Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ARTICLE TYPE

A Temporally Disentangled Contrastive Diffusion Model for Spatiotemporal Imputation

Yakun Chen[1,2] | Kaize Shi[2,3] | Zhangkai Wu[2] | Juan Chen[4] | Xianzhi Wang*[2] | Julian McAuley[5] | Guandong Xu[1,2] | Shui Yu[2]

[1]Centre for Learning, Teaching and Technology, The Education University of Hong Kong, Hong Kong, China

[2]School of Computer Science, University of Technology Sydney, NSW, Australia

[3]School of Mathematics, Physics and Computing, University of Southern Queensland, QLD, Australia

[4]Sophos Ltd, NSW, Australia

[5]Department of Computer Science and Engineering, University of California San Diego, CA, United States

Correspondence

* Corresponding author Xianzhi Wang

Email: xianzhi.wang@uts.edu.au

Abstract

The analysis of spatiotemporal data is essential across many fields, such as transportation, meteorology, and healthcare. Data gathered in practical applications often suffers from incompleteness due to device failures and network disruptions. Spatiotemporal imputation targets the estimation of missing observations by exploiting intrinsic spatial–temporal dependencies. While traditional statistical and machine-learning methods depend on restrictive distributional assumptions, graph- or recurrent-based models accumulate errors through iterative propagation. Diffusion probabilistic models mitigate these issues by sampling directly from a learned data prior instead of recycling past imputations. However, existing conditional diffusion variants still converge toward overly similar reconstructions, obscuring the genuine uncertainty and heterogeneity of real-world traffic, environmental, or clinical streams. Preserving—and faithfully quantifying—this intrinsic diversity is crucial for reliable forecasting and downstream decision-making. We propose C^{2}TSD, a conditional diffusion framework that integrates disentangled temporal representations and contrastive learning to improve generalizability in spatiotemporal imputation. Specifically, the approach uses disentangled temporal representations as conditional information to guide the reverse process. We also enhance the final loss using a contrastive learning strategy to improve representation quality, mitigating the impact of data missing completely at random (MCAR) and noise on learned features. Through comprehensive experiments using three distinct real-world datasets, C^{2}TSD has competitive results compared to leading-edge baselines.

KEYWORDS

Spatiotemporal Imputation, Diffusion Model, Contrastive Learning, Temporal Disentanglement

1 | INTRODUCTION

Spatiotemporal data represent a special type of multivariate time series (MTS) data encompassing both location and time. Unlike other data types, spatiotemporal data are chronologically ordered and autocorrelated, typically exhibiting trends, seasonality, and noise[1]. These data are crucial for generating insight and supporting decision-making in various domains, such as economics, transportation, healthcare, and meteorology[2,3,4,5]. Existing methods for handling spatiotemporal data (e.g., forecasting[6], classification[7], and anomaly detection[8]) typically depend on a complete spatiotemporal dataset to maintain performance. However, real-world spatiotemporal data often contain missing values due to sensor malfunctions, packet loss, poor data integrity management, or human error[9]. These missing values can significantly affect the performance of specific tasks, necessitating effective imputation methods to address these gaps and support downstream tasks.

Spatiotemporal imputation is inherently challenging as it requires capturing spatial and temporal dependencies simultaneously[10]. Traditional statistical and machine learning-based imputation methods require labour-intensive feature engineering. Furthermore, they often make strong assumptions about the data, which may not always hold in real-world scenarios, thus limiting their performance in real-world problems. Although Recurrent Neural Network (RNN)-based models, including those

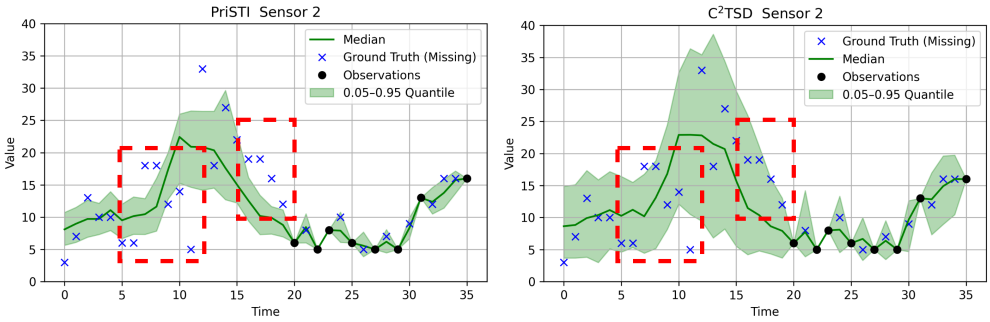


FIGURE 1 Comparison of imputation quality on a challenging long-gap segment of missing values (Sensor 2, Air36 dataset). Black dots are the observed values; blue cross marks are the hidden ground-truth values inside the gap (used only for evaluation). The green curve is the median of 100 generated samples, and the translucent green band denotes the 5th–95th percentiles. The red dashed rectangle highlights a window where approximately 40% of the series is missing. The left one is the PriSTI baseline [14]. The right one is our proposed C^{2}TSD. PriSTI produces a narrow, nearly unimodal credible band that misses several peaks of the true trajectory, illustrating the sample-collapse problem. In contrast, C²TSD, aided by time-series decomposition and contrastive regularisation, yields a wider, adaptive uncertainty band that fully encloses the ground truth and exhibits richer yet plausible variability.

based on Gated Recurrent Units (GRUs) [9], bidirectional structures [11], and the integration of Graph Neural Networks (GNNs) and RNNs [10], have shown advantages in capturing complicated relationships, they still suffer the error accumulation issue due to recurrent structures [12,13], i.e., they heavily rely on previously imputed (and potentially inaccurate) values to infer more missing values, leading to degraded imputation performance.

Diffusion models, especially Denoising Diffusion Probabilistic Models (DDPMs) [15], have emerged as powerful generative approaches capable of modelling complex distributions through iterative denoising processes. These models exhibit superior performance in capturing intricate spatiotemporal dependencies, making them particularly effective for addressing challenges in complex real-world scenarios [16,14]. Recent studies have introduced conditional information into DDPM frameworks, known as Conditional DDPMs, enabling the generation conditioned on observed data to improve predictive accuracy and controllability [16,17]. However, despite their advantages, conditional DDPMs exhibit a critical limitation in spatiotemporal imputation: they tend to produce overly similar samples under strong conditional constraints [18,14]. This homogeneity is particularly problematic because missing spatiotemporal segments often possess multiple plausible underlying realities (e.g., varying traffic conditions or patient vital signs), and such 'average-case' imputations consequently obscure the true data's inherent uncertainty and diversity. This issue often arises because conditioning on raw, entangled data components (trend, seasonality, noise) causes models to excessively replicate typical patterns while neglecting irregular but important variations like anomalous events or rare scenarios [18,19]. Ultimately, this lack of imputation diversity is detrimental, as it can mask crucial data variability, lead to the missed detection of critical events, and result in flawed decision-making. Figure 1 provides an illustrative example of this phenomenon. On a long-gap segment of missing values from the Air36 dataset, we compare the imputation results of the PriSTI [14] baseline and our proposed C^{2}TSD. Following visualization conventions, observed values are marked with black circles, while hidden ground-truth values (used only for evaluation) are denoted by blue crosses. The green line and shaded region represent the median and 90% credible interval, respectively, across 100 imputed samples. As shown, PriSTI produces a narrow credible band that fails to capture several peaks in the ground truth—highlighting the lack of sample diversity.

To address limited diversity in spatiotemporal imputation, we leverage two established research lines. Firstly, time series disentanglement into trend, seasonal, and residual components enhances modelling performance and robustness [20,21]. This decomposition isolates fundamental temporal structures (trend, seasonal), improving predictive accuracy and generalization [22], validated by models like Autoformer [23] and DeepGLO [24]. Decomposing trend and seasonality provides a stable foundation for imputing missing data by allowing generative models to create diverse and realistic residual variations, avoiding over-smoothing. The forecasting success of decomposition with contrastive learning [25] suggests its applicability to imputation. Furthermore, contrastive learning has emerged as an effective approach to enhance the representation quality, diversity, and generalization capacity of generative models [26,27]. Recent studies have demonstrated that contrastive learning encourages models to capture differences between similar samples, thus promoting better generalization and reducing the tendency toward mode collapse in

generative models[26,28]. Specifically, integrating contrastive objectives into generative frameworks such as GANs and diffusion models has shown significant improvements in generated sample diversity and model stability[29]. Motivated by these insights, we propose a novel Conditional diffusion framework named **C^{2}TSD**, a **C**onditional diffusion framework integrating **C**ontrastive learning and **T**rend-**S**eason **D**isentanglement for spatiotemporal imputation. Specifically, the design of conditional information helps guide the generation of high-quality data that conforms to preset conditions. To incorporate spatiotemporal dependencies in the conditional information, we first use trend and seasonal extraction to learn the disentangled temporal dependencies, followed by a GNN encoder to aggregate the geographical neighbour information to learn the spatial dependencies. Besides, we use contrastive learning to improve the stability and robustness of conditional DDPM by mitigating noise and reducing the impact of missing data patterns. Contrastive learning helps ensure similar samples are closer and different samples are farther apart, thus enhancing the quality of generated representations and balancing the trade-off between diversity and fidelity. In summary, this paper makes the following main contributions:

- We propose C^{2}TSD, a contrastive diffusion framework for spatiotemporal imputation, leveraging conditional information by utilizing disentangled temporal representations and spatial relationships.
- We introduce trend-season disentanglement to enhance the robustness of temporal relationship learning, establishing a stable structural context from observed trend and seasonal components to promote diverse, realistic imputations.
- We develop a contrastive learning strategy to facilitate learning spatiotemporal dependencies, enhancing our model's stability and generalizability.
- We conducted extensive experiments on three real-world datasets from the meteorology and transportation fields, demonstrating that C^{2}TSD has competitive performance with baselines.

2 | BACKGROUND

2.1 | Spatiotemporal Imputation

Spatiotemporal imputation focuses on estimating absent values within historical records by leveraging both spatial and temporal relationships. We represent the input spatiotemporal dataset as $\mathbf{X} = \{x_{1:N,1:L}\} \in \mathbb{R}^{N\times L}$, where N denotes the number of variables and L indicates the sequence length. Additionally, we introduce an observation mask $\mathbf{M} = \{m_{1:N,1:L}\} \in \{0,1\}^{N\times L}$, which identifies the missing locations in the dataset. Specifically, $m_{n,l} = 0$ if $x_{n,l}$ is absent, and $m_{n,l} = 1$ if $x_{n,l}$ is present. Consequently, spatiotemporal imputation aims to predict values that closely approximate the actual ground truth, thereby filling the gaps in $\mathbf{X}$ and producing a complete dataset for subsequent analysis and tasks.

2.2 | Graph Formulation

Consider a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{A}\}$, where $\mathcal{V}$ and $\mathcal{E}$ denote the nodes and edges sets, respectively, with N being the total number of nodes. The graph is characterized by the weighted adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$, which encapsulates the connections and weights between the nodes. In spatiotemporal imputation, we use a graph structure to represent the spatial relationship between different time series.

