TimeSAF: Towards LLM-Guided Semantic Asynchronous Fusion for Time Series Forecasting

Fan Zhang[1], Shiming Fan[1], Hua Wang[2,*],

[1]Shandong Technology and Business University, [2]Ludong University
{zhangfan,2024410061}@sdtbu.edu.cn,
hua.wang@ldu.edu.cn

Abstract

Despite the recent success of large language models (LLMs) in time-series forecasting, most existing methods still adopt a Deep Synchronous Fusion strategy, where dense interactions between textual and temporal features are enforced at every layer of the network. This design overlooks the inherent granularity mismatch between modalities and leads to what we term semantic perceptual dissonance: high-level abstract semantics provided by the LLM become inappropriately entangled with the low-level, fine-grained numerical dynamics of time series, making it difficult for semantic priors to effectively guide forecasting. To address this issue, we propose **TimeSAF**, a new framework based on hierarchical asynchronous fusion. Unlike synchronous approaches, TimeSAF explicitly decouples unimodal feature learning from cross-modal interaction. It introduces an independent cross-modal semantic fusion trunk, which uses learnable queries to aggregate global semantics from the temporal and prompt backbones in a bottom-up manner, and a stage-wise semantic refinement decoder that asynchronously injects these high-level signals back into the temporal backbone. This mechanism provides stable and efficient semantic guidance while avoiding interference with low-level temporal dynamics. Extensive experiments on standard long-term forecasting benchmarks show that TimeSAF significantly outperforms state-of-the-art baselines, and further exhibits strong generalization in both few-shot and zero-shot transfer settings.

1 Introduction

Long-term time series forecasting (LTSF) plays a crucial role in a wide range of real-world applications, including power load management (Fan et al., 2025a; Qiu et al., 2025c), traffic flow analysis (Kieu et al., 2024; Shen and Zhang, 2026; Qiu et al., 2025a; Wang et al., 2026b), weather prediction (Liu et al., 2026), and financial markets (Ariyo et al., 2014; Zhang et al., 2026a). Traditional time series analysis methods typically rely on statistical models or deep learning architectures to capture temporal dependencies from historical observations (Zeng et al., 2023; Wang et al., 2024; Han et al., 2024; Ma et al., 2025; Wang et al., 2026a). However, as illustrated in Fig. 1(a), these models are often confined to the numerical modality and overlook the rich contextual information behind the series, such as metadata and event descriptions (Jin et al., 2023; Ge et al., 2025a). This separation between numerical dynamics and semantic context leads to a **semantic perceptual deficit**, which limits the model's ability to generalize across domains and makes it difficult to adapt to data-scarce scenarios (Zhao et al., 2025; Ding et al., 2025) such as few-shot and zero-shot forecasting.

In recent years, large language models (LLMs) have been introduced into time-series forecasting to compensate for the lack of semantic priors, leveraging their strong reasoning ability and rich parametric knowledge. Existing LLM-based methods typically adopt the deep synchronous fusion strategy shown in Fig. 1(b), i.e., layer-wise semantic coupling, where textual and temporal features are tightly aligned at every layer via dense cross-attention or feature concatenation (Wang et al., 2025; Liu et al., 2024c). However, this design ignores the **semantic perceptual dissonance** between discrete text and continuous time series: high-level abstract semantics are compressed into the same representational scale as low-level numerical fluctuations, leading to heavily entangled features that are hard to interpret or control. We term this effect semantic perceptual dissonance, where LLM priors cannot effectively guide temporal forecasting and may even cause negative transfer when fusion is performed at inappropriate depths.

To address these issues, we propose a hierarchi-

*Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 26267–26284

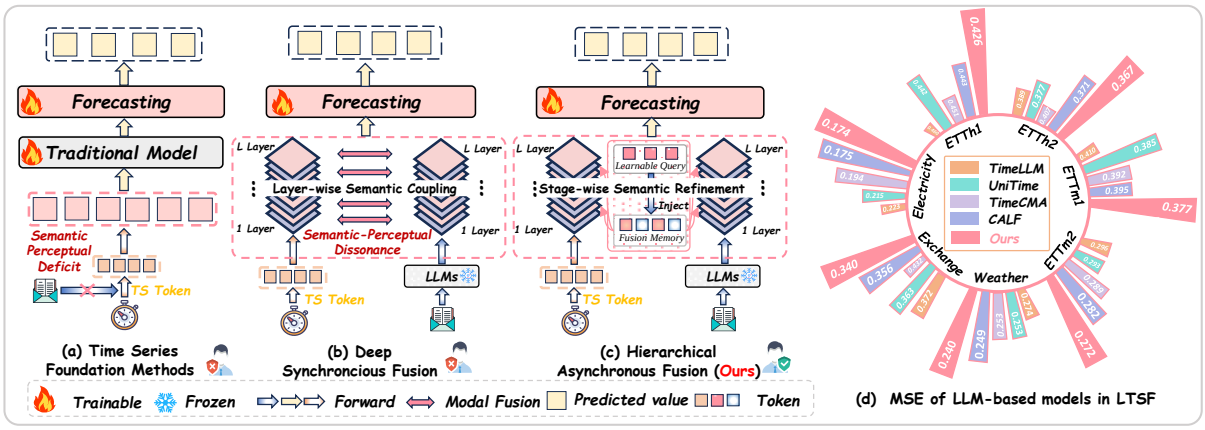


Figure 1: Comparison of strategy and performance between TimeSAF and other methods.

cal asynchronous fusion strategy, as illustrated in Fig. 1(c). Unlike designs that enforce synchronous interaction at every layer, the proposed scheme employs **stage-wise semantic refinement** to restrict cross-modal interaction to a few discrete stages: it first aggregates semantic representations from the time-series backbone and the prompt backbone in a bottom-up manner, and then injects these high-level semantics into deeper layers of the temporal backbone in a top-down manner. This asymmetric, cross-layer interaction prevents numerical modeling and semantic interaction from being entangled at all layers.

Building on this strategy, we develop TimeSAF, a multimodal time-series forecasting framework. Architecturally, TimeSAF constructs a compact semantic memory bank between temporal patterns and textual prompts via a fusion trunk parameterized by learnable queries, while embedding gated asynchronous refinement blocks into both unimodal backbones so that the temporal branch can selectively read from the fusion memory and update its representations. This design deliberately decouples feature extraction and multimodal fusion along the temporal depth, mitigating the interference caused by layer-wise synchronous fusion, while the fine-grained top-down refinement continuously aligns and injects task-relevant semantics from the fusion trunk into the numeric backbone. Extensive experiments show that TimeSAF achieves superior performance to existing methods across multiple LTSF benchmarks and multimodal settings. In summary, the contributions of this paper are as follows:

- **Fusion strategy.** We propose a hierarchical asynchronous fusion strategy that decouples unimodal encoding from cross-modal interaction, effectively alleviating the entanglement between numerical features and textual semantics.
- **Model architecture**. Building on this strategy, we introduce TimeSAF, which incorporates an independent cross-modal semantic fusion trunk and stage-wise semantic refinement decoder. The architecture first aggregates global semantics in a bottom-up manner and then injects the fused semantics back into the temporal backbone in a top-down fashion.
- **Empirical validation.** Extensive experiments on seven public benchmarks demonstrate that the proposed method consistently achieves state-of-the-art performance compared with both LLM-based and non-LLM baselines.

2 Related Work

With the rapid advances of deep learning (Li et al., 2025; Xiao et al., 2026a) in domains such as computer vision (Li et al., 2026b; Chen et al., 2026), video understanding (Chen et al., 2025b), and multimodal representation learning (Xiao et al., 2026b; Ge et al., 2025b), data-driven neural models have also become increasingly prevalent in time series forecasting.

Early time series forecasting predominantly relied on classical statistical models such as ARIMA, VAR, and STL with trend–seasonal decomposition (Siami-Namini et al., 2018; Schorfheide, 2005; Cleveland et al., 1990). These approaches are robust in short-term and single-task settings, but are typically built on stationarity assumptions and are sensitive to high-dimensional nonlinear de-

pendencies and complex noise. With the rise of deep learning, methods based on RNNs, CNNs, Transformers, and MLPs have become mainstream (Sherstinsky, 2020; Wu et al., 2023; Han et al., 2024): RNN/LSTM/GRU model temporal dependencies via recurrent hidden states, convolutional architectures capture local temporal patterns and inter-variable relations, and Transformer-style models leverage global self-attention to better handle long-range dependencies (Li et al., 2026a; Chen et al., 2025a; Hu et al., 2026; Zhang et al., 2026b; Fan et al., 2025b). Building on this, PatchTST (Nie et al., 2023) enhances long-sequence modeling through temporal patching and channel-independent design, while iTransformer (Liu et al., 2024d) and MoE/subspace-based methods (Qiu et al., 2025b) mitigate multivariate heterogeneity and non-stationarity via channel reordering and pattern grouping. Despite this growing architectural diversity, these models still rely solely on historical numerical sequences to make deterministic predictions, which limits cross-domain generalization and leads to suboptimal performance in zero-shot and few-shot regimes.

More recently, large language models (LLMs) have been incorporated into time series forecasting (Gruver et al., 2023; Chang et al., 2025). Time-LLM (Jin et al., 2023) adopts a time-series encoder with LLM interaction, jointly feeding encoded temporal features and textual prompts into the LLM so that it can assist trend understanding and pattern reasoning. GPT4TS (Zhou et al., 2023) converts raw time series into token sequences via encoding, quantization, or descriptive prompts, and lets a pretrained LLM directly generate future trajectories. CALF (Liu et al., 2025b) adopts a dual-stream architecture with dedicated loss functions to achieve deep cross-modal alignment; TimeCMA (Liu et al., 2025a) focuses on cross-channel alignment to alleviate channel entanglement; DualSG (Ding et al., 2025) employs a decoupled dual-stream design, where a numeric stream models fine-grained temporal dynamics and a semantic stream, driven by an LLM, performs trend-level semantic correction and channel-wise reasoning.

Our Work. Unlike prior LLM-based forecasters, TimeSAF first lets the temporal and semantic branches learn stable single-modal representations, and only then injects LLM-derived semantics back into the temporal backbone at a few staged levels, achieving a better balance between semantic guidance and robust numerical modeling.

