STEM-LTS: Integrating Semantic-Temporal Dynamics in LLM-driven Time Series Analysis

Zhe Zhao[1 3], **Pengkun Wang**[1 2 *], **Haibin Wen**[4], **Shuang Wang**[1], **Liheng Yu** [1], **Yang Wang** [1 2 5 *]

[1]University of Science and Technology of China, Hefei 230026, China
[2]Suzhou Institute for Advanced Research, University of Science and Technology of China, Suzhou 215123, China
[3]City University of Hong Kong
[4]The Hong Kong University of Science and Technology (Guangzhou)
[5]Key Laboratory of Precision and Intelligent Chemistry, USTC
{zz4543, ws20021002,xuco}@mail.ustc.edu.cn, {pengkun, zzy0929, angyan}@ustc.edu.cn, haibin65535@gmail.com

Abstract

Time series forecasting plays a crucial role in domains such as finance, healthcare, and climate science. However, as modern time series data become increasingly complex, featuring high dimensionality, intricate spatiotemporal dependencies, and multi-scale evolutionary patterns, traditional analytical methods and existing predictive models face significant challenges. Although Large Language Models (LLMs) excel in capturing long-range dependencies, they still struggle with multi-scale dynamics and seasonal patterns. Moreover, while LLMs' semantic representation capabilities are rich, they often lack explicit alignment with the numerical patterns and temporal structures of time series data, leading to limitations in predictive accuracy and interpretability. To address these challenges, this paper proposes a novel framework, STEM-LTS (Semantic-TEmporal Modeling for Large-scale Time Series). STEM-LTS enhances the ability to capture complex spatiotemporal dependencies by integrating time series decomposition techniques with LLM-based modeling. The semantic-temporal alignment mechanism within the framework significantly improves LLMs' ability to interpret and forecast time series data. Additionally, we develop an adaptive multi-task learning strategy to optimize the model's performance across multiple dimensions. Through extensive experiments on various real-world datasets, we demonstrate that STEM-LTS achieves significant improvements in prediction accuracy, robustness to noise, and interpretability. Our work not only advances LLM-based time series analysis but also offers new perspectives on handling complex temporal data. The code is available at https://github.com/DataLab-atom/STEM-LTS.

Introduction

Time series forecasting remains a cornerstone of data analysis, with critical applications spanning finance, healthcare, and climate science (Box et al. 2015). As we grapple with increasingly complex systems, modern time series data exhibit high dimensionality, intricate spatiotemporal dependencies, and multi-scale evolutionary patterns (Miao et al. 2022; Xu et al. 2024). These characteristics not only challenge traditional analytical methods but also push the boundaries of contemporary predictive models, including the emerging paradigm of Large Language Models (LLMs) in time series analysis (Wen et al. 2022).

The evolution of time series analysis has been marked by a diverse array of methodologies. Classical approaches like ARIMA (Box et al. 2015) and state space models (Durbin and Koopman 2012) have established a robust foundation for capturing linear relationships and cyclical patterns. Concurrently, deep learning architectures such as Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber 1997) and Temporal Convolutional Networks (TCN) (Bai, Kolter, and Koltun 2018) have demonstrated significant potential in modeling complex temporal dependencies. Recent years have witnessed the application of LLMs to time series forecasting, leveraging their powerful semantic understanding and long-range dependency modeling capabilities (Zerveas et al. 2021).

However, despite their impressive performance, LLM-based methods face several challenges when applied to high-dimensional time series data with complex evolutionary patterns (Lim and Zohren 2021). First, while adept at processing sequential data, LLMs are not inherently designed to capture the unique multi-scale dynamics and seasonal patterns inherent in many time series (Oreshkin et al. 2019). Second, the semantic representations learned by LLMs, though rich in contextual understanding, often lack explicit alignment with the numerical patterns and temporal structures specific to time series data (Wu et al. 2021). Lastly, the computational complexity of LLMs can be prohibitive when dealing with high-dimensional time series, necessitating more efficient and focused approaches (Zhou et al. 2021).

To address these multifaceted challenges, we propose STEM-LTS (Semantic-TEmporal Modeling for Large-scale Time Series), an innovative framework that represents a significant advancement in LLM-based time series analysis. STEM-LTS offers a sophisticated solution for high-dimensional, complex time series forecasting by synergistically integrating multi-scale temporal decomposition (Cleveland et al. 1990), semantic-aware sequence modeling (Vaswani et al. 2017), and adaptive multi-task learning (Ruder 2017) within the LLM paradigm. Our research

[*]Corresponding author.

Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

makes several significant contributions:

1. We introduce a unified framework that seamlessly integrates time series decomposition with LLM-based modeling, unveiling intrinsic data structures while harnessing LLMs' semantic power to model complex temporal dependencies and contextual information.

2. We devise a novel semantic-temporal alignment mechanism that significantly enhances the LLM's ability to interpret and forecast time series data, bridging the gap between numerical patterns and semantic representations.

3. We develop an adaptive multi-task learning strategy tailored for LLM-based time series analysis, optimizing model performance across multiple dimensions and enhancing both predictive accuracy and interpretability.

4. We provide rigorous theoretical analyses that elucidate the synergies between temporal decomposition techniques and LLM architectures in preserving and leveraging essential time series characteristics.

Through extensive experiments on diverse datasets, we demonstrate STEM-LTS's significant improvements in prediction accuracy, robustness to noise, and interpretability compared to state-of-the-art methods. Our work not only advances LLM-based time series analysis but also offers new perspectives on handling complex temporal data, potentially transforming approaches to critical forecasting tasks across various domains.

Related Work

Time Series Learning

Time series analysis has evolved from classical statistical approaches like ARIMA (Box et al. 2015) to advanced deep learning architectures. Deep models such as LSTM (Hochreiter and Schmidhuber 1997) and TCN (Bai, Kolter, and Koltun 2018) revolutionized temporal pattern modeling, while the Temporal Fusion Transformer (Lim et al. 2021) introduced attention mechanisms for multi-horizon forecasting.

Decomposition-based approaches gained prominence through TBATS (De Livera, Hyndman, and Snyder 2011) and N-BEATS (Oreshkin et al. 2019), which explicitly model trend and seasonality. Recent works like Autoformer (Wu et al. 2021) and FEDformer (Zhou et al. 2022) further enhanced periodic pattern learning by combining efficient attention mechanisms with Fourier transformations. Our work extends these foundations to address high-dimensional, multi-scale time series challenges.

Large Language Models in Time Series Domain

The success of LLMs, exemplified by GPT-3 (Brown et al. 2020), has inspired innovations in time series analysis. Time2Vec (Kazemi et al. 2019) introduced novel temporal encoding, while Informer (Zhou et al. 2021) developed sparse attention for long sequences. TimeNet (Malhotra et al. 2017) pioneered transfer learning in this domain, and TimesNet (Wu et al. 2023) established a unified framework for periodic learning.

STEM-LTS advances these approaches by integrating LLMs' representational power with domain-specific time series knowledge. Drawing inspiration from CLIP (Radford et al. 2021), we incorporate contrastive learning to bridge numerical patterns and semantic interpretations in time series analysis.

Multi-objective Learning

Multi-objective optimization in machine learning has gained increasing attention due to its practical significance in balancing multiple competing objectives. Traditional approaches often rely on scalarization techniques to transform multiple objectives into a single objective. Recent work by Lin et al. (Lin et al. 2024) introduced smooth Tchebycheff scalarization, providing a more effective way to handle multiple objectives while maintaining solution diversity.

The challenge of handling multiple objectives under uncertainty has been addressed through various bandit algorithms. Xue et al. (Xue et al. 2024) developed novel approaches for multi-objective Lipschitz bandits under lexicographic ordering, while their work on heavy-tailed rewards (Xue et al. 2023) provided theoretical guarantees for robust optimization. Our work extends these ideas to time series domain, proposing a framework that effectively balances multiple learning objectives while maintaining temporal coherence.

Problem Formulation and Preliminary

Given a multivariate time series $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$, where $\mathbf{x}_t \in R^N$, the goal is to learn a function $f : R^{N\times t} \to R^{N\times H}$ that maps historical observations to future values:

$$\hat{\mathbf{x}}_{t+1:t+H} = f(\mathbf{x}_{1:t}; \Theta), \tag{1}$$

where $\hat{\mathbf{x}}_{t+1:t+H} = [\hat{\mathbf{x}}_{t+1}, \hat{\mathbf{x}}_{t+2}, \ldots, \hat{\mathbf{x}}_{t+H}]$ denotes the predicted values over the forecasting horizon H, and Θ represents the learnable parameters. The time series can be decomposed into trend, seasonal, and residual components:

$$\mathbf{x}_t = \mathbf{x}_t^{\text{trend}} + \mathbf{x}_t^{\text{season}} + \mathbf{x}_t^{\text{residual}}. \tag{2}$$

We propose STEM-LTS (Semantic-Temporal Enhanced Modeling for Large-scale Time Series), a framework that integrates semantic-temporal dynamics in a Transformer-based time series analysis. STEM-LTS employs a decomposition function $\mathcal{D} : R^N \to R^{3N}$ to separate the original series into its constituent components: $[\mathbf{x}_t^{\text{trend}}, \mathbf{x}_t^{\text{season}}, \mathbf{x}_t^{\text{residual}}] = \mathcal{D}(\mathbf{x}_t)$. The decomposed components are then mapped into a latent space using a set of encoders $\{\mathcal{E}^{\text{trend}}, \mathcal{E}^{\text{season}}, \mathcal{E}^{\text{residual}}\}$:

$$\mathbf{z}_t^c = \mathcal{E}^c(\mathbf{x}_t^c; \Theta^c), \quad c \in \{\text{trend}, \text{season}, \text{residual}\}, \tag{3}$$

where $\mathbf{z}_t^c \in R^d$ represents the latent representation of component c at time step t, and Θ^c denotes the learnable parameters of the corresponding encoder.

