

# Time-VLM: Exploring Multimodal Vision-Language Models for Augmented Time Series Forecasting

Siru Zhong<sup>1</sup> Weilin Ruan<sup>1</sup> Ming Jin<sup>2</sup> Huan Li<sup>3</sup> Qingsong Wen<sup>4</sup> Yuxuan Liang<sup>1</sup>

## Abstract

Recent advancements in time series forecasting have explored augmenting models with text or vision modalities to improve accuracy. While text provides contextual understanding, it often lacks fine-grained temporal details. Conversely, vision captures intricate temporal patterns but lacks semantic context, limiting the complementary potential of these modalities. To address this, we propose Time-VLM, a novel multimodal framework that leverages pre-trained Vision-Language Models (VLMs) to bridge temporal, visual, and textual modalities for enhanced forecasting. Our framework comprises three key components: (1) a Retrieval-Augmented Learner, which extracts enriched temporal features through memory bank interactions; (2) a Vision-Augmented Learner, which encodes time series as informative images; and (3) a Text-Augmented Learner, which generates contextual textual descriptions. These components collaborate with frozen pre-trained VLMs to produce multimodal embeddings, which are then fused with temporal features for final prediction. Extensive experiments demonstrate that Time-VLM achieves superior performance, particularly in few-shot and zero-shot scenarios, thereby establishing a new direction for multimodal time series forecasting. Code is available at <https://github.com/CityMind-Lab/ICML25-TimeVLM>.

## 1. Introduction

Time series data captures temporal signal evolution and underpins forecasting in diverse domains such as finance

<sup>1</sup>The Hong Kong University of Science and Technology (Guangzhou), China <sup>2</sup>Griffith University, Australia <sup>3</sup>Zhejiang University, China <sup>4</sup>Squirrel Ai Learning, USA. Correspondence to: Yuxuan Liang <yuxliang@outlook.com>.

*Proceedings of the 42<sup>nd</sup> International Conference on Machine Learning*, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

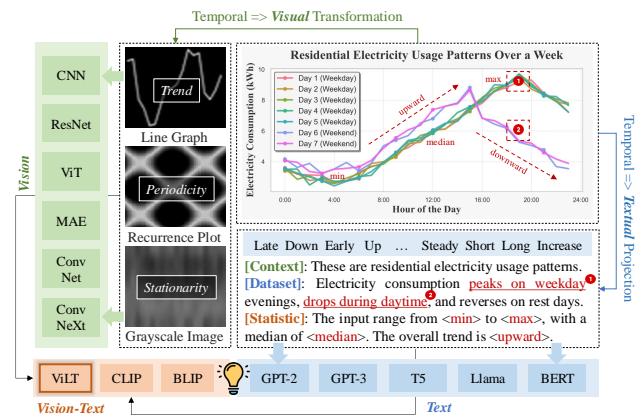


Figure 1: Our Time-VLM combines text (Right) and vision (Left) modalities to augment time series forecasting.

(Idrees et al., 2019), climate (Karevan & Suykens, 2020), energy (Deb et al., 2017), and transportation (Zheng & Huang, 2020). Accurate forecasting supports proactive risk mitigation, efficient resource allocation, and data-driven decision-making. Traditional models like ARIMA, while historically dominant, struggle to capture complex nonlinear patterns. In contrast, deep learning methods—from recurrent neural networks (RNNs) (Medsker et al., 2001) to Transformer-based architectures (Li et al., 2019; Wu et al., 2021; Zhou et al., 2021; Liu et al., 2022a; Zhou et al., 2022; Nie et al., 2023)—leverage innovations such as patch-based feature extraction, auto-correlation mechanisms, and frequency decomposition to model complex temporal dynamics. Despite their success, these models often fail to generalize across domains or adapt to data-limited scenarios, particularly few-shot and zero-shot settings (Liang et al., 2024).

To overcome these issues, researchers have turned to *augmenting time series forecasting with additional modalities*, such as *text* and *images*, which provide complementary information that can enhance predictive accuracy:

- **Text-Augmented Models:** Textual data offers valuable semantics crucial for accurate forecasting. For instance, contextual descriptions or dataset statistics can significantly enrich the understanding of time series patterns (Figure 1, right). Methods like Time-LLM (Jin et al., 2024) and UniTime (Liu et al., 2024b) leverage the superior pre-trained inference capabilities of Large Language

Models (LLMs) by mapping time series into textual representations. However, these approaches encounter two key challenges: (1) the *modality gap* between continuous time series and discrete text leads to information loss during representation alignment, and (2) pre-trained language knowledge is rare for capturing fine-grained temporal patterns, limiting their ability to learn nuanced dynamics.

- **Vision-Augmented Models:** Visual representations of time series—such as line graphs, recurrence plots, or grayscale images—enable models to exploit spatial patterns embedded within temporal data. By transforming sequences into images, methods such as CNNs, ViTs, and MAEs can extract hierarchical features that reveal latent temporal relationships (Figure 1, left). Recent studies (Wu et al., 2023b; Wang et al., 2024; Chen et al., 2024) have demonstrated the natural alignment between time series and vision, as both are continuous and share structural similarities, allowing pre-trained vision models to effectively encode temporal hierarchies. However, these methods lack semantic interpretability, restricting their capacity to incorporate domain-specific knowledge.

Despite advancements in text- and vision-augmented models, integrating both modalities with time series remains underexplored. Current approaches often *focus on single modalities, failing to harness their combined strengths*. To address this gap, we propose Time-VLM, a novel framework that leverages pre-trained VLMs to enhance time series forecasting by unifying temporal, visual, and textual information. VLMs provide a promising foundation, as they excel at aligning visual and textual modalities, making them well-suited for incorporating temporal information to unify the three modalities. By projecting time series into a unified vision-language semantic space, Time-VLM enables rich cross-modal interactions, combining the strengths of both modalities while mitigating their individual limitations. In this paradigm, each modality contributes uniquely: text provides semantic context, vision captures spatial-temporal patterns, and time series encodes sequential dynamics.

Specifically, Time-VLM introduces three key components: (1) a *Retrieval-Augmented Learner* that processes raw time series data through patch-based feature extraction and memory bank interactions to generate enriched temporal representations, capturing both local and global dependencies; (2) a *Vision-Augmented Learner* that adaptively transforms time series into images using multi-scale convolution, frequency encoding, and periodic encoding, preserving both fine-grained details and high-level structures; and (3) a *Text-Augmented Learner* that generates rich textual context (e.g., statistics and dataset descriptions) to complement the visual representations. These modules collaborate with VLMs to integrate temporal, visual, and textual modalities, producing accurate forecasts through a fine-tuned predictor.

Our key contributions can be summarized as follows:

- We propose the first framework that unifies temporal, visual, and textual modalities by leveraging their complementary strengths for enhanced time series forecasting.
- We introduce a retrieval-augmented learner for hierarchical temporal feature enhancement, a vision-augmented learner for adaptive time-series-to-image transformation, and a text-augmented learner for contextual prompt generation, enabling seamless integration with VLMs.
- Extensive evaluations show Time-VLM’s strong performance, especially under data-scarce conditions, offering a brand new paradigm for multimodal time series research.

## 2. Related Work

### Text-Augmented Models for Time Series Forecasting.

The success of LLMs has inspired their application to time series forecasting. Methods like LLMTTime (Gruver et al., 2023) and LLM4TS (Chang et al., 2023) tokenize time series for autoregressive prediction but inherit limitations such as poor arithmetic and recursive reasoning. Approaches including GPT4TS (Zhou et al., 2023) and TimeLLM (Jin et al., 2024) project time series into textual representations to leverage LLMs’ reasoning capabilities, yet face challenges like the modality gap and limited temporal adaptability of word embeddings. UniTime (Liu et al., 2024b) and TimeEFFM (Liu et al., 2024a) incorporate domain knowledge and federated learning, respectively, but remain constrained by their exclusive dependence on textual modeling.

### Vision-Augmented Models for Time Series Forecasting.

Vision provides a natural way to preserve temporal patterns in time series data. Early approaches apply CNNs to matrix-formed time series (Li et al., 2020; Sood et al., 2021), while TimesNet (Wu et al., 2023b) introduces multi-periodic decomposition for unified 2D modeling. VisionTS (Chen et al., 2024) pioneers the use of pre-trained visual encoders with grayscale time series images, and TimeMixer++ (Wang et al., 2024) advances the field through multi-scale frequency-based time-image transformations. Despite their effectiveness in temporal modeling, these methods often lack semantic context, limiting their ability to utilize high-level contextual information for prediction.

### Vision-Language Models.

VLMs like ViLT (Kim et al., 2021), CLIP (Radford et al., 2021), and ALIGN (Jia et al., 2021), have transformed multimodal understanding by aligning image and text representations. Recent progress, like BLIP-2 (Li et al., 2023) and LLaVA (Liu et al., 2023), further enhance multimodal reasoning. However, VLMs remain underexplored for time series analysis. Our work bridges this gap by leveraging VLMs to integrate temporal, visual, and textual modalities, addressing the limitations of unimodal approaches.

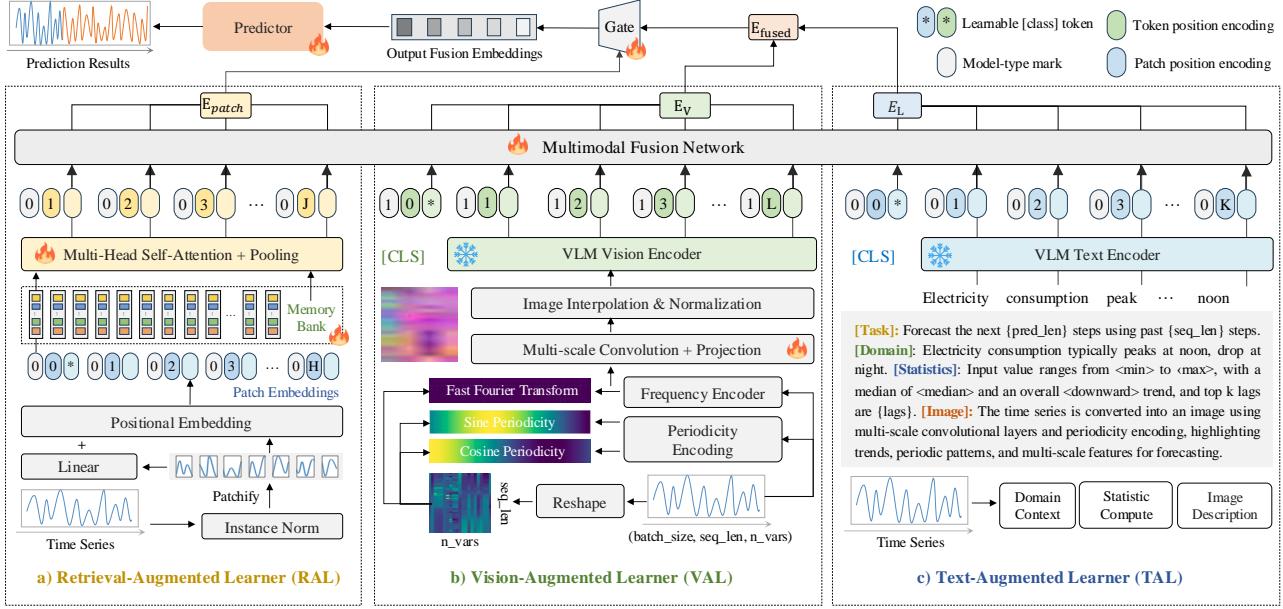


Figure 2: Overview of the Time-VLM framework.

