we partition the series Ω into μ frequency bands at equal intervals $\Omega_B = \langle B_1, B_2, \cdots, B_\mu \rangle$. These bands are ordered from low to high frequency. We apply the Inverse Fast Fourier Transform (IFFT) $\mathscr{F}^{-1}$ to map frequency bands Ω_B back to time domain,

$$S = \mathscr{F}^{-1}(\Omega_B), \tag{8}$$

where $S = \langle s_1, s_2, \cdots, s_\mu \rangle$ is a collection of sub-series corresponding to individual bands in Ω_B. We utilize LLMs to forecast each sub-series s using the same method applied in previous sections. This yields a collection of predicted output series $S^o = \langle s_1^o, s_2^o, \cdots, s_\mu^o \rangle$. Finally, we sum all the sub-series in S^o to obtain the prediction results for the original time series T.

8.2 Comparison Results

The comparison results with baselines are shown in Table 2. It is evident that Freq-Decomp consistently outperforms LLMs-based forecasting methods under the zero-shot scenario. The baselines include: **Promptcast (Xue and Salim, 2023)**, **LLMTime (Gruver et al., 2023)**, textbfOnefitall (Zhou et al., 2023), **Tempo (Cao et al., 2024)**, and **Time-LLM (Jin et al., 2023)**. Details are shown in Appendix A.1 . Aside from the Promptcast and LLMTime, which are designed for zero-shot settings, methods such as Onefitall, Tempo, and Time-LLM are re-customizing LLMs and originally developed for supervised time series forecasting. All methods are compared under the zero-shot scenario for fairness.

Table 3: Results on ETTh1 and ETTm1 Datasets.

Model	ETTh1		ETThm1	
	MAE	MSE	MAE	MSE
Autoformer	0.569	0.693	0.576	0.735
FEDformer	0.502	0.509	0.553	0.698
Onefitall	0.577	0.732	0.558	0.747
PatchTST	0.465	0.485	0.437	0.491
Tempo	0.406	0.400	0.424	0.438
Time-LLM	0.452	0.450	0.397	**0.359**
Freq-Decomp	**0.346**	**0.224**	**0.255**	0.397

Table 4: Ablation study on different LLMs.

Model	AusBeer		AirPassengers		GasRateCO2		HeartRate	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
LLaMA2	0.9513	1.6420	0.9028	1.3850	1.2649	2.6860	1.2618	2.6131
+ Freq-Decomp	**0.6504**	**0.6295**	**0.7125**	**0.8028**	**0.6739**	**0.7108**	**0.8408**	**1.4376**
LLaMA3	1.2948	2.2790	0.9113	1.6871	1.1912	2.4585	1.1178	2.0000
+ Freq-Decomp	**0.9380**	**1.5944**	**0.7848**	**0.8989**	**1.0078**	**1.9738**	**1.1047**	**1.9393**
Vicuna	1.1949	2.4200	0.9777	1.7924	1.6231	3.4377	1.5310	3.2122
+ Freq-Decomp	**0.3437**	**0.1726**	**0.8069**	**1.0032**	**0.8850**	**1.1704**	**1.1797**	**2.0033**
CodeLlama	0.9122	1.3219	1.0653	1.8358	1.2615	2.2176	0.9792	1.4051
+ Freq-Decomp	**0.4096**	**0.2676**	**0.7125**	**0.8028**	**0.9349**	**1.3342**	**0.8330**	**1.0942**
Model	**MonthlyMilk**		**Sunspots**		**Wine**		**Wooly**	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
LLaMA2	1.1724	1.9495	1.1403	1.9716	1.0875	1.6939	1.0561	1.7531
+ Freq-Decomp	**0.9272**	**1.1805**	**0.9634**	**1.3721**	**0.8024**	**1.0458**	**0.8781**	**1.2198**
LLaMA3	1.3069	2.4724	1.1114	1.9073	0.8961	1.6553	1.1070	1.6778
+ Freq-Decomp	**0.9354**	**1.2516**	**0.9903**	**1.8304**	**0.8825**	**1.5543**	**0.7708**	**1.1372**
Vicuna	1.3735	2.5699	1.2510	2.3140	1.0553	2.0337	1.4323	2.8917
+ Freq-Decomp	**1.1324**	**1.9740**	**0.8653**	**1.2834**	**0.8466**	**1.1081**	**0.7452**	**1.0054**
CodeLlama	1.1585	2.0072	1.1951	2.0455	1.2467	2.3468	1.3720	2.8167
+ Freq-Decomp	**0.9208**	**1.1957**	**0.5417**	**0.5424**	**0.7627**	**1.1446**	**0.6665**	**0.7368**

8.3 Results on ETT Dataset

We conduct experiments on the ETT dataset. We introduce two extra baselines: **Autoformer (Wu et al., 2021)** and **FEDformer (Zhou et al., 2022)**. Table 3 summarizes the results on ETTh1 and ETTm1. Our method demonstrates comparable performance advantages compared to these baseline models on both datasets. Notably, these results of Freq-Decomp are achieved without any training on time-series dataset. The results from other models rely on transfer learning, requiring pre-training on source time-series datasets before testing on target datasets. Freq-Decomp eliminates the need for such training, making it independent of specific time-series datasets and associated training overhead. Detailed settings are in Appendix A.2.

8.4 Ablation Study on Different LLMs

Table 4 presents the performance of Freq-Decomp compared to directly utilizing LLMs to forecasting. It is evident that Freq-Decomp achieves significant improvements across different LLMs.

9 Conclusion and Future Works

We explore the capability of LLMs to understand time series. Firstly, we investigate the LLMs' ability to complete predictions for various functions and examine the effects of amplitude, frequency, phase on periodic function series. Additionally, we design three probes to further investigate the LLMs' understanding of time series. Finally, we propose the Freq-Decomp method for enhancing LLMs' time series forecasting capabilities. We leave for future work the exploration of domain generalization in LLMs for time series, multimodal extensions, and advanced inference techniques like in-context learning, chain of thought, etc.

Limitations

This study is exploratory in nature, aiming to investigate how large language models (LLMs) understand and process time series through a series of methods. We conducted an in-depth analysis of LLMs' handling of time series, arriving at clear conclusions and proposing enhancement methods. However, research in this area is still in its early stages. Key aspects such as zero-shot time series prediction, the foundational architecture of time series prediction models, time series processing base on in-context learning, and multimodal processing that integrates time series with other modals require further investigation. Additionally, there are no potential risks associated with the research presented in this paper.

Ethical Considerations

This work relies solely on publicly available benchmark datasets that do not contain sensitive personal information or content related to bias or discrimination. There are no notable ethical or societal risks associated with this research. And we used existing artifacts (e.g., datasets or models) in accordance with their specified intended use. We verified the licenses or usage guidelines before employing them in our research. For the artifacts we created, we clearly specify that they are intended solely for research use.

References

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Guillaume Alain and Yoshua Bengio. 2018. Understanding intermediate layers using linear classifier probes. *Preprint*, arXiv:1610.01644.

Taghreed Alghamdi, Khalid Elgazzar, Magdi Bayoumi, Taysseer Sharaf, and Sumit Shah. 2019. Forecasting traffic congestion using ARIMA modeling. In *15th International Wireless Communications & Mobile Computing Conference, IWCMC 2019, Tangier, Morocco, June 24-28, 2019*, pages 1227–1232. IEEE.

Tom B. Brown and Benjamin Mann et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Defu Cao, Furong Jia, Sercan O Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. 2024. TEMPO: Prompt-based generative pre-trained transformer for time series forecasting. In *The Twelfth International Conference on Learning Representations*.

Sena Chae, Anahita Davoudi, Jiyoun Song, Lauren Evans, Mollie Hobensack, Kathryn H. Bowles, Margaret V. McDonald, Yolanda Barrón, Sarah Collins Rossetti, Kenrick Cato, Sridevi Sridharan, and Maxim Topaz. 2023. Predicting emergency department visits and hospitalizations for patients with heart failure in home healthcare using a time series risk model. *J. Am. Medical Informatics Assoc.*, 30(10):1622–1633.

François Chollet. 2019. On the measure of intelligence. *CoRR*, abs/1911.01547.

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. 2023. Analyzing transformers in embedding space. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 16124–16170. Association for Computational Linguistics.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2023. A survey for in-context learning. *CoRR*, abs/2301.00234.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yuchen Fang, Yanjun Qin, Haiyong Luo, Fang Zhao, and Kai Zheng. 2024. Stwave$^+$: A multi-scale efficient spectral graph attention network with long-term trends for disentangled traffic flow forecasting. *IEEE Trans. Knowl. Data Eng.*, 36(6):2671–2685.

John Cristian Borges Gamboa. 2017. Deep learning for time-series analysis. *CoRR*, abs/1701.01887.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. 2021. Monash time series forecasting archive. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. 2023. Large language models are zero-shot time series forecasters. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Yuechun Gu, Da Yan, Sibo Yan, and Zhe Jiang. 2020. Price forecast with high-frequency finance data: An autoregressive recurrent neural network model with technical indicators. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 2485–2492. ACM.

Tianyu Guo, Wei Hu, Song Mei, Huan Wang, Caiming Xiong, Silvio Savarese, and Yu Bai. 2023. How do transformers learn in-context beyond simple functions? A case study on learning with representations. *CoRR*, abs/2310.10616.

Julien Herzen, Francesco Lässig, Samuele Giuliano Piazzetta, Thomas Neuer, Léo Tafti, Guillaume Raille, Tomas Van Pottelbergh, Marek Pasieka, Andrzej Skrodzki, Nicolas Huguenin, Maxime Dumonal, Jan Koscisz, Dennis Bader, Frédérick Gusset, Mounir Benheddi, Camila Williamson, Michal Kosinski, Matej Petrik, and Gaël Grosch. 2022. Darts: User-friendly modern machine learning for time series. *J. Mach. Learn. Res.*, 23:124:1–124:6.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. 2023. Time-llm: Time series forecasting by reprogramming large language models. *CoRR*, abs/2310.01728.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloé Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross B. Girshick. 2023. Segment anything. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 3992–4003. IEEE.

Siddique Latif, Moazzam Shoukat, Fahad Shamshad, Muhammad Usama, Heriberto Cuayáhuitl, and Björn W. Schuller. 2023. Sparks of large audio models: A survey and outlook. *CoRR*, abs/2308.12792.

Alberto Lerner, Dennis E. Shasha, Zhihua Wang, Xiaojian Zhao, and Yunyue Zhu. 2004. Fast algorithms for time series with applications to finance, physics, music, biology, and other suspects. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, Paris, France, June 13-18, 2004*, pages 965–968. ACM.

Toni J. B. Liu, Nicolas Boullé, Raphaël Sarfati, and Christopher J. Earls. 2024a. Llms learn governing principles of dynamical systems, revealing an in-context neural scaling law. *CoRR*, abs/2402.00795.

Xu Liu, Junfeng Hu, Yuan Li, Shizhe Diao, Yuxuan Liang, Bryan Hooi, and Roger Zimmermann. 2024b. Unitime: A language-empowered unified model for cross-domain time series forecasting. In *Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, May 13-17, 2024*, pages 4095–4106. ACM.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2023. itransformer: Inverted transformers are effective for time series forecasting. *CoRR*, abs/2310.06625.

Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. 2024c. Autotimes: Autoregressive time series forecasters via large language models. *CoRR*, abs/2402.02370.

Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. 2023. Large language models as general pattern machines. *Preprint*, arXiv:2307.04721.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A time series is worth 64 words: Long-term forecasting with transformers. *Preprint*, arXiv:2211.14730.

