# Generative Time Series Forecasting with Diffusion, Denoise and Disentanglement

**Yan Li**[§†*], **Xinjiang Lu**[† ✉], **Yaqing Wang**[†], **Dejing Dou**[†]
[†]Business Intelligence Lab, Baidu Research
[§]Zhejiang University, China
ly21121@zju.edu.cn, {luxinjiang,wangyaqing01,doudejing}@baidu.com

## Abstract

Time series forecasting has been a widely explored task that is of great importance in many applications. However, it is common that real-world time series data are recorded in a short time period, which results in a big gap between the deep model and the limited and noisy time series. In this work, we propose to address the time series forecasting problem with generative modeling and propose a bidirectional variational auto-encoder (BVAE) equipped with diffusion, denoise, and disentanglement, namely $D^3$VAE. Specifically, a coupled diffusion probabilistic model is proposed to augment the time series data without increasing the aleatoric uncertainty contributing to the data and implement a more tractable inference process with the BVAE. To ensure the generated series move toward the true target, we further propose to adapt and integrate the multiscale denoising score matching into the diffusion process for time series forecasting. In addition, to enhance the interpretability and stability of the prediction, we treat the latent variable in a multivariate manner and disentangle them on top of minimizing total correlation. Extensive experiments on both synthetic and real-world data show that $D^3$VAE outperforms competitive algorithms with remarkable margins. Our implementation is available at `https://github.com/ramber1836/d3vae`.

## 1 Introduction

Time series forecasting is of great importance for risk-averse and decision-making. Traditional RNN-based methods capture temporal dependencies of the time series to predict the future. Long short-term memories (LSTMs) and gated recurrent units (GRUs) [105, 35, 32, 80] introduce the gate functions into the cell structure to handle long-term dependencies effectively. The models based on convolutional neural networks (CNNs) capture the complex inner patterns of the time series through convolutional operations [60, 8, 7]. Recently, the Transformer-based models have shown great performance in time series forecasting [101, 106, 55, 61] with the help of multi-head self-attention. However, one big issue of neural networks in time series forecasting is the uncertainty [31, 1] resulting from the properties of the deep structure. The models based on vector autoregression (VAR) [12, 24, 51] try to model the distribution of time series from hidden states which could provide more reliability to the prediction, while the performance is not satisfactory [58].

Interpretable representation learning is another merit of time series forecasting. Variational auto-encoders (VAEs) have shown not only the superiority in modeling latent distributions of the data and reducing the gradient noise [75, 54, 63, 89] but also the interpretability of time series forecasting [26, 27]. However, the interpretability of VAEs might be inferior due to the entangled latent variables.

[*]This work was done when the first author was an intern at Baidu Research under the supervision of the second author.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

There have been efforts to learn representation disentangling [50, 6, 39], which show that the well-disentangled representation can improve the performance and robustness of the algorithm.

Moreover, real-world time series are often noisy and recorded in a short time period, which may result in overfitting and poor generalization issues [30, 81, 107, 93][1]. To this end, we address the time series forecasting problem with generative modeling. Specifically, we propose a bidirectional variational auto-encoder (BVAE) equipped with diffusion, denoise, and disentanglement, namely $D^3$VAE. More specifically, we first propose a coupled diffusion probabilistic model to remedy the limitation of time series data by augmenting the input time series, as well as the output time series, inspired by the forward process of the diffusion model [85, 41, 70, 74]. Besides, we adapt the Nouveau VAE [89] to the time series forecasting task and develop a BVAE as a substitute for the reverse process of the diffusion model. In this way, the expressiveness of the diffusion model plus the tractability of the VAE can be leveraged together for generative time series forecasting. Though the merit of generalizability is helpful, the diffused samples might be corrupted, which results in a generative model moving toward the noisy target. Therefore, we further develop a scaled denoising score-matching network for cleaning diffused target time series. In addition, we disentangle the latent variables of the time series by assuming that different disentangled dimensions of the latent variables correspond to different temporal patterns (such as trend, seasonality, etc.). Our contributions can be summarized as:

- We propose a coupled diffusion probabilistic model aiming to reduce the aleatoric uncertainty of the time series and improve the generalization capability of the generative model.
- We integrate the multiscale denoising score matching into the coupled diffusion process to improve the accuracy of generated results.
- We disentangle the latent variables of the generative model to improve the interpretability for time series forecasting.
- Extensive experiments on both synthetic datasets and real-world datasets demonstrate that $D^3$VAE outperforms competitive baselines with satisfactory margins.

## 2 Methodology

### 2.1 Generative Time Series Forecasting

**Problem Formulation.** Given an input multivariate time series $X = \{x_1, x_2, \cdots, x_n \,|\, x_i \in \mathbb{R}^d\}$ and the corresponding target time series $Y = \{y_{n+1}, y_{n+2}, \cdots, y_{n+m} \,|\, y_j \in \mathbb{R}^{d'}\}$ ($d' \leq d$). We assume that $Y$ can be generated from latent variables $Z \in \Omega_Z$ that can be drawn from the Gaussian distribution $Z \sim p(Z|X)$. The latent distribution can be further formulated as $p_\phi(Z|X) = g_\phi(X)$ where $g_\phi$ denotes a nonlinear function. Then, the data density of the target series is given by:

$$p_\theta(Y) = \int_{\Omega_Z} p_\phi(Z|X)(Y - f_\theta(Z))dZ\,, \tag{1}$$

where $f_\theta$ denotes a parameterized function. The target time series can be obtained directly by sampling from $p_\theta(Y)$.