### 3. Methodology

To address the limitations of single-modality approaches and leverage the complementary strengths of visual, textual, and temporal modalities, we propose Time-VLM, a unified framework that integrates these modalities for enhanced time series forecasting. As illustrated in Figure 2, the framework comprises three core components:

- **Retrieval-Augmented Learner (RAL):** Extracts temporal features from raw time series patches and maintains a memory bank to refine patch embeddings through multi-head self-attention and pooling mechanisms, thereby preserving rich temporal representations and enhancing long-term dependency modeling for robust forecasting.
- **Vision-Augmented Learner (VAL):** Transforms time series into informative three-channel images through multi-scale convolutions, frequency and periodic encoding. The images are processed by a frozen VLM vision encoder to extract hierarchical visual features, capturing both fine-grained details and high-level temporal patterns.
- **Text-Augmented Learner (TAL):** Generates contextual textual prompts for input time series, including statistical features (e.g., mean, variance, trends), domain-specific context (e.g., electricity consumption patterns), and image descriptions. These prompts are encoded by a frozen VLM text encoder to produce textual embeddings.

The image and text embeddings, extracted by the VLM, are integrated with temporal memory features via a gated fusion mechanism, effectively capturing complementary information to improve forecasting accuracy. These enriched multimodal features are then processed by a fine-tuned predictor to generate precise and reliable forecasts.

#### 3.1. Retrieval-Augmented Learner (RAL)

The RAL module extracts high-level temporal features via patch-based processing and retrieval-augmented memory mechanisms. It dynamically retrieves historical patterns and integrates them with current observations, adapting to complex time series structures. It operates in two key stages.

**Patch Embedding:** The input time series  $x_{enc} \in \mathbb{R}^{B \times L \times D}$  is divided into overlapping patches of length  $pl$  with stride  $st$ , where  $B$ ,  $L$ , and  $D$  denote the batch size, sequence length, and number of variables, respectively. Each patch is linearly projected into a  $d_{model}$ -dimensional latent space, and positional embeddings are added to preserve temporal order. This yields patch embeddings  $E_p \in \mathbb{R}^{B \times N_p \times d_{model}}$ , where  $N_p = \frac{L-pl}{st} + 1$  represents the total number of patches.

**Retrieval-Augmented Memory:** A memory bank with maximum capacity  $M$  stores historical patch representations  $\mathcal{M} \in \mathbb{R}^{M \times d_{model}}$ . During each forward pass, the current patch embeddings are averaged across the temporal dimension and added to the memory bank using a circular buffer update strategy, ensuring that the most recent patterns are retained. To better capture temporal dynamics, we introduce a hierarchical memory structure:

• **Local Memory:** Given current patch embeddings  $P \in \mathbb{R}^{B \times N_p \times d_{model}}$ , we retrieve top- $k$  similar patches from the memory bank  $\mathcal{M}$  based on cosine similarity:

$$\text{sim}(P, \mathcal{M}) = P \cdot \mathcal{M}^\top, \quad (1)$$

where  $\mathcal{M} \in \mathbb{R}^{M \times d_{model}}$  stores historical patch representations. The retrieved patches are processed through a

two-layer MLP to extract local memory features:

$$M_{\text{local}}^{(i)} = \text{MLP}(\text{topk}(E_p^{(i)})), \quad i = 1, \dots, B. \quad (2)$$

These features are averaged across patch dimension and combined with the original  $P$  via a residual connection.

- **Global Memory:** To capture long-range dependencies, we apply multi-head self-attention over the current patch embeddings  $P$ , yielding contextualized representations:

$$\text{Attn}(P) = \text{MultiHead}(Q, K, V), \quad (3)$$

where  $Q, K, V$  are linear projections of  $P$ . The global memory is obtained by temporal averaging:

$$M_{\text{global}} = \frac{1}{N_p} \sum_{i=1}^{N_p} \text{Attn}(P)_i. \quad (4)$$

The two memories are fused via element-wise addition:

$$M_{\text{fused}} = M_{\text{local}} + M_{\text{global}}, \quad (5)$$

where  $M_{\text{fused}} \in \mathbb{R}^{B \times N_p \times d_{\text{model}}}$  captures high-level temporal patterns. This fused representation supports dynamic retrieval and integration with other modalities (e.g., vision or text), enabling adaptive context-aware forecasting.

### 3.2. Vision-Augmented Learner (VAL)

The VAL module adaptively transforms the input time series  $x_{\text{enc}} \in \mathbb{R}^{B \times L \times D}$  into image representations, enabling fine-grained and high-level temporal pattern extraction via the VLM vision encoder. The process is in three steps:

**Frequency and Periodicity Encoding:** To capture spectral and temporal dependencies, the VAL module applies two complementary encoding techniques to the input time series, explicitly adding frequency and time-domain information.

1. **Frequency Encoding:** A Fast Fourier Transform (FFT) extracts frequency components from raw input  $x_{\text{enc}}$  as:

$$\text{FFT}(x_{\text{enc}}) = \sum_{t=0}^{L-1} x_{\text{enc}}(t) \cdot e^{-2\pi i k t / L}, \quad (6)$$

where  $k$  is the frequency index. The resulting frequency features are concatenated with the input time series, resulting in a tensor of shape  $\mathbb{R}^{B \times L \times D \times 2}$ .

2. **Periodicity Encoding:** Temporal dependencies are encoded using sine and cosine functions for each time step:

$$\text{encoding}(t) = \left[ \sin\left(\frac{2\pi t}{P}\right), \cos\left(\frac{2\pi t}{P}\right) \right], \quad (7)$$

where  $P$  is the periodicity hyperparameter. These encodings are concatenated with the input time series, resulting in a tensor of shape  $\mathbb{R}^{B \times L \times D \times 3}$ . Complete periodic parameter settings can be found in [Appendix A](#).

**Multi-scale Convolution:** The concat tensor is processed through multiple convolutional layers to extract hierarchical temporal patterns. A 1D convolutional layer captures local dependencies, transforming the input into  $\mathbb{R}^{B \times D \times H_{\text{hidden}} \times L}$ , where  $H_{\text{hidden}}$  is the hidden dimension. Averaging along  $D$  yields  $\mathbb{R}^{B \times H_{\text{hidden}} \times L}$ . Two 2D convolutional layers follow: the first halves the channel dimension, and the second maps features to  $C$  output channels, producing the final output capturing both local and global temporal structures.

**Image Interpolation & Normalization:** The output tensor is resized to the desired image dimensions ( $H, W$ ) using bilinear interpolation. For a target pixel  $(x, y)$ , the interpolated value  $\mathbf{I}(x, y)$  is computed as follows:

$$\mathbf{I}(x, y) = \sum_{i=1}^2 \sum_{j=1}^2 \mathbf{I}(x_i, y_j) \cdot w_{ij}, \quad (8)$$

$$\text{Inorm} = 255 \cdot \frac{\mathbf{I}_{\text{raw}} - \text{Min}(\mathbf{I}_{\text{raw}})}{\text{Max}(\mathbf{I}_{\text{raw}}) - \text{Min}(\mathbf{I}_{\text{raw}}) + \epsilon}, \quad (9)$$

where  $(x_i, y_j)$  are the coordinates of the four nearest neighbors,  $w_{ij}$  are weights based on relative distances, and  $\epsilon = 10^{-5}$  prevents division by zero. Pixel values are scaled to  $[0, 255]$  via min-max normalization, producing the normalized image  $\mathbf{I}_{\text{norm}} \in \mathbb{R}^{B \times C \times H \times W}$  ( $C$  is the number of channels). This ensures alignment with the VLM vision encoder's input distribution for effective feature extraction. Example images and descriptions can see [Appendix C](#).

### 3.3. Text-Augmented Learner (TAL)

The TAL module provides contextual textual representations, either pre-defined (e.g., expert annotations) or dynamically generated, offering flexibility across diverse scenarios.

For dynamically generated prompts, TAL extracts key statistical properties from the input time series, including:

- **Statistical Properties:** Value range (min/max), central tendency (median), and overall trend direction.
- **Contextual Information:** Periodic description, task-specific parameters (input window length and forecasting horizon), and domain-specific dataset characteristics.

These features are formatted into structured textual prompts, such as the example shown in Figure 2(c). When domain-specific knowledge is available (e.g., in medical diagnostics or financial analysis), TAL incorporates pre-defined textual descriptions, which are combined with the dynamically generated prompts to enhance contextual understanding.

The final textual inputs are processed by the VLM text encoder, producing contextual embeddings that complement both visual and temporal features. This dual-path design supporting both static and dynamic textual ensures strong generalization across a wide range of applications, from generic forecasting tasks to specialized domain scenarios.

### 3.4. Multimodal Fusion with VLMs

The multimodal fusion pipeline integrates visual (VAL), textual (TAL), and temporal (RAL) information, leveraging their complementary strengths to enhance time series forecasting. It consists of three key steps:

**Multimodal Embeddings Extraction:** The generated images and text are processed by a frozen VLM (e.g., ViLT or CLIP), producing multimodal embeddings of shape  $\mathbb{R}^{B \times L_f \times d_h}$ , where  $B$  is the batch size,  $L_f$  is the sequence length, and  $d_h$  is the VLM’s hidden dimension. These embeddings capture visual and textual context, leveraging the VLM’s pre-trained multimodal understanding capabilities.

**Temporal Feature Fusion:** To address the distribution shift between temporal and multimodal features, both modalities are projected into a shared  $d_{\text{model}}$ -dimensional space. Temporal memory embeddings  $\mathbf{F}_{\text{tem}}$  from RAL encode high-level temporal patterns and serve as queries in a cross-modal multi-head attention (CM-MHA) mechanism, while the multimodal embeddings  $\mathbf{F}_{\text{mm}}$  from the VLM serve as keys and values. The CM-MHA is defined as:

$$\text{CM-MHA}(Q, K, V) = \text{Cat}(\text{head}_1, \dots, \text{head}_h)W^O, \quad (10)$$

$$\text{head}_i = \text{softmax} \left( \frac{QW_i^Q(KW_i^K)^\top}{\sqrt{d_k}} \right) VW_i^V. \quad (11)$$

where  $Q = \mathbf{F}_{\text{tem}}W^Q$ ,  $K = \mathbf{F}_{\text{mm}}W^K$ , and  $V = \mathbf{F}_{\text{mm}}W^V$ . Here,  $W_i^Q$ ,  $W_i^K$ ,  $W_i^V$ , and  $W^O$  are learnable projection matrices.  $d_k = d_{\text{model}}/h$  is the head dimension, and  $h$  is the number of attention heads. This mechanism aligns and integrates temporal and multimodal features, capturing both fine-grained patterns and high-level context. A residual connection and layer normalization stabilize training:

$$\mathbf{F}_{\text{attn}} = \text{LayerNorm}(\mathbf{F}_{\text{tem}} + \text{CM-MHA}(Q, K, V)). \quad (12)$$

A gated fusion mechanism further enhances the output by dynamically weighting each modality:

$$\mathbf{G} = \sigma(\mathbf{W}_g[\mathbf{F}_{\text{tem}}; \mathbf{F}_{\text{mm}}] + \mathbf{b}_g), \quad (13)$$

$$\mathbf{F}_{\text{fused}} = \mathbf{G} \odot \mathbf{F}_{\text{attn}} + (1 - \mathbf{G}) \odot \mathbf{F}_{\text{mm}}, \quad (14)$$

where  $\mathbf{W}_g$  and  $\mathbf{b}_g$  are learnable parameters, and  $\sigma(\cdot)$  is the sigmoid function. This gated mechanism adaptively balances temporal and multimodal features for robust fusion.

