M^{2}TSF: LLM-EMPOWERED TIME SERIES FORECASTING VIA MULTI-STREAM MULTI-SCALE MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series forecasting (TSF) demands capturing multi-scale temporal dynamics that range from short-term fluctuations to long-term evolving trends. Recent advances have witnessed the growing success of Large Language Models (LLMs) in TSF, particularly in capturing multi-scale temporal dynamics. However, existing methods remain constrained by a single-stream architecture that concatenates multi-scale features into a unified sequence, inevitably entangling heterogeneous temporal patterns and introducing cross-scale noise interference. To address this, we propose **M^{2}TSF**, an LLM-empowered **T**ime **S**eries **F**orecasting framework via **M**ulti-stream **M**ulti-scale modeling, which assigns each temporal scale an independent processing pathway within the LLM. Specifically, a multi-scale tokenization module is designed to obtain the multi-scale representations. Then, a constrained intermediate interaction (CII) module is designed to enable controlled and progressive cross-scale information exchange aligned with the hierarchical semantics of LLM layers, integrating fine-grained features at lower layers and high-level global semantics at deeper layers. In addition, a pattern-aware fusion (PAF) module is introduced to emphasize informative streams while mitigating interference from irrelevant ones via a hyperedge-centric mixture-of-experts (HMoE) mechanism. Extensive experiments on 17 real-world benchmarks across diverse domains demonstrate that M^{2}TSF achieves state-of-the-art performance. Code is available at: https://anonymous.4open.science/r/M2TSF.

1 INTRODUCTION

Time series forecasting (TSF) has become indispensable for various real-world applications, including weather forecasting (Wu et al., 2024; Geng et al., 2025; Bodnar et al., 2025), traffic prediction (Chen et al., 2023a; Wang et al., 2024b), and medical diagnostics (Xue et al., 2022; Wang et al., 2024c). In these real-world applications, the observed time series often manifest complex and heterogeneous multi-scale temporal dynamics (Shang et al., 2024a; 2026). Specifically, time series data contain diverse temporal patterns spanning multiple time scales, ranging from short-term fluctuations and periodic seasonal patterns to long-term evolving trends. Capturing multi-scale temporal dynamics is crucial, as interactions across different time scales often reveal the underlying evolutionary mechanisms of the system.

Benefiting from the powerful representation capabilities of neural networks, deep learning methods, e.g., Graph Neural Networks (GNNs) (Chen et al., 2023a; Cai et al., 2024), Recurrent Neural Networks (RNNs) (Chen et al., 2021), Multi-Layer Perceptrons (MLPs) (Hu et al., 2025; Zeng et al., 2022; Wang et al., 2023), and Transformers (Liu et al., 2023; Shang et al., 2024a), have advanced TSF. Recently, by leveraging the profound sequential modeling and relational reasoning capabilities in pre-trained Large Language Models (LLMs), LLM-based time series forecasting methods (Zhou et al., 2023; Bian et al., 2024; Jin et al., 2024) have achieved significant breakthroughs and reshaped the TSF landscape. To model multi-scale temporal dynamics, existing methods (Pan et al., 2024; Shang et al., 2026) integrate multi-scale analysis into LLMs and achieve promising results.

Although LLM-based methods achieve impressive performance, they are challenged by the single-stream architecture. Specifically, they concatenate multi-scale features into a unified sequence, leading to the entanglement of heterogeneous multi-scale temporal dynamics and introducing irrelevant noise interference from other scales. To alleviate this issue, a straightforward approach is to introduce a

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.

multi-stream architecture, as explored in computer vision (CV) (Zhang & Patel, 2018; Simonyan & Zisserman, 2014) and natural language processing (NLP) (Burtsev & Rumshisky, 2021). In this paradigm, features at each scale are treated as an independent semantic stream, enabling parallel and independent processing within the LLM.

However, converting the model architecture from single-stream to multi-stream is non-trivial, as two main problems need to be addressed. (1) **Intermediate interaction bottleneck**: Although each stream encapsulates information at a distinct scale, these streams are not isolated but rather maintain intricate, latent dependencies. However, existing methods typically adopt a one-shot fusion strategy (Li et al., 2021; Wang et al., 2024a) that merges all scales at a fixed layer, neglecting the progressive nature of feature interaction across scales. This often leads to cross-scale feature contamination and may even induce semantic hallucinations in LLMs (Bai et al., 2024; Shu et al., 2025; Tjio et al., 2022). Therefore, developing a progressive and selective interaction mechanism that enables controlled information exchange while preserving scale-specific characteristics remains a challenge in the multi-stream architecture. (2) **Late fusion bottleneck**: Given the heterogeneous temporal dynamics of different series, the contribution of each stream to the final forecast is inherently unequal (Shi et al., 2025; Hu et al., 2025). Densely aggregating all streams without explicitly filtering out irrelevant ones may introduce noise and dilute salient features, making adaptive stream selection and fusion another key challenge.

Motivated by the above, we propose **M^{2}TSF**, an LLM-empowered **T**ime **S**eries **F**orecasting framework via **M**ulti-stream **M**ulti-scale modeling. To the best of our knowledge, M^{2}TSF is the first work to shift the LLM-based time series forecasting paradigm from single-stream processing to a multi-stream framework, providing an independent processing pathway for temporal features at each scale. Our primary contributions are summarized as follows:

- We introduce a constrained intermediate interaction (CII) module to perform controlled and progressive cross-scale information exchange, which progressively integrates fine-grained features at lower layers and high-level global semantics at deeper layers, thereby deliberating semantic discrepancies across scales and preventing cross-scale feature contamination.
- We design a pattern-aware fusion (PAF) module to adaptively select and combine multi-stream representations via a hyperedge-centric mixture-of-experts (HMoE) mechanism, which can achieve sparse expert activation to emphasize informative streams while mitigating interference from irrelevant ones.
- We conduct experiments on 17 real-world datasets, and the experimental results demonstrate that M^{2}TSF achieves state-of-the-art (SOTA) performance.

2 Related Work

Time Series Forecasting. Deep learning has driven significant advances in time series forecasting (TSF)Hu et al. (2025); Zeng et al. (2022); Wu et al. (2022). Early works leverage Recurrent Neural Networks (RNNs) Chen et al. (2021); Salinas et al. (2020); Lai et al. (2018) and Temporal Convolutional Networks (TCNs) Chen et al. (2020); Bai et al. (2018) to capture sequential dependencies. The advent of Transformer-based models further introduces attention mechanisms to TSF Wu et al. (2021); Liu et al. (2023); Shang et al. (2024a), enabling long-range dependency modeling. To capture multi-scale temporal dynamics, recent methods have integrated multi-scale analysis with deep learning architectures. For example, PathFormer Chen et al. (2023b) achieves competitive performance by combining patches of different sizes with Transformers. MSGNet Cai et al. (2024) combines frequency analysis with graph neural networks (GNNs) to capture multi-scale inter-series correlations, while Ada-MSHyper Shang et al. (2024a) incorporates multi-scale hypergraph structures into a Transformer backbone to capture higher-order multi-scale pattern interactions. Despite their effectiveness, these methods are trained from scratch on limited task-specific data, lacking the rich representational capacity and transferable knowledge essential for generalizing across diverse time series forecasting scenarios.

LLM-Based Time Series Forecasting. Recently, large language models (LLMs) pre-trained on massive corpora have shown promising potential for time series forecasting, owing to their rich representational capacity and transferable knowledge of sequential patterns Shang et al. (2026); Pan et al. (2024); Jin et al. (2024). FPT Zhou et al. (2023) shows that a frozen LLM with only the

embedding and projection layers tuned can serve for general-purpose time series analysis. Time-LLM Jin et al. (2024) further advances this direction, which reprograms time series into text prototypes and prefixes the input with natural language prompts that convey task instructions. To empower LLMs with multi-scale modeling capabilities, recent works have explored injecting multi-scale temporal structures into LLM-based frameworks. For example, LLM4TS Chang et al. (2024) introduces multi-scale temporal encoding during the fine-tuning stage to capture multi-scale pattern interactions. MSH-LLM Shang et al. (2026) leverages a multi-scale hypergraph to enrich temporal semantics and aligns the resulting features with text prototypes before feeding them into a frozen LLM. However, these multi-scale LLM-based methods predominantly adopt a single-stream architecture that concatenates multi-scale features into a unified sequence, which can lead to entanglement of heterogeneous temporal patterns and noise interference across scales.

Multi-Stream Architectures. Multi-stream architectures, which process inputs through parallel and independent pathways, have demonstrated strong advantages in computer vision and natural language processing. In CV, the pioneering two-stream network Simonyan & Zisserman (2014) processes spatial appearance and temporal motion in separate streams, enabling effective action recognition. DID-MDN Zhang & Patel (2018) extends this idea to image restoration, using independent streams to decompose different degradation components. In NLP, multi-stream Transformer Burtsev & Rumshisky (2021) explores parallel semantic pathways to reduce information bottlenecks within attention mechanisms. SUTRA Bendale et al. (2024) further introduces a dual-stream LLM architecture that separates language-agnostic concept reasoning from language-specific generation, enabling effective multilingual knowledge sharing.

However, achieving multi-stream multi-scale modeling in LLM-based time series forecasting is non-trivial, as naïve cross-stream interactions risk cross-scale feature contamination, and dense stream aggregation may dilute scale-specific salient features. Unlike previous multi-scale LLM-based methods, we shift the LLM-based forecasting paradigm from single-stream to multi-stream modeling. We assign each scale an independent processing pathway, subsequently designing modules for progressive cross-scale interaction and adaptive stream fusion.

3 PRELIMINARIES

Problem Formulation. Given the time series with D dimensions, we denote the historical observations as $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T] \in \mathbb{R}^{T \times D}$, where T is the look-back window length and $\boldsymbol{x}_t \in \mathbb{R}^D$ represents the temporal features of the t-th time steps. The goal of time series forecasting is to predict the future values $\hat{\boldsymbol{Y}} \in \mathbb{R}^{F \times D}$ over a forecast horizon of F time steps.

Hypergraph. A hypergraph extends conventional graphs by allowing each hyperedge to connect an arbitrary number of nodes. Formally, let $\mathcal{V} = \{v_1, \ldots, v_N\}$ be a set of nodes and $\mathcal{E} = \{e_1, \ldots, e_M\}$ be a set of hyperedges. The hypergraph is then denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The topology of the hypergraph is encoded by an incidence matrix $\boldsymbol{\mathcal{H}} \in \mathbb{R}^{N \times M}$, with $\boldsymbol{\mathcal{H}}_{nm} = 1$ indicating that node v_n is connected by hyperedge e_m, and $\boldsymbol{\mathcal{H}}_{nm} = 0$ otherwise. Based on $\boldsymbol{\mathcal{H}}$, we define the node degree $d(v_n) = \sum_{m=1}^{M} \boldsymbol{\mathcal{H}}_{nm}$ and the hyperedge degree $d(e_m) = \sum_{n=1}^{N} \boldsymbol{\mathcal{H}}_{nm}$, with the corresponding diagonal degree matrices denoted as $\mathbf{D}_\text{v} \in \mathbb{R}^{N \times N}$ and $\mathbf{D}_\text{e} \in \mathbb{R}^{M \times M}$, respectively.

4 METHODOLOGY

As shown in Figure 1, M^{2}TSF is designed to shift the LLM-based time series forecasting paradigm from single-stream processing to a multi-stream framework, which consists of three modules: (1) **Multi-Scale Tokenization Module**, which generates multi-scale representations to capture temporal dynamics across varying granularities; (2) **Constrained Intermediate Interaction (CII) Module**, which processes the S streams through a modified GPT-2 (Radford et al., 2019) with integrated hypergraph neural network (GNN) layers for cross-stream interaction; and (3) **Pattern-Aware Fusion (PAF) Module**, which adaptively aggregates multi-scale features via a hyperedge-centric mixture-of-experts (HMoE) mechanism to produce the final forecast.