In our problem setting, time series forecasting is to learn the representation $Z$ that captures useful signals of $X$, and map the low dimensional $X$ to the latent space with high expressiveness. The framework overview of $D^3$VAE is demonstrated in Fig. 1. Before diving into the detailed techniques, we first introduce a preliminary proposition.

**Proposition 1.** *Given a time series $X$ and its inherent noise $\epsilon_X$, we have the decomposition: $X = \langle X_r, \epsilon_X \rangle$, where $X_r$ is the ideal time series data without noise. $X_r$ and $\epsilon_X$ are independent of each other. Let $p_\phi(Z|X) = p_\phi(Z|X_r, \epsilon_X)$, the estimated target series $\widehat{Y}$ can be generated with the distribution $p_\theta(\widehat{Y}|Z) = p_\theta(\widehat{Y}_r|Z) \cdot p_\theta(\epsilon_{\widehat{Y}}|Z)$ where $\widehat{Y}_r$ is the ideal part of $\widehat{Y}$ and $\epsilon_{\widehat{Y}}$ is the estimation noise. Without loss of generality, $\widehat{Y}_r$ can be fully captured by the model. That is, $\|Y_r - \widehat{Y}_r\| \longrightarrow 0$ where $Y_r$ is the ideal part of ground truth target series $Y$. In addition, $Y$ can be decomposed as $Y = \langle \widehat{Y}_r, \epsilon_Y \rangle$ ($\epsilon_Y$ denotes the noise of $Y$). Therefore, the error between ground truth and prediction, i.e., $\|Y - \widehat{Y}\| = \|\epsilon_Y - \epsilon_{\widehat{Y}}\| > 0$, can be deemed as the combination of aleatoric uncertainty and epistemic uncertainty.*

[1] Detailed literature review can be found in Appendix A.

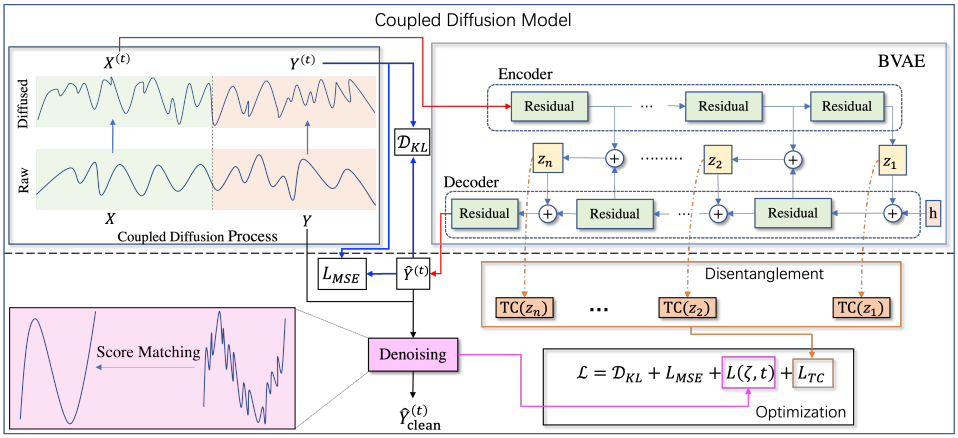



Figure 1: The framework overview of D$^3$VAE. First, both the input and output series are augmented simultaneously with the *coupled diffusion process*. Then the diffused input series are fed into a proposed BVAE model for inference, which can be deemed as a *reverse process*. To make the estimated target move toward the true target series, a denoising score-matching mechanism is further applied. Meanwhile, the latent states in BVAE are leveraged for disentangling such that the model interpretability and reliability can be improved. .

### 2.2 Coupled Diffusion Probabilistic Model

The diffusion probabilistic model (diffusion model for brevity) is a family of latent variable models aiming to generate high-quality samples. To equip the generative time series forecasting model with high expressiveness, a coupled *forward process* is developed to augment the input series and target series synchronously. Besides, in the forecasting task, a more tractable and accurate prediction is expected. To achieve this, we propose a bidirectional variational auto-encoder (BVAE) to take the place of the *reverse process* in the diffusion model. We present the technical details in the following two parts, respectively.