**Forecasting:** The fused embedding is processed by a fine-tuned predictor, consisting of fully connected layers, to generate forecasts  $\hat{y} \in \mathbb{R}^{B \times T_{\text{pred}} \times D}$ . By combining visual, textual, and temporal modalities, the pipeline captures both detailed patterns and high-level context, leveraging pre-trained VLMs for enhanced forecasting across diverse scenes.

### 3.5. Optimization

The model is trained end-to-end using mean squared error (MSE). Given historical observations  $\mathbf{X} \in \mathbb{R}^{N \times T}$  (with  $N$  variables and  $T$  time steps), the objective is to predict future values  $\hat{\mathbf{Y}} \in \mathbb{R}^{N \times H}$  over  $H$  steps:

$$\mathcal{L} = \frac{1}{H} \sum_{h=1}^H \|\hat{\mathbf{Y}}_h - \mathbf{Y}_h\|^2, \quad (15)$$

where  $\hat{\mathbf{Y}}_h$  and  $\mathbf{Y}_h$  denote predicted and ground-truth values at step  $h$ . The pre-trained VLM is kept frozen, and only lightweight components are optimized during fine-tuning:

- **RAL:** Patch embedding, memory retrieval, and attention modules for temporal pattern learning;
- **VAL:** Frequency/periodicity encoding and multi-scale CNNs for visual representation generation;
- **Prediction Head:** A gate network and linear projection fuse multimodal features to generate final forecasts.

This strategy adapts the VLM to time series forecasting, achieving robust performance with minimal overhead.

## 4. Experiments

**Datasets and Metrics.** We evaluate Time-VLM on seven widely used time series datasets across diverse domains: energy (ETTh1, ETTh2, ETTm1, ETTm2), weather, electricity (ECL), and traffic (Zhou et al., 2021; Lai et al., 2018). These datasets are commonly used for benchmarking long-term forecasting models (Wu et al., 2023a), and vary in frequency, dimensionality, and temporal characteristics. For short-term forecasting, we use the M4 benchmark (Makridakis et al., 2018), which includes marketing data across multiple frequencies. Performance is measured using Mean Absolute Error (MAE) and Mean Squared Error (MSE), following standard evaluation practices in this field. Additional details are provided in Appendices A.1 and A.3.

**Baselines.** We compare Time-VLM with state-of-the-art time series models, including text-augmented methods like TimeLLM (2024), GPT4TS (2023), and LLMTTime (2023); vision-augmented methods like TimesNet (2023b); traditional deep models like PatchTST (2023), ESTformer (2022), Non-Stationary Transformer (2022b), FEDformer (2022), Autoformer (2021), Informer (2021), and Reformer (2020); and recent competitive models like DLinear (2023), LightTS (2022), N-HiTS (2023), and N-BEATS (2020). Notably, Time-VLM is the first framework combining three modalities for time series forecasting. Performance results for some baselines are cited from (2024a) where applicable.

**Implementation Details.** We compare Time-VLM against superior baselines using a unified evaluation pipeline under the same configurations as (Wu et al., 2023a) to ensure a fair comparison. ViLT (Kim et al., 2021)

Table 1: Few-shot learning on 5% training data. Results are averaged over forecasting horizons  $H \in \{96, 192, 336, 720\}$ . Lower values indicate better performance. Full results see Section B.1. **Red:** best, **Blue:** second best.

Methods	Time-VLM <sub>143M</sub> (Ours)	Time-LLM <sub>3405M</sub> (2024)	GPT4TS (2023)	DLinear (2023)	PatchTST (2023)	TimesNet (2023a)	FEDformer (2022)	Autoformer (2021)	Stationary (2022b)	ETSformer (2022)	LightTS (2022)	Informer (2021)	Reformer (2020)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
<i>ETTh1</i>	<b>0.442</b>	<b>0.453</b>	0.627	0.543	0.681	<b>0.560</b>	0.750	0.611	0.694	0.569	0.925	0.647	<b>0.658</b>	0.562
<i>ETTh2</i>	<b>0.354</b>	<b>0.402</b>	<b>0.382</b>	<b>0.418</b>	0.400	0.433	0.694	0.577	0.827	0.615	0.439	0.448	0.463	0.454
<i>ETTm1</i>	<b>0.364</b>	<b>0.385</b>	0.425	0.434	0.472	0.450	<b>0.400</b>	<b>0.417</b>	0.526	0.476	0.717	0.561	0.730	0.592
<i>ETTm2</i>	<b>0.262</b>	<b>0.323</b>	<b>0.274</b>	<b>0.323</b>	0.308	<b>0.346</b>	0.399	0.426	0.314	0.352	0.344	0.372	0.381	0.404
<i>Weather</i>	<b>0.240</b>	<b>0.280</b>	<b>0.260</b>	0.309	0.263	<b>0.301</b>	0.263	0.308	0.269	0.303	0.298	0.318	0.309	0.353
<i>ECL</i>	0.218	0.315	<b>0.179</b>	<b>0.268</b>	<b>0.178</b>	<b>0.273</b>	0.176	0.275	0.181	0.277	0.402	0.453	0.266	0.353
<i>Traffic</i>	0.558	0.410	<b>0.423</b>	<b>0.298</b>	0.434	0.305	0.450	0.317	<b>0.418</b>	<b>0.296</b>	0.867	0.493	0.676	0.423

Table 2: Few-shot learning on 10% training data. We use the same protocol in Table 1. Full results see Section B.1.

Methods	Time-VLM <sub>143M</sub> (Ours)	Time-LLM <sub>3405M</sub> (2024)	GPT4TS (2023)	DLinear (2023)	PatchTST (2023)	TimesNet (2023a)	FEDformer (2022)	Autoformer (2021)	Stationary (2022b)	ETSformer (2022)	LightTS (2022)	Informer (2021)	Reformer (2020)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
<i>ETTh1</i>	<b>0.431</b>	<b>0.442</b>	<b>0.556</b>	<b>0.522</b>	0.590	0.525	0.691	0.600	0.633	0.542	0.869	0.628	0.639	0.561
<i>ETTh2</i>	<b>0.361</b>	<b>0.405</b>	<b>0.370</b>	<b>0.394</b>	0.397	0.421	0.605	0.538	0.415	0.431	0.479	0.465	0.460	0.475
<i>ETTm1</i>	<b>0.360</b>	<b>0.382</b>	<b>0.404</b>	<b>0.427</b>	0.464	0.441	0.411	0.429	0.501	0.466	0.677	0.537	0.722	0.605
<i>ETTm2</i>	<b>0.263</b>	<b>0.323</b>	<b>0.277</b>	<b>0.323</b>	0.293	<b>0.335</b>	0.316	0.368	0.296	0.343	0.320	0.353	0.463	0.488
<i>Weather</i>	<b>0.233</b>	<b>0.274</b>	<b>0.234</b>	<b>0.273</b>	0.238	0.275	0.241	0.283	0.242	0.279	0.279	0.301	0.284	0.324
<i>ECL</i>	0.198	0.291	<b>0.175</b>	<b>0.270</b>	<b>0.176</b>	<b>0.269</b>	0.180	0.280	0.180	0.273	0.323	0.392	0.346	0.427
<i>Traffic</i>	0.484	0.357	<b>0.429</b>	<b>0.306</b>	0.440	0.310	0.447	0.313	<b>0.430</b>	<b>0.305</b>	0.951	0.535	0.663	0.425

Table 3: Zero-shot learning results. Full results see Section B.2.

Methods	Time-VLM <sub>143M</sub> (Ours)	Time-LLM <sub>3405M</sub> (2024)	LLMTime (2023)	GPT4TS (2023)	DLinear (2023)	PatchTST (2023)		
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
<i>ETTh1</i> → <i>ETTh2</i>	<b>0.338</b>	<b>0.385</b>	<b>0.353</b>	<b>0.387</b>	0.992	0.708	0.400	0.422
<i>ETTh1</i> → <i>ETTm2</i>	<b>0.293</b>	<b>0.350</b>	<b>0.273</b>	<b>0.340</b>	1.867	0.869	0.325	0.363
<i>ETTh2</i> → <i>ETTh1</i>	<b>0.496</b>	<b>0.480</b>	<b>0.479</b>	<b>0.474</b>	1.961	0.981	0.757	0.578
<i>ETTh2</i> → <i>ETTm2</i>	<b>0.297</b>	<b>0.353</b>	<b>0.272</b>	<b>0.341</b>	1.867	0.869	0.335	0.370
<i>ETTm1</i> → <i>ETTh2</i>	<b>0.354</b>	<b>0.397</b>	<b>0.381</b>	<b>0.412</b>	0.992	0.708	0.433	0.439
<i>ETTm1</i> → <i>ETTm2</i>	<b>0.264</b>	<b>0.319</b>	<b>0.268</b>	<b>0.320</b>	1.867	0.869	0.313	0.348
<i>ETTm2</i> → <i>ETTh2</i>	<b>0.359</b>	<b>0.399</b>	<b>0.354</b>	<b>0.400</b>	0.992	0.708	0.435	0.443
<i>ETTm2</i> → <i>ETTm1</i>	<b>0.432</b>	<b>0.426</b>	<b>0.414</b>	<b>0.438</b>	1.933	0.984	0.769	0.567

("vilt-b32-finetuned-coco") serves as the default vision-language backbone; CLIP and BLIP-2 are also supported. All models are trained with Adam ( $10^{-3}$  initial learning rate, halved per epoch), batch size 32, for up to 10 epochs with early stopping. Experiments run on Nvidia RTX A6000 GPU (48GB). More details are in Appendix A.2.

#### 4.1. Few-shot Forecasting

**Setting.** We evaluate the few-shot long-term forecasting capabilities of Time-VLM by testing its performance using only 5% or 10% of the training data. This setting assesses how effectively Time-VLM integrates pre-trained multimodal knowledge from the VLM with time series-specific features under minimal task-specific supervision.

**Results.** As shown in Table 1 and Table 2, Time-VLM consistently outperforms most baselines across datasets. For example, on ETTh1 with 5% training data, Time-VLM reduces MSE by 29.5% and MAE by 16.6% compared to the second-best model, TimeLLM. On ETTm1 with 10% data, it surpasses TimeLLM by 11.1% in MSE and 10.5% in

MAE. On Weather with 5% data, Time-VLM outperforms TimeLLM by 7.7% in MSE and 9.4% in MAE. The performance gap between Time-VLM and traditional models (e.g., PatchTST, FEDformer) is particularly pronounced in few-shot settings, demonstrating the effectiveness of multimodal integration when data is scarce. This performance gain stems from the model's ability to leverage rich multimodal priors from pre-trained VLMs, while effectively capturing temporal patterns through memory-enhanced attention.