4.1 MULTI-SCALE TOKENIZATION MODULE

As shown in Figure 1(a), the multi-scale tokenization module is designed to capture temporal dynamics across varying granularities. Specifically, the input time series is first decomposed into multi-scale representations, which are subsequently tokenized into patch sequences. Then, to further

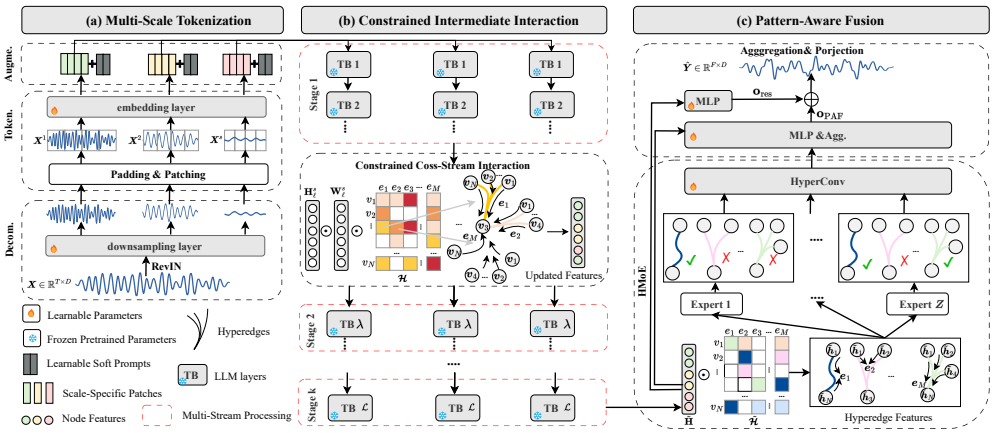

Figure 1: The framework of M^{2}TSF.

bridge the inherent modality gap between time series data and LLMs, we introduce a soft prompt augmentation strategy that enriches the input embeddings with learnable, scale-specific prompts.

Multi-Scale Decomposition. Given an input time series $\mathbf{X}^1 = \mathbf{X} \in \mathbb{R}^{T \times D}$, we first normalize it through reversible instance normalization (RevIN) (Kim et al., 2021; Shang et al., 2026; Liu et al., 2025). Then, we construct a hierarchy of S temporal scales $\{\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^S\}$ through successive convolution-based downsampling:

$$\mathbf{X}^s = \mathrm{ELU}\Big(\mathrm{BN}\Big(\mathrm{Conv1D}(\mathbf{X}^{(s-1)}; w^s)\Big)\Big) \in \mathbb{R}^{T^s \times D}, s \geq 2, \tag{1}$$

where $s \in \{2, \ldots, S\}$ denotes the scale index. ELU and BN represent the ELU activation and batch normalization, respectively. $\mathrm{Conv1D}(\cdot\,; w^s)$ denotes a 1D convolution with kernel size equal to w^s. $\mathbf{X}^s \in \mathbf{X}$ denotes the downsampled sequence at scale s. $T^s = \lfloor T^{s-1}/w^s \rfloor$ represents the downsampled sequence length at scale s.

Patch Tokenization. For each scale s, we segment the downsampled sequence $\mathbf{X}^s$ into patches with scale-adaptive patch size $P^s = P/w^s$ and stride $\Omega^s = \Omega/w^s$, where P and Ω are the base patch size and stride at the finest scale. Padding operations (Chen et al., 2023b; Nie et al., 2022) are applied to ensure full coverage of the sequence. The resulting patch sequence at scale s is:

$$\mathbf{E}^s = \mathrm{Unfold}(\mathrm{Pad}(\mathbf{X}^s); P^s, \Omega^s) \in \mathbb{R}^{N^s \times P^s \times D}, \tag{2}$$

where $N_s = \lfloor (T/w^s - P^s)/\Omega^s \rfloor + 1$ is the number of patches at scale s. Each patch sequence is then projected into the LLM's hidden dimension $\mathcal{D}$ through a learnable embedding layer:

$$\hat{\mathbf{E}}^s = \mathrm{Embed}(\mathbf{E}^s) \in \mathbb{R}^{N^s \times \mathcal{D} \times D}. \tag{3}$$

Soft Prompt Augmentation. To provide scale-specific contextual cues and bridge the modality gap between time series patches and the LLM, we append a set of learnable soft prompt (Sun et al., 2024) tokens $\mathbf{P}^s \in \mathbb{R}^{L^s \times d}$ to each scale-specific patch sequence:

$$\tilde{\mathbf{E}}^s = [\hat{\mathbf{E}}^s; \mathbf{P}^s] \in \mathbb{R}^{(N^s + L^s) \times \mathcal{D} \times D}, \tag{4}$$

where L^s is the number of soft prompt tokens at scale s and $[\cdot\,;\cdot]$ denotes concatenation along the sequence dimension.

4.2 Constrained Intermediate Interaction (CII) Module

Unlike existing LLM-based methods (Shang et al., 2026; Pan et al., 2024) that concatenate multi-scale features into a single unified sequence, M^{2}TSF assigns each scale s an independent processing stream within the LLM. However, naïvely performing multi-scale interactions across streams may introduce semantic hallucinations (Bai et al., 2024; Shu et al., 2025; Tjio et al., 2022), where inconsistent

cross-scale signals lead to spurious temporal patterns. Prior works (Jawahar et al., 2019; Liu et al., 2024b) on probing the internal layers of LLMs have revealed that different layers encode distinct levels of representations: lower layers primarily capture fine-grained features, whereas deeper layers progressively encode high-level global semantic information. Motivated by this observation, we introduce a CII module that enables controlled and progressive cross-scale information exchange aligned with the inherent hierarchical semantics of the LLM.

Multi-Stream Processing. Given the multi-scale patch sequences $\tilde{\mathbf{E}} = \{\tilde{\mathbf{E}}^1, \tilde{\mathbf{E}}^2, \ldots, \tilde{\mathbf{E}}^S\}$, M^{2}TSF treats each $\tilde{\mathbf{E}}^s$ as an independent stream within the LLM to capture scale-specific features. Specifically, we partition the L LLM layers into Λ consecutive stages $\{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_\Lambda\}$, where $\bigcup_{\lambda=1}^{\Lambda} \mathcal{L}_\lambda = \{1, \ldots, L\}$. Within each stage, all streams are processed independently:

$$\mathbf{H}_\ell^s = \mathrm{TransformerBlock}_\ell\big(\mathbf{H}_{\ell-1}^s\big), \quad \forall\, \ell \in L, \;\; s = 1, \ldots, S, \tag{5}$$

where $\mathbf{H}_1^s = \tilde{\mathbf{E}}^s$. Each stream allows scale-specific patch sequences to be modeled without interference from other scales.

Constrained Cross-Stream Interaction. During the cross-stream interaction process, we perform constrained cross-stream interaction to achieve controlled and progressive interactions. As shown in Figure 1(b), we first construct a hypergraph structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to model high-order cross-stream interactions (i.e., simultaneous interactions between multiple streams). The node set $\mathcal{V}$ corresponds to patch tokens across all scales, and each hyperedge $e_m \in \mathcal{E}$ connects a subset of nodes. Then, we design an incidence matrix to indicate the hypergraph structure:

$$\boldsymbol{\mathcal{H}} = SoftMax(SoftPlus(\mathbf{E}_{\text{node}}(\mathbf{E}_{\text{hyper}})^T)), \tag{6}$$

where $\boldsymbol{E}_{\text{node}} \in \mathbb{R}^{N \times D}$ and $\boldsymbol{E}_{\text{hyper}} \in \mathbb{R}^{M \times D}$ are learnable parameters, representing the node embeddings and hyperedge embeddings, respectively. Then, we introduce a sparsity strategy to select the $TopK$ nodes for each hyperedge:

$$\boldsymbol{\mathcal{H}}_{nm} = \begin{cases} \boldsymbol{\mathcal{H}}_{nm}, & \boldsymbol{\mathcal{H}}_{nm} \in TopK(\boldsymbol{\mathcal{H}}_{*m}, \eta) \\ 0, & \boldsymbol{\mathcal{H}}_{nm} \notin TopK(\boldsymbol{\mathcal{H}}_{*m}, \eta) \end{cases}, \tag{7}$$

where η is the threshold of the $TopK$ function and denotes the max number of nodes connected by a hyperedge. Then, to enable controlled and progressive information exchange between streams, we introduce a scale-depth alignment weighting in the hypergraph structure:

$$w_\ell^s = \frac{\exp(-\beta(p_{\ell_\lambda} - q^s)^2)}{\sum_s \exp(-\beta(p_{\ell_\lambda} - q^s)^2)}, \tag{8}$$

where $p_{\ell_\lambda} = \frac{\ell_\lambda - 1}{L-1}$ and $q_s = \frac{s-1}{S-1}$. $\ell_\lambda = max(\mathcal{L}_\lambda)$ denotes the last layer of stage $\mathcal{L}_\lambda$. This weighting encourages high-order interactions to focus on the most relevant temporal scales, enabling progressive feature abstraction from fine-grained patterns in shallow layers to coarse-grained semantics in deeper layers. Then, the hyeredge features at scale s can be represented as follows:

$$\hat{\mathbf{H}}_{\ell_\lambda}^s = w_{\ell_\lambda}^s \cdot \mathbf{H}_{\ell_\lambda}^s. \tag{9}$$

Based on the incidence matrix $\boldsymbol{\mathcal{H}}$ and hyperedge features $\hat{\mathbf{H}}_{\ell_\lambda} = \{\hat{\mathbf{H}}_{\ell_\lambda}^1, \hat{\mathbf{H}}_{\ell_\lambda}^2, \ldots, \hat{\mathbf{H}}_{\ell_\lambda}^S\}$, we use the hypergraph convolution attention (Shang et al., 2024a;b) to capture high-order multi-stream interactions:

$$\tilde{\mathbf{H}}_{\ell_\lambda} = \bigoplus_{j=1}^{J} (\sigma(\mathbf{D}_v^{-1/2} \boldsymbol{\mathcal{H}}_j \mathbf{D}_e^{-1} (\boldsymbol{\mathcal{H}}_j)^\top \mathbf{D}_v^{-1/2} \hat{\mathbf{H}}_{\ell_\lambda} \mathbf{P}_j)) \in \mathbb{R}^{N \times \mathcal{D} \times D}, \tag{10}$$

where $\tilde{\mathbf{H}}_{\ell_\lambda}$ denotes the updated hyperedge features and then fed to the next stage, i.e., $\mathbf{H}_{\ell_\lambda} \leftarrow \tilde{\mathbf{H}}_{\ell_\lambda}$. J is the number of attention heads, and $\mathbf{H}_j$ is the hypergraph incidence matrix for the j-th head. $\mathbf{D}_v$ and $\mathbf{D}_e$ are the diagonal node degree and hyperedge degree matrices, respectively. $\mathbf{P}_j \in \mathbb{R}^{D \times D}$ is a learnable projection matrix for the j-th head, $\sigma(\cdot)$ denotes a nonlinear activation function, and $\bigoplus$ is an aggregation operator (e.g., concatenation or average pooling) that combines the outputs across all heads. This design effectively mitigates cross-scale feature contamination while preserving scale-specific characteristics. The final output of the CII module can be represented as $\tilde{\mathbf{H}} = \{\tilde{\mathbf{H}}^1, \ldots \tilde{\mathbf{H}}^S\}$.

4.3 PATTERN-AWARE FUSION (PAF) MODULE

After obtaining the multi-stream representations, a key challenge arises during the later fusion stage: Given the heterogeneous temporal dynamics captured by different streams, naïvely aggregating all streams in a dense manner can introduce noise and dilute salient features. To address this, we propose the Pattern-Aware Fusion (PAF) module, which adaptively selects and combines multi-stream representations through a hyperedge-centric mixture-of-experts (HMoE) mechanism.