#### 2.2.1 Coupled Diffusion Process

The forward diffusion process is fixed to a Markov chain that gradually adds Gaussian noise to the data [85, 41]. To diffuse the input and output series, we propose a coupled diffusion process, which is demonstrated in Fig. 2. Specifically, given the input $X = X^{(0)} \sim q(X^{(0)})$, the approximate posterior $q(X^{(1:T)}|X^{(0)})$ can be obtained as

$$q(X^{(1:T)}|X^{(0)}) = \prod_{t=1}^{T} q(X^{(t)}|X^{(t-1)}), \quad q(X^{(t)}|X^{(t-1)}) = \mathcal{N}(X^{(t)}; \sqrt{1-\beta_t}X^{(t)}, \beta_t I), \tag{2}$$

where a uniformly increasing variance schedule $\boldsymbol{\beta} = \{\beta_1, \cdots, \beta_T \,|\, \beta_t \in [0,1)\}$ is employed to control the level of noise to be added. Then, let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$, we have

$$q(X^{(t)}|X^{(0)}) = \mathcal{N}(X^{(t)}; \sqrt{\bar{\alpha}_t}X^{(0)}, (1-\bar{\alpha}_t)I). \tag{3}$$

Furthermore, according to Proposition 1 we decompose $X^{(0)}$ as $X^{(0)} = \langle X_r, \epsilon_X \rangle$. Then, with Eq. (3), the diffused $X^{(t)}$ can be decomposed as follows:

$$X^{(t)} = \sqrt{\bar{\alpha}_t}X^{(0)} + (1-\bar{\alpha}_t)\delta_X := \langle \underbrace{\sqrt{\bar{\alpha}_t}X_r}_{\text{ideal part}}, \underbrace{\sqrt{\bar{\alpha}_t}\epsilon_X + (1-\bar{\alpha}_t)\delta_X}_{\text{noisy part}} \rangle, \tag{4}$$

where $\delta_X$ denotes the standard Gaussian noise of $X$. As $\boldsymbol{\alpha}$ can be determined when the variance schedule $\boldsymbol{\beta}$ is known, the ideal part is also determined in the diffusion process. Let $\widetilde{X}_r^{(t)} = \sqrt{\bar{\alpha}_t}X_r$ and $\delta_{\widetilde{X}}^{(t)} = \sqrt{\bar{\alpha}_t}\epsilon_X + (1-\bar{\alpha}_t)\delta_X$, then, according to Proposition 1 and Eq. (4), we have

$$p_\phi(Z^{(t)}|X^{(t)}) = p_\phi(Z^{(t)}|\widetilde{X}_r^{(t)}, \delta_{\widetilde{X}}^{(t)}), \quad p_\theta(\widehat{Y}^{(t)}|Z^{(t)}) = p_\theta(\widehat{Y}_r^{(t)}|Z^{(t)})p_\theta(\delta_{\widehat{Y}}^{(t)}|Z^{(t)}), \tag{5}$$

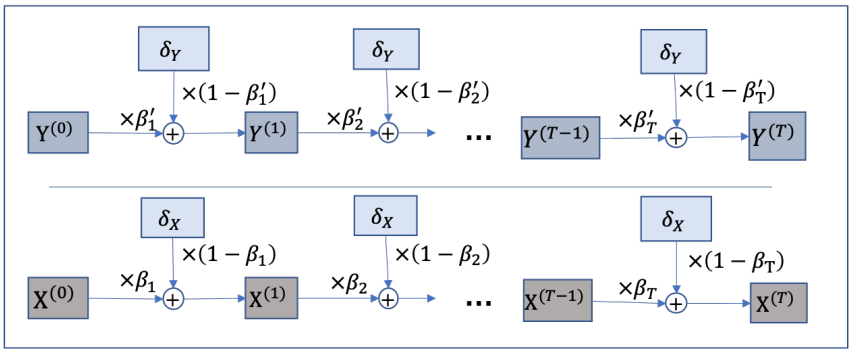



Figure 2: An illustration of the coupled diffusion process. The input $X^{(0)}$ and the corresponding target $Y^{(0)}$ are diffused simultaneously with different variance schedules. $\boldsymbol{\beta} = \{\beta_1, \cdots, \beta_T]$ is the diffusion schedule for the input and $\boldsymbol{\beta}' = [\beta'_1, \cdots, \beta'_T]$ is for the target.

where $\delta^{(t)}_{\widehat{Y}}$ denotes the generated noise of $\widehat{Y}^{(t)}$. To relieve the effect of aleatoric uncertainty resulting from time series data, we further apply the diffusion process to the target series $Y = Y^{(0)} \sim q(Y^{(0)})$. In particular, a scale parameter $\omega \in (0, 1)$ is adopted, such that $\beta'_t = \omega\beta_t, \alpha'_t = 1 - \beta'_t$ and $\bar{\alpha}'_t = \prod_{s=1}^{t} \alpha'_s$. Then, according to Proposition 1, we can obtain the following decomposition (similar to Eq. (4)):

$$Y^{(t)} = \sqrt{\bar{\alpha}'_t} Y^{(0)} + (1 - \bar{\alpha}'_t)\delta_Y := \langle \underbrace{\sqrt{\bar{\alpha}'_t} Y_r}_{\text{ideal part}}, \underbrace{\sqrt{\bar{\alpha}'_t}\epsilon_Y + (1 - \bar{\alpha}'_t)\delta_Y}_{\text{noisy part}} \rangle = \langle \widetilde{Y}^{(t)}_r, \delta^{(t)}_{\widetilde{Y}} \rangle . \tag{6}$$

Consequently, we have $q(Y^{(t)}) = q(\widetilde{Y}^{(t)}_r) q(\delta^{(t)}_{\widetilde{Y}})$. Afterward, with Proposition 1 and Eqs. (5) and (6) we can draw the following conclusions. The proofs can be found in Appendix B.