3 Problem Formulation and Preliminary

3.1 Problem Formulation

Given a multivariate time series $\mathbf{X} = \{x_1, ..., x_L\} \in \mathbb{R}^{L\times N}$ where L denotes the length of the historical window and N the number of variables, we further construct for each variable an offline LLM-derived prompt embedding $\mathbf{E} \in \mathbb{R}^{D_{\text{llm}}\times N}$ where D_{llm} is the dimensionality of the textual feature space. Given an observation window $\mathbf{X}_{t-L+1,t}$ and its associated prompts E, the objective under a forecasting horizon of length H is to learn a mapping:

$$f_\theta\left(\mathbf{X}_{t-L+1,t}, \mathbf{E}\right) \mapsto \hat{Y}_{t+1:t+H} \in \mathbb{R}^{H\times N} \quad (1)$$

3.2 Time Series Encoding Branch

In practical applications, time series often exhibit strong non-stationarity (Liu et al., 2024a). To alleviate this, we first apply reversible instance normalization (RevIN) (Kim et al., 2021) to the input sequence. We then perform segmentation and embedding along the temporal dimension. Given a sequence of length L, with window length P and stride S, the time axis is partitioned into $N_p = \lfloor \frac{L-P}{S} \rfloor + 1$ temporal patches, each containing P consecutive time steps, forming the initial temporal patch tokens $\mathbf{P}^{Time}_{t-L+1,t} \in \mathbb{R}^{N_p\times P}$:

$$\mathbf{P}^{Time}_{t-L+1,t} = \text{Patching}(\mathbf{X}_{t-L+1,t}). \quad (2)$$

A projection layer $g(\cdot)$ is then applied along the temporal dimension to map each patch to a D-dimensional representation. With an added learnable positional embedding e^{pos}, we obtain the final temporal tokens, which are used as input to the subsequent encoder:

$$\mathbf{X}^{time}_{i,t-L+1,t} = g(\mathbf{P}_{i,t-L+1,t}) + e^{pos}. \quad (3)$$

3.3 LLM-based Prompt Encoding Branch

To inject external priors and semantic structure into the model, we introduce an LLM-based prompt encoding branch. This branch leverages a pretrained and frozen GPT-2 backbone to process natural language descriptions associated with segments of the input sequence. Following (Liu et al., 2025a), we automatically generate, for each variable in $\mathbf{X}$, a prompt describing its statistics over the observation window (see Appendix D for details). Each prompt is tokenized by the GPT-2 tokenizer and fed into the frozen GPT-2 encoder, yielding prompt representations $\mathbf{E} \in \mathbb{R}^{D_{\text{llm}}\times N}$, where N is the number of

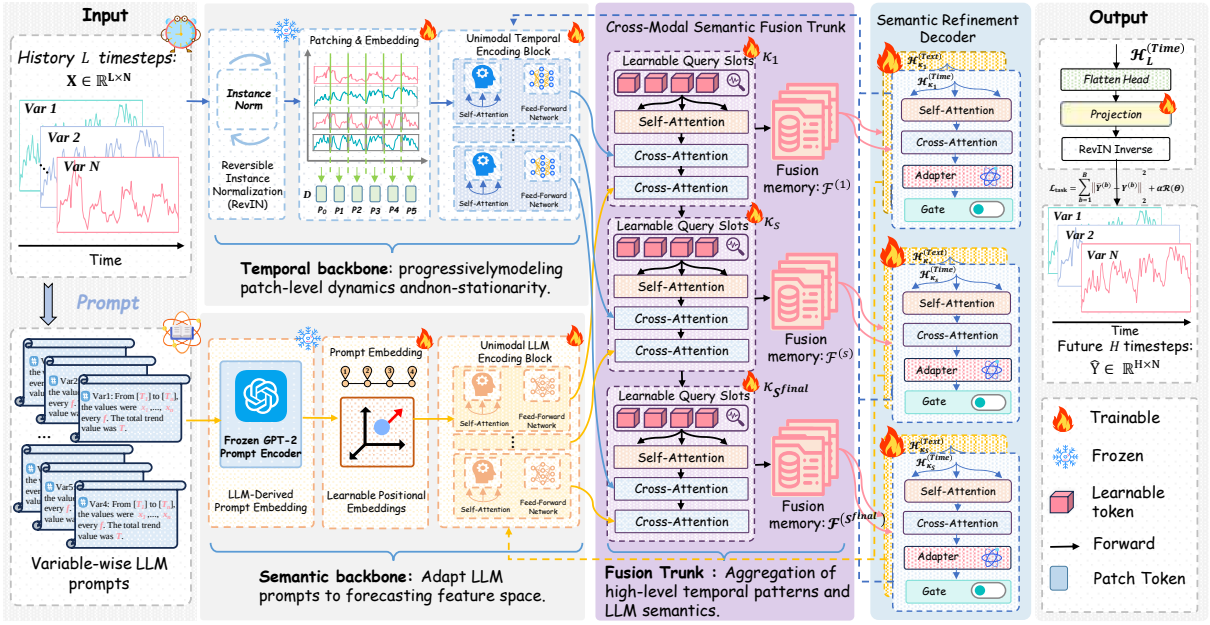


Figure 2: Overall architecture of TimeSAF.

variables and D_{llm} is the LLM embedding dimension (768 for GPT-2). To ensure consistency within a mini-batch, all prompt sequences are padded to a unified length. We then introduce a learnable semantic adaptation module $l(\cdot)$ that maps $\mathbf{E}$ from the original LLM embedding space to the model semantic space $\mathbb{R}^D$, producing node-wise semantic features. A learnable positional embedding e is further added to each variable, resulting in the textual modality embedding $\mathbf{X}^{Text} \in \mathbb{R}^{D \times N}$, which serves as the input to the subsequent semantic encoding branch.

4 Overall Architecture of TimeSAF

In this section, we outline the overall architecture of **TimeSAF**. As illustrated in Fig. 2, given a multivariate historical time series and its associated LLM-based prompts, TimeSAF first encodes the numeric input with a patching-based temporal encoder, while a frozen GPT-2 backbone produces prompt-derived semantic representations. A semantic fusion trunk is inserted at several predefined layers, where a set of learnable queries aggregates information from both the temporal and semantic branches. Subsequently, asynchronous refinement modules inject the fused semantics back into the temporal backbone, and a lightweight prediction head maps the refined temporal tokens to future forecasts.

4.1 Unimodal Encoding Backbones

After obtaining the encoded time-series tokens and LLM-based prompt embeddings, we construct two structurally symmetric unimodal backbones for the numerical and semantic modalities, respectively. Both backbones are composed of several stacked Unimodal Encoding blocks, each consisting of a self-attention layer and a feed-forward network, which progressively extract higher-level representations within each modality without introducing any cross-modal interaction. Let $\mathcal{H}_l$ denote the input to the l-th layer of a unimodal backbone:

$$\mathcal{U}_l = \mathcal{H}_l + SelfAttn(\mathcal{H}_l) \tag{4}$$

Then, a position-wise feed-forward network further transforms the intermediate representation to produce the output of the (l+1)-th layer.

$$\mathcal{H}_{l+1} = \mathcal{U}_l + FFN(\mathcal{U}_l) \tag{5}$$

Here, $SelfAttn(\cdot)$ denotes a multi-head self-attention module with an internal layer normalization, and $FFN(\cdot)$ denotes a position-wise feed-forward network. Stacking multiple Unimodal Encoding blocks on the temporal modality allows the model to progressively enrich the latent representation of the series at the patch level, while on the textual modality, the LLM-derived semantic vectors are gradually adapted to the feature space of the backbone network. It is worth noting that, at this stage, the two backbones evolve strictly within

their own modalities, providing stable and semantically well-formed unimodal representations for subsequent cross-modal fusion.

4.2 Cross-Modal Semantic Fusion Trunk

After the layer-wise encoding of the Unimodal Encoding Backbones, we obtain high-level representations for both the temporal and textual modalities. We then introduce an independent **Cross-Modal Semantic Fusion Trunk**, which performs explicit cross-modal aggregation at a set of designated fusion layers, and compresses information from both branches into a fixed-length fusion memory.

Formally, each unimodal backbone contains dp blocks, and we predefine S fusion stages, yielding $L_S = dp/S$ depth intervals. Let κ_s be the layer index where the s-th fusion stage is triggered. At the beginning of this stage, we instantiate a set of learnable fusion queries $\mathcal{Q}_s^F \in \mathbb{R}^{P_f \times D_f}$, which are broadcast across samples and variables during the forward pass to form the initial fusion representation $\mathcal{H}_{s,0}^F \in \mathbb{R}^{(BN)\times P_f \times D_f}$. When the temporal and textual backbones reach layer κ_s, their hidden states $\mathcal{H}_{\kappa_s}^{Time}$ and $\mathcal{H}_{\kappa_s}^{Text}$ are used as key–value inputs, and a bottom–up semantic aggregation is performed. We first apply self-attention over the fusion queries:

$$\tilde{\mathcal{H}}_s^F = \mathcal{H}_{s,0}^F + Self Attn(\mathcal{H}_{s,0}^F). \tag{6}$$

Then, taking $\tilde{\mathcal{H}}_s^F$ as queries, we retrieve information from the temporal and textual branches via two cross-attention operations:

$$\begin{aligned}\tilde{\mathcal{H}}_s^F &\leftarrow \tilde{\mathcal{H}}_s^F + CrossAttn_{Time}(\tilde{\mathcal{H}}_s^F, \mathcal{H}_{\kappa_s}^{Time}),\\ \tilde{\mathcal{H}}_s^F &\leftarrow \tilde{\mathcal{H}}_s^F + CrossAttn_{Text}(\tilde{\mathcal{H}}_s^F, \mathcal{H}_{\kappa_s}^{Text}).\end{aligned} \tag{7}$$

Finally, a position-wise feed-forward network integrates these signals and yields the fusion memory for the s-th stage:

$$\mathcal{F}^{(s)} = \tilde{\mathcal{H}}_s^F + FFN(\tilde{\mathcal{H}}_s^F). \tag{8}$$

By repeating this procedure over all S stages, the model obtains a set of fusion memories $\{\mathcal{F}^{(1)}, \ldots, \mathcal{F}^{(S)}\}$ that compactly encode high-level semantic information from both backbones. Cross-modal interactions occur only at the designated fusion layers, while the temporal and semantic branches evolve independently in the remaining layers. This stage-wise design allows sufficient multi-modal integration, while avoiding the heavy coupling and optimization instability that arise when cross-attention is inserted at every layer, and provides a clean contextual interface for the subsequent asynchronous refinement modules. A simplified theoretical analysis under linearized assumptions is provided in Appendix E to further support the intuition behind this design.

4.3 Semantic Refinement Decoder

After obtaining the stage-wise fusion memory $\mathcal{F}^{(s)}$ from the semantic fusion trunk, the Semantic Refinement Decoder feeds this high-level semantic context back into both unimodal backbones. Let $m \in \{Time, Text\}$ index the temporal and textual modalities, and denote by $\mathcal{H}_l^{(m)}$ the input to the l-th layer of modality m. Before cross-modal refinement, we first perform an intra-modal self-attention update:

$$\mathcal{U}_l^{(m)} = \mathcal{H}_l^{(m)} + \mathrm{SelfAttn}^{(m)}\big(\mathcal{H}_l^{(m)}\big), \tag{9}$$

where $\mathrm{SelfAttn}^{(m)}$ is a multi-head self-attention block with layer normalization.

The intermediate representation $\mathcal{U}_l^{(m)}$ is then refined using the fusion memory $\mathcal{F}^{(s)}$ via cross-attention:

$$\mathcal{Z}_l^{(m)} = \mathrm{CrossAttn}^{(m)}\big(\mathcal{U}_l^{(m)}, \mathcal{F}^{(s)}\big), \tag{10}$$

where $\mathrm{CrossAttn}^{(m)}(\cdot,\cdot)$ shares the same form as the cross-attention used in the Fusion Block and treats $\mathcal{F}^{(s)}$ as shared key–value context.