HMoE mechanism. As shown in Figure 1(c), to achieve sparse and adaptive stream fusion, we employ a HMoE mechanism that routes hyperedges to specialized expert networks. Specifically, we first obtain the hyperedge features by computing the mean of their connected node features:

$$\mathbf{e}_m = \frac{1}{N} \sum_{v_n \in e_m} \tilde{\mathbf{h}}_n \in \mathbb{R}^{\mathcal{D} \times D}, \quad m = 1, \ldots, |\mathcal{E}|, \tag{11}$$

where $\tilde{\mathbf{h}}_n \in \tilde{\mathbf{H}}$ represents the features of the n-th node. Then, the gating mechanism computes a routing score over Z experts for each hyperedge:

$$\mathbf{g}_m = \text{Softmax}(\mathbf{W}_g \mathbf{e}_m + \xi \cdot \text{Softplus}(\mathbf{W}_n \mathbf{e}_m)) \in \mathbb{R}^Z, \tag{12}$$

where $\mathbf{W}_g, \mathbf{W}_n \in \mathbb{R}^{Z \times \mathcal{D}}$ are learnable parameters, and $\xi \sim \mathcal{N}(0, 1)$ is Gaussian noise injected during training to encourage load balancing and exploration (Hu et al., 2025). Based on the routing score $\mathbf{g}_j$, we apply $TopK$ selection from the expert perspective. Specifically, each expert $z \in Z$ selects the $TopK$ hyperedges with the highest routing scores. The sparse gate is constructed by zeroing out non-selected entries:

$$e_{j,z} = \begin{cases} e_{j,z} & \text{if } e_{j,z} \in TopK(\mathcal{E}, \gamma) \\ 0 & \text{otherwise} \end{cases}, \tag{13}$$

where $\gamma = \max(\lfloor |\mathcal{E}| \cdot c/Z \rfloor, 1)$ is the threshold of the $TopK$ function and c is a capacity factor. For each expert z, we construct a subgraph based on the selected $TopK$ hyperedges and perform the hypergraph convolution operation:

$$\mathbf{H}' = \text{LayerNorm}\Big(\text{HyperConv}(\tilde{\mathbf{H}}, \mathcal{G}_z) + \tilde{\mathbf{H}}\Big) \in \mathbb{R}^{\mathcal{D} \times D}, \tag{14}$$

where $\mathcal{G}_z$ denotes the subgraph induced by the hyperedges assigned to expert z. The refined node features are then aggregated at the hyperedge level and passed through linear mapping:

$$\hat{\mathbf{y}}_{m,z} = \text{MLP}\left(\frac{1}{N} \sum_{v_n \in e_m} \mathbf{h}'_n\right) \in \mathbb{R}^{F \times D}, \quad m \in TopK, \tag{15}$$

where $\hat{\mathbf{y}}_{m,z}$ denotes the m-th aggregated hyperedge features for the z-th expert. F denotes the forecasting length. MLP is the linear mapping function. $\tilde{\mathbf{h}}_n \in \tilde{\mathbf{H}}$ represents the updated features of the n-th node. The expert predictions are combined via the sparse gates:

$$\hat{\mathbf{y}}_m = \sum_{z=1}^{Z} \tilde{g}_{m,z} \cdot \hat{\mathbf{y}}_{m,z} \in \mathbb{R}^{F \times D}. \tag{16}$$

Weighted Aggregation. Finally, we aggregate the hyperedge-level predictions into a single forecast via weighted aggregation:

$$\mathbf{o}_{\text{PAF}} = \sum_{j=1}^{|\mathcal{E}|} \alpha_j \cdot \hat{\mathbf{y}}_m, \quad \alpha_j = \frac{\exp(\mathbf{w}^\top \mathbf{y}_m)}{\sum_{m'} \exp(\mathbf{w}^\top \mathbf{h}_{m'})}, \tag{17}$$

where $\mathbf{w} \in \mathbb{R}^{d'}$ is a learnable importance vector. The output is further modulated by a sigmoid gate $\sigma(\cdot)$ to control the information flow:

$$\mathbf{o}_{\text{PAF}} \leftarrow \sigma(\mathbf{W}_{\text{gate}} \mathbf{o}_{\text{PAF}}) \odot \mathbf{o}_{\text{PAF}}, \tag{18}$$

where $\mathbf{W}_{\text{gate}} \in \mathbb{R}^{F \times F}$ are learnable parameters and $\odot$ denotes element-wise multiplication.

Output Projection. In addition to the PAF output, we also compute a residual prediction from the finest-scale stream, which operates on the original input and preserves the richest temporal details. The last finest-scale outputs of the CII module are reshaped and projected through a linear mapping:

$$\mathbf{o}_{\text{res}} = \text{MLP}(\text{FlattenHead}(\tilde{\mathbf{H}}_L^1[N:,])) \in \mathbb{R}^{F \times D}. \tag{19}$$

The final prediction is obtained by combining the PAF output and the residual prediction:

$$\tilde{\mathbf{Y}} = \lambda \cdot \mathbf{o}_{\text{PAF}} + \mathbf{o}_{\text{res}}, \tag{20}$$

where λ is a tunable coefficient that balances the contributions from the multi-stream fusion and the finest-scale residual pathway. After obtaining the prediction, we apply the inverse normalization to recover the original scale: $\hat{\mathbf{Y}} \leftarrow \tilde{\mathbf{Y}} \cdot \sigma + \mu$.

5 Experiments

5.1 Experimental Protocol

Datasets. For long-term and few-shot forecasting, we evaluate M^{2}TSF on seven widely-used real-world benchmarks spanning diverse domains, including ETT (ETTh1, ETTh2, ETTm1, and ETTm2), Electricity, Traffic, and ILI. For zero-shot forecasting, we further evaluate M^{2}TSF on the M3 (4 subsets) and M4 (6 subsets) collections, yielding 17 benchmark datasets in total.

Baselines. We compare M^{2}TSF against a comprehensive set of state-of-the-art baselines, including LLM4TS methods: MSH-LLM (Shang et al., 2026), S^{2}IP-LLM (Pan et al., 2024), and Time-LLM (Jin et al., 2024); Linear-based methods: AMD (Hu et al., 2025), DLinear (Zeng et al., 2022), and N-HiTS (Challu et al., 2023); CNN-based methods: TimesNet (Wu et al., 2022); Transformer-based methods: iTransformer (Liu et al., 2023), PatchTST (Nie et al., 2022), and Autoformer (Wu et al., 2021).

Experimental Settings. All experiments are conducted on a single NVIDIA RTX PRO 6000 GPU (96GB). GPT-2 (Radford et al., 2019) is used as the default LLM backbone of M^{2}TSF. Following existing methods (Shang et al., 2026; Liu et al., 2023; Pan et al., 2024), the look-back window length T is set to 512. For both long-term and few-shot time series forecasting, performance is evaluated over four prediction horizons: $\{24, 36, 48, 60\}$ for ILI and $\{96, 192, 336, 720\}$ for the other datasets. The best results are **bolded** and the second best results are underlined. Mean Squared Error (MSE) and Mean Absolute Error (MAE) are adopted as the evaluation metrics. Lower values mean better performance. Detailed descriptions of the datasets, baselines, and experimental settings are provided in Appendices A, B, and C, respectively.

5.2 Main Results

Long-term forecasting. Table 1 presents the long-term time series forecasting results. We can observe that: (1) LLM4TS methods (i.e., MSH-LLM, S^{2}IP-LLM, and Time-LLM) consistently outperform other deep learning-based methods (i.e., ASHyper, iTransformer, and MSHyper), demonstrating that the strong sequential modeling capacity and long-range dependency modeling capability acquired by LLMs during pre-training can be successfully adapted to long-horizon time series forecasting. (2) By leveraging multi-scale modeling, MSH-LLM and S^{2}IP-LLM outperform other LLM4TS methods. However, their performance is bottlenecked by the single-stream designs. (3) By achieving the multi-stream architecture and further strengthening it through intermediate interaction and late fusion, M^{2}TSF effectively addresses the above bottleneck and achieves SOTA performance across nearly all benchmarks, reducing the forecasting error by an average of 2.40% and 2.92% over the second-best baseline in terms of MSE and MAE, respectively.

Few-shot forecasting. We evaluate M^{2}TSF under few-shot forecasting settings with 10% and 5% training data. Table 2 shows results under 10% training data. M^{2}TSF delivers the best results in almost all cases, with average error reductions of 10.63% in MSE and 6.92% in MAE compared to representative LLM4TS baselines (i.e., MSH-LLM, S^{2}IP-LLM, and Time-LLM). The reason is that our PAF module can emphasize informative streams while mitigating interference from irrelevant scales even under data-scarce scenarios. The results under 5% training data is given in Appendix E.2.

Table 1: Long-term time series forecasting results. Results are averaged from all forecasting lengths. Full results are listed in Appendix E.1.

Methods	M^{2}TSF (Ours)		MSH-LLM (ICLR 2026)		S^{2}IP-LLM (ICML 2024)		Time-LLM (ICLR 2024)		AutoTimes (NeurIPS 2024)		FPT (NeurIPS 2023)		ASHyper (NeurIPS 2024)		iTransformer (ICLR 2024)		MSHyper (arXiv 2024)		DLinear (AAAI 2023)		TimesNet (ICLR 2023)		FEDformer (ICML 2022)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ILI	**1.334**	**0.731**	1.373	0.758	1.552	0.826	1.713	0.858	1.642	0.838	1.925	0.903	1.851	0.890	2.073	0.941	1.973	0.948	2.169	1.041	2.139	0.931	2.847	1.144
ECL	**0.158**	**0.248**	0.159	0.253	0.163	0.258	0.165	0.261	0.162	0.261	0.167	0.263	0.162	0.253	0.203	0.298	0.176	0.276	0.166	0.264	0.193	0.295	0.214	0.327
Traffic	0.389	**0.260**	**0.381**	0.283	0.406	0.287	0.408	0.291	0.397	0.289	0.414	0.295	0.391	0.289	0.384	0.295	0.393	0.317	0.434	0.295	0.620	0.336	0.610	0.376
ETTh1	**0.384**	**0.416**	0.402	0.420	0.405	0.426	0.414	0.435	0.405	0.437	0.418	0.431	0.416	0.428	0.451	0.462	0.429	0.437	0.419	0.439	0.520	0.503	0.440	0.460
ETTh2	**0.312**	**0.368**	0.342	0.383	0.348	0.392	0.355	0.398	0.358	0.387	0.367	0.402	0.351	0.392	0.382	0.414	0.367	0.393	0.502	0.481	0.425	0.451	0.437	0.449
ETTm1	**0.338**	**0.370**	0.340	0.371	0.343	0.380	0.350	0.383	0.355	0.380	0.355	0.386	0.355	0.381	0.370	0.399	0.388	0.385	0.357	0.380	0.400	0.418	0.448	0.452
ETTm2	**0.248**	**0.306**	0.252	0.311	0.257	0.319	0.272	0.332	0.258	0.347	0.264	0.328	0.263	0.322	0.272	0.331	0.277	0.326	0.276	0.341	0.305	0.355	0.305	0.349

Table 2: Few-shot forecasting results with 10% of the training data. Results are averaged from all forecasting lengths. Full results are listed in Appendix E.2.