**Lemma 1.** *$\forall \varepsilon > 0$, there exists a probabilistic model $f_{\phi,\theta} := (p_\phi, p_\theta)$ to guarantee that $\mathcal{D}_{\mathrm{KL}}(q(\widetilde{Y}^{(t)}_r) || p_\theta(\widehat{Y}^{(t)}_r)) < \varepsilon$, where $\widehat{Y}^{(t)}_r = f_{\phi,\theta}(X^{(t)})$.*

**Lemma 2.** *With the coupled diffusion process, the difference between diffusion noise and generation noise will be reduced, i.e., $\lim_{t\to\infty} \mathcal{D}_{\mathrm{KL}}(q(\delta^{(t)}_{\widetilde{Y}}) || p_\theta(\delta^{(t)}_{\widehat{Y}} | Z^{(t)})) < \mathcal{D}_{\mathrm{KL}}(q(\epsilon_Y) || p_\theta(\epsilon_{\widehat{Y}}))$ .*

Therefore, the uncertainty raised by the generative model and the inherent data noise can be reduced through the coupled diffusion process. In addition, the diffusion process augments the input series, as well as the target series, simultaneously, which can improve the generalization capability for (esp. short) time series forecasting.

#### 2.2.2 Bidirectional Variational Auto-Encoder

Traditionally, in the diffusion model, a reverse process is adopted to generate high-quality samples [85, 41]. However, for the generative time series forecasting problem, not only the expressiveness but also the supervision of the ground truths should be considered. In this work, we employ a more efficient generative model, i.e., bidirectional variational auto-encoder (BVAE) [89], to take the place of the reverse process of the diffusion model. The architecture of BVAE is described in Fig. 1 where $Z$ is treated in a multivariate fashion $Z = \{z_1, \cdots, z_n\}$ ($z_i \in \mathbb{R}^m, z_i = [z_{i,1}, \cdots, z_{i,m}]$) and $z_{i+1} \sim p(z_{i+1}|z_i, X)$. Then, $n$ is determined in accordance with the number of residual blocks in the encoder, as well as the decoder. Another merit of BVAE is that it opens an interface to integrate the disentanglement for improving model interpretability (refer to Section 2.4).

### 2.3 Scaled Denoising Score Matching for Diffused Time Series Cleaning

Although the time series data can be augmented with the aforementioned coupled diffusion probabilistic model, the generative distribution $p_\theta(\widehat{Y}^{(t)})$ tends to move toward the diffused target series $Y^{(t)}$ which has been corrupted [65, 87]. To further "clean" the generated target series, we employ the Denoising Score Matching (DSM) to accelerate the de-uncertainty process without sacrificing the

model flexibility. DSM [90, 65] was proposed to link Denoising Auto-Encoder (DAE) [91] to Score Matching (SM) [43]. Let $\widehat{Y}$ denote the generated target series, then we have the objective

$$L_{\text{DSM}}(\zeta) = \mathbb{E}_{p_{\sigma_0}(\widehat{Y},Y)} \|\nabla_{\widehat{Y}} \log(q_{\sigma_0}(\widehat{Y}|Y)) + \nabla_{\widehat{Y}} E(\widehat{Y};\zeta)\|^2 \,, \tag{7}$$

where $p_{\sigma_0}(\widehat{Y}, Y)$ is the joint density of pairs of corrupted and clean samples $(\widehat{Y}, Y)$, $\nabla_{\widehat{Y}} \log(q_{\sigma_0}(\widehat{Y}|Y))$ is the derivative of log density of the single noise kernel which is dedicated to replacing the Parzen density estimator: $p_{\sigma_0}(\widehat{Y}) = \int q_{\sigma_0}(\widehat{Y}|Y)p(Y)dY$ in score matching, and $E(\widehat{Y};\zeta)$ is the energy function. In the particular case of Gaussian noise, $\log(q_{\sigma_0}(\widehat{Y}|Y)) = -(\widehat{Y} - Y)^2/2{\sigma_0}^2 + C$. Thus, we have

$$L_{\text{DSM}}(\zeta) = \mathbb{E}_{p_{\sigma_0}(\widehat{Y},Y)} \|Y - \widehat{Y} + \sigma_0^2 \nabla_{\widehat{Y}} E(\widehat{Y};\zeta)\|^2 \,. \tag{8}$$

Then, for the diffused target series at step $t$, we can obtain

$$L_{\text{DSM}}(\zeta, t) = \mathbb{E}_{p_{\sigma_0}(\widehat{Y}^{(t)},Y)} \|Y - \widehat{Y}^{(t)} + \sigma_0^2 \nabla_{\widehat{Y}^{(t)}} E(\widehat{Y}^{(t)};\zeta)\|^2 \,. \tag{9}$$

