DISSECTING LLMS FOR TIME SERIES FORECASTING: WHEN THEY HELP AND WHERE THEY STRUGGLE?

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) have been introduced to time series forecasting (TSF) to incorporate contextual knowledge beyond numerical signals. However, existing studies question whether LLMs provide genuine benefits, often reporting comparable performance without LLMs. We show that such conclusions stem from limited evaluation settings and do not hold at scale. We conduct a large-scale study of LLM-based TSF (LLM4TSF) across 8 billion observations, 17 forecasting scenarios, 4 horizons, multiple alignment strategies, and both in-domain and out-of-domain settings. Our results demonstrate that *LLM4TS indeed improves forecasting performance*, with especially large gains in cross-domain generalization. Pre-alignment outperforming post-alignment in over 90% of tasks. Both pretrained knowledge and model architecture of LLMs contribute and play complementary roles: pretraining is critical under distribution shifts, while architecture excels at modeling complex temporal dynamics. Moreover, under large-scale mixed distributions, a fully intact LLM becomes indispensable, as confirmed by token-level routing analysis and prompt-based improvements. Overall, our findings overturn prior negative assessments, establish clear conditions under which LLMs are not only useful, and provide guidance for effective model design.

1 INTRODUCTION

Time series forecasting (TSF) is a fundamental learning task with broad applications across many domains (Liu & Wang, 2024; Kim et al., 2025; Kong et al., 2025). Despite decades of research on statistical and machine learning approaches (De Gooijer & Hyndman, 2006; Masini et al., 2023; Yunita et al., 2025), accurate TSF remains challenging due to the coexistence of long-term trends, periodic patterns, abrupt changes, and stochastic noise. Inspired by the success of Transformers in fields such as speech, vision, and video understanding (Dong et al., 2018; Liu et al., 2021; 2022b; Cao et al., 2022), prior work has introduced Transformer-based architectures to TSF (Tong et al., 2023; Zhang et al., 2024a; Liang et al., 2024; Li et al., 2025). By leveraging attention mechanisms, these models enable more flexible modeling of temporal dependencies.

However, most Transformer-based TSF models are trained from scratch on uni-modal numerical TS. Such representations are inherently abstract and lack explicit encoding of real-world context or environmental factors that often underlie temporal observations (Liu et al., 2024a; Huang et al., 2025; Lei et al., 2025). This limitation has motivated recent explorations of LLM-based time series forecasting (LLM4TSF), aiming to leverage the rich world knowledge embedded in LLMs pretrained on large-scale text corpora (Hu et al., 2025c;b; Wolff et al., 2025). To reduce the modality gap between numerical TS and language representations, existing approaches typically follow two alignment paradigms: *Pre-alignment* methods map TS into language-compatible representations via cross-attention with word embeddings before feeding them into an LLM (Hu et al., 2025c; Liu et al., 2025a). In contrast, *post-alignment* methods jointly fine-tune TS encoders and LLMs through supervised learning, adapting both components simultaneously (Meunier et al., 2025; Liu et al., 2025c).

Despite this progress, a fundamental question remains unresolved: *are LLMs truly indispensable for TSF?* Many studies raise doubts, arguing that existing alignment strategies may induce only pseudo-alignment (Zheng et al., 2025; 2026), or observing that removing the LLM causes little to no performance degradation (Tan et al., 2024; Zheng et al., 2025; Zhang et al., 2025). These findings

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

have sparked an ongoing debate about whether LLMs contribute genuine modeling capability, or merely act as architectural or parameter-level augmentations.

We argue that existing analyses are insufficient to answer this question conclusively. Prior studies are typically conducted on small-scale datasets, rely on only the shallow layers of LLMs (Tan et al., 2024; Zheng et al., 2025), focus primarily on in-domain evaluation (Zhang et al., 2025), and rarely probe the underlying mechanisms responsible for performance differences. Importantly, the core strength of LLMs lies not in their architecture alone, but in their pretrained world knowledge, instruction-following ability, and capacity for multi-task generalization (Brown et al., 2020). Evaluating LLM4TSF under single-task, in-domain settings with partially utilized LLM parameters fails to reflect these capabilities and may lead to misleading conclusions. To address these limitations, we conduct a large-scale study of LLM4TSF across diverse settings. Our evaluation spans 8 billion observations, 17 scenarios, both in-domain and out-of-domain distributions, and four forecasting horizons. We examine representative pre-alignment and post-alignment strategies, and disentangle the roles of pretrained knowledge, model architecture, and alignment design in LLM4TSF.

Our analysis provides clear evidence that *LLM4TSF indeed improves forecasting performance*. First, alignment strategy plays a decisive role: *pre-alignment methods outperform post-alignment* approaches in over 90% of tasks overall. Second, performance gains arise from *a complementary interaction between pretrained knowledge and architectural capacity*. Pretraining proves particularly valuable under distribution shifts and out-of-domain settings, while architectural components excel at capturing complex temporal dynamics. Third, *data diversity is critical*: models trained on multi-source TS consistently outperform single-dataset baselines in more than 70% of in-domain tasks and exhibit stronger cross-domain generalization than TS-specific models. Finally, *LLM4TSF models show clear preferences for certain statistical regimes*, performing especially well on data with frequent transitions and high variability. Further analysis under mixed-distribution and large-scale settings yields additional insights. Unlike low-data regimes where randomizing or removing the LLM has minimal impact (Tan et al., 2024), we find that a fully intact LLM becomes essential for robust forecasting at scale, and partial fine-tuning is no longer sufficient. Token routing analysis provides mechanistic evidence for this effect: the model's decision to route tokens through or around the LLM strongly correlates with forecasting errors, indicating adaptive utilization of LLM capabilities. Moreover, informative textual prompts consistently improve performance, underscoring the importance of semantic guidance beyond simply increasing model size. At the same time, our study identifies key bottlenecks of LLMs in TSF. LLM4TSF does not automatically benefit from larger LLMs without careful alignment, and performance remains sensitive to data distribution, preventing uniformly strong results across all scenarios. In summary, our contributions are as follows:

1. *A large-scale study* that provides the first comprehensive assessment of the benefits and limitations of LLM4TSF.
2. *A principled decomposition of performance gains*, clarifying the distinct and complementary benefits of pretrained knowledge and model architecture.
3. *A routing-based analysis* that links token-level path selection to macroscopic forecasting performance, offering concrete evidence of LLM effectiveness.
4. *Practical guidelines and capability boundaries for applying LLMs to TSF*, informing the design of future LLM-based forecasting systems.

2 PRELIMINARY

2.1 SINGLE & CROSS-DATASET LEARNING PARADIGM

TS data spanning diverse domains and exhibiting substantial statistical heterogeneity (Liu et al., 2024b; Chang et al., 2025). Ideally, a model should not only achieve strong performance on a single dataset, but also be capable of transferring knowledge across datasets (Cheng et al., 2024; Xiao et al., 2025). However, many existing LLM4TSF still adopt a ***single-dataset learning*** paradigm. For example, S^2IP (Pan et al., 2024), FSCA (Hu et al., 2025c), TransDF (Wang et al., 2025b), and CALF (Liu et al., 2025c) are typically trained and evaluated on individual datasets. Such settings are prone to overfitting and limit the potential advantages of LLMs. Inspired by domain-specific TS foundation models trained from scratch on large-scale data (Ansari et al., 2024; Ning et al., 2025;

Liu et al., 2025e), prior work such as UniTime (Liu et al., 2024b) adopts ***cross-dataset learning*** for LLM4TSF, enabling stable in-domain and out-of-domain generalization, while we further introduce fine-grained instructions to support instruction-driven task generalization (Zhou et al., 2023a).

2.2 Core components of LLM4TSF

Typical architecture of LLM4TSF models consist of three core components: ***TS encoder***, ***LLM backbone*** and ***TS decoder***. Both the TS encoder and decoder are implemented as lightweight MLPs, decoupling low-level numerical processing from high-level learning (Chen et al., 2025). The LLM backbone is initialized from a pre-trained LLM to inherit its semantic understanding capabilities.

(I) TS encoder. Given a TS $\mathbf{X}_{1:L} \in \mathbb{R}^{L\times d}$, channel-independent and normalization are applied to mitigate scale variations across variables (Kim et al., 2021). Subsequently, a patching operation is employed to divide each TS into a sequence of local patches (Nie et al., 2023). Specifically, we denote the patch length as P and the stride as S. The patching produces $\mathbf{x}_p \in \mathbb{R}^{P\times N}$, where N denotes the number of patches, defined as $N = \frac{L-P}{S} + 2$. The TS embeddings are then obtained as $\mathbf{X} = f_{\text{enc}}(\mathbf{x}_p)$, where $f_{\text{enc}}(\cdot)$ denotes the TS encoder. ***(II) LLM backbone.*** To activate the prior knowledge in the LLM, textual prompts are introduced to describe background information and task specifications. The text prompts are processed by the LLM tokenizer to obtain prompt embeddings. At the input layer of the LLM, the prompt embeddings are fed together with the TS embeddings. The resulting hidden states are used as integrated representations, expressed as $\mathbf{h} = f_{\text{LLM}}(\mathbf{Z}, \mathbf{X})$, where $\mathbf{Z}$ and $\mathbf{X}$ denote the prompt and TS embeddings. ***(III) TS decoder.*** A TS decoder maps the outputs $\mathbf{h}$ back to forecasts: $\hat{\mathbf{X}}_{L+1:L+H} = f_{\text{dec}}(\mathbf{h})$, where $f_{\text{dec}}(\cdot)$ denotes the TS decoder.

2.3 Alignment Strategy for LLMs on TSF

Applying LLMs to TSF involves bridging the modality gap between TS and text modalities. Existing methods mainly adopt one of two strategies to enable effective cross-modal interaction (Jin et al., 2023; Woo et al., 2024b; Liu et al., 2025c; Meunier et al., 2025; Hu et al., 2025c), namely ***pre-alignment*** and ***post-alignment***, as illustrated in Figure 1.

Pre-alignment. Pre-alignment aligns TS to the textual modality *before* input to the LLM, exploiting the semantic structure of pre-trained language embeddings while keeping the LLM frozen. Let $\mathbf{X} \in \mathbb{R}^{N\times M}$ denote the TS embeddings to be aligned, where N is the number of TS tokens and M is the embedding dimension. Let $\mathbf{D} \in \mathbb{R}^{|\mathcal{A}|\times M}$ denote the language embedding dictionary of the LLM, where $|\mathcal{A}|$ is the vocabulary size. Due to the large size of $\mathbf{D}$, directly aligning TS embeddings with the full dictionary is expensive. Therefore, principal component analysis (PCA) is applied to obtain a set of principal language embeddings: $\hat{\mathbf{D}} = \text{PCA}(\mathbf{D})$, where $\hat{\mathbf{D}} \in \mathbb{R}^{d\times M}$ and $d \ll |\mathcal{A}|$. Alignment is performed via attention, using TS embeddings as queries and the principal language embeddings as keys and values.

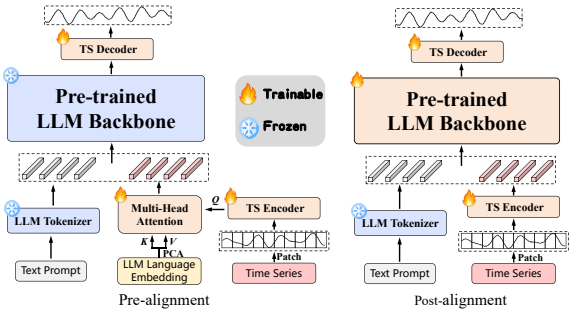

Figure 1: Two mainstream alignment strategies.

Post-alignment. Post-alignment perform modality alignment between TS and text within the representation space of the LLM, by jointly modeling TS embeddings and textual embeddings. In this paradigm, TS embeddings and prompt embeddings are fed into the LLM for cross-modal modeling. Let $\mathbf{X} \in \mathbb{R}^{N\times M}$ denote the TS embeddings and $\mathbf{Z} \in \mathbb{R}^{C\times M}$ denote the prompt embeddings, where C is the number of text tokens. The LLM produces integrated representations as: $\mathbf{h} = f_{\text{LLM}}(\mathbf{Z}, \mathbf{X})$, where $f_{\text{LLM}}(\cdot)$ denotes the forward mapping of the LLM. During training, the LLM parameters are updated using supervision from the TSF task, thereby enabling alignment in the latent space.

3 Benefits of Diverse TS Data in LLM4TSF

Although prior studies have examined LLM-based TSF, their evaluations are mostly conducted under single distribution. In addition, many approaches adapt LLMs using only shallow layers, restrict-

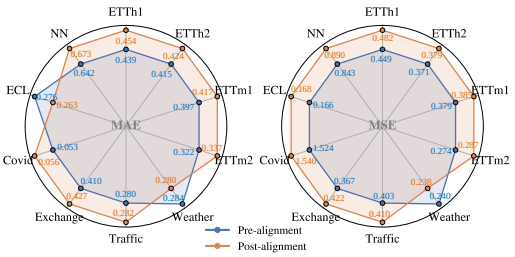

Figure 2: Comparison of LLM4TSF with pre- and post-alignment under a single-dataset.

Table 1: Out-of-domain performance (MSE, averaged over horizons $\{96, 192, 336, 720\}$). **Bold** and underlined indicate the 1st & 2nd best results.

Types	Zero-Shot Test					5% Few-Shot Test	
Models	Pre-align	Post-align	Chronos	UniTS	Moirai	UniTime	TimeLLM
Wind	1.015	**0.963**	1.422	1.358	1.236	1.358	1.321
Solar	0.228	0.274	0.434	0.871	0.936	**0.218**	0.577
AQShunyi	**0.612**	0.688	0.808	0.890	0.668	0.905	0.859
CzenLan	**0.274**	0.288	0.298	0.738	0.660	0.401	0.319
ZafNoo	0.547	0.583	0.550	0.668	**0.543**	0.803	0.594
NASDAQ	**0.735**	0.749	0.873	1.120	1.067	1.122	0.983
PEMS	0.256	0.291	0.686	1.303	**0.243**	0.419	0.416

ing the capacity of pretrained models and making results sensitive to overfitting or dataset-specific artifacts (Pan et al., 2024; Liu et al., 2025a). Consequently, the true capability of LLMs in TSF remains difficult to assess (Tan et al., 2024; Zheng et al., 2025). More importantly, pretrained LLMs are designed to learn transferable representations from diverse data. When evaluated on a single TS dataset, this potential may be underutilized. Motivated by this, we adopt cross-dataset learning with full-scale LLMs and compare its performance against single-dataset learning under both in-domain and out-of-domain settings.

3.1 Experimental Setup

Datasets. We conduct experiments on a collection of 62 real-world, publicly available datasets spanning over 10 domains. The entire dataset collection is denoted as $\mathcal{D}$ and is partitioned into two disjoint subsets, $\mathcal{D} = \mathcal{D}_A \cup \mathcal{D}_B$. The subset $\mathcal{D}_A$ contains 55 datasets and is used for model development, while $\mathcal{D}_B$ consists of the remaining 7 datasets, which are completely excluded from training and used solely for out-of-domain evaluation. For each dataset $\mathcal{D}_i \in \mathcal{D}_A$, we apply train–test splits, where the training split is used for optimization and the held-out test split is used to evaluate in-domain performance. Overall, the combined datasets comprise over 8B observations, providing a diverse testbed for studying in-domain and out-of-domain setting, referring to Appendix F.1.

Models. We consider both the ***pre-alignment*** and ***post-alignment*** strategies introduced in Sec. 2.3. Under each alignment strategy, models are trained following two learning paradigms described in Sec. 2.1, namely ***single-dataset learning*** and ***cross-dataset learning***. This results in two model variants, denoted as ***LLM4TSF (Pre-align)*** and ***LLM4TSF (Post-align)***. The core components of all models follow the architecture described in Sec. 2.2. Specifically, we adopt GPT-2 (Radford et al., 2019) as the LLM backbone. To preserve the full modeling capacity of the pretrained LLM, no layer truncation is applied. At the input stage, for the TS component, the look-back window length is set to $T = 512$. Non-overlapping patch-level sampling is applied with patch size $P = 32$ and stride $S = 32$. All statistical descriptors are computed solely from the 512-step look-back window, ensuring that no future information is leaked, referring to Appendix D.

Test Details. The forecasting horizon H is evaluated at $\{96, 192, 336, 720\}$, covering short-term and long-term forecasting scenarios. Generalization is assessed under two settings: ***in-domain*** and ***out-of-domain test***. The former is conducted on 10 datasets with held-out test splits, and the latter is performed on 7 datasets that are completely excluded from training.

3.2 Evaluation Results

In-domain Test. We first evaluate in-domain performance under both alignment strategies across 10 individually datasets, then compare large-scale cross-dataset joint training with single-dataset learning. The forecasting horizon H is evaluated at $\{96, 192, 336, 720\}$; due to space constraints, results are reported as the average over the four horizons. As shown in Figure 2, under the single-dataset learning paradigm, LLM4TSF with the pre-alignment strategy outperforms its post-alignment counterpart across most datasets. The performance gap is particularly pronounced on small-scale datasets that are prone to overfitting. In addition, we observe that cross-dataset learning leads to better performance, regardless of whether pre- or post-alignment strategies are adopted, as shown in Figure 3. It suggests that large-scale training on diverse TS data is more effective, and full results can be in Appendix H.1.

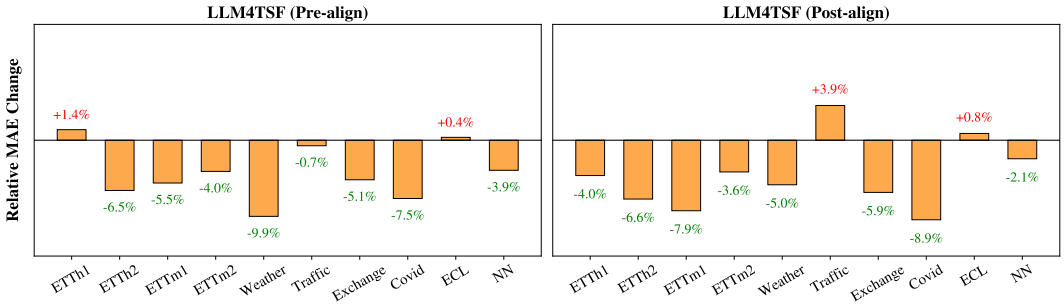

Figure 3: Comparison of LLM4TSF performance with pre- and post-alignment under single- and cross-dataset paradigm. Negative and Positive values indicate MAE decreases and increases.