Methods	M^{2}TSF (Ours)		MSH-LLM (ICLR 2026)		S^{2}IP-LLM (ICML 2024)		Time-LLM (ICLR 2024)		FPT (NeurIPS 2023)		iTransformer (ICLR 2024)		PatchTST (ICLR 2023)		TimesNet (ICLR 2023)		FEDformer (ICML 2022)		NSFormer (NeurIPS 2022)		ETSformer (arXiv 2022)		Autoformer (NeurIPS 2021)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.518**	**0.475**	0.563	0.514	0.593	0.529	0.785	0.553	0.590	0.525	0.910	0.645	0.633	0.542	0.869	0.628	0.639	0.561	0.915	0.639	1.180	0.834	0.702	0.596
ETTh2	**0.350**	**0.394**	0.392	0.423	0.419	0.439	0.424	0.441	0.397	0.421	0.489	0.483	0.415	0.431	0.479	0.465	0.466	0.475	0.462	0.455	0.894	0.713	0.488	0.499
ETTm1	0.421	**0.416**	**0.403**	0.424	0.455	0.435	0.487	0.461	0.464	0.441	0.728	0.565	0.501	0.466	0.677	0.537	0.722	0.605	0.797	0.578	0.980	0.714	0.802	0.628
ETTm2	**0.278**	**0.326**	0.280	0.327	0.284	0.332	0.305	0.344	0.293	0.335	0.336	0.373	0.296	0.343	0.320	0.353	0.463	0.488	0.332	0.366	0.447	0.487	1.342	0.930

Zero-shot forecasting. Following existing methods (Shang et al., 2026; Liu et al., 2024a), we evaluate M^{2}TSF under zero-shot forecasting settings with M3 and M4 datasets. Specifically, M4→M3 datasets indicate training on M4 datasets and testing on M3 datasets, and vice versa. Notably, the M3 and M4 datasets exhibit distinct multi-scale patterns and distributions. As shown in Table 3, M^{2}TSF gives an average of 1.24% SMAPE error reductions over the second-best baseline, which demonstrates its robustness in handling distribution shift scenarios.

Table 3: Zero-shot results in terms of averaged SMAPE. Full results are listed in Appendix E.3.

Methods	M^{2}TSF (Ours)	MSH-LLM (ICLR 2026)	AutoTimes (NeurIPS 2024)	FPT (NeurIPS 2023)	DLinear (AAAI 2023)	PatchTST (ICLR 2023)	TimesNet (ICLR 2023)	NSformer (NeurIPS 2022)	FEDformer (ICML 2022)	Informer (AAAI 2021)	Reformer (ICLR 2019)
M4→M3	**12.454**	12.469	12.750	13.060	14.030	13.390	14.170	15.290	13.530	15.820	13.370
M3→M4	**12.631**	12.968	13.036	13.125	15.337	13.228	14.553	14.327	15.047	19.047	14.092

5.3 Ablation Studies

LLM backbones. We design variants to investigate the impact of LLM backbones (i.e., GPT2-small, GPT-medium, LLaMA2-7B (Touvron et al., 2023), and Qwen2.5-7B) on the ETTh1 dataset. As shown in Table 4, scaling up the backbone generally leads to improved forecasting performance, which aligns with the scaling law observed in NLP (Kaplan et al., 2020). For a fair comparison, we follow existing methods and adopt GPT2-small as the default LLM backbone.

Table 4: Results of different LLM backbones.

Methods	GPT-S		GPT-M		L-7B		Q-7B	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
96	0.345	0.383	**0.343**	**0.382**	0.356	0.391	0.354	0.389
192	0.384	0.409	0.386	0.413	**0.380**	**0.406**	0.383	0.409
336	0.391	0.423	0.389	0.420	**0.384**	**0.417**	0.386	0.418

CII Module. We conduct ablation studies to verify the effectiveness of the CII module: (1) Replacing the multi-stream architecture with the single-stream design, in which token sequences from all scales are concatenated and jointly fed into the LLM (-single). (2) Replacing the adaptive hypergraph with an adaptive graph (-graph). (3) Removing the CII module (w/o CII). The experimental results are shown in Table 5. We can observe that M^{2}TSF performs better than -single and w/o CII, demonstrating the effectiveness of the multi-stream architecture and progressive cross-scale information exchange, respectively. In addition, -graph yields worse performance than M^{2}TSF, which can be attributed to the limited expressiveness of pairwise edges.

PAF Module. We conduct ablation studies to validate the effectiveness of the PAF module: (1) Replacing the PAF module with the naïve MLP-based implementation (-MLP). (2) Replacing the PAF module with the naïve MoE-based implementation, where tokens from different streams are directly routed to expert networks (-MoE). The experimental results are shown in Table 5, where -MLP exhibits the worst performance, showing the effectiveness of dynamic adaptive aggregation. In

addition, M^{2}TSF performs better than -MoE, demonstrating the superiority of performing aggregation and prediction centered on hyperedge-composed temporal patterns.

Table 5: Results of different variants on ETTh1 dataset.

Methods	-single		-graph		w/o CII		-MLP		-MoE		M^{2}TSF	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
96	0.350	0.387	0.349	0.388	0.352	0.390	0.356	0.394	0.352	0.391	**0.345**	**0.383**
192	0.390	0.412	0.390	0.415	0.391	0.415	0.402	0.427	0.399	0.424	**0.384**	**0.409**
336	0.395	0.426	0.403	0.432	0.407	0.436	0.446	0.463	0.439	0.456	**0.391**	**0.423**

5.4 Parameter Studies

Number of LLM Layers. The number of LLM layers L controls the depth of temporal feature extraction within the multi-stream architecture. We vary $L \in \{2, 4, 6, 8\}$ and report the results on ETTh1 dataset in Figure 2(a). The optimal performance is achieved at $w = 6$. When M is too small (i.e., $M \to 1$), the model lacks sufficient capacity to capture high-level global semantic information. When M is too large, the model tends to over-smooth representations, increasing the risk of cross-scale feature contamination.

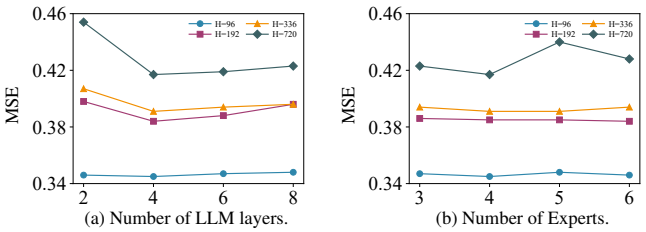

Figure 2: The impact of different hyperparameters.

Number of Experts. The number of experts Z controls the granularity of adaptive stream selection and fusion. We vary $Z \in \{3, 4, 5, 6\}$ and report the results in Figure 2(b). The optimal performance is achieved at $Z = 4$ for most cases. When Z is too small, the model loses sufficient ability to distinguish heterogeneous patterns across streams. When Z is too large, the sparse gating mechanism becomes over-fragmented, resulting in dense fusion that introduces noise from irrelevant scales.

5.5 Visualization Analysis

We conduct t-SNE visualization on the ETTh1 dataset. Figure 3 illustrates the distribution of multi-scale token embeddings from 800 randomly selected samples. Token embeddings of different scales are distinguished by different colors. Figure 3(a) shows that under the single-stream architecture, the multi-scale token embeddings are partially entangled with each other. In contrast, Figure 3(b) shows that, under the multi-stream architecture, the multi-scale token embeddings form several clusters with distinct boundaries. This highlights the superiority of the multi-stream architecture considering the entanglement of heterogeneous multi-scale temporal dynamics.

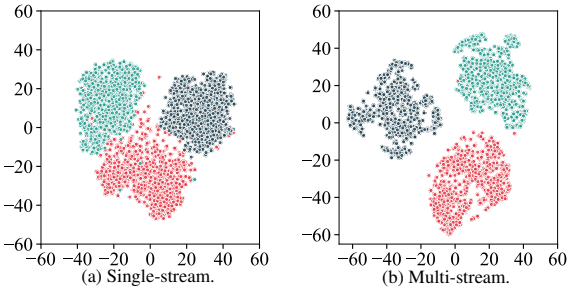

Figure 3: t-SNE visualization.

6 Conclusions

In this paper, we propose M^{2}TSF, a novel LLM-empowered time series forecasting framework that shifts the paradigm from single-stream to multi-stream multi-scale modeling. Specifically, a constrained intermediate interaction (CII) module is designed to enable progressive cross-scale information exchange aligned with the hierarchical semantics of LLM layers. A pattern-aware fusion (PAF) module is further introduced to adaptively select informative streams and suppress irrelevant ones via a hyperedge-centric mixture-of-experts mechanism. Extensive experiments on 17 real-world benchmarks demonstrate that M^{2}TSF achieves state-of-the-art performance across long-term, few-shot, and zero-shot forecasting settings with competitive efficiency.

AI USE STATEMENT

Generative AI assistance was used to aid and polish the writing of this paper. The pretrained language models used as components of M^{2}TSF are described in Section 4, with backbone comparisons in Section 5.3.

ETHICS STATEMENT

The potential societal applications and broader impacts of this work are discussed in Appendix I. The limitations of the method are described in Appendix H.

REPRODUCIBILITY STATEMENT

The model architecture and its components are described in Section 4. Dataset descriptions, baseline descriptions, experimental settings, and full numerical results are provided in Appendices A, B, C, and E, respectively. The anonymous code link is: `https://anonymous.4open.science/r/M2TSF`.

REFERENCES

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arxiv. *arXiv preprint arXiv:1803.01271*, 10, 2018.

Zechen Bai, Pichao Wang, Tianjun Xiao, Tong He, Zongbo Han, Zheng Zhang, and Mike Zheng Shou. Hallucination of multimodal large language models: A survey. *arXiv preprint arXiv:2404.18930*, 2024.

Abhijit Bendale, Michael Sapienza, Steven Ripplinger, Simon Gibbs, Jaewon Lee, and Pranav Mistry. SUTRA: Scalable multilingual language model architecture. *arXiv preprint arXiv:2405.06694*, 2024.

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.

Cristian Bodnar, Wessel P Bruinsma, Ana Lucic, Megan Stanley, Anna Allen, Johannes Brandstetter, Patrick Garvan, Maik Riechert, Jonathan A Weyn, Haiyu Dong, et al. A foundation model for the earth system. *Nature*, 641(8065):1180–1187, 2025.

Mikhail Burtsev and Anna Rumshisky. Multi-stream transformers. *arXiv preprint arXiv:2107.10342*, 2021.

Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. MSGNet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11141–11149, 2024.

Cristian Challu, Kin G Olivares, Boris N Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. NHiTS: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 6989–6997, 2023.

Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. LLM4TS: Two-stage fine-tuning for time-series forecasting with pre-trained LLMs. *arXiv preprint arXiv:2308.08469*, 2024.

Ling Chen, Donghui Chen, Zongjiang Shang, Binqing Wu, Cen Zheng, Bo Wen, and Wei Zhang. Multi-scale adaptive graph neural network for multivariate time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, pp. 10748–10761, 2023a.

Peng Chen, Yingying ZHANG, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. In *Proceedings of the International Conference on Learning Representations*, 2023b.

Yitian Chen, Yanfei Kang, Yixiong Chen, and Zizhuo Wang. Probabilistic forecasting with temporal convolutional neural network. *Neurocomputing*, 399:491–501, 2020.

Zipeng Chen, Qianli Ma, and Zhenxi Lin. Time-aware multi-scale RNNs for time series modeling. In *IJCAI*, pp. 2285–2291, 2021.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the International Conference on Machine Learning*, volume 235, pp. 10148–10167, 2024.

Zhixin Geng, Xu Fan, Xiqiao Lu, Yan Zhang, Guangyuan Yu, Cheng Huang, Qian Wang, Yuewu Li, Weichun Ma, Qi Yu, et al. FuXi-Air: Urban air quality forecasting based on emission-meteorology-pollutant multimodal machine learning. *arXiv preprint arXiv:2506.07616*, 2025.

Yifan Hu, Peiyuan Liu, Peng Zhu, Dawei Cheng, and Tao Dai. Adaptive multi-scale decomposition framework for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 17359–17367, 2025.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 3651–3657, 2019.

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. In *Proceedings of the International Conference on Learning Representations*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

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 *Proceedings of the International Conference on Learning Representations*, 2021.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *Proceedings of the International Conference on Learning Representations*, 2019.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *International ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.