To scale the noise of different levels [65], a monotonically decreasing series of fixed $\sigma$ values $\{\sigma_1, \cdots, \sigma_T \,|\, \sigma_t = 1 - \bar{\alpha}_t\}$ (refer to the aforementioned variance schedule $\boldsymbol{\beta}$ in Section 2.2) is adopted. Therefore, the objective of the multi-scaled DSM is

$$L(\zeta, t) = \mathbb{E}_{q_\sigma(\widehat{Y}^{(t)}|Y)p(Y)} l(\sigma_t) \|Y - \widehat{Y}^{(t)} + \sigma_0^2 \nabla_{\widehat{Y}^{(t)}} E(\widehat{Y}^{(t)};\zeta)\|^2 \,, \tag{10}$$

where $\sigma \in \{\sigma_1, \cdots, \sigma_T\}$ and $l(\sigma_t) = \sigma_t$. With Eq. (10), we can ensure that the gradient has the right magnitude by setting $\sigma_0$.

Note that, in the generative time series forecasting setting, the generated samples will be tested without applying the diffusion process. To further denoise the generated target series $\widehat{Y}$, we apply a single-step gradient denoising jump [78]:

$$\widehat{Y}_{\text{clean}} = \widehat{Y} - \sigma_0^2 \nabla_{\widehat{Y}} E(\widehat{Y};\zeta) \,. \tag{11}$$

The generated results tend to possess larger distribution space than the true target, and the noisy term in Eq. (11) approximates the noise between the generated target series and the "cleaned" target series. Therefore, $\sigma_0^2 \nabla_{\widehat{Y}} E(\widehat{Y};\zeta)$ can be treated as the estimated uncertainty of the prediction.

### 2.4 Disentangling Latent Variables for Interpretation

The interpretability of the time series forecasting model is of great importance for many downstream tasks [88, 38, 44]. Through disentangling the latent variables of the generative model, not only the interpretability but also the reliability of the prediction can be further enhanced [64].

To disentangle the latent variables $Z = \{z_1, \cdots, z_n\}$, we attempt to minimize the Total Correlation (TC) [94, 50], which is a popular metric to measure dependencies among multiple random variables,

$$\text{TC}(z_i) = \mathcal{D}_{\text{KL}}(p_\phi(z_i) || \bar{p}_\phi(z_i)), \qquad \bar{p}_\phi(z_i) = \prod_{j=1}^{m} p_\phi(z_{i,j}) \tag{12}$$

where $m$ denotes the number of factors of $z_i$ that need to be disentangled. In general, lower TC means better disentanglement if the latent variables preserve useful information. However, a very low TC can still be obtained when the latent variables carry no meaningful signals. Through the bidirectional structure of BVAE, such issues can be tackled without too much effort. As shown in Fig. 1, the signals are disseminated in both the encoder and decoder, such that rich semantics are aggregated into the latent variables. Furthermore, to alleviate the effect of potential irregular values, we average the total correlations of $z_{1:n}$, then the loss w.r.t. the TC score of BVAE can be obtained:

$$L_{\text{TC}} = \frac{1}{n} \sum_{i=1}^{n} \text{TC}(z_i) \,. \tag{13}$$

**Algorithm 1** Training Procedure.

**repeat**
  $X^{(0)} \sim q(X^{(0)}), \quad Y^{(0)} \sim q(Y^{(0)}), \quad \delta_X \sim N(0, I_d), \quad \delta_Y \sim N(0, I_d)$
  Randomly choose $t \in \{1, \cdots, T\}$ and with Eqs. (4) and (6),
    $X^{(t)} = \sqrt{\bar{\alpha}_t} X^{(0)} + (1 - \bar{\alpha}_t)\delta_X, \quad Y^{(t)} = \sqrt{\bar{\alpha}'_t} Y^{(0)} + (1 - \bar{\alpha}'_t)\delta_Y$
  Generate the latent variable $Z$ with BVAE, $Z \sim p_\phi(Z|X^{(t)})$
  Sample $\widehat{Y}^{(t)} \sim p_\theta(\widehat{Y}^{(t)}|Z)$ and calculate $\mathcal{D}_{\mathrm{KL}}(q(Y^{(t)})||p_\theta(\widehat{Y}^{(t)}))$
  Calculate DSM loss with Eq. (10)
  Calculate total correlation of $Z$ with Eq. (13)
  Construct the total loss $\mathcal{L}$ with Eq. (14)
  $\theta, \phi \leftarrow \mathrm{argmin}(\mathcal{L})$
**until** Convergence

**Algorithm 2** Forecasting Procedure.

**Input:** $X \sim q(X)$
Sample $Z \sim p_\phi(Z|X)$
Generate $\widehat{Y} \sim p_\theta(\widehat{Y}|Z)$
**Output:** $\widehat{Y}_{\text{clean}}$ and the estimated uncertainty with Eq. (11)