#### 4.2. Zero-shot Forecasting

**Setting.** We evaluate the zero-shot forecasting capability of Time-VLM in cross-domain settings, where the model predicts on unseen datasets by effectively transferring knowledge from unrelated domains. To ensure a rigorous comparison, we conduct experiments using the ETT datasets as source and target domains, following previous setup (Jin et al., 2024). Results are summarized in Table 3.

**Results.** Time-VLM demonstrates strong generalizability, consistently outperforming or matching state-of-the-art baselines while using fewer parameters. For example, in the ETTh1 → ETTh2 transfer setting, Time-VLM achieves a 4.2% lower MSE and 0.5% lower MAE than TimeLLM. In ETTm1 → ETTh2, it outperforms TimeLLM by 7.1% in MSE and 3.6% in MAE. In ETTm2 → ETTh2, Time-VLM performs competitively, closely matching TimeLLM with only a 1.4% difference in MSE and 0.3% in MAE. These results highlight Time-VLM's ability to generalize across domains without fine-tuning, leveraging pre-trained vision-language priors for effective knowledge transfer.

Table 4: Short-term time series forecasting results on M4. The forecasting horizons are in [6, 48] and the three rows provided are weighted averaged from all datasets under different sampling intervals. Full results see Section B.3.

Methods	Time-VLM <sub>143M</sub> (Ours)	Time-LLM <sub>3405M</sub> (2024)	GPT4TS (2023)	TimesNet (2023a)	PatchTST (2023)	N-HiTS (2023)	N-BEATS (2020)	ETSformer (2022)	LightTS (2022)	DLinear (2023)	FEDformer (2022)	Stationary (2022b)	Autoformer (2021)	Informer (2021)	Reformer (2020)
SMAPE	<b>11.894</b>	<u>11.983</u>	12.690	12.880	12.059	12.035	12.250	14.718	13.525	13.639	13.160	12.780	12.909	14.086	18.200
MASE	<b>1.592</b>	<u>1.595</u>	1.808	1.836	1.623	1.625	1.698	2.408	2.111	2.095	1.775	1.756	1.771	2.718	4.223
OWA	<b>0.855</b>	<u>0.859</u>	0.940	0.955	0.869	0.869	0.896	1.172	1.051	1.051	0.949	0.930	0.939	1.230	1.775

Table 5: Long-term forecasting results. We use the same protocol in Table 1. Full results see in Section B.4.

Methods	Time-VLM <sub>143M</sub> (Ours)	Time-LLM <sub>3405M</sub> (2024)	GPT4TS (2023)	DLinear (2023)	PatchTST (2023)	TimesNet (2023a)	FEDformer (2022)	Autoformer (2021)	Stationary (2022b)	ETSformer (2022)	LightTS (2022)	Informer (2021)	Reformer (2020)													
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE											
ETTh1	<b>0.405</b>	<b>0.420</b>	<u>0.408</u>	<u>0.423</u>	0.465	0.455	0.422	0.437	0.413	0.430	0.458	0.450	0.440	0.460	0.496	0.487	0.570	0.537	0.542	0.510	0.491	0.479	1.040	0.795	1.029	0.805
ETTh2	0.341	0.391	<u>0.334</u>	<u>0.383</u>	0.381	0.412	0.431	0.446	<u>0.330</u>	<u>0.379</u>	0.414	0.427	0.437	0.449	0.450	0.459	0.526	0.516	0.439	0.452	0.602	0.543	4.431	1.729	6.736	2.191
ETTm1	<u>0.347</u>	<u>0.377</u>	<b>0.329</b>	<b>0.372</b>	0.388	0.403	0.357	0.378	0.351	0.380	0.400	0.406	0.448	0.452	0.588	0.517	0.481	0.456	0.429	0.425	0.435	0.437	0.961	0.734	0.799	0.671
ETTm2	<b>0.248</b>	<b>0.311</b>	<u>0.251</u>	<u>0.313</u>	0.284	0.339	0.267	0.333	0.255	0.315	0.291	0.333	0.305	0.349	0.327	0.371	0.306	0.347	0.293	0.342	0.409	0.436	1.410	0.810	1.479	0.915
Weather	<b>0.224</b>	<b>0.263</b>	<u>0.225</u>	<u>0.257</u>	0.237	0.270	0.248	0.300	0.225	0.264	0.259	0.287	0.309	0.360	0.338	0.382	0.288	0.314	0.271	0.334	0.261	0.312	0.634	0.548	0.803	0.656
Electricity	0.172	0.273	<b>0.158</b>	<b>0.252</b>	0.167	<u>0.263</u>	0.166	<u>0.263</u>	<u>0.161</u>	<u>0.252</u>	0.192	0.295	0.214	0.327	0.227	0.338	0.193	0.296	0.208	0.323	0.229	0.329	0.311	0.397	0.338	0.422
Traffic	0.419	0.303	<b>0.388</b>	<b>0.264</b>	0.414	0.294	0.433	0.295	<u>0.390</u>	<u>0.263</u>	0.620	0.336	0.610	0.376	0.628	0.379	0.624	0.340	0.621	0.396	0.622	0.392	0.764	0.416	0.741	0.422

### 4.3. Short-term Forecasting

**Setting.** For short-term forecasting, we evaluate Time-VLM on the M4 benchmark, which includes marketing data at various sampling frequencies. Performance is measured using SMAPE, MASE, and OWA metrics, averaged across datasets and sampling intervals (see Table 4).

**Results.** Time-VLM demonstrates strong performance, consistently outperforming state-of-the-art baselines across all metrics. For instance, it surpasses the second-best model, Time-LLM, with improvements of 0.7% in SMAPE, 0.2% in MASE, and 0.5% in OWA, all while utilizing significantly fewer parameters and computational resources. Compared to traditional models like PatchTST and N-HiTS, the performance gains more, highlighting the benefit of multimodal knowledge in short-term forecasting. These gains stem from Time-VLM’s integration of temporal, visual, and textual data, capturing richer features for improved accuracy.

### 4.4. Long-term Forecasting

**Setting.** We evaluate the long-term forecasting capabilities of Time-VLM across multiple horizons and datasets.

**Results.** As shown in Table 5, Time-VLM achieves competitive performance compared to state-of-the-art baselines. On ETTh1, Time-VLM improves upon TimeLLM by 0.7% in both MSE and MAE. On ETTm2, it outperforms TimeLLM by 1.2% in MSE and 0.6% in MAE. However, on the Weather dataset, Time-VLM slightly underperforms TimeLLM, with a 0.4% higher MSE and 2.3% higher MAE.

Overall, Time-VLM demonstrates robust performance across diverse tasks and datasets, highlighting its generalization and efficiency. By leveraging multimodal knowledge, it consistently outperforms state-of-the-art baselines with significantly fewer parameters (143M vs. TimeLLM’s 3405M), making it a practical solution for real-world applications.

### 4.5. Model Analysis

**Ablation Studies.** Table 6 evaluates the contributions of key components of Time-VLM, including the RAL (with Local and Global Memory), VAL, and TAL. The study highlights the importance of each module. Removing the RAL causes a significant performance drop (35.6% in MSE), with its local (RAL\_L) and global (RAL\_G) branches contributing 17.2% and 4.3%, respectively, validating our hierarchical memory design. The VAL is also essential—removing it increases MSE by 9.0%, demonstrating its ability to preserve fine-grained temporal patterns via the VLM vision encoder. In contrast, removing the TAL results in only minor degradation (2.1% in MSE), likely due to the sparsity of textual tokens in the VLM output; for example, ViLT produces just 11 textual tokens out of 156 total, the rest being visual embeddings. While the TAL provides useful semantic context, its impact is limited by the scarcity of textual signals. Future work may explore VLMs with stronger language capabilities for better temporal-semantic alignment.

Table 6: Ablation study on multimodal components over forecasting horizons  $H \in \{96, 192, 336, 720\}$  on Weather dataset, with MSE performance degradation (%Deg) measured for each variant.

Horizon	Full	w/o RAL	w/o RAL_L	w/o RAL_G	w/o VAL	w/o TAL
96	<b>0.160</b>	0.273	0.185	0.165	0.213	<b>0.165</b>
192	<b>0.203</b>	0.297	0.235	0.210	0.237	<b>0.208</b>
336	<b>0.253</b>	0.325	0.295	0.265	0.255	<b>0.258</b>
720	<b>0.317</b>	0.369	0.375	0.330	0.309	<b>0.322</b>
Avg	<b>0.233</b>	0.316	0.273	0.243	0.254	<b>0.238</b>
%Deg	–	35.6% ↑	17.2% ↑	4.3% ↑	9.0% ↑	2.1% ↑

**Multimodal and Few-/Zero-shot Analysis.** To understand the source of Time-VLM’s strong performance in data-scarce scenarios, we examine the relationship between RAL (temporal) and TAL/VAL (multimodal) embeddings. As shown in Figure 3, their complementary nature is evident. The left panel illustrates balanced gate weight distributions, indicating effective fusion of temporal and multimodal sig-

nals. The right panel presents a UMAP visualization revealing distinct yet partially overlapping clusters, confirming successful integration of multimodal information while preserving modality-specific characteristics. The robust few-shot and zero-shot capabilities of Time-VLM stem from its integration of temporal, visual, and textual modalities. Specifically, the RAL models temporal dependencies via memory bank interactions, enabling robust feature extraction even with limited data. The VAL captures interpretable visual patterns—such as trends, seasonality, and periodicity—in domain-agnostic representations, while the TAL generates semantic descriptions that enhance generalization. Together, these components allow Time-VLM to leverage pre-trained multimodal knowledge, making it highly adaptable to new tasks and domains with minimal fine-tuning.

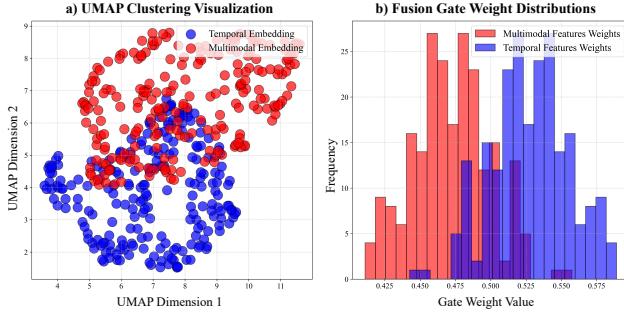


Figure 3: UMAP visualization (left) and gate weight distributions (right) of multimodal and temporal embeddings.

**Interpretability Analysis.** We investigate how pre-trained VLMs can be leveraged for time series forecasting by analyzing the alignment between visual, textual, and temporal representations. To this end, we sample 400 image-text pairs from MSCOCO, the primary pre-training dataset for VLMs, and 200 samples each from time series datasets: ETT, Traffic, Weather, and ECL. Using UMAP, we visualize four types of embeddings in a shared 2D space:

- multimodal embeddings derived from COCO-Pair samples using VLM, reflecting the model’s general pretrained cross-modal knowledge, which is Time-VLM’s main motivation of the time series project to.
- multimodal embeddings from time series-generated image-text pairs processed through the same VLM, representing Time-VLM’s task-specific augmentation.
- Visual-only embeddings from COCO-Image samples extracted via ViT, reflecting pure visual knowledge.
- Text-only embeddings from COCO-Text samples encoded with BERT, capturing linguistic knowledge.