To leverage the power of Transformers, we process the latent representations using a Transformer-based sequence model $\mathcal{T} : R^{d\times t} \to R^d$ to capture long-term dependencies:

$$\mathbf{h}_t = \mathcal{T}([\mathbf{z}_{1:t}]; \Theta^{\text{trans}}), \tag{4}$$

where $[\cdot]$ denotes the concatenation operation, $\mathbf{h}_t \in R^d$ is the output hidden state at time step t, and Θ^{trans} represents the parameters of the Transformer model. Finally, the

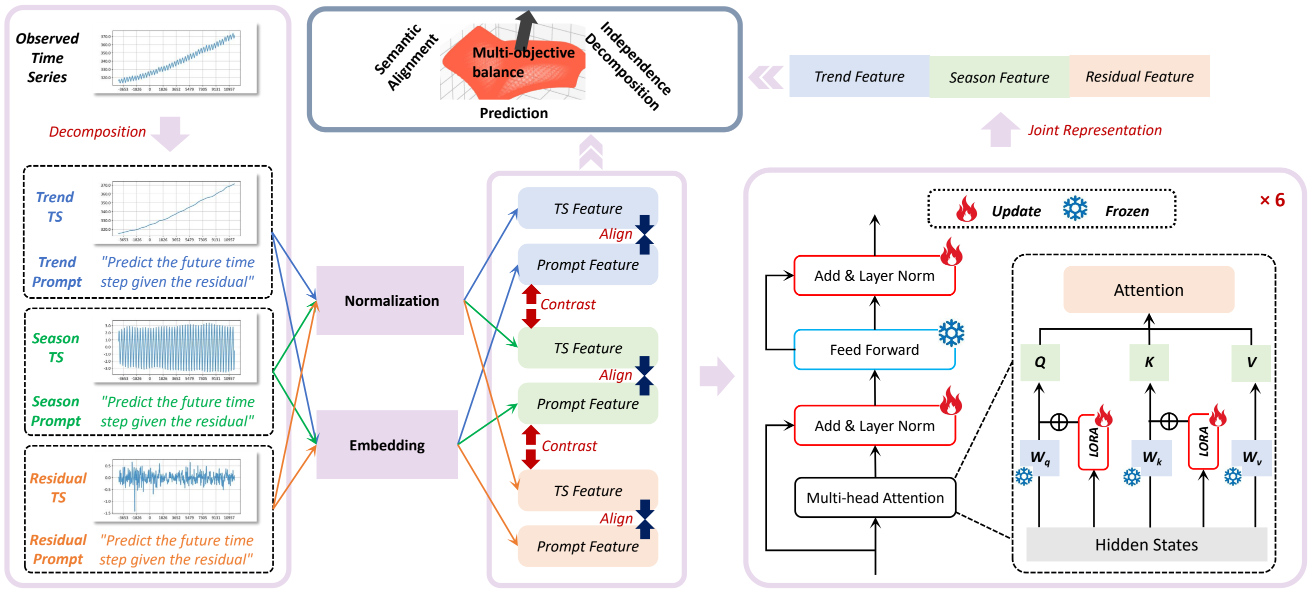

Figure 1: Overview of the STEM-LTS framework. The framework first decomposes the observed time series into three sub-components: trend, seasonality, and residual. Each sub-component is mapped to its corresponding embedding space through independent encoder networks and aligned with learnable prompt embeddings for semantic alignment. The sub-component embeddings are aggregated via an attention mechanism to form a joint representation for subsequent prediction tasks. The entire model is optimized end-to-end by balancing multiple learning objectives (prediction loss, semantic alignment loss, and time series decomposition independence loss) to adaptively capture the complex spatiotemporal dynamics and semantic information in time series data.

output hidden states are passed through a prediction head $\mathcal{F} : R^d \rightarrow R^{N\times H}$ to generate the forecasted values:

$$\hat{\mathbf{x}}_{t+1:t+H} = \mathcal{F}(\mathbf{h}_t; \Theta^{\text{pred}}). \tag{5}$$

The training objective of STEM-LTS is to minimize the forecasting loss (e.g., mean squared error) between the predicted values and the ground truth:

$$\mathcal{L}_{\text{predict}} = \frac{1}{H}\sum_{i=1}^{H} \|\hat{\mathbf{x}}_{t+i} - \mathbf{x}_{t+i}\|_2^2. \tag{6}$$

By incorporating time series decomposition, Transformer-based sequence modeling, and semantic-temporal alignment, STEM-LTS aims to capture the multi-scale dynamics and intricate dependencies present in real-world time series data, enabling accurate and robust forecasting.

Methodology

In this section, we present our proposed STEM-LTS framework for adaptive multivariate time series forecasting. STEM-LTS integrates time series decomposition, semantic-temporal alignment, and adaptive multi-task learning to capture the intricate multi-scale dynamics and establish meaningful connections between numerical patterns and semantic concepts. The overall architecture of STEM-LTS is illustrated in Figure 1.

Multi-scale Time Series Decomposition and Regularization

Given a multivariate time series $\mathbf{X} \in R^{T\times D}$ with T time steps and D feature dimensions, we first decompose it into three sub-components: trend $\mathbf{T}$, seasonality $\mathbf{S}$, and residual $\mathbf{R}$. This decomposition process can be formally expressed as:

$$\mathbf{X} = \mathbf{T} + \mathbf{S} + \mathbf{R} \tag{7}$$

where $\mathbf{T}, \mathbf{S}, \mathbf{R} \in R^{T\times D}$ represent the trend, seasonality, and residual components, respectively.

To facilitate the model in capturing independent and semantically meaningful multi-scale temporal representations, we introduce a novel regularization approach based on the covariance matrix to impose constraints on the correlations among the decomposed components. Initially, we perform concatenation of the three components along the feature dimension, followed by a permutation operation along the temporal dimension to derive a new tensor $\mathbf{Z} \in R^{D\times 3\times T}$:

$$\mathbf{Z} = \pi(\text{concat}(\mathbf{T}, \mathbf{S}, \mathbf{R}), (0, 2, 1)) \tag{8}$$

where $\pi(\cdot)$ denotes the permutation operator and concat$(\cdot)$ represents the concatenation operation along the specified dimension.

Subsequently, for each feature slice $\mathbf{Z}_d \in R^{3\times T}$ of $\mathbf{Z}$, we compute the matrix exponential of the strict lower triangular part of its covariance matrix Σ_d and take the average to obtain the time series decomposition loss $\mathcal{L}_{\text{STL}}$:

$$\mathcal{L}_{\text{STL}} = \frac{1}{D}\sum_{d=1}^{D}\frac{1}{M}\sum_{i>j}[\exp(\text{stril}(\Sigma_d))]_{ij} \tag{9}$$

where

$$\Sigma_d = \frac{1}{T-1}(\mathbf{Z}_d - \bar{\mathbf{Z}}_d)^\top(\mathbf{Z}_d - \bar{\mathbf{Z}}_d) \tag{10}$$

represents the covariance matrix of the d-th feature slice, with $\bar{\mathbf{Z}}_d$ being the mean vector of $\mathbf{Z}_d$. stril$(\cdot)$ denotes the strict lower triangular part of a matrix, $M = \frac{3(3-1)}{2} = 3$ is the number of elements in the strict lower triangular part, and $[\cdot]_{ij}$ represents the (i,j)-th element of a matrix.

By minimizing the time series decomposition loss $\mathcal{L}_{\text{STL}}$, the model is encouraged to learn statistical independence among the trend, seasonality, and residual components. Intuitively, an ideal time series decomposition should minimize the covariances between different components. The proposed covariance matrix-based regularization approach effectively promotes the model to capture more interpretable and independent multi-scale temporal representations, thereby enhancing the performance on downstream forecasting tasks.