### 2.5 Training and Forecasting

**Training Objective.** To reduce the effect of uncertainty, the coupled diffusion equipped with the denoising network is proposed without sacrificing too much generalizability. Then we disentangle the latent variables of the generative model by minimizing the TC of the latent variables. Finally, we reconstruct the loss with several trade-off parameters, and with Eqs. (10), (11) and (13) we have

$$\mathcal{L} = \psi \cdot \mathcal{D}_{\mathrm{KL}}(q(Y^{(t)})||p_\theta(\widehat{Y}^{(t)})) + \lambda \cdot L(\zeta, t) + \gamma \cdot L_{\mathrm{TC}} + L_{\mathrm{mse}}(\widehat{Y}^{(t)}, Y^{(t)}), \tag{14}$$

where $L_{\mathrm{mse}}$ calculates the mean square error (MSE) between $\widehat{Y}^{(t)}$ and $Y^{(t)}$. We minimize the above objective to learn the generative model accordingly.

**Algorithms.** Algorithm 1 displays the complete training procedure of D$^3$VAE with the loss function in Eq. (14). For inference, as described in Algorithm 2, given the input series $X$, the target series can be generated directly from the distribution $p_\theta$ which is conditioned on the latent states drawn from distribution $p_\phi$.

## 3 Experiments

### 3.1 Experiment Settings

**Datasets.** We generate two synthetic datasets suggested by [23],

$$w_t = a \cdot w_{t-1} + \tanh(b \cdot w_{t-2}) + \sin(w_{t-3}) + \mathcal{N}(0, 0.5I)$$
$$X = [w_1, w_2, ..., w_N] \cdot F + \mathcal{N}(0, 0.5I),$$

where $w_t \in \mathbb{R}^2$ and $0 \leq w_{t,1}, w_{t,2} \leq 1$ $(t = 1, 2, 3)$, $F \in \mathbb{R}^{2\times k} \sim \mathcal{U}[-1, 1]$, $k$ denotes the dimensionality and $N$ is the number of time points, $a, b$ are two constants. We set $a = 0.9, b = 0.2, k = 20$ to generate $\mathrm{D}_1$, and $a = 0.5, b = 0.5, k = 40$ for $\mathrm{D}_2$, and $N = 800$ for both $\mathrm{D}_1$ and $\mathrm{D}_2$.

Six real-world datasets with diverse spatiotemporal dynamics are selected, including Traffic [58], Electricity[2], Weather[3], Wind (Wind Power) [4], and ETTs [106] (ETTm1 and ETTh1). To highlight the uncertainty in short time series scenarios, for each dataset, we slice a subset from the starting point to make sure that each sliced dataset contains at most 1000 time points. Subsequently, we

[2]https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014
[3]https://www.bgc-jena.mpg.de/wetter/
[4]The dataset is available at https://github.com/ramber1836/d3vae/data.

Table 1: Performance comparisons on synthetic data in terms of MSE and CRPS. The best results are boldfaced.

| Model | | D$^3$VAE | NVAE | $\beta$-TCVAE | f-VAE | DeepAR | TimeGrad | GP-copula | VAE |
|---|---|---|---|---|---|---|---|---|---|
| $D_1$ | 8 | **0.512**$_{\pm.033}$ | $1.201_{\pm.027}$ | $0.631_{\pm.003}$ | $0.854_{\pm.099}$ | $1.153_{\pm.125}$ | $0.966_{\pm.102}$ | $1.202_{\pm.108}$ | $0.912_{\pm.132}$ |
| | | **0.585**$_{\pm.021}$ | $0.905_{\pm.011}$ | $0.658_{\pm.002}$ | $0.745_{\pm.036}$ | $0.758_{\pm.038}$ | $0.698_{\pm.024}$ | $0.773_{\pm.033}$ | $0.786_{\pm.053}$ |
| | 16 | **0.571**$_{\pm.025}$ | $1.184_{\pm.025}$ | $0.758_{\pm.047}$ | $1.046_{\pm.270}$ | $0.911_{\pm.046}$ | $0.945_{\pm.315}$ | $0.915_{\pm.059}$ | $0.908_{\pm.177}$ |
| | | **0.625**$_{\pm.013}$ | $0.897_{\pm.012}$ | $0.747_{\pm.027}$ | $0.835_{\pm.108}$ | $0.699_{\pm.014}$ | $0.709_{\pm.100}$ | $0.704_{\pm.020}$ | $0.765_{\pm.067}$ |
| $D_2$ | 8 | **0.599**$_{\pm.049}$ | $1.966_{\pm.047}$ | $3.096_{\pm.197}$ | $3.353_{\pm.430}$ | $0.977_{\pm.137}$ | $0.963_{\pm.385}$ | $1.037_{\pm.082}$ | $3.079_{\pm.345}$ |
| | | **0.628**$_{\pm.027}$ | $1.255_{\pm.021}$ | $1.680_{\pm.062}$ | $1.640_{\pm.154}$ | $0.727_{\pm.058}$ | $0.706_{\pm.123}$ | $0.753_{\pm.026}$ | $1.504_{\pm.098}$ |
| | 16 | **0.786**$_{\pm.041}$ | $1.955_{\pm.051}$ | $3.067_{\pm.443}$ | $3.109_{\pm.428}$ | $0.972_{\pm.144}$ | $0.850_{\pm.061}$ | $1.082_{\pm.071}$ | $3.132_{\pm.160}$ |
| | | $0.728_{\pm.026}$ | $1.251_{\pm.020}$ | $1.643_{\pm.183}$ | $1.558_{\pm.157}$ | $0.720_{\pm.050}$ | **0.649**$_{\pm.017}$ | $0.762_{\pm.008}$ | $1.560_{\pm.060}$ |