2.3 | DDPMs

DDPMs[15] are generative models designed to learn a distribution $p_\theta(\mathbf{x}_0)$ to closely match a given data distribution $q(\mathbf{x}_0)$. Consider $\mathbf{x}_t$ for $t = 1, ..., T$ as a series of latent variables within the same sample space as $\mathbf{x}_0$. DDPMs are structured as latent variable models consisting of two main processes: forward and reverse.

Forward Process. The forward process is a Markov chain, systematically introducing Gaussian noise to the initial data $\mathbf{x}_0$ based on a specified variance schedule $\beta_1, ..., \beta_T$:

$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \tag{1}$$

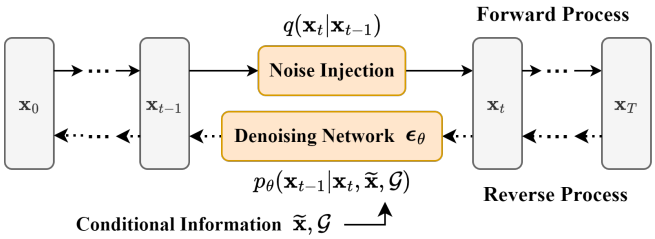


FIGURE 2 Pipeline of Conditional DDPM for Spatiotemporal Imputation. Our framework C^{2}TSD uses a trained denoising network $\boldsymbol{\epsilon}_\theta$ to sample $\mathbf{x}_{t-1}$ step by step in the reverse process, under the guidance of the conditional information generated from the linear interpolation $\widetilde{\mathbf{x}}$ and the graph structure $\mathcal{G}$.

where $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ is a Gaussian distribution and defined as:

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) := \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right), \tag{2}$$

and $\beta_t \in (0,1)$ is noise level at forward step t. The forward process admits sampling $\mathbf{x}_t$ at any step t in closed form:

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\overline{\alpha}_t}\mathbf{x}_0, (1-\overline{\alpha}_t)\mathbf{I}\right) \tag{3}$$

where $\alpha_t := 1 - \beta_t$ and $\overline{\alpha}_t = \prod_{i=1}^{t} \alpha_i$. Then $\mathbf{x}_t$ can be expressed as $\mathbf{x}_t = \sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Reverse Process. The reverse process aims to denoise $\mathbf{x}_T$ to reconstruct $\mathbf{x}_0$. It is formulated as a Markov chain utilizing learned Gaussian transitions, beginning with $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$:

$$\begin{aligned} p_\theta(\mathbf{x}_{0:T}) &:= p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t), \\ p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) &:= \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\sigma}_\theta(\mathbf{x}_t, t)\right). \end{aligned} \tag{4}$$

Aligned with [15], we set $\boldsymbol{\sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathbf{I}$ and use the same parameterization as $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$:

$$\begin{aligned} \boldsymbol{\mu}_\theta(\mathbf{x}_t, t) &= \frac{1}{\sqrt{\overline{\alpha}_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right), \\ \sigma_t^2 &= \frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}\beta_t, \end{aligned} \tag{5}$$

where $\boldsymbol{\epsilon}_\theta$ represents a trainable denoising network that determines the amount of noise to remove at each reverse process step. The reverse process is trained by solving the optimization problem below:

$$\min_\theta \mathcal{L}(\theta) := \min_\theta \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right\|^2\right]. \tag{6}$$

During the training phase, $\mathbf{x}_0$ is known. Recall that in the forward process, $\mathbf{x}_t = \sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}$. Thus, the training objective for unconditional generation can be expressed as:

$$\min_\theta \mathcal{L}(\theta) := \min_\theta \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}, t}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]. \tag{7}$$

Algorithm 1 Training process

Input: The incomplete spatiotemporal data $\mathbf{x}$, the graph structure $\mathcal{G}$, the number of iteration N, the number of diffusion steps T, variance schedule $\{\beta_1, ..., \beta_T\}$.
Output: Trained denoising network $\boldsymbol{\epsilon}_\theta$.
for $i = 1$ **to** N **do**
  $\mathbf{x}_0 \leftarrow \text{Mask}(\mathbf{x})$;
  $\widetilde{\mathbf{x}} \leftarrow \text{Interpolate}(\mathbf{x}_0)$;
  Sample $t \sim \text{Uniform}(\{1, \dots, T\})$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
  $\mathbf{x}^t \leftarrow \sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}$;
  Update the gradient $\nabla_\theta \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G})\|_2^2$.
end for

2.4 | Conditional DDPM for Spatiotemporal Imputation

To tailor the original unconditional DDPM for our task, we modify the reverse process in Eq. (4) and (5) into a conditional version, illustrated in Fig. 2. We conditioned the reverse process on the linear interpolation [‡] of the incomplete observed values $\widetilde{\mathbf{x}}$ and graph $\mathcal{G}$. Thus, the conditioned reverse process can be formulated as follows:

$$\begin{aligned} p_\theta\left(\mathbf{x}_{0:T} \mid \widetilde{\mathbf{x}}, \mathcal{G}\right) &= p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \widetilde{\mathbf{x}}, \mathcal{G}\right), \\ p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \widetilde{\mathbf{x}}, \mathcal{G}\right) &= \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta\left(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G}\right), \sigma_t^2 \mathbf{I}\right), \\ \boldsymbol{\mu}_\theta\left(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G}\right) &= \frac{1}{\sqrt{\overline{\alpha}_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G}\right)\right). \end{aligned} \tag{8}$$

Additionally, the training objective in Eq. (6) can be reformulated in a conditional version:

$$\min_\theta \mathcal{L}(\theta) := \min_\theta \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}, t}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G})\right\|^2\right]. \tag{9}$$

2.4.1 | Training Process

Since the true values of real-world missing data are unknown, we artificially inject missing patterns for the model evaluation. During the training process, we randomly mask some positions of input $\mathbf{x}$ to generate an imputation target dataset $\mathbf{x}_0$ and a corresponding mask matrix $\mathbf{m}$ to denote the injected missing value locations. The remaining observations serve as conditional information for the imputation process. Following [10], we apply point and block mask strategies to generate imputation targets with different missing patterns. After we get the imputation target $\mathbf{x}_0$, we use a linear interpolation method to generate interpolated conditional $\widetilde{x}$. During each training pass, we randomly select the diffusion step t and generate standard Gaussian noise ϵ to compute the noisy imputation target $\mathbf{x}_t$. Subsequently, we update the gradient of the training objective as defined in Eq. (9). Details of the training process of our proposed framework are illustrated in Algorithm 1.

2.4.2 | Imputation Process

In the imputation process, we utilize the observed mask $\mathbf{m}$ and the incomplete spatiotemporal data $\mathbf{x}$ from the test dataset. Using $\mathbf{x}$, we first compute the interpolated conditional data $\widetilde{\mathbf{x}}$. The trained denoising network $\boldsymbol{\epsilon}_\theta$ is then employed to iteratively sample $\mathbf{x}_{t-1}$ according to Eq. (8), guided by $\widetilde{\mathbf{x}}$ and $\mathcal{G}$. Details of the imputation process of our proposed framework can be found in Algorithm 2.

[‡] We calculate the missing values by the linear interpolation of the nearest observed time points. This is implemented by https://github.com/patrick-kidger/torchcde.

Algorithm 2 Imputation (Inference) Process

Input: The input incomplete spatiotemporal data $\mathbf{x}$, the graph structure $\mathcal{G}$, the number of diffusion steps T, the trained denoising network $\boldsymbol{\epsilon}_\theta$.
Output: Missing values of the imputation target $\mathbf{x}_0$.
$\widetilde{\mathbf{x}} \leftarrow \texttt{Interpolate}(\mathbf{x})$;
Set $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
for $t = T$ **to** 1 **do**
  $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G}) \leftarrow \frac{1}{\sqrt{\overline{\alpha}_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G}) \right)$;
  $\mathbf{x}_{t-1} \leftarrow \mathcal{N}(\boldsymbol{\mu}_\theta(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G}), \sigma_t^2 \mathbf{I})$.
end for

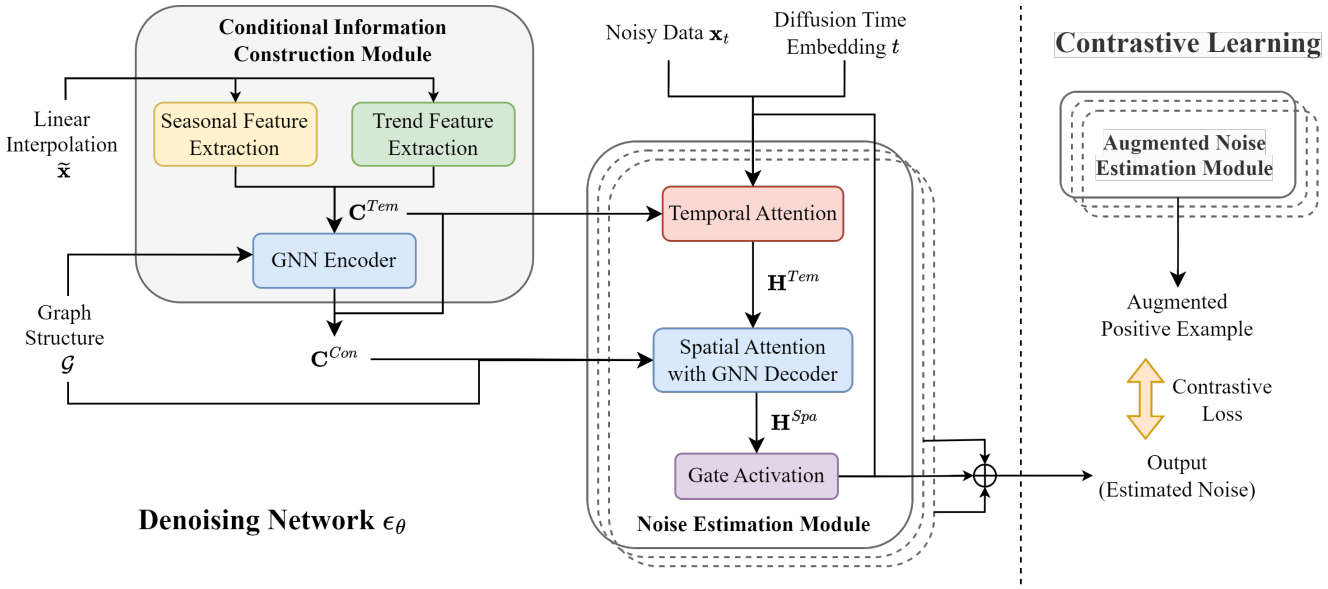


FIGURE 3 C^{2}TSD Overview. The denoising network $\boldsymbol{\epsilon}_\theta$ includes the conditional information construction module and the noise estimation module. The Conditional Information Construction Module (Section 3.1.1) processes the linear interpolation $\widetilde{\mathbf{x}}$ and graph structure $\mathcal{G}$ to learn the conditional representation $\mathbf{C}^{Con}$, which will later be used as guidance for the following module. The Noise Estimation Module (Section 3.1.2) takes the noisy data $\mathbf{x}_t$, the conditional feature $\mathbf{C}^{Con}$, and the graph structure $\mathcal{G}$ as the input to convert noisy information into spatiotemporal data with imputed values. Besides, the contrastive learning strategy (Section 3.2) is used to redesign the final loss function to improve the model performance and generalizability.