Prompt-based Semantic Alignment

We introduce a prompt-based semantic alignment mechanism to align decomposed time series components with semantic concepts. We design three independent encoder networks $f_{enc}^{trend}(\cdot)$, $f_{enc}^{season}(\cdot)$, and $f_{enc}^{residual}(\cdot)$ to map trend, seasonality, and residual components to their corresponding embedding spaces:

$$\begin{bmatrix}\mathbf{H}_{trend}\\ \mathbf{H}_{season}\\ \mathbf{H}_{residual}\end{bmatrix} = \begin{bmatrix} f_{enc}^{trend}(\mathbf{T};\Theta_{enc}^{trend})\\ f_{enc}^{season}(\mathbf{S};\Theta_{enc}^{season})\\ f_{enc}^{residual}(\mathbf{R};\Theta_{enc}^{residual})\end{bmatrix} \tag{11}$$

where $\mathbf{H}_{trend}, \mathbf{H}_{season}, \mathbf{H}_{residual} \in R^{T\times d}$ represent trend, seasonality, and residual embeddings, respectively, and d is the embedding dimension. The encoder networks are implemented using Transformer-based architectures, such as GPT-2 (Radford et al. 2019), to capture long-range dependencies.

Let $\mathbf{P}_{trend}, \mathbf{P}_{season}, \mathbf{P}_{residual} \in R^d$ denote learnable prompt embeddings for trend, seasonality, and residual components, serving as semantic anchors. We adopt a contrastive objective inspired by Contrastive Language-Image Pre-training (CLIP) (Radford et al. 2021), maximizing the cosine similarity between component embeddings and their associated prompt embeddings while minimizing similarity with non-associated prompt embeddings. The contrastive loss for the trend component is defined as:

$$\mathcal{L}_{contrast}^{trend} = -\frac{1}{T}\sum_{t=1}^{T}\log\left(\frac{\exp(\text{sim}(\mathbf{h}_t^{\text{trend}}, \mathbf{P}_{\text{trend}})/\tau)}{\sum_{c\in\{\text{trend,season,residual}\}}} \frac{\exp(\text{sim}(\mathbf{h}_t^{\text{trend}}, \mathbf{P}_{\text{trend}})/\tau)}{\exp(\text{sim}(\mathbf{h}_t^{\text{trend}}, \mathbf{P}_{c})/\tau)}\right) \tag{12}$$

where sim$(\cdot)$ denotes the cosine similarity function and τ is a temperature hyperparameter. Contrastive losses for seasonality and residual components are defined similarly. The total contrastive loss for semantic alignment is the sum of all component contrastive losses:

$$\mathcal{L}_{align} = \frac{1}{N}\sum_{i=1}^{N}\left(\mathcal{L}_{contrast}^{\text{trend}} + \mathcal{L}_{contrast}^{\text{season}} + \mathcal{L}_{contrast}^{\text{residual}}\right) \tag{13}$$

In addition to contrastive loss-based semantic alignment, we introduce a CLIP loss-based semantic alignment technique. We concatenate trend, seasonality, and residual components along the feature dimension and input them into a pre-trained CLIP text encoder to obtain semantic embedding representations:

$$\mathbf{H}_{clip} = f_{CLIP}(\text{concat}(\mathbf{T}, \mathbf{S}, \mathbf{R})) \tag{14}$$

where $\mathbf{H}_{clip} \in R^{T\times d_{clip}}$ is the generated CLIP embedding. We compute the cosine similarity between the combined embedding of time series components $\mathbf{H}_{comb} \in R^{T\times d}$ and the CLIP embedding $\mathbf{H}_{clip}$ as the CLIP loss:

$$\mathcal{L}_{clip} = -\frac{1}{T}\sum_{t=1}^{T}\text{sim}(\mathbf{h}_t^{comb}, \mathbf{h}_t^{clip}) \tag{15}$$

The final semantic alignment loss is a weighted sum of the contrastive loss and CLIP loss:

$$\mathcal{L}_{semantic} = \mathcal{L}_{align} + \lambda\mathcal{L}_{clip} \tag{16}$$

where λ is a hyperparameter balancing the two losses. By minimizing $\mathcal{L}_{semantic}$, the model learns to align time series component embeddings with prompt embeddings and universal semantic representations captured by the CLIP encoder, enhancing the semantic relevance and interpretability of learned time series representations.

Unified Loss Function with Dynamic Weighting

To adaptively balance different learning objectives, we formulate the training process as a multi-task learning problem(Xue et al. 2024, 2023; Lin et al. 2024). The overall loss function $\mathcal{L}$ is defined using the log-sum-exp operation over three independent loss terms:

$$\begin{aligned}\mathcal{L} = \frac{1}{\beta}\log\big(&\exp(\beta\mathcal{L}_{predict})\\ &+\exp(\beta\mathcal{L}_{semantic})\\ &+\exp(\beta\mathcal{L}_{\text{STL}})\big)\end{aligned} \tag{17}$$

where $\mathcal{L}_{predict}$ is the forecasting loss, $\mathcal{L}_{semantic}$ is the semantic alignment loss, and $\mathcal{L}_{\text{STL}}$ is the time series decomposition loss. The hyperparameter $\beta > 0$ controls the smoothness of the log-sum-exp function.

The forecasting loss $\mathcal{L}_{predict}$ measures the discrepancy between predicted values $\hat{\mathbf{X}}_{t+1:t+H}$ and true future values $\mathbf{X}_{t+1:t+H}$. Predictions are generated by concatenating learned embeddings of trend, seasonality, and residual

Table 1: Transfer learning of long-term forecasting results on time series benchmark datasets. We use prediction length $O \in \{96, 192, 336, 720\}$. A lower MSE indicates better performance. Hereafter, for the tables, the best line are marked in gray, respectively with MSE/MAE.