To obtain modality-specific refinement while keeping the injection strength controllable, each modality is equipped with an independent linear adapter $\mathcal{W}_{ad}^{(m)}$, and a scalar gate $g \in \mathbb{R}$ shared across modalities. The gated refinement residual is

$$\mathcal{R}_l^{(m)} = \sigma(g)\, \mathcal{W}_{ad}^{(m)}\big(\mathcal{Z}_l^{(m)}\big), \tag{11}$$

where $\sigma(\cdot)$ is the sigmoid function, and $\mathcal{R}_l^{(m)}$ represents the refined signal contributed by the fusion memory. Finally, we add this residual back to the intra-modal representation and apply a position-wise feed-forward network to obtain the layer output:

$$\begin{aligned}\hat{\mathcal{H}}_l^{(m)} &= \mathcal{U}_l{}^{(m)} + \mathcal{R}_l^{(m)}\\ \mathcal{H}_{l+1}^{(m)} &= \hat{\mathcal{H}}_l^{(m)} + FFN^{(m)}(\hat{\mathcal{H}}_l^{(m)})\end{aligned} \tag{12}$$

Architecturally, the temporal and textual branches share the same fusion memory $\mathcal{F}^{(s)}$ inside the Semantic Refinement Decoder, but project it

Table 1: Average MSE and MAE over four prediction lengths. All experiments fix the lookback length $T = 96$. The prediction length set is $H \in \{96, 192, 336, 720\}$. The best result is **red**, the second best result is underlined. Our full results are in Appendix A.

Models	TimeSAF		CALF		TimeCMA		Time-FFM		UniTime		Time-LLM		GPT4TS		iTransformer		PatchTST		Crossformer		FEDformer	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.377**	**0.390**	0.395	0.390	0.392	0.405	0.399	0.402	0.385	0.399	0.410	0.409	0.396	0.400	0.407	0.410	0.387	0.400	0.502	0.502	0.448	0.452
ETTm2	**0.272**	**0.320**	0.282	0.321	0.289	0.332	0.286	0.332	0.293	0.334	0.296	0.340	0.293	0.338	0.288	0.332	0.281	0.326	1.216	0.707	0.304	0.349
ETTh1	**0.426**	**0.428**	0.443	0.435	0.451	0.449	0.442	0.434	0.442	0.447	0.460	0.449	0.456	0.450	0.454	0.447	0.469	0.454	0.620	0.572	0.440	0.459
ETTh2	**0.367**	**0.394**	0.371	0.394	0.407	0.419	0.382	0.406	0.377	0.402	0.389	0.408	0.456	0.413	0.383	0.407	0.387	0.407	0.942	0.683	0.436	0.449
Weather	**0.240**	**0.268**	0.249	0.273	0.253	0.283	0.270	0.288	0.253	0.276	0.274	0.290	0.278	0.297	0.257	0.278	0.259	0.281	0.258	0.315	0.308	0.360
Exchange	0.340	**0.385**	0.356	0.402	0.432	0.446	**0.338**	0.391	0.363	0.404	0.372	0.416	0.370	0.409	0.360	0.403	0.367	0.404	0.470	0.477	0.518	0.428
Electricity	**0.174**	**0.264**	0.175	0.265	0.194	0.287	0.216	0.299	0.215	0.304	0.223	0.309	0.217	0.307	0.178	0.270	0.205	0.290	0.244	0.334	0.213	0.326

back to their own representation spaces through separate adapters $\mathcal{W}_{ad}^{(Time)}$ and $\mathcal{W}_{ad}^{(Text)}$. This shared but modality-specific refinement drives the two branches toward a compatible latent subspace. At the beginning layer κ_s of each fusion stage, a new fusion memory $\mathcal{F}^{(s)}$ is computed by the Cross-Modal Semantic Fusion Trunk and then reused within that stage to update $\mathcal{H}_l^{Time}$ and $\mathcal{H}_l^{Text}$ via $\mathcal{R}_l^{Time}$ and $\mathcal{R}_l^{Text}$, respectively, until the next stage produces a new fusion memory that replaces it as the contextual signal.

4.4 Output Projection and Optimization Objective

Finally, the last-layer temporal representation $\mathcal{H}_d p^{Time}$ is fed into a linear output head, which first flattens the patch-wise features and then maps them to the prediction horizon:

$$\mathbf{Y} = Flatten(\mathcal{H}_d p^{Time}) W_{\text{out}} + \mathbf{b}_{\text{out}}. \quad (13)$$

The resulting forecast $\mathbf{Y}$ is then de-normalized via the inverse RevIN operation to obtain $\hat{\mathbf{Y}} \in \mathbb{R}^{H \times N}$ on the original value scale. TimeSAF is trained with a mean squared error loss plus standard ℓ_2 weight decay. Let Θ denote all trainable parameters; the overall objective is

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \alpha \mathcal{R}(\Theta), \quad (14)$$

where $\alpha \geq 0$ is a balancing coefficient, and

$$\mathcal{L}_{\text{pred}} = \frac{1}{B} \sum_{b=1}^{B} \left\| \hat{\mathbf{Y}}^{(b)} - \mathbf{Y}^{(b)} \right\|_2^2, \quad \mathcal{R}(\Theta) = \sum_{\theta \in \Theta} \theta^2 \quad (15)$$

The objective $\mathcal{L}$ is minimized by back-propagation.

5 Experiments

Datasets and Metrics. We evaluate TimeSAF on seven widely used multivariate time series benchmarks: the four subsets of the Electricity Transformer Temperature (ETT) dataset (ETTh1, ETTh2, ETTm1, and ETTm2), together with Electricity, Weather, and Exchange. In line with standard practice in forecasting studies, we adopt Mean Absolute Error (MAE) and Mean Squared Error (MSE) as our primary evaluation metrics. The detailed statistics of these datasets are summarized in Appendix B.

Baselines. We compare TimeSAF against a diverse set of recent and representative forecasting models. (1) LLM-based models: CALF (Liu et al., 2025b), TimeCMA (Liu et al., 2025a),Time-FFM (Liu et al., 2024b) , UniTime (Liu et al., 2024c), Time-LLM (Jin et al., 2023), and GPT4TS (Zhou et al., 2023). (2) Transformer-based models: iTransformer(Liu et al., 2024d), PatchTST (Nie et al., 2023), Crossformer (Zhang and Yan, 2023), and FEDformer (Zhou et al., 2022). (3) CNN-based models: TimesNet (Wu et al., 2023) and MICN (Wang et al., 2023). (4) MLP-based models: DLinear (Zeng et al., 2023).

Implementation Details. We conduct all experiments under a unified evaluation pipeline and adopt the same configuration as (Wu et al., 2023) to ensure a fair comparison with strong baselines. We use a pretrained GPT-2 model (the first six layers) (Wu et al., 2023) as the default LLM backbone. TimeSAF is optimized using the Adam optimizer, trained for up to 50 epochs with early stopping. All experiments are run on 8 NVIDIA GeForce RTX 3090 GPUs (24 GB each). See Appendix C for

Table 2: Few-shot forecasting performance on ETT datasets using only 10% of the training data. All experiments fix the lookback length $T = 96$. The prediction horizon is set to $H \in \{96, 192, 336, 720\}$. Our full results are in Appendix A.

Models	TimeSAF		CALF		TimeCMA		Time-LLM		GPT4TS		PatchTST		Crossformer		FEDformer		TimesNet		DLinear		MICN	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.483**	**0.449**	0.504	0.462	0.594	0.504	0.636	0.512	0.608	0.500	0.557	0.483	1.340	0.848	0.696	0.572	0.673	0.534	0.567	0.499	0.970	0.674
ETTm2	**0.294**	**0.330**	0.302	0.330	0.309	0.344	0.308	0.343	0.303	0.336	0.295	0.334	1.985	1.048	0.356	0.392	0.321	0.354	0.329	0.382	1.073	0.716
ETTh1	**0.622**	**0.527**	0.644	0.541	0.721	0.575	0.765	0.584	0.689	0.555	0.683	0.645	1.744	0.914	0.750	0.607	0.865	0.625	0.647	0.552	1.405	0.814
ETTh2	0.422	**0.423**	**0.419**	0.427	0.451	0.448	0.589	0.498	0.579	0.497	0.550	0.487	3.139	1.378	0.553	0.525	0.476	0.463	0.441	0.458	2.533	1.158

Table 3: Zero-shot forecasting performance on the ETT datasets, where prediction lengths $H \in \{96, 192, 336, 720\}$. "h1→m1" indicates that models trained on ETTh1 are evaluated on ETTm1, and similarly for the other transfer settings. Our full results are in Appendix A.

Models	TimeSAF		CALF		TimeCMA		Time-LLM		GPT4TS		PatchTST		Crossformer		FEDformer		TimesNet		DLinear		MICN	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
h1 → m1	**0.749**	**0.563**	0.755	0.574	0.820	0.590	0.847	0.600	0.798	0.574	0.894	0.610	0.999	0.736	0.765	0.588	0.794	0.575	1.439	0.870	0.760	0.577
h1 → m2	**0.313**	**0.355**	0.316	0.355	0.329	0.365	0.315	0.357	0.317	0.359	0.318	0.362	1.120	0.789	0.357	0.403	0.339	0.370	2.428	1.236	0.399	0.439
h2 → m1	0.873	0.604	0.836	**0.586**	1.087	0.673	0.868	0.595	0.920	0.610	0.871	0.596	1.195	0.711	**0.741**	0.588	1.286	0.705	0.764	0.601	0.778	0.594
h2 → m2	**0.317**	**0.360**	0.319	0.360	0.349	0.386	0.322	0.363	0.331	0.371	0.420	0.433	2.043	1.124	0.365	0.405	0.361	0.390	0.527	0.519	0.496	0.496

more details.

5.1 Long-term Forecasting

Setups. For a fair comparison, we fix the input sequence length to $L = 96$ and consider four forecasting horizons $H \in \{96, 192, 336, 720\}$. Consistent with the TFB-based setup (Xu et al., 2023), we maintain the same configuration but do not use the "Drop-Last" trick during training to ensure fair comparison.

Results. The overall results are summarized in Table 1. On the aggregated comparison across all datasets and forecasting horizons, TimeSAF consistently outperforms all baselines and achieves the best performance on all 14 aggregated metrics. Compared with state-of-the-art LLM-based methods (CALF, TimeCMA, Time-FFM, UniTime, and Time-LLM), TimeSAF reduces MSE by 3.10%, 8.83%, 6.58%, and 10.31%, respectively. It also clearly surpasses Transformer-, CNN-, and MLP-based models, with improvements over these baselines typically exceeding 5%. These experimental results demonstrate that our TimeSAF can fully utilize the temporal patterns and semantic information in a limited input sequence, thus enabling accurate predictions.

5.2 Few/zero-shot Learning

Setups. Given that large language models (LLMs) have demonstrated strong generalization in few-shot and zero-shot learning settings (Zhou et al., 2023; Jin et al., 2023), this property is particularly relevant for real-world time series forecasting under data-scarce scenarios. Therefore, we also evaluate the performance of TimeSAF in few-shot and zero-shot regimes. In the few-shot setting, we use the ETT datasets and restrict the training data to only 10% of the original training split. In the zero-shot setting, the model trained on one dataset is directly deployed to a new, unseen dataset without any additional training, while keeping the training hyperparameters identical to those used in the long-term forecasting experiments.