Hao Niu, Yun Xiong, Xiaosu Wang, Wenjing Yu, Yao Zhang, and Weizu Yang. 2023a. KeFVP: Knowledge-enhanced financial volatility prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11499–11513, Singapore. Association for Computational Linguistics.

Mengjia Niu, Yuchen Zhao, and Hamed Haddadi. 2023b. Effective abnormal activity detection on multivariate time series healthcare data. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking, ACM MobiCom 2023, Madrid, Spain, October 2-6, 2023*, pages 134:1–134:3. ACM.

Catherine Olsson and Nelson Elhage et al. 2022. In-context learning and induction heads. *CoRR*, abs/2209.11895.

Maxime Oquab and Timothée Darcet et al. 2023. Dinov2: Learning robust visual features without supervision. *CoRR*, abs/2304.07193.

Linlu Qiu, Liwei Jiang, Ximing Lu, Melanie Sclar, Valentina Pyatkin, Chandra Bhagavatula, Bailin Wang, Yoon Kim, Yejin Choi, Nouha Dziri, and Xiang Ren. 2023. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. *CoRR*, abs/2310.08559.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision. *Preprint*, arXiv:2212.04356.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. 2023. Are emergent abilities of large language models a mirage? In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Chong Min John Tan and Mehul Motani. 2023. Large language model (LLM) as a system of multiple expert agents: An approach to solve the abstraction and reasoning corpus (ARC) challenge. *CoRR*, abs/2310.05146.

Hugo Touvron and Louis Martin et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. 2021. Time series data augmentation for deep learning: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4653–4660. ijcai.org.

Qingsong Wen, Linxiao Yang, Tian Zhou, and Liang Sun. 2022. Robust time series analysis and applications: An industrial perspective. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 4836–4837. ACM.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. 2023. Transformers in time series: A survey. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 6778–6786. ijcai.org.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22419–22430.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4166–4176. Association for Computational Linguistics.

Hao Xue and Flora D. Salim. 2023. Promptcast: A new prompt-based learning paradigm for time series forecasting. *Preprint*, arXiv:2210.08964.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting? In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 11121–11128. AAAI Press.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 11106–11115. AAAI Press.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 27268–27286. PMLR.

Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. 2023. One fits all: Power general time series analysis by pretrained LM. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

A Appendix

A.1 Baselines

The baselines in Section include: **Promptcast (Xue and Salim, 2023)** utilizes LLMs as forecasters by converting time series into numerical tokens; **LLMTime (Gruver et al., 2023)** transforms the time series data by tokenizing and rescaling it, treating the processed time series as numerical tokens; **Onefitall (Zhou et al., 2023)** leverages LLMs' certain intermediate layers, enhancing them with temporal embeddings and prediction heads for forecasting; **Tempo (Cao et al., 2024)** explores a soft prompting strategy to fine-tune specific parameters of LLMs for forecasting. **Time-LLM (Jin et al., 2023)** introduces a reprogramming approach to map time series into the textual space of LLMs. **Autoformer (Wu et al., 2021)** revise Transoformer by introducing the Auto-correlation to replace the Self-attention. **FEDformer (Zhou et al., 2022)** introduce a frequency enhanced decomposed Transformer to reduce prediction error and enhance processing efficiency

A.2 Settings for ETT Dataset

For this experiment, the forecasting horizon is set to 96. The results for Tempo and FEDformer are sourced from Table 1 of the Tempo paper [3], while the results for other baseline models are obtained from Table 16 of the Time-LLM paper (ETTh2 ->

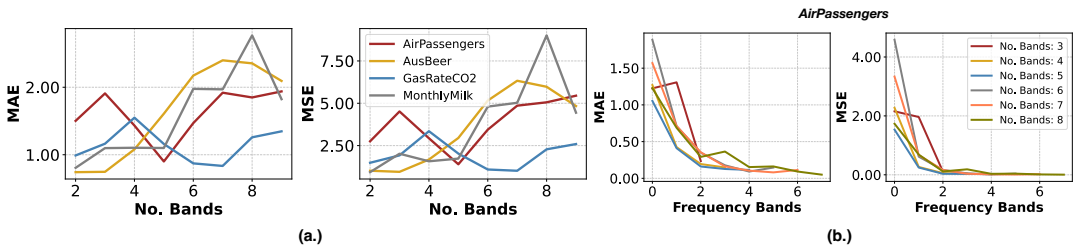


Figure 9: (a.) The impact of the quantities of decomposed frequency Bands. (b.) The impact of frequency bands on performance, ranging from low to high frequencies.

ETTh1 row). The LLMs employed in this experiments is LLaMA2 (7B).

A.3 Analyzing the Number of Frequency Bands

Figure 9 (a.) illustrates the impact of the number of frequency bands. It shows that dividing the input time series into approximately five frequency bands yields the best prediction results. Figure 9 (b.) displays the performance of LLMs across different frequency bands on the entire dataset. Regardless of how the frequency bands are divided (with band counts ranging from 3 to 8), LLMs perform poorly on low-frequency bands, while their prediction accuracy improves significantly on high-frequency bands.

A.4 Implementation Details

All the deep networks are implemented in Pytorch and run on A800-SXM4-80GB GPUs. Owing to the zero-shot scenario, LLMs were employed for inference on datasets. We draw 5 samples from LLMs, and use the median statistics of samples to calculate MSE and MAE. In addition, due to the relatively small number of samples in the Darts dataset, the runtime remains within 30 minutes. In contrast, for the ETT and Monash datasets, the larger data scale leads to longer runtimes, ranging from 5 to 10 hours. During the training of linear probing (Section 6), we utilized 30 epochs, a prediction length of 20, a learning rate of 0.0005. The LLMs used in this work include LLaMA3 (8B), Vicuna (13B), CodeLlama (7B), LLaMA2 (7B), and Qwen (14B).

In addition, for exploration of fundamental time series patterns, we also employ basic functions as well as their synthetic forms to judge LLMs, which are described in Section 4.

B Dataset Statistics

B.1 Darts

Darts(Herzen et al., 2022) is a powerful time series forecasting library that provides a unified Time Series data container and consistent API, supports various classical and deep learning models, and offers comprehensive features such as co-variate handling, probabilistic forecasting, ensemble learning, and more, suitable for a wide range of time series modeling and forecasting tasks. More details are show in Table 5.

B.2 Monash

Monash(Godahewa et al., 2021) datasets span diverse domains like tourism, banking, web, energy, sales, economics, transportation, health, and nature. They have varying sampling rates, from yearly to high-frequency 4-second intervals, and include both univariate and multivariate series aligned with known timestamps, from which 58 derived datasets with different frequencies and missing value treatments have been created, with 7 newly curated datasets and 23 standardized from various sources, all carefully vetted for inclusion in the repository. We selected four of the datasets for our experiments, more details are shown in Table 6.

C Basic Function

We used 8 basic function, including sine, cosine, absolute, linear, logarithm, polynomial, reciprocal and relu function. And we randomly generate four samples for each function. The detailed results based on LLaMA2 are presented in Table 7. Additionally, the Table 10 presents the outcomes of different basic function series on the Vicuna and Qwen. It can be observed that, compared to other function sequences, LLMs are particularly more adept at predicting periodic functions (sine and cosine) series.

Table 5: Using (Mean(Stabard Deviation)) statistics for the data in columns L(Input) and L(Prediction).

Datasets(Darts)	L**(Input)**	L**(Prediction)**	**#Case**
AirPassengersDataset	115(0)	29(0)	1
AusBeerDataset	168(0)	43(0)	1
GasRateCO2Dataset	236(0)	60(0)	1
MonthlyMilkDataset	134(0)	34(0)	1
SunspotsDataset	564(0)	141(0)	1
WineDataset	140(0)	36(0)	1
WoolyDataset	95(0)	24(0)	1
HeartRateDataset	720(0)	180(0)	1

Table 6: Using (Mean(Standard Deviation)) statistics for the data in columns L(Input) and L(Prediction).

Datasets(Monash)	L**(Input)**	L**(Prediction)**	**#Case**
bitcoin	4156.89(467.27)	30.00(0.00)	18
nn5 daily	735.00(0.00)	56.00(0.00)	111
fred md	716.00(0.00)	12.00(0.00)	107
tourism monthly	274.58(55.58)	24.00(0.00)	366

Table 7: Four samples were randomly generated for each function based on different **Basic** function types. Complementary results are based on LLaMA2-7b-chat.

Genre	**Expression (Basic)**	**MAE** ↓	**MSE** ↓
Sine $x \mapsto \sin\left(\frac{2\pi}{b}(x-c)\right)$	$1.1 * np.sin((2 * np.pi/2.8) * (x - 4.0)) + -27.4$	9.6658e-04	1.1862e-06
	$0.1 * np.sin((2 * np.pi/4.5) * (x - 0.2)) + -29.1$	6.2864e-04	6.3579e-07
	$8.4 * np.sin((2 * np.pi/5.0) * (x - 5.5)) + -9.6$	8.4970e-04	1.1654e-06
	$-0.3 * np.sin((2 * np.pi/1.3) * (x - 6.2)) + -27.1$	6.3207e-04	6.3277e-07
	Mean(Std)	**7.6925e-04 (1.6733e-04)**	**9.0504e-07 (3.1276e-07)**
Cosine $x \mapsto \cos\left(\frac{2\pi}{b}(x-c)\right)$	$3.5 * np.cos((2 * np.pi/3.9) * (x - 4.0)) + 9.2$	1.0383e-03	1.2949e-06
	$-8.2 * np.cos((2 * np.pi/1.6) * (x - 1.3)) + -13.6$	9.6355e-04	1.2982e-06
	$-4.9 * np.cos((2 * np.pi/3.2) * (x - 1.3)) + 25.0$	9.2881e-04	1.0349e-06
	$2.9 * np.cos((2 * np.pi/2.3) * (x - 2.2)) + 7.9$	9.3550e-04	1.0040e-06
	Mean(Std)	**9.6654e-04 (5.0151e-05)**	**1.1580e-06 (1.6049e-07)**
Absolute $x \mapsto \lvert x \rvert$	$1.2 * np.abs(x)$	8.2498e-01	2.0839e+00
	$9.2 * np.abs(x)$	8.2498e-01	2.0839e+00
	$1.0 * np.abs(x)$	8.2498e-01	2.0839e+00
	$21.7 * np.abs(x)$	8.2498e-01	2.0839e+00
	Mean(Std)	**8.2498e-01 (0.0000e+00)**	**2.0839e+00 (0.0000e+00)**
Linear $x \mapsto ax + b$	$2.7 * x + -8.7$	8.2498e-01	2.0839e+00
	$2.5 * x + 5.4$	8.2229e-01	2.0700e+00
	$3.9 * x + -4.5$	8.2498e-01	2.0839e+00
	$2.2 * x + -7.0$	1.2426e+00	3.0252e+00
	Mean(Std)	**9.2871e-01 (2.0926e-01)**	**2.3158e+00 (4.7301e-01)**
Logarithm $x \mapsto a\frac{log(x)}{log(b)} + c$	$-7.1 * (np.log(x)/np.log(6.6)) + 2.4$	2.6773e-02	1.1731e-03
	$-2.6 * (np.log(x)/np.log(1.5)) + -21.3$	4.6028e-02	3.6697e-03
	$-13.6 * (np.log(x)/np.log(9.4)) + 3.0$	5.7175e-02	5.0419e-03
	$1.6 * (np.log(x)/np.log(9.6)) + 20.3$	1.0371e+00	2.5663e+00
	Mean(Std)	**2.9177e-01 (4.9705e-01)**	**6.4405e-01 (1.2815e+00)**
Polynomial $x \mapsto \sum_{i=1}^{n} ax^2$	$0.8 * x * *5 + 2.4 * x * *4 + -4.5 * x * *3 + -4.6 * x * *2 + 3.4 * x * *1 + -4.0 * x * *0$	5.3509e-01	1.2728e+00
	$-1.2 * x * *2 + 0.9 * x * *1 + 0.8 * x * *0$	6.5239e-01	5.6994e-01
	$2.7 * x * *2 + 4.2 * x * *1 + -4.7 * x * *0$	6.3632e-01	1.6108e+00
	$-2.7 * x * *2 + 4.6 * x * *1 + 1.0 * x * *0$	7.4325e-01	7.3617e-01
	Mean(Std)	**6.4176e-01 (8.5286e-02)**	**1.0474e+00 (4.8064e-01)**
Reciprocal $x \mapsto \frac{1}{ax}$	$1/(-2.0 * x)$	1.0940e+00	2.6786e+00
	$1/(8.7 * x)$	1.7257e-01	4.0537e-02
	$1/(-1.3 * x)$	1.0940e+00	2.6786e+00
	$1/(1.9 * x)$	1.6460e-01	3.8106e-02
	Mean(Std)	**6.3129e-01 (5.3430e-01)**	**1.3590e+00 (1.5238e+00)**
ReLU $x \mapsto max(0, x)$	$-29.5 * np.where(x > 0, x, x * 0.8) + 8.7$	4.4993e-01	2.7988e-01
	$-12.8 * np.where(x > 0, x, x * 0.5) + 16.0$	1.7776e+00	3.4890e+00
	$6.9 * np.where(x > 0, x, x * 0.5) + -6.4$	8.2498e-01	2.0839e+00
	$6.0 * np.where(x > 0, x, x * 0.9) + -22.5$	8.2498e-01	2.0839e+00
	Mean(Std)	**9.6937e-01 (5.6708e-01)**	**1.9842e+00 (1.3152e+00)**