3 | APPROACH

Our approach (Fig. 3) is fundamentally based on the DDPM. We extend DDPM to the conditional one by adding conditional information to align it with our spatiotemporal imputation task. Then, we design the denoising network $\boldsymbol{\epsilon}_\theta$, containing two main modules: conditional information construction module and noise estimation module. Last, we redesign the final loss by adding a contrastive learning loss together with the diffusion model loss in Eq. (6) to improve the stability and robustness of the denoising network $\boldsymbol{\epsilon}_\theta$ training process.

3.1 | Denoising Network

The denoising network $\boldsymbol{\epsilon}_\theta$ is used to sample $\mathbf{x}_t$ in the reverse process. The interpolation of the observed values and the graph structure serve as conditional information to guide the reverse process. We use a conditional information construction module to process the conditional information to learn coarse spatiotemporal dependencies. The generated conditional representation is a

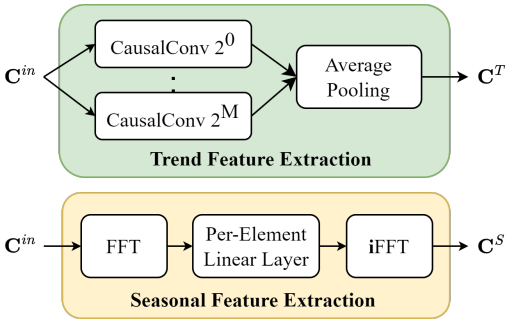


FIGURE 4 Details of Trend Feature Extraction and Seasonal Feature Extraction. The Trend Feature Extraction (the upper box) consists of a stack of causal convolution layers with different kernel sizes. The Seasonal Feature Extraction (the lower box) is implemented by Fast Fourier Transform (FFT) and inverse Fast Fourier Transform (iFFT) layers. $\mathbf{C}^{in}$ is a representation mapped from the linear interpolation of the observed values $\widetilde{\mathbf{x}}$ to the latent space.

coarse guidance that will be used to calculate attention weights in the noise estimation module and improve the module's ability to learn spatiotemporal dependencies.

3.1.1 | Conditional Information Construction Module

The conditional information construction module generates conditional representations to capture spatiotemporal dependencies from coarse-interpolating the observed values and the graph structure. This helps calculate attention weights in the noise estimation module. The module works as follows: first, we map the interpolation of the observed values $\widetilde{\mathbf{x}}$ to the latent space, $\mathbf{C}^{in} = Conv(\widetilde{\mathbf{x}})$; then, we use two separate modules to construct trend and seasonal representations from the intermediate representation $\mathbf{C}^{in}$; finally, a GNN Encoder aggregates the neighbour information with the graph structure $\mathcal{G}$ to generate the final conditional representation $\mathbf{C}^{Con}$.

A prior study [14] accepts both conditional information $\widetilde{\mathbf{x}}$ and noisy information $\mathbf{x}_t$ as inputs and directly applies a graph convolution module to aggregate the neighbour representations and generate the spatiotemporal conditional representations. However, the direct and simple structure can only learn entangled temporal dependencies within the observed values. Imputation accuracy can be significantly compromised if underlying components or patterns evolve unexpectedly within missing data segments. To address this challenge, we propose to derive separate representations for trend and seasonal features of spatiotemporal data (as illustrated in Fig. 4). By distinctly modelling these foundational temporal structures from observed data, our framework establishes a stable and robust context. This well-defined base then enables the generative process to more effectively model and instill diverse, plausible variations within the remaining residual signals of the imputation.

Trend Feature Extraction is designed as a mixture of $M+1$ autoregressive experts, with $M = \lfloor log_2(L/2) \rfloor$ and L is the sequence length. Each expert is implemented by a 1-dimensional causal convolution, with d input channels and d_T output channels. The kernel size for the m-th expert is set to 2^m. Every expert generates an output matrix $\mathbf{C}^{T,m} = CausalConv(\mathbf{C}^{in}, 2^m)$. To synthesize the trend representation $\mathbf{C}^T$, we apply an average-pooling operation over the collective outputs of these experts, which can be formulated as

$$\mathbf{C}^T = AvgPool\left(\mathbf{C}^{T,0}, \mathbf{C}^{T,1}, \cdots, \mathbf{C}^{T,M}\right) = \frac{1}{M+1}\sum_{m=0}^{M} \mathbf{C}^{T,i}. \tag{10}$$

We derive seasonal representations in the frequency domain inspired by the methodology in [25]. Specifically, *Seasonal Feature Extraction* employs a trainable Fourier layer to capture seasonal features from the frequency domain. This involves a discrete Fourier transform (DFT), which converts signals from the time domain to the frequency domain, expressed as $\mathcal{F}(\mathbf{C}^{in}) \in \mathbb{C}^{F\times(d\cdot N)}$. Here, $F = \lfloor L/2 \rfloor + 1$ signifies the number of frequencies obtained. The subsequent trainable Fourier layer uses a per-element linear layer to perform an affine transformation on each frequency with distinct complex-valued parameters. After processing in the frequency domain, the representations are transformed to the time domain to calculate the seasonal representation $\mathbf{C}^S \in \mathbb{R}^{L\times(d_S\cdot N)}$

via an inverse DFT layer. The i, k-th element of the seasonal representation is given by:

$$\mathbf{C}_{i,k}^{S} = \mathcal{F}^{-1}\left(\sum_{j=1}^{d} \mathbf{W}_{i,j,k}\mathcal{F}\left(\mathbf{C}_{i,j} + \mathbf{B}_{i,k}\right)\right), \tag{11}$$

where $\mathbf{W} \in \mathbb{C}^{F\times(d\cdot N)\times(d_S\cdot N)}$ and $\mathbf{B} \in \mathbb{C}^{F\times(d_S\cdot N)}$ represent learnable parameters of linear layer.

The intermediate temporal representation $\mathbf{C}^{Tem}$ is the concatenation of the trend and seasonal representations, $\mathbf{C}^{Tem} = [\mathbf{C}^{T}; \mathbf{C}^{S}] \in \mathbb{R}^{L\times(d\cdot N)}$, where $\mathbf{C}^{T} \in \mathbb{R}^{L\times(d_T\cdot N)}$, $\mathbf{C}^{S} \in \mathbb{R}^{L\times(d_S\cdot N)}$, and $d = d_T + d_S$. Besides, we adopt a GNN Encoder to capture the spatial dependencies and generate the final conditional representation, which can be formulated as:

$$\begin{aligned} \mathbf{C}^{Spa} &= GNN\left(\mathbf{C}^{Tem}, \mathbf{A}\right), \\ \mathbf{C}^{Con} &= MLP\left(Norm\left(\mathbf{C}^{Spa} + \mathbf{C}^{Tem}\right)\right), \end{aligned} \tag{12}$$

where $\mathbf{C}^{Spa}$ represents the intermediate spatial conditional information, while $\mathbf{C}^{Con} \in \mathbb{R}^{N\times L\times d}$ denotes the final conditional representation. The adjacency matrix $\mathbf{A}$, derived from the graph structure $\mathcal{G}$, encapsulates the geographical relationships. The notation $GNN(\cdot)$ refers to the GNN encoder, which can be implemented using any GNN architecture. Our experiment employed the same module from Graph Wavenet [1]. The adjacency matrix $\mathbf{A}$ is computed using the sensor geographic distance information between spatial features, as described in [10]. The conditional representation $\mathbf{C}^{Con}$ serves as a solution to constructing conditional information; it contains the disentangled temporal dependencies and aggregated spatial geographical relationships when compared with the initial interpolated conditional information $\widetilde{\mathbf{x}}$.

3.1.2 | Noise Estimation Module

This module employs two attention modules to reduce the randomness introduced by Gaussian noise in data distributions. It utilizes conditional information $\mathbf{C}^{Con}$ to capture spatiotemporal dependencies in noisy data $\mathbf{x}_t$. The module takes three inputs: the noisy information $\mathbf{H}^{in} = Conv\left(\widetilde{\mathbf{x}}\|\mathbf{x}_t\right)$, the conditional representations $\mathbf{C}^{Con}$, and the adjacency matrix $\mathbf{A}$. It first constructs a temporal feature $\mathbf{H}^{Tem}$ using a Temporal Attention Module $\mathcal{T}(\cdot)$. Then, the output is passed to a Spatial Attention Module with GNN decoder $\mathcal{S}(\cdot)$ to construct a spatiotemporal feature $\mathbf{H}^{spa}$. The above process can be formulated as:

$$\begin{aligned} \mathbf{H}^{tem} &= \mathcal{T}\left(\mathbf{H}^{in}\right) = Attn_{tem}\left(\mathbf{H}^{in}\right), \\ \mathbf{H}^{spa} &= \mathcal{S}\left(\mathbf{H}^{tem}, \mathbf{A}\right) = MLP\left(Norm\left(Attn_{spa}\left(\mathbf{H}^{tem}\right) + \mathbf{H}^{tem}\right) + Norm\left(GNN\left(\mathbf{H}^{tem}, \mathbf{A}\right) + \mathbf{H}^{tem}\right)\right), \end{aligned} \tag{13}$$

where $Attn_{tem}(\cdot)$ and $Attn_{spa}(\cdot)$ are temporal and spatial attention mechanisms, respectively. Here, we additionally use the conditional representation $\mathbf{C}^{Con}$ in the attention calculation to mitigate the impact of noise accumulated during diffusion steps. We formulate the temporal attention $Attn_{tem}(\cdot)$ as:

$$Attn_{tem}(\mathcal{A}_T, \mathcal{V}_T) = SoftMax\left(\frac{\mathcal{Q}_T\mathcal{K}_T^{T}}{\sqrt{d}}\right)\cdot\mathcal{V}_T, \tag{14}$$

where $W_T^{Q}, W_T^{K}, W_T^{V} \in \mathbb{R}^{d\times d}$ are all learnable projection parameters; $\mathcal{Q}_T = \mathbf{C}^{Tem}\cdot\mathbf{W}_T^{Q}$, $\mathcal{K}_T = \mathbf{C}^{Tem}\cdot\mathbf{W}_T^{K}$, and $\mathcal{V}_T = \mathbf{H}^{in}\cdot\mathbf{W}_T^{V}$. We use temporal conditional representation $\mathbf{C}^{Tem}$ to calculate the query and key in the temporal attention. $Attn_{spa}(\mathcal{A}_S, \mathcal{V}_S)$ is modified in a similar way. We use the final conditional representation to calculate the query and key in the spatial attention. $\mathcal{Q}_S = \mathbf{C}^{Con}\cdot\mathbf{W}_S^{Q}$, $\mathcal{K}_S = \mathbf{C}^{Con}\cdot\mathbf{W}_S^{K}$, and $\mathcal{V}_S = \mathbf{H}^{tem}\cdot\mathbf{W}_T^{V}$.