Weisheng Li, Dongwen Cao, Yidong Peng, and Chao Yang. MSNet: A multi-stream fusion network for remote sensing spatiotemporal fusion based on transformer and convolution. *Remote Sensing*, 13(18):3724, 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*, volume 39, pp. 18780–18788, 2025.

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, 2022.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2023.

Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Autotimes: Autoregressive time series forecasters via large language models. *arXiv preprint arXiv:2402.02370*, 2024a.

Zhu Liu, Cunliang Kong, Ying Liu, and Maosong Sun. Fantastic semantics and where to find them: Investigating which layers of generative LLMs reflect lexical semantics. In *Findings of the Association for Computational Linguistics*, pp. 14551–14558, 2024b.

Spyros Makridakis and Michele Hibon. The m3-competition: results, conclusions and implications. *International journal of forecasting*, 16(4):451–476, 2000.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition: Results, findings, conclusion and way forward. *International Journal of forecasting*, 34(4):802–808, 2018.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *Proceedings of the International Conference on Learning Representations*, 2022.

Zijie Pan, Yushan Jiang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. S2IP-LLM: Semantic space informed prompt learning with LLM for time series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

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.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3): 1181–1191, 2020.

Zongjiang Shang, Ling Chen, Binqing Wu, and Dongliang Cui. Ada-MSHyper: Adaptive multi-scale hypergraph transformer for time series forecasting. In *Advances in Neural Information Processing Systems*, 2024a.

Zongjiang Shang, Ling Chen, Binqing Wu, and Liangcui Dong. MSHyper: Multi-scale hypergraph transformer for long-range time series forecasting. *arXiv preprint arXiv:2401.09261*, 2024b.

Zongjiang Shang, Dongliang Cui, Binqing Wu, and Ling Chen. Multi-scale hypergraph meets LLMs: Aligning large language models for time series analysis. In *Proceedings of the International Conference on Learning Representations*, 2026.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-MoE: Billion-scale time series foundation models with mixture of experts. In *Proceedings of the International Conference on Learning Representations*, 2025.

Yan Shu, Hangui Lin, Yexin Liu, Yan Zhang, Gangyan Zeng, Yan Li, Yu Zhou, Ser-Nam Lim, Harry Yang, and Nicu Sebe. When semantics mislead vision: Mitigating large multimodal models hallucinations in scene text spotting and understanding. *arXiv preprint arXiv:2506.05551*, 2025.

Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *Advances in neural information processing systems*, 27, 2014.

Chenxi Sun, Hongyan Li, Yaliang Li, and Shenda Hong. TEST: Text prototype aligned embedding to activate LLM's ability for time series. In *Proceedings of the International Conference on Learning Representations*, 2024.

Gabriel Tjio, Ping Liu, Joey Tianyi Zhou, and Rick Siow Mong Goh. Adversarial semantic hallucination for domain generalized semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 318–327, 2022.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and JUN ZHOU. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *Proceedings of the International Conference on Learning Representations*, 2023.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Shengtong Ju, Zhixuan Chu, and Ming Jin. Timemixer++: A general time series pattern machine for universal predictive analysis. *arXiv preprint arXiv:2410.16032*, 2024a.

Shun Wang, Yong Zhang, Xuanqi Lin, Yongli Hu, Qingming Huang, and Baocai Yin. Dynamic hypergraph structure learning for multivariate time series forecasting. *IEEE Transactions on Big Data*, 10(4):556–567, 2024b.

Yihe Wang, Nan Huang, Taida Li, Yujun Yan, and Xiang Zhang. Medformer: A multi-granularity patching transformer for medical time-series classification. *Advances in Neural Information Processing Systems*, 37:36314–36341, 2024c.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. ETSformer: Exponential smoothing transformers for time-series forecasting. *arXiv preprint arXiv:2202.01381*, 2022.

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 International Conference on Machine Learning*, volume 235, pp. 53140–53164, 2024.

Binqing Wu, Weiqi Chen, Wengwei Wang, Bingqing Peng, Liang Sun, and Ling Chen. WeatherGNN: Exploiting meteo-and spatial-dependencies for local numerical weather prediction bias-correction. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 2433–2441, 2024.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, pp. 22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-variation modeling for general time series analysis. In *Proceedings of the International Conference on Learning Representations*, 2022.

Jiawei Xue, Takahiro Yabe, Kota Tsubouchi, Jianzhu Ma, and Satish Ukkusuri. Multiwave COVID-19 prediction from social awareness using web search and mobility data. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4279–4289, 2022.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*, 2022.

He Zhang and Vishal M Patel. Density-aware single image de-raining using a multi-stream dense network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 695–704, 2018.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 11106–11115, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the International Conference on Machine Learning*, pp. 27268–27286, 2022.

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, 2023.

A Description of Dataset Details

For the long-term forecasting task and few-shot forecasting task, we evaluate on the well-established seven benchmarks, including four ETT datasets (Zhou et al., 2021) (i.e., ETTh1, ETTh2, ETTm1, and ETTm2), ECL[1], Traffic[2], and ILI (Wu et al., 2022). Dataset statistics are summarized in Table 6. In detail, ETT is built upon transformer load signals gathered from two stations at distinct temporal granularities. ILI contains weekly influenza case numbers alongside corresponding illness ratios. ECL assembles hourly power consumption traces from 321 individual users. Traffic dataset records road occupancy from various sensors on San Francisco highways. Following existing methods (Shang et al., 2026; Pan et al., 2024; Zhou et al., 2023; Shang et al., 2024a; Zhou et al., 2021), we partition each dataset into training, validation, and testing subsets on chronological order. Specifically, a 6:2:2 ratio is adopted for the ETT datasets (ETTh1, ETTh2, ETTm1, and ETTm2), whereas a 7:2:1 ratio is applied to the Traffic, ECL, and ILI datasets. In the few-shot forecasting task, we retain only a fraction (5% or 10%) of the training data, while keeping the validation and testing sets intact.

Table 6: Dataset details of long-term forecasting task and few-shot forecasting task.

Dataset	Variates	Timesteps	Granularity	Domain
ETTh1	7	17,420	1 hour	Energy
ETTh2	7	17,420	1 hour	Energy
ETTm1	7	69,680	15 min	Energy
ETTm2	7	69,680	15 min	Energy
ECL	321	26,304	1 hour	Electricity
Traffic	862	17,544	1 hour	Transportation
ILI	7	966	7 days	Epidemiology

Furthermore, for the zero-shot forecasting task, we evaluate on the M4(Makridakis et al., 2018) and M3(Makridakis & Hibon, 2000) datasets, which contain 6 and 4 subsets respectively. Dataset statistics are summarized in Table 7. M4 is a large and diverse dataset that contains time series of various frequencies and fields, including business, financial and economic forecasting. M3 is smaller than M4, but also contains time series from diverse domains and frequencies.

Table 7: Dataset details of zero-shot forecasting task.

Dataset	Length	Horizon	Frequency
M4 Yearly	23000	6	Yearly
M4 Quarterly	24000	8	Quarterly
M4 Monthly	48000	18	Monthly
M4 Weekly	359	13	Weekly
M4 Daily	4227	14	Daily
M4 Hourly	414	48	Hourly
M3 Yearly	645	6	Yearly
M3 Quarterly	756	8	Quarterly
M3 Monthly	1428	18	Monthly
M3 Others	174	8	Monthly

B Description of Baselines

We compare M^{2}TSF with 17 baseline models, and the following provides a detailed introduction of these baseline models:

MSH-LLM (Shang et al., 2026): MSH-LLM utilizes a hyperedging mechanism to enrich the multi-scale semantic representations of time series and incorporates a mixture-of-prompts mechanism to strengthen the capability of LLMs in capturing multi-scale temporal patterns.

[1] https://archive.ics.uci.edu/dataset/321/electricityloaddiagrams20112014

[2] http://pems.dot.ca.gov

S^2IP-LLM (Pan et al., 2024): S^2IP-LLM bridges the gap between time series data and pre-trained language models by aligning time series embeddings with the semantic space of LLMs, and introduces semantic-anchor-based prompts to enhance the model's ability to capture complex temporal patterns.

Time-LLM (Jin et al., 2024): Time-LLM reprograms the input time series into the textual representation space of LLMs through a patch-based reprogramming mechanism and employs a Prompt-as-Prefix mechanism to inject domain-specific contextual knowledge.

AutoTimes (Liu et al., 2024a): AutoTimes leverages frozen LLMs as autoregressive forecasters for time series prediction and introduces time series-specific prompts to inject contextual knowledge.

FPT (Zhou et al., 2023): FPT adapts pre-trained LLMs to time series analysis by fine-tuning only a few critical parameters and establishes a unified framework that generalizes across various time series tasks.

Ada-MSHyper (Shang et al., 2024a): Ada-MSHyper adopts an adaptive hypergraph to model multi-scale high-order interactions and designs a constraint mechanism to resolve the entanglement of temporal variations.

iTransformer (Liu et al., 2023): iTransformer inverts the conventional Transformer design by tokenizing each variate of the time series independently, where attention layers model correlations among variates and feed-forward layers extract nonlinear features.

MSHyper (Shang et al., 2024b): MSHyper introduces a rule-based approach to construct hypergraphs at multiple scales and proposes a tri-stage message passing mechanism, thereby enabling the modeling of group-wise interactions among temporal patterns of different scales.

DLinear (Zeng et al., 2022): DLinear performs seasonal-trend decomposition on the input time series and leverages a separate linear layer for each component to model temporal dependencies.

TimesNet (Wu et al., 2022): TimesNet leverages multi-periodicity analysis to transform 1D time series into 2D tensors, enabling the modeling of complex temporal variations from a two-dimensional perspective.

FEDformer (Zhou et al., 2022): FEDformer integrates a decomposition mechanism to extract the global properties of time series and designs a frequency-enhanced block that captures structural information in the frequency domain.

PatchTST (Nie et al., 2022): PatchTST divides time series into subsequence-level patches and feeds them into a Transformer as input tokens, modeling temporal dependencies in a channel-independent fashion.

NSFormer (Liu et al., 2022): NSFormer incorporates a stationarization module to enhance the predictability of time series and a de-stationary attention module to mitigate the over-stationarization problem.

ETSformer (Woo et al., 2022): ETSformer reformulates the Transformer architecture under the principles of exponential smoothing, where the conventional self-attention mechanism is replaced by exponential smoothing attention and frequency attention.

Autoformer (Wu et al., 2021): Autoformer designs a decomposition-based Transformer architecture that progressively separates trend and seasonal components, and replaces the conventional self-attention with an auto-correlation mechanism.

Informer (Zhou et al., 2021): Informer introduces a sparse self-attention mechanism that selects dominant queries via KL-divergence to reduce computational complexity, and a generative style decoder that produces predictions in a single forward pass to accelerate inference.

Reformer (Kitaev et al., 2019): Reformer employs locality-sensitive hashing (LSH) as an approximation strategy for attention computation and introduces reversible residual layers to alleviate the computational burden.

C EXPERIMENTAL SETTINGS

M^{2}TSF is implemented in Python 3.10 and PyTorch 2.7. All experiments are conducted on a single NVIDIA RTX PRO 6000 GPU(96GB). We use Adam (Kingma, 2014) as the optimizer with the initial learning rate tuned as a hyperparameter. By default, M^{2}TSF uses GPT2-small as the backbone LLM. The total number of scales S is set to 3. For other key hyperparameters, we perform an extensive hyperparameter search to identify optimal configurations. The detailed search space of key hyperparameters is given in Table 8. The source code is available on Anonymous GitHub [3].

Table 8: The search space of key hyperparameters.