D Periodic Function Setup

In terms of the distinct combinations of amplitude ($\mathcal{A}$), frequency ($\mathcal{F}$), and phase ($\mathcal{P}$), we synthesize a series of synthetic functions. Then, we feed them into LLM, to judge the impact of the three factors and synthetic complexity on performance.

Our specific settings for amplitude, frequency, and phase are as follows:

- Amplitude: [1.0, 5.0, 10.0, 20.0, 50.0, 100.0],
- Phase: [0.0, 1.5],
- Frequency: Three segments:
  - In the range [0.1, 1.0] with a step size of 0.1.
  - In the range [1.0, 10.0] with a step size of 1.0.
  - In the range [10.0, 200.0] with a step size of 10.0.

These settings allowed us to systematically investigate the influence of different amplitude, frequency, and phase parameters on the predictive capabilities of LLMs.

E Metric Details

We use MAE (Mean Absolute Error) and MSE (Mean Squared Error) as measurement metrics, and their formula details are as follows:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n} \tag{9}$$

$$MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n} \tag{10}$$

F Frequency Decomposition on Monash.

We present the comparison results between Freq-Decomp and LLMTime on the Monash dataset in the Table 8. It can be seen that our method Freq-Decomp demonstrates a significant advantage over the baseline method LLMTime on the Monash dataset.

G Zero-shot Forecasting with Text Prompts

G.1 Method

The analysis in the previous Sections demonstrates that LLMs possess a certain degree of zero-shot predictive capability for time series data. Considering that LLMs inherently excel at processing textual data, this section conducts cross-modal time series forecasting experiments by incorporating relevant text prompts. Specifically, given the time series $T = \langle t_1, t_2, \cdots, t_n \rangle$, this section adds corresponding text prompt information $P = \langle w_1, w_2, \cdots, w_n \rangle$ and concatenates both of the sequences $[P; T]$ before inputting them into the LLMs for zero-shot forecasting. Here, $[;]$ denotes the concatenation operation.

G.2 Results

The cross-modal time series forecasting results with text prompts are presented in Table 9. LLMTime refers to the baseline model without the inclusion of text prompt information P, while Cross-Modal represents the prediction performance when both text prompts and time series information $[P; T]$ are input together. This experiment was conducted on the Darts dataset. It is evident that the Cross-Modal approach consistently outperforms the baseline model LLMTime across all datasets when text prompt information is incorporated. This indicates that methods based on LLMs can effectively jointly represent time series and text information, enhancing cross-modal time series prediction performance through the utilization of textual information. The corresponding text prompts P are shown below.

- AirPassengers. This time-series data is characterized by a high degree of volatility and an overall upward trend, suggesting that this volatile upward trend is likely to continue in the future, and therefore short-term volatility and the possibility of long-term growth should be considered in the forecast.
- AusBeer. The trend of this set of time-series data shows a gradual rise then a peak and finally a fall.
- GasRateCO2. This is a highly volatile period of time-series data that exhibits a certain degree of cyclicality and randomness. Forecasts of future trends need to take into account a variety of factors such as historical trends, cyclical variations and possible external influences.
- MonthlyMilk. This time-series data shows a continuous upward trend, accompanied by frequent and varying fluctuations, presenting strong dynamics and complexity.

Table 8: Frequency Decomposition (Freq-Decomp) results on Monash.

Dataset(Monash)	tourism monthly		bitcoin		fred md		nn5 weekly	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
LLMTime	1.0571	1.7150	1.2866	2.4910	0.8447	1.1519	1.0022	2.1980
Freq-Decomp(LLaMA2)	**0.3933**	**1.0036**	**0.1243**	**0.1019**	**0.4337**	**0.9981**	**0.7551**	**0.9970**

Table 9: Zero-shot forecasting results with text prompts.

Dataset(Darts)	AusBeer		AirPassengers		GasRateCO2		HeartRate	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
LLMTime	0.9513	1.6420	0.9028	1.3850	1.2649	2.6860	1.2618	2.6131
Cross-Modal	**0.8158**	**1.1210**	**0.7574**	**0.7831**	**0.9619**	**1.5557**	**1.0732**	**1.9151**
Dataset(Darts)	**MonthlyMilk**		**Sunspots**		**Wine**		**Wooly**	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
LLMTime	1.1724	1.9495	1.1403	1.9716	1.0875	1.6939	**1.0561**	1.7531
Cross-Modal	**1.0050**	**1.7312**	**0.9335**	**1.3555**	**0.7582**	**0.8402**	1.0858	**1.6711**

Table 10: Samples were randomly generated for each function based on different **Basic** function types.

Genre		Vicuna		Qwen	
		MAE ↓	MSE ↓	MAE ↓	MSE ↓
Sine	Samples	9.0164e-04	9.5720e-07	9.0164e-04	9.5720e-07
		6.2864e-04	6.3579e-07	6.2864e-04	6.3579e-07
		9.2959e-04	1.2074e-06	9.2959e-04	1.2074e-06
		6.3207e-04	6.3277e-07	6.3207e-04	6.3277e-07
	Mean(Std)	**7.7298e-04(1.4298e-04)**	**8.5829e-07(2.4085e-07)**	**7.7298e-04(1.4298e-04)**	**8.5829e-07(2.4085e-07)**
Cosine	Samples	1.0383e-03	1.2949e-06	9.3550e-04	1.0040e-06
		9.6355e-04	1.2982e-06	8.4973e-04	1.0686e-06
		8.9707e-05	1.2365e-07	3.3118e-04	4.5646e-07
		9.2881e-04	1.0349e-06	9.2881e-04	1.0349e-06
	Mean(Std)	**7.5509e-04(3.8619e-04)**	**9.3791e-07(4.8210e-07)**	**7.6130e-04(2.5061e-04)**	**8.9099e-07(2.5191e-07)**
Absolue	Samples	8.0035e-01	2.0240e+00	8.3408e-01	2.1060e+00
		7.9831e-01	2.0189e+00	8.3408e-01	2.1060e+00
		8.0097e-01	2.0259e+00	8.3408e-01	2.1060e+00
		8.0081e-01	2.0261e+00	8.3408e-01	2.1060e+00
	Mean(Std)	**8.0011e-01(1.0639e-03)**	**2.0237e+00(2.9038e-03)**	**8.3408e-01(0.0000e+00)**	**2.1060e+00(0.0000e+00)**
Linear	Samples	7.9990e-01	2.0235e+00	8.3391e-01	2.1052e+00
		8.0004e-01	2.0234e+00	8.3382e-01	2.1049e+00
		1.9788e-01	6.0302e-02	1.9745e-01	6.0487e-02
		7.9969e-01	2.0227e+00	8.3408e-01	2.1060e+00
	Mean(Std)	**6.4938e-01(2.6067e-01)**	**1.5325e+00(8.4996e-01)**	**6.7482e-01(2.7561e-01)**	**1.5941e+00(8.8546e-01)**
Logarithm	Samples	2.2265e-02	6.8289e-04	2.1808e-02	6.8705e-04
		3.4981e-02	1.5917e-03	2.5749e-02	9.0281e-04
		3.6029e-02	1.6956e-03	1.1935e-02	1.9467e-04
		3.4981e-02	1.5917e-03	3.4981e-02	1.5917e-03
	Mean(Std)	**3.2064e-02(5.6736e-03)**	**1.3905e-03(4.1072e-04)**	**2.3618e-02(8.2678e-03)**	**8.4406e-04(5.0219e-04)**
Polynomial	Samples	3.9087e-01	8.0511e-01	4.4991e-01	1.0453e+00
		6.3900e-01	5.4080e-01	8.1822e-01	8.8458e-01
		6.5851e-01	5.8106e-01	6.6118e-01	5.8204e-01
		5.4461e-01	1.3841e+00	6.0543e-01	1.5586e+00
	Mean(Std)	**5.5825e-01(1.0580e-01)**	**8.2777e-01(3.3661e-01)**	**6.3369e-01(1.3170e-01)**	**1.0176e+00(3.5386e-01)**
Reciprocal	Samples	1.0940e+00	2.6786e+00	1.0940e+00	2.6786e+00
		2.3126e-01	7.6730e-02	2.0173e-01	5.8156e-02
		1.0940e+00	2.6786e+00	1.0940e+00	2.6786e+00
		1.0940e+00	2.6786e+00	1.0940e+00	2.6786e+00
	Mean(Std)	**8.7832e-01(3.7358e-01)**	**2.0281e+00(1.1266e+00)**	**8.7093e-01(3.8636e-01)**	**2.0235e+00(1.1347e+00)**
ReLU	Samples	2.0259e-01	6.1406e-02	2.0396e-01	6.4134e-02
		2.0443e-01	6.3682e-02	1.9960e-01	6.1490e-02
		7.9998e-01	2.0233e+00	8.3408e-01	2.1060e+00
		8.0001e-01	2.0240e+00	8.3408e-01	2.1060e+00
	Mean(Std)	**5.0175e-01(2.9824e-01)**	**1.0431e+00(9.8055e-01)**	**5.1793e-01(3.1615e-01)**	**1.0844e+00(1.0216e+00)**

- Sunspots. This time-series data is characterized by a combination of high-frequency oscillations and a slow upward trend over a long period of time, as evidenced by sharp fluctuations between spikes and troughs and an overall gradual increase in peaks over time.

- Wine. This time-series data exhibits a characteristic of high-frequency oscillations superimposed on a long-term uptrend, where each spike and trough corresponds to a large change over a short period of time, while overall, the curve shows a progressively higher pattern over time.