3.2 | Contrastive Learning and Loss Function

We redesign the final loss function using a contrastive learning strategy to improve the training of the conditional DDPM. We enhance the final loss function by incorporating a contrastive learning approach to optimize the training of the conditional DDPM. As noted in [17], conditional guidance is an effective method for balancing diversity and fidelity. Nonetheless, using conditional information might lead the diffusion model to generate highly similar samples, which can impede its ability to manage unseen spatiotemporal dependencies. By leveraging contrastive learning, we aim to elevate the quality of the generated representations and improve our model's generalizability. Additionally, our designed missing pattern is Missing Completely

at Random (MCAR)[30], meaning the missing data is independent of both observed and unobserved data. This increases the complexity of our model in capturing complex spatiotemporal dependencies. The conditional information construction module helps mitigate this challenge by leveraging disentangled temporal representations. Nevertheless, the model can still mistakenly interpret noisy information as data features and relationships. Contrastive learning addresses this by bringing similar samples closer together and pushing different samples further apart in the feature space, thereby reducing the likelihood of the model learning from noise.

Contrastive-based approaches create positive and negative samples via data augmentation or context sampling. The model is then trained by maximizing the Mutual Information (MI) between the two positive samples[31]. Data augmentation techniques are frequently employed to produce various views of an input sample, allowing the model to learn representations by increasing the similarity of views from the same sample while decreasing the similarity between views from different samples. We follow SimCLR[26], a typical multi-view invariance-based framework, to design the positive and negative pairs. The core is to obtain different views of the input samples. As mentioned in Section 3.1.2, the predicted noise, i.e., the final output of the noise estimation module, is generated by the stacked residual layer structure. To generate an augmented positive example, we apply another same-structured residual layer without parameter sharing to process the noisy information $\mathbf{H}^{in}$. The negative examples are not sampled explicitly. We treated the other augmented examples within a batch as negative examples. The remaining samples in the batch are considered negative samples. The contrastive loss $\mathcal{L}_C$ for a positive pair of examples (i, j) is calculated as:

$$\mathcal{L}_C = -\log \frac{\exp\left(\text{sim}\left(\mathbf{z}_t^i, \mathbf{z}_t^j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbf{1}_{[k\neq i]} \exp\left(\text{sim}\left(\mathbf{z}_t^i, \mathbf{z}_t^k\right)/\tau\right)}, \tag{15}$$

where $\text{sim}(\cdot,\cdot)$ refers to the cosine similarity calculation, $\mathbf{1}_{[k\neq i]} \in \{0, 1\}$ is an indicator function that evaluates to 1 if and only if $k \neq i$, and τ represents the temperature coefficient. Here, N denotes the batch size. The contrastive loss operates on augmented example pairs from a batch, leading to $2N$ examples and yielding $2(N-1)$ negative samples. The final contrastive loss is computed over all positive pairs, (i, j) and (j, i), in the batch.

The overall model loss combines the diffusion model loss and the contrastive loss. Referring to the training objective in Eq. (9), the reconstruction loss $\mathcal{L}_D(\theta)$ for the conditional diffusion model is employed to select the appropriate noise space for predicting the generated data. This can be defined as follows

$$\mathcal{L}_D(\theta) = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon},t}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t \mid \widetilde{\mathbf{x}}, \mathcal{G})\right\|^2\right]. \tag{16}$$

Applying the contrastive loss at the same weight as the reconstruction loss is unreasonable. The ranges of cross-entropy loss (contrastive loss) and mean square error (reconstruction loss) are different. If they are added directly, the loss function with a larger value will dominate the total loss, and the smaller one will be ignored. This will affect the direction of the model gradient update, causing unstable parameter updates and thus affecting the training convergence. To avoid this problem, we introduce a weight coefficient α to control the magnitude of the contrastive loss. The overall model loss $\mathcal{L}$ can be formulated as:

$$\mathcal{L} = \mathcal{L}_D(\theta) + \alpha\mathcal{L}_C. \tag{17}$$

4 | EXPERIMENTS

We conduct experiments in this section to evaluate our model, including experimental settings (datasets, evaluation metrics, baselines), experimental design, and results analysis.

4.1 | Datasets and Evaluation Metrics

We use three datasets from the meteorology and transportation fields. Each dataset in our study exhibits distinct characteristics and poses specific challenges relevant to spatiotemporal imputation. This diversity renders them suitable for evaluating the efficacy of the C^{2}TSD framework. (1) **AQI-36**[32]: This air quality dataset consists of PM2.5 measurements recorded by 36 monitoring stations located in Beijing, China, from May 2014 to April 2014. The sampling rate is 1 hour, and the windows of data length are 36 steps. (2) **PEMS-BAY**[33]: The traffic dataset used is acquired from the Performance Measurement System

(PeMS) of the California Transportation Agencies (CalTrans). For our experiment, we have selected data from 325 sensors in the Bay Area from January 1st, 2017, to May 31st, 2017. The sampling rate is 5 minutes, and the windows of data length are 24 steps, corresponding to 2 hours. (3) **METR-LA** [33]: This dataset consists of traffic data gathered from loop detectors on highways in Los Angeles County. For our experiment, we have chosen data from 207 sensors covering a period of four months, from March 1st, 2012, to June 30th, 2012. The sampling rate is 5 minutes, and the windows of data length are 24 steps, corresponding to 2 hours.

Following [14], we utilize three performance metrics: *Mean Absolute Error (MAE)* and *Mean Squared Error (MSE)* offer deterministic evaluations of the error between the imputed data and the actual values; *Continuous Ranked Probability Score (CRPS)* [34] measures the closeness between the predicted probability distribution and the observed data.

4.2 | Baselines

We evaluate the performance of C^2TSD through comparisons with a selection of state-of-the-art baselines, including statistical methods (MEAN, DA, kNN, Linear), machine learning methods (MICE, VAR, KF), matrix factorization methods (TRMF, BATF), deterministic deep learning methods (BRITS, GRIN, rGAIN), and probabilistic generative methods (V-RIN, GP-VAE, CSDI, PriSTI). The diverse baselines ensure the comparative evaluation is comprehensive. Below, we provide a brief introduction to these baseline models: **MEAN**: This method imputes missing values using each node's historical average value. **DA**: Daily averages at relevant time steps for imputation. **kNN**: This approach calculates and uses the average value of geographically proximate nodes for imputation. **Linear**: Linear interpolation of time series, computed at the level of each individual variable. **KF**: A Kalman Filter approach is applied to impute missing time series data for each variable individually. **MICE** [35]: Multivariate Imputation by Chained Equations. **VAR**: A vector autoregressive model used as a single-step predictor. **TRMF** [36]: Temporal Regularized Matrix Factorization. **BATF** [37]: Bayesian Augmented Tensor Factorization integrates spatiotemporal domain knowledge. **BATF** [37]: Bayesian Augmented Tensor Factorization integrates spatiotemporal domain knowledge. **BRITS** [11]: A bidirectional RNN-based method for multivariate time series imputation (MTSI). **GRIN** [10]: Combine bi-GRU with graph convolution network for MTSI. **rGAIN** [38]: Extend the GAIN model with a bi-recurrent encoder-decoder structure. **V-RIN** [39]: Use VAE to enhance deterministic imputation to provide the probability imputation result. **GP-VAE** [40]: A probabilistic imputation method for time series combining a Variational Autoencoder with Gaussian processes. **CSDI** [16]: A conditional diffusion probability model-based imputation method that views different nodes as multiple feature dependencies. **SSSD** [41]: A diffusion model replacing the noise estimation network with a structured state space model to capture long-term time series dependencies. **PriSTI** [14]: A conditional diffusion framework using a feature extraction module to calculate spatiotemporal attention weights. **Score-CDM** [42]: A score-weighted convolutional diffusion model to adaptively balance local and global temporal features through spectral and time domain transformations. **ImputeFormer** [43]: A transformer-based model integrating low-rank properties to achieve a balance between strong inductive bias and high expressiveness.

4.3 | Experimental Design

4.3.1 | Datasets Split

We divided the data into training, validation, and test sets in alignment with the dataset division methodology employed by [10]. For the AQI-36 dataset, March, June, September, and December were selected as the test set. The validation set includes the last 10% data corresponding to February, May, August, and November, while the remainder forms the training set. As for the PEMS-BAY and METR-LA datasets, a chronological division strategy was applied, with 70% data allocated to the training set, 10% to the validation set, and the rest to the test set.

4.3.2 | Choice of Imputation Targets

Although the datasets inherently contain missing values, they lack the ground truth for evaluation purposes. Therefore, the common practice in the field of research is to artificially introduce missing values into test portions of each dataset to facilitate the performance evaluation. For the AQI-36 dataset, we simulated the distribution of real-world missing data, creating a relevant setting for evaluating imputation performance following [44]. For the traffic datasets PEMS-BAY and METR-LA, we used a mask matrix [10] to inject missing values manually. Specifically, we employed two missing patterns to create the masks for missing

TABLE 1 Performance of compared methods with respect to MAE and MSE. Bold denotes the best; Underline denotes the second best. *Improv.* denotes the performance improvement of C^{2}TSD over the best baseline. The results of our model are the average of 5 runs. **–** denotes that the corresponding metrics are not reported in the original research.

