TEXT-CONDITIONED TIME-SERIES GENERATION: CROSS-MODAL ALIGNMENT AND FINE-GRAINED CONTROL IN TIME–FREQUENCY IMAGE SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Text-conditioned time-series generation enables flexible and fine-grained control over generated time series, supporting a wide range of applications. Despite its potential, existing methods face two key challenges: (1) limited alignment between time series and text (TS–text alignment); and (2) limited adherence to fine-grained textual instructions. In this paper, we propose TextPlot, an image-based diffusion framework for text-conditioned time-series generation. To address the first challenge, we introduce the short-time Fourier transform (STFT) as an intermediate representation, mapping a one-dimensional time series to a two-dimensional time–frequency image with explicit temporal and frequency coordinates. The resulting time–frequency image provides a structured representation for relating textual descriptions to temporal dynamics. For the second challenge, TextPlot employs a dual-path text-conditioning mechanism. Global pooling provides sentence-level context, while token-level cross-attention preserves fine-grained textual information. These two conditioning paths jointly guide a 2D U-Net to generate time–frequency images through iterative denoising. Across one synthetic and four real-world datasets, TextPlot achieves superior TS–text alignment compared with the evaluated baselines. Further experiments show average relative improvements of 79.1% and 83.5% over the evaluated baselines in following original and edited textual requirements, respectively.

1 INTRODUCTION

Time-series data support a wide range of applications, including financial trading (Sezer & Ozbayoglu, 2018), weather forecasting (Rasp et al., 2020), and medical monitoring (Johnson et al., 2016). However, obtaining sufficient and representative data to model different scenarios can be challenging (Gu et al., 2025; Ge et al., 2025). Time-series generation helps address this limitation by synthesizing realistic sequences for data augmentation and scenario simulation (Yoon et al., 2019; Zhang et al., 2026; Zhou et al., 2026). Early methods mainly relied on structured conditions such as metadata (Narasimhan et al., 2024), time-series attributes (Jing et al., 2024), or class labels (Lan et al., 2026), whereas recent studies have adopted natural-language descriptions to enable more flexible and fine-grained generation (Ge et al., 2025; Gu et al., 2025; Li et al., 2025; Wang et al., 2025b). Natural-language-conditioned time-series generation allows textual descriptions to convey diverse temporal patterns, playing a crucial role in modern society.

Since natural language and time series organize information in fundamentally different forms, natural-language-conditioned generation faces a substantial modality gap. Existing studies have explored different ways to connect the two modalities. For example, T2S learns a compact latent representation of time series and uses text embeddings to guide generation (Ge et al., 2025). Time-LLM uses cross-attention to reprogram time-series patches into text prototype representations (Jin et al., 2024). However, Zheng et al. (2025) identify "pseudo-alignment," where apparent alignment in embedding space does not necessarily reflect meaningful correspondence between time series and language. Figure 1(a) illustrates TS–text misalignment, with a generated series that does not match its conditioning description. These observations highlight the need for an effective bridge between the two modalities. To facilitate TS–text alignment, introducing an intermediate modality is a possible approach. Evidence from Yashwante & Yu (2026) shows that structured image representations

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

derived from time series externalize latent temporal structures, allowing images to serve as semantic intermediaries between time series and language. This observation motivates the use of a structured image representation for aligning time series and text.

While natural language provides richer semantic descriptions, its fine-grained information is difficult to extract and apply during generation. To incorporate such detailed textual information, LFFD uses token-level language representations and cross-attention to guide local temporal patterns (Zhou et al., 2026), while MTransformer converts textual descriptions into frequency-specific conditions to guide the generation of trends and fluctuations (Zhang, 2026). However, ConTSG-Bench reports that generated local patterns often fail to match their segment-level descriptions on real-world data, indicating that the evaluated generators still struggle to follow fine-grained textual requirements (Lan et al., 2026). As illustrated in Figure 1(b), these results highlight the difficulty of extracting fine-grained textual instructions and applying them during generation. This motivates a conditioning mechanism that supports fine-grained textual instruction following.

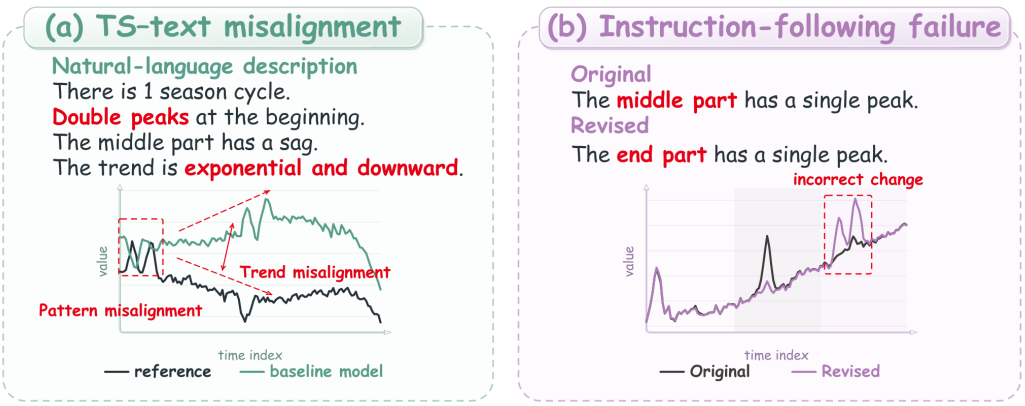


Figure 1: Examples of TS–text misalignment and failures to follow revised textual instructions. (a) TS–text misalignment: the description specifies a downward exponential trend and double peaks near the beginning, whereas the generated series initially rises and fails to reproduce the specified local pattern. (b) Instruction-following failure: changing the target to a single peak in the end region still yields two peaks in the revised output.

Building on the two motivations, we propose TextPlot, an image-based diffusion framework for natural-language-conditioned time-series generation. We first recognize that STFT, widely used in signal processing to analyze how frequency content evolves over time (Allen & Rabiner, 1977), provides a structured view of temporal dynamics. TextPlot uses this representation to organize time series into time–frequency images with explicit temporal and frequency coordinates, providing an intermediate representation for connecting textual descriptions with temporal patterns. Second, considering that textual descriptions convey both overall context and fine-grained requirements, TextPlot introduces a dual-path text-conditioning mechanism to capture these complementary aspects. The global pathway conditions denoising on the description as a whole, while the token-level pathway allows time–frequency features to attend selectively to individual textual details. Together, these pathways guide the U-Net in generating time–frequency images that reflect both the overall description and its fine-grained requirements.

Our main contributions are as follows:

- We introduce an STFT-based time–frequency image as an intermediate representation to bridge the modality gap between text and time series. By explicitly organizing temporal positions and spectral content, this representation provides a structured basis for connecting textual descriptions with the temporal patterns they describe.
- We propose TextPlot, an image-based diffusion framework that generates time series consistent with both the overall patterns and fine-grained requirements expressed in natural language. The framework brings TS–text alignment and textual instruction following together within a unified generative process.

- Experiments on five datasets demonstrate stronger TS–text alignment than the evaluated baselines. Further evaluations show substantial improvements in following both original and revised textual instructions.

2 Related Work

2.1 Text Conditioning in Time-Series Generation

Conditioning in time-series generation has evolved from structured signals, such as metadata, time-series attributes, and class labels (Narasimhan et al., 2024; Gu et al., 2025; Lan et al., 2026), to natural language (Ge et al., 2025; Gu et al., 2025; Li et al., 2025; Wang et al., 2025b). Although structured conditions are easy to use, their predefined spaces limit the diversity of generated sequences. In contrast, natural language can describe temporal dynamics in richer semantic detail, enabling more precise and flexible control over generation. Recent methods exploit natural-language conditions in different ways. For instance, T2S uses text embeddings to modulate its diffusion Transformer through adaptive layer normalization, guiding time-series generation in latent space (Ge et al., 2025), VerbalTS models text at multiple temporal resolutions (Gu et al., 2025), and BRIDGE combines textual descriptions with semantic prototypes (Li et al., 2025). However, ConTSG-Bench reports near-chance segment-level retrieval accuracy for most evaluated models on real-world data as the candidate pool grows (Lan et al., 2026). These results suggest insufficient utilization of textual conditions, reflecting limited adherence to fine-grained textual instructions.

2.2 Time-Series–Text Alignment

In cross-modal learning, experimental results indicate that independently pretrained time-series and language representations are nearly orthogonal and remain difficult to align even with contrastive learning (Yashwante & Yu, 2026). This modality gap calls for meaningful correspondence mechanisms between time series and natural language. Studies on TS–text alignment mainly map time-series features into a language embedding space, align temporal features with semantic prototypes, or introduce cross-modal interaction at different levels in the model (Liu et al., 2025a; Wang et al., 2025a; Sun et al., 2025). However, because these representations are learned under different modality-specific structures and objectives, feature-level alignment may not guarantee a meaningful temporal–semantic correspondence (Liu et al., 2025b; Zheng et al., 2025; Zhang et al., 2026). These results show that existing alignment mechanisms provide only a partial solution to the modality gap between time series and natural language.

Recently, studies have further explored using an intermediate modality to connect time series with natural language. Research shows that structured image representations derived from time series can externalize latent temporal structures, making trends, peaks, and periodic patterns more directly observable than in raw numerical sequences. Consequently, structured images can serve as semantic intermediaries between time series and language (Yashwante & Yu, 2026). Recent studies use images to connect time series and language. For instance, Insight Miner converts time-series windows into chart images and uses vision-language models to extract temporal semantics (Zhang et al., 2025b). More recent methods, including VISTA, VisMoE, and Time-VLM, combine line-chart or other image representations with textual prompts or descriptions to reduce the mismatch between numerical sequences and language (Khezresmaeilzadeh et al., 2025; Liu et al., 2025c; Zhong et al., 2025). ST-VLM further constructs vision, text, and time-series modalities within a unified framework for spatiotemporal prediction (Zhao et al., 2026). These studies motivate our investigation of structured image representations for connecting time series with language.

3 Method

In this section, we introduce TextPlot, which consists of three key components: an STFT-based generative state, a dual-path text-conditioning mechanism, and a 2D U-Net for conditional diffusion. First, the STFT-based state represents a time series as a two-dimensional time–frequency image, enabling joint modeling of temporal and spectral structures (Section 3.2). Then the dual-path conditioning mechanism provides global and token-level representations, preserving overall semantics while retaining fine-grained information (Section 3.3). Finally, the conditional diffusion model gen-

erates time–frequency images under both conditioning paths (Section 3.4). The generated images are then converted back into time series using the inverse short-time Fourier transform (iSTFT). The overall architecture of TextPlot is illustrated in Figure 2.