Parameters	Choise
batch_size	{ 64, 128, 256, 512}
d_ff	{128, 256, 512}
llm_layers	{2, 4, 6, 8}
layer_index	{0_1, 0_2, 0_3, 0, 1, 2, 3}
learning_rate	{5e-05, 0.0001, 0.0003, 0.0005, 0.001}
in_dropout	{0, 0.1, 0.2}
out_dropout	{0, 0.1, 0.2}
hyper_num	{16, 18, 20}
num_experts	{2, 4, 6, 8}
capacity_factor	{0.5, 1.0, 2.0}

D EVALUATION METRICS

Consistent with existing methods (Shang et al., 2026; Zhou et al., 2023; Liu et al., 2024a; Jin et al., 2024), for long-term time series forecasting and few-shot forecasting, Mean Squared Error (MSE) and Mean Absolute Error (MAE) are employed as our evaluation metrics, which can be formulated as follows:

$$\text{MSE} = \frac{1}{H}\left\|\widehat{\mathbf{Y}}^{\text{O}}_{T+1:T+H} - \mathbf{Y}^{\text{O}}_{T+1:T+H}\right\|_2^2, \quad \text{MAE} = \frac{1}{H}\left|\widehat{\mathbf{Y}}^{\text{O}}_{T+1:T+H} - \mathbf{Y}^{\text{O}}_{T+1:T+H}\right|, \tag{21}$$

where T and H are the input and output lengths, $\widehat{\mathbf{Y}}^{\text{O}}_{T+1:T+H}$ and $\mathbf{Y}^{\text{O}}_{T+1:T+H}$ are the forecasting results and ground truth, respectively. For zero-shot forecasting tasks, the Symmetric Mean Absolute Percentage Error (SMAPE) is employed as evaluation metrics, which is defined as follows:

$$\text{SMAPE} = \frac{200}{H}\sum_{h=1}^{H}\frac{|\widehat{\mathbf{Y}}^{\text{O}}_{T+1:T+H} - \mathbf{Y}^{\text{O}}_{T+1:T+H}|}{|\mathbf{Y}^{\text{O}}_{T+1:T+H}|}, \tag{22}$$

where the specific meanings of T, H, $\widehat{\mathbf{Y}}^{\text{O}}_{T+1:T+H}$, and $\mathbf{Y}^{\text{O}}_{T+1:T+H}$ are as defined above.

E FULL RESULTS

We compare M^{2}TSF with baselines in three different tasks: long-term time series forecasting, few-shot forecasting and zero-shot forecasting. For a fair comparison, we follow the unified experimental settings used in existing works (Shang et al., 2026; Pan et al., 2024; Jin et al., 2024). The average results refer to the arithmetic mean of different forecasting results. The best results are **bolded** and the second best results are underlined.

E.1 LONG-TERM TIME SERIES FORECASTING

Table 9 summarizes the full results of long-term time series forecasting. Consistent with existing method (Shang et al., 2026; Pan et al., 2024; Jin et al., 2024), the input time series length T is set to

[3] https://anonymous.4open.science/r/M2TSF

512 except for the ILI dataset. Performance is evaluated over four prediction horizons: {24, 36, 48, 60} for ILI and {96, 192, 336, 720} for the other datasets. We can observe that M^{2}TSF achieves the SOTA results in 59 out of 70 cases across 7 time series datasets.

Table 9: Full results of long-term time series forecasting in terms of MSE and MAE.

Methods		M^{2}TSF (Ours)		MSH-LLM (ICLR 2026)		S^{2}IP-LLM (ICLR 2024)		Time-LLM (ICLR 2024)		AutoTimes* (NeurIPS 2024)		FPT (NeurIPS 2023)		ASHyper* (NeurIPS 2024)		iTransformer (ICLR 2024)		MSHyper* (arXiv 2024)		DLinear (AAAI 2023)		TimesNet (ICLR 2023)		FEDformer (ICML 2022)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ILI	24	**1.260**	**0.686**	1.306	0.711	1.467	0.778	1.622	0.806	1.552	0.789	2.063	0.881	1.956	0.875	1.694	0.874	2.043	0.910	2.215	1.081	2.317	0.934	3.228	1.260
	36	**1.264**	**0.722**	1.328	0.759	1.534	0.841	1.695	0.857	1.623	0.853	1.868	0.892	1.771	0.886	2.229	0.983	1.879	0.940	1.963	0.963	1.972	0.920	2.679	1.080
	48	**1.445**	**0.757**	1.467	0.769	1.608	0.836	1.654	0.863	1.701	0.848	1.790	0.884	1.720	0.864	2.382	0.995	1.821	0.916	2.130	1.024	2.238	0.940	2.622	1.078
	60	**1.368**	**0.758**	1.392	0.791	1.597	0.849	1.880	0.905	1.690	0.861	1.979	0.957	1.958	0.936	1.988	0.913	2.147	1.026	2.368	1.096	2.027	0.928	2.857	1.157
	Avg	**1.334**	**0.731**	1.373	0.758	1.552	0.826	1.713	0.858	1.642	0.838	1.925	0.903	1.851	0.890	2.073	0.941	1.973	0.948	2.169	1.041	2.139	0.931	2.847	1.144
ECL	96	0.129	**0.222**	**0.127**	0.231	0.135	0.230	0.137	0.237	0.134	0.233	0.139	0.238	0.129	0.234	0.147	0.248	0.147	0.251	0.140	0.237	0.168	0.272	0.193	0.308
	192	**0.146**	0.238	0.150	0.242	0.149	0.247	0.150	0.249	0.150	0.247	0.153	0.251	0.154	**0.227**	0.165	0.267	0.167	0.269	0.153	0.249	0.184	0.289	0.201	0.315
	336	0.163	**0.254**	**0.162**	0.258	0.167	0.266	0.168	0.266	0.165	0.264	0.169	0.266	0.165	0.262	0.178	0.279	0.174	0.275	0.169	0.267	0.198	0.300	0.214	0.329
	720	**0.195**	**0.276**	0.198	0.279	0.200	0.287	0.203	0.293	0.199	0.298	0.206	0.297	0.201	0.290	0.322	0.398	0.216	0.308	0.203	0.301	0.220	0.320	0.246	0.355
	Avg	**0.158**	**0.248**	0.159	0.253	0.163	0.258	0.165	0.261	0.162	0.261	0.167	0.263	0.162	0.253	0.203	0.298	0.176	0.276	0.166	0.264	0.193	0.295	0.214	0.327
Traffic	96	**0.363**	**0.241**	0.365	0.270	0.379	0.274	0.380	0.277	0.366	0.279	0.388	0.282	0.368	0.277	0.367	0.288	0.394	0.389	0.410	0.282	0.593	0.321	0.587	0.366
	192	0.378	**0.252**	**0.372**	0.281	0.397	0.282	0.399	0.288	0.395	0.287	0.407	0.290	0.379	0.288	0.378	0.293	0.375	0.289	0.423	0.287	0.617	0.336	0.604	0.373
	336	0.390	**0.263**	**0.385**	0.279	0.407	0.289	0.408	0.290	0.406	0.283	0.412	0.294	0.397	0.292	0.389	0.294	0.395	0.283	0.436	0.296	0.629	0.336	0.621	0.383
	720	0.425	**0.283**	0.402	0.303	0.440	0.301	0.445	0.308	0.421	0.305	0.450	0.312	0.421	0.298	**0.401**	0.304	0.407	0.308	0.466	0.315	0.640	0.350	0.626	0.382
	Avg	0.389	**0.260**	**0.381**	0.283	0.406	0.287	0.408	0.291	0.397	0.289	0.414	0.295	0.391	0.289	0.384	0.295	0.393	0.317	0.434	0.295	0.620	0.336	0.610	0.376
ETTh1	96	**0.345**	**0.383**	0.360	0.388	0.366	0.396	0.383	0.410	0.368	0.395	0.379	0.402	0.368	0.391	0.395	0.420	0.372	0.417	0.367	0.396	0.468	0.475	0.376	0.419
	192	**0.384**	**0.409**	0.398	0.411	0.401	0.420	0.419	0.435	0.404	0.415	0.415	0.424	0.429	0.417	0.427	0.441	0.418	0.432	0.401	0.419	0.484	0.485	0.420	0.448
	336	**0.391**	**0.423**	0.415	0.432	0.412	0.431	0.426	0.440	0.408	0.435	0.435	0.440	0.419	0.438	0.445	0.457	0.451	0.440	0.434	0.449	0.536	0.516	0.459	0.465
	720	**0.417**	**0.447**	0.436	**0.447**	0.440	0.458	0.428	0.456	0.439	0.503	0.441	0.459	0.446	0.465	0.537	0.530	0.476	0.458	0.472	0.493	0.593	0.537	0.506	0.507
	Avg	**0.384**	**0.416**	0.402	0.420	0.405	0.426	0.414	0.435	0.405	0.437	0.418	0.431	0.416	0.428	0.451	0.462	0.429	0.437	0.419	0.439	0.520	0.503	0.440	0.460
ETTh2	96	**0.253**	**0.321**	0.273	0.331	0.278	0.340	0.297	0.357	0.282	0.329	0.289	0.347	0.274	0.335	0.304	0.360	0.287	0.331	0.301	0.367	0.376	0.415	0.358	0.397
	192	**0.304**	**0.356**	0.335	0.372	0.346	0.385	0.349	0.390	0.352	0.391	0.358	0.392	0.352	0.377	0.377	0.403	0.372	0.389	0.394	0.427	0.409	0.440	0.429	0.439
	336	**0.310**	**0.372**	0.363	0.400	0.367	0.406	0.373	0.408	0.382	0.403	0.383	0.414	0.369	0.427	0.405	0.429	0.407	0.423	0.506	0.495	0.425	0.455	0.496	0.487
	720	**0.382**	**0.421**	0.396	0.428	0.400	0.436	0.400	0.436	0.417	0.425	0.438	0.456	0.407	0.430	0.443	0.464	0.400	0.428	0.805	0.635	0.488	0.494	0.463	0.474
	Avg	**0.312**	**0.368**	0.342	0.383	0.348	0.392	0.355	0.398	0.358	0.387	0.367	0.402	0.351	0.392	0.382	0.414	0.367	0.393	0.502	0.481	0.425	0.451	0.437	0.449
ETTm1	96	**0.278**	**0.336**	0.285	0.340	0.288	0.346	0.291	0.346	0.301	0.347	0.296	0.353	0.297	0.338	0.312	0.366	0.323	0.348	0.304	0.348	0.329	0.377	0.379	0.419
	192	0.319	**0.358**	**0.313**	**0.358**	0.323	0.365	0.336	0.373	0.331	0.371	0.335	0.373	0.333	0.367	0.347	0.385	0.368	0.369	0.336	0.367	0.371	0.401	0.426	0.441
	336	**0.348**	**0.377**	0.355	**0.377**	0.359	0.390	0.362	0.390	0.365	0.380	0.369	0.394	0.365	0.388	0.379	0.404	0.392	0.390	0.368	0.387	0.417	0.428	0.445	0.459
	720	0.405	**0.408**	0.405	0.410	**0.403**	0.418	0.410	0.421	0.423	0.422	0.418	0.424	0.425	0.431	0.441	0.442	0.469	0.433	0.421	0.418	0.483	0.464	0.543	0.490
	Avg	**0.338**	**0.370**	0.340	0.371	0.343	0.380	0.350	0.383	0.355	0.380	0.355	0.386	0.355	0.381	0.370	0.399	0.388	0.385	0.357	0.380	0.400	0.418	0.448	0.452
ETTm2	96	**0.160**	**0.246**	0.161	**0.246**	0.165	0.257	0.184	0.275	0.167	0.261	0.170	0.264	0.168	0.256	0.179	0.271	0.168	0.254	0.168	0.263	0.201	0.286	0.203	0.287
	192	0.216	0.285	0.218	**0.284**	0.222	0.299	0.238	0.310	**0.214**	0.311	0.231	0.306	0.229	0.301	0.242	0.313	0.243	0.311	0.229	0.310	0.260	0.329	0.269	0.328
	336	**0.269**	**0.320**	0.271	**0.320**	0.277	0.330	0.286	0.340	0.284	0.325	0.280	0.339	0.281	0.334	0.288	0.344	0.299	0.338	0.289	0.352	0.331	0.376	0.325	0.366
	720	**0.346**	**0.372**	0.358	0.392	0.363	0.390	0.379	0.403	0.367	0.492	0.373	0.402	0.372	0.397	0.378	0.397	0.397	0.399	0.416	0.437	0.428	0.430	0.421	0.415
	Avg	**0.248**	**0.306**	0.252	0.311	0.257	0.319	0.272	0.332	0.258	0.347	0.264	0.328	0.263	0.322	0.272	0.331	0.277	0.326	0.276	0.341	0.305	0.355	0.305	0.349

E.2 Few-shot forecasting

Table 10 and Table 11 summarize the results of few-shot learning under 10% training data and 5% training data. In the scope of 10% few-shot forecasting, M^{2}TSF achieves an average error reduction of 10.63% and 6.92% compared to LLM4TS methods (i.e., MSH-LLM, S^{2}IP-LLM, and Time-LLM) in MSE and MAE, respectively. For learning under 5% training data, we can observe that MSH-LLM still achieves SOTA results. Specifically, MSH-LLM achieves an average error reduction of 11.72% and 8.10% compared to LLM4TS methods (i.e., MSH-LLM, S^{2}IP-LLM, and Time-LLM) in MSE and MAE, respectively.