	AQI-36		PEMS-BAY				METR-LA			
			Point missing		Block missing		Point missing		Block missing	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Mean	53.48$_{\pm 0.00}$	4578.08$_{\pm 0.00}$	5.42$_{\pm 0.00}$	86.59$_{\pm 0.00}$	5.46$_{\pm 0.00}$	87.56$_{\pm 0.00}$	7.56$_{\pm 0.00}$	142.22$_{\pm 0.00}$	7.48$_{\pm 0.00}$	139.54$_{\pm 0.00}$
DA	50.51$_{\pm 0.00}$	4416.10$_{\pm 0.00}$	3.35$_{\pm 0.00}$	44.50$_{\pm 0.00}$	3.30$_{\pm 0.00}$	43.76$_{\pm 0.00}$	14.57$_{\pm 0.00}$	448.66$_{\pm 0.00}$	14.53$_{\pm 0.00}$	445.08$_{\pm 0.00}$
kNN	30.21$_{\pm 0.00}$	2892.31$_{\pm 0.00}$	4.30$_{\pm 0.00}$	49.80$_{\pm 0.00}$	4.30$_{\pm 0.00}$	49.90$_{\pm 0.00}$	7.88$_{\pm 0.00}$	129.29$_{\pm 0.00}$	7.79$_{\pm 0.00}$	124.61$_{\pm 0.00}$
Linear	14.46$_{\pm 0.00}$	673.92$_{\pm 0.00}$	0.76$_{\pm 0.00}$	1.74$_{\pm 0.00}$	1.54$_{\pm 0.00}$	14.14$_{\pm 0.00}$	2.43$_{\pm 0.00}$	14.75$_{\pm 0.00}$	3.26$_{\pm 0.00}$	33.76$_{\pm 0.00}$
KF	54.09$_{\pm 0.00}$	4942.26$_{\pm 0.00}$	5.68$_{\pm 0.00}$	93.32$_{\pm 0.00}$	5.64$_{\pm 0.00}$	93.19$_{\pm 0.00}$	16.66$_{\pm 0.00}$	529.96$_{\pm 0.00}$	16.75$_{\pm 0.00}$	534.69$_{\pm 0.00}$
MICE	30.37$_{\pm 0.09}$	2594.06$_{\pm 7.17}$	3.09$_{\pm 0.02}$	31.43$_{\pm 0.41}$	2.94$_{\pm 0.02}$	28.28$_{\pm 0.37}$	4.42$_{\pm 0.07}$	55.07$_{\pm 1.46}$	4.22$_{\pm 0.05}$	51.07$_{\pm 1.25}$
VAR	15.64$_{\pm 0.08}$	833.46$_{\pm 13.85}$	1.30$_{\pm 0.00}$	6.52$_{\pm 0.01}$	2.09$_{\pm 0.10}$	16.06$_{\pm 0.73}$	2.69$_{\pm 0.00}$	21.10$_{\pm 0.02}$	3.11$_{\pm 0.08}$	28.00$_{\pm 0.76}$
TRMF	15.46$_{\pm 0.06}$	1379.05$_{\pm 34.83}$	1.85$_{\pm 0.00}$	10.03$_{\pm 0.00}$	1.95$_{\pm 0.01}$	11.21$_{\pm 0.06}$	2.86$_{\pm 0.00}$	20.39$_{\pm 0.02}$	2.96$_{\pm 0.00}$	22.65$_{\pm 0.13}$
BATF	15.21$_{\pm 0.27}$	662.87$_{\pm 29.55}$	2.05$_{\pm 0.00}$	14.90$_{\pm 0.06}$	2.05$_{\pm 0.00}$	14.48$_{\pm 0.01}$	3.58$_{\pm 0.01}$	36.05$_{\pm 0.02}$	3.56$_{\pm 0.01}$	35.39$_{\pm 0.03}$
BRITS	14.50$_{\pm 0.35}$	622.36$_{\pm 65.16}$	1.47$_{\pm 0.00}$	7.94$_{\pm 0.03}$	1.70$_{\pm 0.01}$	10.50$_{\pm 0.07}$	2.34$_{\pm 0.00}$	16.46$_{\pm 0.05}$	2.34$_{\pm 0.01}$	17.00$_{\pm 0.14}$
GRIN	12.08$_{\pm 0.47}$	523.14$_{\pm 57.17}$	0.67$_{\pm 0.00}$	1.55$_{\pm 0.01}$	1.14$_{\pm 0.01}$	6.60$_{\pm 0.10}$	1.91$_{\pm 0.00}$	10.41$_{\pm 0.03}$	2.03$_{\pm 0.00}$	13.26$_{\pm 0.05}$
V-RIN	10.00$_{\pm 0.10}$	838.05$_{\pm 24.74}$	1.21$_{\pm 0.03}$	6.08$_{\pm 0.29}$	2.49$_{\pm 0.04}$	36.12$_{\pm 0.66}$	3.96$_{\pm 0.08}$	49.98$_{\pm 1.30}$	6.84$_{\pm 0.17}$	150.08$_{\pm 6.13}$
GP-VAE	25.71$_{\pm 0.30}$	2589.53$_{\pm 59.14}$	3.41$_{\pm 0.23}$	38.95$_{\pm 4.16}$	2.86$_{\pm 0.15}$	26.80$_{\pm 2.10}$	6.57$_{\pm 0.10}$	127.26$_{\pm 3.97}$	6.55$_{\pm 0.09}$	122.33$_{\pm 2.05}$
rGAIN	15.37$_{\pm 0.26}$	641.92$_{\pm 33.89}$	1.88$_{\pm 0.02}$	10.37$_{\pm 0.20}$	2.18$_{\pm 0.01}$	13.96$_{\pm 0.20}$	2.83$_{\pm 0.01}$	20.03$_{\pm 0.09}$	2.90$_{\pm 0.01}$	21.67$_{\pm 0.15}$
CSDI	9.51$_{\pm 0.10}$	352.46$_{\pm 7.50}$	**0.57**$_{\pm 0.00}$	1.12$_{\pm 0.03}$	0.86$_{\pm 0.00}$	4.39$_{\pm 0.02}$	1.79$_{\pm 0.00}$	8.96$_{\pm 0.08}$	1.98$_{\pm 0.00}$	12.62$_{\pm 0.60}$
SSSD	–	–	0.97$_{\pm 0.01}$	2.98$_{\pm 0.03}$	1.03$_{\pm 0.01}$	7.32$_{\pm 0.05}$	2.83$_{\pm 0.02}$	21.95$_{\pm 0.14}$	2.95$_{\pm 0.01}$	23.48$_{\pm 0.09}$
PriSTI	9.22$_{\pm 0.07}$	324.95$_{\pm 7.03}$	**0.57**$_{\pm 0.00}$	**1.09**$_{\pm 0.00}$	**0.78**$_{\pm 0.00}$	3.50$_{\pm 0.01}$	1.76$_{\pm 0.00}$	8.63$_{\pm 0.05}$	1.89$_{\pm 0.00}$	11.21$_{\pm 0.02}$
Score-CDM	13.74$_{\pm _}$	–	0.65$_{\pm _}$	1.46$_{\pm _}$	1.55$_{\pm _}$	–	1.93$_{\pm _}$	12.89$_{\pm _}$	2.60$_{\pm _}$	–
ImputeFormer	11.58$_{\pm _}$	–	0.64$_{\pm _}$	–	0.95$_{\pm _}$	–	1.80$_{\pm _}$	–	1.86$_{\pm _}$	–
C^{2}TSD	**9.09**$_{\pm 0.03}$	**309.81**$_{\pm 6.35}$	**0.57**$_{\pm 0.00}$	**1.09**$_{\pm 0.00}$	0.81$_{\pm 0.00}$	**3.23**$_{\pm 0.01}$	**1.70**$_{\pm 0.00}$	**8.47**$_{\pm 0.03}$	**1.79**$_{\pm 0.00}$	**10.65**$_{\pm 0.04}$
Improv.	*1.43%*	*4.89%*	*0.00%*	*0.00%*	*-3.85%*	*8.48%*	*3.28%*	*1.95%*	*5.91%*	*5.24%*

TABLE 2 Performance of generative methods in CRPS. "Point" and "Block" denote the respective missing patterns.

	AQI-36	PEMS-BAY		METR-LA	
		Point	Block	Point	Block
GP-VAE	0.3377	0.0568	0.0436	0.0977	0.1118
V-RIN	0.3154	0.0191	0.3940	0.0781	0.1283
CSDI	0.1056	**0.0067**	0.0127	0.0235	0.0260
PriSTI	0.1025	**0.0067**	**0.0094**	0.0231	0.0249
C^{2}TSD	**0.0995**	**0.0067**	0.0097	**0.0223**	**0.0237**

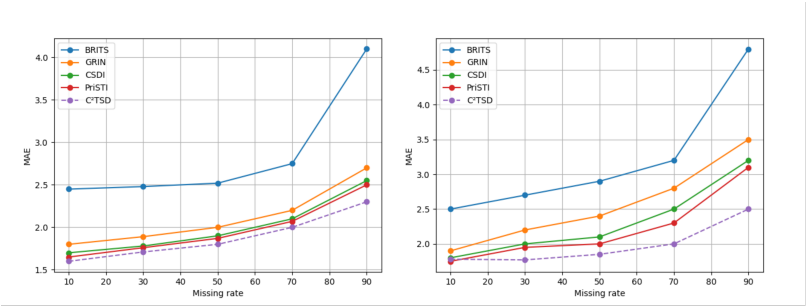
values: (1) Point missing, where we randomly masked 25% of the observations, and (2) Block missing, which randomly masks 5% of the observations and then further masks consecutive observations with a probability of 0.15%.

4.3.3 | Training Environment and Hyperparameters

We trained our model on an Nvidia A40 GPU equipped with 48 GB memory and applied the Adam optimizer, well-known for its adaptability and robust performance, to optimize C^{2}TSD's parameters. Additionally, we applied a cosine annealing decay strategy to the learning rate. This strategy gradually reduced the learning rate following a cosine function to fine-tune the model's performance over time. We configured hyperparameters to balance the learning efficiency and performance. Specifically, we set the learning rate to 10^{-3} and the decay to 10^{-5} to enable effective learning while preventing overshooting during optimization. Regarding the forward process, the diffusion step T was set to 100 for the AQI-36 dataset and 50 for two traffic datasets; the minimum (β_1) and maximum (β_T) noise levels were set to 10^{-4} and 0.2, respectively.

4.4 | Overall Performance

The comparative evaluation of C^{2}TSD against baseline methods is presented in Table 1 (MAE/MSE) and Table 2 (CRPS for probabilistic methods, using 100 samples per missing value). The standard deviation of the results is not presented in Table 2



(a) MAE on METR-LA (Point) (b) MAE on METR-LA (Block)

FIGURE 5 The imputation results with different missing ratios for the METR-LA dataset.

due to the low variability of the CRPS metric across five experimental trials. Specifically, the standard error was less than 0.001 for CSDI, PriSTI, and our proposed C^{2}TSD. Our analysis reveals three principal observations that merit deeper discussion:

4.4.1 | Performance Hierarchy

The results demonstrate a consistent performance ranking: diffusion-based methods (C^{2}TSD > PriSTI > CSDI) outperform VAE-based approaches (V-RIN, GP-VAE), which in turn surpass GCN-enhanced RNN methods (GRIN) and conventional baselines (linear interpolation, MICE). This hierarchy suggests two critical factors influencing imputation quality: **Spatiotemporal dependency modelling**: Methods explicitly capturing both temporal dynamics and spatial correlations (e.g., GRIN's graph convolutions) show advantages over pure temporal models (rGAIN, BRITS); **Uncertainty handling**: Diffusion-based approaches (C^{2}TSD, PriSTI) outperform VAE-based methods, likely due to their iterative refinement process that better propagates uncertainty in missing patterns.

4.4.2 | Architectural Advantages

The performance superiority of C^{2}TSD stems primarily from its trend-seasonal disentanglement and contrastive spatiotemporal learning. First, the explicit separation of trend (long-term patterns) and seasonal (periodic fluctuations) components enables robust modelling of realistic complex temporal patterns. This is evidenced by its MAE and MSE improvement on METR-LA, where urban traffic exhibits both abrupt trends and multi-scale periodicity. Second, the contrastive learning strategy strengthens dependency capture by aligning representations across variables and timesteps—particularly effective for datasets with strong spatial correlations (8.48% MSE gain on PEMS-BAY). These innovations, combined with adaptive noise scheduling that adjusts diffusion intensity based on missing patterns, allow the model to balance global consistency and local detail recovery. The design explains C^{2}TSD's robustness in complex scenarios (e.g., 4.89% improvement in MSE on AQI-36) while maintaining efficiency in simpler cases (PEMS-BAY's smooth patterns).