- Wooly. The trend of this set of time series data shows frequent upward and downward fluctuations, and on the whole shows an upward and then downward trend, with certain cyclical characteristics.

- HeartRate. The trend of this set of time-series data is characterized by a random fluctuation

H Probing Results under Time-LLM Settings

Currently, research on time series prediction using LLMs can be categorized into two main approaches. The first approach retains the original LLMs and leverages their intrinsic embedding layers to encode time series information into the language space. The second approach modifies the LLMs (e.g., Time-LLM(Jin et al., 2023), Onefitall(Zhou et al., 2023)) by bypassing the LLMs' embedding layers for time series encoding and instead introduces additional temporal embedding layers and prediction head layers. The latter approach requires sufficient data to align the newly introduced layers with the LLMs' hidden space. As shown in our comparison with other state-of-the-art LLM-based forecasting models, the second approach is less suitable for zero-shot scenarios. Among this second category, Time-LLM is a representative work. It enables prediction using the LLaMA2 model, and thus we analyze the performance of the LLaMA2 model under the Time-LLM setup using the three probing experiments proposed in our work.

H.1 Experimental Procedure

Alignment of new layers. We first trained the newly introduced layers in Time-LLM to align their representational space with that of the LLMs. The ETTH1 dataset was used for this purpose, and the LLaMA2 model, consistent with that used in our work, was employed. Importantly, the LLaMA2 parameters were not adjusted; only the parameters of the newly added layers (e.g., reprogramming layer, temporal embedding layer, output mapping layer in Time-LLM) were fine-tuned.

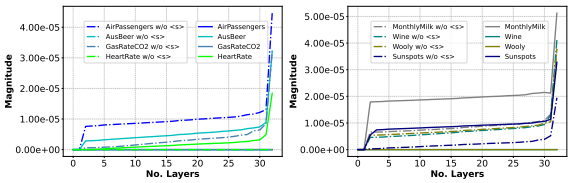


Figure 10: The variation in the average token impact values across different layers based on LLaMA2-7B.

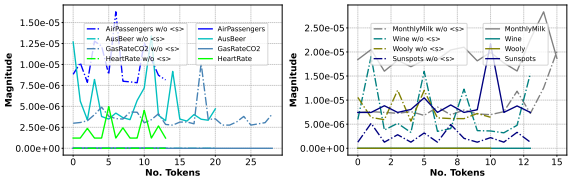


Figure 11: The variation in the average token impact values across token series (time series) based on LLaMA2-7B.

Probing experiments. After aligning the new layers, the model was initialized with the fine-tuned parameters, with LLaMA2 continuing to use its pre-trained parameters. Under this setup, the three proposed probing experiments were conducted on LLaMA2, with all model parameters frozen.

H.2 Results

H.2.1 Token Perturbation Probing

The results of Token Perturbation Probing are exhibited in Figure 10 and 11.

Layer-wise Analysis. The initial layers exhibit minimal mutual influence, consistent with the results observed in our work, where LLMs were directly used for zero-shot time series prediction through in-context learning. However, since Time-LLM does not generate predictions but instead uses an external trainable prediction head (output mapping layer in Time-LLM), the trend in the final layer is opposite to that observed in our work and newly uploaded results. This highlights the differences between generating predictions using LLMs and using an external prediction head.

Sequence-wise Analysis. Our work (using LLMs with in-context learning to generate predic-

tions) shows a tendency to focus on the initial tokens of the input sequence, especially when special tokens (e.g., <s> token) are present. In contrast, Time-LLM, which uses an external temporal embedding layer and prediction head, shows a different trend along the input sequence. LLMs do not focus primarily on the initial tokens but rather on identifying relevant tokens across the entire sequence. This is logical, as Time-LLM passes the entire sequence's encoded representation to the external prediction head, which requires the involvement of all tokens in the sequence, consistent with our analysis.

In summary, Token Perturbation Probing under the Time-LLM setup, when applied to LLaMA2-7b, reveals differences compared to direct generation-based prediction using LLMs. However, these differences are consistent with the characteristics of the Time-LLM setup, indicating that Token Perturbation Probing applies not only to generation-based time series prediction with LLMs but also to scenarios where additional temporal embedding layers and prediction heads are introduced.

H.2.2 Linear Probing

The results of Linear Probing are exhibited in Figure 12.

Layer-wise Variation. Due to differences in prediction methods (generation vs. external prediction head) and encoding approaches (using LLMs' native embeddings vs. external temporal embedding layers), the results of Linear Probing differ slightly. Compared to the results in our work, Linear Probing under the Time-LLM setup shows more pronounced variations along the layers of LLMs. We speculate that this may be due to the different encoding and prediction methods employed in Time-LLM. The patch operation used in Time-LLM results in a shorter encoded sequence than the original time series, making it easier for LLMs to operate on the entire sequence, leading to more pronounced variations.

Prediction Preparation. Direct generation-based prediction with LLMs requires the model to prepare for generation in the final layers, leading to a relatively stable phase in these layers. In contrast, under the Time-LLM setup, where an external prediction head is used, the final layers do not need to prepare for generation, resulting in no obvious stable phase.

In summary, Linear Probing can also detect the differences under the Time-LLM setup compared to direct prediction using LLMs, which aligns with the encoding and prediction characteristics of Time-LLM.

H.2.3 Vocabulary Mapping Probing

The results of Linear Probing are exhibited in Figure 13.

Special Token Mapping. Under the Time-LLM setup, time series data undergoes a patch operation and is fed into LLMs via an external temporal encoder. Consequently, unlike directly inputting time series numerical sequences into LLMs, Vocabulary Mapping does not map to numerical characters in the vocabulary. In this experiment, we analyzed the distribution of mappings to special tokens (e.g., <s> token) versus non-special tokens.

Layer-wise Distribution of Special Tokens. It is evident that when directly using LLMs for generation-based prediction, special tokens are more frequently mapped in the later layers of LLMs. In contrast, under the Time-LLM setup, special tokens are mainly mapped in the earlier layers of LLMs. This is likely due to the different prediction methods: direct generation-based prediction with LLMs requires special tokens in the later layers to assist in the generation, whereas, under the Time-LLM setup, LLMs' task is to encode time series information for the external prediction head, which reduces the need to focus on special tokens in the later layers.

Overall, due to the differences in prediction methods and the modifications made to LLMs for encoding and predicting time series, the results of these three probing experiments under the Time-LLM setup differ from those observed with direct generation-based prediction using LLMs. However, this also demonstrates that the probing methods proposed in this paper are applicable to more complex modifications of LLMs, yielding probing results consistent with the characteristics of the respective setups.

I Detailed Vocabulary Mapping Results

I.1 Numeric Mapping

We present nine examples from the Synthetic datasets using the numeric mapping method. As Figure 14 shows, almost all time series tokens are mapped into numeric tokens in the first 3 layers of LLM. As the number of layers increases, there is

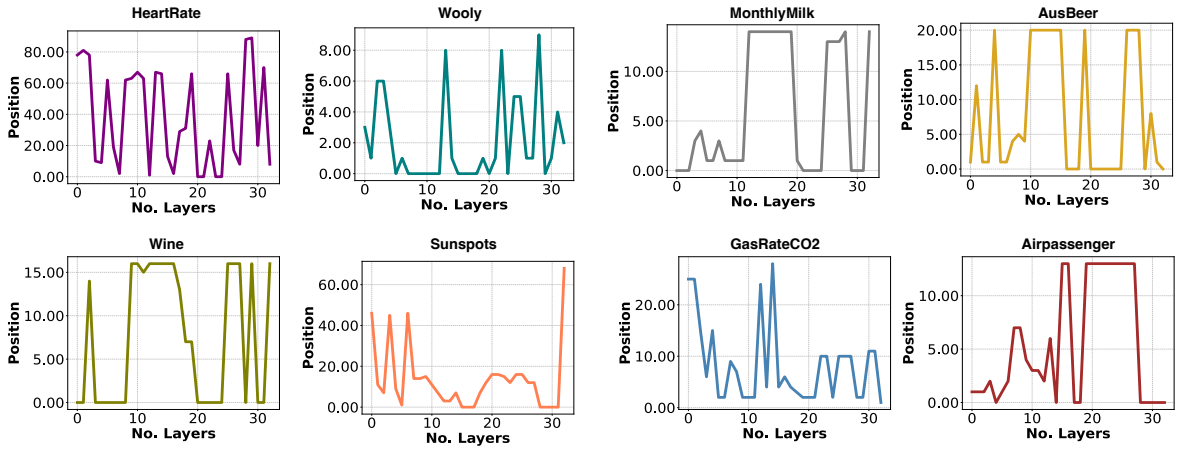


Figure 12: Linear probing results under Time-LLM settings.

a decreasing trend in the proportion of numeric tokens. However, when the length of the time series increases, there is an increase in the proportion of tokens that are mapped to numeric values.

I.2 <s> Mapping

In addition, we further analyze the representation of time series tokens across layers by mapping them to the <s> token using the vocabulary mapping probe. As depicted in Figure 15, the tokens mapped to <s> are predominantly distributed in layers 20-30 for the Synthetic datasets. Similarly, these mappings primarily occur in the initial 60 portions of the time series. These findings reinforce the notion that these layers capture global patterns and leverage information from the early segments of the time series. The consistent observation of such mappings across different datasets suggests the importance of integrating global information for subsequent time series prediction tasks.