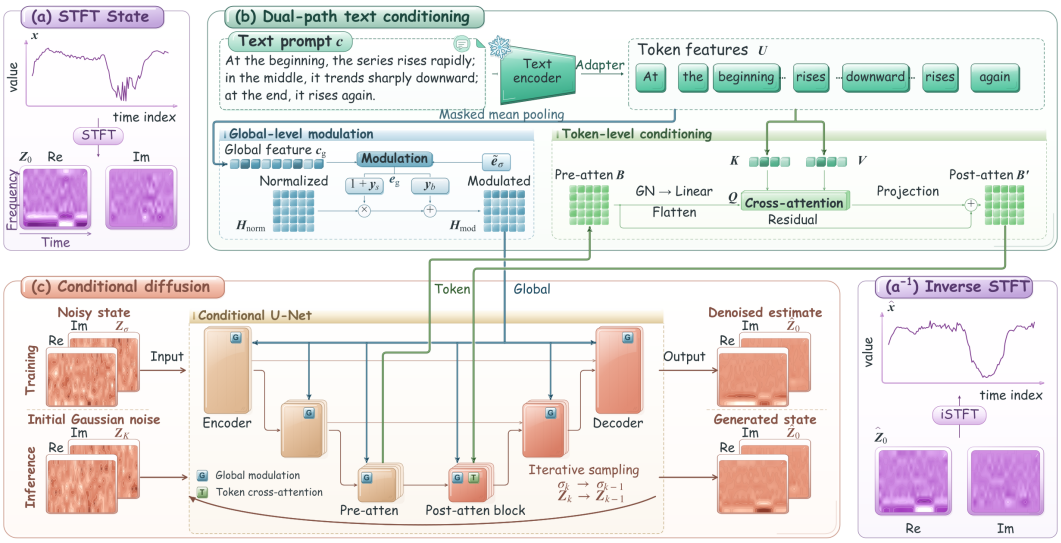


Figure 2: Overall architecture of TextPlot. The model converts time series into time–frequency images, incorporates natural-language information through dual-path conditioning, and generates time series using a conditional U-Net.

3.1 Problem Formulation

We study natural-language-conditioned time-series generation. Consider a paired sample of a time series $\boldsymbol{x} \in \mathbb{R}^L$ and its textual description $\boldsymbol{c} \in \mathbb{N}^M$. Here, L denotes the sequence length, and M denotes the number of text tokens. Our goal is to learn a conditional generative model $p_\theta(\boldsymbol{x} \mid \boldsymbol{c})$ and generate a time series $\hat{\boldsymbol{x}} \sim p_\theta(\boldsymbol{x} \mid \boldsymbol{c})$ that is consistent with the given textual description.

3.2 STFT-Based Generative State

Motivated by the structured image representation analysis in Section 2.2, we represent each time series in a structured two-dimensional form for text-conditioned generation. To construct this representation, we decompose the time series into frequency components using STFT. STFT uses a sliding window that moves along the time axis. At each position, the window extracts a local segment for the fast Fourier transform (FFT). These local frequency representations are then organized according to the temporal positions of the segments.

Specifically, using a Hann window of length W, a hop size H, and a W-point FFT, we compute

$$X_{f,t} = \sum_{n=0}^{W-1} \tilde{x}_{tH+n}\, w_W[n]\, e^{-\mathrm{i}2\pi fn/W}, \tag{1}$$

where $\tilde{\boldsymbol{x}}$ denotes the reflection-padded input, $w_W[n]$ is the Hann window, and $f = 0, \ldots, F-1$ and $t = 0, \ldots, T-1$ index the F frequency bins and T time frames, respectively. Here, the window length W controls the trade-off between temporal and frequency resolution, and we examine the effect of this key hyperparameter in Section 4.5.

After this transformation, we obtain a complex time–frequency representation $\boldsymbol{X} \in \mathbb{C}^{F\times T}$. The real and imaginary parts of $\boldsymbol{X}$ are represented as two channels, yielding $\boldsymbol{X}_{\text{img}} \in \mathbb{R}^{2\times F\times T}$. This two-channel representation preserves both the magnitude and phase of the complex STFT coefficients, allowing reconstruction of the original sequence via the iSTFT with negligible information loss (Naiman et al., 2024; Welker et al., 2022).

In TextPlot, we use this two-channel representation to construct the generative state $\boldsymbol{Z}_0$ for conditional diffusion, as shown in Figure 2(a). Following Richter et al. (2023), we apply the phase-preserving mapping $|c| \mapsto \alpha|c|^{0.5}$ to the coefficient magnitudes, where α is a scaling factor. This mapping compresses the magnitude range to better represent spectral components with smaller magnitudes. After diffusion sampling produces the denoised state $\hat{\boldsymbol{Z}}_0$, we perform reconstruction as shown in Figure 2(a^{-1}). We recombine the two channels of the denoised state into complex coefficients, reverse the magnitude mapping, and apply iSTFT to obtain the generated time series $\hat{\boldsymbol{x}}$.

3.3 Dual-Path Text Conditioning

Textual descriptions can combine an overall pattern with specific requirements for local segments. For example, "an upward trend with a brief dip in the middle" requires the generated series to follow the overall rising trend while placing a short downward deviation in the specified region. Therefore, the model needs to maintain consistency with the overall pattern and accurately capture local details to achieve fine-grained control. TextPlot addresses textual requirements through a dual-path text-conditioning mechanism, as illustrated in Figure 2(b): the global path provides sentence-level context, while the token-level path allows the denoising network to attend to specific textual requirements at different time–frequency positions.

Global conditioning. A frozen LongCLIP encoder (Zhang et al., 2025a) extracts contextual token features. The features are then mapped to shared features $\boldsymbol{U}$ through a trainable token-wise adapter (Appendix B.2). We average these features over valid token positions to obtain a global text vector $\boldsymbol{c}_{\mathrm{g}}$ that summarizes the description. To obtain $\boldsymbol{e}_{\mathrm{g}}$, the global conditioning vector, we first apply a learned linear projection $\boldsymbol{W}_{\mathrm{g}}$ to $\boldsymbol{c}_{\mathrm{g}}$, then add the noise-level embedding $\tilde{\boldsymbol{e}}_\sigma$ and apply SiLU:

$$\boldsymbol{e}_{\mathrm{g}} = \mathrm{SiLU}\left(\tilde{\boldsymbol{e}}_\sigma + \boldsymbol{W}_{\mathrm{g}}\boldsymbol{c}_{\mathrm{g}}\right). \tag{2}$$

Within each U-Net residual block, we use $\boldsymbol{e}_{\mathrm{g}}$ to modulate the intermediate features. First, we pass $\boldsymbol{e}_{\mathrm{g}}$ through a learned linear layer and split its output into two channel-wise vectors, $\boldsymbol{y}_s$ and $\boldsymbol{y}_b$. Next, we apply GroupNorm to the output of the first convolution in the block to obtain the normalized feature map $\boldsymbol{H}_{\mathrm{norm}}$. Finally, we use $\boldsymbol{y}_s$ and $\boldsymbol{y}_b$ to adjust the scale and shift of these normalized features:

$$\boldsymbol{H}_{\mathrm{mod}} = (1 + \boldsymbol{y}_s) \odot \boldsymbol{H}_{\mathrm{norm}} + \boldsymbol{y}_b. \tag{3}$$

Here, $\odot$ denotes element-wise multiplication. The vectors $\boldsymbol{y}_s$ and $\boldsymbol{y}_b$ are broadcast across all time–frequency positions, providing global textual guidance throughout the denoising network.

Token-level conditioning. We use cross-attention at the U-Net bottleneck to incorporate the adapted token features $\boldsymbol{U}$ (Vaswani et al., 2017; Rombach et al., 2022). The bottleneck feature map $\boldsymbol{B}$ provides the queries, while $\boldsymbol{U}$ provides the keys and values. We apply GroupNorm (GN) to $\boldsymbol{B}$, then arrange the features at its time–frequency positions into a sequence for subsequent cross-attention. The token features $\boldsymbol{U}$ are normalized using LayerNorm (LN). For each attention head h, learned linear projections produce the queries, keys, and values:

$$\boldsymbol{Q}_h = \mathrm{flat}(\mathrm{GN}(\boldsymbol{B}))\boldsymbol{W}_{\mathrm{Q},h}, \quad \boldsymbol{K}_h = \mathrm{LN}(\boldsymbol{U})\boldsymbol{W}_{\mathrm{K},h}, \quad \boldsymbol{V}_h = \mathrm{LN}(\boldsymbol{U})\boldsymbol{W}_{\mathrm{V},h}. \tag{4}$$

Each head aggregates textual information through scaled dot-product attention:

$$\boldsymbol{A}_h = \mathrm{softmax}\left(\frac{\boldsymbol{Q}_h\boldsymbol{K}_h^\top}{\sqrt{d_{\mathrm{h}}}} + \boldsymbol{\Gamma}\right)\boldsymbol{V}_h, \tag{5}$$

where d_{h} is the dimension of each attention head and $\boldsymbol{\Gamma}$ masks padding tokens. We concatenate the outputs of all attention heads along the feature dimension to obtain $\boldsymbol{A}$. A learned output projection $\boldsymbol{W}_{\mathrm{O}}$ maps $\boldsymbol{A}$ back to the channel dimension of $\boldsymbol{B}$. The projected features are then placed back into the original time–frequency grid in the same order and added to $\boldsymbol{B}$:

$$\boldsymbol{B}' = \boldsymbol{B} + \mathrm{reshape}\left(\boldsymbol{A}\boldsymbol{W}_{\mathrm{O}}\right). \tag{6}$$

Here, reshape converts the projected attention output $\boldsymbol{A}\boldsymbol{W}_{\mathrm{O}}$ from a sequence into a feature map with the same spatial layout and channel dimension as $\boldsymbol{B}$.

3.4 Conditional Diffusion and Time-Series Reconstruction

As illustrated in Figure 2(c), TextPlot performs conditional diffusion in the time–frequency image space. During training, each time series is transformed into a clean generative state $\boldsymbol{Z}_0$ using the representation described in Section 3.2. We sample a noise level σ and add Gaussian noise to obtain $\boldsymbol{Z}_\sigma$. The model is trained to recover $\boldsymbol{Z}_0$ from $\boldsymbol{Z}_\sigma$ under the paired textual conditions, using an EDM denoising objective (Karras et al., 2022) and an STFT consistency loss that encourages agreement between the predicted spectrum and the STFT of its reconstructed time series. Details of the training objective are provided in Appendix B.6. During inference, we initialize $\boldsymbol{Z}_K$ with Gaussian noise, where K denotes the number of denoising steps. At each step k, the U-Net estimates the clean representation from $\boldsymbol{Z}_k$, conditioned on the noise level σ_k and the text. The sampler uses this estimate to update the state to $\boldsymbol{Z}_{k-1}$ at a lower noise level. After iterative denoising, we obtain the generated state $\hat{\boldsymbol{Z}}_0$ and convert it back into the time series $\hat{\boldsymbol{x}}$ using the reconstruction procedure in Section 3.2, as illustrated in Figure 2(a^{-1}).

4 Experiments

In this section, we evaluate TextPlot through experiments addressing the following four research questions (RQs):

- **RQ1:** How does TextPlot compare with existing methods in TS–text alignment?
- **RQ2:** How accurately does TextPlot satisfy fine-grained natural-language conditions?
- **RQ3:** How do TextPlot's components affect generation performance?
- **RQ4:** How does the STFT window length affect TextPlot's performance?

4.1 Experimental Settings

Datasets We evaluate TextPlot on five datasets, described as follows: (i) Synth-U, the fully synthetic dataset released with VerbalTS (Gu et al., 2025), which pairs synthetic series with manually constructed descriptions; (ii) real-world ETTm dataset (Zhou et al., 2021) used by VerbalTS (Gu et al., 2025); and (iii) three real-world datasets from TSFragment-600K (Ge et al., 2025): AirQuality, Traffic, and Electricity. For datasets from TSFragment-600K, we use sequences of length 48.