Table 2: Ablation results of **LLM4TSF (Pre-align)** and **LLM4TSF (Post-align)**, averaged over horizons $\{96, 192, 336, 720\}$. For each dataset, the best MAE or MSE are highlighted in red or blue.

Models	LLM4TSF (Pre-align)						LLM4TSF (Post-align)					
	w/ Pre-training		w/o Pre-training		w/o LLM		w/ Pre-training		w/o Pre-training		w/o LLM	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
In-Domain Test												
ETTh1	0.445	0.447	0.460	0.471	0.432	0.438	0.436	0.431	0.428	0.420	0.435	0.442
ETTh2	0.388	0.348	0.414	0.377	0.407	0.363	0.396	0.357	0.429	0.403	0.439	0.424
ETTm1	0.375	0.353	0.393	0.382	0.363	0.328	0.384	0.354	0.373	0.340	0.378	0.351
ETTm2	0.309	0.252	0.330	0.290	0.315	0.271	0.325	0.265	0.341	0.290	0.355	0.307
Weather	0.256	0.225	0.275	0.244	0.268	0.240	0.266	0.226	0.275	0.242	0.287	0.261
Traffic	0.278	0.401	0.285	0.416	0.263	0.377	0.293	0.418	0.288	0.406	0.281	0.390
Exchange	0.389	0.332	0.423	0.426	0.407	0.385	0.402	0.384	0.465	0.517	0.477	0.539
Covid	0.049	1.383	0.066	2.135	0.053	1.539	0.051	1.436	0.059	1.722	0.063	1.995
ECL	0.268	0.168	0.277	0.184	0.255	0.158	0.265	0.169	0.274	0.177	0.269	0.181
NN	0.617	0.804	0.630	0.821	0.625	0.828	0.659	0.865	0.672	0.913	0.661	0.884
Out-of-Domain Test												
Wind	0.737	1.015	0.779	1.655	0.769	1.447	0.722	0.963	0.755	1.324	0.772	1.564
Solar	0.295	0.228	0.361	0.293	0.349	0.287	0.322	0.274	0.341	0.297	0.353	0.311
AQShunyi	0.445	0.612	0.464	0.679	0.451	0.638	0.472	0.688	0.487	0.706	0.493	0.717
CzenLan	0.308	0.274	0.367	0.342	0.359	0.311	0.319	0.288	0.336	0.301	0.355	0.337
ZafNoo	0.458	0.547	0.447	0.548	0.430	0.512	0.471	0.583	0.465	0.574	0.490	0.622
NASDAQ	0.635	0.735	0.936	1.955	0.922	1.854	0.640	0.749	0.672	0.793	0.715	1.023
PEMS	0.301	0.256	0.321	0.270	0.317	0.275	0.332	0.291	0.314	0.266	0.328	0.281

Out-of-domain Test. We compare LLM4TSF against three TS foundation models trained from scratch, namely Chronos (Ansari et al., 2024), UniTS (Gao et al., 2024), and Moirai (Woo et al., 2024a). All comparisons follow a fair evaluation setting, with detailed configurations provided in Appendix E. Notably, TSFMs use larger pretraining resources than ours. In addition, we include two LLM-based TSF models, UniTime (Liu et al., 2024b) and TimeLLM (Jin et al., 2023), which are trained using single-dataset few-shot learning with only 5% of the training data. As shown in Table 1, both LLM4TSF (Pre-align) and LLM4TSF (Post-align) achieve overall superior performance in zero-shot test compared to TS foundation models. Moreover, they even outperform LLM-based TSF models trained with 5% data. Full results are reported in Appendix H.5.

These results indicate that, when trained with pretrained LLM parameters and large-scale TS data, not only performs well in in-domain settings but also exhibits strong cross-domain generalization. Moreover, this advantage is amplified with increasing data scale, as shown in Figure 15.

4 Where do the Gains Really Come from?

Setup. Sec. 3 shows that LLM4TSF trained under cross-dataset learning achieves strong performance across a wide range of scenarios. However, it remains unclear where these performance gains come from. In this section, we conduct a attribution study by either removing the LLM module or randomly initializing parameters, thereby disrupting the pretrained knowledge and keeping the overall architecture and settings unchanged. This design enables us to isolate and evaluate the actual contribution, identifying the fundamental sources of gains.

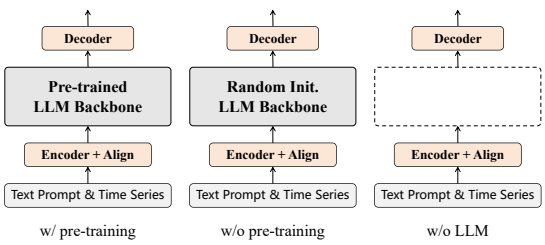

Figure 4: Original architecture and its two variants.

To investigate the impact of LLM parameters in TSF, we consider three model configurations(as shown in Figure 4) with different levels of reliance on pretrained LLMs: (1) ***w/ pre-training***, where the LLM retains its pretrained weights; in this setting, the LLM components are frozen in LLM4TSF(Pre-align) and fine-tuned in LLM4TSF(Post-align); (2) ***w/o pre-training***, which follows the same architecture as the pretrained setting but randomly initializes all LLM parameters; and (3) ***w/o LLM***, where the LLM components are entirely removed from the architecture. The three are trained following the same procedure as in Sec. 3.1 and evaluated on many scenarios.

Main Results. As shown in Table 2, when comparing the w/ pre-training against the two ablated counterparts (w/o pre-training and w/o LLM), LLM4TSF(Pre-align) with pretrained LLM parameters achieves the lowest forecasting error on ***6/10*** datasets in the in-domain test and ***6/7*** datasets in the out-of-domain test. Similarly, LLM4TSF(Post-align) with pre-training attains the lowest error on 7/10 in-domain and 5/7 out-of-domain datasets. To better understand the role of LLMs in TSF, we disentangle the effects of LLM, to distinguish whether gains arise from pretrained semantic priors or merely from architectural modeling capacity.

(I) w/ pre-training vs. w/o pre-training. we observe that LLM4TSF(Pre-align) benefits from LLM prior knowledge: training with pretrained LLM parameters outperforms random initialization on all datasets (i.e., ***10/10*** in-domain and ***7/7*** out-of-domain test). For LLM4TSF(Post-align), models w/ pre-training perform better on ***7/10*** in-domain datasets and ***5/7*** out-of-domain test. **(II) w/o pre-training vs. w/o LLM.** For LLM4TSF(Pre-align), retaining a randomly initialized LLM consistently underperforms directly removing the LLM backbone across all tests (***10/10*** in-domain and ***7/7*** out-of-domain). In contrast, for LLM4TSF(Post-align), retaining a randomly initialized LLM outperforms removing the LLM backbone on ***7/10*** in-domain tests and on ***all*** out-of-domain tests.

Takeaways. The gains of LLM4TSF arise from both LLM parameters and architectural capacity. Moreover, we identify three interesting findings: ***(1) Across different alignment strategies, pre-trained priors contribute to performance to varying extents;*** and ***(2) freezing LLM causes randomly initialized models to collapse, therefore under post-alignment, fully trainable LLMs can be optimized from scratch and outperform w/o LLM. (3) pre-alignment achieves stronger performance under single- & cross-dataset settings.***

Why Pre-alignment Is More Effective: A Distributional Space Perspective. We analyze the compatibility between TS and language spaces from a representation-distribution perspective. Specifically, we adopt two metrics: Normalized Subspace Projection Error (NSPE) (Mu & Viswanath, 2018; Ethayarajh, 2019) and Relative Fréchet Distance (RFD) (Dowson & Landau, 1982; Heusel et al., 2017). NSPE measures the normalized projection error of TS embeddings onto the principal language subspace, while RFD compares the FD between two modality distributions. Lower NSPE or FD indicates better alignment with the pretrained LLM language space. Detailed metric definitions are provided in Appendix G.1.

As shown in Figure 5, in the in-domain layer-wise analysis, pre-alignment achieves a much lower NSPE at Init.(Init. denotes TS embeddings before the LLM), showing that TS embeddings are already closer to the pretrained language space before entering the LLM. In contrast, post-alignment relies on LLM layers to gradually reduce the modality gap, but its final NSPE remains higher. Consistently, $\mathrm{RFD} < 1$ across all layers indicates a smaller FD for pre-alignment. Thus, pre-alignment more directly makes TS embeddings easier for the pretrained LLM to utilize. Additionally, we provide a detailed and comprehensive theoretical analysis in Appendix G.2.

5 Understanding When LLM4TSF Works

In Sec. 4, We observe that in 71% of the in-domain and out-of-domain tests, the w/ pre-training setting consistently outperforms both w/o pre-training and w/o LLM. Nevertheless, a small number of datasets exhibit failure cases where such benefits do not materialize. Accordingly, in this section, we first analyze ***(i) why LLMs may fail under certain TSF scenarios***, and investigate ***(ii) how to better leverage LLM strengths to maximize their effectiveness in TSF***.

5.1 Exploring LLM Preferences over TS Distributions

Differences in the statistical properties of TS. TS data exhibit substantial differences in statistical properties (Wang et al., 2025d; Siru et al., 2025; Cini et al., 2025). On the one hand, TS often

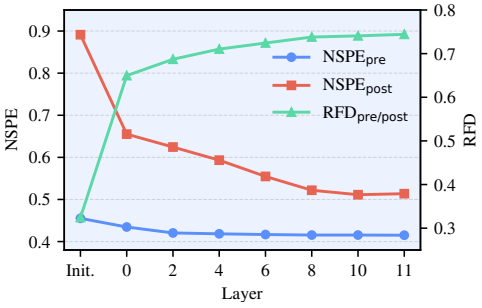

Figure 5: Representation alignment analysis.

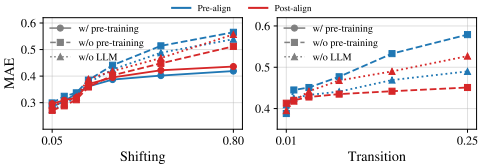

Figure 6: Effect of shifting and transition.

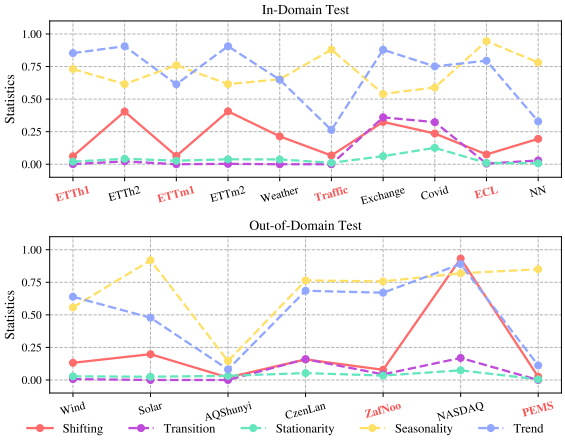

Figure 7: Statistical properties of different datasets. Datasets highlighted in red indicate cases where the w/ pre-training setting underperforms the corresponding ablation baselines during test evaluation.

contain prominent global structural characteristics, which describe stable regularities over extended time horizons (Wang et al., 2025a; Majeedi et al., 2025). On the other hand, TS exhibit local dynamic variations, including distribution shifting, changes in stationarity, and regime transitions (Masserano et al., 2024; Hu et al., 2025a), which reflect the evolving of temporal data.

Dataset-level statistical analysis. We analyze the statistical properties of all datasets used for evaluation from five perspectives, as shown in Figure 13. **(I) in-domain datasets.** In terms of shifting, ETTh2, ETTm2, Weather, Exchange, Covid, and NN exhibit high shifting values, indicating pronounced changes in data distributions over time. The transition shows that ETTh2, ETTm2, Weather, Exchange, and Covid experience more frequent and complex state changes, reflecting multi-stage and multi-pattern dynamics. Regarding stationarity, Exchange and Covid display lower stationarity levels (larger values indicate poorer stationarity). However, seasonality and trend are balanced across datasets. **(II) out-of-domain datasets.** NASDAQ exhibits an exceptionally high shifting value. In terms of stationarity, both CzenLan and NASDAQ show low stationarity levels. By contrast, the remaining properties—transition, seasonality, and trend, are relatively balanced across datasets. More details are provided in Appendix F.2.

Interactions between properties and performance. ***Based on the analysis, we observe a clear association between the TS properties and the model performance.*** When a TS exhibits strong shifting, pretrained LLM tends to yield more pronounced gains. In contrast, when shifting is weak, satisfactory performance can often be achieved using only the encoder and decoder modules, as shown in Figure 7. Moreover, under post-alignment with trainable LLM parameters, a randomly initialized LLM can still outperform removing the LLM backbone on high-transition datasets. We use controlled synthetic data to isolate the effects of shifting and transition on forecasting error, as shown in Figure 6. More details are provided in Appendix H.6 & Appendix I. As shifting and transition increase, MAE rises and LLM advantages become clearer. Even w/o pretraining, a trainable LLM outperforms non-LLM models under high-transition regimes.

A Distribution-space View of Shifting. As shown in Table 3, under post-alignment, as shifting increases from 0.1 to 0.7, the Init. NSPE rises from 0.7840 to 0.9014, indicating stronger initial mismatch between TS embeddings and the language space. Meanwhile, the NSPE reduction brought by the LLM backbone increases from 0.3598 to 0.3942, suggesting that stronger shifting makes the model more reliant on the pretrained LLM to narrow the modality gap. Additionally, we provide a detailed and comprehensive theoretical analysis in Appendix G.2.

Table 3: Effect of shifting on NSPE.

Shift	$\text{NSPE}_{\text{Init.}}$	$\text{NSPE}_{\text{Layer-11}}$	Drop
0.1	0.7840	0.4242	0.3598
0.3	0.8457	0.4611	0.3846
0.5	0.8627	0.4728	0.3899
0.7	0.9014	0.5172	0.3942

Takeaways. We find that shifting and transition play distinct roles in determining the effectiveness of LLM backbone in TSF: ***(1) when shifting is strong, pretrained LLM parameters are more likely***

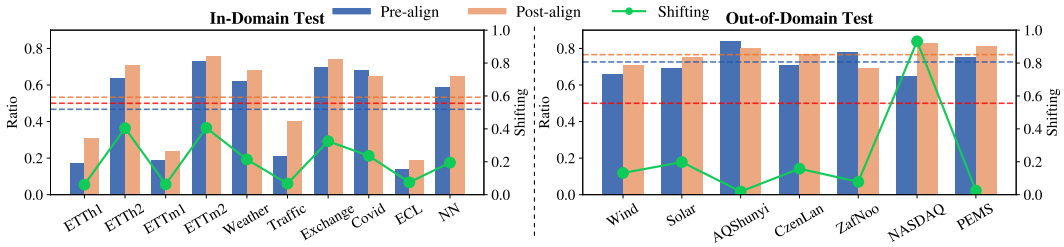

Figure 8: Relationship between the token ratio passing through the LLM (averaged over horizons $\{96, 192, 336, 720\}$) and shifting across different datasets. The "- - -" & "- - -" indicate the average ratios of each dataset, and the "- - -" denotes the reference value of 0.5.

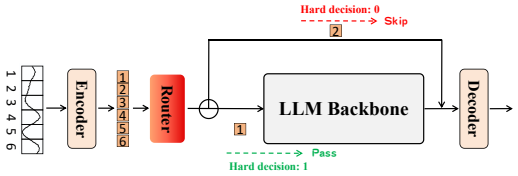

Figure 9: Routing analysis illustration. For each TS token, a trained router decides whether to route it to the LLM or skip. The LLM backbone is either w/ pre-training or w/o pre-training.

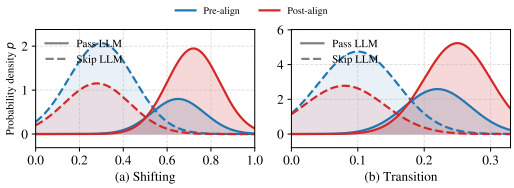

Figure 10: (a) With pre-trained, tokens of passing LLM are concentrated in high shifting regions. (b) With pre-trained, tokens of passing LLM are concentrated in high transition.

to provide meaningful performance benefits; (2) when transition is high, a trainable Transformer backbone, even without pretrained LLM parameters, outperforms the w/o LLM variant.

5.2 Routing Analysis: Pass Through the LLM or Skip?

Previous analyses show the benefits of LLMs in TSF are scenario-dependent. We therefore examine, when LLMs are more likely to play an active role, as shown in the Figure 9. Specifically, we employ a routing mechanism to assign path preferences to individual TS segments, referring to Appendix J. Based on these observations, we pose the following four research questions to guide analysis:

RQ1: What properties of TS are likely to rely on LLMs? We train LLM4TSF (Pre-align) & (Post-align) under a cross-dataset learning setting, where each token is routed to either pass through or skip the LLM. As shown in Figure 8, datasets with stronger shifting or unseen (out-of-domain) distributions tend to exhibit higher passing ratios. Moreover, the Post-align setting leads to higher passing ratios compared to Pre-align. Since TS segments within a dataset can vary, dataset-level analysis alone cannot fully explain router decisions. Therefore, we analyze the relationship between the properties of all test-set TS segments and their passing ratios to the LLM. Specifically, we visualize the joint density $p(x, D)$, where x denotes properties (e.g., shifting or transition) and $D \in \{\text{pass}, \text{skip}\}$ indicates the routing decision. Each density curve is normalized such that its integral corresponds to the routing ratio $\mathbb{P}(D = d)$, as shown in Figure 10. More results are provided in Appendix H.6.