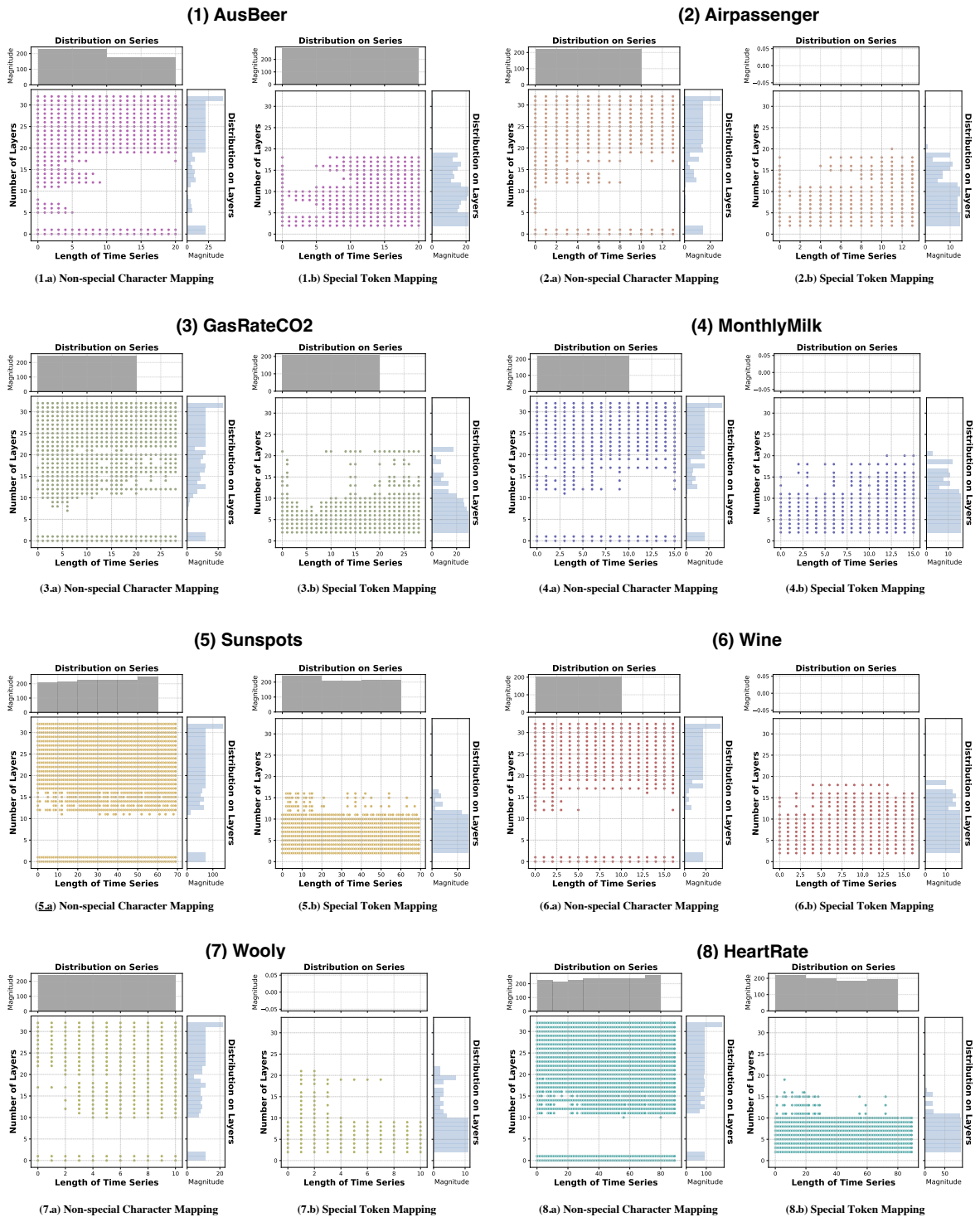
J Additional Linear Probe Results

As show in Figure 16, there is fluction in the positions in the initial layers, but they mostly stabilize after the 10th layer, except for the Linear+Cosine dataset, which stabilizes after 20th layer. This finding suggests that, for most datasets, the initial layers of the LLMs go through an exploratory phase where the positions of the time series tokens with the smallest MSE may vary. However, as the number of layers increases, these positions gradually stabilize, indicating that the model has learned the stable patterns present in the datasets. This holds true for most datasets, except for the linear+cosine dataset, where stable positions with the smallest MSE are obtained at deeper layers.

K Additional Token Perturbation Probe

Figure 17 and Figure 18 depict 3D visualizations of different token impact values on the AirPassengers dataset, specifically focusing on the first 19 tokens. The distinction between the two figures lies in the presence or absence of the token <s>. In Figure 17, which includes the <s> token, it was observed that almost all tokens across all layers exhibited a significantly high level of attention towards the first token <s>. Furthermore, as the token position increased (indicating attention towards more tokens), the attention values towards the first token <s> gradually decreased. On the other hand, Figure 18, which excludes the <s> token, showed that although the attention towards the first token <s> remained high for all tokens, the attention towards other tokens was not as low as in Figure 17. In other words, the difference in attention towards preceding tokens was not as pronounced as seen in Figure 17. Additionally, there was an overall decreasing trend in attention values across tokens when the <s> token was absent.

Similarly, Figure 19 and Figure 20 both depict 3D visualizations of different layer impact values on the AirPassengers dataset, focusing on the first 31 tokens. The distinction between the two lies in Figure 19 including the <s> token, while Figure 20 excludes it. The observations in these figures align with the previous ones, indicating that almost all tokens exhibit significantly high attention towards the first token <s>. Furthermore, in Figure 20 (without <s> token), the difference in attention towards preceding tokens is not as substantial as in Figure 19, while overall attention values show a decreasing trend.



(1.a) Non-special Character Mapping (1.b) Special Token Mapping (2.a) Non-special Character Mapping (2.b) Special Token Mapping

(3.a) Non-special Character Mapping (3.b) Special Token Mapping (4.a) Non-special Character Mapping (4.b) Special Token Mapping

(5.a) Non-special Character Mapping (5.b) Special Token Mapping (6.a) Non-special Character Mapping (6.b) Special Token Mapping

(7.a) Non-special Character Mapping (7.b) Special Token Mapping (8.a) Non-special Character Mapping (8.b) Special Token Mapping

Figure 13: Vocabulary mapping probing results under Time-LLM settings.

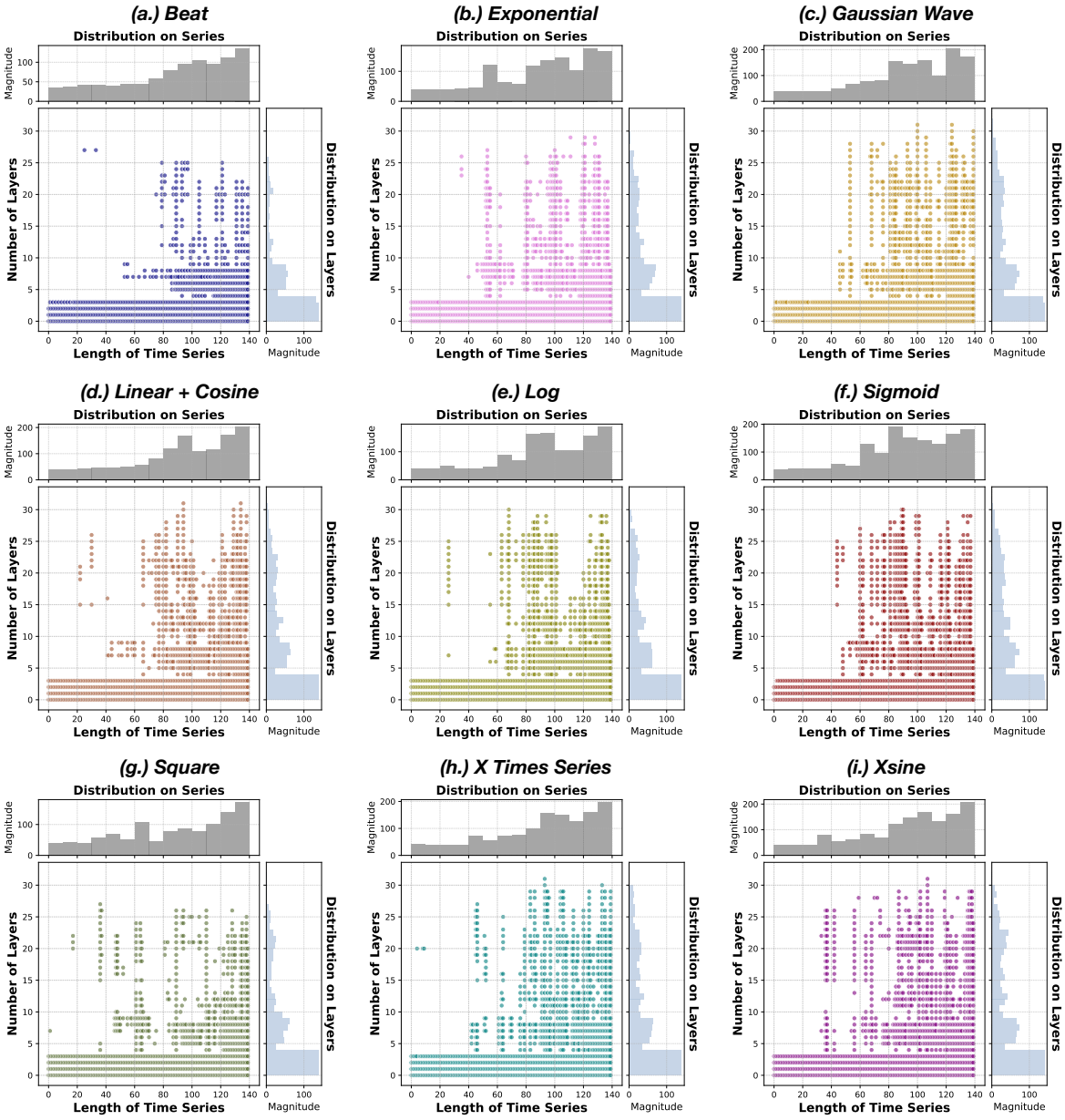


Figure 14: Distribution of layers and series mapped to numbers with marginal distribution statistics on Synthetic datasets.

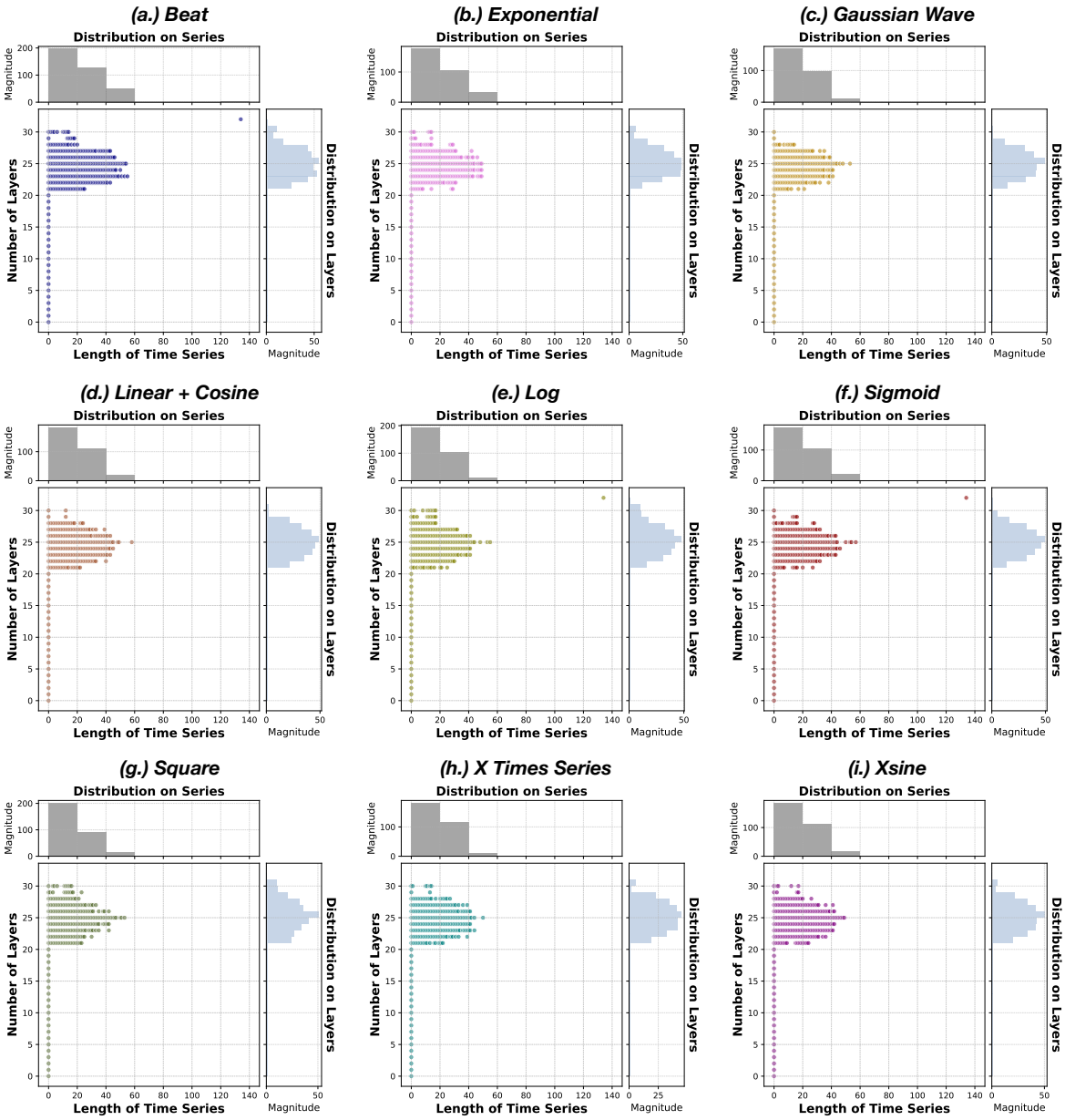


Figure 15: Distribution of layers and series mapped to <*s*> with marginal distribution statistics on Synthetic datasets.