Evaluation Metrics We adopt three primary metrics from ConTSG-Bench (Lan et al., 2026) to measure TS–text alignment in Section 4.2: (i) Joint Fréchet Time Series Distance (J-FTSD) (Narasimhan et al., 2024) measures the distance between real and generated time series text joint embedding distributions. (ii) Fréchet Inception Distance (FID) (Heusel et al., 2017) compares the feature distributions of real and generated time series. (iii) Contrastive Time Series Text Pretraining (CTTP) score measures the semantic similarity between the generated time series and the text condition. For the fine-grained instruction following experiments in Section 4.3, we adopt two metrics: (i) Atomic Satisfaction measures the fraction of fine-grained conditions satisfied by the generated series, and (ii) Edited-Attribute Satisfaction measures how often the generated series satisfies the revised requirement when an individual condition in the description is changed. Both metrics compare the generated attributes with the textual requirements using dataset-specific evaluators, with metric definitions and evaluation details provided in Appendices D.2 and D.3.

Baselines We compare TextPlot with five baselines: VerbalTS (Gu et al., 2025), T2S (Ge et al., 2025), BRIDGE (Li et al., 2025), DiffuSETS (Lai et al., 2025), and a text-based retrieval baseline. Section 4.2 includes all baselines to evaluate overall generation fidelity and TS–text alignment. In Section 4.3, we focus on VerbalTS and T2S as representative textual generation methods for detailed analysis of fine-grained instruction following. Implementation details and adaptation procedures are provided in Appendix C.3.

4.2 TS–Text Alignment Comparison

In this subsection, we evaluate our method and other baselines for TS–text alignment (RQ1) on one synthetic and four real-world datasets. As shown in Table 1, TextPlot achieves the best performance in 13 of the 15 dataset–metric combinations (86.7%), including the highest CTTP score on all five

datasets. On ETTm, TextPlot achieves a CTTP score of 12.988, representing a 91.1% improvement over the score of 6.795 achieved by VerbalTS, the strongest baseline. These results demonstrate stronger TS–text alignment across the evaluated datasets, although TextPlot trails Retrieval in FID and J-FTSD on AirQuality. Additional metrics are reported in Appendix E.

Table 1: Generation results on one synthetic and four real-world datasets, evaluated using J-FTSD, FID, CTTP. Higher CTTP and lower FID/J-FTSD are better. Bold indicates the best reported point estimate within each dataset.

Dataset	Metric	TextPlot	VerbalTS	T2S	BRIDGE	DiffuSETS	Retrieval
Synth-U	CTTP ↑	**48.920**	37.242	24.632	32.507	26.318	33.996
	FID ↓	**3.586**	39.778	101.003	48.631	64.835	5.961
	J-FTSD ↓	**4.669**	44.468	108.633	55.997	77.629	15.241
ETTm	CTTP ↑	**12.988**	6.795	2.029	3.236	-3.672	4.341
	FID ↓	**4.267**	25.441	71.587	47.689	87.394	4.269
	J-FTSD ↓	**4.999**	27.576	75.690	52.116	92.496	6.724
AirQuality	CTTP ↑	**7.728**	7.401	6.709	5.958	4.022	7.145
	FID ↓	14.737	16.501	46.601	39.546	31.267	**10.360**
	J-FTSD ↓	15.555	17.328	47.507	40.592	32.408	**11.180**
Traffic	CTTP ↑	**5.317**	4.672	4.866	3.522	2.073	4.856
	FID ↓	**0.796**	5.030	23.137	31.171	68.735	2.525
	J-FTSD ↓	**0.916**	5.160	23.308	31.450	69.123	2.654
Electricity	CTTP ↑	**6.644**	5.624	5.214	4.647	4.784	5.619
	FID ↓	**3.130**	11.200	64.944	55.210	24.060	5.973
	J-FTSD ↓	**3.288**	11.381	65.224	55.507	24.384	6.152

4.3 Comparison of Fine-Grained Instruction Following

In this subsection, we conduct a paired textual-editing evaluation to measure fine-grained instruction following (RQ2). We use Atomic Satisfaction to measure the proportion of individual requirements in a textual description satisfied by the generated series. Furthermore, we modify one condition at a time and evaluate whether the generated series reflects the requested change using Edited-Attribute Satisfaction. As shown in Table 2, TextPlot consistently outperforms both baselines across all four dataset–metric combinations. TextPlot achieves average relative improvements of 79.1% in Atomic Satisfaction and 83.5% in Edited-Attribute Satisfaction. Figure 3 presents qualitative examples. These results indicate that TextPlot not only follows the explicit conditions in the original descriptions more faithfully, but also responds more accurately to changes in individual textual requirements, demonstrating stronger fine-grained instruction following.

Table 2: Fine-grained textual instruction following on Synth-U and ETTm, evaluated using Atomic Satisfaction and Edited-Attribute Satisfaction. Values are percentages (mean ± sample standard deviation across three training seeds); higher is better for both metrics. Bold indicates the best reported point estimate within each dataset.

Dataset	Method	Atomic Satisfaction	Edited-Attribute Satisfaction
Synth-U	TextPlot	**98.720** ± 0.020	**98.080** ± 0.700
Synth-U	VerbalTS	83.070 ± 0.430	83.850 ± 0.340
Synth-U	T2S	35.180 ± 0.980	34.940 ± 1.950
ETTm	TextPlot	**73.290** ± 1.680	**62.560** ± 0.780
ETTm	VerbalTS	49.340 ± 1.000	40.600 ± 1.830
ETTm	T2S	43.530 ± 6.020	34.330 ± 2.930

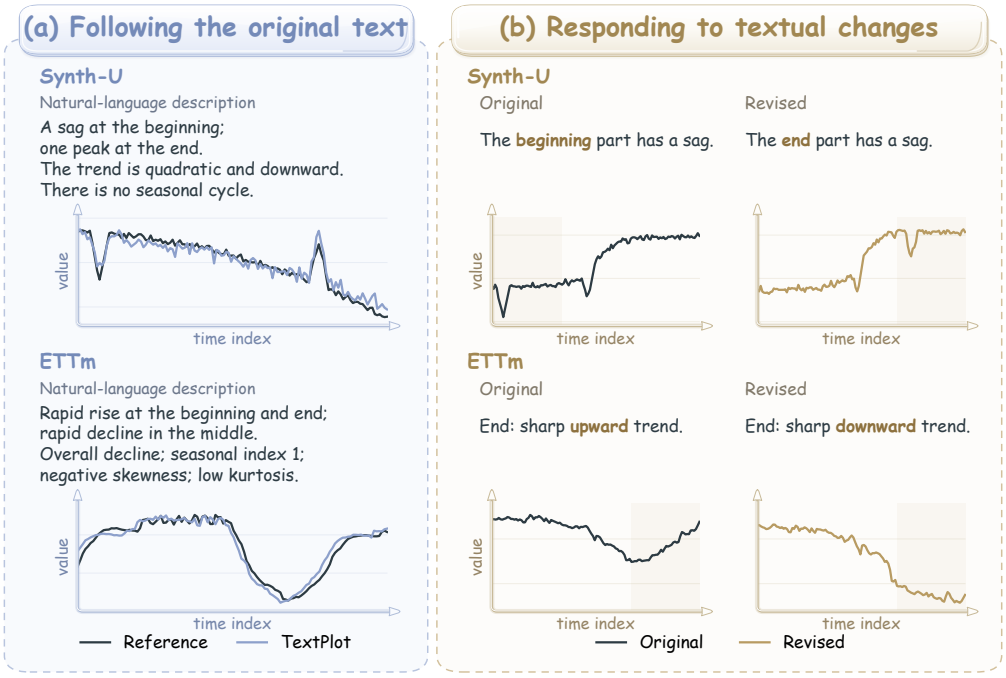


Figure 3: Illustration of fine-grained textual instruction following on Synth-U and ETTm. (a) Reference time series (dark gray) and TextPlot outputs (blue) under the original descriptions. (b) TextPlot outputs generated from the original (dark gray) and revised (gold) descriptions. Highlighted words indicate the changed requirements: the location of a sag in Synth-U and the ending trend direction in ETTm. Shaded regions mark the corresponding temporal segments.

4.4 Ablation Study

We conduct two ablation experiments to examine the effects of time-series representations, denoising backbones, and text-conditioning paths on generation performance. They examine whether the proposed time–frequency representation supports TS–text alignment and how the two conditioning paths contribute to fine-grained instruction following. This analysis links the architectural choices in TextPlot to the two challenges motivating our framework.

The first experiment evaluates the impact of different time-series representations and denoising backbones on TS–text alignment. As shown in Table 3, replacing STFT with delay embedding increases FID and J-FTSD on both datasets. Replacing U-Net with DiT-PM under the STFT representation degrades all three metrics on Synth-U, while yielding mixed results on Traffic. More importantly, the direct time-domain 1D U-Net provides a key comparison for our proposed time–frequency intermediate representation. Relative to the 1D U-Net, STFT–U-Net improves CTTP by 15.4% on Synth-U and achieves lower FID and J-FTSD on both datasets, although the 1D U-Net obtains higher CTTP on Traffic. This finding highlights STFT's role as a cross-modal bridge, making temporal and spectral structure explicitly accessible to text-conditioned generation. Together, these findings underscore the importance of representation and backbone choices, supporting the STFT–U-Net configuration adopted in TextPlot for high-quality text-conditioned time-series generation.

The second experiment examines the impact of disabling the global or token-level conditioning path on fine-grained control. We disable each path separately at inference time, yielding the w/o Global and w/o Token variants. As shown in Table 4, disabling either path reduces Atomic Satisfaction and Edited-Attribute Satisfaction on both datasets, with a larger decline when the token-level path is disabled. On Synth-U, removing token-level conditioning lowers Atomic Satisfaction and Edited-Attribute Satisfaction by 63.30 and 64.01 percentage points, respectively, compared with declines of 11.12 and 11.04 points when removing global conditioning. These results show that token-level conditioning substantially improves fine-grained instruction following, while global conditioning provides overall semantic guidance during generation.

Table 3: Representation and backbone comparisons.

Metric	U-Net			DiT-PM	
	STFT	Raw 1D	Delay embedding	STFT	Delay embedding
			Synth-U		
CTTP ↑	**48.920** ± 0.230	42.391 ± 0.273	41.189 ± 0.030	42.310 ± 0.174	41.982 ± 0.168
FID ↓	**3.586** ± 0.447	21.056 ± 0.962	24.808 ± 0.051	20.869 ± 1.414	22.938 ± 0.351
J-FTSD ↓	**4.669** ± 0.513	24.108 ± 1.036	28.159 ± 0.062	23.841 ± 1.376	26.081 ± 0.410
			Traffic		
CTTP ↑	5.317 ± 0.008	**5.475** ± 0.020	5.438 ± 0.004	5.379 ± 0.072	5.449 ± 0.025
FID ↓	**0.796** ± 0.138	1.705 ± 0.091	1.954 ± 0.144	0.933 ± 0.043	2.034 ± 0.268
J-FTSD ↓	**0.916** ± 0.139	1.819 ± 0.089	2.066 ± 0.144	1.046 ± 0.044	2.154 ± 0.263

Table 4: Conditioning-path ablation. AS: Atomic Satisfaction; EAS: Edited-Attribute Satisfaction.

Model variant	AS ↑	EAS ↑
	Synth-U	
TextPlot	**98.720** ± 0.020	**98.080** ± 0.700
w/o Global	87.600 ± 13.770	87.040 ± 15.080
w/o Token	35.420 ± 9.080	34.070 ± 9.010
	ETTm	
TextPlot	**73.290** ± 1.680	**62.560** ± 0.780
w/o Global	57.610 ± 5.350	48.950 ± 2.340
w/o Token	53.390 ± 4.070	42.180 ± 1.750

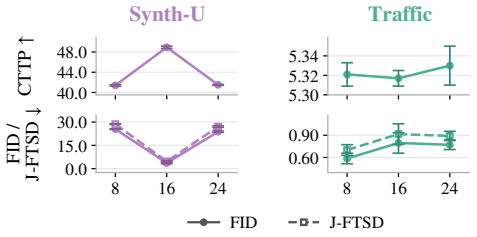


Figure 4: STFT window-length sensitivity. Means ± sample standard deviations over three training seeds.