Horizon	Model	ECL	Traffic	Weather	Ettm1	Ettm2	Etth1	Etth2
		MSE/MAE	MSE/MAE	MSE/MAE	MSE/MAE	MSE/MAE	MSE/MAE	MSE/MAE
96	STEM-LTS*	0.142/0.251	0.428/0.286	0.161/0.210	0.316/0.361	0.138/0.229	0.336/0.328	0.256/0.270
	TEMPO*	0.196/0.295	0.530/0.379	0.222/0.265	0.422/0.423	0.197/0.283	0.425/0.424	0.317/0.352
	LLM4TS*	0.151/0.258	0.507/0.365	0.204/0.262	0.468/0.419	0.176/0.255	0.381/0.392	0.307/0.345
	GPT4TS*	0.188/0.286	0.523/0.380	0.232/0.281	0.484/0.442	0.192/0.275	0.408/0.414	0.337/0.372
	T5	0.185/0.282	0.508/0.366	0.217/0.271	0.529/0.464	0.190/0.268	0.400/0.409	0.328/0.366
	PatchTST	0.489/0.546	1.023/0.641	0.247/0.301	0.733/0.554	0.273/0.345	0.570/0.518	0.379/0.412
	Timesnet	0.293/0.369	0.585/0.401	0.247/0.295	0.518/0.470	0.202/0.290	0.407/0.423	0.315/0.362
	FEDformer	0.300/0.399	0.835/0.564	0.292/0.346	0.698/0.553	0.665/0.634	0.509/0.502	0.385/0.426
	ETSformer	0.707/0.638	1.419/0.795	0.453/0.416	1.117/0.678	0.353/0.404	0.469/0.457	0.405/0.428
	Informer	0.512/0.531	1.400/0.830	0.837/0.711	0.880/0.657	0.263/0.360	0.642/0.562	0.704/0.651
	DLinear	0.195/0.292	0.609/0.424	0.212/0.275	0.624/0.522	0.264/0.352	0.414/0.421	0.334/0.389
192	STEM-LTS*	0.165/0.273	0.451/0.286	0.183/0.236	0.334/0.393	0.162/0.249	0.367/0.338	0.274/0.300
	TEMPO*	0.213/0.310	0.561/0.391	0.286/0.314	0.512/0.472	0.255/0.318	0.462/0.457	0.408/0.412
	LLM4TS*	0.192/0.297	0.516/0.361	0.223/0.288	0.480/0.441	0.256/0.317	0.424/0.419	0.370/0.386
	GPT4TS*	0.209/0.302	0.524/0.379	0.283/0.323	0.508/0.461	0.248/0.308	0.437/0.433	0.380/0.400
	T5	0.205/0.302	0.524/0.374	0.277/0.321	0.523/0.454	0.246/0.306	0.428/0.426	0.413/0.410
	PatchTST	0.465/0.535	0.992/0.633	0.277/0.324	0.739/0.563	0.299/0.355	0.580/0.528	0.387/0.417
	Timesnet	0.283/0.366	0.640/0.431	0.316/0.342	0.550/0.490	0.261/0.318	0.439/0.439	0.394/0.406
	FEDformer	0.390/0.468	0.869/0.579	0.372/0.426	0.819/0.608	0.358/0.416	0.683/0.596	0.921/0.748
	ETSformer	0.721/0.645	0.995/0.658	0.545/0.466	1.598/0.803	0.390/0.416	0.548/0.503	0.476/0.468
	Informer	0.625/0.619	0.872/0.506	0.431/0.455	1.461/0.892	0.494/0.516	0.798/0.632	0.455/0.883
	DLinear	0.204/0.300	0.595/0.412	0.259/0.308	0.599/0.511	0.292/0.365	0.439/0.437	0.381/0.415
336	STEM-LTS*	0.175/0.301	0.476/0.308	0.225/0.236	0.433/0.405	0.207/0.291	0.384/0.360	0.300/0.335
	TEMPO*	0.234/0.329	0.589/0.403	0.337/0.349	0.511/0.476	0.275/0.319	0.476/0.467	0.419/0.452
	LLM4TS*	0.207/0.291	0.500/0.359	0.381/0.362	0.703/0.615	0.281/0.313	0.435/0.426	0.414/0.432
	GPT4TS*	0.226/0.315	0.535/0.383	0.407/0.379	0.655/0.523	0.299/0.343	0.450/0.442	0.407/0.423
	T5	0.229/0.321	0.550/0.391	0.330/0.330	0.572/0.504	0.316/0.346	0.442/0.438	0.416/0.427
	PatchTST	0.531/0.569	0.987/0.626	0.317/0.347	0.755/0.576	0.342/0.382	0.677/0.573	0.386/0.425
	Timesnet	0.733/0.633	1.609/0.864	0.359/0.372	0.638/0.532	0.380/0.392	0.555/0.503	0.384/0.413
	FEDformer	0.317/0.406	1.006/0.640	0.639/0.600	0.785/0.624	0.372/0.424	0.582/0.542	-/5.755
	ETSformer	0.862/0.707	0.940/0.621	0.487/0.444	1.154/0.682	0.409/0.428	0.728/0.585	0.446/0.451
	Informer	1.222/0.863	0.978/0.507	0.370/0.412	0.949/0.631	0.788/0.622	1.125/0.810	1.389/0.848
	DLinear	0.231/0.325	0.624/0.427	0.304/0.342	0.622/0.534	0.361/0.411	0.463/0.464	0.471/0.482
720	STEM-LTS*	0.192/0.265	0.491/0.364	0.310/0.288	0.455/0.415	0.225/0.269	0.402/0.385	0.327/0.355
	LLM4TS*	0.210/0.290	0.506/0.371	0.363/0.321	0.479/0.414	0.254/0.331	0.462/0.481	0.375/0.400
	TEMPO*	0.281/0.365	0.636/0.420	0.427/0.403	0.614/0.529	0.315/0.368	0.462/0.451	0.420/0.438
	GPT4TS*	0.223/0.315	0.553/0.391	0.375/0.363	0.580/0.500	0.294/0.351	0.441/0.442	0.392/0.417
	T5	0.266/0.351	0.578/0.404	0.528/0.451	0.694/0.568	0.394/0.397	0.443/0.458	0.425/0.440
	PatchTST	0.475/0.532	1.152/0.706	0.375/0.388	0.739/0.570	0.421/0.421	0.540/0.521	0.425/0.448
	Timesnet	1.166/0.859	1.974/0.971	0.423/0.405	0.723/0.577	0.399/0.409	0.438/0.461	0.394/0.431
	FEDformer	0.423/0.480	0.965/0.652	0.409/0.425	0.816/0.614	0.455/0.462	0.688/0.618	0.427/0.452
	ETSformer	0.666/0.640	0.798/0.518	0.592/0.506	1.038/0.665	0.444/0.438	0.615/0.561	0.446/0.466
	Informer	0.881/0.778	1.532/0.800	1.133/0.842	0.779/0.616	1.075/0.725	0.836/0.687	1.330/0.866
	DLinear	0.259/0.352	0.623/0.420	0.363/0.389	0.639/0.559	0.515/0.490	0.467/0.481	0.639/0.559

Table 2: SMAPE results of EBITDA from TETS and GDELT. The results for EBITDA include outliers removed where SMAPE exceeds 0.8/0.9. The best results are marked in **bold**, and the second-best results are <u>underlined</u>, respectively, for 0.8 & 0.9. (*Sectors*: CC: Consumer Cyclical; CD: Consumer Defensive; Ind: Industrials; RE: Real Estate; *Events*: 11: Disapprove; 17: Coerce; 19: Fight.)

				EBITDA Dataset					
Sectors	STEM-LTS	TEMPO	LLM4TS	GPT4TS	T5	Informer	PatchTST	Reformer	DLinear
CC	**30.35/31.56**	35.81/36.48	34.06/35.38	33.98/35.56	<u>33.42/35.33</u>	41.12/43.17	41.44/43.18	37.23/39.09	33.53/35.65
CD	**25.84/26.50**	27.45/28.00	27.32/27.79	27.16/27.45	<u>26.44/26.79</u>	35.65/36.08	31.60/31.98	29.93/30.36	27.01/28.04
Ind	**26.99/27.38**	29.01/29.42	28.68/29.11	27.90/28.63	<u>27.30/28.12</u>	34.83/35.87	33.84/34.87	30.23/31.28	27.59/28.84
RE	**28.80/29.31**	31.08/31.56	30.63/31.00	30.82/31.54	30.10/30.64	36.40/37.22	37.63/38.31	31.23/31.69	<u>29.95/30.92</u>
				GDELT Dataset					
11	**39.88**	41.75	41.23	40.43	41.04	42.00	40.45	46.72	<u>40.14</u>
17	**41.59**	42.56	42.60	<u>41.20</u>	41.24	44.44	42.72	48.08	42.45
19	**44.00**	45.13	44.59	<u>44.06</u>	44.29	47.45	45.49	48.30	45.40

components, then inputting them into a prediction module $f_{pred}(\cdot)$:

$$\hat{\mathbf{X}}_{t+1:t+H} = f_{pred}([\mathbf{H}_{trend} \oplus \mathbf{H}_{season} \oplus \mathbf{H}_{residual}]; \Theta_{pred}) \tag{18}$$

where $\hat{\mathbf{X}}_{t+1:t+H} \in R^{H \times D}$ represents the predicted values over the forecast horizon H. The prediction module $f_{pred}(\cdot)$ is implemented using a Transformer-based sequence model.

During training, the STEM-LTS framework is optimized end-to-end by minimizing the adaptive multi-task loss function $\mathcal{L}$. Model parameters are iteratively updated to minimize the overall loss and improve forecasting performance. The adaptive multi-task learning framework enables the model to leverage complementary information from different loss terms and adapt to the characteristics of time series data during training.

Experiments

We rigorously evaluate STEM-LTS through extensive experiments on diverse real-world datasets, demonstrating its superiority in temporal dependency modeling and semantic pattern alignment. Our analysis focuses on temporal component correlation and loss weighting strategies to validate improvements in prediction accuracy, interpretability, and training efficiency. Detailed results are provided in the Appendix.

Experimental Setup

To ensure comprehensive evaluation, we conduct experiments on three complementary datasets: EBITDA for corporate financial metrics with long-term trends and cyclical patterns (Lai et al. 2018), GDELT for global event dynamics with complex temporal dependencies (Zhou et al. 2021), and standard benchmarks (ECL, Traffic, Weather, Ettm1/2, Etth1/2) representing diverse domains and temporal complexities (Wu et al. 2021; Zhou et al. 2021). Performance is measured using standard MSE and MAE metrics, with detailed dataset descriptions provided in Appendix B.

We compare STEM-LTS against both classical statistical models like DLinear (Zeng et al. 2023) and advanced deep architectures, including Transformer-based models (TEMPO (Cao et al. 2024), LLM4TS (Chang et al. 2024), GPT4TS (Zhou et al. 2023)), attention-based approaches (T5 (Raffel et al. 2020), PatchTST (Nie et al. 2023), TimesNet (Wu et al. 2023)), and their extensions (FEDformer (Zhou et al. 2022), ETSformer (Woo et al. 2023), Informer (Zhou et al. 2021)).

Implementation Details

All experiments were conducted on a server equipped with two NVIDIA Tesla V100-PCIE-16GB GPUs. STEM-LTS was implemented using the PyTorch deep learning framework (Paszke et al. 2019), with TEMPO as the backbone network. For each dataset, we first loaded the pre-trained TEMPO model parameters into STEM-LTS and then fine-tuned it using the Adam optimizer (Kingma and Ba 2015) and CosineAnnealingLR scheduler (Loshchilov and Hutter 2017) for 10 epochs. The initial learning rate was set to 0.001, with a maximum of 20 training iterations per epoch and a minimum learning rate of 1e-8. These hyperparameters were optimized through grid search and cross-validation.

Experimental Results

Results on EBITDA and GDELT Datasets Table 2 presents the SMAPE results of STEM-LTS and the baselines on the EBITDA and GDELT datasets. For the EBITDA dataset, we report results with outliers removed using thresholds of 0.8 and 0.9. STEM-LTS consistently outperforms all baselines across both datasets and all sectors/events, achieving the best performance in all cases. The second-best results are underlined, highlighting the competitive performance of some baselines, such as T5 and GPT4TS.