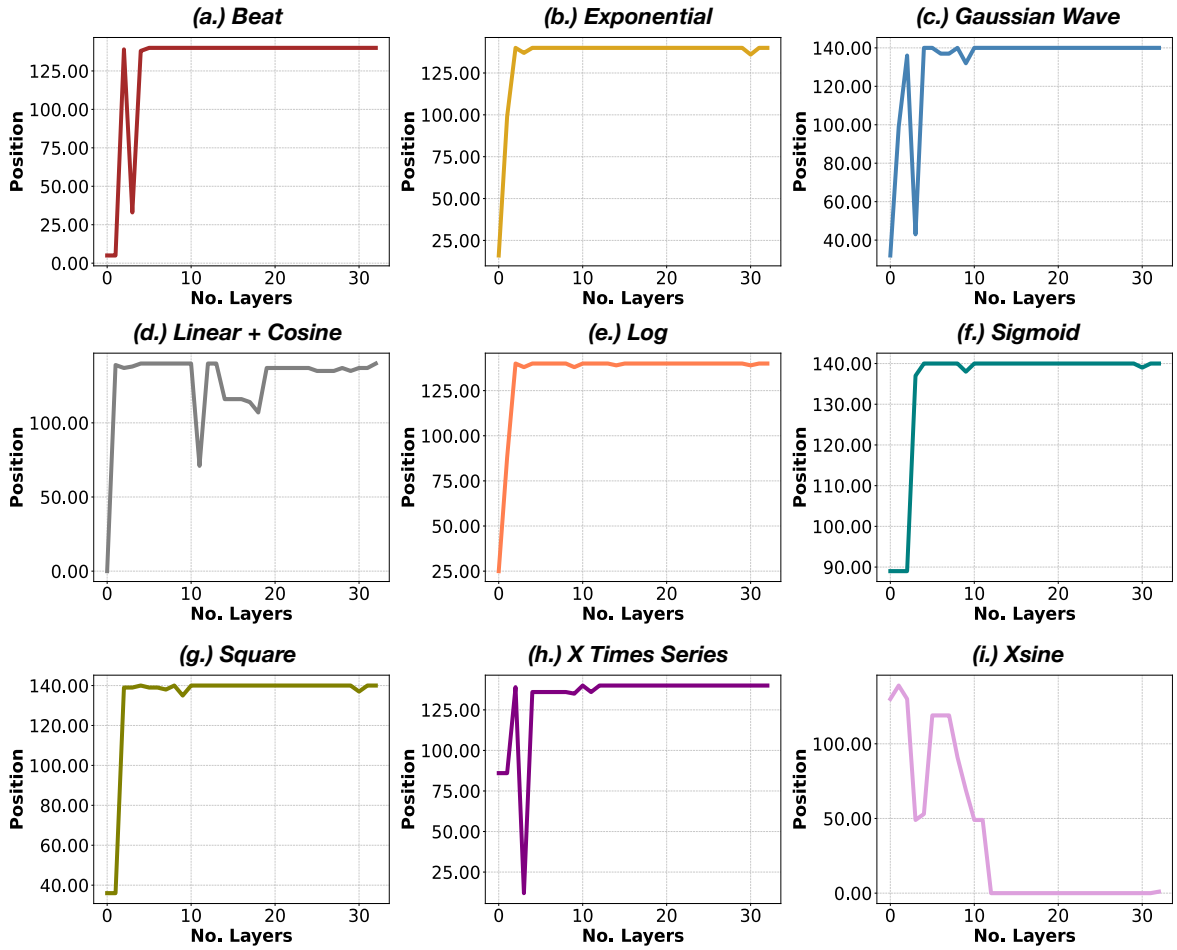


Figure 16: The variation in the time series token positions with the smallest MSE across different layers, as obtained through linear probing on Synthetic datasets.

This finding suggests that the initial token <*s*> holds considerable importance and captures the attention of the model across multiple layers. However, as the model attends to more tokens and the token position increases, the relative importance of the first token <*s*> diminishes. This observation highlights the evolving relationship between tokens and the decreasing emphasis on the initial token as the model processes a wider context of tokens.

L Visualization

We also provide visualization of the prediction results of Freq-Decomp and LLMTime on the Darts dataset. These results, based on different LLMs, are displayed in Figures 22, 23, and 24.

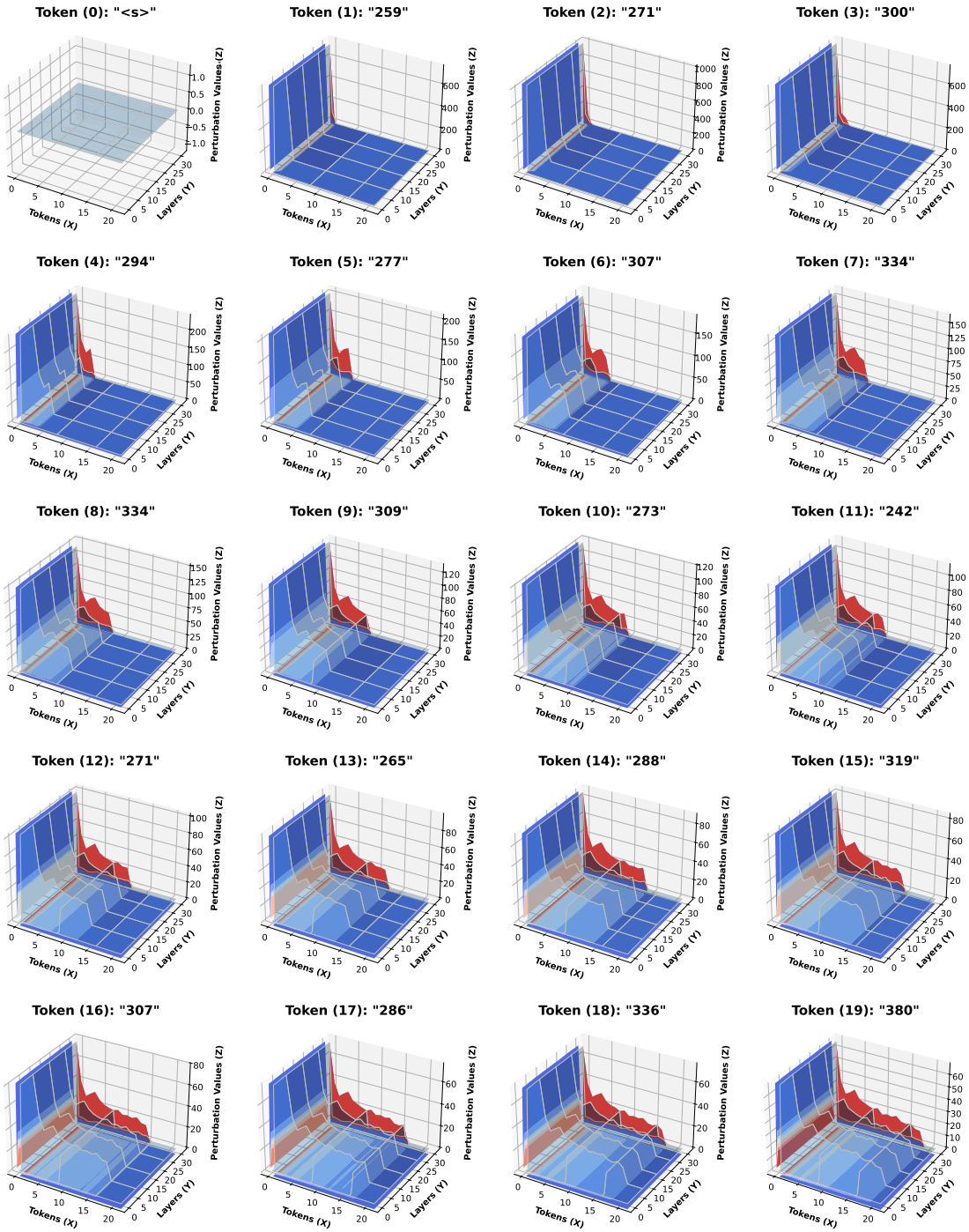


Figure 17: 3D visualizations of different token impact values on AirPassengers datasets.

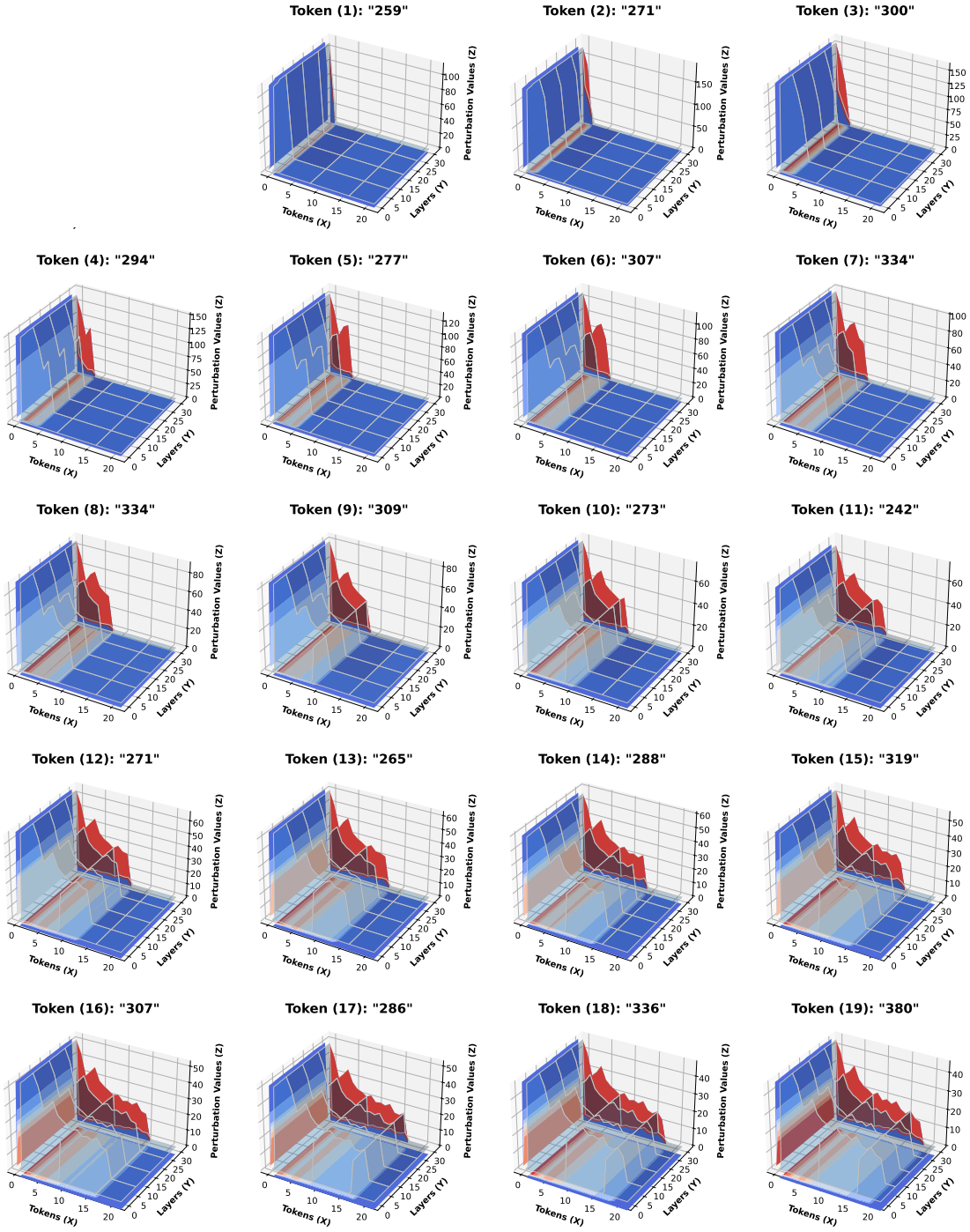


Figure 18: 3D visualizations of different token impact values on AirPassengers datasets without <*s*>.