4.4.3 | Dataset-Specific Behavior

The exceptional case where PriSTI approaches C^{2}TSD's performance on PEMS-BAY (MSE: 3.23 v.s. 3.50) may stem from the dataset's characteristics: Minimal inherent missingness reduces error propagation challenges; Smooth traffic patterns (vs. METR-LA's urban complexity) decrease the need for sophisticated distribution modelling; Supported by linear interpolation's competitive performance (MAE: 1.70 v.s. 1.76 and 1.79 v.s. 1.89), indicating simpler dependencies.

4.4.4 | Performance with Different Missing Ratio

Fig 5 presents the MAE of different imputation methods under varying missing rates across two different missing patterns. As expected, all methods show an increasing MAE as the missing rate rises, reflecting the challenge of imputing highly sparse data.

TABLE 3 Ablation Studies. "Point" and "Block" denote the respective missing patterns.

Method	AQI-36		METR-LA			
			Point		Block	
	MAE	MSE	MAE	MSE	MAE	MSE
C^2TSD	**9.09**$_{\pm 0.03}$	**309.81**$_{\pm 6.35}$	**1.70**$_{\pm 0.00}$	**8.47**$_{\pm 0.03}$	**1.79**$_{\pm 0.00}$	**10.65**$_{\pm 0.04}$
w/o CL	10.18$_{\pm 0.04}$	389.23$_{\pm 8.32}$	2.18$_{\pm 0.02}$	16.83$_{\pm 0.05}$	2.27$_{\pm 0.01}$	17.83$_{\pm 0.05}$
w/o TFD	10.06$_{\pm 0.05}$	380.94$_{\pm 7.61}$	2.07$_{\pm 0.01}$	13.66$_{\pm 0.04}$	2.14$_{\pm 0.01}$	15.13$_{\pm 0.05}$
w/o SFD	9.80$_{\pm 0.04}$	361.93$_{\pm 9.05}$	2.09$_{\pm 0.01}$	14.70$_{\pm 0.05}$	2.07$_{\pm 0.01}$	14.19$_{\pm 0.04}$
w/o Linear	10.51$_{\pm 0.06}$	450.28$_{\pm 11.78}$	1.74$_{\pm 0.00}$	8.51$_{\pm 0.04}$	1.88$_{\pm 0.00}$	11.36$_{\pm 0.04}$

Among the baselines, BRITS exhibits the most significant performance degradation, while GRIN and CSDI perform relatively better but still show a noticeable rise in error at higher missing rates. C^2TSD consistently achieves lower MAE than most baselines, particularly in high missing rate scenarios, demonstrating its robustness in handling extreme data sparsity. Compared to PriSTI, which also shows competitive performance, C^2TSD maintains a smaller error across different missing rates, especially when missingness exceeds 50%. At the missing rate of 90%, C^2TSD outperforms PriSTI, highlighting its superior ability to capture spatiotemporal dependencies and reconstruct missing values more accurately. This improvement suggests that C^2TSD's approach to leveraging conditional dependencies is more effective than PriSTI's interpolation strategy, particularly when data availability is severely limited. Overall, the results confirm that C^2TSD is more robust than other baselines, offering more reliable and stable imputation performance, particularly in highly sparse scenarios.

4.5 | Ablation Study

The impact of the core modules of our approach is evaluated by contrasting it with the following different variants: *w/o CL*: Remove the contrastive loss $\mathcal{L}_C$ in Eq. (15); only calculate the gradient of the reconstruction loss $\mathcal{L}_D(\theta)$; *w/o TFD*: Remove the trend representation $\mathbf{C}^T$ in Eq. (10) in the conditional information construction module; only keep the seasonal representation $\mathbf{C}^S$ for using to construct the conditional information; *w/o SFD*: Opposite to the previous variant, remove the seasonal representation $\mathbf{C}^S$ in Eq. (11) in the conditional information construction module; only keep the trend representation $\mathbf{C}^T$ for the conditional information; *w/o Linear*: Remove linear interpolation $\widetilde{\mathbf{x}}$, to keep model structure consistency, we use incomplete data x in conditional information construction module to generate $\mathbf{C}^{Con}$.

The results (Table 3) demonstrate that each component (contrastive loss, trend representation, seasonal representation, linear interpolation) contributes to the superior performance of C^2TSD. We have omitted the results for the PEMS-BAY dataset, as they align with those presented in Table 3. Among the three components analyzed, the contrastive learning strategy has the most substantial impact on C^2TSD's performance—its removal results in a considerable decline in effectiveness. This finding highlights that the contrastive learning strategy enhances the model's capability to learn complex spatiotemporal dependencies. The trend component generally contributes more significantly than the seasonal component, as indicated by the performance differences between *w/o Trend* and *w/o Season*. Specifically, excluding the trend component leads to a more significant performance drop on the AQI-36 dataset under both missing patterns and the METR-LA dataset under the block missing pattern, as shown in Table 3. This can be attributed to the complex factors affecting PM2.5 concentration and the absence of clear periodicity in the AQI-36 dataset. Conversely, the seasonal component is more crucial than the trend component for the METR-LA dataset under the point-missing pattern. This may be because the average traffic speed typically does not exhibit rapid trend changes over short intervals, which are more common in the point-missing pattern. Specifically, the exclusion of the linear interpolation component (*w/o Linear*) results in the most significant performance drop on the AQI-36 dataset, with MAE increasing by 15.6% and MSE by 45.3%, highlighting its critical role in handling missing data. While the impact on the METR-LA dataset is relatively smaller, the consistent increase in MAE and MSE across both point and block missing patterns underscores the importance of integrating linear interpolation for robust imputation.

4.6 | Case Study: Sample Diversity Evaluation

To further investigate whether the additional diversity introduced by our method leads to more reliable uncertainty estimates, we conduct a case study on the AQI-36 and METR-LA datasets. Unlike the main experiments, which focus on pointwise accuracy (MAE) and distributional scoring (CRPS), this study emphasizes three uncertainty-focused metrics: mean prediction interval

TABLE 4 Sample Diversity Analysis on the AQI-36 and METR-LA datasets. Higher is better for MPIW; PICP should be close to 90%; Lower is better for WIS.

Method	AQI-36			METR-LA					
				Point missing			Block missing		
	MPIW ↑	PICP(%)	WIS ↓	MPIW ↑	PICP(%)	WIS ↓	MPIW ↑	PICP(%)	WIS ↓
PriSTI	28.64	81.23	34.62	5.80	80.55	6.61	5.65	79.07	7.08
C^2TSD	**32.67**	**83.33**	**34.59**	**6.09**	**81.75**	**6.38**	**6.48**	**81.86**	**6.64**

width (MPIW), prediction–interval coverage probability (PICP), and weighted interval score (WIS). Below is the detailed introduction for the three metrics:

Following [45,46], we draw $S = 100$ Monte Carlo samples per missing value—identical to the CRPS computation in Table 2—and compute all quantiles, prediction intervals, and scores from these samples. Let $y^{(i)}$ denote the ground-truth value at index i, and $\{\hat{y}^{(s,i)}\}_{s=1}^{S}$ the imputed samples. For a nominal level $1-\alpha$ (e.g., $\alpha = 0.10$ for a 90% interval), define:

$$\ell_i \;:=\; q^{(i)}_{\alpha/2}, \quad u_i \;:=\; q^{(i)}_{1-\alpha/2}, \quad \text{where } q^{(i)}_{\tau} = \text{Quantile}_{\tau}(\hat{y}^{(1,i)}, \ldots, \hat{y}^{(S,i)}). \tag{18}$$

MPIW measures the average width of the central (1–α) interval and reflects the sharpness (or diversity) of the uncertainty estimates. A larger MPIW indicates a broader, more diverse set of plausible imputations:

$$\text{MPIW}_{1-\alpha} \;=\; \frac{1}{N}\sum_{i=1}^{N}\big(u_i - \ell_i\big). \tag{19}$$

PICP quantifies calibration by measuring the fraction of ground-truth values that fall within the central prediction interval. Ideally, $\text{PICP}_{1-\alpha} \approx 1$–$\alpha$. Too low implies under-confidence; too high suggests over-conservativeness:

$$\text{PICP}_{1-\alpha} \;=\; \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\{y^{(i)} \in [\ell_i, u_i]\}. \tag{20}$$

WIS [47] is a strictly proper scoring rule that jointly captures both sharpness and calibration by aggregating interval scores over multiple levels of α. We adopt $K = 11$ intervals with $\alpha_k \in \{0.02, 0.05, \ldots, 0.90\}$, and weights $w_0 = \frac{1}{2}$, $w_k = \frac{\alpha_k}{2}$. Each level's interval score is:

$$\text{IS}_{\alpha_k}(y, \ell, u) = (u\text{–}\ell) + \frac{2}{\alpha_k}(\ell\text{–}y)_+ + \frac{2}{\alpha_k}(y\text{–}u)_+, \quad (x)_+ = \max(0, x). \tag{21}$$

With predictive median $\tilde{y}^{(i)} = q^{(i)}_{0.5}$, the final WIS is:

$$\text{WIS}^{(i)} = \frac{w_0|\tilde{y}^{(i)}\text{–}y^{(i)}| + \sum_{k=1}^{K} w_k\ \text{IS}_{\alpha_k}(y^{(i)}, \ell_{k,i}, u_{k,i})}{w_0 + \sum_{k=1}^{K} w_k}, \quad \text{WIS} = \frac{1}{N}\sum_{i=1}^{N}\text{WIS}^{(i)}. \tag{22}$$

Table 4 presents a comparative analysis of uncertainty quantification metrics between our method (C^2TSD) and the baseline (PriSTI) across the AQI-36 and METR-LA datasets. In the AQI-36 dataset, C^2TSD produces a broader prediction interval, as evidenced by its higher MPIW (32.67 vs. 28.64), which signifies increased diversity. Along with a higher PICP (83.33% vs. 81.23%), this demonstrates improved calibration, ensuring the prediction intervals closely match the ground truth. Importantly, the WIS remains comparable (34.59 vs. 34.62), confirming that the broader intervals do not come at the expense of predictive quality. In the METR-LA dataset, C^2TSD consistently outperforms PriSTI in both Point and Block missing patterns. Specifically, C^2TSD achieves a higher MPIW (6.09 vs. 5.80 in Point missing; 6.48 vs. 5.65 in Block missing), indicating a broader uncertainty range that captures more variability, and lower WIS (6.38 vs. 6.61 in Point; 6.64 vs. 7.08 in Block), reflecting more accurate uncertainty estimates. C^2TSD also maintains similar or slightly lower PICP values (80.55% vs. 81.75% in Point; 79.07% vs. 81.86% in Block), confirming that the model not only increases sample diversity but also ensures that the uncertainty estimation is well-calibrated. These results provide strong quantitative evidence of the benefits of increased sample diversity. C^2TSD not only generates richer and sharper uncertainty bands, but also maintains or improves calibration and predictive accuracy.