Few-shot Learning. Table 2 reports the results on the challenging few-shot forecasting task under different prediction horizons. With only a small fraction of training samples, TimeSAF achieves the best performance on 7 out of 8 overall metrics, indicating that the model can effectively extract useful patterns from limited historical data. Compared with LLM-based baselines (CALF, TimeCMA, Time-LLM, and GPT4TS), TimeSAF achieves relative MSE improvements of 2.38%, 10.93%, 18.91%, and 15.09%, respectively. In addi-

tion, TimeSAF outperforms the strong Transformer-based baseline PatchTST by 11.46%, further confirming its advantage in the few-shot setting.

Zero-shot Learning. To further assess the cross-dataset generalization ability of TimeSAF, we conduct zero-shot transfer experiments, where the model is trained on one dataset and then directly evaluated on a different dataset without any additional fine-tuning. As reported in Table 3, TimeSAF achieves the lowest average MSE on three out of four transfer directions. Overall, its zero-shot performance is comparable to the strongest LLM-based baseline, CALF, while consistently outperforming TimeCMA, Time-LLM, and GPT4TS, with average MSE reductions of 10.60%, 3.29%, and 4.19%, respectively. These results indicate that the proposed asynchronous fusion framework offers highly competitive zero-shot transfer capability compared with existing LLM-enhanced forecasting models.

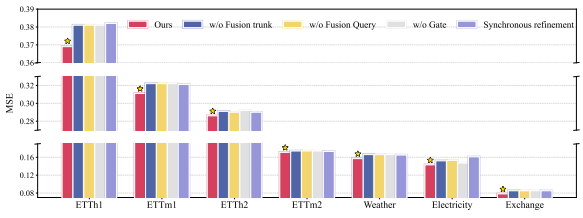


Figure 3: Ablation studies of different variants of TimeSAF on multiple datasets.

5.3 Ablation Study

To rigorously assess the contribution of each component in TimeSAF, we conduct ablation studies guided by four questions: ❶ Is the semantic fusion trunk necessary? ❷ Does explicitly modeling fusion query slots outperform directly aggregating on unimodal features? ❸ Does gated semantic injection help stabilize asynchronous refinement? ❹ Is stage-wise asynchronous interaction superior to layer-wise synchronous updates?

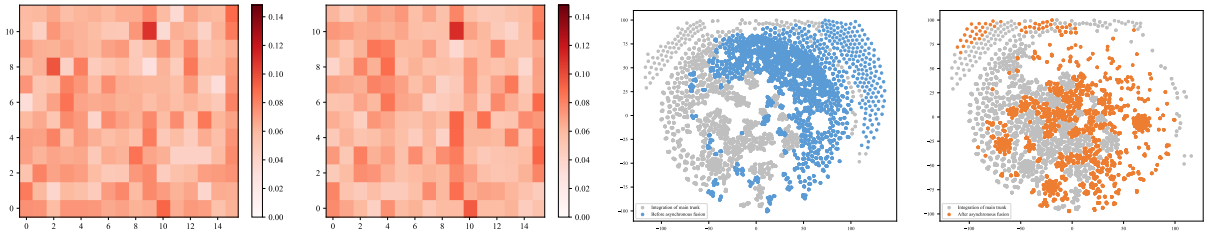
Accordingly, we construct four variants: **w/o Fusion Trunk**, which removes the semantic fusion trunk; **w/o Fusion Query**, which discards the independent fusion query slots and performs cross-modal interaction directly on unimodal backbone features; **w/o Gate**, which removes the scalar gating factor; and **Synchronous Refinement**, which replaces the proposed stage-wise asynchronous scheme with synchronous refinement within each layer. As shown in Fig. 3, all variants suffer noticeable performance drops compared to the full TimeSAF model, confirming the utility of each component. Notably, Synchronous Refinement performs worst on most datasets, as synchronous fusion fails to mitigate semantic–perceptual dissonance, preventing high-level semantics from being properly formed and from effectively guiding low-level temporal representations. Overall, these results validate both the necessity of the proposed modules and the effectiveness of the hierarchical asynchronous fusion architecture.

5.4 Attention Flow Visualization

To qualitatively examine how the proposed hierarchical asynchronous fusion operates, we visualize the cross-attention maps between the semantic fusion trunk and the temporal trunk. As shown in Fig. 4(a)–(b), we plot (i) the attention from fusion queries to temporal patches at the fusion stage, and (ii) the attention from temporal patches back to fusion queries at the subsequent refinement stage. The two maps exhibit highly consistent patterns: temporal regions that receive strong attention from certain fusion queries during fusion tend to query the same slots during refinement. In other words, the time positions selected and aggregated by the fusion trunk later become the main recipients of its top-down feedback. This aligned attention flow supports our design intuition that the model first uses the fusion trunk to gather information from salient temporal regions, and then reuses the fused semantics to refine exactly those regions, forming a stable "aggregate–refine" loop that is absent in conventional synchronous fusion schemes.

5.5 T-SNE Visualization

We further examine the effect of asynchronous refinement from the perspective of the representation space. For a fixed refinement stage, we project (i) temporal features before refinement, (ii) temporal features after refinement, and (iii) the corresponding fusion features into a shared two-dimensional space using t-SNE, as shown in Fig. 4(c)–(d). Before refinement, the temporal features form a cluster clearly separated from the fusion features, indicating a mismatch between unimodal temporal representations and the fused semantic space. After applying the asynchronous refinement module, the temporal cluster moves toward and partially overlaps with the fusion cluster, suggesting that the refined temporal representations become better aligned with the fusion space. This embedding-level alignment supports that the proposed asyn-



(a) (b) (c) (d)

Figure 4: Visualization of the proposed asynchronous fusion mechanism on the Exchange dataset. (a) Cross-attention maps from fusion queries to temporal patches in the fusion stage. (b) Cross-attention maps from temporal patches to fusion queries in the refinement stage. (c) t-SNE projection of temporal features and fusion features before refinement. (d) t-SNE projection of temporal features and fusion features after refinement.

chronous fusion mechanism effectively narrows the representational gap between the temporal backbone and the fusion backbone.

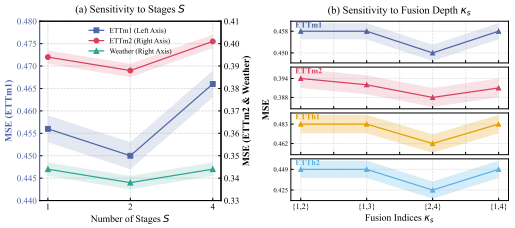


Figure 5: Sensitivity of TimeSAF to fusion configuration.

5.6 Sensitivity to stage-wise fusion

As shown in Fig. 5. We first vary the number of fusion stages $S \in \{1, 2, 4\}$ on several datasets. Moving from a single stage to $S = 2$ consistently reduces the prediction error, while further increasing to $S = 4$ brings little or no benefit, indicating that a small number of fusion stages is sufficient. Fixing $S = 2$, we then shift the fusion layer indices κ_s and observe that placing fusion blocks at middle/deeper layers yields slightly better results than very shallow or edge-heavy placements. Overall, TimeSAF is reasonably robust to S and κ_s, and we adopt $S = 2$ with a middle–deep fusion placement as the default setting.

6 Conclusion

We presented TimeSAF, a hierarchical asynchronous fusion framework for multimodal time-series forecasting. By decoupling unimodal encoding from cross-modal interaction and introducing a semantic fusion trunk with stage-wise refinement, TimeSAF mitigates semantic perceptual dissonance and allows LLM priors to guide temporal modeling more reliably. Experiments on standard LTSF benchmarks, as well as few-shot and zero-shot settings, show that TimeSAF achieves competitive or superior performance over strong Transformer-based and LLM-enhanced baselines, while remaining conceptually simple and practically deployable.

Limitations

This work still has several limitations. First, TimeSAF is mainly evaluated on a set of standard LTSF benchmarks and constructed few-shot/zero-shot settings, and lacks large-scale studies in real industrial deployments or more complex multimodal environments. Our current implementation relies on rule-based templates to convert time-series statistics into natural language prompts. While this design is effective, it may not fully exploit the reasoning capability of language models compared with leveraging rich unstructured external knowledge (e.g., financial news or weather reports). Due to hardware constraints, our experiments primarily use GPT-2 as the semantic backbone. Although TimeSAF itself is model-agnostic, we have not yet systematically explored its behavior when scaled to larger or more advanced foundation models (such as LLaMA-3 or GPT-4). Future work can conduct more extensive empirical validation on larger-scale and more diverse real-world tasks.

Ethical considerations

This work studies TimeSAF on public, aggregated benchmark datasets (ETT, Electricity, Weather, Exchange), which do not contain identifiable personal information. We adhere to the licenses and usage policies of the original data providers and do not introduce any additional sensitive data. Although TimeSAF is a generic forecasting framework, we

do not conduct a dedicated fairness or bias analysis, and applying the model in high-stakes domains should therefore involve domain experts and proper risk assessment. Our experiments use a frozen medium-scale GPT-2 backbone and moderate GPU resources, which limits but does not eliminate the environmental footprint. We encourage responsible use of TimeSAF as a decision-support tool rather than an autonomous decision-maker, and discourage applications that lack transparency or may cause societal harm.

Acknowledgements

This work was supported in part by the following: the National Natural Science Foundation of China under Grant Nos. U24A20219, 62272281, U24A20328, 62576193, the Yantai Natural Science Foundation under Grant No. 2024JCYJ034, and the Youth Innovation Technology Project of Higher School in Shandong Province under Grant No. 2023KJ212.

References

Adebiyi A Ariyo, Adewumi O Adewumi, and Charles K Ayo. 2014. Stock price prediction using the arima model. In *2014 UKSim-AMSS 16th international conference on computer modelling and simulation*, pages 106–112. IEEE.

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

Zhiwei Chen, Yupeng Hu, Zhiheng Fu, Zixu Li, Jiale Huang, Qinlei Huang, and Yinwei Wei. 2026. Intent: Invariance and discrimination-aware noise mitigation for robust composed image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 20463–20471.

Zhiwei Chen, Yupeng Hu, Zixu Li, Zhiheng Fu, Xuemeng Song, and Liqiang Nie. 2025a. Offset: Segmentation-based focus shift revision for composed image retrieval. In *Proceedings of the ACM International Conference on Multimedia*, page 6113–6122.

Zhiwei Chen, Yupeng Hu, Zixu Li, Zhiheng Fu, Haokun Wen, and Weili Guan. 2025b. Hud: Hierarchical uncertainty-aware disambiguation network for composed video retrieval. In *Proceedings of the ACM International Conference on Multimedia*, page 6143–6152.

Robert B Cleveland, William S Cleveland, Jean E McRae, Irma Terpenning, and 1 others. 1990. Stl: A seasonal-trend decomposition. *J. off. Stat*, 6(1):3–73.