4.5 STFT Window-Length Sensitivity

To examine the effect of STFT window length on generation performance (RQ4), we vary W over $\{8, 16, 24\}$ on Synth-U and Traffic, with $W = 16$ as the default. In Figure 4, the $W = 16$ reference from the main experiments achieves higher CTTP on Synth-U, while CTTP varies by less than 1% across window lengths on Traffic. For FID and J-FTSD, $W = 16$ yields substantially lower distances than the other reported configurations on Synth-U, whereas Traffic achieves its lowest distances at $W = 8$. Overall, the reported Synth-U results favor the default, while Traffic shows similar CTTP across windows and better generation fidelity with a shorter window.

5 Conclusion

Despite recent advances in text-conditioned time-series generation, limited TS–text alignment and limited adherence to fine-grained textual instructions remain two key challenges. We address these challenges through an STFT-based time–frequency image representation and a dual-path text-conditioning mechanism. The benefits of our approach are twofold. First, the STFT-based image representation bridges text and time series by making temporal positions and spectral content explicit, helping connect textual descriptions with the corresponding temporal patterns. Second, dual-path conditioning combines sentence-level context with token-level guidance to improve fine-grained instruction following. Evaluations across one synthetic and four real-world datasets demonstrate stronger TS–text alignment, while targeted instruction-following experiments show improved satisfaction of both original and revised textual requirements. Nevertheless, preserving the distributional characteristics of time series on some datasets remains a challenge for future work. We believe that TextPlot could facilitate dataset construction and broaden access to controllable time-series generation, particularly in data-scarce settings. By supporting the development of time-series models, these capabilities could contribute to more efficient resource management and improved public services.

AI USE STATEMENT

Generative AI tools assisted with code implementation, result analysis, and manuscript preparation. The authors reviewed the AI-assisted materials and take full responsibility for the final content.

REFERENCES

J. B. Allen and L. R. Rabiner. A unified approach to short-time Fourier analysis and synthesis. *Proceedings of the IEEE*, 65(11):1558–1564, November 1977. ISSN 0018-9219. doi: 10.1109/proc.1977.10770. URL https://doi.org/10.1109/proc.1977.10770.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf.

Yunfeng Ge, Jiawei Li, Yiji Zhao, Haomin Wen, Zhao Li, Meikang Qiu, Hongyan Li, Ming Jin, and Shirui Pan. T2S: High-resolution time series generation with text-to-series diffusion models. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 5208–5216. International Joint Conferences on Artificial Intelligence Organization, August 2025. doi: 10.24963/ijcai.2025/580. URL https://doi.org/10.24963/ijcai.2025/580. Main Track.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, March 2007. ISSN 0162-1459, 1537-274X. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

Shuqi Gu, Chuyue Li, Baoyu Jing, and Kan Ren. VerbalTS: Generating time series from texts. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 20448–20476. PMLR, July 2025. URL https://proceedings.mlr.press/v267/gu25a.html.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Unb5CVPtae.

Baoyu Jing, Shuqi Gu, Tianyu Chen, Zhiyu Yang, Dongsheng Li, Jingrui He, and Kan Ren. Towards editing time series. In *Advances in Neural Information Processing Systems*, volume 37, pp. 37561–37593. Curran Associates, Inc., 2024. doi: 10.52202/079017-1186. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/423d0909791493b7c10916fd328c2913-Paper-Conference.pdf.

Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 3(1):160035, May 2016. ISSN 2052-4463. doi: 10.1038/sdata.2016.35. URL https://www.nature.com/articles/sdata201635.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 26565–26577. Curran Associates, Inc., 2022. doi: 10.52202/068431-1926. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/a98846e9d9cc01cfb87eb694d946ce6b-Paper-Conference.pdf.

Tina Khezresmaeilzadeh, Parsa Razmara, Seyedarmin Azizi, Mohammad Erfan Sadeghi, and Erfan Baghaei Potraghloo. VISTA: Vision-language inference for training-free stock time-series analysis, May 2025. URL https://arxiv.org/abs/2505.18570. arXiv preprint arXiv:2505.18570.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/0234c510bc6d908b28c70ff313743079-Paper.pdf.

Yongfan Lai, Jiabo Chen, Qinghao Zhao, Deyun Zhang, Yue Wang, Shijia Geng, Hongyan Li, and Shenda Hong. DiffuSETS: 12-lead ECG generation conditioned on clinical text reports and patient-specific information. *Patterns*, 6(10):101291, October 2025. ISSN 2666-3899. doi: 10.1016/j.patter.2025.101291. URL https://doi.org/10.1016/j.patter.2025.101291.

Shaocheng Lan, Shuqi Gu, Zhangzhi Xiong, and Kan Ren. ConTSG-Bench: A unified benchmark for conditional time series generation, March 2026. URL https://arxiv.org/abs/2603.04767. arXiv preprint arXiv:2603.04767.

Hao Li, Yu-Hao Huang, Chang Xu, Viktor Schlegel, Renhe Jiang, Riza Batista-Navarro, Goran Nenadic, and Jiang Bian. BRIDGE: Bootstrapping text to control time-series generation via multi-agent iterative optimization and diffusion modeling. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 34742–34773. PMLR, July 2025. URL https://proceedings.mlr.press/v267/li25ah.html.

Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. TimeCMA: Towards LLM-empowered multivariate time series forecasting via cross-modality alignment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39 (18):18780–18788, April 2025a. ISSN 2374-3468. doi: 10.1609/aaai.v39i18.34067. URL https://ojs.aaai.org/index.php/AAAI/article/view/34067.

Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. CALF: Aligning LLMs for time series forecasting via cross-modal fine-tuning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(18):18915–18923, April 2025b. ISSN 2374-3468. doi: 10.1609/aaai.v39i18.34082. URL https://ojs.aaai.org/index.php/AAAI/article/view/34082.

Xingyu Liu, Min Gao, Zongwei Wang, and Yinbing Bai. Seeing sequences like humans: Pattern classification driven time-series forecasting via vision language models. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 1914–1924, Seoul Republic of Korea, November 2025c. doi: 10.1145/3746252.3761199. URL https://doi.org/10.1145/3746252.3761199.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7176–7185. PMLR, July 2020. URL https://proceedings.mlr.press/v119/naeem20a.html.

Ilan Naiman, Nimrod Berman, Itai Pemper, Idan Arbiv, Gal Fadlon, and Omri Azencot. Utilizing image transforms and diffusion models for generative modeling of short and long time series. In *Advances in Neural Information Processing Systems*, volume 37, pp. 121699–121730. Curran Associates, Inc., 2024. doi: 10.52202/079017-3868. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/dc6748383752138af7f00b3185a0a404-Paper-Conference.pdf.

Sai Shankar Narasimhan, Shubhankar Agarwal, Oguzhan Akcin, Sujay Sanghavi, and Sandeep P. Chinchali. Time Weaver: A conditional time series generation model. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 37293–37320. PMLR, July 2024. URL https://proceedings.mlr.press/v235/narasimhan24a.html.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182, Paris, France, October 2023. IEEE. doi: 10.1109/iccv51070.2023.00387. URL https://ieeexplore.ieee.org/document/10377858.

Stephan Rasp, Peter D. Dueben, Sebastian Scher, Jonathan A. Weyn, Soukayna Mouatadid, and Nils Thuerey. WeatherBench: A benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, 12(11):e2020MS002203, November 2020. ISSN 1942-2466. doi: 10.1029/2020ms002203. URL https://doi.org/10.1029/2020ms002203.

Julius Richter, Simon Welker, Jean-Marie Lemercier, Bunlong Lay, and Timo Gerkmann. Speech enhancement and dereverberation with diffusion-based generative models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2351–2364, June 2023. ISSN 2329-9290, 2329-9304. doi: 10.1109/taslp.2023.3285241. URL https://doi.org/10.1109/taslp.2023.3285241.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10674–10685, New Orleans, LA, USA, June 2022. IEEE. doi: 10.1109/cvpr52688.2022.01042. URL https://ieeexplore.ieee.org/document/9878449.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4. doi: 10.1007/978-3-319-24574-4_28. URL https://doi.org/10.1007/978-3-319-24574-4_28.

Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, February 1978. ISSN 0096-3518. doi: 10.1109/tassp.1978.1163055. URL https://doi.org/10.1109/tassp.1978.1163055.

Omer Berat Sezer and Ahmet Murat Ozbayoglu. Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Applied Soft Computing*, 70:525–538, September 2018. ISSN 1568-4946. doi: 10.1016/j.asoc.2018.04.024. URL https://doi.org/10.1016/j.asoc.2018.04.024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Siming Sun, Kai Zhang, Xuejun Jiang, Wenchao Meng, and Qinmin Yang. Enhancing LLMs for time series forecasting via structure-guided cross-modal alignment, May 2025. URL https://arxiv.org/abs/2505.13175. arXiv preprint arXiv:2505.13175.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Pengfei Wang, Huanran Zheng, Qi'ao Xu, Silong Dai, Yiqiao Wang, Wenjing Yue, Wei Zhu, Tianwen Qian, and Liang Zhao. TS-HTFA: Advancing time-series forecasting via hierarchical text-free alignment with large language models. *Symmetry*, 17(3):401, March 2025a. ISSN 2073-8994. doi: 10.3390/sym17030401. URL https://doi.org/10.3390/sym17030401.

Shiyu Wang, Yihao Feng, Tian Lan, Ning Yu, Yu Bai, Ran Xu, Huan Wang, Caiming Xiong, and Silvio Savarese. Text2Data: Low-resource data generation with textual control. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(20):21252–21260, April 2025b. ISSN 2374-3468. doi: 10.1609/aaai.v39i20.35424. URL https://ojs.aaai.org/index.php/AAAI/article/view/35424.

Simon Welker, Julius Richter, and Timo Gerkmann. Speech enhancement with score-based generative models in the complex STFT domain. In *Interspeech 2022*, pp. 2928–2932, 2022. doi: 10.21437/Interspeech.2022-10653. URL https://www.isca-archive.org/interspeech_2022/welker22_interspeech.html.

Pratham Yashwante and Rose Yu. Time series, vision, and language: Exploring the limits of alignment in contrastive representation spaces, February 2026. URL https://arxiv.org/abs/2602.19367. arXiv preprint arXiv:2602.19367.

Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf.

Beichen Zhang, Pan Zhang, Xiaoyi Dong, Yuhang Zang, and Jiaqi Wang. Long-CLIP: Unlocking the long-text capability of CLIP. In *Computer Vision – ECCV 2024*, pp. 310–325, Cham, 2025a. Springer Nature Switzerland. ISBN 978-3-031-72983-6. doi: 10.1007/978-3-031-72983-6_18. URL https://doi.org/10.1007/978-3-031-72983-6_18.

Haochen Zhang, Gengwei Zhang, Laura Yao, Nicholas Konz, and Tianlong Chen. GALA: Generation-aware cross-modal alignment for text-to-time-series synthesis, August 2026. URL https://arxiv.org/abs/2608.13741. arXiv preprint arXiv:2608.13741.

Shijie Zhang. Spectral-aware text-to-time series generation with billion-scale multimodal meteorological data, March 2026. URL https://arxiv.org/abs/2603.27135. arXiv preprint arXiv:2603.27135.