Table 2: The performance comparisons on real-world datasets in terms of MSE and CRPS, and the best results are in boldface.

| Model | | D$^3$VAE | NVAE | $\beta$-TCVAE | f-VAE | DeepAR | TimeGrad | GP-copula | VAE |
|---|---|---|---|---|---|---|---|---|---|
| Traffic | 8 | **0.081**$_{\pm.003}$ | $1.300_{\pm.024}$ | $1.003_{\pm.006}$ | $0.982_{\pm.059}$ | $3.895_{\pm.306}$ | $3.695_{\pm.246}$ | $4.299_{\pm.372}$ | $0.794_{\pm.130}$ |
| | | **0.207**$_{\pm.003}$ | $0.593_{\pm.004}$ | $0.894_{\pm.003}$ | $0.666_{\pm.032}$ | $1.391_{\pm.071}$ | $1.410_{\pm.027}$ | $1.408_{\pm.046}$ | $0.759_{\pm.07}$ |
| | 16 | **0.081**$_{\pm.009}$ | $1.271_{\pm.019}$ | $0.997_{\pm.004}$ | $0.998_{\pm.042}$ | $4.141_{\pm.320}$ | $3.495_{\pm.362}$ | $4.575_{\pm.141}$ | $0.632_{\pm.057}$ |
| | | **0.200**$_{\pm.014}$ | $0.589_{\pm.001}$ | $0.893_{\pm.002}$ | $0.692_{\pm.026}$ | $1.338_{\pm.043}$ | $1.329_{\pm.057}$ | $1.506_{\pm.025}$ | $0.671_{\pm.038}$ |
| Electricity | 8 | **0.251**$_{\pm.015}$ | $1.134_{\pm.029}$ | $0.901_{\pm.052}$ | $0.893_{\pm.069}$ | $2.934_{\pm.173}$ | $2.703_{\pm.087}$ | $2.924_{\pm.218}$ | $0.853_{\pm.040}$ |
| | | **0.398**$_{\pm.011}$ | $0.542_{\pm.003}$ | $0.831_{\pm.004}$ | $0.809_{\pm.024}$ | $1.244_{\pm.037}$ | $1.208_{\pm.024}$ | $1.249_{\pm.048}$ | $0.795_{\pm.016}$ |
| | 16 | **0.308**$_{\pm.030}$ | $1.150_{\pm.032}$ | $0.850_{\pm.003}$ | $0.807_{\pm.034}$ | $2.803_{\pm.199}$ | $2.770_{\pm.237}$ | $3.065_{\pm.186}$ | $0.846_{\pm.062}$ |
| | | **0.437**$_{\pm.020}$ | $0.531_{\pm.003}$ | $0.814_{\pm.002}$ | $0.782_{\pm.024}$ | $1.220_{\pm.048}$ | $1.240_{\pm.048}$ | $1.307_{\pm.042}$ | $0.793_{\pm.029}$ |
| Weather | 8 | **0.169**$_{\pm.022}$ | $0.801_{\pm.024}$ | $0.234_{\pm.042}$ | $0.591_{\pm.198}$ | $2.317_{\pm.357}$ | $2.715_{\pm.189}$ | $2.412_{\pm.761}$ | $0.560_{\pm.192}$ |
| | | **0.357**$_{\pm.024}$ | $0.757_{\pm.013}$ | $0.404_{\pm.040}$ | $0.565_{\pm.080}$ | $0.858_{\pm.078}$ | $0.920_{\pm.013}$ | $0.897_{\pm.115}$ | $0.572_{\pm.077}$ |
| | 16 | **0.187**$_{\pm.047}$ | $0.811_{\pm.016}$ | $0.212_{\pm.012}$ | $0.530_{\pm.167}$ | $1.269_{\pm.187}$ | $1.110_{\pm.083}$ | $1.357_{\pm.145}$ | $0.424_{\pm.141}$ |
| | | **0.361**$_{\pm.046}$ | $0.759_{\pm.009}$ | $0.388_{\pm.014}$ | $0.547_{\pm.067}$ | $0.783_{\pm.059}$ | $0.733_{\pm.016}$ | $0.811_{\pm.032}$ | $0.503_{\pm.068}$ |
| ETTm1 | 8 | **0.527**$_{\pm.073}$ | $0.921_{\pm.026}$ | $1.538_{\pm.254}$ | $2.326_{\pm.445}$ | $2.204_{\pm.420}$ | $1.877_{\pm.245}$ | $2.024_{\pm.143}$ | $2.375_{\pm.405}$ |
| | | **0.557**$_{0.048}$ | $0.760_{\pm.026}$ | $1.015_{\pm.112}$ | $1.260_{\pm.167}$ | $0.984_{\pm.074}$ | $0.908_{\pm.038}$ | $0.961_{\pm.027}$ | $1.258_{\pm.104}$ |
| | 16 | **0.968**$_{\pm.104}$ | $1.100_{\pm.032}$ | $1.744_{\pm.100}$ | $2.339_{\pm.270}$ | $2.350_{\pm.170}$ | $2.032_{\pm.234}$ | $2.486_{\pm.207}$ | $2.321_{\pm.469}$ |
| | | **0.821**$_{\pm.072}$ | $0.822_{\pm.026}$ | $1.104_{\pm.041}$ | $1.249_{\pm.088}$ | $0.974_{\pm.016}$ | $0.919_{\pm.031}$ | $0.984_{\pm.016}$ | $1.259_{\pm.132}$ |
| ETTh1 | 8 | **0.292**$_{\pm.036}$ | $0.483_{\pm.017}$ | $0.703_{\pm.054}$ | $0.870_{\pm.134}$ | $3.451_{\pm.335}$ | $4.259_{\pm1.13}$ | $4.278_{\pm1.12}$ | $1.006_{\pm.281}$ |
| | | **0.424**$_{\pm.033}$ | $0.461_{\pm.011}$ | $0.644_{\pm.038}$ | $0.730_{\pm.060}$ | $1.194_{\pm.034}$ | $1.092_{\pm.028}$ | $1.169_{\pm.055}$ | $0.762_{\pm.115}$ |
| | 16 | **0.374**$_{\pm.061}$ | $0.488_{\pm.010}$ | $0.681_{\pm.018}$ | $0.983_{\pm.139}$ | $1.929_{\pm.105}$ | $1.332_{\pm.125}$ | $1.701_{\pm.088}$ | $0.681_{\pm.104}$ |
| | | $0.488_{\pm.039}$ | **0.463**$_{\pm.018}$ | $0.640_{\pm.008}$ | $0.760_{\pm.062}$ | $1.029_{\pm.030}$ | $0.879_{\pm.037}$ | $0.999_{\pm.023}$ | $0.641_{\pm.055}$ |
| Wind | 8 | **0.681**$_{\pm.075}$ | $1.854_{\pm.032}$ | $1.321_{\pm.379}$ | $1.942_{\pm.101}$ | $12.53_{\pm2.25}$ | $12.67_{\pm1.75}$ | $11.35_{\pm6.61}$ | $2.006_{\pm.145}$ |
| | | **0.596**$_{\pm.052}$ | $1.223_{\pm.014}$ | $0.863_{\pm.143}$ | $1.067_{\pm.086}$ | $1.370_{\pm.107}$ | $1.440_{\pm.059}$ | $1.305_{\pm.369}$ | $1.103_{\pm.100}$ |
| | 16 | $1.033_{\pm.062}$ | $1.955_{\pm.015}$ | **0.894**$_{\pm.038}$ | $1.262_{\pm.178}$ | $13.96_{\pm.1.53}$ | $12.86_{\pm2.60}$ | $13.79_{\pm5.37}$ | $1.138_{\pm.205}$ |
| | | **0.757**$_{\pm.053}$ | $1.247_{\pm.011}$ | $0.785_{\pm.037}$ | $0.843_{\pm.066}$ | $1.347_{\pm.060}$ | $1.240_{\pm.070}$ | $1.261_{\pm.171}$ | $0.862_{\pm.092}$ |