Kuiye Ding, Fanda Fan, Yao Wang, Ruijie Jian, Xiaorui Wang, Luqi Gong, Yishan Jiang, Chunjie Luo, and Jianfeng Zhan. 2025. Dualsg: A dual-stream explicit semantic-guided multivariate time series forecasting framework. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 508–517.

Shiming Fan, Hua Wang, and Fan Zhang. 2025a. Cawformer: A cross variable attention with discrete wavelet denoising for multivariate time series forecasting. *Knowledge-Based Systems*, page 113846.

Shiming Fan, Hua Wang, and Fan Zhang. 2025b. Fsmamba: A dual-expert architecture with fast global attention and local-enhanced state-space mamba for time series forecasting. *Knowledge-Based Systems*, page 115233.

Yunfeng Ge, Ming Jin, Yiji Zhao, Hongyan Li, Bo Du, Chang Xu, and Shirui Pan. 2025a. Eventtsf: Event-aware non-stationary time series forecasting. *arXiv preprint arXiv:2508.13434*.

Yunfeng Ge, Jiawei Li, Yiji Zhao, Haomin Wen, Zhao Li, Meikang Qiu, Hongyan Li, Ming Jin, and Shirui Pan. 2025b. T2s: High-resolution time series generation with text-to-series diffusion models. *arXiv preprint arXiv:2505.02417*.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. 2023. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36:19622–19635.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. 2024. Softs: Efficient multivariate time series forecasting with series-core fusion. *arXiv preprint arXiv:2404.14197*.

Yupeng Hu, Zixu Li, Zhiwei Chen, Qinlei Huang, Zhiheng Fu, Mingzhu Xu, and Liqiang Nie. 2026. Refine: Composed video retrieval via shared and differential semantics enhancement. *ACM Transactions on Multimedia Computing, Communications and Applications*.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and 1 others. 2023. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*.

Duc Kieu, Tung Kieu, Peng Han, Bin Yang, Christian S Jensen, and Bac Le. 2024. Team: Topological evolution-aware framework for traffic forecasting–extended version. *arXiv preprint arXiv:2410.19192*.

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

Zixu Li, Zhiwei Chen, Haokun Wen, Zhiheng Fu, Yupeng Hu, and Weili Guan. 2025. Encoder: Entity mining and modification relation binding for composed image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 5101–5109.

Zixu Li, Yupeng Hu, Zhiwei Chen, Qinlei Huang, Guozhi Qiu, Zhiheng Fu, and Meng Liu. 2026a. Retrack: Evidence-driven dual-stream directional anchor calibration network for composed video retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 23373–23381.

Zixu Li, Yupeng Hu, Zhiwei Chen, Shiqi Zhang, Qinlei Huang, Zhiheng Fu, and Yinwei Wei. 2026b. Habit: Chrono-synergia robust progressive learning framework for composed image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 6762–6770.

Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. 2025a. Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18780–18788.

Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. 2025b. Calf: Aligning llms for time series forecasting via cross-modal fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18915–18923.

Peiyuan Liu, Beiliang Wu, Yifan Hu, Naiqi Li, Tao Dai, Jigang Bao, and Shu-tao Xia. 2024a. Timebridge: Non-stationarity matters for long-term time series forecasting. *arXiv preprint arXiv:2410.04442*.

Qingxiang Liu, Xu Liu, Chenghao Liu, Qingsong Wen, and Yuxuan Liang. 2024b. Time-ffm: Towards lm-empowered federated foundation model for time series forecasting. *Advances in Neural Information Processing Systems*, 37:94512–94538.

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

Xvyuan Liu, Xiangfei Qiu, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Jilin Hu, and Bin Yang. 2026. Rethinking irregular time series forecasting: A simple yet effective baseline. In *AAAI*.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2024d. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*.

Jiaming Ma, Binwu Wang, Qihe Huang, Guanjun Wang, Pengkun Wang, Zhengyang Zhou, and Yang Wang. 2025. Mofo: Empowering long-term time series forecasting with periodic pattern modeling. *Proc. Adv. Neural Inf. Process. Syst*.

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

Xiangfei Qiu, Xingjian Wu, Hanyin Cheng, Xvyuan Liu, Chenjuan Guo, Jilin Hu, and Bin Yang. 2025a. Dbloss: Decomposition-based loss function for time series forecasting. In *NeurIPS*.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. 2025b. DUET: Dual clustering enhanced multivariate time series forecasting. In *SIGKDD*, pages 1185–1196.

Xiangfei Qiu, Yuhan Zhu, Zhengyu Li, Xingjian Wu, Bin Yang, and Jilin Hu. 2025c. Dag: A dual correlation network for time series forecasting with exogenous variables. *arXiv preprint arXiv:2509.14933*.

Frank Schorfheide. 2005. Var forecasting under misspecification. *Journal of Econometrics*, 128(1):99–136.

Qiannan Shen and Jing Zhang. 2026. Mftformer: Meteorological-frequency-temporal transformer with block-aligned fusion for traffic flow prediction. *Research Square*. Preprint, doi:10.21203/rs.3.rs-8770196/v1.

Alex Sherstinsky. 2020. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. 2018. A comparison of arima and lstm in forecasting time series. In *2018 17th IEEE international conference on machine learning and applications (ICMLA)*, pages 1394–1401. Ieee.

Hua Wang, Jinghao Lu, and Fan Zhang. 2026a. Eeotfv: Escape-explore optimizer for web-scale time-series forecasting and vision analysis. *arXiv preprint arXiv:2602.02551*.

Hua Wang, Jinghao Lu, and Fan Zhang. 2026b. Idealtsf: Can non-ideal data contribute to enhancing the performance of time series forecasting models? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 26224–26232.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. 2023. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The Eleventh International Conference on Learning Representations*.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. 2024. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*.

Shunnan Wang, Min Gao, Zongwei Wang, Yibing Bai, Feng Jiang, and Guansong Pang. 2025. Freqllm: frequency-aware large language models for time series forecasting. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pages 3389–3397.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. Timesnet: Temporal 2d-variation modeling for general time series analysis.

Canran Xiao, Jiabao Dou, Zhiming Lin, Zong Ke, and Liwei Hou. 2026a. From points to coalitions: Hierarchical contrastive shapley values for prioritizing data samples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 15995–16003.

Canran Xiao, Tianxiang Xu, Yiyang Jiang, Haoyu Gao, Yuhan Wu, and 1 others. 2026b. Reversible primitive–composition alignment for continual vision–language learning. In *The Fourteenth International Conference on Learning Representations*.

Zhijian Xu, Ailing Zeng, and Qiang Xu. 2023. Fits: Modeling time series with $10k$ parameters. *arXiv preprint arXiv:2307.03756*.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128.

Fan Zhang, Shiming Fan, and Hua Wang. 2026a. Timetk: A multi-offset temporal interaction framework combining transformer and kolmogorov-arnold networks for time series forecasting. *arXiv preprint arXiv:2602.11190*.

Fan Zhang, Zhiwei Gu, and Hua Wang. 2026b. Decoding with structured awareness: integrating directional, frequency-spatial, and structural attention for medical image segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 12421–12429.

Yunhao Zhang and Junchi Yan. 2023. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*.

Zhe Zhao, Pengkun Wang, Haibin Wen, Shuang Wang, Liheng Yu, and Yang Wang. 2025. Stem-lts: Integrating semantic-temporal dynamics in llm-driven time series analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 22858–22866.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pages 27268–27286. PMLR.

Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, and 1 others. 2023. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355.

A Performance of Long-term Multivariate Forecasting

This appendix reports the complete numerical results for all forecasting experiments. Table 4 summarizes long-term multivariate forecasting on all benchmarks, using a fixed input length of $L = 96$ and four prediction horizons $H \in \{96, 192, 336, 720\}$. Table 5 provides the corresponding few-shot results, where each model uses only 10% of the original training set, $L = 96$ and $H \in \{96, 192, 336, 720\}$. Table 6 reports the zero-shot transfer setting, in which models are trained on a source dataset and directly evaluated on a different target dataset without any further fine-tuning, still with $L = 96$ and $H \in \{96, 192, 336, 720\}$. For all tables, we list MSE and MAE for TimeSAF and all baselines.

B Dataset Descriptions

We extensively evaluate our model on seven widely recognized real-world datasets, covering diverse domains such as energy, weather, and economics. Table 7 summarizes the key statistics of these datasets. Specifically, the ETT dataset is divided into training, validation, and test sets with a ratio of 6:2:2, whereas all other datasets follow a 7:1:2 split.

- **ETT (Electricity Transformer Temperature):** This dataset comprises two years of data (July 2016 to July 2018) collected from electricity transformers, including oil temperature and power load features. It is categorized into four subsets based on sampling frequency: **ETTh1/ETTh2** (hourly) and **ETTm1/ETTm2** (every 15 minutes), allowing for evaluation at different temporal granularities.

- **Electricity:** This dataset monitors the hourly electricity consumption (in kW) of 321 clients from 2012 to 2014. The timestamps follow Portuguese time, requiring specific handling for Daylight Saving Time (DST). Specifically, measurements during the skipped hour in March (1:00 AM - 2:00 AM) are set to

Table 4: All experiments fix the lookback length $T = 96$. The prediction length set is $H \in \{96, 192, 336, 720\}$. The best result is **red**, the second best result is underlined.