Yunkai Zhang, Yawen Zhang, Ming Zheng, Kezhen Chen, Chongyang Gao, Ruian Ge, Siyuan Teng, Amine Jelloul, Jinmeng Rao, Xiaoyuan Guo, Chiang-Wei Fang, Zeyu Zheng, and Jie Yang. Insight Miner: A time series analysis dataset for cross-domain alignment with natural language, December 2025b. URL https://arxiv.org/abs/2512.11251. arXiv preprint arXiv:2512.11251.

Tong Zhao, Junping Du, Zhe Xue, Meiyu Liang, Aijing Li, Xiaolong Meng, and Dandan Liu. ST-VLM: A spatial-to-image multimodal spatial-temporal prediction framework with vision-language model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):16441–16449, March 2026. ISSN 2374-3468. doi: 10.1609/aaai.v40i19.38683. URL https://ojs.aaai.org/index.php/AAAI/article/view/38683.

Liangwei Nathan Zheng, Chang Dong, Wei Emma Zhang, Lin Yue, Miao Xu, Olaf Maennel, and Weitong Chen. Understanding why large language models can be ineffective in time series analysis: The impact of modality alignment. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 4026–4037, Toronto ON Canada, August 2025. doi: 10.1145/3711896.3737169. URL https://doi.org/10.1145/3711896.3737169.

Siru Zhong, Weilin Ruan, Ming Jin, Huan Li, Qingsong Wen, and Yuxuan Liang. Time-VLM: Exploring multimodal vision-language models for augmented time series forecasting, February 2025. URL https://arxiv.org/abs/2502.04395. arXiv preprint arXiv:2502.04395.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11106–11115, May 2021. ISSN 2374-3468. doi: 10.1609/aaai.v35i12.17325. URL https://ojs.aaai.org/index.php/AAAI/article/view/17325.

Zhenghao Zhou, Yiyan Li, Fei Xie, Lu Wang, Bo Wang, Jiansheng Wang, Zheng Yan, and Mo-Yuen Chow. An LLM-enabled frequency-aware flow diffusion model for natural-language-guided power system scenario generation, February 2026. URL https://arxiv.org/abs/2602.19522. arXiv preprint arXiv:2602.19522.

A Supplementary Experiments

A.1 Mixed-Length Generation

To evaluate generation across multiple sequence lengths, we follow the mixed-length setting of T2S (Ge et al., 2025) and use Traffic sequences with lengths $L \in \{24, 48, 96\}$. For each method, we pool the training samples from all three lengths. We compare TextPlot with VerbalTS and T2S using CTTP, FID, and J-FTSD, evaluated separately on the validation set of each length.

As shown in Table A.1, TextPlot achieves the highest CTTP scores across all three sequence lengths and the lowest FID and J-FTSD at lengths 48 and 96. At length 24, VerbalTS obtains slightly lower FID and J-FTSD, while TextPlot retains the strongest TS–text alignment. These results demonstrate that a single TextPlot model can generate time series of multiple lengths with consistently strong textual alignment and competitive generation quality.

Table A.1: Mixed-length generation on Traffic. Each method uses one shared model per training seed for all three lengths. Results are means ± sample standard deviations across three training seeds. Bold indicates the best mean at each length.

Length	Method	CTTP ↑	FID ↓	J-FTSD ↓
24	TextPlot	**6.877** ± 0.021	3.262 ± 0.344	3.436 ± 0.347
	VerbalTS	6.343 ± 0.084	**3.165** ± 0.109	**3.338** ± 0.108
	T2S	6.276 ± 0.160	12.374 ± 5.979	12.609 ± 5.992
48	TextPlot	**5.276** ± 0.023	**1.076** ± 0.244	**1.210** ± 0.246
	VerbalTS	4.682 ± 0.097	6.697 ± 0.656	6.829 ± 0.660
	T2S	4.828 ± 0.350	22.921 ± 14.517	23.096 ± 14.542
96	TextPlot	**4.519** ± 0.026	**0.502** ± 0.082	**0.596** ± 0.082
	VerbalTS	3.874 ± 0.077	7.402 ± 2.182	7.525 ± 2.188
	T2S	4.162 ± 0.258	20.813 ± 10.648	20.957 ± 10.660

A.2 Performance across Different Numbers of Conditions

To assess instruction following across complexity, we group descriptions by explicit-condition count: 3–7 on Synth-U and 7–10 on ETTm. Using the Section 4.3 samples and protocol, we compare TextPlot, VerbalTS, and T2S with Atomic Satisfaction in each group. TextPlot leads every group: its Synth-U score drops by 1.842 percentage points from three to seven conditions and stays above 97%, while ETTm scores stay above 72% through ten conditions. Overall, the results show robust instruction following as complexity increases.

Table A.2: Atomic Satisfaction (%) by condition count on Synth-U and ETTm. Results are means ± sample standard deviations across three training seeds.

Dataset	Conditions	TextPlot	VerbalTS	T2S
Synth-U	3	99.735 ± 0.183	95.026 ± 2.863	41.534 ± 2.471
	4	99.230 ± 0.054	93.479 ± 0.278	37.225 ± 2.977
	5	99.058 ± 0.065	87.988 ± 0.459	35.622 ± 1.179
	6	98.575 ± 0.040	79.472 ± 0.392	34.106 ± 0.523
	7	97.893 ± 0.227	72.940 ± 0.848	34.627 ± 0.290
ETTm	7	73.332 ± 1.898	47.947 ± 1.706	40.125 ± 6.035
	8	73.840 ± 1.564	49.037 ± 0.774	43.358 ± 6.442
	9	73.017 ± 1.490	50.012 ± 1.107	44.691 ± 5.748
	10	72.005 ± 2.343	50.743 ± 1.630	46.226 ± 5.703

B TextPlot Implementation Details

B.1 Time-Series Representation and Preprocessing

We use the short-time Fourier transform (STFT) to construct a time–frequency representation of each time series. Specifically, a Hann window of length 16 with a hop size of 4 and centered framing slides along the time axis. At each position, we weight the 16 samples within the window and apply a 16-point fast Fourier transform (FFT) to obtain the local spectrum. These spectra are arranged in temporal order, forming a representation with frequency along the vertical axis and window position along the horizontal axis. Reflection padding is applied at both ends of the sequence to cover boundary positions.

We then apply square-root compression to the magnitudes of the complex spectral coefficients while preserving their phases, using a scaling factor of 0.15, and store their real and imaginary components as two channels for the generative model. After generation, we combine the two channels into a complex spectrum, invert the magnitude compression, and apply inverse STFT to reconstruct the time series.

B.2 Text Encoding and Feature Adaptation

We use a frozen LongCLIP text encoder (Zhang et al., 2025a) to process each input description and retain a 768-dimensional feature for each token. The encoder parameters remain fixed during generator training, and its output features are precomputed and cached.

Each token feature is then passed through a shared trainable adapter consisting of a linear projection from 768 to 512 dimensions, LayerNorm, a LeakyReLU activation with negative slope 0.2, and a second linear projection from 512 to 64 dimensions. The resulting 64-dimensional token features are used by both the global and token-level conditioning paths. The two paths share the adapter, which is trained jointly with the generative model.

Tokenized descriptions shorter than the specified length are padded, and an attention mask distinguishes valid tokens from padding. Padded positions are excluded from both global feature pooling and token-level attention.

B.3 Global Conditioning Path

We first average the adapted token features over valid positions identified by the attention mask, excluding padding tokens, to obtain a 64-dimensional global text vector. A linear layer without bias then projects this vector to the U-Net conditioning embedding dimension: 384 for all datasets.

The projected text vector is added to the embedding of the current noise level and passed through a SiLU activation to form a conditioning vector shared across the residual blocks. Within each block, a separate linear layer maps this vector to channel-wise scale and shift parameters. These parameters modulate the GroupNorm-normalized features after the first convolution and are shared across all spatial positions, allowing textual information to guide feature updates throughout the encoder, bottleneck, and decoder.

B.4 Token-Level Conditioning Path

We insert a cross-attention module between the two residual blocks at the U-Net bottleneck. The noisy time–frequency representation passes through the U-Net encoder and the first bottleneck residual block, producing a feature map with 192 channels. The cross-attention module applies GroupNorm to this feature map, flattens its spatial positions into a sequence, and uses a linear projection to construct the queries. The 64-dimensional text token features from the adapter are normalized using LayerNorm and projected into 192-dimensional keys and values through two separate linear layers. Cross-attention uses three heads, each with 64 dimensions, allowing each time–frequency position to aggregate relevant textual information.

The outputs of the three heads are concatenated, linearly projected back to 192 dimensions, reshaped to the original feature map dimensions, and added to the module input. Both the weights and bias of

this output projection are initialized to zero, so the module initially preserves its input features and gradually learns to adjust them based on the text conditions.

B.5 Diffusion Backbone

We use a two-dimensional U-Net (Ronneberger et al., 2015; Dhariwal & Nichol, 2021) with EDM preconditioning as the denoising network, taking the real and imaginary components of the time–frequency representation as two input channels. To accommodate downsampling and upsampling in the U-Net, we append zeros along the frequency and time axes, setting the input size to $2 \times 16 \times 48$ for Synth-U and ETTm, and $2 \times 16 \times 16$ for AirQuality, Traffic, and Electricity. The two channels store the real and imaginary components, while the remaining dimensions specify the height along the frequency axis and the width along the time axis. Zero padding preserves the original spectral coefficients and their positions.

The U-Net consists of an encoder, a bottleneck, and a decoder, each using residual blocks to extract and update features. The residual blocks use GroupNorm, SiLU activations, and 3×3 convolutions, with a dropout probability of 0.1. The encoder downsamples the feature map once, halving its frequency and time dimensions before processing it at the bottleneck. The decoder then upsamples it once to restore the original dimensions and receives features of the corresponding size from the encoder through skip connections to preserve local time–frequency details. The global conditioning path injects global textual information into the residual blocks throughout the encoder, bottleneck, and decoder, while the token-level conditioning path provides fine-grained textual information through the cross-attention module between the two bottleneck residual blocks.

B.6 Training Objective

We use the EDM weighted denoising loss with $\log \sigma \sim \mathcal{N}(-1.2, 1.2^2)$ and $\sigma_{\text{data}} = 0.5$. The loss weights for the original spectral region and the zero-padded region are set to 1 and 0.05, respectively.

We additionally use an STFT consistency loss to encourage agreement between the predicted spectrum and the STFT of its reconstructed time series. Specifically, we crop the zero-padded region from the prediction, invert the magnitude compression, and reconstruct the time series using iSTFT. We then reapply STFT and magnitude compression to the reconstructed series and compute the mean squared error between the resulting representation and the predicted spectrum. A numerical-stability constant of 10^{-9} is used in this consistency transformation. This loss is computed only over the valid spectral region and added to the denoising loss with a coefficient of 0.1.

B.7 Optimization and Sampling

We use the AdamW optimizer (Loshchilov & Hutter, 2019) with a batch size of 128 and a weight decay of 10^{-5}. The learning rate is 2×10^{-4} for Synth-U, ETTm, Traffic, and Electricity, and 1×10^{-4} for AirQuality. The main experiments use fixed optimizer-update budgets: 16,320 updates for Synth-U, ETTm, AirQuality, and Traffic, and 16,272 updates for Electricity, with three independent random seeds. We apply gradient clipping with a norm threshold of 1 and maintain an exponential moving average (EMA) of the model parameters.

For generation, we use the EMA weights from the final training checkpoint and the default Karras–Heun sampler of EDM (Karras et al., 2022) with 18 sampling steps.