Results on Benchmark Datasets Table 1 shows the transfer learning results for long-term forecasting on the benchmark datasets, with prediction lengths ranging from 96 to

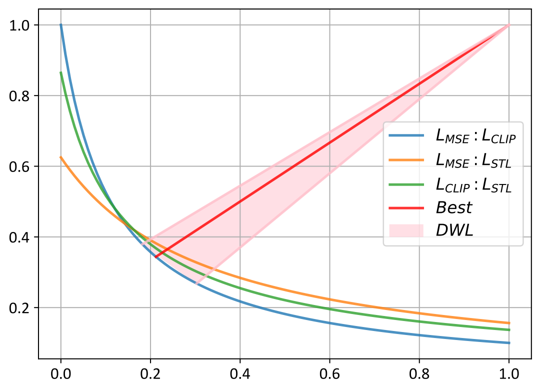

Figure 2: Dynamic Weighting vs. Static Weighting: This figure compares the normalized losses using static weighting (blue, green, yellow lines) and dynamic weighting (DWL, pink shaded area). While static weighting requires exploration of all possible combinations to find the optimal weights, dynamic weighting efficiently converges near the optimal solution in a single run, balancing computational efficiency and loss minimization.

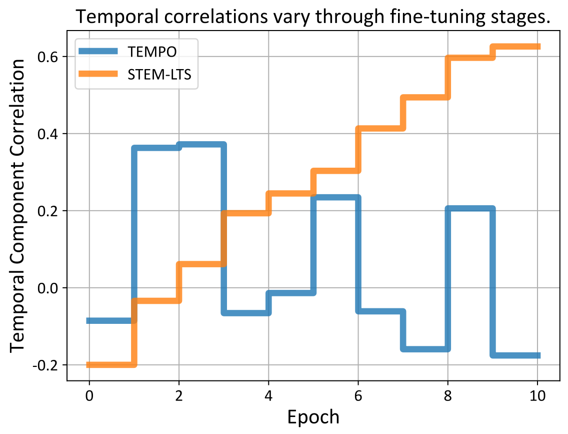

Figure 3: Comparison of Temporal Component Correlation: This figure illustrates the variation of temporal component correlation during fine-tuning stages for TEMPO (blue) and STEM-LTS (orange). STEM-LTS consistently enhances correlation, demonstrating superior alignment of time series and semantic features compared to TEMPO.

720. STEM-LTS demonstrates superior performance compared to the baselines, achieving the lowest MSE and MAE scores across all datasets and prediction horizons. The best results are highlighted in gray. These results underscore the effectiveness of STEM-LTS in capturing complex spatiotemporal dependencies and multi-scale dynamics in time series data.

Our experimental results demonstrate the superiority of STEM-LTS over state-of-the-art methods in both domain-specific and benchmark datasets. The framework's integration of semantic-temporal modeling and adaptive multi-task learning within the LLM paradigm enables accurate and robust time series forecasting, even for long-term horizons and complex patterns.

Experimental Analysis and Discussion

Temporal Component Correlation Analysis. Figure 3 shows the temporal component correlation metrics for STEM-LTS and TEMPO during fine-tuning. STEM-LTS continuously improves the metrics, while TEMPO exhibits fluctuations and inferior performance. This suggests that STEM-LTS better aligns time series numerical patterns with semantic representations, capturing complex temporal dependencies and enhancing prediction accuracy and interpretability. By integrating time series decomposition with LLMs, STEM-LTS demonstrates superior performance and robustness in time series analysis.

Loss Weighting Strategy Analysis. Figure 2 compares different loss weighting strategies. Static weighting requires traversing all possible weight combinations to find the optimal solution, which is challenging for large models and lacks generalization. In contrast, the dynamic weighting method (DWL) converges near the optimum in a single execution, achieving a balance between computational efficiency and loss minimization. STEM-LTS utilizes dynamic weight allocation to effectively balance the weights of different loss terms, improving training efficiency and prediction performance, demonstrating its advantage in capturing complex temporal dependencies.

Conclusion

STEM-LTS integrates time series decomposition, semantic temporal alignment, and large language models for effective time series analysis. It captures complex temporal dependencies, aligns numerical patterns with semantic representations, and leverages knowledge from pre-trained language models. The temporal component correlation analysis and loss weighting strategy analysis demonstrate STEM-LTS's effectiveness in improving prediction accuracy, interpretability, and training efficiency.

Acknowledgements

This work was supported by the Natural Science Foundation of China Youth Project (No. 62402472), the Natural Science Foundation of Jiangsu Province of China Youth Project (No. BK20240461,BK20240460), the Research Grants Council of the Hong Kong Special Administrative Region, China (GRF Project No. CityU 11215723), National Natural Science Foundation of China (No.62072427, No.12227901), the Project of Stable Support for Youth Team in Basic Research Field, CAS (No.YSBR-005), Academic Leaders Cultivation Program, USTC and the Key Basic Research Foundation of Shenzhen, China (JCYJ20220818100005011).

References

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. In *International Conference on Learning Representations*.

Box, G. E.; Jenkins, G. M.; Reinsel, G. C.; and Ljung, G. M. 2015. Time series analysis: forecasting and control. *John Wiley & Sons*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Cao, D.; Jia, F.; Arik, S. O.; Pfister, T.; Zheng, Y.; Ye, W.; and Liu, Y. 2024. TEMPO: Prompt-based Generative Pre-trained Transformer for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations*.

Chang, C.; Wang, W.-Y.; Peng, W.-C.; and Chen, T.-F. 2024. LLM4TS: Aligning Pre-Trained LLMs as Data-Efficient Time-Series Forecasters. *arXiv preprint arXiv:2308.08459*.

Cleveland, R. B.; Cleveland, W. S.; McRae, J. E.; and Terpenning, I. 1990. STL: A seasonal-trend decomposition. *Journal of official statistics*, 6(1): 3–73.

De Livera, A. M.; Hyndman, R. J.; and Snyder, R. D. 2011. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American statistical association*, 106(496): 1513–1527.

Durbin, J.; and Koopman, S. J. 2012. Time series analysis by state space methods. *Oxford university press*.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Jia, F.; Wang, K.; Zheng, Y.; Cao, D.; and Liu, Y. 2024. GPT4MTS: Prompt-based Large Language Model for Multimodal Time-series Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 23343–23351.

Kazemi, S. M.; Goel, R.; Eghbali, S.; Ramanan, J.; Sahota, J.; Thakur, S.; Wu, S.; Smyth, C.; Poupart, P.; and Brubaker, M. 2019. Time2vec: Learning a vector representation of time. In *International Joint Conference on Artificial Intelligence*, 2561–2567.

Kingma, D. P.; and Ba, J. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Lai, G.; Chang, W.-C.; Yang, Y.; and Liu, H. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, 95–104.

Lim, B.; Arık, S. Ö.; Loeff, N.; and Pfister, T. 2021. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4): 1748–1764.

Lim, B.; and Zohren, S. 2021. Time series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194): 20200209.

Lin, X.; Zhang, X.; Yang, Z.; Liu, F.; Wang, Z.; and Zhang, Q. 2024. Smooth Tchebycheff Scalarization for Multi-Objective Optimization. *arXiv preprint arXiv:2402.19078*.

Loshchilov, I.; and Hutter, F. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations*.

Malhotra, P.; TV, V.; Vig, L.; Agarwal, P.; and Shroff, G. 2017. Timenet: Pre-trained deep recurrent neural network for time series classification. In *25th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 607–612.

Miao, H.; Shen, J.; Cao, J.; Xia, J.; and Wang, S. 2022. MBA-STNet: Bayes-enhanced Discriminative Multi-task Learning for Flow Prediction. *TKDE*.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *International Conference on Learning Representations*.

Oreshkin, B. N.; Carpov, D.; Chapados, N.; and Bengio, Y. 2019. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*.

Papadimitriou, A.; Patel, U.; Kim, L.; Bang, G.; Nematzadeh, A.; and Liu, X. 2020. A multi-faceted approach to large scale financial forecasting. In *Proceedings of the First ACM International Conference on AI in Finance*, 1–8.

Paszke, A.; et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32: 8026–8037.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 8748–8763. PMLR.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. In *Journal of Machine Learning Research*, volume 21, 1–67.

Ruder, S. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wen, Q.; Zhou, T.; Zhang, C.; Chen, W.; Ma, Z.; Yan, J.; and Sun, L. 2022. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*.

Woo, D.; Park, S.; Jung, J.; and Kim, T. 2023. ETSformer: Exponential Smoothing Transformers for Time-series Forecasting. *Advances in Neural Information Processing Systems*, 36.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, volume 34, 22419–22430.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *International Conference on Learning Representations*.

Xu, R.; Miao, H.; Wang, S.; Yu, P. S.; and Wang, J. 2024. PeFAD: A Parameter-Efficient Federated Framework for Time Series Anomaly Detection. In *SIGKDD*, 3621–3632.

Xue, B.; Cheng, J.; Liu, F.; Wang, Y.; and Zhang, Q. 2024. Multiobjective Lipschitz Bandits under Lexicographic Ordering. *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, 16238–16246.