RQ2: Do pre-trained parameters influence dependency? We compare pretrained and initialized LLMs to assess how pretraining affects LLM usage. In Table 4, datasets with higher shifting, such as ETTh2, ETTm2, Weather, and Exchange, tend to pass more tokens through the LLM, but this tendency weakens under w/o pre-training. In contrast, lower-shifting datasets, such as ETTh1, ETTm1, Traffic, and ECL, show the opposite pattern. For LLM4TSF (Post-align), datasets with lower transition are more likely to skip the LLM, showing that the encoder and decoder alone may be sufficient. Overall, the results in Table 2 and Table 4, indicate that tokens favor lower-error paths, with shifting and transition driving path selection.

Table 4: Impact of LLM parameters on the token passing ratio, averaged over horizons.

	LLM4TSF (Pre-align)			LLM4TSF (Post-align)		
	w/ Pre-training	w/o Pre-training	Change	w/ Pre-training	w/o Pre-training	Change
In-Domain Test						
ETTh1	**17%**	**24%**	+7%	**31%**	57%	+26%
ETTh2	64%	**43%**	-21%	71%	62%	-9%
ETTm1	**19%**	**14%**	-5%	**24%**	66%	+42%
ETTm2	73%	**28%**	-45%	76%	68%	-8%
Weather	62%	**39%**	-23%	68%	62%	-6%
Traffic	**21%**	**27%**	+6%	**40%**	**46%**	+6%
Exchange	70%	**31%**	-39%	74%	59%	-15%
Covid	68%	**15%**	-53%	65%	55%	-10%
ECL	**14%**	**21%**	+7%	**19%**	**37%**	+18%
NN	59%	**26%**	-23%	65%	**43%**	-22%
Out-of-Domain Test						
Wind	66%	**14%**	-52%	71%	65%	-6%
Solar	69%	**18%**	-51%	75%	60%	-15%
AQShunyi	84%	**25%**	-59%	80%	58%	-22%
CzenLan	71%	**21%**	-50%	77%	71%	-6%
ZafNoo	78%	**26%**	-52%	69%	79%	+10%
NASDAQ	65%	**19%**	-46%	83%	75%	-8%
PEMS	75%	**15%**	-60%	81%	85%	+4%

*Ratio **<50%** indicates a stronger tendency to skip the LLM.

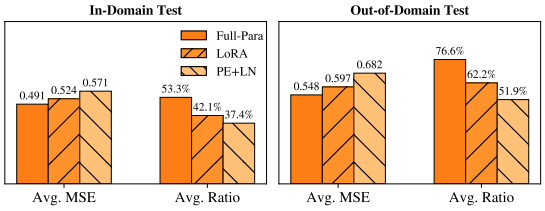

Figure 11: LLM4TSF (Post-align) performance and token passing ratios under three strategies.

RQ3: Do training strategies modulate the LLMs? By comparing different training strategies, we analyze whether they alter the overall tendency to use or skip LLM. Taking full-parameter fine-tuning as the baseline, we consider LoRA (Hu et al., 2022) and a lightweight strategy that updates only positional embeddings and layer normalization, following prior work (Zhou et al., 2023b; Hu et al., 2025c). Figure 11 show that Full-Para enables the most effective utilization of the LLM, yielding the best overall performance. It indicates that unrestricted parameter optimization may better exploit the model's representational capacity.

RQ4: Do stronger LLM lead to greater gains? We replace the LLM backbone with more stronger Qwen-3 (Yang et al., 2025) and observe that stronger general capabilities do not consistently translate into improved performance, as shown in Table 17. However, we find that artificially truncating the model by retaining only half of its layers, as in prior work (Jin et al., 2023; Liu et al., 2024c; 2025a; Hu et al., 2025c), leads to noticeable performance degradation (see Appendix H.3). More importantly, omitting prompts weakens the effectiveness of LLMs and leads to noticeable performance degradation, an effect that is particularly pronounced in out-of-domain scenarios. It suggests that enriching prompt information is more impactful than indiscriminately scaling up the model.

RQ5: Does routing causally contribute to performance gains? We perform interventions to verify the effect of learned routing decisions. Forcing high-shifting sub-sequences (> 0.2) through the LLM reduces MSE from 1.383 to 1.214 on Covid and from 0.735 to 0.642 on NASDAQ. Larger gains are observed on datasets with stronger shifting patterns. Meanwhile, suppressing the LLM for low-shifting sub-sequences (< 0.1) generally preserves performance. This pattern supports selective LLM activation under suitable temporal regimes. See Appendix C for detailed results.

6 Conclusion

Discussion. ***(1) Cross-dataset learning is a crucial prerequisite for unlocking the full potential of LLMs.*** By exposing LLMs to diverse data, cross-dataset learning enables stronger performance. ***(2) Pre-alignment provides a more effective integration strategy for LLM4TSF.*** Aligning TS inputs with word embeddings to lead to more compatible representations, resulting in lower errors. ***(3) The advantages of LLM4TSF arise from both prior knowledge and architectural modeling capacity.*** They both contribute to improved forecasting capability. ***(4) LLM4TSF exhibit inherent preferences toward certain TS properties.*** In particular, LLM tend to achieve advantages on TS with pronounced shifting or complex transition patterns. ***(5) Both a complete backbone and sufficient parameter optimization are essential for achieving strong performance.*** Preserving the full LLM backbone and enabling adequate parameter optimization are necessary to leverage the LLMs. ***(6) The routing analysis provides direct evidence for the macroscopic performance.*** The routing offer an view of how LLM4TSF allocates modeling capacity, serving as a concrete explanation. ***(7) Blindly scaling up LLM does not lead to better performance.*** Simply increasing LLM size without adequate modality alignment may yield limited returns. More analysis are provided in Appendix B.

Conclusion. In this work, we revisit the role of LLM4TSF and provide a clear characterization of when and why LLM4TSF models are effective. In addition, we show that the benefits of LLMs are neither universal nor incidental, but arise from the interplay between LLM knowledge and architectural capacity under specific distributions. By fine-grained routing analysis, we reveal that LLMs exhibit consistent preferences over TS with different statistical properties, which explains their empirical gains and limitations. We not only clarify ongoing debates surrounding LLM4TSF, but also offer guidance for designing more principled models. Our work has two main limitations. ***First***, our study relies on large-scale cross-dataset training, while such heterogeneous TS collections may be difficult to obtain in practice. ***Second***, our analysis focuses primarily on forecasting, and extending the findings to broader TS tasks.

Limitations. ***First***, our study relies on large-scale cross-dataset training, while such heterogeneous TS collections may be difficult to obtain in practice. ***Second***, our analysis focuses primarily on forecasting, and extending the findings to broader TS tasks.

AI USE STATEMENT

Generative AI tools were used solely for language polishing, including improving grammar, clarity, fluency, and overall readability of the manuscript. They were not relied upon to determine the scientific claims, experimental results, or conclusions of this work. All AI-assisted revisions were carefully reviewed, verified, and, when necessary, further modified by the authors to ensure accuracy and consistency with the original technical content.We take full responsibility for the final manuscript and all of its contents.

REPRODUCIBILITY STATEMENT

We provide detailed information to facilitate reproduction of our experiments. Dataset statistics, splits, preprocessing procedures, and data sources are documented in Appendix F. The construction of controlled synthetic time series is described in Appendix I, while the textual prompts used for different datasets are provided in Appendix D. Complete training hyperparameters and routing configurations are reported in the Reproducibility Details section, and the implementation of the token-level routing mechanism, including Gumbel-Softmax and the straight-through estimator, is described in Appendix J. Additional experimental results and analyses are provided in Appendix B, Appendix C, and the supplementary experimental sections.

ETHICS STATEMENT

This work does not involve human participants, personal or sensitive data, or interventions that raise specific ethical concerns. All experiments are conducted on publicly available datasets. Therefore, no additional ethical approval or consideration is required for this study.

REFERENCES

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024, 2024.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Yuxuan Bian, Xuan Ju, Jiangtong Li, Zhijian Xu, Dawei Cheng, and Qiang Xu. Multi-patch prediction: Adapting llms for time series representation learning. *arXiv preprint arXiv:2402.04852*, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Hu Cao, Yueyue Wang, Joy Chen, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, and Manning Wang. Swin-unet: Unet-like pure transformer for medical image segmentation. In *European conference on computer vision*, pp. 205–218. Springer, 2022.

Vladimir Ceperic and Tomislav Markovic. Transforming time-series data for improved llm-based forecasting through adaptive encoding. *Int. J. Simul. Syst. Sci. Technol*, 25:8–1, 2024.

Ching Chang, Wei-Yao Wang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Aligning pre-trained llms as data-efficient time-series forecasters. *ACM Transactions on Intelligent Systems and Technology*, 16(3):1–20, 2025.

Yu Chen, Nathalia Céspedes, and Payam Barnaghi. A closer look at transformers for time series forecasting: Understanding why they work and where they struggle. In *International Conference on Machine Learning*, pp. 7763–7780. PMLR, 2025.

De Cheng, Zhipeng Xu, Xinyang Jiang, Nannan Wang, Dongsheng Li, and Xinbo Gao. Disentangled prompt representation for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23595–23604, 2024.

Andrea Cini, Alexander Jenkins, Danilo Mandic, Cesare Alippi, and Filippo Maria Bianchi. Relational conformal prediction for correlated time series. *arXiv preprint arXiv:2502.09443*, 2025.

Jan G De Gooijer and Rob J Hyndman. 25 years of time series forecasting. *International journal of forecasting*, 22(3):443–473, 2006.

Linhao Dong, Shuang Xu, and Bo Xu. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5884–5888. IEEE, 2018.

David C Dowson and BV666017 Landau. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 12(3):450–455, 1982.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 55–65, 2019.

Shanghua Gao, Teddy Koker, Owen Queen, Tom Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. Units: A unified multi-task time series model. *Advances in Neural Information Processing Systems*, 37:140589–140631, 2024.

Huan He, Owen Queen, Teddy Koker, Consuelo Cuevas, Theodoros Tsiligkaridis, and Marinka Zitnik. Domain adaptation for time series under feature and label shifts. In *International conference on machine learning*, pp. 12746–12774. PMLR, 2023.

Benedikt Heidrich, Marian Turowski, Kaleb Phipps, Kai Schmieder, Wolfgang Süß, Ralf Mikut, and Veit Hagenmeyer. Controlling non-stationarity and periodicities in time series generation using conditional invertible neural networks. *Applied Intelligence*, 53(8):8826–8843, 2023.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. Timefilter: Patch-specific spatial-temporal graph filtration for time series forecasting. *arXiv preprint arXiv:2501.13041*, 2025a.

Yiyang Hu, Haibin Liao, Mou Wu, and Li Yuan. Sst-llm: time series forecasting based on large language models. In *International Symposium on Artificial Intelligence Innovations (IS-AII 2025)*, volume 13681, pp. 198–206. SPIE, 2025b.

Yuxiao Hu, Qian Li, Dongxiao Zhang, Jinyue Yan, and Yuntian Chen. Context-alignment: Activating and enhancing llms capabilities in time series. In *International Conference on Learning Representations*, volume 2025, 2025c.

Longfei Huang, Shupeng Zhong, Xiangyu Wu, and Ruoxuan Li. The solution for the cvpr2024 nice image captioning challenge. *arXiv preprint arXiv:2404.12739*, 2024.

Ruizheng Huang, Zhicheng Zhang, and Yong Wang. Cross-moe: An efficient temporal prediction framework integrating textual modality. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 29915–29926, 2025.

Chuong Huynh, Jinyu Yang, Ashish Tawari, Mubarak Shah, Son Tran, Raffay Hamid, Trishul Chilimbi, and Abhinav Shrivastava. Collm: A large language model for composed image retrieval. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 3994–4004, 2025a.

Ngoc Dung Huynh, Mohamed Reda Bouadjenek, Sunil Aryal, Imran Razzak, and Hakim Hacid. Visual question answering: from early developments to recent advances–a survey. *arXiv preprint arXiv:2501.03939*, 2025b.

Saachi Jain, Hadi Salman, Alaa Khaddaj, Eric Wong, Sung Min Park, and Aleksander Mkadry. A data-based perspective on transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3613–3622, 2023.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Yushan Jiang, Zijie Pan, Xikun Zhang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. Empowering time series analysis with large language models: A survey. *arXiv preprint arXiv:2402.03182*, 2024.

Yushan Jiang, Wenchao Yu, Geon Lee, Dongjin Song, Kijung Shin, Wei Cheng, Yanchi Liu, and Haifeng Chen. Explainable multi-modal time series prediction with llm-in-the-loop. *arXiv preprint arXiv:2503.01013*, 2025.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

Jongseon Kim, Hyungjoon Kim, HyunGi Kim, Dongjun Lee, and Sungroh Yoon. A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges. *Artificial Intelligence Review*, 58(7):1–95, 2025.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International conference on learning representations*, 2021.

Xiangjie Kong, Zhenghao Chen, Weiyao Liu, Kaili Ning, Lechao Zhang, Syauqie Muhammad Marier, Yichen Liu, Yuhao Chen, and Feng Xia. Deep learning for time series forecasting: a survey. *International Journal of Machine Learning and Cybernetics*, pp. 1–34, 2025.

Kartik Kuckreja, Muhammad Sohail Danish, Muzammal Naseer, Abhijit Das, Salman Khan, and Fahad Shahbaz Khan. Geochat: Grounded large vision-language model for remote sensing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27831–27840, 2024.

Geon Lee, Wenchao Yu, Kijung Shin, Wei Cheng, and Haifeng Chen. Timecap: Learning to contextualize, augment, and predict time series events with large language model agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18082–18090, 2025.

Peixuan Lei, Jie Song, Yuzhe Hao, Tao Chen, Yuxuan Zhang, LEI JIA, Yuanxiang Li, et al. Itformer: Bridging time series and natural language for multi-modal qa with large-scale multitask dataset. In *Forty-second International Conference on Machine Learning*, 2025.

Ming Li, Muyu Yang, Shaolong Chen, Huangyongxiang Li, Gaosong Xing, and Shuting Li. Fcpformer: Enhancing long-term multivariate time series forecasting with frequency compensation. *Sensors*, 25(18):5646, 2025.

Xiao Liang, Erkun Yang, Cheng Deng, and Yanhua Yang. Crossformer: Cross-modal representation learning via heterogeneous graph transformer. *ACM Transactions on Multimedia Computing, Communications and Applications*, 20(12):1–21, 2024.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 5971–5984, 2024.

Yang Lin, Irena Koprinska, and Mashud Rana. Ssdnet: State space decomposition neural network for time series forecasting. In *2021 IEEE International conference on data mining (ICDM)*, pp. 370–378. IEEE, 2021.

Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025a.

Chenxi Liu, Shaowen Zhou, Qianxiong Xu, Hao Miao, Cheng Long, Ziyue Li, and Rui Zhao. Towards cross-modality modeling for time series analytics: A survey in the llm era. *arXiv preprint arXiv:2505.02583*, 2025b.

Haoxin Liu, Shangqing Xu, Zhiyuan Zhao, Lingkai Kong, Harshavardhan Prabhakar Kamarthi, Aditya Sasanur, Megha Sharma, Jiaming Cui, Qingsong Wen, Chao Zhang, et al. Time-mmd: Multi-domain multimodal dataset for time series analysis. *Advances in Neural Information Processing Systems*, 37:77888–77933, 2024a.

Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. Calf: Aligning llms for time series forecasting via cross-modal fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025c.

Xinhe Liu and Wenmin Wang. Deep time series forecasting models: A comprehensive survey. *Mathematics*, 12(10):1504, 2024.

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

Yaru Liu, Lei Wang, and Bing Feng Ng. Multitask-transfer-learning method for random-force frequency identification considering multisource uncertainties. *AIAA Journal*, 63(6):2345–2360, 2025d.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in neural information processing systems*, 35: 9881–9893, 2022a.

Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Autotimes: Autoregressive time series forecasters via large language models. *Advances in Neural Information Processing Systems*, 37:122154–122184, 2024c.

Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models. In *International Conference on Machine Learning*, pp. 39295–39317. PMLR, 2025e.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3202–3211, 2022b.

Abrar Majeedi, Viswanatha Reddy Gajjala, Satya Sai Srinath Namburi GNVV, Nada Magdi Elkordi, and Yin Li. Lets forecast: Learning embedology for time series forecasting. *arXiv preprint arXiv:2506.06454*, 2025.

Ricardo P Masini, Marcelo C Medeiros, and Eduardo F Mendes. Machine learning advances for time series forecasting. *Journal of economic surveys*, 37(1):76–111, 2023.

Luca Masserano, Abdul Fatir Ansari, Boran Han, Xiyuan Zhang, Christos Faloutsos, Michael W Mahoney, Andrew Gordon Wilson, Youngsuk Park, Syama Rangapuram, Danielle C Maddix, et al. Enhancing foundation models for time series forecasting via wavelet-based tokenization. *arXiv preprint arXiv:2412.05244*, 2024.

Romain Meunier, Farah Benamara, Véronique Moriceau, Zhongzheng Qiao, and Savitha Ramasamy. Crisists: Coupling social media textual data and meteorological time series for urgency classification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16082–16099, 2025.

Jong Hak Moon, Hyungyung Lee, Woncheol Shin, Young-Hak Kim, and Edward Choi. Multi-modal understanding and generation for medical images and text via vision-language pre-training. *IEEE Journal of Biomedical and Health Informatics*, 26(12):6070–6080, 2022.

Jiaqi Mu and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*, 2018.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

Kanghui Ning, Zijie Pan, Yu Liu, Yushan Jiang, James Yiming Zhang, Kashif Rasul, Anderson Schneider, Lintao Ma, Yuriy Nevmyvaka, and Dongjin Song. Ts-rag: Retrieval-augmented generation based time series foundation models are stronger zero-shot forecaster. *arXiv preprint arXiv:2503.07649*, 2025.