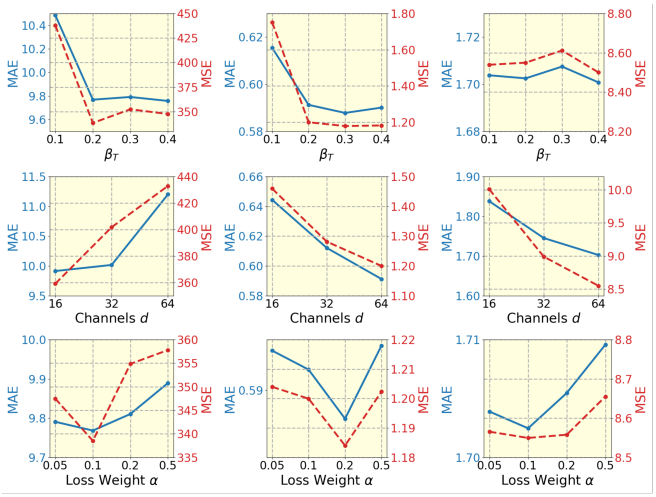


FIGURE 6 Sensitivity study of key hyperparameters. The blue solid lines represent MAE (Mean Absolute Error), and the red dashed lines represent MSE (Mean Squared Error) for different hyperparameters: maximum noise level β_T, number of channels (d), and loss weight (α) on the AQI-36 (a), PEMS-BAY (b), and METR-LA (c) datasets.

4.7 | Computational Complexity Analysis

The computational and memory requirements of C^{2}TSD are analyzed to provide insights into its scalability and practical applicability. For space complexity, our model primarily stores intermediate states during the diffusion process $\mathbf{O}(T \cdot N \cdot L \cdot d)$, adjacency matrices for the GNN $\mathbf{O}(N^2)$, and attention weights for spatiotemporal modelling $\mathbf{O}(L^2 + N^2)$. This leads to an overall space complexity of: $\mathbf{O}\left(T \cdot N \cdot L \cdot d + N^2 + L^2\right)$. The exponential term 2^M in trend extraction and the quadratic terms N^2 and L^2 present significant scalability challenges for large graphs or long sequences. Future work could address these limitations by exploring sparse GNN architectures, lightweight trend modelling techniques, or accelerated diffusion sampling methods. The time complexity of C^{2}TSD is primarily governed by three key components: (1) Trend extraction using $M + 1$ autoregressive experts, which requires $\mathbf{O}\left(L \cdot d \cdot 2^M\right)$ operations due to causal convolutions with kernel sizes 2^m; (2) GNN encoder for modelling spatial dependencies, incurring $\mathbf{O}\left(N^2 \cdot d\right)$ for dense graphs; and (3) Diffusion processes, where both forward and reverse steps scale linearly with the number of diffusion steps T, resulting in $\mathbf{O}\left(T \cdot N \cdot L \cdot d\right)$. Additional contributions to the time complexity include $\mathbf{O}(L \log L)$ for seasonal decomposition via Fourier transforms and $\mathbf{O}\left(L^2 \cdot d + N^2 \cdot d\right)$ for spatiotemporal attention mechanisms. The overall time complexity of the method can be summarized as: $\mathbf{O}\left(d \cdot \left(L \cdot 2^M + N^2 + T \cdot N \cdot L + L^2\right)\right)$.

4.8 | Hyperparameter Analysis

4.8.1 | Maximum Noise Level

This section analyzes the sensitivity of the noise regulation parameter β_T. As shown in Fig. 6, the parameter's impact on model performance varies across datasets. The model performs robustly on AQI-36 and PEMS-BAY with $\beta_T \in [0.2, 0.4]$. Lower values (e.g., 0.1) slow the diffusion process, indicating a trade-off between noise intensity and computational efficiency. The METR-LA dataset exhibits stable performance across all tested β_T values, suggesting dataset-specific resilience. Hardware limitations restricted our investigation to $\beta_T \leq 0.4$. Based on these findings, we set $\beta_T = 0.2$ for our comparative experiments in Section 4.4. This value offers a stable balance between performance and computational efficiency.

4.8.2 | Channel Size of the Hidden State

This section analyzes the channel size d. Our results (Fig. 6) reveal dataset-specific optimal values: 16 for AQI-36 and 64 for PEMS-BAY and METR-LA, likely due to fundamental differences in dataset complexity. For the simpler AQI-36 dataset, $d = 16$

is sufficient; increasing it fails to improve performance and risks overfitting. In contrast, the complex PEMS-BAY and METR-LA datasets require $d = 64$ to capture intricate spatiotemporal dependencies, as smaller values result in a notable performance decline. These findings underscore the importance of dataset-specific parameter tuning, suggesting that d should be selected based on data dimensionality and complexity. Future work could explore adaptive mechanisms for automatically determining d.

4.8.3 | Weight of the Contrastive Loss

This section analyzes the hyperparameter α, which balances the reconstruction and contrastive losses. Proper calibration is critical, and our results (Fig. 6) show the optimal α is dataset-specific: 0.1 for AQI-36 and METR-LA, and 0.2 for PEMS-BAY. Deviating from these values leads to suboptimal performance, as it unbalances the two loss terms, highlighting the importance of careful tuning. Compared to β_T and d, α is less sensitive, exhibiting a broad stable range of $[0.05, 0.5]$. While the model maintains robust performance within this range, fine-tuning α can still yield noticeable, dataset-specific improvements. This suggests α plays a pivotal but manageable role in optimizing the model's performance by balancing its loss components.

5 | RELATED WORK

5.1 | Traditional Methods for Time Series Imputation

Various statistical and machine-learning approaches have been utilized to estimate missing values, such as ARMA[48], EM[49], and kNN[50]. These models often depend on strong assumptions regarding *temporal stationarity* and *inter-series similarity* of MTS. However, such assumptions may not adequately capture the complexities inherent in real-world MTS, potentially resulting in unsatisfactory performance in practical applications. Furthermore, low-rank matrix factorization has become a promising approach for spatiotemporal imputation by exploiting inherent spatial and temporal patterns guided by prior knowledge. Noteworthy implementations include TRMF[36] and BATF[37]. TRMF incorporates temporal dependencies within a temporally regularized matrix factorization framework, while BATF integrates domain knowledge from transportation systems into an enhanced tensor factorization model for traffic data. Additionally, TIDER[51] combines matrix factorization with the disentanglement of MTS representations.

Deep learning-based methods, particularly RNNs, GANs, and their variants, are increasingly utilized for MTSI. GRU-D[9] incorporates masking and time interval representations into its deep model architecture, employing recurrent structures to capture long-term temporal dependencies. Following this, BRITS[11] introduced a bidirectional RNN-based model designed to predict multiple correlated missing values in time series. This model effectively manages extensive correlated missing values and adapts to nonlinear dynamics, enabling a data-driven imputation approach. GAN-based methods have also demonstrated significant effectiveness in imputing missing values. These approaches allow the generator to learn the overall data distribution, predicting missing values through iterative adversarial interactions with the discriminator[38,52]. Graph neural network (GNN)-based methods have been employed to enhance the learning of spatial dependencies for imputation tasks[53,54,55]. For example, STGNN-DAE[56] utilizes power grid topology and time series data from each grid meter, capturing both spatial and temporal correlations. Another notable approach, GRIN[10], introduces a spatiotemporal encoder that integrates variable and temporal dependencies. While showing strengths in capturing spatiotemporal dependencies, recurrent methods cannot avoid the error accumulation problem. Graph-based models still exhibit lower robustness to noisy, incomplete, or unfamiliar data, as their deterministic output mechanism may not effectively adapt to input data with dynamic temporal patterns.

5.2 | Diffusion Method in MTS

Diffusion models[57,58,59] have been extensively utilized for forecasting, detecting anomalies, and imputing missing values in MTS. DDPMs[15] represent the first diffusion models applied to forecasting tasks. TimeGrad[60] introduces noise into the data at each forecasting step, then uses a backward transition kernel conditioned on past time series to progressively denoise; the hidden state, computed using a recurrent framework, is utilized to model the conditional distribution. ScoreGrad[61] targets the same distribution as TimeGrad but employs a conditional score-matching module based on SDEs, transitioning the diffusion process from discrete to continuous and replacing the diffusion steps with an integration interval. D^3VAE[62] first focuses on dealing with limited and noisy time series data using a bidirectional variational autoencoder (BVAE). Recent work has introduced graph

structure to time series tasks. DiffSTG[63] proposes a UGnet, a combination of a U-Net-based network and GNN, to process spatiotemporal dependencies. GCRDD[64] designs a graph-modified GRU, which calculates the weight matrix in GRU with a graph convolution module in the noise-matching network.

Besides, there are also some diffusion-based works focusing on MTSI, where spatiotemporal imputation is a special type. CSDI[16] firstly applied the diffusion model to MTSI, which adopts DiffWave as the noise estimation network. SSSD[41] replaces the noise estimation network with a structured state space model to capture long-term time series dependencies. PriSTI[14] deals with the imputation problem on spatiotemporal data and first considers the geographic relationship for learning spatial information among different sensors. Our model follows the pipeline of using a conditional diffusion model for spatiotemporal imputation tasks. We first focus on using the temporal disentanglement concept to redesign the processing of conditional information to improve the model's ability to capture complex and dynamic temporal dependencies in real-world scenarios. The contrastive learning strategy improves the conditional diffusion model pipeline's training stability and representation quality.

6 | CONCLUSION

We introduce C^{2}TSD, a conditional diffusion framework that integrates temporal disentanglement and contrastive learning for spatiotemporal imputation. Our model predicts missing values by employing a reverse process enhanced by contrastive learning and disentangled temporal (trend and seasonality) dependencies as key conditional information to guide this process. Our framework distinguishes itself with a unique reverse process guided by the disentanglement of temporal elements—trend and seasonality—as essential conditional representations. Besides, the contrastive learning strategy also helps learning spatiotemporal dependencies and enhances the stability and generalization of C^{2}TSD. Our model has demonstrated superior performance to a number of competitive baselines on three real-world datasets.

For future work, we will consider leveraging prior knowledge of spatiotemporal dependencies to design conditional information. Since designing the noise prediction model is still a handcrafted task, introducing inductive bias to deep learning models for spatiotemporal dependencies is worth investigating. One promising direction is integrating advanced graph learning techniques, which have shown potential in capturing complex relationships in spatial data[65,66,67]. However, incorporating advanced graphs may introduce challenges related to computational cost and model complexity, especially in large-scale scenarios. Lastly, simplifying diffusion models for large-scale applications remains an important avenue for future research.