As shown in Figure 4, both COCO-Image and COCO-Text form distinct, separate clusters, remaining isolated from time-series features. This suggests that while low-level visual patterns in COCO-Image resemble temporal dynamics,

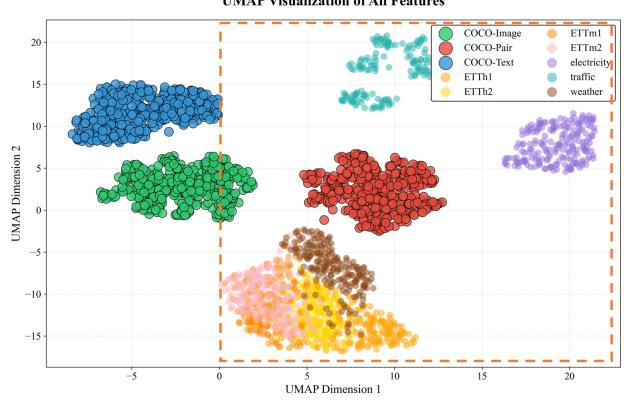


Figure 4: Interpretability visualization of Time-VLM: multimodal feature alignment via UMAP.

pure image representations retain modality-specific characteristics. Similarly, COCO-Text forms a completely separate cluster, highlighting significant modality gaps. In contrast, COCO-Pair exhibits maximal overlap with time-series data, demonstrating strong cross-modal complementarity. The textual semantics in COCO-Pair bridge visual and temporal modalities, enhancing their alignment. Notably, COCO-Pair is positioned near the center of time-series clusters, suggesting its role as a key mediator between modalities.

These observations motivate Time-VLM’s design: instead of relying on single-modality projections, we embed time series into multimodal space for richer semantic understanding. The model’s strong performance in data-scarce settings stems from pre-trained VLM knowledge. With more time series data, better alignment of multimodal embeddings is expected, further enhancing performance.

**Computation Studies.** Time-VLM demonstrates strong computational efficiency, as shown in Table 7. With only 143.6M parameters (1/20 of Time-LLM’s 3404.6M), memory usage scales from 1968 MiB (Weather) to 24916 MiB (Traffic), adapting to dataset complexity. Inference speed ranges from 0.2057s/iter (ECL) to 0.4809s/iter (ETTh1), efficiently handling varying loads. In contrast, Time-LLM requires over 37GB of memory even for smaller datasets like ETTh1 and ETTh2, making it infeasible for larger datasets such as Weather, ECL, and Traffic. This highlights Time-VLM’s lightweight design and practical scalability.

Table 7: Computational efficiency comparison between Time-VLM and Time-LLM across datasets. “-” denotes memory exceeds 49GB, infeasible on a single GPU. Results are averaged over multiple prediction steps under consistent conditions.

Method	Metric	ETTh1	ETTh2	ETTm1	ETTm2	Weather	ECL	Traffic
Time-VLM	Param. (M)	143.6	143.6	143.6	143.6	143.6	143.6	143.6
	Mem. (MiB)	2630	2630	2640	2640	1968	10818	24916
	Speed (s/iter)	0.481	0.438	0.277	0.210	0.296	0.206	0.323
Time-LLM	Param. (M)	3404.6	3404.6	3404.6	3404.6	-	-	-
	Mem. (MiB)	37723	37723	37849	37849	-	-	-
	Speed (s/iter)	0.607	0.553	0.349	0.265	-	-	-

**Hyperparameter Studies.** We analyze the impact of key hyperparameters on performance, as shown in Figure 5. Sequence length performs best between 96 and 1024 timesteps, with 512 being optimal for most datasets. Longer input introduce noise without significant gains, indicating that local temporal patterns are sufficient for accurate forecasting. The normalization constant peaks at 0.4, reflecting a balance between feature scaling and training stability. Model dimension shows dataset-dependent behavior: values of 128–256 suffice for short-term datasets like ETTh1 and ETTh2, while longer horizons and higher-dimensional data (e.g., Traffic, Weather) benefit from larger dimensions (up to 512), suggesting greater capacity is needed to model complex dynamics and variable interactions. Similarly, the gate network dimension—responsible for multimodal fusion—achieves optimal performance at 256 for medium-range forecasts. For more challenging settings like long-horizon or high-variable inputs, increasing it to 336 or 512 further improves results, highlighting the importance of adaptive fusion in capturing complex cross-modal relationships.

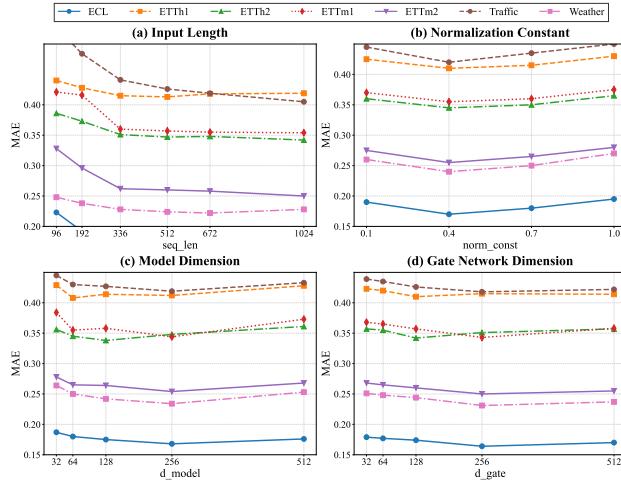


Figure 5: Hyperparameters sensitivity analysis on input length, normalization constant, dimension of model and dimension of gate network, reflected by MAE.

**VLM Variants Analysis.** We conduct ablation studies on different VLM backbones and custom combinations to assess their impact on forecasting performance and computational efficiency. We evaluate three widely-used VLMs—ViLT, CLIP, and BLIP-2—with varying size, and find that increased model size does not improve accuracy. Notably, although BLIP-2 is the largest (3.7B parameters, 25GB+ memory), it underperforms ViLT and CLIP in terms of MSE (0.342 vs. 0.337) and exhibits slow training speed (0.98 s/iter), limiting its practical use. In contrast, lightweight models like ViLT (128.9M parameters, 1346 MiB memory) and CLIP (168.4M, 1174 MiB) achieve comparable or better accuracy at a fraction of the cost. To

evaluate the benefit of VLMs’ pre-trained cross-modal alignment, we construct a modular baseline using separately trained vision and language encoders: ViT-B/16 and BERT-Base. As shown in Table 8, this custom combination underperforms all pretrained VLMs across metrics, achieving an average MSE of 0.348 versus 0.336 for ViLT, with no compensating gain in speed (0.17 s/iter). These findings highlight that the cross-modal alignment in VLMs offers a key inductive bias for time series modeling. Modular approaches, lacking such a unified space, fail to match this performance—demonstrating the value of pre-aligned multimodal representations for efficient fusion and forecasting.

Table 8: Comparison of different VLM variants on ETTh2 in terms of performance and computational efficiency.

VLM Type	Params (M)	Mem. (MiB)	Speed (s/iter)	MSE (avg)	MAE (avg)
ViLT	128.9	1346	0.36	0.336	0.388
CLIP	168.4	1174	0.12	0.339	0.391
BLIP-2	3763.1	25200	0.98	0.342	0.393
Custom	213.2	1474	0.17	0.348	0.397

## 5. Conclusion

We presented Time-VLM, a novel framework that leverages pretrained VLMs to unify temporal, visual, and textual modalities for time series forecasting. By integrating the RAL, VAL, and TAL modules, Time-VLM bridges modality gaps and enables rich cross-modal interactions. Notably, it operates solely on raw time series data without requiring external information, enabling fair comparisons and demonstrating the ability to generate textual and visual representations internally for self-augmentation. This design not only improves accuracy but also highlights the framework’s robustness—particularly in domains where auxiliary data is scarce. Extensive experiments show that Time-VLM achieves superior performance across diverse datasets, especially in few-shot and zero-shot settings, outperforming existing methods while maintaining computational efficiency. Our work establishes a new direction in multimodal time series forecasting by highlighting the potential of VLMs in capturing both temporal dynamics and semantic context.

**Limitations.** Despite its strengths, Time-VLM has several limitations. First, the TAL module provides semantic context but has limited impact due to current VLMs’ constrained understanding of time series semantics. Second, while excelling in low-data regimes, full-shot performance slightly lags behind specialized unimodal models on certain tasks (e.g., ECL, Traffic), suggesting room for domain-specific adaptation. Third, although computationally efficient compared to LLM-based methods, deployment on resource-constrained devices remains challenging. These limitations suggest promising directions for future work, including temporally aware VLMs, improved time series imaging, visual distillation, and enhanced text encoders with stronger temporal reasoning. For details, see Appendix D.

## Acknowledgements

This work is mainly supported by the Guangdong Basic and Applied Basic Research Foundation (No. 2025A1515011994). This work is also supported by the National Natural Science Foundation of China (No. 62402414, No. 62402420), Guangzhou Municipal Science and Technology Project (No. 2023A03J0011), the Guangzhou Industrial Information and Intelligent Key Laboratory Project (No. 2024A03J0628), and a grant from State Key Laboratory of Resources and Environmental Information System, and Guangdong Provincial Key Lab of Integrated Communication, Sensing and Computation for Ubiquitous Internet of Things (No. 2023B1212010007).

## Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning by integrating temporal, visual, and textual modalities for time series forecasting. While our approach improves accuracy and cross-domain generalization, we acknowledge potential risks such as data privacy concerns, algorithmic bias, and increased computational costs. We encourage further research into mitigating these risks to ensure responsible deployment in high-stakes applications.

## References

- Challu, C., Olivares, K. G., Oreshkin, B. N., Ramirez, F. G., Canseco, M. M., and Dubrawski, A. Nhits: neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 6989–6997, 2023.
- Chang, C., Peng, W.-C., and Chen, T.-F. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.
- Chen, M., Shen, L., Li, Z., Wang, X. J., Sun, J., and Liu, C. Visions: Visual masked autoencoders are free-lunch zero-shot time series forecasters, 2024. URL <https://arxiv.org/abs/2408.17253>.
- Deb, C., Zhang, F., Yang, J., Lee, S. E., and Shah, K. W. A review on time series forecasting techniques for building energy consumption. *Renewable and Sustainable Energy Reviews*, 74:902–924, 2017.
- Gruver, N., Finzi, M., Qiu, S., and Wilson, A. G. Large language models are zero-shot time series forecasters. In *Advances in Neural Information Processing Systems*, 2023.
- Idrees, S. M., Alam, M. A., and Agarwal, P. A prediction approach for stock market volatility based on time series data. *IEEE Access*, 7:17287–17298, 2019.
- Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q., Sung, Y.-H., Li, Z., and Duerig, T. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pp. 4904–4916. PMLR, 2021.
- Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J. Y., Shi, X., Chen, P.-Y., Liang, Y., Li, Y.-F., Pan, S., and Wen, Q. Time-LLM: Time series forecasting by reprogramming large language models. In *International Conference on Learning Representations (ICLR)*, 2024.
- Karevan, Z. and Suykens, J. A. Transductive lstm for time-series prediction: An application to weather forecasting. *Neural Networks*, 125:1–9, 2020.
- Kim, W., Son, B., and Kim, I. Vilt: Vision-and-language transformer without convolution or region supervision. In *International conference on machine learning*, pp. 5583–5594. PMLR, 2021.
- Kitaev, N., Kaiser, Ł., and Levskaya, A. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2020.
- Lai, G., Chang, W.-C., Yang, Y., and Liu, H. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.
- Li, J., Li, D., Savarese, S., et al. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.
- Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Advances in neural information processing systems*, 2019.
- Li, X., Kang, Y., and Li, F. Forecasting with time series imaging. *Expert Systems with Applications*, 160:113680, 2020.
- Liang, Y., Wen, H., Nie, Y., Jiang, Y., Jin, M., Song, D., Pan, S., and Wen, Q. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 6555–6565, 2024.
- Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.
- Liu, Q., Liu, X., Liu, C., Wen, Q., and Liang, Y. Time-ffm: Towards lm-empowered federated foundation model for time series forecasting. *arXiv preprint arXiv:2405.14252*, 2024a.

- Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., and Dusdar, S. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations*, 2022a.
- Liu, X., Hu, J., Li, Y., Diao, S., Liang, Y., Hooi, B., and Zimmermann, R. Unitime: A language-empowered unified model for cross-domain time series forecasting. In *Proceedings of the ACM on Web Conference 2024*, pp. 4095–4106, 2024b.
- Liu, Y., Wu, H., Wang, J., and Long, M. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, 35:9881–9893, 2022b.
- Makridakis, S., Spiliotis, E., and Assimakopoulos, V. The m4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, 34(4): 802–808, 2018.
- Medsker, L. R., Jain, L., et al. Recurrent neural networks. *Design and Applications*, 5(64-67):2, 2001.
- Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.
- Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. N-beats: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*, 2020.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.
- Sood, S., Zeng, Z., Cohen, N., Balch, T., and Veloso, M. Visual time series forecasting: an image-driven approach. In *Proceedings of the Second ACM International Conference on AI in Finance*, pp. 1–9, 2021.
- Wang, S., Li, J., Shi, X., Ye, Z., Mo, B., Lin, W., Ju, S., Chu, Z., and Jin, M. Timemixer++: A general time series pattern machine for universal predictive analysis. *arXiv preprint arXiv:2410.16032*, 2024.
- Woo, G., Liu, C., Sahoo, D., Kumar, A., and Hoi, S. Ets-former: Exponential smoothing transformers for time-series forecasting. In *arXiv preprint arXiv:2202.01381*, 2022.
- Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.
- Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., and Long, M. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023a.
- Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., and Long, M. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023b.
- Zeng, A., Chen, M., Zhang, L., and Xu, Q. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 2023.
- Zhang, T., Zhang, Y., Cao, W., Bian, J., Yi, X., Zheng, S., and Li, J. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint arXiv:2207.01186*, 2022.
- Zheng, J. and Huang, M. Traffic flow forecast through time series analysis based on deep learning. *IEEE Access*, 8: 82562–82570, 2020.
- Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on Artificial Intelligence*, pp. 11106–11115, 2021.
- Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., and Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, pp. 27268–27286, 2022.
- Zhou, T., Niu, P., Wang, X., Sun, L., and Jin, R. One fits all: Power general time series analysis by pretrained lm. In *Advances in Neural Information Processing Systems*, 2023.

## A. Experimental Details

### A.1. Dataset Details

Table 9: Summary of benchmark datasets. Each dataset includes multiple time series (Dim.) with varying sequence lengths, split into training, validation, and testing sets. Data are collected at different frequencies across various domains.

Tasks	Dataset	Dim.	Series Length	Dataset Size	Frequency	Domain	Periodicity
Long-term Forecasting	ETTm1	7	{96, 192, 336, 720}	(34465, 11521, 11521)	15 min	Temperature	96
	ETTm2	7	{96, 192, 336, 720}	(34465, 11521, 11521)	15 min	Temperature	96
	ETTh1	7	{96, 192, 336, 720}	(8545, 2881, 2881)	1 hour	Temperature	24
	ETTh2	7	{96, 192, 336, 720}	(8545, 2881, 2881)	1 hour	Temperature	24
	Electricity	321	{96, 192, 336, 720}	(18317, 2633, 5261)	1 hour	Electricity	24
	Traffic	862	{96, 192, 336, 720}	(12185, 1757, 3509)	1 hour	Transportation	24
Short-term Forecasting	Weather	21	{96, 192, 336, 720}	(36792, 5271, 10540)	10 min	Weather	144
	M4 - Yearly	1	6	(23000, 0, 23000)	Yearly	Demographic	1
	M4 - Quarterly	1	8	(24000, 0, 24000)	Quarterly	Finance	4
	M4 - Monthly	1	18	(48000, 0, 48000)	Monthly	Industry	3
	M4 - Weekly	1	13	(359, 0, 359)	Weekly	Macro	4
	M4 - Daily	1	14	(4227, 0, 4227)	Daily	Micro	1
	M4 - Hourly	1	48	(414, 0, 414)	Hourly	Other	24

The benchmark datasets used in our experiments are summarized in Table 9. These datasets span diverse domains, including temperature monitoring (*ETTm1*, *ETTm2*, *ETTh1*, *ETTh2*), electricity consumption (*Electricity*), transportation (*Traffic*), and weather forecasting (*Weather*). Each dataset contains multiple time series with varying sequence lengths, split into training, validation, and testing sets. The datasets are collected at different frequencies, ranging from 15 minutes to yearly intervals, and exhibit distinct periodic patterns. For short-term forecasting, we utilize the M4 benchmark, which includes datasets with yearly, quarterly, monthly, weekly, daily, and hourly frequencies, covering domains such as finance, industry, and demographics. This diverse collection of datasets ensures a comprehensive evaluation of our method.

#### A.1.1. DATASET DESCRIPTION

The datasets used in our experiments are described below:

- **ECL:** Measurements of electric power consumption in one household with a one-minute sampling rate over 4 years. It includes various electrical quantities and sub-metering values, totaling 2,075,259 measurements from a house in Sceaux, France (December 2006 to November 2010).
- **ETT:** The Electricity Transformer Temperature (ETT) dataset, crucial for electric power deployment, contains 2 years of data from two counties in China. Subsets *ETTh1* and *ETTh2* provide 1-hour-level data, while *ETTm1* offers 15-minute-level data. Each point includes the target "oil temperature" and 6 power load features, with a 12/4/4 month train/val/test split.
- **Traffic:** Hourly data from the California Department of Transportation, describing road occupancy rates measured by sensors on San Francisco Bay area freeways.
- **Weather:** Recorded every 10 minutes throughout 2020, this dataset includes 21 meteorological indicators, such as air temperature and humidity.
- **M4:** A collection of 100,000 time series from the Makridakis Forecasting Competition, including yearly, quarterly, monthly, weekly, daily, and hourly data. The training sets have minimum observations of 13 (yearly), 16 (quarterly), 42 (monthly), 80 (weekly), 93 (daily), and 700 (hourly). Forecasts required are 6 (yearly), 8 (quarterly), 18 (monthly), 13 (weekly), 14 (daily), and 48 (hourly).

#### A.1.2. PERIODICITY PARAMETER

The *Periodicity* column in Table 9 specifies the periodicity hyperparameter  $P$  used in the periodicity encoding process. This parameter is derived from the inherent characteristics of each dataset and reflects the dominant temporal patterns, such as

daily, weekly, or seasonal cycles. For example, in the *ETTm1* and *ETTm2* datasets, which are sampled every 15 minutes, the periodicity  $P = 96$  corresponds to a daily cycle ( $24 \text{ hours} \times 4 \text{ samples per hour}$ ). Similarly, for the *ETTh1* and *ETTh2* datasets, sampled hourly,  $P = 24$  represents a daily cycle. The *Weather* dataset, sampled every 10 minutes, has  $P = 144$ , reflecting a daily cycle ( $24 \text{ hours} \times 6 \text{ samples per hour}$ ). For the M4 benchmark datasets, the periodicity values are set based on their sampling frequencies:  $P = 1$  for yearly data,  $P = 4$  for quarterly and weekly data,  $P = 3$  for monthly data, and  $P = 24$  for hourly data. These values are used in the periodicity encoding formula:

$$\text{encoding}(t) = \left[ \sin\left(\frac{2\pi t}{P}\right), \cos\left(\frac{2\pi t}{P}\right) \right], \quad (16)$$

where  $t$  is the time step and  $P$  is the periodicity hyperparameter. The resulting encodings are concatenated with the input time series, enriching the model's ability to capture temporal dependencies and periodic patterns.

## A.2. Optimization Settings

### A.2.1. MODEL ARCHITECTURE PARAMETERS

Time-VLM consists of several key components, each with specific parameter configurations. Image representations are set to a size of  $64 \times 64$ , balancing computational efficiency and temporal information preservation. The model backbone utilizes a hidden dimension of  $d_{\text{model}} = 128$ , while the encoder-decoder structure comprises  $e_{\text{layers}} = 2$  encoder layers and  $d_{\text{layers}} = 1$  decoder layer. A dropout rate of 0.1 is applied to mitigate overfitting during training. For efficient data loading, the model employs  $\text{num\_workers} = 32$  to parallelize data preprocessing tasks.

The gated fusion module is designed with a dimension of  $d_{\text{fusion}} = 256$ , facilitating the effective integration of multimodal features. The VLM component generates multimodal embeddings with a token length of  $vlm_{\text{fused\_len}} = 156$  and a hidden dimension of  $vlm_{\text{hidden\_dim}} = 768$ , ensuring seamless compatibility with the pre-trained VLM's architecture.

Table 10: Default Model Architecture Parameters

Parameter	Default Value	Description
image_size	64	Size of generated image representation
d_model	128	Dimension of hidden embeddings
d_fusion	256	Dimension of gated fusion module
num_workers	32	Number of data loader workers
e_layers	2	Number of encoder layers
d_layers	1	Number of decoder layers
dropout	0.1	Dropout rate
vlm_fused_len	156	Token length of VLM multimodal embedding
vlm_hidden_dim	768	Hidden dimension of VLM

### A.2.2. TRAINING PARAMETERS

We adopt a comprehensive training strategy with both general and task-specific parameters. The model is trained with a batch size of 32 and an initial learning rate of 0.001, using the *AdamW* optimizer. Early stopping with a patience of 3 epochs is implemented to prevent overfitting. The training process employs Mean Squared Error (MSE) as the primary loss function and runs for a maximum of 10 epochs. For time series processing, we use an input sequence length of 512 and prediction lengths of 96, 192, 336, or 720, depending on the task. The output dimension ( $c_{\text{out}}$ ) varies by dataset: 7 for ETTh1/h2/m1/m2, 21 for Weather, 321 for Electricity, and 862 for Traffic. The periodicity parameter is set to 24 for ETTh1/h2, Electricity, and Traffic; 96 for ETTm1/m2; and 144 for Weather, ensuring alignment with dataset-specific temporal patterns. A normalization coefficient of 0.4 is applied to stabilize training dynamics. The patch embedding module uses a patch length of 16, a stride of 8, and padding of 8 to process the input sequences. The temporal memory mechanism employs 8 learnable queries and 4 attention heads to capture high-level dependencies. Additionally, the training process leverages automatic mixed precision (AMP) to accelerate training while maintaining numerical stability.