François Painblanc, Laetitia Chapel, Nicolas Courty, Chloé Friguet, Charlotte Pelletier, and Romain Tavenard. Match-and-deform: Time series domain adaptation through optimal transport and temporal alignment. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 341–356. Springer, 2023.

Zijie Pan, Yushan Jiang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. Ssip-llm: Semantic space informed prompt learning with llm for time series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

Xin Qiu, Junlong Tong, Yirong Sun, Yunpu Ma, and Xiaoyu Shen. The few govern the many: Unveiling few-layer dominance for time series models. *arXiv preprint arXiv:2511.07237*, 2025.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Zhong Siru, Ruan Weilin, Ming Jin, Li Huan, Wen Qingsong, and Liang Yuxuan. Time-vlm: Exploring multimodal vision-language models for augmented time series forecasting. In *Forty-Second International Conference on Machine Learning (ICML 2025)*. Proceedings of Machine Learning Research, 2025.

Siming Sun, Kai Zhang, Xuejun Jiang, Wenchao Meng, and Qinmin Yang. Enhancing llms for time series forecasting via structure-guided cross-modal alignment. *arXiv preprint arXiv:2505.13175*, 2025.

Mingtian Tan, Mike Merrill, Vinayak Gupta, Tim Althoff, and Tom Hartvigsen. Are language models actually useful for time series forecasting? *Advances in Neural Information Processing Systems*, 37:60162–60191, 2024.

Xiaoyu Tao, Shilong Zhang, Mingyue Cheng, Daoyu Wang, Tingyue Pan, Bokai Pan, Changqing Zhang, and Shijin Wang. From values to tokens: An llm-driven framework for context-aware time series forecasting via symbolic discretization. *arXiv preprint arXiv:2508.09191*, 2025.

Junlong Tong, Liping Xie, and Kanjian Zhang. Probabilistic decomposition transformer for time series forecasting. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 478–486. SIAM, 2023.

Bin Wang, Heming Yang, and Jinfang Sheng. Timecf: A timemixer-based model with adaptive convolution and sharpness-aware minimization frequency domain loss for long-term time seris forecasting. *arXiv preprint arXiv:2505.17532*, 2025a.

Hao Wang, Licheng Pan, Zhichao Chen, Xu Chen, Qingyang Dai, Lei Wang, Haoxuan Li, and Zhouchen Lin. Transdf: Time-series forecasting needs transformed label alignment. *arXiv preprint arXiv:2505.17847*, 2025b.

Lan Wang, Wei Ao, Vishnu Naresh Boddeti, and Ser-Nam Lim. Generative zero-shot composed image retrieval. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 29690–29700, 2025c.

Xinlei Wang, Maike Feng, Jing Qiu, Jinjin Gu, and Junhua Zhao. From news to forecast: Integrating event analysis in llm-based time series forecasting with reflection. *Advances in Neural Information Processing Systems*, 37:58118–58153, 2024.

Yihang Wang, Yuying Qiu, Peng Chen, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. Towards a general time series forecasting model with unified representation and adaptive transfer. In *Forty-second International Conference on Machine Learning*, 2025d.

Ziqiao Wang and Yongyi Mao. On f-divergence principled domain adaptation: An improved framework. *Advances in Neural Information Processing Systems*, 37:6711–6748, 2024.

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):9, 2016.

Michał Wiliński, Mononito Goswami, Willa Potosnak, Nina Żukowska, and Artur Dubrawski. Exploring representations and interventions in time series foundation models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=goVzfYtj58.

Malcolm L Wolff, Shenghao Yang, Kari Torkkola, and Michael W Mahoney. Using pre-trained llms for multivariate time series forecasting. *arXiv preprint arXiv:2501.06386*, 2025.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024a.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*, 2024b.

Congxi Xiao, Jingbo Zhou, Yixiong Xiao, Xinjiang Lu, Le Zhang, and Hui Xiong. Timefound: A foundation model for time series forecasting. *arXiv preprint arXiv:2503.04118*, 2025.

Jiahong Xiong, Chengsen Wang, Haifeng Sun, Yuhan Jing, Qi Qi, Zirui Zhuang, Lei Zhang, Jianxin Liao, and Jingyu Wang. Beyond statistical analysis: Multimodal framework for time series forecasting with llm-driven temporal pattern. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 6696–6704, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025.

Haiyang Yu, Siyang Yi, Ke Niu, Minghan Zhuo, and Bin Li. Umit: Unifying medical imaging tasks via vision-language models. *arXiv preprint arXiv:2503.15892*, 2025.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

Ariana Yunita, MHD Iqbal Pratama, Muhammad Zaki Almuzakki, Hani Ramadhan, Emelia Akashah P Akhir, Andi Besse Firdausiah Mansur, and Ahmad Hoirul Basori. Performance analysis of neural network architectures for time series forecasting: A comparative study of rnn, lstm, gru, and hybrid models. *MethodsX*, 15:103462, 2025.

Yang Zhan, Zhitong Xiong, and Yuan Yuan. Skyeyegpt: Unifying remote sensing vision-language tasks via instruction tuning with large language model. *ISPRS Journal of Photogrammetry and Remote Sensing*, 221:64–77, 2025.

Wenchang Zhang, Hua Wang, and Fan Zhang. Skip-timeformer: Skip-time interaction transformer for long sequence time-series forecasting. In *International joint conference on artificial intelligence*, pp. 5499–5507, 2024a.

Xinyu Zhang, Shanshan Feng, and Xutao Li. From text to time? rethinking the effectiveness of the large language model for time series forecasting. *arXiv preprint arXiv:2504.08818*, 2025.

Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh K Gupta, and Jingbo Shang. Large language models for time series: A survey. *arXiv preprint arXiv:2402.01801*, 2024b.

Yichi Zhang, Yinpeng Dong, Siyuan Zhang, Tianzan Min, Hang Su, and Jun Zhu. Exploring the transferability of visual prompting for multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26562–26572, 2024c.

Liangwei Nathan Zheng, Chang Dong, Wei Emma Zhang, Lin Yue, Miao Xu, Olaf Maennel, and Weitong Chen. Understanding why large language models can be ineffective in time series analysis: The impact of modality alignment. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 4026–4037, 2025.

Liangwei Nathan Zheng, Wenhao Liang, Wei Emma Zhang, Miao Xu, Olaf Maennel, and Weitong Chen. Lifting manifolds to mitigate pseudo-alignment in llm4ts. In *Proceedings of the ACM Web Conference 2026*, pp. 3764–3775, 2026.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023a.

Shiqiao Zhou, Holger Schöner, Huanbo Lyu, Edouard Fouché, and Shuo Wang. Balm-tsf: Balanced multimodal alignment for llm-based time series forecasting. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 4498–4508, 2025.

Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355, 2023b.

A Related Work

A.1 LLMs for TSF

Directly applying fully pretrained LLMs to TSF tasks poses a central challenge in achieving effective modality alignment, as LLMs are originally optimized for discrete textual tokens rather than continuous temporal signals. Bridging the representational gap between TS data and the linguistic embedding space of LLMs has therefore become a key research focus (Jiang et al., 2024; Wang et al., 2024; Liu et al., 2024c; 2025b; Jiang et al., 2025). Existing studies addressing this challenge can be broadly categorized into two representative paradigms, depending on whether alignment is performed before or after the LLM is involved in the modeling pipeline (Zhou et al., 2025; Xiong et al., 2025; Hu et al., 2025b; Tao et al., 2025). The first category follows a pre-alignment strategy, which aims to align TS and textual modalities prior to LLM input. In this paradigm, raw or encoded TS are transformed into intermediate representations that are compatible with the LLM token embedding space, often via learnable projection layers or modality-specific encoders. These TS representations are then concatenated or interleaved with textual prompts and fed into a frozen LLM for downstream forecasting tasks. By keeping the LLM parameters fixed, this approach preserves the pretrained linguistic and semantic knowledge of the LLM while enabling it to process TS information in a unified embedding space. As a result, pre-alignment methods offer strong parameter efficiency and stability, making them particularly attractive in low-resource or deployment-constrained scenarios (Zhang et al., 2024b; Ceperic & Markovic, 2024). The second category adopts a post-alignment strategy, in which LLMs are fine-tuned to adapt to TSF tasks by reducing the representational discrepancy between textual and TS modalities within the LLM embedding space (Bian

et al., 2024; Lee et al., 2025; Sun et al., 2025). Rather than enforcing compatibility before input, these methods rely on task-driven learning signals to implicitly or explicitly align modalities during training. This is commonly achieved through joint optimization objectives, cross-modal alignment losses, or auxiliary supervision that encourages coherent representations across modalities. While the original LLM architecture is typically preserved, a subset of model parameters is updated to improve task-specific performance. Consequently, post-alignment approaches offer greater flexibility and expressive power at the cost of increased training complexity and computational overhead.

A.2 Transfer Learning

Transfer learning has become a widely adopted paradigm in deep learning, demonstrating remarkable effectiveness across a wide range of domains (Weiss et al., 2016; Jain et al., 2023; Wang & Mao, 2024; Zhang et al., 2024c; Liu et al., 2025d). By pretraining models on large-scale datasets and transferring them to downstream tasks, prior work has shown substantial improvements in both performance and data efficiency. In vision–language research, for example, pretrained multimodal models jointly learn representations from images and text, enabling effective transfer to applications such as image retrieval (Wang et al., 2025c; Huynh et al., 2025a), image captioning (Yu et al., 2022; Huang et al., 2024), and visual question answering (Lin et al., 2024; Huynh et al., 2025b). In speech and language modeling, pretraining strategies that integrate acoustic signals with textual supervision have significantly advanced automatic speech recognition and semantic understanding. Similar successes have also been observed in domains such as healthcare (Moon et al., 2022; Yu et al., 2025) and remote sensing (Kuckreja et al., 2024; Zhan et al., 2025). These advances suggest that when different modalities exhibit strong semantic complementarity, large-scale pretrained models, particularly LLMs, can learn representations with high transferability and generalization capability. However, this assumption does not directly extend to TS data and TSF tasks. Unlike images or speech signals, TS data represent real-world processes in a highly abstract numerical form, often lacking explicit semantic grounding in natural language. As a result, the correspondence between TS and textual modalities is inherently ambiguous, making it difficult to directly transfer pretrained knowledge from LLMs to TSF. At the same time, generalization, especially under cross-domain settings, is a fundamental requirement of practical TSF systems. In many applications, the target domain may differ from the source domain in terms of data distributions, temporal patterns, or underlying dynamics, while labeled TS data in the target domain are often limited or unavailable. Models trained with strong domain-specific inductive biases therefore tend to suffer significant performance degradation when deployed across domains. In this context, LLMs offer a promising opportunity for cross-domain TSF due to their strong generalization abilities.

B Analysis

(1) Cross-dataset learning is a crucial prerequisite for unlocking the full potential of LLMs. Compared to small-scale training on a single dataset, diverse cross-dataset learning within a unified framework more effectively exploits the capabilities of LLMs. This strategy not only alleviates overfitting but also leads to stronger overall performance. Specifically, it surpasses single-dataset baselines on in-domain tasks and consistently outperforms a range of strong time-series foundation models as well as LLM-based approaches in out-of-domain evaluations, with the performance gains becoming pronounced as the diversity and scale of training data grow, as shown in Figure 15.

(2) Pre-alignment provides a more effective integration strategy for LLM4TSF. We observe that aligning TS inputs with word embeddings before feeding them into LLMs yields lower forecasting errors than performing alignment between text and TS representations within the LLM space. This finding suggests that pre-alignment enables a more compatible input representation, leading to more effective utilization of the pretrained LLM parameters.

(3) The advantages of LLM4TSF arise from both pretrained knowledge and architectural modeling capacity. Our analysis shows that the performance improvements of LLM4TSF models stem from two complementary sources. Pretrained parameters endow the model with rich prior knowledge, while the expressive Transformer-based architecture provides strong sequence modeling capacity. Together, these factors enable LLM4TSF to achieve strong performance across a wide range of forecasting scenarios.

(4) LLM4TSF exhibit inherent preferences toward certain TS properties. On the one hand, pretrained knowledge provides strong priors that are particularly beneficial when data distributions shift substantially over time or when models are evaluated on previously unseen out-of-domain datasets. On the other hand, the expressive architecture of LLMs offers strong modeling capacity, enabling them to better capture complex temporal dynamics characterized by frequent or abrupt transitions in underlying patterns. In contrast, factors such as stationarity, seasonality, or trends are not the primary driver, suggesting that simpler models may be sufficient when such characteristics dominate.

(5) Both a complete architecture and sufficient parameter optimization are essential for achieving strong performance. Our results show that full-parameter fine-tuning consistently outperforms parameter-efficient alternatives such as LoRA or partial adaptation strategies (e.g., positional encoding and layer normalization tuning). Moreover, artificially truncating the LLM by retaining only its shallow layers leads to noticeable performance degradation, as it undermines the model's ability to fully exploit its architectural depth and pretrained capacity, as shown in the Figure 16.

(6) The routing mechanism provides direct evidence for the observed macroscopic performance. We not only observe performance gaps in overall MAE/MSE, but also gain fine-grained insight into model decision-making. Empirically, the model prefers paths that yield lower prediction errors. This token-level routing preference closely aligns with macroscopic performance trends, offering direct micro-level evidence for when and why incorporating LLMs leads to performance gains.

(7) Blindly scaling up LLM backbones does not necessarily lead to better performance. Our experiments reveal that simply replacing the backbone with a larger LLM does not consistently yield performance improvements. One possible reason is the inherent modality gap between natural language and TS data, which makes direct alignment of increasingly large LLMs with TSF tasks more challenging. In addition, prior studies have shown that large-scale TS models often contain substantial internal redundancy (Qiu et al., 2025; Wiliński et al., 2025), which may limit the practical benefits of naive model scaling.

C Routing Intervention

We perform explicit routing interventions to examine whether the learned routing decisions directly affect forecasting performance. For high-shifting sub-sequences (> 0.2), we force them to pass through the LLM pathway regardless of the original routing decision. Conversely, for low-shifting sub-sequences (< 0.1), we suppress the LLM pathway and force them to bypass the LLM.

As shown in Tables 5 and 6, forcing high-shifting sub-sequences through the LLM generally improves performance, with larger gains on datasets such as Covid and NASDAQ. In contrast, bypassing the LLM for low-shifting sub-sequences largely preserves performance and can even yield slight improvements on several datasets. This behavior supports that the routing mechanism selectively activates the LLM according to temporal dynamics.

D Text Prompt

ETTh1 prompt. The ETTh1 designed for time-series forecasting at 1-hour intervals, contains data points with the target variable "oil temperature" and six power load features. Given the past 512 observations, predict the next 96 time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

ETTh2 prompt. The ETTh2 designed for time-series forecasting at 1-hour intervals, contains data points with the target variable "oil temperature" and six power load features. Given the past 512 observations, predict the next 96 time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

ETTm1 prompt. The ETTm1 designed for time-series forecasting at 15-minutes intervals, contains data points with the target variable "oil temperature" and six power load features. Given the past 512 observations, predict the next 96 time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

ETTm2 prompt. The ETTm2 designed for time-series forecasting at 15-minutes intervals, contains data points with the target variable "oil temperature" and six power load features. Given the

Table 5: Routing intervention for high-shifting sub-sequences (> 0.2), which are forced to pass through the LLM pathway.

Dataset	Original MSE	Intervention MSE
ID		
ETTh1	0.447	0.440
ETTh2	0.348	0.334
ETTm1	0.353	0.347
ETTm2	0.252	0.238
Weather	0.225	0.212
Traffic	0.401	0.393
Exchange	0.332	0.320
Covid	1.383	1.214
ECL	0.168	0.161
NN	0.804	0.755
OOD		
Wind	1.015	0.936
Solar	0.228	0.202
AQShunyi	0.612	0.604
CzenLan	0.274	0.253
ZafNoo	0.547	0.518
NASDAQ	0.735	0.642
PEMS	0.256	0.259

Table 6: Routing intervention for low-shifting sub-sequences (< 0.1), which are forced to skip the LLM pathway.

Dataset	Original MSE	Intervention MSE
ID		
ETTh1	0.447	0.449
ETTh2	0.348	0.342
ETTm1	0.353	0.357
ETTm2	0.252	0.242
Weather	0.225	0.221
Traffic	0.401	0.395
Exchange	0.332	0.325
Covid	1.383	1.361
ECL	0.168	0.162
NN	0.804	0.789
OOD		
Wind	1.015	0.977
Solar	0.228	0.215
AQShunyi	0.612	0.615
CzenLan	0.274	0.266
ZafNoo	0.547	0.532
NASDAQ	0.735	0.683
PEMS	0.256	0.258

past 512 observations, predict the next 96 time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

Weather prompt. The Weather dataset is designed for time-series forecasting with data recorded every 10 minutes and contains 21 meteorological indicators, such as air temperature, humidity, and wind-related variables. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

Traffic prompt. The Traffic dataset describes road occupancy conditions and contains hourly measurements collected from highway sensors. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

Exchange prompt. The Exchange dataset contains daily exchange rate data for eight countries and is commonly used for time-series analysis and forecasting of currency fluctuations. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

ECL prompt. The ECL dataset represents the hourly electricity consumption, recorded in kilowatts. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

NN prompt. The NN dataset consists of 111 daily time series drawn from a homogeneous population of empirical cash demand data. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

Wind prompt. The Wind dataset consists of time-series data recorded at 15-minute intervals, and includes variables such as wind speed, wind direction, temperature, pressure, humidity, and wind power. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

Solar prompt. The Solar dataset contains 137 time series representing solar power production, recorded at 10-minute intervals. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

AQShunyi prompt. The AQShunyi dataset includes 11 hourly time series capturing air quality and meteorological conditions in the Shunyi District of Beijing, with variables such as PM2.5, PM10, SO_2, NO_2, CO, O_3, temperature, air pressure, humidity, wind speed, and precipitation. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

Czelan prompt. The Czelan dataset contains eight time series of plant sap flow measurements recorded at half-hour intervals. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

ZafNoo prompt. The ZafNoo dataset consists of 11 time series of plant sap flow measurements, recorded at half-hour intervals. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

PEMS prompt. The PEMS dataset includes 48 months of hourly data describing road occupancy rates measured by various traffic sensors. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

NASDAQ prompt. The NASDAQ dataset encompasses companies listed on the NASDAQ stock exchange, with regularly updated time-series data excluding test listings. Given the past input length observations, predict the next prediction length time steps. The input window includes a minimum value of {min value}, a maximum value of {max value}, and a median value of {median value}.