Xue, B.; Wang, Y.; Wan, Y.; Yi, J.; and Zhang, L. 2023. Efficient Algorithms for Generalized Linear Bandits with Heavy-tailed Rewards. In *Advances in Neural Information Processing Systems 36*, 70880–70891.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are Transformers Effective for Time Series Forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 11121–11128.

Zerveas, G.; Jayaraman, S.; Patel, D.; Bhamidipaty, A.; and Eickhoff, C. 2021. A transformer-based framework for multivariate time series representation learning. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2114–2124.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 11106–11115.

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In *International Conference on Machine Learning*, 27268–27286.

Zhou, T.; Niu, P.; Wang, X.; Sun, L.; and Jin, R. 2023. One Fits All: Power General Time Series Analysis by Pretrained LM. *arXiv preprint arXiv:2302.11939*.

Reproducibility Checklist

This paper:

- Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes)
- Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes)
- Provides well marked pedagogical references for less-familiare readers to gain background necessary to replicate the paper (yes)

Does this paper make theoretical contributions? (no)
If yes, please complete the list below.

- All assumptions and restrictions are stated clearly and formally. (yes/partial/no)
- All novel claims are stated formally (e.g., in theorem statements). (yes/partial/no)
- Proofs of all novel claims are included. (yes/partial/no)
- Proof sketches or intuitions are given for complex and/or novel results. (yes/partial/no)
- Appropriate citations to theoretical tools used are given. (yes/partial/no)
- All theoretical claims are demonstrated empirically to hold. (yes/partial/no/NA)
- All experimental code used to eliminate or disprove claims is included. (yes/no/NA)

Does this paper rely on one or more datasets? (yes)
If yes, please complete the list below.

- A motivation is given for why the experiments are conducted on the selected datasets (yes)
- All novel datasets introduced in this paper are included in a data appendix. (NA)
- All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (NA)
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations. (yes)
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available. (yes)
- All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing. (NA)

Does this paper include computational experiments? (yes)
If yes, please complete the list below.

- Any code required for pre-processing data is included in the appendix. (yes).
- All source code required for conducting and analyzing the experiments is included in a code appendix. (yes)
- All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (yes)
- All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes)
- If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results. (NA)
- This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks. (yes)
- This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics. (yes)
- This paper states the number of algorithm runs used to compute each reported result. (yes)
- Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information. (yes)
- The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank). (yes)
- This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments. (NA)
- This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting. (NA)

Appendix of STEM-LTS

The content of the **Appendix** is summarized as follows:

A) In Sec. A, we provide a theoretical analysis of the STEM-LTS framework, focusing on the convergence properties and generalization bounds of the proposed multi-scale time series decomposition and regularization approach.

B) In Sec. B, we demonstrate the details of datasets and baselines used in our experiments, including the EBITDA, GDELT, and various time series benchmarks from different domains. We also provide a detailed description of the baseline methods.

C) In Sec. C, we present the pseudocode of the STEM-LTS algorithm, outlining the training procedure and key components of our method.

D) In Sec. D, we provide more analysis of our method, including an ablation study to assess the effectiveness of different components in STEM-LTS and a semantic alignment visualization using t-SNE to demonstrate the ability of our model to capture meaningful semantic alignments between time series components and their corresponding prompts.

E) In Sec. E, we analyze the time complexity of the STEM-LTS algorithm, focusing on the logsumexp function used in multi-objective optimization. We highlight the efficiency and scalability of our method, with an overall time complexity of $\mathcal{O}(T)$ and the ability of the logsumexp function to simultaneously optimize multiple objectives with constant time complexity.

A Theoretical Analysis of STEM-LTS

In this section, we provide a theoretical analysis of the STEM-LTS framework, focusing on the convergence properties and generalization bounds of the proposed multi-scale time series decomposition and regularization approach. We introduce several lemmas, theorems, and propositions to support our claims and provide rigorous mathematical proofs.

Lemma 1. *Let $\mathbf{Z}_d \in R^{3\times T}$ be the d-th feature slice of the decomposed time series tensor $\mathbf{Z}$, and let Σ_d be its covariance matrix. Suppose the eigenvalues of Σ_d are bounded by $\lambda_{\max}$. Then, the strict lower triangular part of the matrix exponential of Σ_d satisfies:*

$$\sum_{i>j}[\exp(\textit{stril}(\Sigma_d))]_{ij} \leq \frac{3(3-1)}{2}\exp(\lambda_{\max}) \tag{19}$$

Proof. The matrix exponential of a matrix $\mathbf{A}$ is defined as:

$$\exp(\mathbf{A}) = \sum_{k=0}^{\infty}\frac{\mathbf{A}^k}{k!} \tag{20}$$

For a strict lower triangular matrix $\mathbf{L}$, we have $\mathbf{L}^k = \mathbf{0}$ for all $k \geq 3$, as the product of three or more strict lower triangular matrices is always zero. Therefore, the matrix exponential of stril(Σ_d) can be written as:

$$\exp(\text{stril}(\Sigma_d)) = \mathbf{I} + \text{stril}(\Sigma_d) + \frac{1}{2}(\text{stril}(\Sigma_d))^2 \tag{21}$$

Since the eigenvalues of Σ_d are bounded by $\lambda_{\max}$, the elements of stril(Σ_d) are also bounded by $\lambda_{\max}$. Thus, we have:

$$\sum_{i>j}[\exp(\text{stril}(\Sigma_d))]_{ij} \leq \frac{3(3-1)}{2}\exp(\lambda_{\max}) \tag{22}$$

which completes the proof. □

Theorem 1. *Suppose the time series decomposition loss $\mathcal{L}_{STL}$ is minimized using a gradient descent algorithm with a learning rate η satisfying $0 < \eta \leq \frac{2}{\beta}$, where β is the smoothness parameter in the log-sum-exp loss function. Then, the algorithm converges to a stationary point of the overall loss function $\mathcal{L}$.*

Proof. The overall loss function $\mathcal{L}$ is a smooth function of the model parameters due to the log-sum-exp operation. Its gradient with respect to the model parameters Θ can be written as:

$$\begin{aligned}\nabla_\Theta\mathcal{L} = \frac{1}{\sum_i \exp(\beta\mathcal{L}_i)}\big(&\exp(\beta\mathcal{L}_{predict})\nabla_\Theta\mathcal{L}_{predict}\\ &+\exp(\beta\mathcal{L}_{semantic})\nabla_\Theta\mathcal{L}_{semantic}\\ &+\exp(\beta\mathcal{L}_{\text{STL}})\nabla_\Theta\mathcal{L}_{\text{STL}}\big)\end{aligned} \tag{23}$$

where $\mathcal{L}_i \in \{\mathcal{L}_{predict}, \mathcal{L}_{semantic}, \mathcal{L}_{\text{STL}}\}$. The gradient of the time series decomposition loss $\mathcal{L}_{\text{STL}}$ with respect to the model parameters can be computed using the chain rule:

$$\nabla_\Theta\mathcal{L}_{\text{STL}} = \frac{1}{D}\sum_{d=1}^{D}\frac{1}{M}\sum_{i>j}\nabla_\Theta[\exp(\text{stril}(\Sigma_d))]_{ij} \tag{24}$$

By Lemma 1, the gradient of $\mathcal{L}_{\text{STL}}$ is bounded. Therefore, the overall gradient $\nabla_\Theta\mathcal{L}$ is also bounded, as it is a convex combination of bounded gradients.

Given a learning rate η satisfying $0 < \eta \leq \frac{2}{\beta}$, the gradient descent update rule for the model parameters is:

$$\Theta^{(t+1)} = \Theta^{(t)} - \eta\nabla_\Theta\mathcal{L}(\Theta^{(t)}) \tag{25}$$

where $\Theta^{(t)}$ denotes the model parameters at iteration t. By the convergence theorem of gradient descent for smooth functions with bounded gradients (Nesterov, 2013), the algorithm converges to a stationary point of the overall loss function $\mathcal{L}$. □

Proposition 1. *Let $\mathcal{H}$ be the hypothesis class of STEM-LTS models, and let $\mathcal{S} = \{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1}^N$ be a training set of N samples, where $\mathbf{X}_i \in R^{T\times D}$ is an input time series and $\mathbf{Y}_i \in R^{H\times D}$ is the corresponding target future values. Suppose the loss function $\ell(\cdot)$ is bounded by B. Then, for any $\delta > 0$, with probability at least $1-\delta$, the generalization error of the learned STEM-LTS model $\hat{f} \in \mathcal{H}$ satisfies:*

$$\begin{aligned}&E_{(\mathbf{X},\mathbf{Y})\sim\mathcal{D}}[\ell(\hat{f}(\mathbf{X}),\mathbf{Y})] \leq\\ &\frac{1}{N}\sum_{i=1}^{N}\ell(\hat{f}(\mathbf{X}_i),\mathbf{Y}_i)+\\ &B\sqrt{\frac{2\log(2/\delta)}{N}}\end{aligned} \tag{26}$$

where $\mathcal{D}$ is the underlying data distribution.