Models		TimeSAF		CALF		TimeCMA		Time-FFM		UniTime		Time-LLM		GPT4TS		iTransformer		PatchTST		Crossformer		FEDformer	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.311**	0.351	0.319	**0.348**	0.330	0.371	0.336	0.369	0.322	0.363	0.359	0.381	0.335	0.369	0.334	0.368	0.329	0.367	0.360	0.401	0.379	0.419
	192	**0.357**	**0.372**	0.373	0.375	0.368	0.389	0.378	0.389	0.366	0.387	0.383	0.393	0.374	0.385	0.377	0.391	0.367	0.385	0.402	0.440	0.426	0.441
	336	**0.390**	**0.399**	0.411	0.400	0.402	0.411	0.411	0.410	0.398	0.407	0.416	0.414	0.407	0.406	0.426	0.420	0.399	0.410	0.543	0.528	0.445	0.459
	720	**0.450**	**0.438**	0.478	0.438	0.471	0.450	0.469	0.441	0.454	0.440	0.483	0.449	0.469	0.442	0.491	0.459	0.454	0.439	0.704	0.642	0.543	0.490
	avg	**0.377**	**0.390**	0.395	0.390	0.392	0.405	0.399	0.402	0.385	0.399	0.410	0.409	0.396	0.400	0.407	0.410	0.387	0.400	0.502	0.502	0.448	0.452
ETTm2	96	**0.171**	0.255	0.175	**0.253**	0.181	0.263	0.181	0.267	0.183	0.266	0.193	0.280	0.190	0.275	0.180	0.264	0.175	0.259	0.273	0.356	0.203	0.287
	192	**0.236**	**0.299**	0.242	0.299	0.256	0.315	0.247	0.308	0.251	0.310	0.257	0.318	0.253	0.313	0.250	0.309	0.241	0.302	0.426	0.487	0.269	0.328
	336	**0.294**	**0.336**	0.303	0.337	0.309	0.347	0.309	0.347	0.319	0.351	0.317	0.353	0.321	0.360	0.311	0.348	0.305	0.343	1.013	0.714	0.325	0.366
	720	**0.388**	**0.393**	0.409	0.398	0.411	0.404	0.406	0.404	0.420	0.410	0.419	0.411	0.411	0.406	0.412	0.407	0.402	0.400	3.154	1.274	0.421	0.415
	avg	**0.272**	**0.320**	0.282	0.321	0.289	0.332	0.286	0.332	0.293	0.334	0.296	0.340	0.293	0.338	0.288	0.332	0.281	0.326	1.216	0.707	0.304	0.349
ETTh1	96	**0.369**	**0.391**	0.376	0.393	0.399	0.402	0.385	0.400	0.397	0.418	0.398	0.410	0.398	0.424	0.386	0.405	0.414	0.419	0.420	0.439	0.376	0.419
	192	0.423	**0.420**	0.432	0.427	0.442	0.441	0.439	0.430	0.434	0.439	0.451	0.440	0.449	0.427	0.441	0.436	0.460	0.445	0.540	0.519	**0.420**	0.448
	336	**0.451**	**0.438**	0.479	0.451	0.477	0.477	0.480	0.449	0.468	0.457	0.508	0.471	0.492	0.466	0.487	0.458	0.501	0.466	0.722	0.648	0.459	0.465
	720	**0.462**	0.466	0.486	0.472	0.488	0.478	0.462	**0.456**	0.469	0.477	0.483	0.478	0.487	0.483	0.503	0.491	0.500	0.488	0.799	0.685	0.506	0.507
	avg	**0.426**	**0.428**	0.443	0.435	0.451	0.449	0.442	0.434	0.442	0.447	0.460	0.449	0.456	0.450	0.454	0.447	0.469	0.454	0.620	0.572	0.440	0.459
ETTh2	96	**0.286**	0.338	0.290	**0.337**	0.326	0.363	0.301	0.351	0.296	0.345	0.295	0.346	0.398	0.360	0.297	0.349	0.302	0.348	0.745	0.584	0.358	0.397
	192	0.370	0.390	**0.366**	**0.385**	0.420	0.420	0.378	0.397	0.374	0.394	0.386	0.399	0.449	0.405	0.380	0.400	0.388	0.400	0.877	0.656	0.429	0.439
	336	**0.391**	**0.410**	0.416	0.423	0.442	0.444	0.422	0.431	0.415	0.427	0.447	0.443	0.492	0.437	0.428	0.432	0.426	0.433	1.043	0.731	0.496	0.487
	720	0.425	0.438	**0.415**	**0.434**	0.442	0.452	0.427	0.444	0.425	0.444	0.428	0.444	0.487	0.453	0.427	0.445	0.431	0.446	1.104	0.763	0.463	0.474
	avg	**0.367**	**0.394**	0.371	0.394	0.407	0.419	0.382	0.406	0.377	0.402	0.389	0.408	0.456	0.413	0.383	0.407	0.387	0.407	0.942	0.683	0.436	0.449
Weather	96	**0.157**	**0.200**	0.166	0.205	0.176	0.221	0.191	0.230	0.171	0.214	0.195	0.233	0.203	0.244	0.174	0.214	0.177	0.218	0.158	0.230	0.217	0.296
	192	**0.204**	**0.247**	0.214	0.253	0.215	0.256	0.236	0.267	0.217	0.254	0.240	0.269	0.247	0.277	0.221	0.254	0.225	0.259	0.206	0.277	0.276	0.336
	336	**0.262**	**0.287**	0.268	0.291	0.276	0.302	0.289	0.303	0.274	0.293	0.293	0.306	0.297	0.311	0.278	0.296	0.278	0.297	0.272	0.335	0.339	0.380
	720	**0.338**	**0.340**	0.348	0.345	0.348	0.353	0.362	0.350	0.351	0.343	0.368	0.354	0.368	0.356	0.358	0.347	0.354	0.348	0.398	0.418	0.403	0.428
	avg	**0.240**	**0.268**	0.249	0.273	0.253	0.283	0.270	0.288	0.253	0.276	0.274	0.290	0.278	0.297	0.257	0.278	0.259	0.281	0.258	0.315	0.308	0.360
Exchange	96	**0.078**	**0.196**	0.082	0.201	0.097	0.221	0.081	0.201	0.086	0.209	0.087	0.208	0.091	0.212	0.086	0.206	0.088	0.205	0.139	0.265	0.148	0.278
	192	0.170	**0.293**	0.173	0.296	0.195	0.320	**0.168**	0.293	0.174	0.299	0.173	0.299	0.183	0.304	0.177	0.299	0.176	0.299	0.241	0.375	0.271	0.315
	336	0.312	0.406	0.345	0.425	0.372	0.449	**0.299**	**0.396**	0.319	0.408	0.375	0.454	0.328	0.417	0.331	0.417	0.301	0.397	0.392	0.468	0.460	0.427
	720	**0.801**	**0.647**	0.824	0.686	1.067	0.794	0.805	0.674	0.875	0.701	0.853	0.703	0.880	0.704	0.847	0.691	0.901	0.714	1.110	0.802	1.195	0.695
	avg	0.340	**0.385**	0.356	0.402	0.432	0.446	**0.338**	0.391	0.363	0.404	0.372	0.416	0.370	0.409	0.360	0.403	0.367	0.404	0.470	0.477	0.518	0.428
Electricity	96	**0.143**	**0.237**	0.145	0.239	0.146	0.247	0.198	0.282	0.196	0.287	0.204	0.293	0.197	0.290	0.148	0.240	0.181	0.270	0.219	0.314	0.193	0.308
	192	**0.160**	0.254	0.161	**0.252**	0.163	0.264	0.199	0.285	0.199	0.291	0.207	0.295	0.201	0.292	0.162	0.253	0.188	0.274	0.231	0.322	0.201	0.315
	336	**0.176**	**0.266**	0.177	0.269	0.223	0.323	0.212	0.298	0.214	0.305	0.219	0.308	0.217	0.309	0.178	0.269	0.204	0.293	0.246	0.337	0.214	0.329
	720	**0.219**	**0.300**	0.219	0.302	0.245	0.317	0.253	0.330	0.254	0.335	0.263	0.341	0.253	0.339	0.225	0.317	0.246	0.324	0.280	0.363	0.246	0.355
	avg	**0.174**	**0.264**	0.175	0.265	0.194	0.287	0.216	0.299	0.215	0.304	0.223	0.309	0.217	0.307	0.178	0.270	0.205	0.290	0.244	0.334	0.213	0.326

zero, while the overlapping hour in October is handled by aggregating the values.

- **Weather:** Recorded by the Max Planck Institute for Biogeochemistry, this dataset consists of 21 meteorological indicators collected every 10 minutes throughout 2020, capturing fine-grained climatic variations.
- **Exchange:** This financial dataset tracks the daily exchange rates of eight major countries (Australia, UK, Canada, Switzerland, China, Japan, New Zealand, and Singapore) against the US dollar, covering a long historical period from 1990 to 2016.

C Implementation Details

Our model is implemented in Python 3.10 with the PyTorch 2.2 framework. All training and inference are conducted on a compute cluster equipped with eight NVIDIA GeForce RTX 3090 GPUs. We adopt the Adam optimizer and perform grid search over learning rates in $\{1 \times 10^{-4}, 3 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}\}$, while the batch size is selected from $\{16, 32, 48, 64\}$. For key architectural hyperparameters, we conduct extensive tuning to determine the final configuration: (1) the total number of layers in the unimodal backbones is chosen from $\{2, 4\}$; (2) the number of fusion layers is selected from $\{1, 2, 4\}$; and (3) the model dimension D is searched over $\{64, 128, 256, 512\}$. To ensure reproducibility, all experiments use a fixed random seed of 2024. For baseline models, to maintain fairness and consistency, the first two baselines are re-implemented using their official code under the same experimental environment, UniTime (Liu et al., 2024c) results are directly taken from its original paper, and the remaining baselines follow the reported results in the iTransformer (Liu et al., 2024d) and CALF (Liu et al., 2025b) papers.

Table 5: Few-shot forecasting performance on ETT datasets using only 10% of the training data. The prediction horizon is set to $H \in \{96, 192, 336, 720\}$. We report the average MSE and MAE over all horizons; The best result is **red**, the second best result is underlined.

Models		TimeSAF		CALF†		TimeCMA†		Time-LLM		GPT4TS		PatchTST		Crossformer		FEDformer		TimesNet		DLinear		MICN	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.442**	**0.430**	0.468	0.445	0.568	0.483	0.587	0.491	0.615	0.497	0.558	0.478	1.037	0.705	0.604	0.530	0.583	0.503	0.552	0.488	0.677	0.585
	192	**0.457**	**0.437**	0.479	0.446	0.532	0.475	0.606	0.490	0.597	0.492	0.539	0.471	1.170	0.778	0.641	0.546	0.608	0.515	0.546	0.487	0.784	0.627
	336	**0.476**	**0.450**	0.499	0.463	0.548	0.489	0.719	0.555	0.597	0.501	0.558	0.488	1.463	0.913	0.768	0.606	0.733	0.572	0.567	0.501	0.972	0.684
	720	**0.559**	**0.479**	0.572	0.496	0.729	0.571	0.632	0.514	0.623	0.513	0.574	0.498	1.693	0.997	0.771	0.606	0.768	0.548	0.606	0.522	1.449	0.800
	avg	**0.483**	**0.449**	0.504	0.462	0.594	0.504	0.636	0.512	0.608	0.500	0.557	0.483	1.340	0.848	0.696	0.572	0.673	0.534	0.567	0.499	0.970	0.674
ETTm2	96	**0.182**	**0.261**	0.190	0.268	0.205	0.280	0.189	0.270	0.187	0.266	0.189	0.268	1.397	0.866	0.222	0.314	0.214	0.288	0.225	0.320	0.389	0.448
	192	**0.248**	**0.303**	0.257	0.311	0.269	0.320	0.264	0.319	0.253	0.308	0.248	0.307	1.757	0.987	0.284	0.351	0.271	0.325	0.291	0.362	0.622	0.575
	336	**0.311**	0.343	0.323	**0.334**	0.325	0.354	0.327	0.358	0.332	0.353	0.311	0.346	2.075	1.086	0.392	0.419	0.329	0.356	0.354	0.402	1.055	0.755
	720	**0.435**	0.416	0.441	**0.410**	0.436	0.421	0.454	0.428	0.438	0.417	0.435	0.418	2.712	1.253	0.527	0.485	0.473	0.448	0.446	0.447	2.226	1.087
	avg	**0.294**	**0.330**	0.302	0.330	0.309	0.344	0.308	0.343	0.303	0.336	0.295	0.334	1.985	1.048	0.356	0.392	0.321	0.354	0.329	0.382	1.073	0.716
ETTh1	96	**0.430**	**0.426**	0.468	0.457	0.655	0.540	0.500	0.464	0.462	0.449	0.433	0.428	1.129	0.775	0.651	0.563	0.855	0.625	0.590	0.515	0.689	0.592
	192	**0.491**	**0.459**	0.550	0.501	0.673	0.550	0.590	0.516	0.551	0.495	0.509	0.474	1.832	0.922	0.666	0.562	0.791	0.589	0.634	0.541	1.160	0.748
	336	0.615	0.549	0.581	0.521	0.717	0.568	0.638	0.542	0.630	0.539	**0.572**	**0.509**	2.022	0.973	0.767	0.602	0.939	0.648	0.659	0.554	1.747	0.899
	720	0.955	0.674	0.978	0.685	0.841	0.641	1.334	0.816	1.113	0.738	1.221	0.773	1.903	0.986	0.918	0.703	0.876	0.641	**0.708**	**0.598**	2.024	1.019
	avg	**0.622**	**0.527**	0.644	0.541	0.721	0.575	0.765	0.584	0.689	0.555	0.683	0.645	1.744	0.914	0.750	0.607	0.865	0.625	0.647	0.552	1.405	0.814
ETTh2	96	**0.311**	**0.348**	0.314	0.360	0.339	0.375	0.329	0.365	0.327	0.359	0.314	0.354	2.482	1.206	0.359	0.404	0.372	0.405	0.361	0.407	0.510	0.502
	192	0.412	**0.404**	0.404	0.411	0.443	0.437	0.414	0.413	**0.403**	0.405	0.420	0.415	3.136	1.372	0.460	0.461	0.483	0.463	0.444	0.453	1.809	1.036
	336	**0.452**	**0.450**	0.458	0.452	0.513	0.486	0.579	0.506	0.568	0.499	0.543	0.489	2.925	1.331	0.569	0.530	0.541	0.496	0.509	0.501	3.250	1.419
	720	0.513	0.493	0.502	0.487	0.506	0.492	1.034	0.711	1.020	0.725	0.926	0.691	4.014	1.603	0.827	0.707	0.510	0.491	**0.453**	**0.471**	4.564	1.676
	avg	0.422	**0.423**	**0.419**	0.427	0.451	0.448	0.589	0.498	0.579	0.497	0.550	0.487	3.139	1.378	0.553	0.525	0.476	0.463	0.441	0.458	2.533	1.158