C Experimental Settings

C.1 Datasets and Preprocessing

We use the released time-series–text pairs from VerbalTS for Synth-U and from TSFragment-600K for AirQuality, Traffic, and Electricity. For ETTm, we use the time-series–text dataset released with VerbalTS (Gu et al., 2025). We reorganize the data into chronological splits and remove unsuitable samples, retaining the original textual descriptions for the remaining series. The mixed-length generation experiment in Appendix A.1 uses Traffic sequences of lengths 24, 48, and 96 from TSFragment-600K.

C.2 Evaluation and Result Aggregation

Following the evaluation implementations of VerbalTS (Gu et al., 2025) and ConTSG-Bench (Lan et al., 2026), we evaluate generative models using pointwise lower-median aggregation. For the main TS–text alignment comparison, representation and backbone ablation, and STFT window-length sensitivity experiments, we generate ten time series per text description and aggregate them into a single sequence before computing CTTP, FID, and J-FTSD. These comparisons therefore assess performance under this aggregation protocol. For the conditioning-path ablation, we generate three time series per description and evaluate Atomic Satisfaction and Edited-Attribute Satisfaction on individual generated sequences following the evaluation protocol in Appendix D. For each experiment, we compute the evaluation results separately for each training seed and report the mean and sample standard deviation across the three training seeds.

C.3 Baseline Implementation Details

VerbalTS. We use the official implementation of VerbalTS (Gu et al., 2025), retaining its multi-scale generation architecture and text-conditioning mechanism. The model is trained for 500 epochs using Adam with a batch size of 512, a learning rate of 10^{-3}, and a weight decay of 10^{-6}. Generation uses 50-step DDIM sampling (Song et al., 2021).

T2S. We adapt the ConTSG-Bench implementation (Lan et al., 2026) of T2S (Ge et al., 2025), retaining its autoencoder and latent flow-matching architecture. Training comprises 50 epochs of autoencoder pretraining and 450 epochs of text-conditioned training, using AdamW with a batch size of 256, a learning rate of 10^{-4}, and a weight decay of 10^{-4}.

BRIDGE. We adapt the ConTSG-Bench implementation (Lan et al., 2026) of BRIDGE (Li et al., 2025). During generation, prototypes are drawn from a fixed pool of training examples and reused across sampling batches. The model is trained for 500 epochs with a batch size of 256, a learning rate of 10^{-3}, a weight decay of 10^{-4}, and a cosine learning-rate schedule. Generation uses 50-step DDIM sampling.

DiffuSETS. We adapt the ConTSG-Bench implementation (Lan et al., 2026) of DiffuSETS (Lai et al., 2025) to the time series and natural-language descriptions used in our experiments. Training comprises 50 epochs of variational autoencoder pretraining and 450 epochs of conditional diffusion training, using AdamW with a batch size of 256, a learning rate of 10^{-4}, and a weight decay of 10^{-4}. Generation uses 50-step DDIM sampling.

Retrieval. This baseline retrieves training descriptions with the highest cosine similarity to the input description and returns their associated time series. It does not require training a generative model.

C.4 Ablation Configurations

1D U-Net. We use a one-dimensional U-Net that performs diffusion denoising directly on time-domain sequences as a comparison to the two-dimensional STFT representation. This model retains the same text encoder, feature adapter, global and token-level conditioning paths, and EDM diffusion framework. Two-dimensional convolutions and resampling operations are replaced with one-dimensional operations along the temporal axis, with each 3×3 convolution replaced by a one-dimensional convolution of kernel size 9. Both the input and output are single-channel sequences. The model retains the U-Net's channel configuration, residual blocks, and skip connections, yielding a parameter count close to that of the corresponding two-dimensional U-Net. Training uses the time-domain denoising loss without STFT transformation, spectral magnitude compression, or STFT consistency loss.

Delay Representation. We adopt the delay embedding used in ImagenTime (Naiman et al., 2024), arranging consecutive segments of a time series as columns of a two-dimensional matrix. Each column contains 16 time points, and the starting positions of adjacent columns are separated by 8 time points. We zero-pad the final segment if it contains fewer than 16 points and pad the matrix to meet the network's input size requirements. After generation, we place the columns back at their corresponding temporal positions in order, using values from the later column at overlapping positions to reconstruct the time series.

DiT Backbone. We replace the U-Net with a DiT (Peebles & Xie, 2023) while retaining the same text encoder, feature adapter, and EDM diffusion framework. The two-dimensional input is divided into 2×2 patches, linearly embedded, and combined with two-dimensional sinusoidal positional encodings before being processed by six Transformer blocks. We set the hidden dimension to 256, the number of attention heads to 8, and the feed-forward expansion ratio to 3. The global text condition is added to the noise-level embedding and modulates each block through adaptive LayerNorm. Token-level text conditions are injected through a cross-attention module after the third block. The output tokens are projected and reassembled into a two-dimensional representation with the same dimensions as the input. This configuration gives the DiT and the corresponding U-Net comparable parameter counts.

D Fine-Grained Instruction-Following Evaluation

We evaluate fine-grained instruction following using full validation captions and pairs of captions that differ in one textual requirement. The following protocol specifies how the pairs are constructed, how the generated outputs are scored, and how the attribute evaluators are built and checked.

D.1 Construction of the Textual-Editing Evaluation Set

We evaluate full-caption instruction following on all 4,000 Synth-U and 1,603 ETTm validation captions. To evaluate responses to targeted changes, we construct paired captions by replacing the smallest text span needed to change one requirement, while preserving the remaining text and the number of stated requirements. The editable attributes and pair counts are listed in Table D.1. The transitions shown are examples; each category includes multiple transitions supported by the data.

Table D.1: Editable attributes and numbers of caption pairs. Transitions are examples of the supported edits.

Dataset	Edited attribute	Example transition	Caption pairs
Synth-U	Trend type	Exponential → logistic	210
Synth-U	Trend direction	Up → down	210
Synth-U	Seasonal cycle count	0 → 2 cycles	210
Synth-U	High-frequency period label	64 → 16	210
Synth-U	Local event type	Sag → double peaks	210
Synth-U	Local event position	End → beginning	210
ETTm	Trend strength	Slow upward trend → sharp upward trend	210
ETTm	Trend direction/state	Slowly rise → slowly drop	210
ETTm	Local peak count	1 → 2 peaks	169
ETTm	Caption-defined seasonal index	"around 1 pi" → "around 2 pi"	116
ETTm	Skewness	Symmetrical → negative	210
ETTm	Kurtosis	Normal → low	210

Note: The seasonal index follows the dataset's annotation convention and does not denote a period in physical time units.

We sample caption pairs across groups defined by variable, location, and attribute transition, selecting up to 210 pairs per edit category. Each source caption is used at most once within a category.

D.2 Paired Sampling and Satisfaction Metrics

TextPlot, VerbalTS, and T2S are evaluated on the same original descriptions and edited pairs using three training seeds (20260516–20260518). For each seed, we generate three samples per descrip-

tion. Within each model, corresponding samples from the original and edited descriptions use the same sampling seed. The conditioning-path ablations follow the same protocol.

Atomic Satisfaction (AS) measures the fraction of requirements in a description that a generated time series satisfies. Let $\mathcal{A}(\boldsymbol{c})$ be the set of explicitly stated requirements in description $\boldsymbol{c}$, and let $s(\boldsymbol{x}, a)$ equal 1 if the evaluator judges that generated series $\boldsymbol{x}$ satisfies requirement a, and 0 otherwise. The score for one generated series is

$$\mathrm{AS}(\boldsymbol{x}, \boldsymbol{c}) = \frac{100}{|\mathcal{A}(\boldsymbol{c})|} \sum_{a \in \mathcal{A}(\boldsymbol{c})} s(\boldsymbol{x}, a).$$

For example, satisfying four of five requirements gives an AS of 80%. For each training seed, we average these scores equally across descriptions and generated samples. Only explicitly stated requirements are evaluated, and unrecognized predictions count as unsatisfied.

Edited-Attribute Satisfaction (EAS) measures whether a generated series satisfies the requirements associated with the attribute being edited. Let e denote that attribute and $\mathcal{A}_e(\boldsymbol{c}) \subseteq \mathcal{A}(\boldsymbol{c})$ contain its requirements in description $\boldsymbol{c}$. The score for one generated series is

$$\mathrm{EAS}(\boldsymbol{x}, \boldsymbol{c}; e) = \frac{100}{|\mathcal{A}_e(\boldsymbol{c})|} \sum_{a \in \mathcal{A}_e(\boldsymbol{c})} s(\boldsymbol{x}, a).$$

For example, when an upward trend is changed to a downward trend, we check for an upward trend in samples generated from the original description and a downward trend in samples generated from the edited description. For each training seed, we average the scores equally over both sides of each pair, all edited pairs, and generated samples.

Both metrics are computed on individual generated sequences. We report the mean and sample standard deviation across the three training seeds.

D.3 Construction and Validation of the Attribute Evaluators

Following the classifier-based evaluation approach in ConTSG-Bench (Lan et al., 2026), we construct dataset-specific attribute evaluators to assess fine-grained controllability. Each evaluator receives a time series and predicts its attributes, which are then compared with the requirements in the corresponding description.

The evaluators are built using reference time series and attribute labels extracted from their descriptions. For Synth-U, we train tree-based classifiers on trend, frequency, and local-shape features to recognize trend type and direction, periodicity, high-frequency components, and local events. Local events are identified separately in the beginning, middle, and end of the series. For ETTm, we combine trained trend and period classifiers with numerical rules for peak counts, skewness, and kurtosis, using a training-calibrated mapping for trend direction and strength expressions. The learned components are fitted on training data, checked against validation annotations, and frozen before scoring generated outputs. On reference validation sequences, the evaluators recover 99.41% and 99.80% of the annotated requirements on average for Synth-U and ETTm, respectively.

E Supplementary Results for TS–Text Alignment

The main text reports CTTP, FID, and J-FTSD as the primary metrics. Following ConTSG-Bench (Lan et al., 2026), we additionally report metrics assessing generation fidelity, condition adherence, and errors between generated and reference time series. Their definitions are summarized below.

- **Auto-Correlation Difference (ACD)** evaluates the preservation of temporal dependencies by comparing the autocorrelation profiles of real and generated time series.
- **Skewness Difference (SD)** measures the difference in skewness between real and generated data distributions, reflecting differences in distributional asymmetry.
- **Kurtosis Difference (KD)** measures the difference in kurtosis between real and generated data distributions, reflecting differences in extreme deviations from the mean.
- **Marginal Distribution Difference (MDD)** measures how closely the marginal distributions of real and generated time series agree, using empirical histograms to approximate their probability densities.

- **Precision and Recall (PRDC-Precision and PRDC-Recall)** evaluate generation fidelity in time-series embedding space using nearest-neighbor neighborhoods (Kynkäänniemi et al., 2019; Naeem et al., 2020). Precision measures the fraction of generated points within the real-data manifold, while recall measures the fraction of real points within the generated-data manifold. **PRDC-F1** is their harmonic mean.
- **Joint Precision and Recall (Joint PRDC-Precision and Joint PRDC-Recall)** extend precision and recall to the joint feature space of time series and conditions to evaluate condition adherence. **Joint PRDC-F1** is their harmonic mean.
- **Dynamic Time Warping (DTW)** (Sakoe & Chiba, 1978) measures the alignment distance between a generated time series and the ground-truth reference associated with the same condition.
- **Continuous Ranked Probability Score (CRPS)** (Gneiting & Raftery, 2007) generalizes mean absolute error to probabilistic predictions. It compares the empirical cumulative distribution of generated samples with the reference observation and evaluates both calibration and sharpness.
- **Euclidean Distance (ED)** measures the Euclidean distance between generated and reference time series.
- **Weighted Absolute Percentage Error (WAPE)** is the sum of absolute errors between generated and reference values divided by the sum of absolute reference values, multiplied by 100.