E BASELINES

Chronos (Ansari et al., 2024) reformulates time series forecasting for Transformer-based architectures via a two-stage pre-processing strategy consisting of normalization and discretization. Specifically, each observation is normalized by the mean absolute value of its historical window to ensure scale consistency across different series. The normalized values are then quantized into a finite set of bins, converting the continuous time series into discrete token sequences. These tokens are subsequently modeled using a T5-style Transformer trained with a cross-entropy objective, allowing the model to learn generalizable TS representations while fully exploiting the strengths of sequence modeling frameworks.

UniTS (Gao et al., 2024) is a unified time series foundation model that supports a universal task formulation, enabling a wide range of TS tasks including forecasting, classification, imputation, and anomaly detection within a single framework. This capability is realized through a unified network backbone that integrates sequence-wise and variable-wise attention mechanisms together with a dynamic linear operator, allowing the model to flexibly capture temporal dependencies and inter-variable relationships. The entire architecture is trained end-to-end as a single unified model across tasks. Extensive experiments conducted on 38 datasets spanning multiple domains demonstrate that UniTS consistently outperforms specialized task-specific models as well as repurposed natural language-based LLMs.

Moirai (Woo et al., 2024a) is a large-scale foundation model for TSF that departs from the conventional practice of training separate models for individual datasets. Through architectural enhancements to the standard Transformer, Moirai enables effective cross-frequency learning, accommodates arbitrary multivariate input dimensionalities, and adapts to heterogeneous data distributions across diverse domains. The model is trained on the Large-scale Open Time Series Archive, comprising more than 27 billion observations spanning nine domains. Owing to its large-scale pre-training, Moirai exhibits strong zero-shot forecasting performance, often matching or outperforming models that are fully fine-tuned on target datasets.

UniTime (Liu et al., 2024b) is a unified foundation model for time series analysis that supports multiple tasks under a single modeling framework, including forecasting, classification, imputation, and anomaly detection. It introduces a unified task formulation together with a shared backbone architecture, allowing different TS tasks to be jointly learned without task-specific redesign. By leveraging a task-agnostic representation and a unified training objective, UniTime achieves strong generalization across tasks and datasets. Extensive evaluations demonstrate that UniTime consistently outperforms task-specific baselines and exhibits robust zero-shot and few-shot performance when transferred to unseen datasets and tasks, highlighting its effectiveness as a general-purpose time series model.

Time-LLM (Jin et al., 2023) performs time series forecasting by reusing a pretrained large language model whose parameters remain entirely frozen. To adapt the model to TS inputs and outputs, it introduces two lightweight trainable modules: Patch Reprogramming, which converts TS segments into LLM-compatible representations, and Output Projection, which maps the model outputs to the forecasting space. In addition, the method follows a channel-independent formulation, decomposing multivariate forecasting problems into multiple parallel univariate prediction tasks, thereby enabling efficient adaptation without modifying the backbone LLM.

To ensure fair comparisons, we control the training and evaluation settings across different baselines. It is worth noting that several time series foundation models are pretrained with substantially larger computational and data resources than our models. For example, Chronos and Moirai contain approximately 200M and 300M parameters and are pretrained on roughly 200B and 230B observations, respectively. In contrast, our Pre-align and Post-align models contain about 126M and 124M parameters and are pretrained on approximately 8B observations. Moreover, UniTime and Time-LLM adopt a 5% few-shot setting, where a subset of target-domain training data is used for adaptation, providing additional task-specific supervision compared with our Pre-align and Post-align evaluation settings. The model configurations and pretraining scales are summarized in Table 7.

Table 7: Comparison of model scale and pretraining data.

Model	Layers	Hidden Dim.	Params.	Training Data
Chronos	24	1,024	200M	∼200B
Moirai	24	1,024	300M	∼230B
Pre-align	12	678	126M	∼8B
Post-align	12	678	124M	∼8B

F Datasets

F.1 General Overview

The training datasets cover a wide range of real-world domains, as shown in Figure 12, including web, transportation, energy, nature, environment, climate, sales, economics, healthcare, and industry. Such broad domain coverage introduces substantial diversity in temporal patterns, scales, seasonalities, and noise characteristics, which is crucial for learning transferable representations and improving generalization in cross-domain TSF settings. In addition, the datasets used for in-domain and out-of-domain tests are listed separately, as summarized in Table 8. Among them, 10 datasets, including ETT (four subsets), Weather, Traffic, Exchange, Covid, ECL, and NN, are further split into training, validation, and test subsets. The held-out test splits of these datasets are used for in-domain evaluation. For the ETT datasets, we follow the standard train–validation–test split ratio of 6:2:2, while all other datasets in set A are split using a 7:1:2 ratio. Set B contains 7 datasets, namely Wind, Solar, AQShunyi, CzenLan, ZafNoo, NASDAQ, and PEMS. Set B are completely excluded from the training process and are used solely for out-of-domain test. Some of the collected datasets contain N/A or invalid values. To ensure data quality and training stability, we apply linear interpolation to impute missing values in all datasets.

To ensure reproducibility and avoid the use of restricted data, all datasets used in this study are collected from publicly available sources, as shown in Table 9. We do not use any private, proprietary, or personally collected data. Our dataset collection is built upon widely used benchmarks and

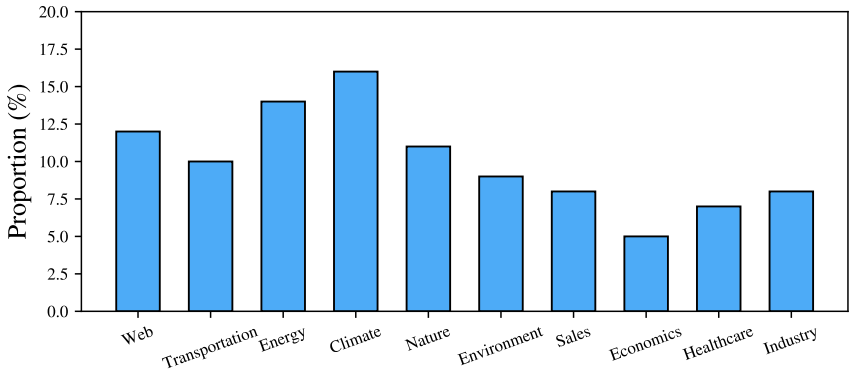

Figure 12: Proportional distribution of training datasets across ten real-world domains.

Table 8: Summary of datasets used in experiments. The table reports the number of variables and timestamps for each time series dataset.

Dataset	Variables	Timestamps	Dataset	Variables	Timestamps
ETTh1	7	14,400	Wind	7	48,673
ETTh2	7	14,400	Solar	137	52,560
ETTm1	7	57,600	AQShunyi	11	35,064
ETTm2	7	57,600	CzenLan	11	19,934
Weather	21	52,696	ZafNoo	11	19,225
Traffic	862	17,544	NASDAQ	5	1,244
Exchange	8	7,588	PEMS	170	17,856
Covid	948	1,392			
ECL	321	26,304			
NN	111	791			

publicly released datasets from prior work, covering a broad range of real-world domains, including web systems, transportation, energy, climate, environment, economics, healthcare, and industry.

F.2 Statistical Properties of Dataset

We visualize the statistical properties of all datasets used in testing, as shown in the Figure 13. Shifting describes how the activation regions of the TS evolve along the temporal axis, capturing changes in distributional structure over time. Stationarity reflects whether the statistical properties of a TS remain stable over time, indicating the presence or absence of distributional drift. Transition characterizes the sequential dependency between symbolic states, reflecting the complexity of temporal transitions and local dynamical behavior. Seasonality measures the strength of recurring temporal patterns at fixed intervals, which are commonly observed in real-world periodic processes. Trend characterizes the long-term directional movement of the series, revealing persistent growth or decline patterns. The corresponding computation procedures are summarized in Table 10–14.

Table 9: Summary of public datasets used in this study.

Domain	Datasets
Web	Wiki2020, Web Traffic, Wiki-Rolling, Azure VM, Alibaba Cluster Trace
Transportation	PEMS, Traffic, LargeST, Q-Traffic, Uber TLC, Rideshare, Vehicle Trips, Beijing Subway, SHMetro
Energy	ETTh1, ETTh2, ETTm1, ETTm2, ECL, Wind, Solar, BDG-2, London Smart Meters
Climate	Weather, WeatherBench, Subseasonal, Temperature Rain, Oikolab Weather
Nature	CzenLan, ZafNoo, Air Quality, Saugeen River Flow, Sunspot, US Births
Environment	AQShunyi, AQWan, EDGAR Emissions, GHAP PM2.5, Air Quality (UCI)
Sales	NN, Hierarchical Sales, Favorita Sales, Restaurant
Economics	Exchange, NASDAQ, GoDaddy, Yahoo Finance, NYSE, World Bank Indicators, S&P 500
Healthcare	Covid, CDC Fluview, WHO NREVSS, Project Tycho, MIMIC-III, MIT-BIH Arrhythmia
Industry	SWaT, WADI, C-MAPSS, SECOM, Bosch, Paderborn

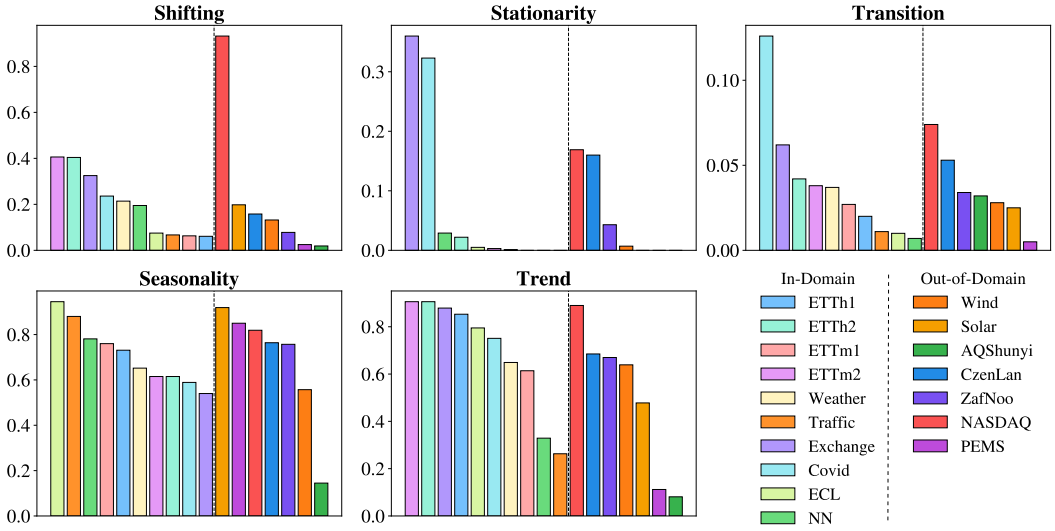

Figure 13: Analysis of the 17 in-domain and out-of-domain datasets from Shifting, Stationarity, Transition, Seasonality, and Trend.

Table 10: Algorithm: Shifting Computation

Input: Time series $\mathbf{X} \in \mathbb{R}^{T\times 1}$
Output: Shifting $\delta \in (0, 1)$

Normalize $\mathbf{X}$ using z-score normalization to obtain $\mathbf{Z} \in \mathbb{R}^{T\times 1}$.
Compute $Z_{\min} = \min(\mathbf{Z})$ and $Z_{\max} = \max(\mathbf{Z})$.
Construct m uniformly spaced value levels:
$\ell_i = Z_{\min} + \frac{i-1}{m-1}(Z_{\max} - Z_{\min}), \quad i = 1, \ldots, m.$
for each level ℓ_i **do**
  Identify activated time indices $\mathcal{T}_i = \{\, t \mid Z_t > \ell_i,\ 1 \le t \le T \,\}$.
  Compute the temporal center $c_i = \mathrm{median}(\mathcal{T}_i)$.
Apply min–max normalization to $\{c_i\}_{i=1}^m$ to obtain $\{\tilde{c}_i\}_{i=1}^m$.
return $\delta = |\mathrm{median}(\{\tilde{c}_1, \tilde{c}_2, \ldots, \tilde{c}_m\})|$.

Table 11: Algorithm 2: Stationarity Computation

Input: Time series $\mathbf{X} = \langle x_1, x_2, \ldots, x_T \rangle \in \mathbb{R}^{T \times 1}$
Output: Stationarity indicator $\gamma \in \{0, 1\}$ of $\mathbf{X}$

1: Compute the Augmented Dickey–Fuller (ADF) statistic $s \leftarrow \mathrm{ADF}(\mathbf{X})$.
2: **return** $\gamma = \begin{cases} 1, & \text{if } s \leq 0.05, \\ 0, & \text{otherwise.} \end{cases}$

Table 12: Algorithm: Transition Computation

Input: Time series $\mathbf{X} \in \mathbb{R}^{T \times 1}$
Output: Transition $\Delta \in \left(0, \frac{1}{3}\right)$

1: Estimate the characteristic lag τ as the first zero-crossing point of the autocorrelation function of $\mathbf{X}$.
2: Downsample $\mathbf{X}$ with stride τ to obtain a reduced sequence $\mathbf{Y} \in \mathbb{R}^{T' \times 1}$.
3: Obtain the rank ordering of $\mathbf{Y}$ by computing the permutation index $\mathbf{r} = \mathrm{argsort}(\mathbf{Y})$.
4: Discretize $\mathbf{Y}$ into a symbolic sequence $\mathbf{Z} \in \{0, 1, 2\}^{T'}$ via $Z_j = \left\lfloor \frac{3\, r_j}{T'} \right\rfloor, \quad j = 1, \ldots, T'$.
5: Initialize a transition count matrix $\mathbf{M} \in \mathbb{R}^{3 \times 3}$ with zeros.
6: **for** $j = 1$ **to** $T' - 1$ **do**
    Increment the transition count $M_{Z_j, Z_{j+1}} \leftarrow M_{Z_j, Z_{j+1}} + 1$.
7: Normalize the transition matrix by sequence length: $\mathbf{M}' = \frac{1}{T'}\mathbf{M}$.
8: Compute the covariance matrix $\mathbf{C}$ of the column vectors of $\mathbf{M}'$.
9: **return** $\Delta = \mathrm{tr}(\mathbf{C})$.

Table 13: Algorithm: Seasonality Computation

Input: Time series $\mathbf{X} = \{x_1, x_2, \ldots, x_T\} \in \mathbb{R}^{T \times 1}$
Output: Seasonality value $\zeta \in (0, 1)$

1: Decompose $\mathbf{X}$ into seasonal, trend, and residual components using STL: $\mathbf{X} = \mathbf{S} + \mathbf{T} + \mathbf{R}$.
2: Compute the variance of the residual component $\mathbf{R}$.
3: Compute the variance of the combined seasonal and residual component $\mathbf{S} + \mathbf{R}$.
4: **return** $\zeta = \max\left(0,\ 1 - \frac{\mathrm{var}(\mathbf{R})}{\mathrm{var}(\mathbf{S}+\mathbf{R})}\right)$.

Table 14: Algorithm: Trend Computation

Input: Time series $\mathbf{X} = \{x_1, x_2, \ldots, x_T\} \in \mathbb{R}^{T \times 1}$
Output: Trend value $\beta \in (0, 1)$

1: Decompose $\mathbf{X}$ into seasonal, trend, and residual components using STL: $\mathbf{X} = \mathbf{S} + \mathbf{T} + \mathbf{R}$.
2: Compute the variance of the residual component $\mathbf{R}$.
3: Compute the variance of the combined non-seasonal component $\mathbf{T} + \mathbf{R}$.
4: **return** $\beta = \max\left(0,\ 1 - \frac{\mathrm{var}(\mathbf{R})}{\mathrm{var}(\mathbf{T}+\mathbf{R})}\right)$.

G Distribution-space Metrics and Theoretical Analysis

G.1 Distribution-space Metrics

To analyze whether time series embeddings can be effectively received and processed by a pretrained LLM, we introduce two distribution-space metrics: Normalized Subspace Projection Error (NSPE) (Mu & Viswanath, 2018; Ethayarajh, 2019) and Fréchet Distance (FD) (Heusel et al., 2017; Dowson & Landau, 1982). The motivation is that the input embedding space of a pretrained LLM is not an arbitrary high-dimensional space. Instead, its language embeddings are learned from large-scale language corpora and thus form a structured representation space with specific geometric and statistical properties.

Normalized Subspace Projection Error. NSPE measures how much of a time series embedding lies outside the principal subspace of the pretrained embedding space. Let $E \in \mathbb{R}^{|\mathcal{V}| \times d}$ denote the pretrained embedding matrix of the LLM, where $|\mathcal{V}|$ is the vocabulary size and d is the embedding dimension. We perform principal component analysis on E and use the top-k principal directions to approximate the dominant language subspace. Let $U_k \in \mathbb{R}^{d \times k}$ denote the orthonormal basis formed by these top-k principal directions. For a time series embedding $z \in \mathbb{R}^d$, NSPE is defined as:

$$\mathrm{NSPE}(z) = \frac{\|z - U_k U_k^\top z\|_2}{\|z\|_2}.$$

Here, $U_k U_k^\top z$ is the projection of z onto the principal language subspace, and $z - U_k U_k^\top z$ is the residual component that cannot be explained by this subspace. A smaller NSPE indicates that more of the time series embedding lies within the dominant subspace of pretrained language embeddings, suggesting stronger geometric compatibility with the LLM input space. Conversely, a larger NSPE indicates that the embedding contains more components outside the language subspace, implying a more severe modality mismatch.