Proof. The proof follows from the standard generalization bound for bounded loss functions using Hoeffding's inequality (Shalev-Shwartz & Ben-David, 2014). Let $\mathcal{L}_{\mathcal{D}}(f) = E_{(\mathbf{X},\mathbf{Y})\sim\mathcal{D}}[\ell(f(\mathbf{X}), \mathbf{Y})]$ be the expected loss over the data distribution, and let $\mathcal{L}_{\mathcal{S}}(f) = \frac{1}{N}\sum_{i=1}^{N}\ell(f(\mathbf{X}_i), \mathbf{Y}_i)$ be the empirical loss over the training set. By Hoeffding's inequality, for any fixed $f \in \mathcal{H}$ and $\epsilon > 0$, we have:

$$P[|\mathcal{L}_{\mathcal{D}}(f) - \mathcal{L}_{\mathcal{S}}(f)| \geq \epsilon] \leq 2\exp\left(-\frac{2N\epsilon^2}{B^2}\right) \tag{27}$$

Setting the right-hand side equal to δ and solving for ϵ yields:

$$\epsilon = B\sqrt{\frac{\log(2/\delta)}{2N}} \tag{28}$$

Therefore, with probability at least $1-\delta$, for any $f \in \mathcal{H}$, we have:

$$\mathcal{L}_{\mathcal{D}}(f) \leq \mathcal{L}_{\mathcal{S}}(f) + B\sqrt{\frac{2\log(2/\delta)}{N}} \tag{29}$$

Since this bound holds for any $f \in \mathcal{H}$, it also holds for the learned model $\hat{f}$, which completes the proof. □

Proposition 1 provides a generalization bound for the STEM-LTS framework, showing that the expected loss over the data distribution can be bounded by the empirical loss on the training set plus a complexity term that depends on the sample size and the loss function bound. This result suggests that the STEM-LTS model can generalize well to unseen data, given a sufficiently large training set and a bounded loss function.

In summary, the theoretical analysis presented in this section demonstrates the convergence properties and generalization bounds of the STEM-LTS framework. The convergence of the gradient descent algorithm to a stationary point of the overall loss function is guaranteed under certain conditions on the learning rate, while the generalization bound provides insights into the model's performance on unseen data. These theoretical results support the effectiveness and robustness of the proposed STEM-LTS approach for multivariate time series forecasting.

B Detailed Description of Experiments

B.1 EBITDA Dataset

The EBITDA (Earnings Before Interest, Taxes, Depreciation, and Amortization) dataset contains financial earnings data from publicly listed companies across various industries, such as consumer cyclical, industrials, and energy (Papadimitriou et al. 2020). This dataset is particularly useful for analyzing long-term trends and cyclical patterns in company financials. The time series in this dataset exhibit complex dependencies, seasonality, and varying volatility, making it a challenging benchmark for evaluating the performance of forecasting models. The dataset includes quarterly EBITDA values for a large number of companies over a period of 10 years, providing a rich set of time series with diverse characteristics.

B.2 GDELT Dataset

The Global Database of Events, Language, and Tone (GDELT) is a comprehensive dataset that includes global event information, such as disapproval, coercion, and conflicts (Jia et al. 2024). GDELT captures the complex dynamics of global events and their interactions across different countries and regions. The dataset is constructed by processing and analyzing a vast amount of news articles and reports from various sources worldwide. Each event in the dataset is characterized by multiple attributes, such as event type, location, timestamp, and actors involved. The time series in GDELT exhibit intricate patterns, abrupt changes, and long-range dependencies, reflecting the complex nature of global events. This dataset serves as a valuable benchmark for evaluating the ability of forecasting models to capture and predict the evolution of global dynamics.

B.3 Time Series Benchmarks

In addition to the EBITDA and GDELT datasets, we also evaluated our proposed STEM-LTS framework on a set of standard time series benchmarks from various domains, including energy, transportation, and meteorology . These benchmarks encompass a wide range of time series complexities and characteristics, allowing for a comprehensive assessment of the performance of forecasting models. The specific benchmarks used in our experiments are:

- **ECL:** The Electricity Consumption Load (ECL) dataset contains electricity consumption data from different regions, exhibiting daily and weekly seasonality patterns and long-term trends.
- **Traffic:** The Traffic dataset includes traffic flow measurements from various road segments, capturing the temporal dynamics of traffic conditions.
- **Weather:** The Weather dataset contains meteorological measurements, such as temperature, humidity, and wind speed, from multiple weather stations, exhibiting complex spatial and temporal dependencies.
- **Ettm1/2:** The Electricity Transformer Temperature (Ettm) datasets contain temperature measurements from electrical transformers, reflecting the thermal dynamics of the transformers under different load conditions.
- **Etth1/2:** The Electricity Total Temperature (Etth) datasets include electricity consumption and temperature measurements, capturing the relationship between energy demand and weather conditions.

These benchmark datasets provide a diverse set of time series with different characteristics, allowing for a rigorous evaluation of the generalization ability and robustness of forecasting models across various domains.

B.4 Detailed Description of Baselines

Traditional Methods We compared our proposed STEM-LTS framework against classical statistical models, such as DLinear (Zeng et al. 2023). DLinear is a linear dynamical system model that assumes a linear relationship between the past and future values of a time series. It captures the temporal dependencies using a state-space representation and estimates the model parameters using maximum likelihood estimation. While DLinear is a simple and interpretable model, it may struggle to capture complex nonlinear patterns and long-range dependencies in time series data.

Deep Learning Models We also evaluated STEM-LTS against state-of-the-art deep learning models for time series forecasting, including Transformer-based and self-attention models. These models have shown promising results in capturing complex temporal dependencies and achieving high forecasting accuracy. The specific models considered in our experiments are:

- **TEMPO** (**?**): TEMPO is a Transformer-based model that utilizes self-attention mechanisms to capture long-range dependencies in time series data. It employs a multi-head attention architecture to learn multiple levels of temporal representations.
- **LLM4TS** (Chang et al. 2024): LLM4TS is a large language model adapted for time series forecasting. It leverages the pre-training capabilities of language models to learn meaningful representations of time series data and generates forecasts using a decoder architecture.
- **GPT4TS** (Zhou et al. 2023): GPT4TS is a variant of the GPT (Generative Pre-trained Transformer) model specifically designed for time series forecasting. It employs a generative approach to model the probability distribution of future values conditioned on the past observations.
- **T5** (Raffel et al. 2020): T5 is a text-to-text Transformer model that has been adapted for time series forecasting. It treats time series data as a sequence of tokens and learns to generate future values based on the input sequence.
- **PatchTST** (Nie et al. 2023): PatchTST is a patch-based Transformer model that divides time series into overlapping patches and applies self-attention mechanisms to capture dependencies within and across patches.
- **TimesNet** (Wu et al. 2023): TimesNet is a neural network architecture that combines convolutional and recurrent layers to capture both local and global temporal patterns in time series data. It employs a hierarchical structure to learn multi-scale representations.

In addition to these models, we also compared STEM-LTS against improved variants of Transformer-based models, such as:

- **FEDformer** (Zhou et al. 2022): FEDformer is an enhanced version of the Transformer model that incorporates frequency-aware encoding and disentangling techniques to capture periodic patterns and improve forecasting performance.
- **ETSformer** (Woo et al. 2023): ETSformer is an extension of the Transformer model that integrates exponential smoothing techniques to capture trend and seasonality components in time series data.
- **Informer** (Zhou et al. 2021): Informer is a Transformer-based model that employs a probabilistic attention mechanism to focus on informative temporal regions and reduce computational complexity.

These baseline models represent the state-of-the-art in time series forecasting and provide a comprehensive comparison to evaluate the effectiveness of our proposed STEM-LTS framework. By comparing against these diverse models, we aim to demonstrate the superior performance and robustness of STEM-LTS in capturing complex temporal dynamics and generating accurate forecasts across various datasets and application domains.