Table 6: Full results for zero-shot forecasting on the ETT datasets, where prediction lengths $H \in \{96, 192, 336, 720\}$. "h1→m1" indicates that models trained on ETTh1 are evaluated on ETTm1, and the same applies to other items. The best result is **red**, the second best result is underlined.

Models	Metric	TimeSAF		CALF		TimeCMA		Time-LLM		GPT4TS		PatchTST		Crossformer		FEDformer		TimesNet		MICN		DLinear	
h1 → m1	96	**0.729**	**0.544**	0.767	0.564	0.809	0.577	0.804	0.565	0.809	0.563	0.908	0.596	0.856	0.649	0.731	0.561	0.764	0.563	0.832	0.621	0.735	0.554
	192	0.750	**0.554**	0.753	0.570	0.814	0.584	0.827	0.593	0.799	0.567	0.927	0.616	0.906	0.684	**0.746**	0.573	0.798	0.562	1.288	0.854	0.752	0.570
	336	**0.744**	**0.564**	0.745	0.575	0.814	0.591	0.835	0.600	0.803	0.577	0.920	0.621	1.104	0.796	0.775	0.596	0.790	0.584	1.721	0.972	0.749	0.579
	720	0.774	**0.588**	**0.758**	0.590	0.841	0.607	0.922	0.644	0.783	0.589	0.822	0.608	1.131	0.816	0.808	0.625	0.827	0.594	1.915	1.036	0.805	0.606
	avg	**0.749**	**0.563**	0.755	0.574	0.820	0.590	0.847	0.600	0.798	0.574	0.894	0.610	0.999	0.736	0.765	0.588	0.794	0.575	1.439	0.870	0.760	0.577
h1 → m2	96	0.217	0.301	0.218	0.301	0.229	0.310	**0.212**	**0.298**	0.218	0.304	0.219	0.305	0.611	0.588	0.257	0.345	0.245	0.322	0.496	0.556	0.239	0.343
	192	**0.276**	0.337	0.278	**0.334**	0.289	0.343	0.277	0.338	0.279	0.338	0.280	0.341	0.789	0.685	0.318	0.380	0.293	0.346	1.798	1.137	0.320	0.397
	336	**0.334**	**0.367**	0.338	0.369	0.347	0.376	0.336	0.371	0.342	0.376	0.341	0.376	1.469	0.927	0.375	0.417	0.361	0.382	2.929	1.472	0.409	0.453
	720	**0.426**	**0.416**	0.431	0.418	0.450	0.430	0.435	0.424	0.431	0.419	0.432	0.426	1.612	0.957	0.480	0.472	0.460	0.432	4.489	1.782	0.629	0.565
	avg	**0.313**	**0.355**	0.316	0.355	0.329	0.365	0.315	0.357	0.317	0.359	0.318	0.362	1.120	0.789	0.357	0.403	0.339	0.370	2.428	1.236	0.399	0.439
h2 → m1	96	0.848	0.575	0.897	0.589	1.295	0.729	0.891	0.587	0.985	0.604	0.815	**0.560**	1.032	0.620	**0.734**	0.578	1.205	0.678	0.743	0.577	0.762	0.567
	192	0.863	0.602	0.864	0.584	1.196	0.701	0.850	**0.583**	0.872	0.600	0.900	0.606	1.176	0.676	**0.723**	0.594	1.159	0.670	0.750	0.588	0.785	0.588
	336	0.849	0.604	0.816	**0.585**	1.087	0.670	0.853	0.594	0.926	0.614	0.906	0.602	1.199	0.718	**0.750**	0.590	1.197	0.689	0.764	0.606	0.767	0.594
	720	0.932	0.634	0.768	**0.589**	0.770	0.590	0.879	0.616	0.899	0.624	0.866	0.619	1.373	0.832	**0.760**	0.592	1.583	0.784	0.801	0.634	0.800	0.627
	avg	0.873	0.604	0.836	**0.586**	1.087	0.673	0.868	0.595	0.920	0.610	0.871	0.596	1.195	0.711	**0.741**	0.588	1.286	0.705	0.764	0.601	0.778	0.594
h2 → m2	96	**0.220**	**0.300**	0.225	0.310	0.272	0.351	0.228	0.311	0.235	0.316	0.288	0.345	0.821	0.634	0.261	0.347	0.244	0.324	0.327	0.414	0.264	0.366
	192	**0.281**	**0.339**	0.283	0.342	0.323	0.376	0.283	0.341	0.287	0.346	0.344	0.375	1.732	1.018	0.313	0.370	0.331	0.374	0.450	0.485	0.394	0.452
	336	**0.338**	**0.372**	0.340	0.373	0.369	0.397	0.343	0.376	0.361	0.391	0.438	0.425	2.587	1.393	0.401	0.431	0.386	0.405	0.526	0.526	0.506	0.513
	720	**0.428**	0.430	0.429	**0.418**	0.432	0.418	0.437	0.424	0.444	0.433	0.611	0.588	3.034	1.452	0.487	0.472	0.485	0.458	0.806	0.652	0.822	0.655
	avg	**0.317**	**0.360**	0.319	0.360	0.349	0.386	0.322	0.363	0.331	0.371	0.420	0.433	2.043	1.124	0.365	0.405	0.361	0.390	0.527	0.519	0.496	0.496

C.1 Evaluation Metrics

We choose mean square error and mean absolute error as the commonly used performance evaluation indicators in time series forecasting. Their mathematical definitions are as follows:

$$MSE = \frac{1}{H}\sum_{i=1}^{H}\left(\mathbf{Y}_i - \hat{\mathbf{Y}}_i\right)^2$$
$$MAE = \frac{1}{H}\sum_{i=1}^{H}\left|\mathbf{Y}_i - \hat{\mathbf{Y}}_i\right| \quad (16)$$

where $\mathbf{Y}_i$ denotes the true value, $\hat{\mathbf{Y}}_i$ is the predicted value, and H denotes the size of the prediction window.

D Prompt Description

For all experiments, we convert each multivariate time-series window into a short natural-language description before feeding it into the frozen GPT-2 encoder. As shown in Fig. 6. For a given variable,

Table 7: Dataset descriptions. *Variables* denotes the dimension of the multivariate time series. *Frequency* indicates the sampling interval.

Dataset	Variables	Frequency	Scope	Time Range	Predicted Window
ETTh1, ETTh2	7	1 Hour	Energy	2016/07 – 2018/07	{96,192,336,720}
ETTm1, ETTm2	7	15 Mins	Energy	2016/07 – 2018/07	{96,192,336,720}
Electricity	321	1 Hour	Energy	2012 – 2014	{96,192,336,720}
Weather	21	10 Mins	Weather	2020 Whole Year	{96,192,336,720}
Exchange	8	1 Day	Finance	1990 – 2016	{96,192,336,720}

we instantiate the following template:

> From [T_1] to [T_n], the values were [x_1, ..., x_n] every [f]. The total trend value was [T].

Here [T_1] and [T_n] denote the start and end timestamps of the window, [x_i] are the observed values sampled at the dataset-specific resolution Δt (15 minutes for ETTm1/ETTm2, 1 hour for ETTh1/ETTh2/ECL, 10 minutes for Weather, and 1 day for Exchange), and [T] is a scalar trend statistic over the window. This template is applied independently to each variable, and the resulting tokenized prompts are padded and stacked along the variable dimension to construct the textual input tensor used in our model.

E Theoretical Rationale of Hierarchical Asynchronous Fusion

In this section we give a stylized variance analysis to explain why deep layer-wise semantic coupling is more sensitive to semantic noise than the proposed hierarchical asynchronous fusion. The goal is not to provide a strict generalization bound, but to show how semantic noise can accumulate with depth in a simplified setting.

E.1 Simplified Setup

For clarity, we consider one scalar feature dimension and linearize the forward propagation around a fixed point. Let h_l denote the hidden state at layer l, and let $F(\cdot)$ summarize the deterministic transformation of self-attention and FFN. We assume that at each fusion operation, the semantic branch provides a signal that can be decomposed as

$$s_l = \mu + \varepsilon_l,$$

where μ is the useful semantic component shared across layers and ε_l is zero-mean semantic noise (including mismatch between text and time series as well as structural noise), with

$$\mathbb{E}[\varepsilon_l] = 0, \qquad \mathrm{Var}(\varepsilon_l) \le \sigma^2.$$

We emphasize that this analysis is purely illustrative and relies on linearization and independence assumptions; it is intended to clarify the intuition behind hierarchical asynchronous fusion rather than to serve as a rigorous guarantee for the full nonlinear model.