Fréchet Distance. While NSPE measures the geometric deviation of individual embeddings from the principal language subspace, FD measures the distribution-level discrepancy between a set of time series embeddings and the pretrained language embedding distribution. Given two embedding distributions with means μ_1, μ_2 and covariance matrices Σ_1, Σ_2, FD is defined as:

$$\mathrm{FD} = \|\mu_1 - \mu_2\|_2^2 + \mathrm{Tr}\left(\Sigma_1 + \Sigma_2 - 2(\Sigma_1\Sigma_2)^{1/2}\right).$$

The first term measures the distance between the distribution centers, while the second term measures the discrepancy between their covariance structures. Therefore, a smaller FD indicates that the time series embeddings are closer to the pretrained language embeddings in terms of both global location and distributional shape. To directly compare pre-alignment and post-alignment, we further define Relative Fréchet Distance (RFD) as:

$$\mathrm{RFD} = \frac{\mathrm{FD}(Z_{\mathrm{pre}}, E)}{\mathrm{FD}(Z_{\mathrm{post}}, E)},$$

where Z_{pre} and Z_{post} denote the time series embeddings produced by pre-alignment and post-alignment, respectively. When $\mathrm{RFD} < 1$, pre-alignment yields a smaller Fréchet Distance to the pretrained language embedding distribution than post-alignment, indicating better distribution-level compatibility. When $\mathrm{RFD} > 1$, post-alignment is closer to the language embedding distribution.

G.2 Theoretical Analysis

We next provide a theoretical interpretation of the above metrics and connect them to the empirical observations in Sec. 4. Our analysis focuses on three questions: (i) why pre-alignment provides a more compatible input representation for pretrained LLMs; (ii) why stronger distribution shifting makes pretrained LLM parameters more likely to provide meaningful benefits; and (iii) why, under high transition, a trainable Transformer backbone can outperform the variant without the LLM module even when pretrained parameters are removed.

G.2.1 Why Pre-alignment Reduces the Modality Gap

Let $\mathcal{S}_E = \mathrm{span}(U_k)$ denote the dominant subspace of the pretrained language embedding matrix. Since U_k has orthonormal columns, $P_E = U_k U_k^\top$ is the orthogonal projection matrix onto $\mathcal{S}_E$. For

any time series embedding $z \in \mathbb{R}^d$, we decompose it as $z = z_{\|} + z_{\perp}, z_{\|} = P_E z, z_{\perp} = (I - P_E)z$. Here, $z_{\|}$ is the component lying in the principal language subspace, while $z_{\perp}$ is the component orthogonal to this subspace. By definition:

$$\mathrm{NSPE}(z) = \frac{\|(I - P_E)z\|_2}{\|z\|_2} = \frac{\|z_{\perp}\|_2}{\|z\|_2}.$$

Thus, NSPE directly measures the relative magnitude of the out-of-language-subspace component.

Proposition 1. Let z_{pre} and z_{post} be time series embeddings produced by pre-alignment and post-alignment, respectively. If $\mathrm{NSPE}(z_{\mathrm{pre}}) < \mathrm{NSPE}(z_{\mathrm{post}})$, then z_{pre} has a smaller relative residual outside the dominant language subspace than z_{post}.

Proof. Since $P_E = U_k U_k^{\top}$ is an orthogonal projection, every embedding z admits the orthogonal decomposition $z = P_E z + (I - P_E)z$. Therefore, $\mathrm{NSPE}(z) = \frac{\|(I-P_E)z\|_2}{\|z\|_2}$. Applying this identity to z_{pre} and z_{post}, the condition $\mathrm{NSPE}(z_{\mathrm{pre}}) < \mathrm{NSPE}(z_{\mathrm{post}})$ is equivalent to:

$$\frac{\|(I - P_E)z_{\mathrm{pre}}\|_2}{\|z_{\mathrm{pre}}\|_2} < \frac{\|(I - P_E)z_{\mathrm{post}}\|_2}{\|z_{\mathrm{post}}\|_2}.$$

Hence, the pre-aligned embedding contains a smaller relative component outside $\mathcal{S}_E$, indicating stronger geometric compatibility with the pretrained language embedding space.

Proposition 2. If $\mathrm{RFD} < 1$, then the embedding distribution induced by pre-alignment is closer to the pretrained language embedding distribution than that induced by post-alignment in terms of Fréchet Distance.

Proof. By definition, $\mathrm{RFD} = \frac{\mathrm{FD}(Z_{\mathrm{pre}}, E)}{\mathrm{FD}(Z_{\mathrm{post}}, E)}$. Since Fréchet Distance is non-negative and $\mathrm{FD}(Z_{\mathrm{post}}, E) > 0$ in non-degenerate cases, $\mathrm{RFD} < 1$ implies $\mathrm{FD}(Z_{\mathrm{pre}}, E) < \mathrm{FD}(Z_{\mathrm{post}}, E)$. Therefore, Z_{pre} has a smaller discrepancy from E in terms of both mean and covariance structure. This means that pre-alignment yields a distribution that is globally closer to the pretrained language embedding distribution.

Theorem 1. Assume that the dominant language subspace $\mathcal{S}_E$ captures the major geometric structure of the pretrained LLM input space. If $\mathbb{E}_{z \sim Z_{\mathrm{pre}}}[\mathrm{NSPE}(z)] < \mathbb{E}_{z \sim Z_{\mathrm{post}}}[\mathrm{NSPE}(z)]$ and $\mathrm{RFD} < 1$, then pre-alignment yields a smaller input-side modality gap than post-alignment.

Proof. The first inequality and Proposition 1 imply that, on average, embeddings from Z_{pre} have smaller relative components outside the dominant language subspace $\mathcal{S}_E$. Thus, pre-alignment reduces the local geometric mismatch between time series embeddings and the pretrained language embedding space. Meanwhile, $\mathrm{RFD} < 1$ and Proposition 2 imply $\mathrm{FD}(Z_{\mathrm{pre}}, E) < \mathrm{FD}(Z_{\mathrm{post}}, E)$, which means that pre-alignment also reduces the global distributional mismatch in terms of the first- and second-order statistics of the embedding distributions.

Combining the two results, pre-alignment simultaneously reduces the local subspace residual and the global distribution discrepancy before the time series embeddings enter the LLM backbone. Therefore, it provides a more compatible input representation for the pretrained LLM than post-alignment.

G.2.2 Why Stronger Shifting Increases the Value of Pretrained LLM Parameters

We now analyze the role of distribution shifting. Let $\delta \geq 0$ denote the strength of distribution shifting in the time series data, and let Z_δ^0 denote the time series embedding distribution before entering the LLM backbone under shifting level δ.

Assumption 1. Shifting increases the initial modality gap. There exists a non-decreasing function $m(\delta)$ such that $\mathbb{E}_{z \sim Z_\delta^0}[\mathrm{NSPE}(z)] \geq m(\delta)$, where $m(\delta_2) \geq m(\delta_1)$ whenever $\delta_2 > \delta_1$. This assumption formalizes the intuition that stronger shifting makes time series embeddings deviate more from the pretrained language embedding space before LLM processing.

Let T_{pt} denote the transformation induced by a pretrained LLM backbone, and let T_{rand} denote the transformation induced by a randomly initialized or non-pretrained backbone. Define the NSPE reduction induced by a transformation T as:

$$\Delta_T(\delta) = \mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(z)] - \mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(T(z))].$$

Assumption 2. Pretrained LLMs reduce the modality gap more effectively under strong shifting. For sufficiently large δ, the pretrained backbone induces a larger NSPE reduction than the non-pretrained backbone:

$$\Delta_{T_{\mathrm{pt}}}(\delta) > \Delta_{T_{\mathrm{rand}}}(\delta).$$

This assumption captures the role of pretrained LLM parameters as a structured representation map. Since pretrained LLMs are optimized over large-scale language distributions, their layers encode inductive biases and stable geometric transformations that can better regularize mismatched inputs toward the pretrained representation space.

Proposition 3. Under Assumption 1, stronger shifting increases the lower bound of the initial modality gap.

Proof. For two shifting levels $\delta_2 > \delta_1$, Assumption 1 gives $m(\delta_2) \geq m(\delta_1)$. Since $\mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(z)] \geq m(\delta)$, we have $\mathbb{E}_{z\sim Z_{\delta_2}^0}[\mathrm{NSPE}(z)] \geq m(\delta_2) \geq m(\delta_1)$. Thus, the lower bound of the initial NSPE becomes larger as shifting increases. This indicates that stronger shifting creates a larger initial mismatch between time series embeddings and the pretrained language space.

Theorem 2. Under Assumptions 1 and 2, when shifting is strong, pretrained LLM parameters are more likely to provide meaningful performance benefits by reducing the modality gap.

Proof. By definition,

$$\Delta_{T_{\mathrm{pt}}}(\delta) = \mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(z)] - \mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(T_{\mathrm{pt}}(z))]$$

and

$$\Delta_{T_{\mathrm{rand}}}(\delta) = \mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(z)] - \mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(T_{\mathrm{rand}}(z))].$$

For sufficiently large δ, Assumption 2 gives $\Delta_{T_{\mathrm{pt}}}(\delta) > \Delta_{T_{\mathrm{rand}}}(\delta)$. Therefore,

$$\mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(T_{\mathrm{pt}}(z))] < \mathbb{E}_{z\sim Z_\delta^0}[\mathrm{NSPE}(T_{\mathrm{rand}}(z))].$$

This shows that the pretrained LLM backbone maps the shifted time series embeddings closer to the dominant language subspace than the non-pretrained backbone. Moreover, by Proposition 3, stronger shifting increases the initial modality gap. Therefore, under strong shifting, there is a larger mismatch to be corrected, and the stronger gap-reduction ability of pretrained LLM parameters becomes more useful. If the downstream loss is locally non-decreasing with respect to the modality gap, i.e.,

$$\mathcal{R}(T) = \mathbb{E}[\ell(f_T(u), y)] \quad \text{satisfies} \quad \mathcal{R}(T_1) \leq \mathcal{R}(T_2) \text{ when } \mathbb{E}[\mathrm{NSPE}(T_1(z))] \leq \mathbb{E}[\mathrm{NSPE}(T_2(z))],$$

then the smaller post-backbone NSPE induced by T_{pt} implies a lower or equal expected risk than T_{rand}. Hence, pretrained LLM parameters are more likely to provide meaningful performance benefits when shifting is strong.

G.2.3 Why High Transition Favors a Trainable Transformer Backbone

We finally analyze why, under high transition, a trainable Transformer backbone can outperform the variant without the LLM module even when pretrained parameters are removed. Let $\tau \geq 0$ denote the transition strength of a time series. High transition indicates frequent or abrupt changes among latent temporal regimes. Let u denote the historical input window and y the forecasting target.

Assumption 3. High transition induces regime-dependent dynamics. The forecasting target is generated by $y = F_\tau(u) + \epsilon$, where ϵ is zero-mean noise. When τ is large, F_τ can be decomposed into multiple regime-dependent maps:

$$F_\tau(u) = \sum_{r=1}^{M} \pi_r(u) F_r(u),$$

where $\pi_r(u) \geq 0$, $\sum_{r=1}^{M} \pi_r(u) = 1$, and each F_r corresponds to a latent temporal regime. Larger τ corresponds to more frequent regime changes or sharper differences among the regime-specific maps.

Let $\mathcal{F}_{\mathrm{no}}$ denote the function class of the variant without the LLM module, and let $\mathcal{F}_{\mathrm{tr}}$ denote the function class of the model with a trainable Transformer backbone but without pretrained parameters.

Assumption 4. Function class inclusion. The trainable Transformer backbone has a function class that contains the w/o LLM variant: $\mathcal{F}_{\mathrm{no}} \subseteq \mathcal{F}_{\mathrm{tr}}$. This is reasonable because the w/o LLM variant can be viewed as a restricted case where the additional Transformer transformations are removed or degenerated to identity mappings.

Proposition 4. If $\mathcal{F}_{\mathrm{no}} \subseteq \mathcal{F}_{\mathrm{tr}}$, then the optimal empirical risk of the trainable Transformer function class is no larger than that of the w/o LLM function class.

Proof. Let the empirical risk on a dataset $\mathcal{D}$ be

$$\widehat{\mathcal{R}}(f) = \frac{1}{|\mathcal{D}|} \sum_{(u,y)\in\mathcal{D}} \ell(f(u), y).$$

Since $\mathcal{F}_{\mathrm{no}} \subseteq \mathcal{F}_{\mathrm{tr}}$, minimizing over $\mathcal{F}_{\mathrm{tr}}$ optimizes over a superset of the functions available to $\mathcal{F}_{\mathrm{no}}$. Hence,

$$\min_{f\in\mathcal{F}_{\mathrm{tr}}} \widehat{\mathcal{R}}(f) \leq \min_{f\in\mathcal{F}_{\mathrm{no}}} \widehat{\mathcal{R}}(f).$$

Assumption 5. High transition creates approximation error for the w/o LLM variant. For high τ, the w/o LLM function class has a non-negligible approximation error:

$$\inf_{f\in\mathcal{F}_{\mathrm{no}}} \mathbb{E}[\ell(f(u), y)] - \mathbb{E}[\ell(F_\tau(u), y)] \geq \epsilon_{\mathrm{no}}(\tau),$$

where $\epsilon_{\mathrm{no}}(\tau)$ is non-decreasing with respect to τ.

Assumption 6. A trainable Transformer reduces the approximation error under high transition. For high τ, the trainable Transformer function class can better approximate regime-dependent dynamics: $\epsilon_{\mathrm{tr}}(\tau) < \epsilon_{\mathrm{no}}(\tau)$, where

$$\epsilon_{\mathrm{tr}}(\tau) = \inf_{f\in\mathcal{F}_{\mathrm{tr}}} \mathbb{E}[\ell(f(u), y)] - \mathbb{E}[\ell(F_\tau(u), y)].$$

Theorem 3. Under Assumptions 3–6, when transition is high, a trainable Transformer backbone, even without pretrained LLM parameters, can outperform the w/o LLM variant.

Proof. By Assumption 3, high transition induces regime-dependent forecasting dynamics:

$$F_\tau(u) = \sum_{r=1}^{M} \pi_r(u) F_r(u).$$

Such dynamics require the model to infer input-dependent regime weights $\pi_r(u)$ and apply different regime-specific mappings F_r. This increases the complexity of the target function as τ grows.

By Assumption 5, the w/o LLM function class incurs approximation error at least $\epsilon_{\mathrm{no}}(\tau)$ under high transition:

$$\inf_{f\in\mathcal{F}_{\mathrm{no}}} \mathbb{E}[\ell(f(u), y)] \geq \mathbb{E}[\ell(F_\tau(u), y)] + \epsilon_{\mathrm{no}}(\tau).$$

By Assumption 6, the trainable Transformer function class has a smaller approximation error:

$$\inf_{f\in\mathcal{F}_{\mathrm{tr}}} \mathbb{E}[\ell(f(u), y)] = \mathbb{E}[\ell(F_\tau(u), y)] + \epsilon_{\mathrm{tr}}(\tau),$$

with $\epsilon_{\mathrm{tr}}(\tau) < \epsilon_{\mathrm{no}}(\tau)$. Therefore,

$$\inf_{f\in\mathcal{F}_{\mathrm{tr}}} \mathbb{E}[\ell(f(u), y)] < \inf_{f\in\mathcal{F}_{\mathrm{no}}} \mathbb{E}[\ell(f(u), y)].$$

This shows that, at the population-risk level, the trainable Transformer backbone has a lower achievable risk under high transition.

Proposition 4 shows that the trainable Transformer function class also has no larger optimal empirical risk than the w/o LLM function class. Hence, when optimization is sufficiently effective and the additional capacity does not introduce excessive generalization error, the trainable Transformer backbone can outperform the w/o LLM variant even without pretrained LLM parameters.

Summary. The above analysis provides a distribution-space explanation for our empirical findings. First, NSPE and RFD show that pre-alignment reduces both local subspace mismatch and global distribution mismatch before the embeddings enter the LLM. Second, stronger shifting enlarges the initial modality gap, making the structured representation space of pretrained LLM parameters more useful for reducing this gap. Third, high transition induces more complex regime-dependent dynamics, under which a trainable Transformer backbone provides a richer function class than the w/o LLM variant and can yield better performance even without pretrained parameters.

H More Results

H.1 Single- and Cross-Dataset Evaluation Results

We report the MAE and MSE results under both single-dataset and cross-dataset learning strategies, evaluated with two alignment schemes across four forecasting horizons {96, 192, 336, and 720}, as summarized in Table 15 and Table 16.

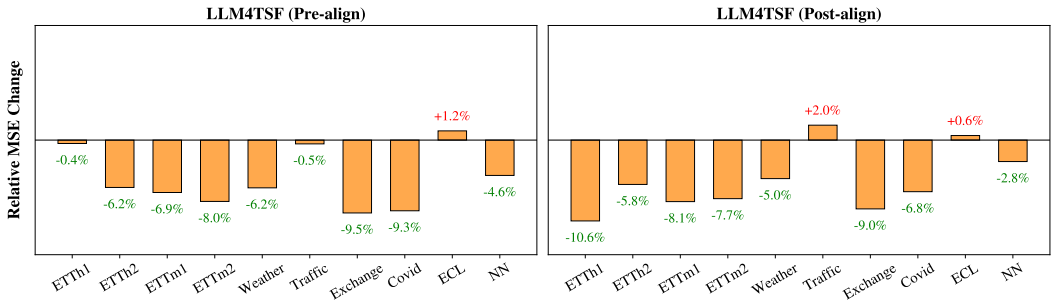

Figure 14: Comparison of LLM4TSF performance with pre- and post-alignment under single- and cross-dataset paradigm. Negative and Positive values indicate MSE decreases and increases under cross-dataset learning compared to single-dataset learning.

H.2 Effect of Training Data in Cross-Dataset Learning

During cross-dataset learning, we vary the ratio of training data and gradually increase the available data volume. We observe that both MSE and MAE consistently decrease as the training data ratio increases, indicating a strong correlation between model performance and the amount of training data. This trend suggests that cross-dataset forecasting benefits substantially from larger and more diverse training samples. Results are shown in Figure 15.