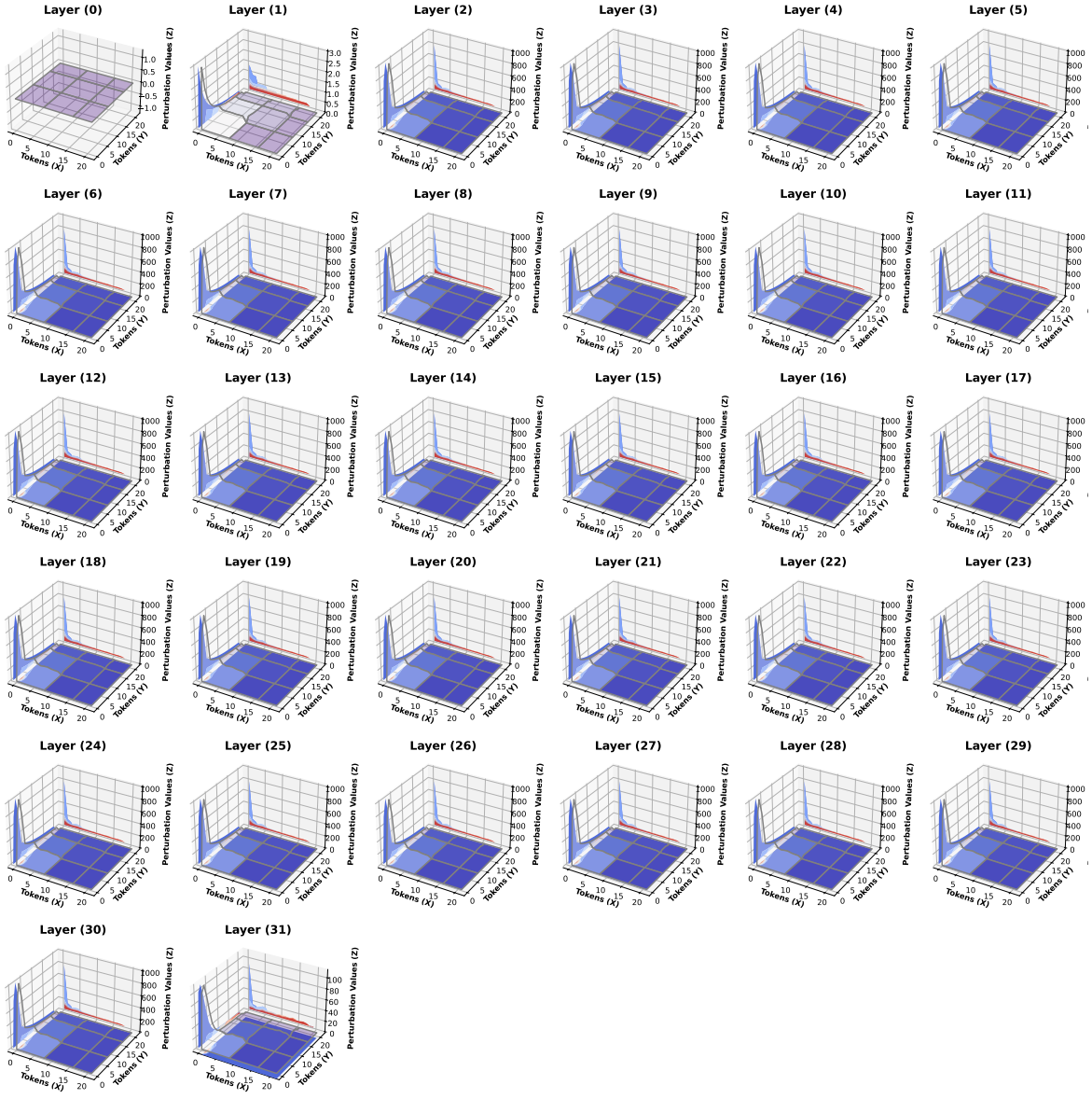


Figure 19: 3D visualizations of different Layer impact values on AirPassengers datasets.

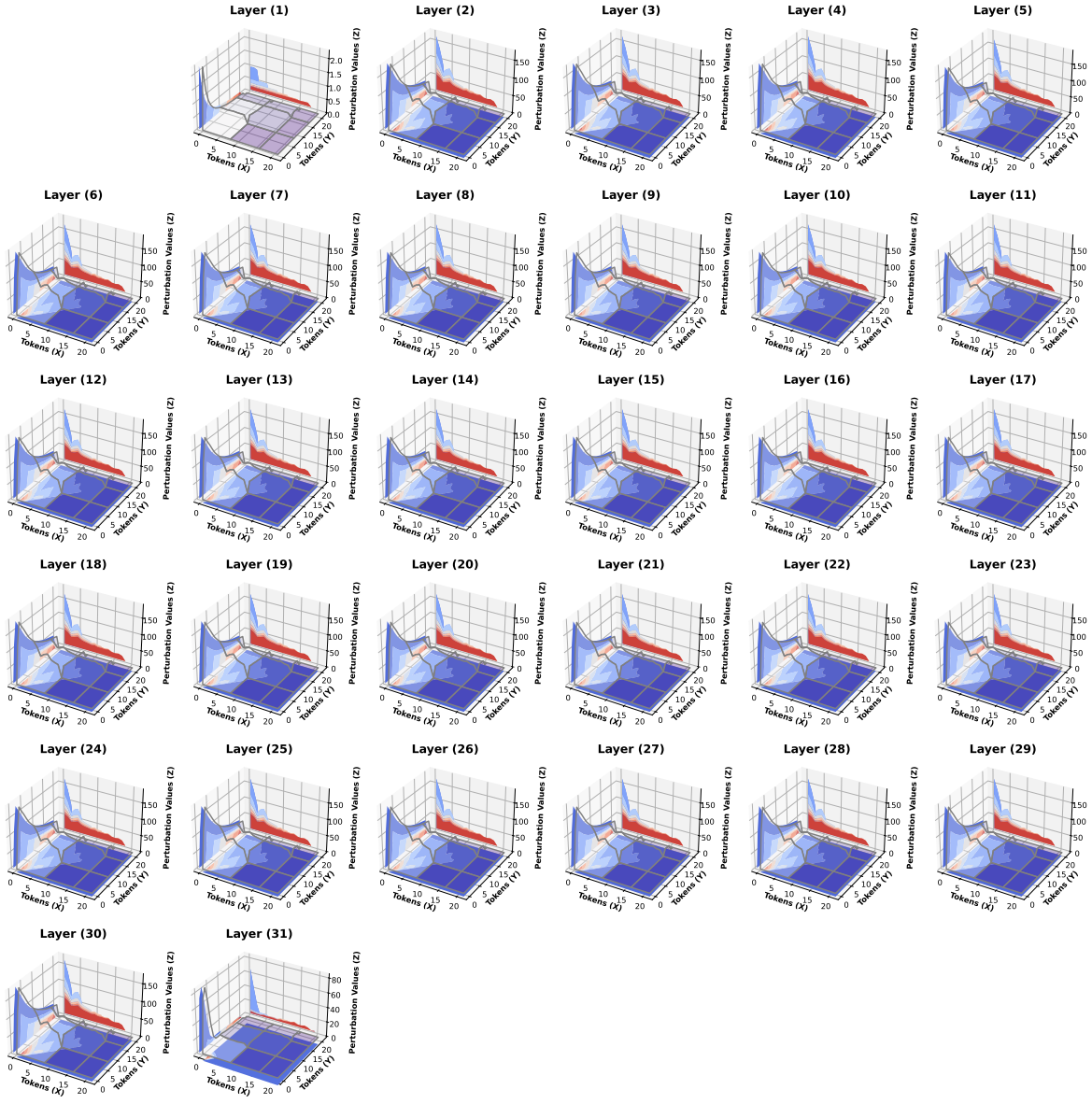


Figure 20: 3D visualizations of different Layer impact values on AirPassengers datasets without <*s*>.

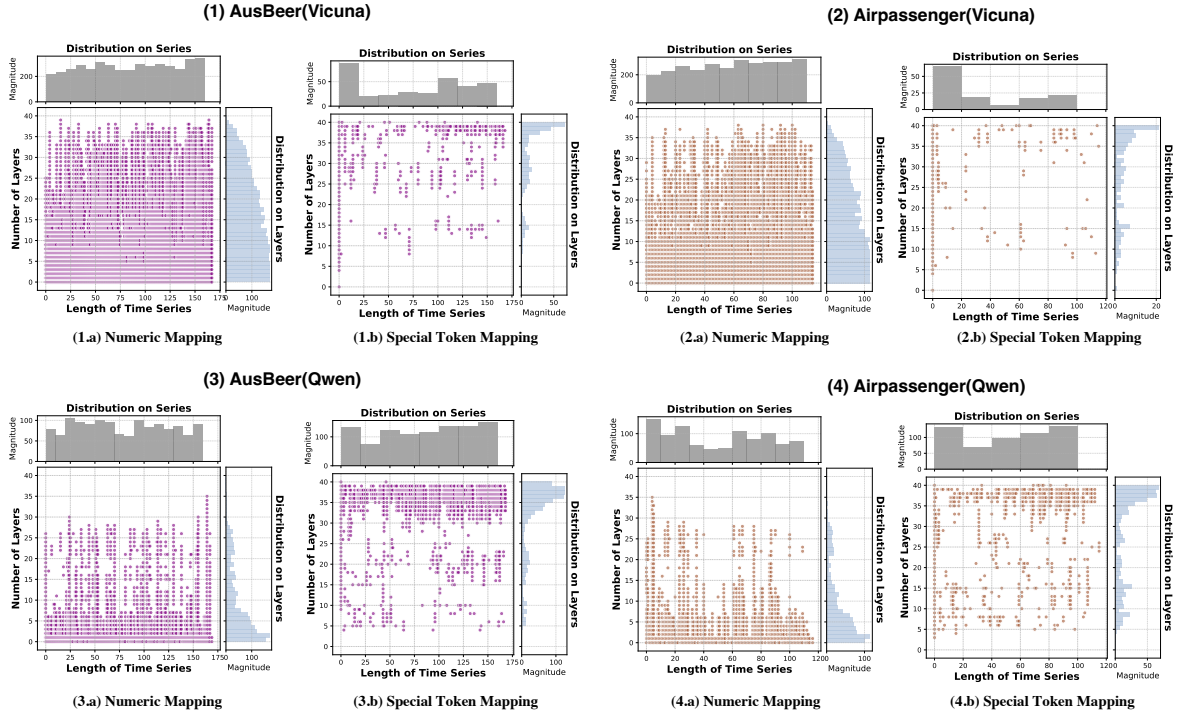


Figure 21: The results of Vocabulary Mapping Probing based on Vicuna and Qwen.