C STEM-LTS Algorithm

```
Algorithm 1: STEM-LTS Training Procedure
Input:
  • $\mathbf{x}$: input time series
  • $\mathbf{y}$: target values
  • $f_\theta$: time series encoder
  • $f_{de}$: time series decoder
  • $f_\gamma$: GPT-2 embedding function
  • $f_{gpt}$: GPT-2 model
  • $\mathcal{L}_{mse}, \mathcal{L}_{stl}, \mathcal{L}_{clip}$: loss functions
  • Opt: optimizer
  • Sch: learning rate scheduler
Output: Trained STEM-LTS model
1. Initialize $f_\theta, f_\gamma, f_{gpt}$ from pretrained TEMPO model
2. Opt ← Adam([$f_\theta, f_\gamma, f_{gpt}$], lr=$10^{-3}$)
3. Sch ← CosineAnnealingLR(Opt, $T_{max} = 20$, $\eta_{min} = 10^{-8}$)
4. while not converged do
   (a) // Seasonal and Trend decomposition using Loess
   (b) $\mathbf{X_T}, \mathbf{X_S}, \mathbf{X_R} \leftarrow$ STL($\mathbf{x}$)
   (c) $\mathbf{T_T} \leftarrow$ "Predict the future timestep given the Trend!"
   (d) $\mathbf{T_S} \leftarrow$ "Predict the future timestep given the Seasonal!"
   (e) $\mathbf{T_R} \leftarrow$ "Predict the future timestep given the Residual!"
   (f) // Encode time series and prompts
   (g) $\mathbf{x_t}, \mathbf{x_s}, \mathbf{x_r} \leftarrow f_\theta(\mathbf{X_T}), f_\theta(\mathbf{X_S}), f_\theta(\mathbf{X_R})$
   (h) $\mathbf{t_t}, \mathbf{t_s}, \mathbf{t_r} \leftarrow f_\gamma(\mathbf{T_T}), f_\gamma(\mathbf{T_S}), f_\gamma(\mathbf{T_R})$
   (i) // Generate future predictions
   (j) $\mathbf{T} \leftarrow f_{gpt}$(concat($\mathbf{x_t}, \mathbf{t_t}$))
   (k) $\mathbf{S} \leftarrow f_{gpt}$(concat($\mathbf{x_s}, \mathbf{t_s}$))
   (l) $\mathbf{R} \leftarrow f_{gpt}$(concat($\mathbf{x_r}, \mathbf{t_r}$))
   (m) // Aggregate predictions
   (n) $\mathbf{Y} \leftarrow f_{de}$(mean($\mathbf{T}, \mathbf{S}, \mathbf{R}$))
   (o) // Compute losses
   (p) $\mathcal{L}_c \leftarrow \frac{1}{3}(\mathcal{L}_{clip}(\mathbf{x_t}, \mathbf{t_t}) + \mathcal{L}_{clip}(\mathbf{x_s}, \mathbf{t_s}) + \mathcal{L}_{clip}(\mathbf{x_r}, \mathbf{t_r}))$
   (q) $\mathcal{L}_m \leftarrow \mathcal{L}_{mse}(\mathbf{Y}, \mathbf{y})$
   (r) $\mathcal{L}_s \leftarrow \mathcal{L}_{stl}(\mathbf{Y}, \mathbf{y})$
   (s) // Compute gradient using log-sum-exp
   (t) $\mathbf{g} \leftarrow \mu \cdot$ logsumexp($[\mathcal{L}_c, \mathcal{L}_m, \mathcal{L}_s]/\mu$)
   (u) // Update model parameters
   (v) Opt.step($\mathbf{g}$)
   (w) Sch.step()
5. end while
```

D More Analysis

Ablation Study

To assess the effectiveness of different components in our proposed STEM-LTS framework, we conduct an ablation study on three representative datasets: ECL, Weather, and Ettm2. Specifically, we evaluate the performance of our model under the following settings:

- STEM-LTS (ours): The complete proposed framework.
- ours w.o. DWL: STEM-LTS without the dynamic weighting loss (DWL) module.
- ours w.o. Clip&DWL: STEM-LTS without both the CLIP loss and the DWL module.
- ours w.o. STL&DWL: STEM-LTS without both the time series decomposition loss (STL) and the DWL module.

Figure 4 presents the results of the ablation study in terms of MSE and MAE metrics. The complete STEM-LTS model consistently achieves the best performance across all datasets, demonstrating the efficacy of our proposed components. Removing the DWL module (ours w.o. DWL) leads to a slight performance drop, indicating the importance of adaptively balancing different learning objectives. Further removing the CLIP loss (ours w.o. Clip&DWL) and the STL module (ours w.o. STL&DWL) results in more significant performance degradation, highlighting their crucial roles in capturing semantic alignments and learning meaningful temporal representations. These results validate the effectiveness of our key components and their synergistic contributions to the overall performance of STEM-LTS.

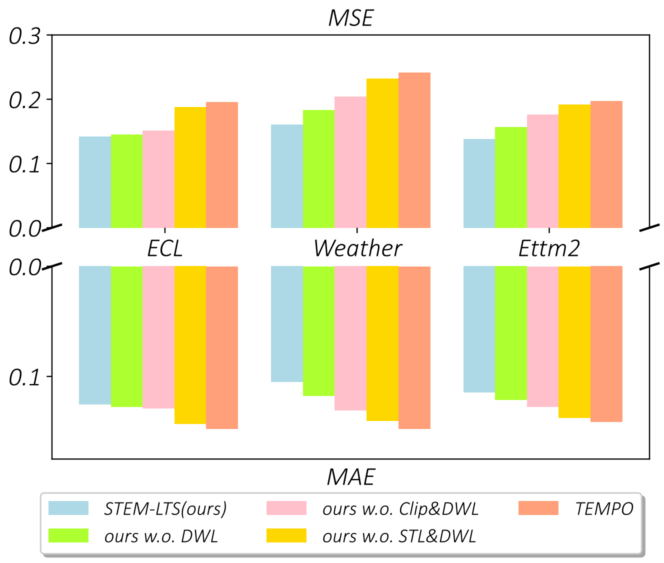

Figure 4: Ablation study results on ECL, Weather, and Ettm2 datasets.

Semantic Alignment Visualization

To further demonstrate the ability of STEM-LTS in capturing meaningful semantic alignments between time series components and their corresponding prompts, we perform t-SNE visualization on the learned feature representations. Figure 5 illustrates the t-SNE plot of the trend, season, and residual time series features along with their respective prompt features. The clear clustering of time series features with their associated prompt features indicates the effectiveness of our semantic alignment mechanism in bridging the gap between numerical patterns and semantic concepts.

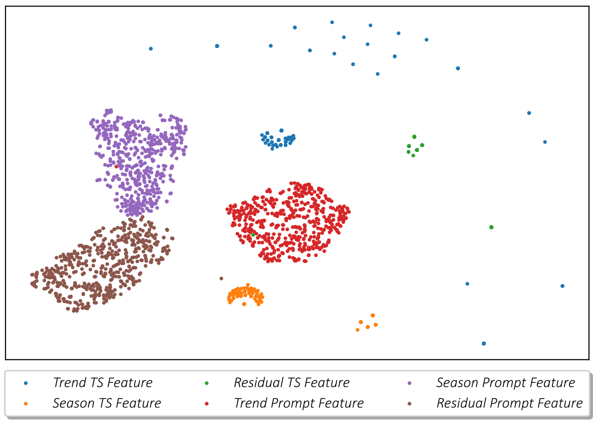

Figure 5: t-SNE visualization of time series features and prompt features.

Moreover, we visualize the t-SNE plot of the trend, season, and residual features separately in Figure 6. The distinct clusters formed by each component further validate the capability of STEM-LTS in learning disentangled and semantically meaningful representations for different time series components. This visual evidence corroborates the quantitative results and highlights the interpretability of our proposed framework.

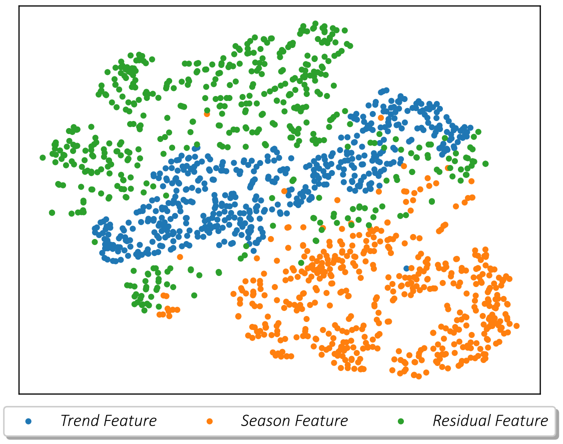

Figure 6: t-SNE visualization of trend, season, and residual features.

The ablation study and visual analysis provided offer additional insights into the functioning and effectiveness of our STEM-LTS framework. The results underscore the importance of each proposed component and their collective contribution to achieving state-of-the-art performance in multivariate time series forecasting while maintaining interpretability and semantic alignment.

E Complexity Analysis

This section analyzes the time complexity of the STEM-LTS algorithm, focusing on the logsumexp function used in multi-objective optimization. The overall time complexity of the STEM-LTS algorithm is $\mathcal{O}(T)$, where T is the length of the time series. The multi-objective optimization part, which uses the logsumexp function, has a time complexity of $\mathcal{O}(1)$. The logsumexp function efficiently achieves multi-objective optimization by computing the "soft maximum" of the input loss functions. It converts loss values of different scales to the same scale and combines them through summation. This allows simultaneous optimization of multiple objectives in a single forward pass, without explicitly computing gradients for each objective. The key advantage of using the logsumexp function is its ability to realize multi-objective optimization with $\mathcal{O}(1)$ time complexity, regardless of the time series length. This significantly improves the training efficiency of the STEM-LTS algorithm, making it suitable for handling long time series while effectively balancing multiple optimization objectives.