Figures E.2 and E.3 present results for 14 supplementary metrics. Bars and error bars show means and sample standard deviations across three training seeds, respectively. Dark outlines indicate the best mean or point estimate within each dataset.

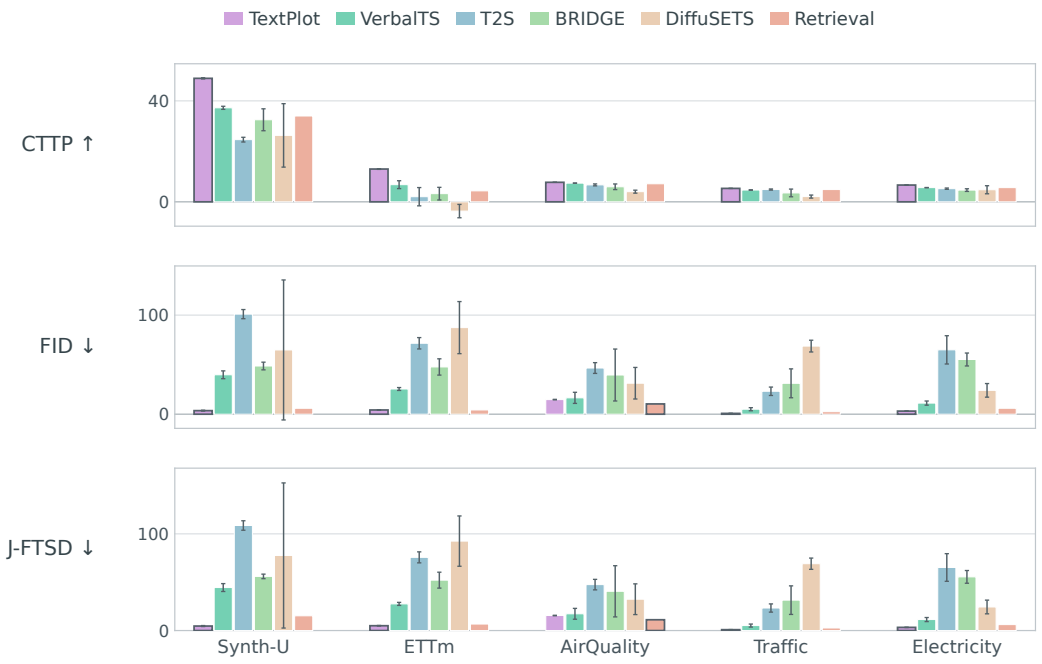


Figure E.1: Primary metrics from Table 1. Error bars show ± one sample standard deviation across three training seeds. Retrieval is a single estimate without error bars. Dark outlines mark the best result within each dataset.



Figure E.2: Supplementary statistical and reference-based error metrics on five datasets. Lower values are better. Bars show means, error bars show sample standard deviations, and dark outlines mark the best reported means.

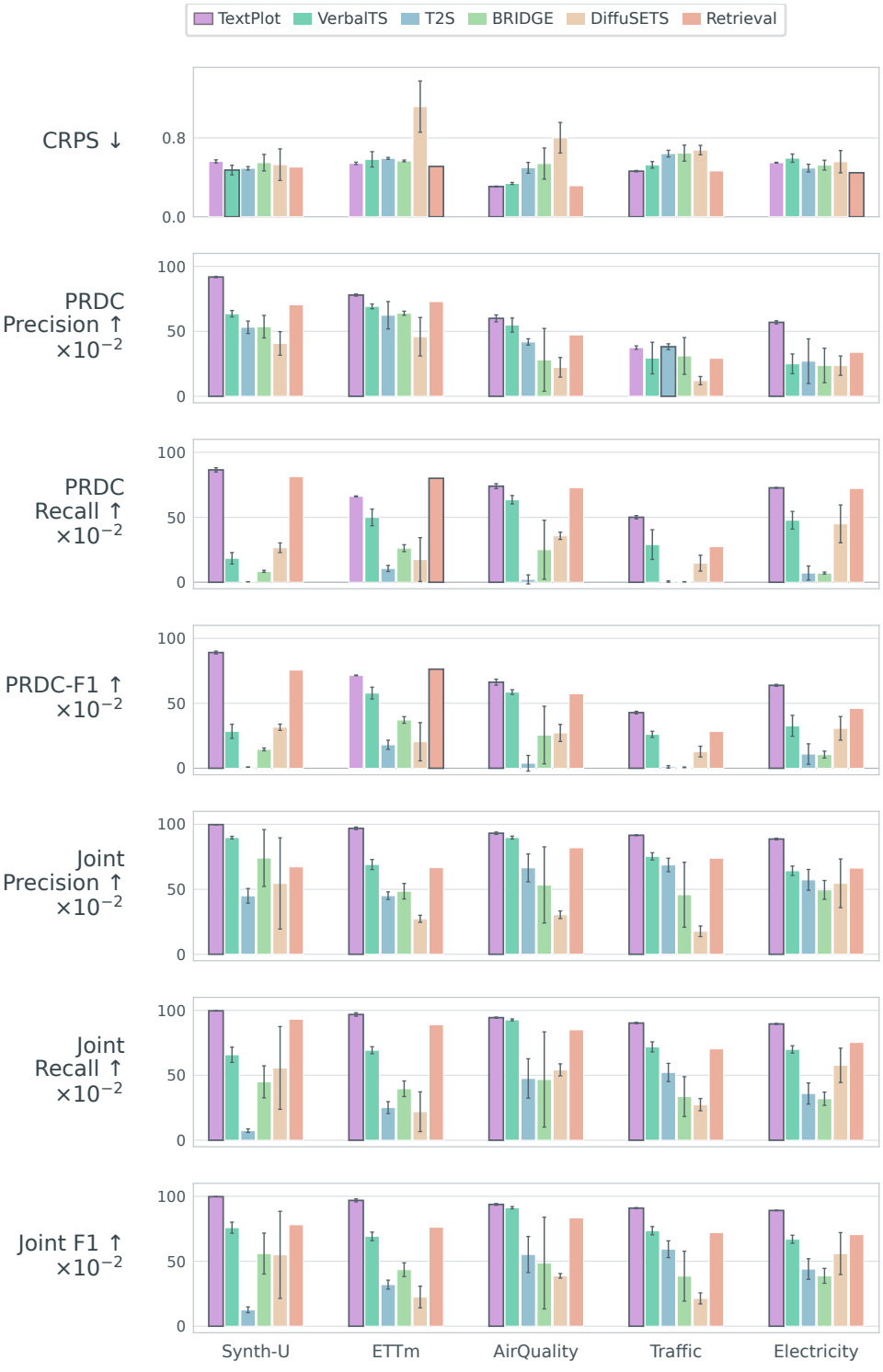


Figure E.3: Continuation of Figure E.2.

F Qualitative Examples of Generation Quality and Textual Control

We compare generation examples from TextPlot, VerbalTS, T2S, BRIDGE, and DiffuSETS, and examine responses to textual changes using TextPlot, VerbalTS, and T2S.

F.1 Generation Quality

Figures F.1 and F.2 place the reference sequence and the five model outputs side by side under the same natural language description. These selected examples illustrate broad temporal shapes and local details, including the early double peak in Synth-U, the middle-to-late decline and recovery in ETTm, and the fluctuations in the three TSFragment datasets.

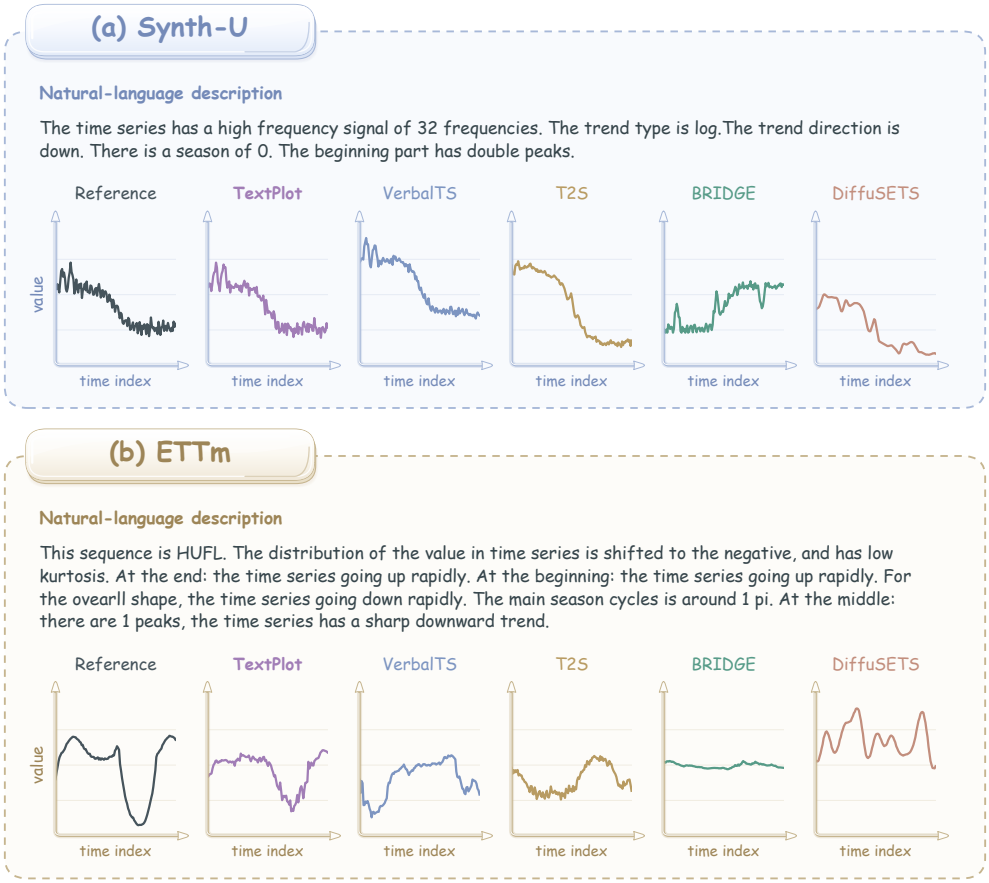


Figure F.1: Selected generation examples on Synth-U and ETTm. Each row presents the original natural language description, its reference sequence, and individual samples from TextPlot, VerbalTS, T2S, BRIDGE, and DiffuSETS. All panels within a row share the same axis limits. Values are shown in the normalized space used by the models.