E.2 Noise Accumulation in Deep Synchronous Fusion

In deep synchronous fusion, semantic information is injected at every layer. A linearized update can be written as

$$h_{l+1}^{\text{syn}} \approx F\big(h_l^{\text{syn}}\big) + \lambda(\mu + \varepsilon_l), \tag{17}$$

where λ controls the injection strength. Unrolling L layers gives

$$h_L^{\text{syn}} \approx h_0 + \sum_{l=0}^{L-1} F\big(h_l^{\text{syn}}\big) + \sum_{l=0}^{L-1} \lambda(\mu + \varepsilon_l). \tag{18}$$

The noise accumulated from the semantic branch is

$$E_{\text{syn}} = \sum_{l=0}^{L-1} \lambda \varepsilon_l. \tag{19}$$

Its variance is

$$\mathrm{Var}(E_{\text{syn}}) = \lambda^2 \,\mathrm{Var}\left(\sum_{l=0}^{L-1} \varepsilon_l\right). \tag{20}$$

By Cauchy–Schwarz,

$$\mathrm{Var}\left(\sum_{l=0}^{L-1} \varepsilon_l\right) \le \Big(\sum_{l=0}^{L-1} \sqrt{\mathrm{Var}(\varepsilon_l)}\Big)^2 \le L^2\sigma^2, \tag{21}$$

Figure 6: Hint templates for specific datasets are used to transcribe multivariate time series segments into natural language descriptions. Each template jointly encodes the time interval, numerical sequence, sampling frequency, and trend statistics to establish a unified representation between structured time series data and language model input.

thus we obtain the following upper bound:

$$\mathrm{Var}(E_{\text{syn}}) \le L^2 \lambda^2 \sigma^2. \tag{22}$$

This shows that, in the worst case, the variance of semantic noise in deep synchronous fusion can grow quadratically with the network depth L. When the per-layer noises are positively correlated (which is plausible since they are generated from the same prompt and fusion mechanism), this bound can be nearly tight.

E.3 Noise Accumulation in Hierarchical Asynchronous Fusion

In TimeSAF, semantic information is injected only at a small set of fusion layers. Suppose the backbone has depth L and there are S fusion stages, with fusion indices collected in a set $\mathcal{K}_{\text{fusion}}$. For each fusion stage $s \in \mathcal{K}_{\text{fusion}}$, a linearized update of the temporal branch can be written as

$$h^{\text{asy}}_{\kappa_s+1} \approx F\big(h^{\text{asy}}_{\kappa_s}\big) + \lambda_s(\mu + \varepsilon_s), \tag{23}$$

where λ_s denotes the semantic injection strength at stage s. Semantic noise is now accumulated only at these S layers:

$$E_{\text{asy}} = \sum_{s \in \mathcal{K}_{\text{fusion}}} \lambda_s \varepsilon_s. \tag{24}$$

Similarly,

$$\begin{aligned} \mathrm{Var}(E_{\text{asy}}) &= \mathrm{Var}\left(\sum_{s \in \mathcal{K}_{\text{fusion}}} \lambda_s \varepsilon_s \right) \\ &\le \Big(\sum_{s \in \mathcal{K}_{\text{fusion}}} |\lambda_s| \sqrt{\mathrm{Var}(\varepsilon_s)} \Big)^2 \\ &\le S^2 \lambda_{\max^2} \sigma^2 \end{aligned} \tag{25}$$

where $\lambda_{\max} = \max_s |\lambda_s|$. If we assume that the injection strengths of the two schemes are comparable, i.e., $\lambda_{\max} \approx \lambda$, then combining (22) and (25) yields

$$\frac{\mathrm{Var}(E_{\text{asy}})}{\mathrm{Var}(E_{\text{syn}})} \lesssim \frac{S^2}{L^2} \ll 1, \tag{26}$$

because in practice $S \ll L$ (e.g., $S = 2$ vs. $L = 6$ in our experiments). Moreover, TimeSAF

introduces a learnable gating factor $\sigma(g) \in [0, 1]$ on each refinement connection, which effectively scales down λ_s and further suppresses semantic noise injection when prompts are uninformative. This provides an additional safeguard against semantic perceptual dissonance.

E.4 Discussion

The above analysis is based on a linearized one-dimensional abstraction and ignores higher-order nonlinear effects. Nevertheless, it clearly shows that repeatedly injecting noisy semantic signals at every layer can lead to much stronger noise accumulation than injecting them at a small number of carefully selected fusion stages. This provides a theoretical rationale for why the proposed hierarchical asynchronous fusion is empirically more robust than deep layer-wise semantic coupling.

F Algorithm Pseudocode

Algorithm 1 outlines the forward pass of TimeSAF using a PyTorch-like style. The core distinction of our asynchronous fusion is highlighted in the conditional execution of the fusion block.

G Prompt Variant Ablation

To study whether the informativeness of prompts affects the semantic features, we conduct a prompt-variant ablation with several concise prompt formulations. For simplicity and transferability, we avoid complex prompts that require extensive domain-specific analysis; instead, we construct textual descriptions using lightweight statistical cues. This design enables reproducible prompt generation for new datasets without additional data-specific engineering, which is aligned with practical zero-shot evaluation. Specifically, we compare the full prompt used in Time-SAF (**Time-SAF**) with three simplified prompt types: **Domain** (domain-only prompt), **Timestamp** (numeric-description-only prompt), and **Instruction** (instruction-only prompt). The results are reported in Table 8. Overall, the concise statistic-based prompting still provides stable gains.

Table 8: Prompt variant ablation (MSE; lower is better).

Dataset	H	Time-SAF	Domain	Timestamp	Instruction
ETTh2	96	0.286	0.289	0.292	0.291
	192	0.370	0.375	0.384	0.376
	336	0.391	0.398	0.390	0.395
	720	0.425	0.428	0.426	0.429
Exchange	96	0.078	0.081	0.085	0.080
	192	0.170	0.175	0.178	0.173
	336	0.312	0.323	0.319	0.329
	720	0.801	0.812	0.802	0.806
Weather	96	0.157	0.156	0.159	0.162
	192	0.204	0.208	0.210	0.209
	336	0.262	0.266	0.270	0.269
	720	0.338	0.339	0.338	0.341

Table 9: Cross-dataset zero-shot transfer results (lower is better).

Transfer Setting	H	Time-SAF		Time-FFM	
		MSE	MAE	MSE	MAE
ETTm2→Electricity	96	0.521	0.581	0.616	0.613
	192	0.345	0.335	0.649	0.631
	336	0.486	0.478	0.687	0.651
	720	0.612	0.597	0.736	0.675
ETTm1→Weather	96	0.235	0.268	0.235	0.270
	192	0.291	0.321	0.289	0.312
	336	0.318	0.325	0.329	0.336
	720	0.397	0.371	0.402	0.381

H Additional Cross-Dataset Zero-shot Transfer

Prior zero-shot evaluations mainly within the ETT family (e.g., h→m transfers) correspond to relatively mild domain shifts, which may not sufficiently demonstrate robustness under larger domain gaps. We add two cross-dataset zero-shot transfer settings as an initial validation (Table 9): ETTm2→Electricity and ETTm1→Weather. As shown, under these more challenging transfer scenarios, Time-SAF achieves better overall performance than the representative semantic fusion baseline Time-FFM, suggesting that our fusion strategy remains reasonably robust under larger domain shifts. We appreciate the reviewer's suggestion to include broader cross-domain tests, and we will further expand the evaluation with more cross-domain transfer tasks and more systematic domain-gap analyses in a subsequent version.

I Additional Ablation: Trunk-Decoder after Fusion Trunk

To test whether the Cross-Modal Semantic Fusion Trunk can directly yield the final forecasts, we construct a structural variant termed *Trunk-Decoder*. Specifically, we attach a lightweight decoder after the fusion trunk and use the trunk outputs to directly generate the H-step predictions. All other settings (data splits, input length, training objec-

Table 10: Structural comparison between Fusion Trunk and Trunk-Decoder (MSE; lower is better).

Dataset	H	Fusion Trunk	Trunk-Decoder
ETTm1	96	0.311	0.320
	192	0.357	0.363
	336	0.390	0.401
	720	0.450	0.462
Weather	96	0.157	0.162
	192	0.204	0.213
	336	0.262	0.266
	720	0.338	0.340
Electricity	96	0.143	0.150
	192	0.160	0.162
	336	0.176	0.177
	720	0.219	0.228

tive, and hyperparameters) are kept identical to the default model (*Fusion Trunk*), which follows the pathway "fusion representation → controlled injection → prediction head."

The results are summarized in Table 10. As shown, Trunk-Decoder works properly and produces reasonable forecasts, yet it is overall slightly weaker than the original Fusion Trunk pipeline across multiple datasets and horizons. This indicates that the trunk representation is indeed predictive, while the controlled injection pathway better preserves fine-grained numerical dynamics and more stably captures long-horizon structures, leading to superior overall performance.

Algorithm 1 Forward pass of **TimeSAF**

Require: Historical multivariate series $\mathbf{X} \in \mathbb{R}^{B\times L\times N}$, LLM-based prompts $\mathbf{E}^{LLM}$, fusion layer indices $\{\kappa_s\}_{s=1}^{S}$, refinement layer index set $\mathcal{R}$, parameters of TimeSAF.
Ensure: Forecast $\hat{\mathbf{Y}} \in \mathbb{R}^{H\times N}$.

$\mathbf{X}_{norm} \leftarrow \text{RevIN}(\mathbf{X}, \texttt{"norm"})$
$\mathcal{H}_0^{Time} \leftarrow \text{TimeSeriesEncoder}(\mathbf{X}_{norm})$ ▷ patching + projection + positional encoding
$\mathcal{H}_0^{Text} \leftarrow \text{PromptEncoder}(\mathbf{E}^{LLM})$ ▷ LLM projection + positional encoding
$s \leftarrow 1$; $\mathcal{F}^{(s)} \leftarrow \texttt{None}$
for $\ell = 1$ to dp **do**
  if $\ell \in \mathcal{R}$ **and** $\mathcal{F}^{(s)} \neq \texttt{None}$ **then** ▷ asynchronous semantic refinement
    $\mathcal{H}_\ell^{Time} \leftarrow \text{RefiningBlock}_\ell^{Time}\big(\mathcal{H}_{\ell-1}^{Time}, \mathcal{F}^{(s)}\big)$
    $\mathcal{H}_\ell^{Text} \leftarrow \text{RefiningBlock}_\ell^{Text}\big(\mathcal{H}_{\ell-1}^{Text}, \mathcal{F}^{(s)}\big)$
  else ▷ pure unimodal encoding
    $\mathcal{H}_\ell^{Time} \leftarrow \text{UnimodalBlock}_\ell^{Time}(\mathcal{H}_{\ell-1}^{Time})$
    $\mathcal{H}_\ell^{Text} \leftarrow \text{UnimodalBlock}_\ell^{Text}(\mathcal{H}_{\ell-1}^{Text})$
  end if
  if $\ell = \kappa_s$ **then** ▷ stage-wise semantic fusion (bottom-up)
    $\mathcal{H}_{s,0}^{F} \leftarrow \text{Repeat}\big(\mathcal{Q}_s^{F}, B \times N\big)$
    $\mathcal{F}^{(s)} \leftarrow \text{FusionBlock}_s\big(\mathcal{H}_{s,0}^{F}, \mathcal{H}_\ell^{Time}, \mathcal{H}_\ell^{Text}\big)$
    $s \leftarrow s + 1$
  end if
end for
$\mathbf{Y} \leftarrow \text{OutputHead}(\mathcal{H}_{dp}^{Time})$ ▷ flatten patches + linear projection
$\hat{\mathbf{Y}} \leftarrow \text{RevIN}(\mathbf{Y}, \texttt{"denorm"})$
return $\hat{\mathbf{Y}}$