Table 11: Default Training Parameters

Parameter	Default Value	Description
batch_size	32	Training batch size
learning_rate	0.001	Initial learning rate
training_epochs	10	Number of training epochs
patience	3	Early stopping patience
loss	MSE	Mean square error
seq_len	512	Input sequence length
c_out	7 (ETTh1/h2/m1/m2) 21 (Weather) 321 (Electricity) 862 (Traffic)	Output dimension (dataset-specific)
pred_len	96/192/336/720	Prediction length
periodicity	24 (ETTh1/h2/Electricity/Traffic) 96 (ETTm1/m2) 144 (Weather)	Dataset periodicity (dataset-specific)
norm_const	0.4	Normalization coefficient
patch_len	16	Patch length
padding	8	Padding length
stride	8	Stride length
num_queries	8	Number of learnable queries for temporal memory
n_heads	4	Number of attention heads

### A.3. Evaluation Metrics

For evaluation, we utilize mean squared error (MSE) and mean absolute error (MAE) for long-term forecasting. For short-term forecasting on the M4 benchmark, we adopt symmetric mean absolute percentage error (SMAPE), mean absolute scaled error (MASE), and overall weighted average (OWA), following the evaluation protocol of N-BEATS (Oreshkin et al., 2020). OWA is a specific metric used in the M4 competition. The metrics are calculated as follows:

$$\begin{aligned}
 \text{MSE} &= \frac{1}{H} \sum_{h=1}^T (\mathbf{Y}_h - \hat{\mathbf{Y}}_h)^2, & \text{MAE} &= \frac{1}{H} \sum_{h=1}^H |\mathbf{Y}_h - \hat{\mathbf{Y}}_h|, \\
 \text{SMAPE} &= \frac{200}{H} \sum_{h=1}^H \frac{|\mathbf{Y}_h - \hat{\mathbf{Y}}_h|}{|\mathbf{Y}_h| + |\hat{\mathbf{Y}}_h|}, & \text{MAPE} &= \frac{100}{H} \sum_{h=1}^H \frac{|\mathbf{Y}_h - \hat{\mathbf{Y}}_h|}{|\mathbf{Y}_h|}, \\
 \text{MASE} &= \frac{1}{H} \sum_{h=1}^H \frac{|\mathbf{Y}_h - \hat{\mathbf{Y}}_h|}{\frac{1}{H-s} \sum_{j=s+1}^H |\mathbf{Y}_j - \mathbf{Y}_{j-s}|}, & \text{OWA} &= \frac{1}{2} \left[ \frac{\text{SMAPE}}{\text{SMAPE}_{\text{Naïve2}}} + \frac{\text{MASE}}{\text{MASE}_{\text{Naïve2}}} \right],
 \end{aligned}$$

where  $s$  is the periodicity of the time series,  $H$  is the prediction horizon, and  $\mathbf{Y}_h$  and  $\hat{\mathbf{Y}}_h$  are the ground truth and prediction at time step  $h$ , respectively.





## B.4. Long-term Forecasting

Table 16: Full long-term forecasting results. We use the same protocol as in Table 1.

Methods	Time-VLM	Time-LLM	GPT4TS	DLinear	PatchTST	TimesNet	FEDformer	Autoformer	Stationary	ETSformer	LightTS	Informer	Reformer	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
<i>ETTh1</i>	96	<b>0.361</b> <b>0.386</b>	<b>0.362</b> <b>0.392</b>	0.376	0.397	0.375	0.399	0.384	0.402	0.376	0.419	0.449	0.459	0.513
	192	<b>0.397</b> <b>0.415</b>	<b>0.398</b> <b>0.418</b>	0.416	0.418	0.405	0.416	0.413	0.421	0.436	0.429	0.420	0.448	0.500
	336	<b>0.420</b> <b>0.421</b>	<b>0.430</b> <b>0.427</b>	0.442	0.433	0.439	0.443	<b>0.422</b>	0.436	0.491	0.469	0.450	0.465	0.521
	720	<b>0.441</b> <b>0.458</b>	<b>0.442</b> <b>0.457</b>	0.477	0.456	0.472	0.490	0.447	0.466	0.521	0.500	0.506	0.507	0.514
	Avg	<b>0.405</b> <b>0.420</b>	<b>0.408</b> <b>0.423</b>	0.465	0.455	0.422	0.437	0.413	0.430	0.458	0.450	0.440	0.460	0.496
<i>ETTh2</i>	96	<b>0.267</b>	0.335	<b>0.268</b> <b>0.328</b>	0.285	0.342	0.289	0.353	0.274	0.336	0.340	0.374	0.358	0.397
	192	<b>0.326</b> <b>0.373</b>	<b>0.329</b> <b>0.375</b>	0.354	0.389	0.383	0.418	0.339	<b>0.379</b>	0.402	0.414	0.429	0.439	0.456
	336	<b>0.357</b> <b>0.406</b>	0.368	0.409	0.373	0.407	0.448	0.465	<b>0.329</b> <b>0.380</b>	0.452	0.452	0.496	0.487	0.482
	720	0.412	0.449	<b>0.372</b> <b>0.420</b>	0.406	0.441	0.605	0.551	<b>0.379</b> <b>0.422</b>	0.462	0.468	0.463	0.474	0.515
	Avg	0.341	0.391	<b>0.334</b> <b>0.383</b>	0.381	0.412	0.431	0.446	<b>0.330</b> <b>0.379</b>	0.414	0.427	0.437	0.449	0.450
<i>ETTm1</i>	96	0.304	0.346	<b>0.272</b> <b>0.334</b>	0.292	0.346	0.299	0.343	<b>0.290</b> <b>0.342</b>	0.338	0.375	0.370	0.419	0.505
	192	0.332	<b>0.366</b>	<b>0.310</b> <b>0.358</b>	0.332	0.372	0.335	0.365	<b>0.332</b>	0.369	0.374	0.387	0.426	0.441
	336	0.364	<b>0.383</b>	<b>0.352</b> <b>0.384</b>	0.366	0.394	0.369	0.386	0.366	0.392	0.410	0.411	0.445	0.459
	720	0.402	<b>0.410</b>	<b>0.383</b> <b>0.411</b>	0.417	0.421	0.425	0.421	0.416	0.420	0.478	0.450	0.543	0.490
	Avg	<b>0.350</b> <b>0.377</b>	<b>0.329</b> <b>0.372</b>	0.388	0.403	0.357	0.378	0.351	0.380	0.400	0.406	0.448	0.452	0.588
<i>ETTm2</i>	96	<b>0.160</b> <b>0.250</b>	<b>0.161</b> <b>0.253</b>	0.173	0.262	0.167	0.269	0.165	0.255	0.187	0.267	0.203	0.287	0.255
	192	<b>0.215</b> <b>0.291</b>	<b>0.219</b> <b>0.293</b>	0.229	<b>0.301</b> <b>0.224</b>	0.303	0.220	0.229	0.249	0.309	0.260	0.328	0.281	0.340
	336	<b>0.270</b> <b>0.325</b>	<b>0.271</b> <b>0.329</b>	0.286	0.341	0.281	0.342	0.274	0.329	0.321	0.351	0.325	0.366	0.339
	720	<b>0.348</b> <b>0.378</b>	<b>0.352</b> <b>0.379</b>	0.378	0.401	0.397	0.421	0.362	0.385	0.408	0.403	0.421	0.415	0.433
	Avg	<b>0.248</b> <b>0.311</b>	<b>0.251</b> <b>0.313</b>	0.284	0.339	0.267	0.333	0.255	0.315	0.291	0.333	0.305	0.349	0.327
<i>Weather</i>	96	<b>0.148</b> <b>0.200</b>	<b>0.147</b>	0.201	0.162	0.212	0.176	0.237	0.149	<b>0.198</b>	0.172	0.220	0.217	0.296
	192	<b>0.193</b> <b>0.240</b>	<b>0.189</b> <b>0.234</b>	0.204	0.248	0.220	0.282	0.194	0.241	0.219	0.261	0.276	0.336	0.307
	336	<b>0.243</b> <b>0.281</b>	<b>0.262</b> <b>0.279</b>	0.254	0.284	0.265	0.319	<b>0.245</b>	0.282	0.280	0.306	0.339	0.380	0.359
	720	<b>0.312</b> <b>0.332</b>	<b>0.304</b> <b>0.316</b>	0.326	0.337	0.333	0.362	0.314	0.334	0.365	0.359	0.403	0.428	0.419
	Avg	<b>0.224</b> <b>0.263</b>	<b>0.225</b> <b>0.257</b>	0.237	0.270	0.248	0.300	0.225	0.264	0.259	0.287	0.309	0.360	0.338
<i>Electricity</i>	96	0.142	0.245	<b>0.131</b> <b>0.224</b>	0.139	0.238	0.140	0.237	<b>0.129</b> <b>0.222</b>	0.168	0.272	0.193	0.308	0.201
	192	0.157	0.260	<b>0.152</b> <b>0.241</b>	0.153	0.251	0.153	0.249	<b>0.157</b> <b>0.240</b>	0.184	0.289	0.201	0.315	0.222
	336	0.174	0.276	<b>0.160</b> <b>0.248</b>	0.169	0.266	0.169	0.267	<b>0.163</b> <b>0.259</b>	0.198	0.300	0.214	0.329	0.231
	720	0.214	0.308	<b>0.192</b> <b>0.298</b>	0.206	0.297	0.203	0.301	<b>0.197</b> <b>0.290</b>	0.220	0.320	0.246	0.355	0.254
	Avg	0.172	0.273	<b>0.158</b> <b>0.252</b>	0.167	<b>0.263</b>	0.166	<b>0.263</b>	<b>0.161</b> <b>0.252</b>	0.192	0.295	0.214	0.327	0.227
<i>Traffic</i>	96	0.393	0.290	<b>0.362</b> <b>0.248</b>	0.388	0.282	0.410	0.282	<b>0.360</b> <b>0.249</b>	0.593	0.321	0.587	0.366	0.613
	192	0.405	0.296	<b>0.374</b> <b>0.247</b>	0.407	0.290	0.423	0.287	<b>0.379</b> <b>0.256</b>	0.617	0.336	0.604	0.373	0.616
	336	0.420	0.305	<b>0.385</b> <b>0.271</b>	0.412	0.294	0.436	0.296	<b>0.392</b> <b>0.264</b>	0.629	0.336	0.621	0.383	0.618
	720	0.459	0.323	<b>0.430</b> <b>0.288</b>	0.450	0.312	0.466	0.315	<b>0.432</b> <b>0.286</b>	0.640	0.350	0.626	0.382	0.660
	Avg	0.419	0.303	<b>0.388</b> <b>0.264</b>	0.414	0.294	0.433	0.295	<b>0.390</b> <b>0.263</b>	0.620	0.336	0.610	0.376	0.624

## C. Visualizations

### C.1. Visualization of Generated Time Series Images

The image generation module employs advanced techniques—frequency and periodicity Encoding, multi-scale convolution, interpolation and normalization—to create informative and discriminative image representations of time series data. These representations enhance downstream VLMs for improved forecasting. As shown in Figure 6, the generated images capture key temporal characteristics through the following features:

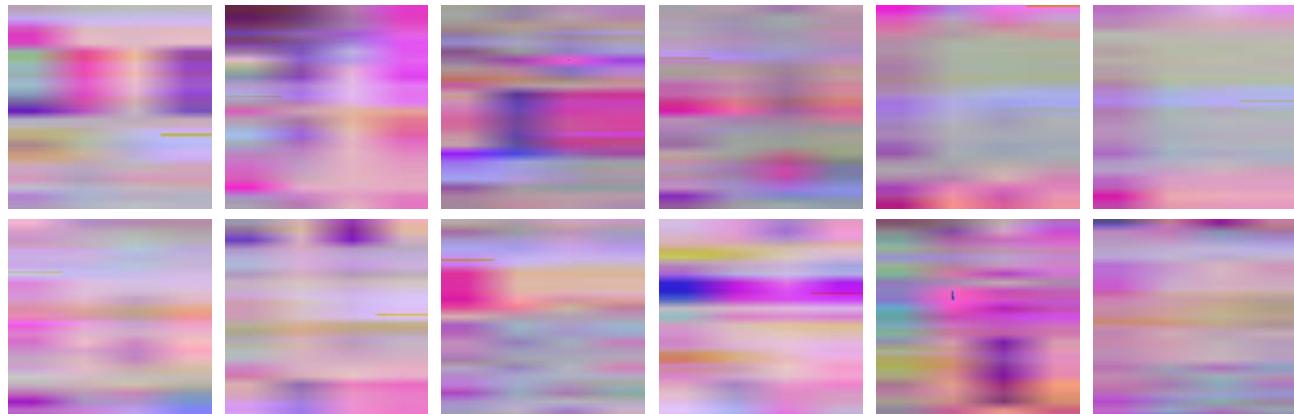


Figure 6: Time series transformed images, capturing key temporal characteristics, including trends, stationarity, seasonality, sudden changes, and frequency-domain patterns.