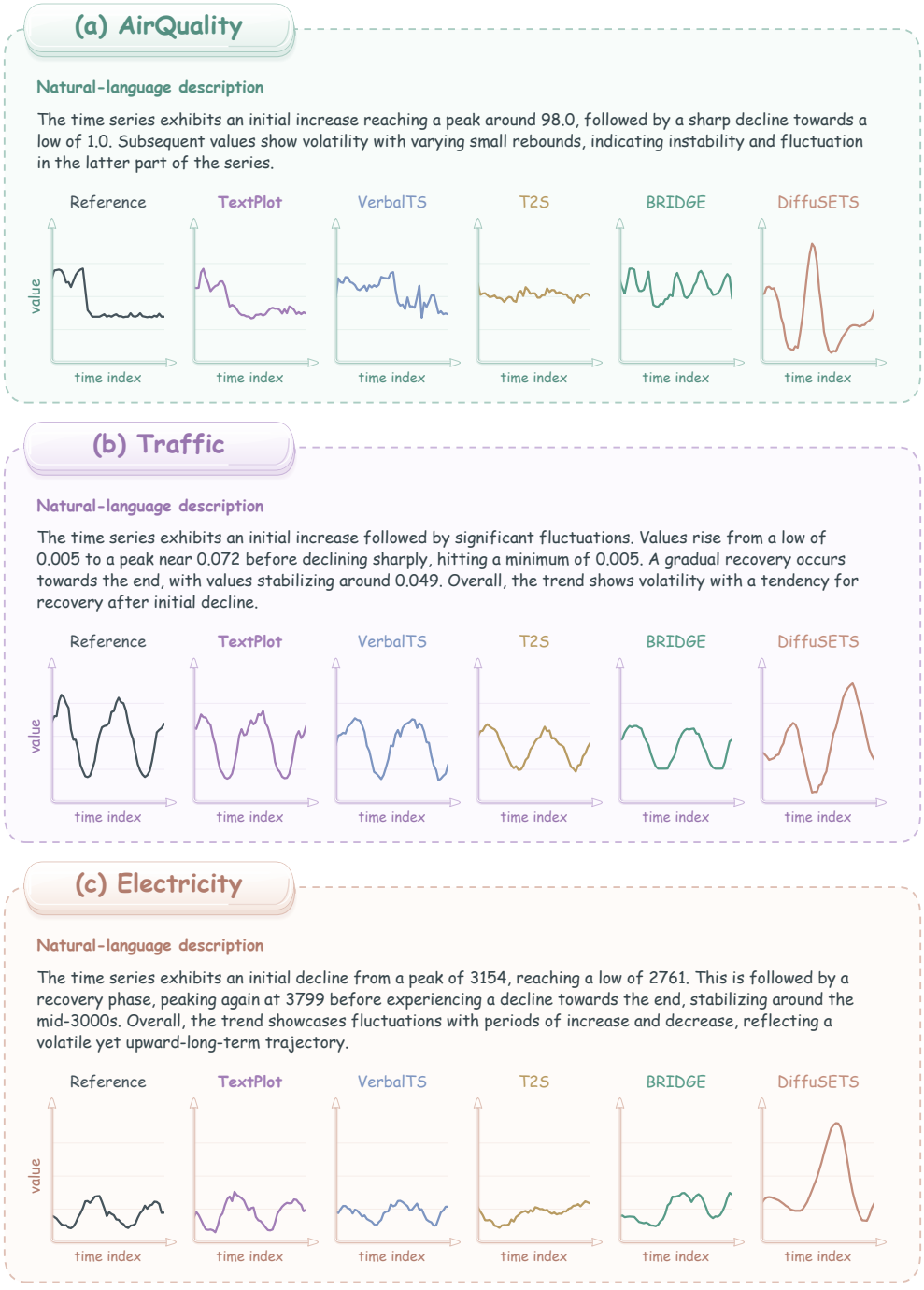


Figure F.2: Selected generation examples on AirQuality, Traffic, and Electricity, using the same layout as Figure F.1. Natural language descriptions retain the original dataset wording, including any values expressed in the source units; the plotted curves use the models' normalized values.

F.2 Responses to Single-Condition Textual Changes

Figures F.3 and F.4 illustrate how generated time series respond to changes in a single textual requirement. For each model, the original and modified descriptions use the same sampling seed.

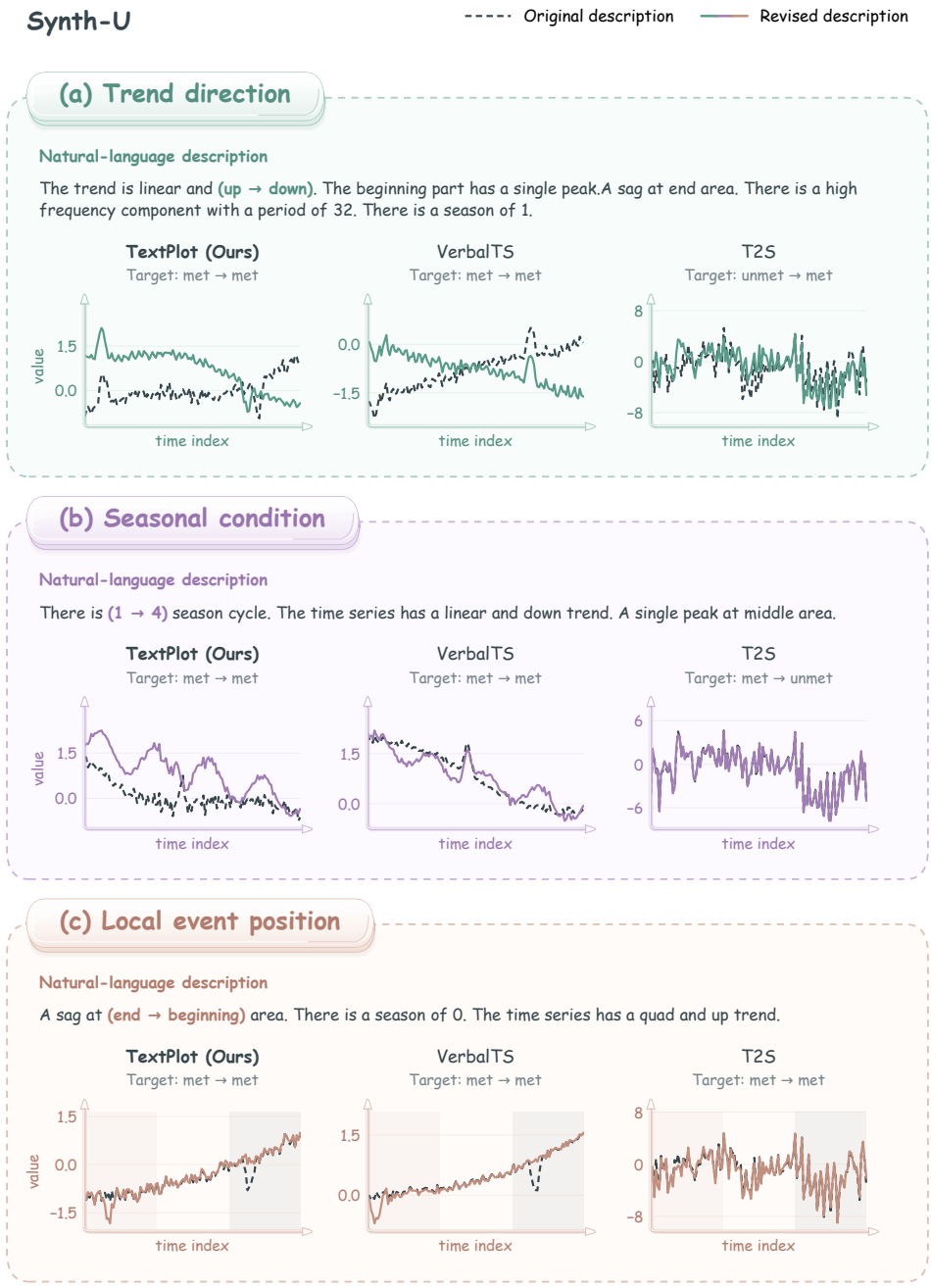


Figure F.3: Responses to single-condition changes in natural language descriptions on Synth-U: (a) trend direction, (b) seasonal cycle count, and (c) sag position. Gray dashed and colored solid curves show samples generated from the original and revised descriptions, respectively. Shading marks the specified event locations. Target labels indicate whether the evaluator judges the original and revised requirements to be satisfied, in that order. Vertical scales vary across models.

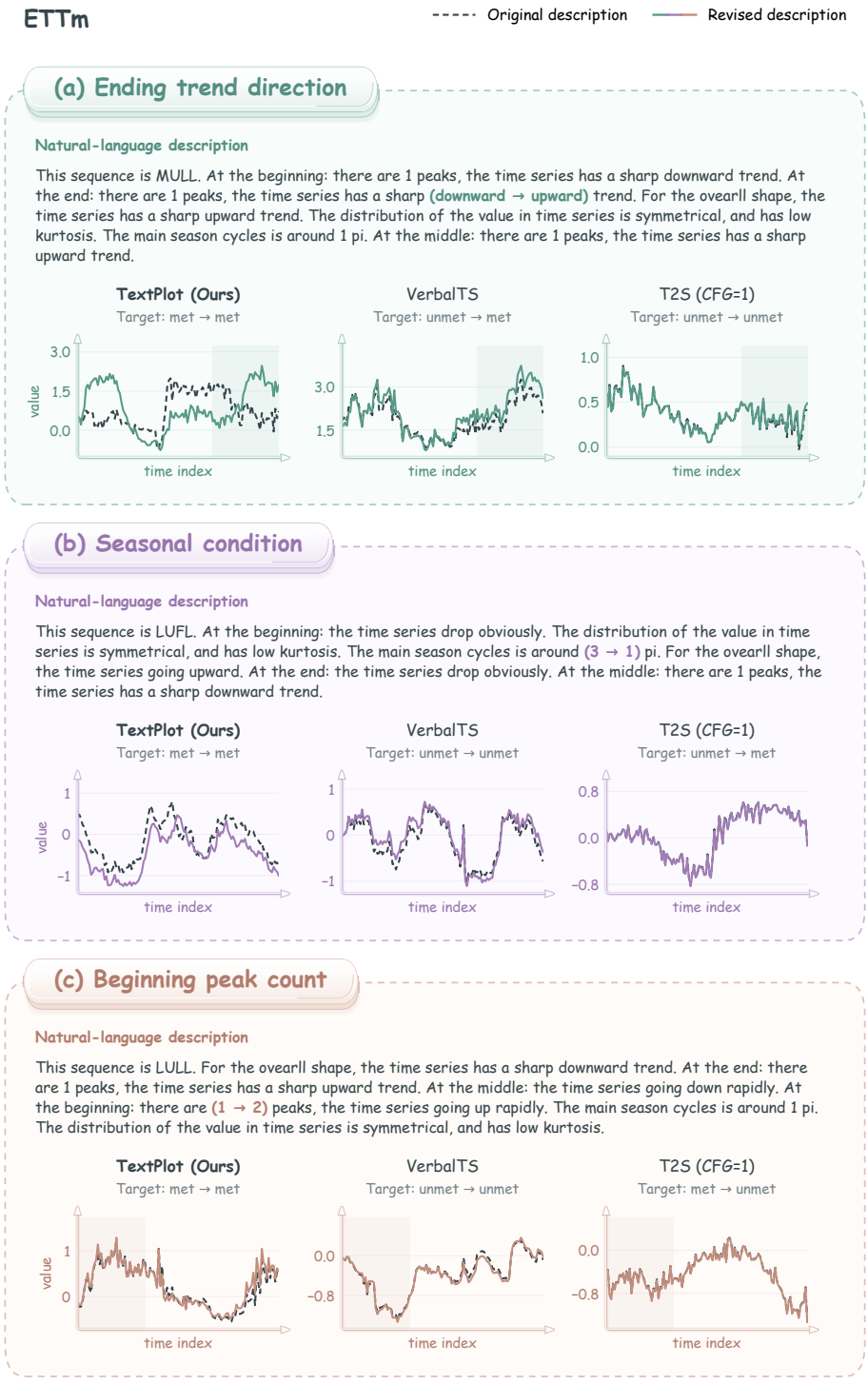


Figure F.4: Responses to single-condition changes in natural language descriptions on ETTm: (a) ending trend direction, (b) seasonal index, and (c) beginning peak count. Shading marks the specified temporal segments. Line styles, target labels, and axis conventions follow Figure F.3.