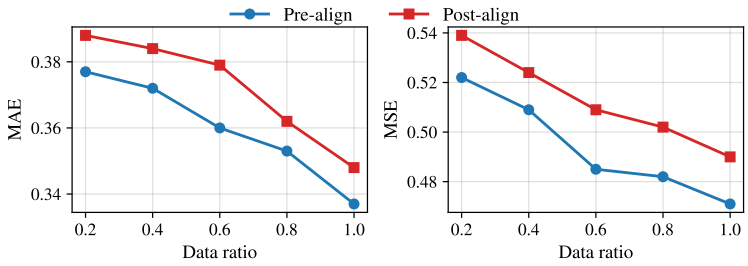

Figure 15: Effect of training data on cross-dataset learning performance.

Table 15: MAE comparison under different settings (Error bars over five random seeds).

Dataset	Horizon	Pre-Single	Pre-Cross	Post-Single	Post-Cross
ETTh1	96	0.407±0.002	0.414±0.004	0.421±0.001	0.398±0.003
	192	0.435±0.005	0.437±0.000	0.432±0.003	0.420±0.001
	336	0.452±0.001	0.448±0.006	0.462±0.002	0.446±0.000
	720	0.461±0.004	0.479±0.002	0.499±0.005	0.481±0.003
ETTh2	96	0.368±0.003	0.333±0.001	0.374±0.005	0.352±0.002
	192	0.399±0.001	0.374±0.000	0.416±0.004	0.379±0.006
	336	0.435±0.004	0.411±0.002	0.446±0.003	0.418±0.001
	720	0.459±0.005	0.435±0.003	0.458±0.002	0.435±0.000
ETTm1	96	0.342±0.001	0.355±0.004	0.374±0.002	0.345±0.003
	192	0.391±0.003	0.359±0.001	0.399±0.005	0.370±0.000
	336	0.417±0.006	0.377±0.002	0.428±0.004	0.396±0.001
	720	0.438±0.002	0.410±0.005	0.467±0.003	0.426±0.004
ETTm2	96	0.253±0.002	0.248±0.000	0.264±0.001	0.260±0.004
	192	0.297±0.003	0.277±0.002	0.313±0.006	0.300±0.000
	336	0.347±0.001	0.328±0.004	0.369±0.002	0.344±0.003
	720	0.392±0.005	0.382±0.001	0.402±0.004	0.396±0.002
Weather	96	0.226±0.001	0.197±0.003	0.221±0.000	0.203±0.002
	192	0.264±0.004	0.231±0.002	0.254±0.005	0.247±0.001
	336	0.286±0.000	0.269±0.006	0.295±0.003	0.281±0.002
	720	0.359±0.003	0.325±0.001	0.348±0.004	0.331±0.005
Traffic	96	0.254±0.002	0.241±0.000	0.271±0.004	0.276±0.001
	192	0.274±0.005	0.282±0.002	0.279±0.001	0.290±0.003
	336	0.282±0.001	0.288±0.006	0.277±0.000	0.295±0.004
	720	0.311±0.003	0.302±0.001	0.300±0.005	0.310±0.002
Exchange	96	0.225±0.000	0.213±0.002	0.231±0.004	0.202±0.001
	192	0.312±0.003	0.294±0.001	0.325±0.005	0.296±0.002
	336	0.415±0.006	0.393±0.004	0.432±0.002	0.410±0.003
	720	0.686±0.004	0.655±0.005	0.718±0.006	0.698±0.001
Covid	96	0.044±0.000	0.041±0.000	0.046±0.001	0.041±0.000
	192	0.049±0.001	0.045±0.000	0.052±0.002	0.048±0.001
	336	0.058±0.000	0.051±0.001	0.060±0.001	0.056±0.002
	720	0.061±0.001	0.057±0.002	0.064±0.000	0.059±0.001
ECL	96	0.233±0.003	0.228±0.000	0.237±0.001	0.239±0.004
	192	0.259±0.001	0.253±0.002	0.251±0.000	0.254±0.003
	336	0.274±0.004	0.280±0.001	0.267±0.002	0.269±0.000
	720	0.303±0.002	0.311±0.005	0.298±0.003	0.298±0.000
NN	96	0.613±0.006	0.584±0.002	0.641±0.004	0.622±0.003
	192	0.622±0.003	0.601±0.005	0.650±0.002	0.642±0.004
	336	0.640±0.001	0.623±0.004	0.677±0.006	0.663±0.002
	720	0.691±0.005	0.659±0.003	0.723±0.004	0.709±0.001

Table 16: MSE comparison under different settings (Error bars over five random seeds).

Dataset	Horizon	Pre-Single	Pre-Cross	Post-Single	Post-Cross
ETTh1	96	0.404±0.004	0.412±0.002	0.420±0.007	0.377±0.003
	192	0.435±0.006	0.439±0.003	0.455±0.004	0.416±0.008
	336	0.472±0.002	0.455±0.005	0.488±0.006	0.444±0.003
	720	0.485±0.007	0.481±0.004	0.566±0.008	0.488±0.005
ETTh2	96	0.303±0.003	0.279±0.006	0.307±0.002	0.283±0.004
	192	0.361±0.005	0.347±0.002	0.375±0.007	0.354±0.003
	336	0.396±0.008	0.372±0.004	0.409±0.005	0.389±0.006
	720	0.425±0.003	0.395±0.007	0.425±0.004	0.403±0.002
ETTm1	96	0.322±0.005	0.291±0.002	0.336±0.006	0.290±0.004
	192	0.355±0.003	0.332±0.007	0.363±0.004	0.331±0.002
	336	0.386±0.006	0.361±0.003	0.392±0.008	0.369±0.005
	720	0.454±0.004	0.426±0.006	0.449±0.002	0.427±0.007
ETTm2	96	0.171±0.002	0.165±0.004	0.179±0.003	0.169±0.006
	192	0.244±0.005	0.231±0.003	0.254±0.007	0.226±0.002
	336	0.310±0.004	0.274±0.008	0.327±0.005	0.288±0.003
	720	0.371±0.006	0.339±0.002	0.388±0.004	0.377±0.007
Weather	96	0.158±0.003	0.151±0.005	0.161±0.002	0.147±0.004
	192	0.209±0.006	0.193±0.003	0.195±0.008	0.185±0.002
	336	0.266±0.004	0.244±0.007	0.262±0.005	0.248±0.003
	720	0.325±0.008	0.311±0.004	0.335±0.006	0.322±0.005
Traffic	96	0.376±0.004	0.359±0.002	0.385±0.007	0.391±0.005
	192	0.391±0.006	0.403±0.003	0.404±0.005	0.413±0.008
	336	0.401±0.002	0.412±0.007	0.406±0.004	0.418±0.006
	720	0.443±0.005	0.429±0.004	0.443±0.002	0.451±0.007
Exchange	96	0.084±0.002	0.080±0.003	0.099±0.006	0.085±0.004
	192	0.185±0.005	0.172±0.002	0.201±0.007	0.156±0.003
	336	0.368±0.008	0.321±0.004	0.379±0.005	0.327±0.006
	720	0.829±0.006	0.754±0.005	1.010±0.008	0.969±0.007
Covid	96	1.032±0.004	1.011±0.006	1.057±0.003	1.023±0.005
	192	1.355±0.007	1.246±0.004	1.343±0.006	1.215±0.002
	336	1.689±0.005	1.318±0.008	1.657±0.004	1.518±0.006
	720	2.021±0.008	1.955±0.005	2.103±0.007	1.986±0.006
ECL	96	0.135±0.003	0.129±0.002	0.139±0.005	0.139±0.004
	192	0.148±0.006	0.142±0.003	0.155±0.002	0.156±0.007
	336	0.158±0.004	0.166±0.008	0.170±0.003	0.171±0.005
	720	0.221±0.007	0.234±0.005	0.207±0.002	0.208±0.004
NN	96	0.807±0.006	0.754±0.003	0.816±0.005	0.811±0.008
	192	0.824±0.004	0.786±0.007	0.833±0.002	0.829±0.006
	336	0.818±0.005	0.799±0.004	0.894±0.008	0.854±0.003
	720	0.921±0.007	0.877±0.006	1.016±0.005	0.965±0.008

H.3 Impact of Model Completeness

To examine the effect of model completeness, we compare a full LLM with a truncated variant that retains only the first 50% of layers. Both models are trained on the same data under identical settings and evaluated on in-domain and out-of-domain test sets, as shown in the Figure 16. The results indicate that truncating the model degrades performance, leading to higher MSE.

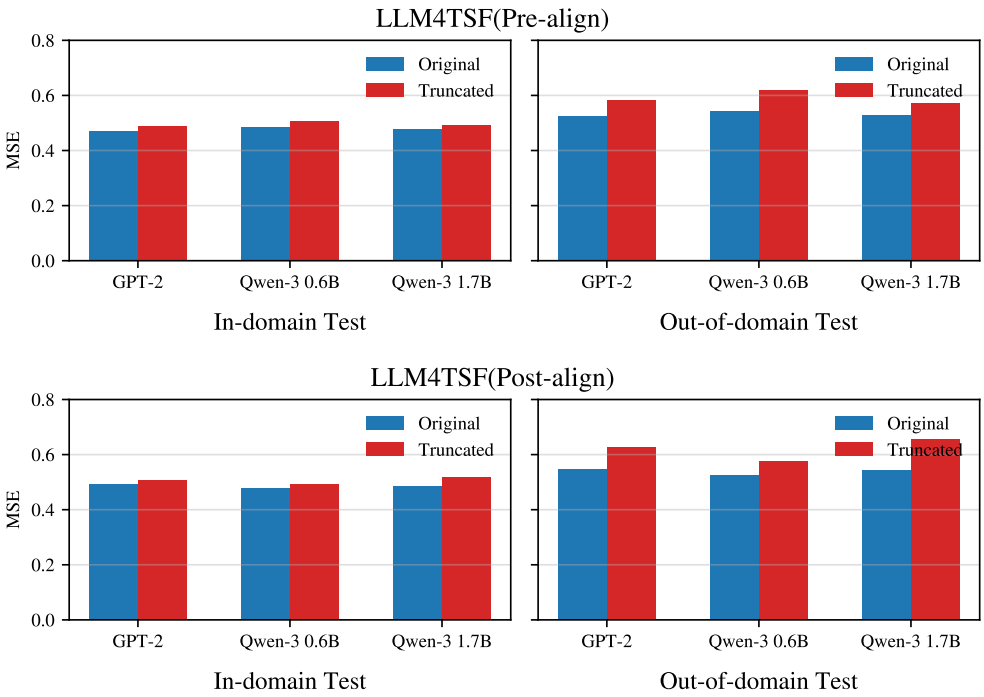

Figure 16: Effect of model truncation on forecasting performance.

H.4 Effect of LLM Backbone and Prompt

As shown in Table 17, replacing GPT-2 with stronger Qwen-3 backbones does not consistently improve forecasting performance, suggesting that larger LLM capacity alone is not the dominant factor. In contrast, removing prompts consistently increases MSE and reduces the proportion of tokens routed through the LLM, with the effect being particularly pronounced under OOD evaluation.

We further enrich the textual prompt with seasonality and trend descriptors computed solely from the observed input window, avoiding any future information leakage. As shown in Table 18, incorporating these temporal descriptors reduces forecasting error on most datasets across both prediction horizons, with particularly clear gains on Exchange, Covid, and ECL. We find that more informative temporal prompts can further improve forecasting performance.

We examine model scalability by evaluating Qwen-3 4B under both Pre-align and Post-align settings and comparing it with the 0.6B and 1.7B variants. As shown in Table 19, increasing the backbone to 4B consistently degrades performance across both ID and OOD settings at horizons 96 and 720. This suggests that simply scaling the language backbone does not necessarily benefit time series forecasting, and an appropriate balance between temporal modeling and language capacity may be more important than model size alone.

Table 17: Comparison of average MSE and token ratios across different LLM backbones.

	In-Domain Test w/ Prompt		In-Domain Test w/o Prompt		Out-of-Domain Test w/ Prompt		Out-of-Domain Test w/o Prompt	
LLM	MSE	Ratio(%)	MSE	Ratio(%)	MSE	Ratio(%)	MSE	Ratio(%)
LLM4TSF(Pre-align)								
GPT-2	0.471	46.7	0.488↑	41.5↓	0.524	72.6	0.673↑	37.9↓
Qwen-3 0.6B	0.485	48.2	0.495↑	42.2↓	0.541	70.9	0.726↑	40.6↓
Qwen-3 1.7B	0.477	45.9	0.502↑	42.9↓	0.529	74.6	0.679↑	35.5↓
LLM4TSF(Post-align)								
GPT-2	0.491	53.3	0.511↑	47.7↓	0.548	76.6	0.626↑	38.5↓
Qwen-3 0.6B	0.477	55.6	0.527↑	51.3↓	0.525	73.9	0.619↑	34.9↓
Qwen-3 1.7B	0.484	52.7	0.535↑	48.6↓	0.541	77.3	0.685↑	28.8↓

Table 18: Effect of incorporating seasonality and trend information into the textual prompt.

Dataset	Horizon = 96		Horizon = 192	
	Original	+ S&T Prompt	Original	+ S&T Prompt
ETTh1	0.412	0.407	0.439	0.425
ETTh2	0.279	0.266	0.347	0.333
ETTm1	0.291	0.286	0.332	0.325
ETTm2	0.165	0.158	0.231	0.220
Weather	0.151	0.154	0.193	0.190
Traffic	0.359	0.362	0.403	0.410
Exchange	0.080	0.071	0.172	0.156
Covid	1.011	0.984	1.246	1.174
ECL	0.129	0.119	0.142	0.131
NN	0.754	0.736	0.786	0.754

H.5 Baseline Results in Out-of-Domain Tests

To assess the out-of-domain generalization of the two alignment strategies after cross-dataset training, we evaluate LLM4TSF (Pre-align) and LLM4TSF (Post-align) on seven unseen datasets, and compare them with three large-scale TS foundation models trained from scratch, namely Chronos (Ansari et al., 2024), UniTS (Gao et al., 2024), and Moirai (Woo et al., 2024a). All methods are evaluated under a zero-shot setting, with model configurations and hyperparameters adopted from the original implementations. We further include two LLM-based TSF models, UniTime (Liu et al., 2024b) and TimeLLM (Jin et al., 2023), which are trained using single-dataset few-shot learning with only 5% of the training data, as additional baselines, as shown in Table 20 & 21.

Table 19: Effect of LLM scale under Pre-align and Post-align settings.

Setting	Backbone	ID-96	ID-720	OOD-96	OOD-720
Pre-align	Qwen-3 0.6B	0.374	0.631	0.432	0.858
	Qwen-3 1.7B	0.368	0.640	0.418	0.726
	Qwen-3 4B	0.399	0.685	0.459	0.927
Post-align	Qwen-3 0.6B	0.370	0.647	0.395	0.757
	Qwen-3 1.7B	0.375	0.654	0.414	0.783
	Qwen-3 4B	0.389	0.679	0.447	0.965

Table 20: MAE of baseline out-of-domain test performance.

Dataset	Horizon	Chronos	UniTS	MOIRAI	UniTime	TimeLLM
Wind	96	0.696	0.755	0.640	0.685	0.664
	192	0.767	0.823	0.722	0.786	0.767
	336	0.848	0.881	0.809	0.878	0.862
	720	0.934	0.945	0.866	0.953	0.945
Solar	96	0.327	0.611	0.477	0.281	0.313
	192	0.339	0.655	0.526	0.270	0.377
	336	0.342	0.679	0.554	0.284	0.395
	720	0.358	0.753	0.597	0.278	0.411
AQShunyi	96	0.491	0.509	0.460	0.546	0.530
	192	0.522	0.538	0.475	0.578	0.551
	336	0.537	0.577	0.497	0.572	0.565
	720	0.565	0.594	0.563	0.592	0.584
CzenLan	96	0.246	0.487	0.477	0.377	0.318
	192	0.277	0.523	0.515	0.397	0.346
	336	0.312	0.566	0.573	0.403	0.354
	720	0.388	0.638	0.622	0.434	0.399
ZafNoo	96	0.399	0.553	0.404	0.603	0.485
	192	0.437	0.571	0.452	0.661	0.479
	336	0.478	0.604	0.481	0.691	0.565
	720	0.507	0.669	0.514	0.724	0.577
NASDAQ	96	0.494	0.769	0.577	0.614	0.503
	192	0.623	0.811	0.653	0.668	0.638
	336	0.744	0.853	0.786	0.719	0.727
	720	0.786	0.860	0.813	0.885	0.822
PEMS	96	0.424	0.839	0.255	0.356	0.339
	192	0.486	0.854	0.271	0.364	0.375
	336	0.501	0.889	0.288	0.389	0.382
	720	0.597	0.916	0.304	0.418	0.419

Table 21: MSE of baseline out-of-domain test performance.