Table 10: Full results of few-shot forecasting under 10% training data in terms of MSE and MAE.

Methods		M^{2}TSF (Ours)		MSH-LLM (ICLR 2026)		S^{2}IP-LLM (ICML 2024)		Time-LLM (ICLR 2024)		FPT (NeurIPS 2023)		iTransformer (ICLR 2024)		PatchTST (ICLR 2024)		TimesNet (ICLR 2023)		FEDformer (ICML 2022)		NSFormer (NeurIPS 2022)		ETSformer (arXiv 2022)		Autoformer (NeurIPS 2021)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.424**	**0.432**	0.460	0.450	0.481	0.474	0.720	0.533	0.458	0.456	0.790	0.586	0.516	0.485	0.861	0.628	0.512	0.499	0.918	0.639	1.112	0.806	0.613	0.552
	192	**0.483**	**0.451**	0.516	0.488	0.518	0.491	0.747	0.545	0.570	0.516	0.837	0.609	0.598	0.524	0.797	0.593	0.624	0.555	0.915	0.629	1.155	0.823	0.722	0.598
	336	**0.498**	**0.470**	0.594	0.537	0.664	0.570	0.793	0.551	0.608	0.535	0.780	0.575	0.657	0.550	0.941	0.648	0.691	0.574	0.939	0.644	1.179	0.832	0.750	0.619
	720	**0.668**	**0.547**	0.680	0.581	0.711	0.584	0.880	0.584	0.725	0.591	1.234	0.811	0.762	0.610	0.877	0.641	0.728	0.614	0.887	0.645	1.273	0.874	0.721	0.616
	Avg	**0.518**	**0.475**	0.563	0.514	0.593	0.529	0.785	0.553	0.590	0.525	0.910	0.860	0.633	0.542	0.869	0.628	0.639	0.561	0.915	0.639	1.180	0.834	0.702	0.596
ETTh2	96	**0.294**	**0.352**	0.331	0.366	0.354	0.400	0.334	0.381	0.331	0.374	0.404	0.435	0.353	0.389	0.378	0.409	0.382	0.416	0.389	0.411	0.678	0.619	0.413	0.451
	192	**0.337**	**0.382**	0.374	0.414	0.401	0.423	0.430	0.438	0.402	0.411	0.470	0.474	0.403	0.414	0.490	0.467	0.478	0.474	0.473	0.455	0.785	0.666	0.474	0.477
	336	**0.349**	**0.402**	0.396	0.432	0.442	0.450	0.449	0.458	0.406	0.433	0.489	0.485	0.426	0.441	0.537	0.494	0.504	0.501	0.477	0.472	0.839	0.694	0.547	0.543
	720	**0.421**	**0.441**	0.465	0.478	0.480	0.486	0.485	0.490	0.449	0.464	0.593	0.538	0.477	0.480	0.510	0.491	0.499	0.509	0.507	0.480	1.273	0.874	0.516	0.523
	Avg	**0.350**	**0.394**	0.392	0.423	0.419	0.439	0.424	0.441	0.397	0.421	0.489	0.483	0.415	0.431	0.479	0.465	0.466	0.475	0.462	0.455	0.894	0.713	0.488	0.499
ETTm1	96	0.358	**0.381**	**0.349**	0.383	0.388	0.401	0.412	0.422	0.390	0.404	0.709	0.556	0.410	0.419	0.583	0.501	0.578	0.518	0.761	0.568	0.911	0.688	0.774	0.614
	192	0.393	**0.399**	**0.377**	0.410	0.422	0.421	0.447	0.438	0.429	0.423	0.717	0.548	0.437	0.434	0.630	0.528	0.617	0.546	0.781	0.574	0.955	0.703	0.754	0.592
	336	0.438	**0.423**	**0.405**	0.434	0.456	0.430	0.497	0.465	0.469	0.439	0.735	0.575	0.476	0.454	0.725	0.568	0.998	0.775	0.803	0.587	0.991	0.719	0.869	0.677
	720	0.495	**0.459**	**0.482**	**0.468**	0.554	0.490	0.594	0.521	0.569	0.498	0.752	0.584	0.681	0.556	0.769	0.549	0.693	0.579	0.844	0.581	1.062	0.747	0.810	0.630
	Avg	0.421	**0.416**	**0.403**	0.424	0.455	0.435	0.487	0.461	0.464	0.441	0.728	0.565	0.501	0.466	0.677	0.537	0.722	0.605	0.797	0.578	0.980	0.714	0.802	0.628
ETTm2	96	0.188	0.266	**0.178**	**0.261**	0.192	0.274	0.224	0.296	0.188	0.269	0.245	0.322	0.191	0.274	0.212	0.285	0.291	0.399	0.229	0.308	0.331	0.430	0.352	0.454
	192	0.246	0.307	**0.238**	**0.304**	0.246	0.313	0.260	0.317	0.251	0.309	0.274	0.338	0.252	0.317	0.270	0.323	0.307	0.379	0.291	0.343	0.400	0.464	0.694	0.691
	336	**0.294**	**0.339**	0.299	0.341	0.301	0.340	0.312	0.349	0.307	0.346	0.361	0.394	0.306	0.353	0.323	0.353	0.543	0.559	0.348	0.376	0.469	0.498	2.408	1.407
	720	**0.384**	**0.393**	0.403	0.401	0.400	0.403	0.424	0.416	0.426	0.417	0.467	0.442	0.433	0.427	0.474	0.449	0.712	0.614	0.461	0.438	0.589	0.557	1.913	1.166
	Avg	**0.278**	**0.326**	0.280	0.327	0.284	0.332	0.305	0.344	0.293	0.335	0.336	0.373	0.296	0.343	0.320	0.353	0.463	0.488	0.332	0.366	0.447	0.487	1.342	0.930

Table 11: Full results of few-shot learning under 5% training data in terms of MSE and MAE. '- -' indicates 5% training data is insufficient to constitute a training set.

Methods		M^{2}TSF (Ours)		MSH-LLM (ICLR 2026)		S^{2}IP-LLM (ICML 2024)		Time-LLM (ICLR 2024)		FPT (NeurIPS 2023)		iTransformer (ICLR 2024)		PatchTST (ICLR 2024)		TimesNet (ICLR 2023)		FEDformer (ICML 2022)		NSFormer (NeurIPS 2022)		ETSformer (arXiv 2022)		Autoformer (NeurIPS 2021)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.516	**0.466**	**0.489**	0.475	0.500	0.493	0.732	0.556	0.543	0.506	0.808	0.610	0.557	0.519	0.892	0.625	0.593	0.529	0.952	0.650	1.169	0.832	0.681	0.570
	192	**0.635**	**0.512**	0.658	0.535	0.690	0.539	0.872	0.604	0.748	0.580	0.928	0.658	0.711	0.570	0.940	0.665	0.652	0.563	0.943	0.645	1.221	0.853	0.725	0.602
	336	**0.674**	**0.546**	0.738	0.600	0.761	0.620	1.071	0.721	0.754	0.595	1.475	0.861	0.816	0.619	0.945	0.653	0.731	0.594	0.935	0.644	1.179	0.832	0.761	0.624
	720	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Avg	**0.608**	**0.508**	0.628	0.537	0.650	0.550	0.891	0.627	0.681	0.560	1.070	0.710	0.694	0.569	0.925	0.647	0.658	0.562	0.943	0.646	1.189	0.839	0.722	0.598
ETTh2	96	**0.309**	**0.371**	0.342	0.389	0.363	0.409	0.399	0.420	0.376	0.421	0.397	0.427	0.401	0.421	0.409	0.420	0.390	0.424	0.408	0.423	0.678	0.619	0.428	0.468
	192	**0.357**	**0.393**	0.375	0.412	0.375	0.411	0.487	0.479	0.418	0.441	0.438	0.445	0.452	0.455	0.483	0.464	0.457	0.465	0.497	0.468	0.845	0.697	0.496	0.504
	336	**0.389**	0.421	0.401	**0.419**	0.403	0.421	0.858	0.660	0.408	0.439	0.631	0.553	0.464	0.469	0.499	0.479	0.477	0.483	0.507	0.481	0.905	0.727	0.486	0.496
	720	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Avg	**0.352**	**0.395**	0.373	0.407	0.380	0.413	0.581	0.519	0.400	0.433	0.488	0.475	0.827	0.615	0.439	0.448	0.463	0.454	0.470	0.489	0.809	0.681	0.441	0.457
ETTm1	96	0.341	0.374	**0.328**	**0.365**	0.357	0.390	0.422	0.424	0.386	0.405	0.589	0.510	0.399	0.414	0.606	0.518	0.628	0.544	0.823	0.587	1.031	0.747	0.726	0.578
	192	0.372	**0.391**	**0.353**	0.395	0.432	0.434	0.448	0.440	0.440	0.438	0.703	0.565	0.441	0.436	0.681	0.539	0.666	0.566	0.844	0.591	1.087	0.766	0.750	0.591
	336	0.414	0.415	**0.394**	**0.412**	0.440	0.442	0.519	0.482	0.485	0.459	0.898	0.641	0.499	0.467	0.786	0.597	0.807	0.628	0.870	0.603	1.138	0.787	0.851	0.659
	720	0.523	**0.480**	**0.518**	0.483	0.593	0.521	0.708	0.573	0.577	0.499	0.948	0.671	0.767	0.587	0.796	0.593	0.822	0.633	0.893	0.611	1.245	0.831	0.857	0.655
	Avg	0.413	0.415	**0.398**	**0.414**	0.455	0.446	0.524	0.479	0.472	0.450	0.784	0.596	0.526	0.476	0.717	0.561	0.730	0.592	0.857	0.598	1.125	0.782	0.796	0.620
ETTm2	96	0.192	0.269	**0.179**	**0.264**	0.197	0.278	0.225	0.300	0.199	0.280	0.265	0.339	0.206	0.288	0.220	0.299	0.229	0.320	0.238	0.316	0.404	0.485	0.232	0.322
	192	0.250	**0.308**	**0.242**	0.309	0.254	0.322	0.275	0.334	0.256	0.316	0.310	0.362	0.264	0.324	0.311	0.361	0.394	0.361	0.298	0.349	0.479	0.521	0.291	0.357
	336	**0.300**	**0.342**	**0.300**	0.344	0.315	0.350	0.339	0.371	0.318	0.353	0.373	0.399	0.334	0.367	0.338	0.366	0.378	0.427	0.353	0.380	0.552	0.555	0.478	0.517
	720	**0.391**	**0.397**	0.411	0.414	0.421	0.421	0.464	0.441	0.460	0.436	0.478	0.454	0.454	0.432	0.509	0.465	0.523	0.510	0.475	0.445	0.701	0.627	0.553	0.538
	Avg	**0.283**	**0.329**	**0.283**	0.333	0.296	0.342	0.325	0.361	0.308	0.346	0.356	0.388	0.314	0.352	0.344	0.372	0.381	0.404	0.341	0.372	0.534	0.547	0.388	0.433

E.3 Zero-shot forecasting

Following existing works (Shang et al., 2026; Zhou et al., 2023), M3 and M4 datasets are used to evaluate the zero-shot capabilities of models. M4 $\rightarrow$ M3 means training on M4 dataset and testing on M3 dataset. In detail, models trained on the corresponding subsets of M4 are used to test on M3 Yearly, M3 Quarterly, and M3 Monthly. For M3 Others, the model trained on M4 Quarterly is used to test. For M3 $\rightarrow$ M4, similarly, for M4 Yearly, M4 Quarterly, and M4 Monthly, we directly employ models trained on corresponding subsets of M3 to test. For the other subsets, M4 Weekly, M4 Daily, and M4 Hourly, we perform inference using the model trained on M3 Monthly.