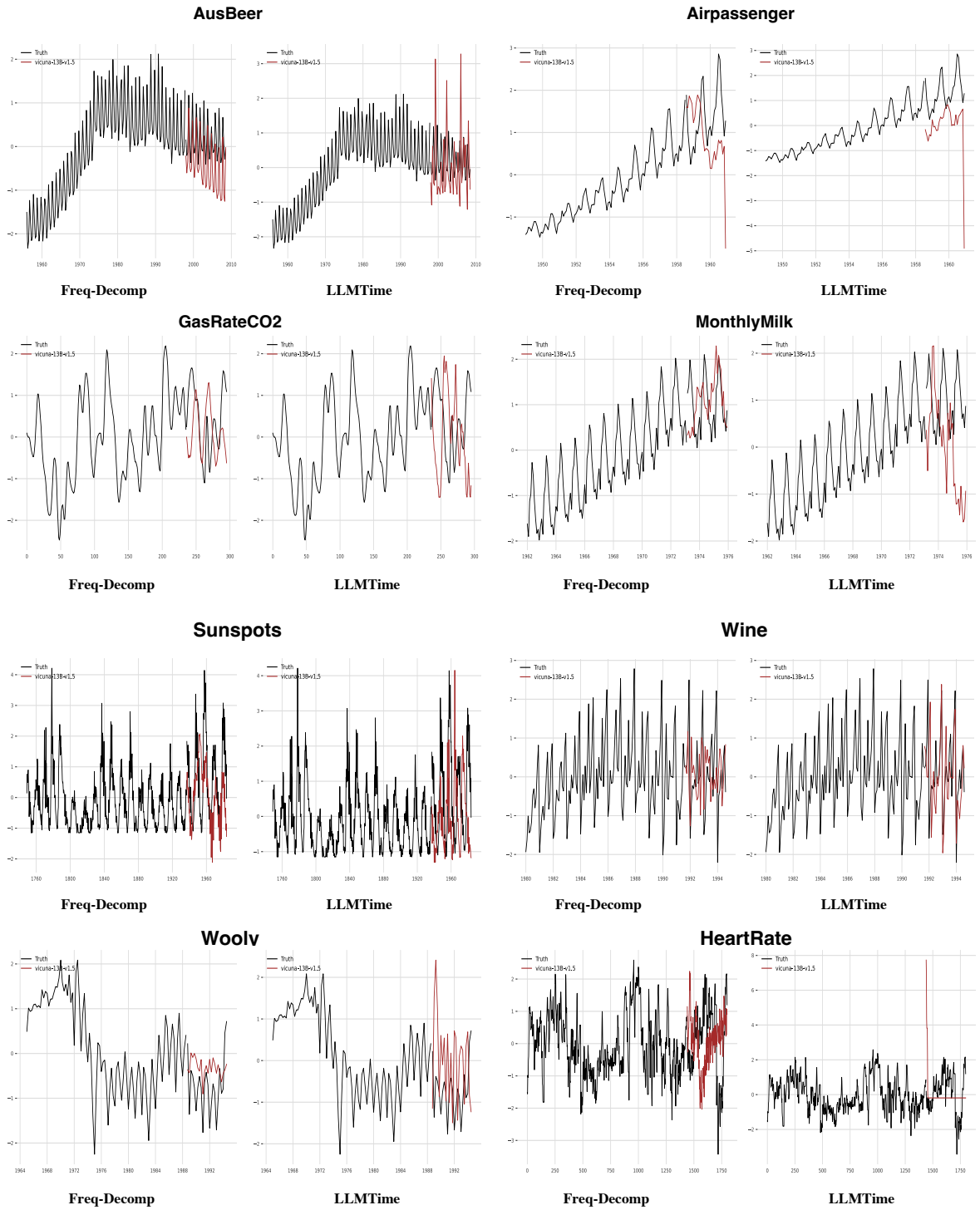


Figure 22: Visualization of prediction results based on Vicuna.

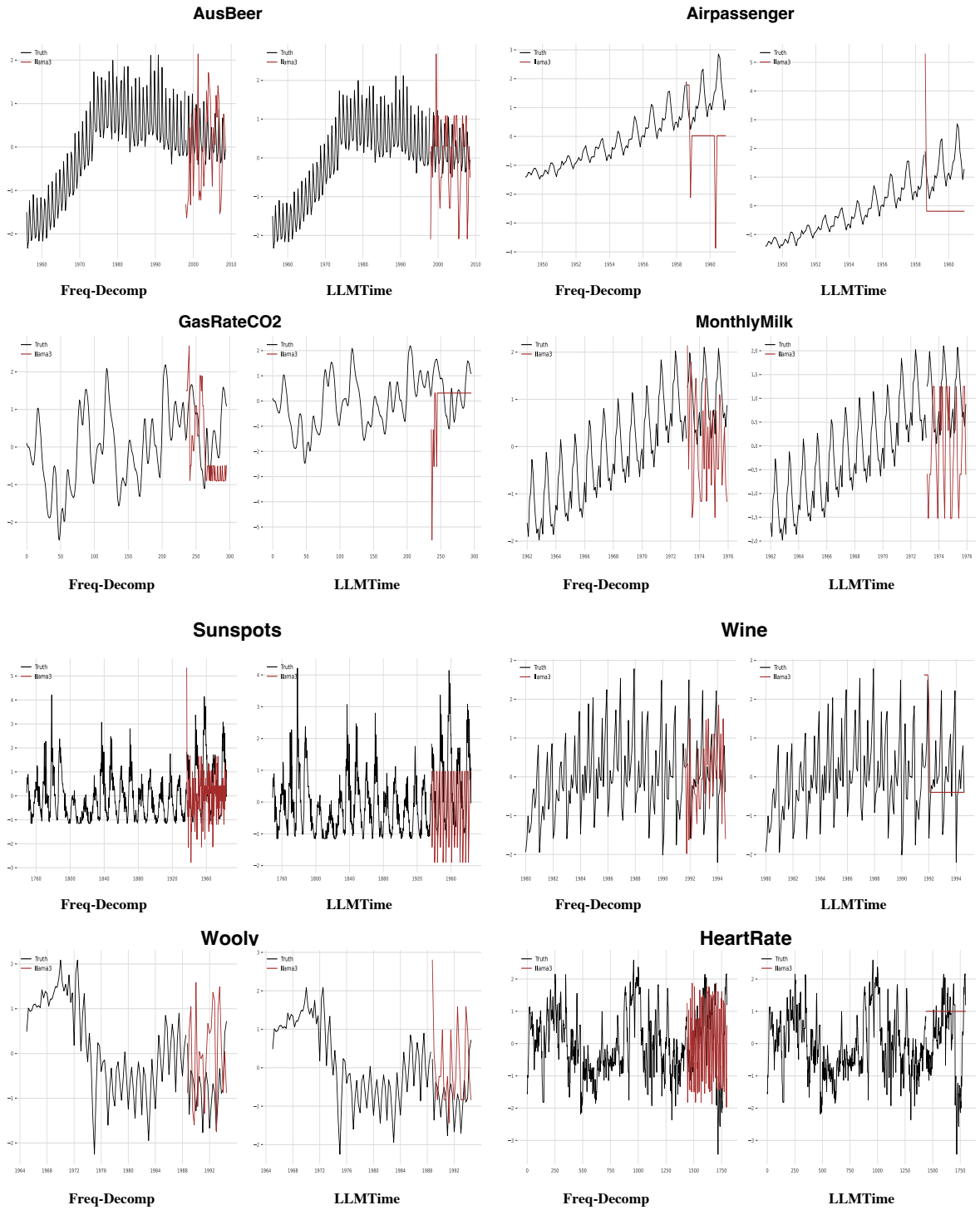


Figure 23: Visualization of prediction results based on LLaMA3.

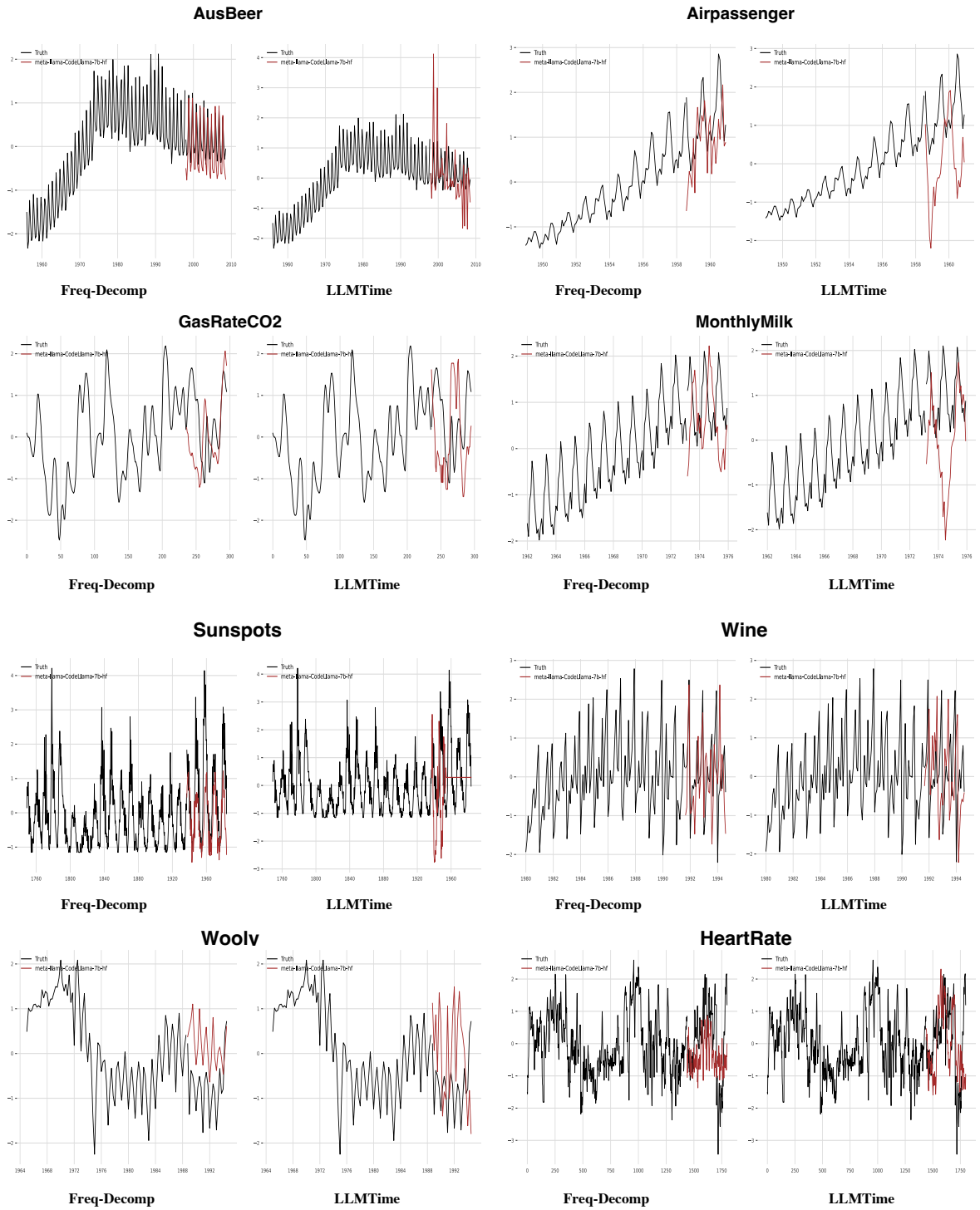


Figure 24: Visualization of prediction results based on CodeLlama.