Dataset	Horizon	Chronos	UniTS	MOIRAI	UniTime	TimeLLM
Wind	96	1.250	1.038	0.963	1.022	0.981
	192	1.357	1.224	1.199	1.241	1.201
	336	1.428	1.516	1.268	1.482	1.444
	720	1.653	1.653	1.513	1.688	1.658
Solar	96	0.418	0.779	0.858	0.216	0.402
	192	0.403	0.822	0.913	0.209	0.518
	336	0.425	0.913	0.977	0.228	0.655
	720	0.488	0.968	0.995	0.220	0.733
AQShunyi	96	0.733	0.855	0.607	0.868	0.788
	192	0.779	0.874	0.622	0.912	0.853
	336	0.850	0.902	0.685	0.893	0.876
	720	0.871	0.928	0.759	0.945	0.920
CzenLan	96	0.249	0.649	0.629	0.359	0.263
	192	0.271	0.711	0.644	0.396	0.308
	336	0.308	0.768	0.671	0.399	0.318
	720	0.363	0.825	0.695	0.450	0.385
ZafNoo	96	0.475	0.585	0.455	0.679	0.536
	192	0.511	0.655	0.516	0.790	0.530
	336	0.560	0.694	0.579	0.853	0.645
	720	0.654	0.736	0.623	0.891	0.663
NASDAQ	96	0.506	0.954	0.714	0.833	0.655
	192	0.569	1.036	0.968	1.016	0.926
	336	1.112	1.255	1.253	1.188	1.033
	720	1.305	1.236	1.333	1.452	1.317
PEMS	96	0.485	1.094	0.159	0.349	0.337
	192	0.622	1.186	0.197	0.413	0.425
	336	0.765	1.455	0.251	0.453	0.444
	720	0.872	1.476	0.365	0.461	0.456

H.6 Impact of Statistical Properties

In addition to shifting and transition, we further analyze how variations in stationarity, seasonality, and trend affect the performance of different models. We observe that stationarity plays a non-negligible role in forecasting difficulty: as the degree of stationarity decreases, prediction errors consistently increase. However, under varying stationarity levels, the performance gap between models with and without pre-training, as well as those with and without LLM components, remains relatively small, indicating no significant advantage for either strategy in this regime. Moreover, our results suggest that changes in seasonality strength and trend magnitude do not fundamentally alter the overall forecasting difficulty, and the relative performance of different models remains largely stable across these settings, as shown in Figure 17.

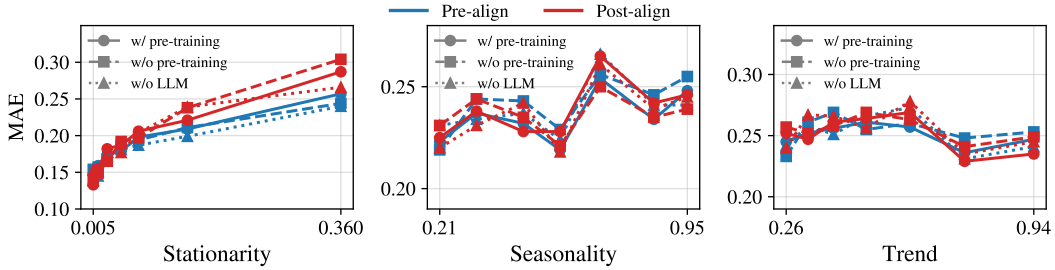

Figure 17: Impact of stationarity, seasonality, and trend on forecasting performance

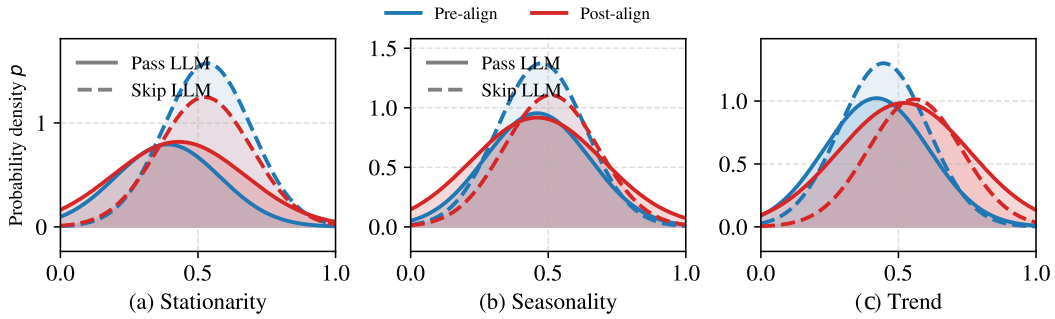

Figure 18: When using LLMs w/ pre-training, the distribution of samples passed the LLM is insensitive to variations in stationarity, seasonality, and trend, exhibiting a relatively uniform pattern.

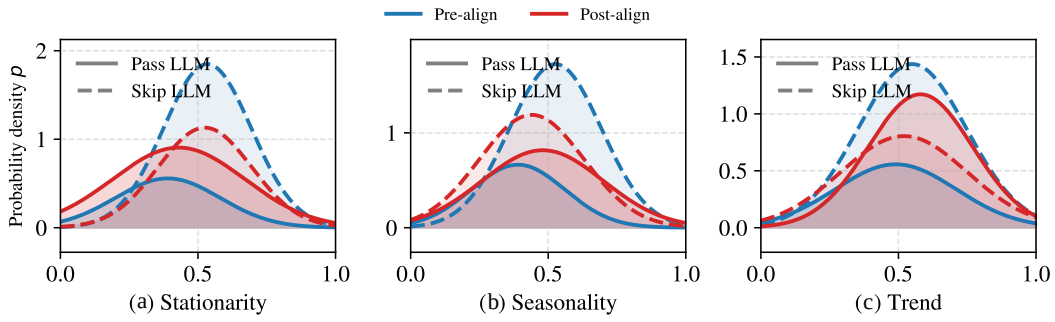

Figure 19: Under the LLM w/ pre-training setting, the pass distribution remains uniform with respect to these properties; however, due to the randomly initialized LLM under the pre-alignment scheme, the proportion of skipped samples is higher than that of passed samples.

I SYNTHETIC TS GENERATION

To isolate the effect of individual temporal properties, we independently construct synthetic TS using property-specific generative operators. Each operator controls a particular temporal characteristic while preserving intrinsic structure, enabling systematic analysis of forecasting difficulty. For synthetic data generation, we construct 100 time series for each attribute, with each series containing 20,000 observations. We adopt a standard dataset splitting protocol for training, validation, and testing. The forecasting setup uses an input length of 512 time steps to predict the next 192 horizons. To ensure that the effect of each attribute is examined in isolation and to avoid confounding factors, we strictly separate the synthetic datasets corresponding to different attributes. As a result, time series generated under different attribute controls do not overlap between training and testing phases, preventing interference across attributes during model evaluation.

Shifting Synthesis. Shifting describes gradual and continuous distributional drift over time. We model shifting as a time-dependent transformation applied to an underlying latent temporal structure (He et al., 2023). Let $v(t)$ denote a latent oscillatory process. A shifted TS is defined as $x_t = \mathcal{S}_s\big(v\big)(t)$, where $\mathcal{S}_s(\cdot)$ denotes a shifting operator parameterized by strength s. The operator induces smooth temporal drift through joint modulation of phase, amplitude, and noise statistics, causing the marginal distribution of x_t to vary gradually with time. Larger values of s correspond to stronger and more complex shifting behavior.

Stationarity Synthesis. Stationarity describes whether the statistical properties of a TS remain invariant over time. We synthesize sequences with varying degrees of stationarity by applying time-dependent distributional transformations (Liu et al., 2022a). Let $v(t)$ be a stationary latent process. The generated TS is defined as $x_t = \sigma_s(t)\, v(t) + \mu_s(t)$, where $\mu_s(t)$ and $\sigma_s(t)$ denote time-varying mean and scale functions. Increasing s induces progressively stronger departures from stationarity, while preserving local temporal regularities in $v(t)$.

Transition Synthesis. Transition characterizes the complexity of temporal dependency structures governing state evolution. We construct transition-dominated sequences by composing the TS from latent regimes with structured switching behavior (Painblanc et al., 2023). Let $\{\pi_k\}_{k=1}^{K}$ denote a finite set of latent temporal patterns. The generated sequence is given by $x_t = \pi_{z_t}(t)$, where z_t is a latent regime index evolving according to a transition operator $z_t \sim \mathcal{T}_s\big(z_{1:t-1}\big)$. Here, $\mathcal{T}_s(\cdot)$ denotes a transition mechanism whose effective dependency order increases with s. As s grows, transitions become increasingly context-dependent, requiring longer temporal history to infer future regimes.

Seasonality Synthesis. Seasonality reflects recurring temporal patterns across one or multiple time scales. We generate seasonal TS by modulating periodic structures with time-varying amplitude and phase (Heidrich et al., 2023). Let $\mathcal{P}(t)$ denote a collection of periodic basis functions. A seasonal sequence is defined as $x_t = \mathcal{A}_s(t)\, \mathcal{P}\big(t + \phi_s(t)\big)$, where $\mathcal{A}_s(t)$ and $\phi_s(t)$ denote amplitude and phase modulation functions, respectively. The parameter s controls the strength and complexity of seasonal variation, allowing a smooth transition from simple stationary periodicity to multi-scale and non-aligned seasonal patterns.

Trend Synthesis. Trend captures long-term directional movement that is not necessarily governed by a globally extrapolatable function (Lin et al., 2021). We construct trend-dominated sequences by composing bounded temporal patterns with structured trend components. Let $v(t)$ denote a bounded latent pattern. A trend-augmented TS is given by $x_t = v(t) + g_s(t)$, where $g_s(t)$ denotes a trend function whose functional form and smoothness vary with s. Larger values of s correspond to more complex and non-linear trend behavior, yielding increasing difficulty for parametric extrapolation methods.

Across all properties, the strength parameter s controls temporal complexity while preserving underlying structure. This design ensures that forecasting difficulty increases in a controlled and interpretable manner, favoring models capable of capturing long-range dependencies and adaptive temporal representations. Five types of synthetic TS generated by controlling different statistical properties, as illustrated in the Figure 20.

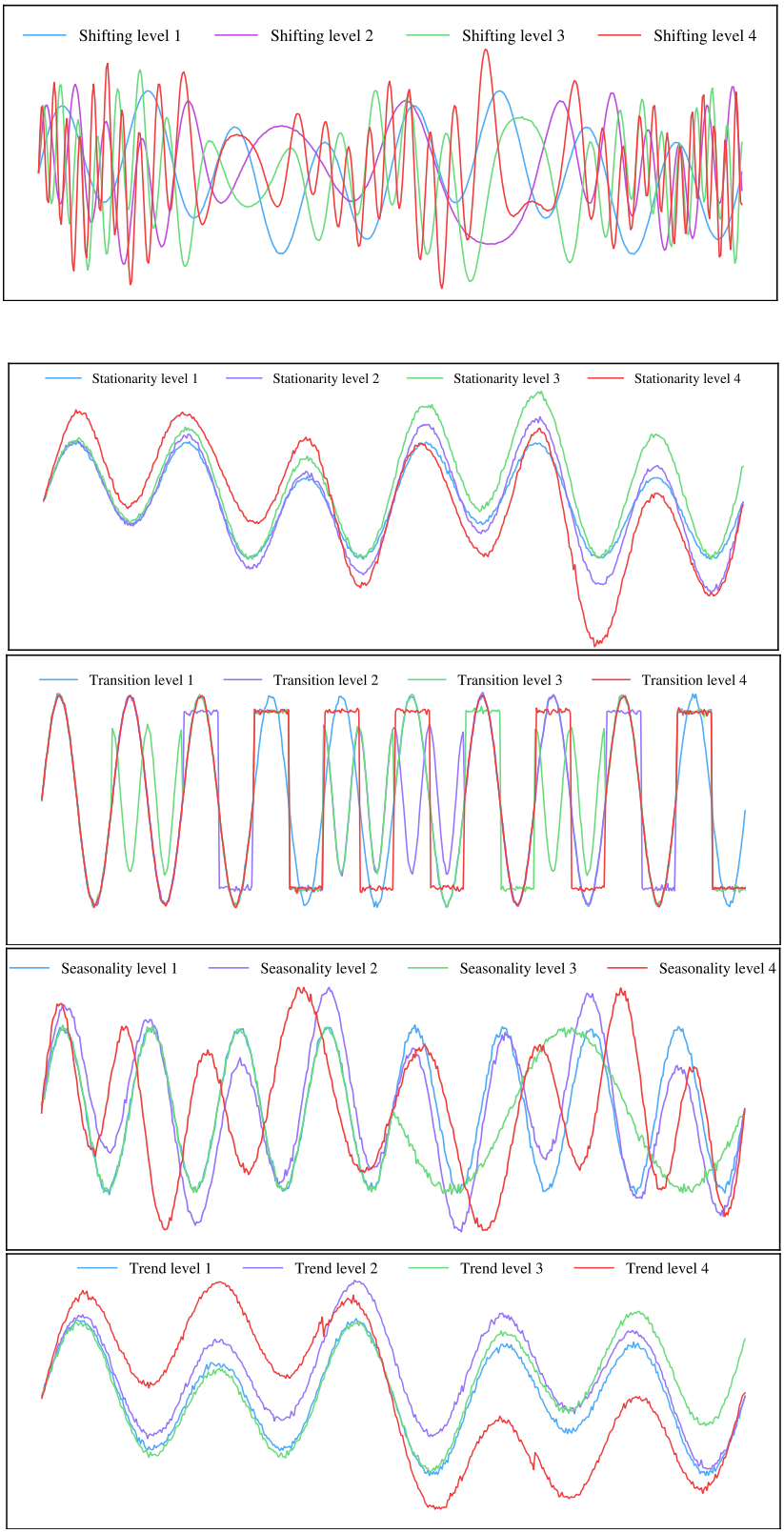

Figure 20: Illustration of synthetic TS generated under different property control levels, where higher levels indicate stronger effects.

J ROUTER MECHANISM

J.1 GUMBEL-SOFTMAX AND STE

For each token, a routing module generates a pair of logits $\mathbf{z} = [z_1, z_2]$, representing the routing preferences over two possible paths. To obtain a discrete routing decision, we employ the Gumbel-Max trick (Jang et al., 2016), where independent Gumbel noise variables $g_i \sim \mathrm{Gumbel}(0, 1)$ are added to the logits, and the routing outcome is sampled as:

$$\mathbf{y}_{\text{hard}} = \text{one-hot}\Big(\arg\max_i (z_i + g_i)\Big).$$

However, the $\arg\max(\cdot)$ operation is non-differentiable, which prevents gradients from propagating through the routing decision. To overcome this limitation, we adopt the Gumbel-Softmax relaxation to obtain a continuous and differentiable approximation of the discrete routing variable:

$$y_i = \frac{\exp((z_i + g_i)/\tau)}{\sum_j \exp((z_j + g_j)/\tau)},$$

where τ is a temperature parameter that controls the smoothness of the resulting distribution. As τ decreases, the output distribution becomes increasingly peaked, approaching a one-hot representation.

In practice, we further integrate the Gumbel-Softmax mechanism with the Straight-Through Estimator (STE) (Bengio et al., 2013). During the forward pass, the discrete routing decision $\mathbf{y}_{\text{hard}}$ is used to perform token-level path selection, while in the backward pass, gradients are propagated through the continuous approximation $\mathbf{y}_{\text{soft}}$. This is achieved by constructing the final routing variable as:

$$\mathbf{y} = \mathbf{y}_{\text{hard}} - \mathrm{detach}(\mathbf{y}_{\text{soft}}) + \mathbf{y}_{\text{soft}}.$$

This design enables effective discrete routing at inference time while maintaining differentiability during training.

J.2 ALGORITHM: TOKEN-LEVEL ROUTING

Table 22: Algorithm: Token-Level Routing

Require: Dataset D, routing module parameters θ, temperature τ
Output: Differentiable token-level routing decisions $\mathbf{y}$

Initialize routing module parameters θ
for each training iteration do
  Sample a token of TS segments $X \sim D$
  for each token t **in** X **do**
    Compute routing logits $\mathbf{z}_t = [z_{t,1}, z_{t,2}]$ using the routing module
    Sample Gumbel noise $g_i \sim \mathrm{Gumbel}(0, 1)$ for each routing option
    Obtain discrete routing decision using Gumbel-Max:
    $\mathbf{y}_t^{\text{hard}} \leftarrow \text{one-hot}(\arg\max_i(z_{t,i} + g_i))$
    Compute continuous routing weights via Gumbel-Softmax relaxation:
    $y_{t,i}^{\text{soft}} \leftarrow \dfrac{\exp((z_{t,i} + g_i)/\tau)}{\sum_j \exp((z_{t,j} + g_j)/\tau)}$
    Apply Straight-Through Estimator (STE):
    $\mathbf{y}_t \leftarrow \mathbf{y}_t^{\text{hard}} - \mathrm{detach}(\mathbf{y}_t^{\text{soft}}) + \mathbf{y}_t^{\text{soft}}$
    Route token t according to $\mathbf{y}_t$
  end for
  Backpropagate gradients through $\mathbf{y}_{\text{soft}}$ and update θ
end for
return learned routing mechanism

K REPRODUCIBILITY DETAILS

To ensure that the results and conclusions can be reproduced accurately, we provide the exact training configurations used in our experiments, as summarized in the Table 23. Additional details of the routing analysis are provided in Table 24.

Table 23: Training Configuration

Hyperparameter	**Value**
Framework	HuggingFace Transformers
Distributed Training	DeepSpeed, Accelerate
GPU	$4\times$ NVIDIA A100 40GB / H100 80GB
Random Seed	Fixed 2026
Optimizer	AdamW
Learning Rate	1×10^{-4}
Adam β_1	0.9
Adam β_2	0.95
Adam ϵ	1×10^{-6}
Weight Decay	0.01
LR Scheduler	Cosine decay
Warmup Ratio	0
Precision	bf16
DeepSpeed Stage	ZeRO Stage 2
Micro Batch Size per GPU	64
Gradient Accumulation Steps	1
Gradient Clipping	Default

Table 24: Routing Analysis Configuration

Routing Parameter	**Value**
Number of Routing Paths	2
Routing Module Output	Logits $\mathbf{z} = [z_1, z_2]$
Gumbel Noise Distribution	Gumbel$(0, 1)$
Temperature τ (init)	1.0
Temperature τ (final)	0.1
Discrete Sampling	Gumbel-Max for $\mathbf{y}_{\text{hard}}$
Straight-Through Estimator	Enabled
Forward Pass Routing	Hard one-hot $\mathbf{y}_{\text{hard}}$
Backward Pass Gradient	Soft $\mathbf{y}_{\text{soft}}$
Inference-time Routing	$\arg\max(\mathbf{z})$
Routing Regularization	Entropy penalty on $\mathbf{y}_{\text{soft}}$
Entropy Weight	1×10^{-3}
Target Routing Ratio	0.5
Ratio Penalty Weight	1×10^{-2}
Router Optimizer	AdamW
Router Learning Rate	3×10^{-4}
Weight Decay	0.01
Gradient Clipping	1.0