Table 12 reports the full zero-shot forecasting results. M^{2}TSF achieves state-of-the-art performance in 7 out of 10 cases, which we attribute to its effective leveraging of the transfer learning capabilities of pretrained LLMs.

Table 12: Full results of zero-shot forecasting in terms of SMAPE.

	Method	M^{2}TSF (Ours)	MSH-LLM (ICLR 2026)	AutoTimes (NeurIPS 2024)	FPT (NeurIPS 2023)	DLinear (AAAI 2023)	PatchTST (ICLR 2023)	TimesNet (ICLR 2023)	NSformer (NeurIPS 2022)	FEDformer (ICML 2022)	Informer (AAAI 2021)	Reformer (ICLR 2019)
M4 $\rightarrow$ M3	Yearly	**15.572**	15.650	15.710	16.420	17.430	15.990	18.750	17.050	16.000	19.700	16.030
	Quarterly	**9.159**	9.240	9.350	10.130	9.740	9.620	12.260	12.560	9.480	13.000	9.760
	Monthly	13.668	**13.570**	14.060	14.100	15.650	14.710	14.010	16.820	15.120	15.910	14.800
	Others	5.250	5.663	5.790	**4.810**	6.810	9.440	6.880	8.130	8.940	13.030	7.530
	Average	**12.454**	12.469	12.750	13.060	14.030	13.390	14.170	15.290	13.530	15.820	13.370
M3 $\rightarrow$ M4	Yearly	**13.455**	13.645	13.728	13.740	14.193	13.966	15.655	14.988	13.887	18.542	15.652
	Quarterly	**10.675**	10.703	10.742	10.787	18.856	10.929	11.877	11.686	11.513	16.907	11.051
	Monthly	**14.464**	14.489	14.558	14.630	14.765	14.664	16.165	16.098	18.154	23.454	15.604
	Others	6.241	**6.132**	6.259	7.081	9.194	7.087	6.863	6.977	7.529	7.348	7.001
	Average	**12.631**	12.968	13.036	13.125	15.337	13.228	14.553	14.327	15.047	19.047	14.092

F Ablation Studies

Layer-wise scale alignment strategies. To investigate the effectiveness of our layer-wise scale alignment strategies, we conduct ablation studies on ETTh1 dataset by carefully designing the following two variants: (1) C2F: It performs coarse-to-fine alignment, which emphasizes coarse-grained semantics at shallow layers and emphasizes fine-grained semantics at coarse scales. (2) UNI: It gives equal weight to coarse-grained semantics and fine-grained semantics at different scales.

Table 13: Results of different layer-wise scale alignment strategies on the ETTh1 dataset.

Methods	C2F		UNI		M^{2}TSF	
Metric	MSE	MAE	MSE	MAE	MSE	MAE
96	0.355	0.390	0.353	0.388	**0.345**	**0.383**
192	0.390	0.416	0.387	0.414	**0.384**	**0.409**
336	0.404	0.430	0.399	0.427	**0.391**	**0.423**
720	0.429	0.457	0.420	0.450	**0.417**	**0.447**

The experimental results are shown in Table 13. We can observe that M^{2}TSF performs better than C2F and UNI, showing the effectiveness of our layer-wise scale alignment strategies. In addition, performance progressively degrades from M^{2}TSF to UNI and further to C2F. The reason may be that lower layers capture fine-grained semantics while deeper layers capture higher-level semantics. Both imposing a fixed alignment direction (C2F) and applying uniform weighting (UNI) contradict this intrinsic layer-wise distribution and disrupt the information flow across scales, leading to performance degradation.

G Model Analysis

G.1 Robustness Analysis

All experimental results are averaged over three runs. To evaluate the robustness of M^{2}TSF and rule out the influence of random variation, we report the standard deviation and statistical significance level (T-test) between M^{2}TSF and the second-best baseline (i.e., MSH-LLM) for long-term, few-shot, and zero-shot forecasting. The experimental results are shown in Tables 14, 15, and 16. † indicates that in some cases, MSH-LLM achieves a better MSE, and the reported percent denotes the statistical significance of that difference.

Table 14: Standard deviations and statistical significance test results for M^{2}TSF and MSH-LLM in long-term time series forecasting.

Dataset	M^{2}TSF	MSH-LLM	Significance Level
Metric	MSE MAE	MSE MAE	Percent
ILI	1.334±0.0019 0.731±0.0016	1.373±0.0022 0.758±0.0009	99%
Electricity	0.158±0.0003 0.248±0.0009	0.159±0.0003 0.253±0.0005	95%
Traffic	0.389±0.0004 0.260±0.0009	0.381±0.0003 0.283±0.0005	99%†
ETT	0.321±0.0013 0.365±0.0009	0.334±0.0013 0.371±0.0011	99%

Table 15: Standard deviations and statistical significance test results for M^{2}TSF and MSH-LLM in few-shot time series forecasting.

Dataset	M^{2}TSF	MSH-LLM	Significance Level
Metric	MSE MAE	MSE MAE	Percent
ETTh1	0.518±0.0009 0.475±0.0007	0.563±0.0025 0.514±0.0023	99%
ETTh2	0.350±0.0008 0.394±0.0003	0.392±0.0031 0.423±0.0019	99%
ETTm1	0.421±0.0006 0.416±0.0005	0.403±0.0017 0.424±0.0015	99%†
ETTm2	0.278±0.0004 0.326±0.0003	0.280±0.0003 0.327±0.0004	95%

Table 16: Standard deviations and statistical significance test results for M^{2}TSF and MSH-LLM in zero-shot time series forecasting.

Dataset	M^{2}TSF	MSH-LLM	Significance Level
Metric	SMAPE	SMAPE	Percent
M4→M3	12.454±0.0025	12.469±0.0035	99%
M3→M4	12.631±0.0021	12.968±0.0035	99%

From Tables 14, 15, and 16, we can observe that M^{2}TSF significantly outperforms the second-best baseline at the 95% significance level on all datasets, and at the 99% level on the majority of them, verifying that the reported gains are not due to random variation but reflect genuine improvements. The experimental results indicating that the performance improvements achieved by M^{2}TSF are substantial and consistent across all datasets under different settings.

G.2 Efficiency Analysis

We compare the computation cost of M^{2}TSF against representative baselines (i.e., MSH-LLM, S^{2}IP-LLM, Time-LLM, and FPT) on the ETTh1 dataset with the output length of 720 and batch size of 32. For a fair comparison, we use GPT-2 as the base LLM and rerun baselines under unified settings. As shown in Table 17, M^{2}TSF achieves a favorable trade-off between performance and computation cost.

Table 17: Computation cost.

Methods	Training Time	# Parameters	GPU Occupation	MSE results
M^{2}TSF	0.099s	**11,133,337**	8,458MB	**0.430**
MSH-LLM	0.108s	75,852,238	7,872MB	0.451
S^{2}IP-LLM	0.440s	63,636,512	9,991MB	0.459
Time-LLM	0.114s	53,441,968	5,403MB	0.460
FPT	**0.014s**	36,209,616	**2,632MB**	0.463

G.3 Compared with Time Series Foundation Models

In this work, we use LLMs as the primary backbones of M^{2}TSF. Time series foundation models (TSFMs) have also demonstrated strong forecasting performance, particularly in zero-shot and few-shot settings. To assess the effectiveness of M^{2}TSF relative to TSFMs, we have included TimesFM (Das et al., 2024), Chronos (Ansari et al., 2024), and Moirai (Woo et al., 2024) as additional baselines for both few-shot and zero-shot forecasting. The experimental results are reported in Tables 18 and 19.

Table 18: Few-shot time series forecasting results for M^{2}TSF, MSH-LLM, and time series foundation models. The best results are highlighted in bold.

Dataset	M^{2}TSF		MSH-LLM		TimesFM		Chronos		Moirai	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.518**	**0.475**	0.563	0.514	0.597	0.534	0.589	0.529	0.616	0.545
ETTh2	**0.350**	**0.394**	0.392	0.423	0.423	0.449	0.410	0.438	0.437	0.456
ETTm1	0.421	**0.416**	**0.403**	0.424	0.458	0.439	0.446	0.427	0.465	0.443
ETTm2	**0.278**	**0.326**	0.280	0.327	0.294	0.342	0.287	0.339	0.301	0.350

Table 19: Zero-shot forecasting results for M^{2}TSF, MSH-LLM, and time series foundation models on M3 and M4 datasets. The best results are highlighted in bold.

Dataset	M^{2}TSF	MSH-LLM	TimesFM	Chronos	Moirai
Metric	SMAPE	SMAPE	SMAPE	SMAPE	SMAPE
M4→M3	**12.454**	12.469	13.005	12.974	13.378
M3→M4	**12.631**	12.968	13.125	13.002	13.541

From Tables 18 and 19, we can observe that: (1) LLM-based time series forecasting methods (i.e., M^{2}TSF and MSH-LLM) perform better than TSFMs (i.e., TimesFM, Chronos, and Moirai). Specifically, compared with the best TSFM (i.e., Chronos), M^{2}TSF reduces forecasting errors by an average of 8.86% and 3.43% in terms of MSE and SMAPE. This is because LLMs pretrained on large-scale text corpora possess rich semantic priors and cross-domain knowledge, which endow them with strong feature extraction and generalization abilities. (2) M^{2}TSF performs better than other LLM-based methods in most cases. This is because, by leveraging the CII and PAF modules, M^{2}TSF can achieve progressive cross-scale interaction and adaptive stream fusion that are beneficial for forecasting.

H Limitations and Future Works

Despite the promising results, M^{2}TSF has certain limitations. First, the current framework relies on manually pre-defined temporal scales, which might not be optimal for datasets with complex,

non-stationary patterns. Second, the computational scalability remains a challenge. As each scale is processed via an independent stream, the GPU memory usage grows in line with the number of temporal scales, potentially limiting its applicability in some scenarios.

In the future, we will extend our work in the following directions. First, we aim to investigate adaptive multi-scale decomposition mechanisms, allowing the model to dynamically adjust temporal granularities based on input data. Second, we will focus on optimizing the computational efficiency of the multi-stream architecture. Methods like sparse pathway activation and token compression are promising.

I BROADER IMPACTS

M^2TSF advances time series forecasting by enabling more accurate predictions across diverse real-world domains, including weather forecasting, traffic prediction, and medical diagnostics. The multi-stream multi-scale framework can help improve decision-making in these critical applications, potentially benefiting public safety and healthcare. This paper mainly focuses on scientific research and has no obvious negative social impact.