REFERENCES

1. Wu Z, Pan S, Long G, Jiang J, Zhang C. Graph wavenet for deep spatial-temporal graph modeling. In: IJCAI. 2019:1907–1913.
2. Li Q, Tan J, Wang J, Chen H. A multimodal event-driven lstm model for stock prediction using online news. *IEEE Transactions on Knowledge and Data Engineering.* 2020;33(10):3323–3337.
3. Tedjopurnomo DA, Bao Z, Zheng B, Choudhury FM, Qin AK. A survey on modern deep neural network for traffic prediction: Trends, methods and challenges. *IEEE Transactions on Knowledge and Data Engineering.* 2020;34(4):1544–1561.
4. Di Martino F, Delmastro F. Explainable AI for clinical and remote health applications: a survey on tabular and time series data. *Artificial Intelligence Review.* 2023;56(6):5261–5315.
5. Bauer P, Thorpe A, Brunet G. The quiet revolution of numerical weather prediction. *Nature.* 2015;525(7567):47–55.
6. Du H, Du S, Li W. Probabilistic time series forecasting with deep non-linear state space models. *CAAI Transactions on Intelligence Technology.* 2023;8(1):3–13.
7. Ismail Fawaz H, Forestier G, Weber J, Idoumghar L, Muller PA. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery.* 2019;33(4):917–963.
8. Blázquez-García A, Conde A, Mori U, Lozano JA. A review on outlier/anomaly detection in time series data. *ACM Computing Surveys.* 2021;54(3):1–33.
9. Che Z, Purushotham S, Cho K, Sontag D, Liu Y. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports.* 2018;8(1):1–12.
10. Cini A, Marisca I, Alippi C. Filling the G_ap_s: Multivariate Time Series Imputation by Graph Neural Networks. In: ICLR. 2022:1–15.
11. Cao W, Wang D, Li J, Zhou H, Li L, Li Y. Brits: Bidirectional recurrent imputation for time series. *Advances in Neural Information Processing Systems.* 2018;31:1–10.
12. Wang J, Du W, Cao W, et al. Deep Learning for Multivariate Time Series Imputation: A Survey. *arXiv preprint arXiv:2402.04059.* 2024.
13. Liu Y, Yu R, Zheng S, Zhan E, Yue Y. Naomi: Non-autoregressive multiresolution sequence imputation. *Advances in Neural Information Processing Systems.* 2019;32:1–10.
14. Liu M, Huang H, Feng H, Sun L, Du B, Fu Y. PriSTI: A Conditional Diffusion Framework for Spatiotemporal Imputation. In: IEEE. 2023:1927–1939.
15. Ho J, Jain A, Abbeel P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems.* 2020;33:6840–6851.
16. Tashiro Y, Song J, Song Y, Ermon S. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems.* 2021;34:24804–24816.
17. Dhariwal P, Nichol A. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems.* 2021;34:8780–8794.

18. Nichol AQ, Dhariwal P. Improved denoising diffusion probabilistic models. In: PMLR. 2021:8162–8171.
19. Li S, Chen Y, Xiong H. Channel-aware Contrastive Conditional Diffusion for Multivariate Probabilistic Time Series Forecasting. *arXiv preprint arXiv:2410.02168.* 2024.
20. Cleveland RB, Cleveland WS, McRae JE, Terpenning I. STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics.* 1990;6(1):3–33.
21. Hyndman RJ, Athanasopoulos G. *Forecasting: principles and practice*. OTexts, 2018.
22. Bandara K, Bergmeir C, Hewamalage H. LSTM-MSNet: Leveraging forecasts on sets of related time series with multiple seasonal patterns. *IEEE Transactions on Neural Networks and Learning Systems.* 2020;32(4):1586–1599.
23. Wu H, Xu J, Wang J, Long M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In: . 34. NeurIPS. 2021.
24. Sen R, Yu HF, Dhillon IS. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In: . 32. NeurIPS. 2019.
25. Woo G, Liu C, Sahoo D, Kumar A, Hoi S. CoST: Contrastive Learning of Disentangled Seasonal-Trend Representations for Time Series Forecasting. In: ICLR. 2022:1–15.
26. Chen T, Kornblith S, Norouzi M, Hinton G. A simple framework for contrastive learning of visual representations. In: PMLR. 2020:1597–1607.
27. Kang M, Park J. Contragan: Contrastive learning for conditional image generation. *Advances in Neural Information Processing Systems.* 2020;33:21357–21369.
28. Kang M, Zhu S, Park J. Contrastive generative adversarial networks. In: . 35. AAAI. 2021:8116–8124.
29. Jeong S, Park C, Choi Y, Yoon S. Contrastive Conditional Diffusion Models for Multivariate Time Series Imputation. In: AAAI. 2023.
30. Sun Y, Li J, Xu Y, Zhang T, Wang X. Deep learning versus conventional methods for missing data imputation: A review and comparative study. *Expert Systems with Applications.* 2023:120201.
31. Zhang K, Wen Q, Zhang C, et al. Self-supervised learning for time series analysis: Taxonomy, progress, and prospects. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2024.
32. Zheng Y, Capra L, Wolfson O, Yang H. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology.* 2014;5(3):1–55.
33. Li Y, Yu R, Shahabi C, Liu Y. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In: ICLR. 2018:1–15.
34. Hersbach H. Decomposition of the continuous ranked probability score for ensemble prediction systems. *Weather and Forecasting.* 2000;15(5):559–570.
35. White IR, Royston P, Wood AM. Multiple imputation using chained equations: issues and guidance for practice. *Statistics in Medicine.* 2011;30(4):377–399.
36. Yu HF, Rao N, Dhillon IS. Temporal regularized matrix factorization for high-dimensional time series prediction. *Advances in Neural Information Processing Systems.* 2016;29:1–9.
37. Chen X, He Z, Chen Y, Lu Y, Wang J. Missing traffic data imputation and pattern discovery with a Bayesian augmented tensor factorization model. *Transportation Research Part C: Emerging Technologies.* 2019;104:66–77.
38. Yoon J, Jordon J, Schaar M. Gain: Missing data imputation using generative adversarial nets. In: PMLR. 2018:5689–5698.
39. Mulyadi AW, Jun E, Suk HI. Uncertainty-aware variational-recurrent imputation network for clinical time series. *IEEE Transactions on Cybernetics.* 2021;52(9):9684–9694.
40. Fortuin V, Baranchuk D, Rätsch G, Mandt S. Gp-vae: Deep probabilistic time series imputation. In: PMLR. 2020:1651–1661.
41. Alcaraz JL, Strodthoff N. Diffusion-based Time Series Imputation and Forecasting with Structured State Space Models. *Transactions on Machine Learning Research.* 2022.
42. Zhang S, Wang S, Miao H, Chen H, Fan C, Zhang J. Score-CDM: Score-weighted convolutional diffusion model for multivariate time series imputation. *arXiv preprint arXiv:2405.13075.* 2024.
43. Nie T, Qin G, Ma W, Mei Y, Sun J. ImputeFormer: Low rankness-induced transformers for generalizable spatiotemporal imputation. In: ACM. 2024:2260–2271.
44. Yi X, Zheng Y, Zhang J, Li T. ST-MVL: filling missing values in geo-sensory time series data. In: IJCAI. 2016:1–7.
45. Bracher J, Ray EL, Tibshirani RJ, Reich NG. Evaluating epidemic forecasts in an interval format. *PLoS Computational Biology.* 2021;17(2):e1008618.
46. Rasul K, Seward C, Schuster I, Vollgraf R. Autoregressive Denoising Diffusion Models for Multivariate Probabilistic Time Series Forecasting. In: 2021.
47. Gneiting T, Raftery AE. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association.* 2007;102(477):359–378.
48. Ansley CF, Kohn R. On the estimation of ARIMA models with missing values. In: Parzen E., ed. *Time Series Analysis of Irregularly Observed Data*Springer. 1984:9–37.
49. Shumway RH, Stoffer DS. An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis.* 1982;3(4):253–264.
50. Hastie T, Tibshirani R, Friedman JH, Friedman JH. *The elements of statistical learning: data mining, inference, and prediction*. 2. Springer, 2009.
51. Liu S, Li X, Cong G, Chen Y, Jiang Y. Multivariate Time-series Imputation with Disentangled Temporal Representations. In: ICLR. 2022:1–15.
52. Luo Y, Zhang Y, Cai X, Yuan X. E2gan: End-to-end generative adversarial network for multivariate time series imputation. In: AAAI Press. 2019:3094–3100.
53. Chen Y, Li Z, Yang C, Wang X, Long G, Xu G. Adaptive graph recurrent network for multivariate time series imputation. In: Springer Nature Singapore Singapore. 2022:64–73.
54. Chen Y, Shi K, Wang X, Xu G. MTSTI: A multi-task learning framework for spatiotemporal imputation. In: Springer Nature Switzerland Cham. 2023:180–194.
55. Chen Y, Hu R, Li Z, et al. Exploring explicit and implicit graph learning for multivariate time series imputation. *Engineering Applications of Artificial Intelligence.* 2024;127:107217.
56. Kuppannagari SR, Fu Y, Chueng CM, Prasanna VK. Spatio-temporal missing data imputation for smart power grids. In: ACM. 2021:458–465.
57. Wu Z, Cao L, Qi L. eVAE: Evolutionary Variational Autoencoder. *IEEE Transactions on Neural Networks and Learning Systems.* 2024.

58. Wu Z, Zhang Q, Zhou J, Chen H, Liu Y. WAVAE: A Weakly Augmented Variational Autoencoder for Time Series Anomaly Detection. *Information Fusion.* 2025:103462.
59. In: .
60. Rasul K, Seward C, Schuster I, Vollgraf R. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In: PMLR. 2021:8857–8868.
61. Yan T, Zhang H, Zhou T, Zhan Y, Xia Y. ScoreGrad: Multivariate probabilistic time series forecasting with continuous energy-based generative models. *arXiv preprint arXiv:2106.10121.* 2021.
62. Li Y, Lu X, Wang Y, Dou D. Generative time series forecasting with diffusion, denoise, and disentanglement. *Advances in Neural Information Processing Systems.* 2022;35:23009–23022.
63. Wen H, Lin Y, Xia Y, et al. Diffstg: Probabilistic spatio-temporal graph forecasting with denoising diffusion models. In: ACM. 2023:1–12.
64. Li R, Li X, Gao S, Choy SB, Gao J. Graph convolution recurrent denoising diffusion model for multivariate probabilistic temporal forecasting. In: Springer. 2023:661–676.
65. Li J, Zheng R, Feng H, Li M, Zhuang X. Permutation equivariant graph framelets for heterophilous graph learning. *IEEE Transactions on Neural Networks and Learning Systems.* 2024.
66. Li M, Micheli A, Wang YG, et al. Guest editorial: Deep neural networks for graphs: Theory, models, algorithms, and applications. *IEEE Transactions on Neural Networks and Learning Systems.* 2024;35(4):4367–4372.
67. Bai L, Cui L, Wang Y, et al. Haqjsk: Hierarchical-aligned quantum jensen-shannon kernels for graph classification. *IEEE Transactions on Knowledge and Data Engineering.* 2024.
68. Qiu J, Jammalamadaka SR, Ning N. Multivariate Bayesian Structural Time Series Model.. *J. Mach. Learn. Res..* 2018;19(1):2744–2776.
69. Lin L, Li Z, Li R, Li X, Gao J. Diffusion models for time-series applications: a survey. *Frontiers of Information Technology & Electronic Engineering.* 2023:1–23.
70. Wen Q, Gao J, Song X, Sun L, Xu H, Zhu S. RobustSTL: A robust seasonal-trend decomposition algorithm for long time series. In: . 33. AAAI. 2019:5409–5416.