- **Frequency-Domain Information:** FFT integration captures frequency-domain characteristics, visualized as distinct textures—fine-grained for high-frequency components and broader color regions for low-frequency components.
- **Multi-scale Periodic Encoding:** Temporal dependencies at multiple scales (e.g., daily, weekly) are encoded, visible as regular patterns such as repeating vertical bands for daily cycles or broader horizontal patterns for weekly cycles.
- **Image Interpolation:** Bilinear interpolation ensures smooth and coherent images, preserving essential time series characteristics through seamless transitions between color intensities.
- **Color Trends:** Color intensity corresponds to time series values—darker regions (e.g., deep blue) indicate lower values, while brighter regions (e.g., yellow) represent higher values, enabling easy identification of trends.
- **Abrupt Changes and Anomalies:** Sudden shifts in color intensity (e.g., sharp transitions from dark to bright) highlight abrupt changes or anomalies, crucial for identifying irregular events like traffic spikes or weather shifts.

## C.2. Visualization of prediction results

The prediction results in Figures 7, 8, 9, and 10 demonstrate Time-VLM’s ability to accurately forecast time series across diverse datasets and prediction horizons. For datasets with clear periodic structures, such as the daily cycles in ETTh1 and ETTm1, Time-VLM captures both global trends and fine-grained temporal patterns effectively. This is evident in the close alignment between the true values (solid lines) and predicted values (dashed lines) across all horizons. Similarly, for the ECL dataset, which exhibits regular consumption patterns, Time-VLM delivers highly accurate forecasts, showcasing its strength in handling structured environments.

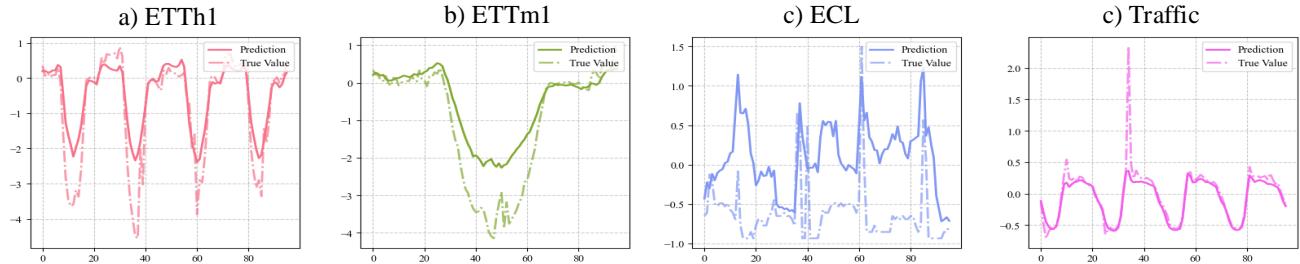


Figure 7: Prediction results visualization for ETTh1, ETTm1, ECL, and Traffic datasets at 96 prediction lengths. True values (solid line) and predicted values (dashed line) are shown for each dataset and horizon.

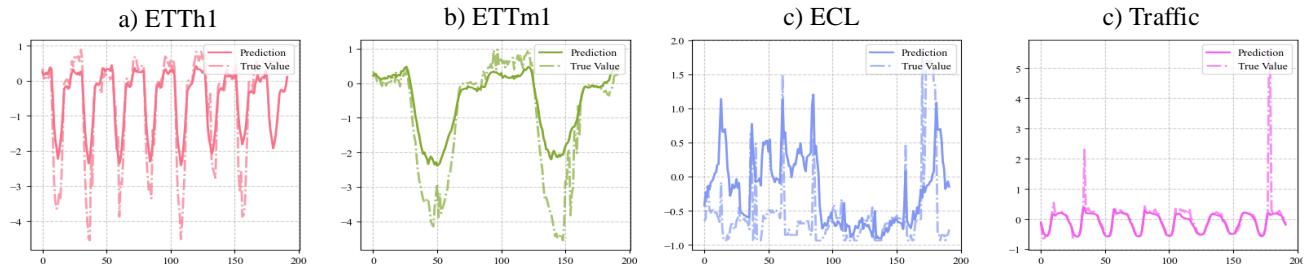


Figure 8: Prediction results visualization for ETTh1, ETTm1, ECL, and Traffic datasets at 192 prediction lengths. True values (solid line) and predicted values (dashed line) are shown for each dataset and horizon.

However, performance varies for datasets with irregular or abrupt changes. On the Traffic dataset, which is characterized by non-stationary patterns, Time-VLM shows slight deviations in capturing sudden fluctuations, particularly at longer horizons (e.g., 336 and 720). These deviations highlight the challenges of modeling highly irregular data and suggest opportunities for refining the time series-to-image transformation process to better handle such scenarios.

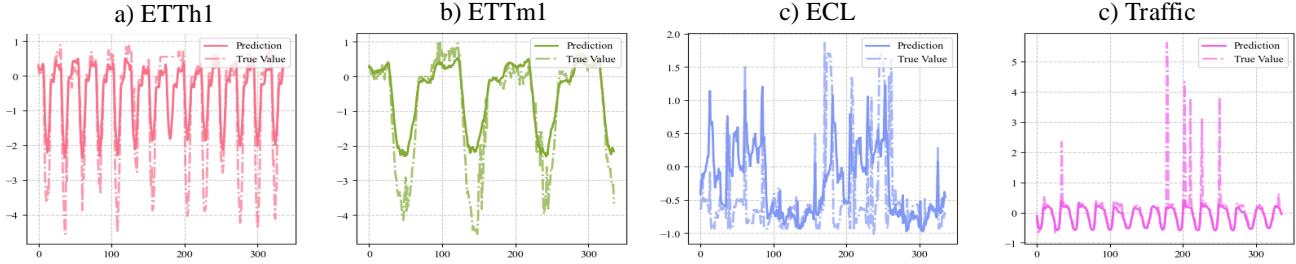


Figure 9: Prediction results visualization for ETTh1, ETTm1, ECL, and Traffic datasets at 336 prediction lengths. True values (solid line) and predicted values (dashed line) are shown for each dataset and horizon.

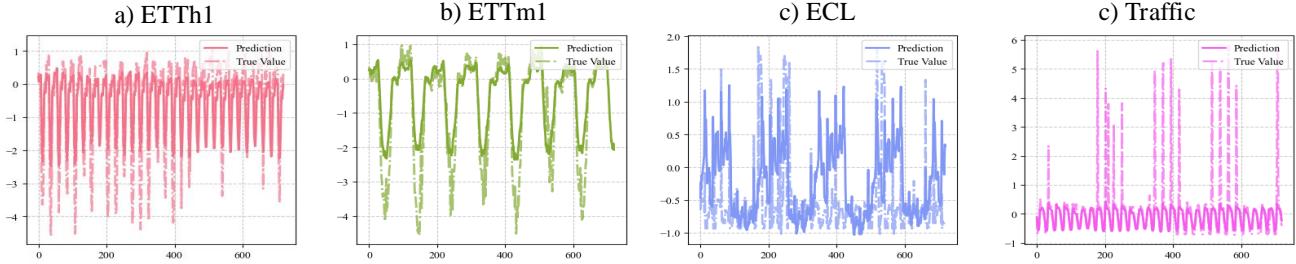


Figure 10: Prediction results visualization for ETTh1, ETTm1, ECL, and Traffic datasets at 720 prediction lengths. True values (solid line) and predicted values (dashed line) are shown for each dataset and horizon.

## D. Future Work

### D.1. Limitations

While Time-VLM demonstrates significant improvements in time series forecasting by integrating temporal, visual, and textual modalities, it has some limitations.

First, the framework performs less robustly on datasets with highly volatile or irregular patterns, such as those with sudden changes or non-stationary trends, compared to datasets with periodic structures. This limitation may arise from the current visual transformation techniques, which may not adequately capture abrupt temporal dynamics or sudden shifts. Future work could refine these transformations to better handle such irregularities.

Second, the current implementation relies on pre-trained VLMs like ViLT and CLIP, which are optimized for natural vision-language tasks rather than time series forecasting. While these models excel in visual understanding, their textual capabilities are limited, often supporting only shorter text inputs and lacking domain-specific knowledge relevant to time series. This restricts their ability to fully utilize textual context for forecasting. Future work could involve developing larger, domain-specific VLMs trained on multimodal time series datasets to address these limitations.

### D.2. Future Work

Building on the current framework, several promising directions for future research emerge:

- **Optimizing Visual Transformations:** Future work could focus on developing adaptive visual transformation techniques that better preserve temporal dynamics, especially for datasets with irregular or non-stationary patterns, to more effectively highlight sudden changes and complex trends.
- **Scaling Multimodal VLMs for Enhanced Forecasting:** While the current framework uses smaller pre-trained Vision-Language Models (VLMs), scaling to larger models could improve forecasting accuracy. Investigating trade-offs between model size, computational efficiency, and performance is a promising direction for future research. Additionally, studying different VLM architectures could identify optimal designs for temporal modeling.
- **Interpretable Multimodal Learning for Time Series Analysis:** Understanding the contributions of visual and textual modalities in time series forecasting is crucial for improving model transparency. Future work could explore the

interpretability of multimodal features, analyzing how different types of information contribute to performance gains. This would provide deeper insights into temporal dependencies and enhance trust in multimodal forecasting models.

- **Pre-training Multimodal Foundation Models for Time Series Analysis:** Existing VLMs are not designed to handle time series data, limiting their ability to capture domain-specific temporal context. Future research could focus on constructing large-scale multimodal datasets that pair time series data with rich textual and visual annotations, enabling the development of models specifically optimized for time series forecasting. Additionally, this multimodal framework could be extended to support multi-task learning, enhancing the model's versatility for tasks such as anomaly detection, classification, or imputation. This would allow the model to capture a broader range of temporal patterns and dependencies, improving its applicability across various domains.

By addressing these directions, future research can build on the foundation laid by Time-VLM, advancing the field of multimodal time series forecasting while ensuring responsible and ethical deployment in real-world applications.