obtained 5%-Traffic, 3%-Electricity, 2%-Weather, 2%-Wind, 1%-ETTm1, and 5%-ETTh1. The statistical descriptions of the real-world datasets can be found in Appendix C.1. All datasets are split chronologically and adopt the same train/validation/test ratios, i.e. 7:1:2.

**Baselines.** We compare D$^3$VAE with one GP (Gaussian Process) based method (GP-copula [76]), two auto-regressive methods (DeepAR [77] and TimeGrad [74]), and four VAE-based methods, i.e., vanilla VAE, NVAE [89], factor-VAE (f-VAE for short) [50] and $\beta$-TCVAE [15].

**Implementation Details.** An input-$l_x$-predict-$l_y$ window is applied to roll the train, validation, and test sets with stride one time step, respectively, and this setting is adopted for all datasets. Hereinafter, the last dimension of the multivariate time series is selected as the target variable by default.

We use the Adam optimizer with an initial learning rate of $5e-4$. The batch size is 16 and the training is set to 20 epochs at most equipped with early stopping. The number of disentanglement factors is chosen from $\{4, 8\}$, and $\beta_T \in \boldsymbol{\beta}$ is set to range from 0.01 to 0.1 with different diffusion steps $T \in [100, 1000]$, then $\omega$ is set to 0.1. The trade-off hyperparameters are set as $\psi = 0.05, \lambda = 0.1, \gamma = 0.001$ for ETTs, and $\psi = 0.5, \lambda = 1.0, \gamma = 0.01$ for others. All the experiments were carried out on a Linux machine with a single NVIDIA P40 GPU. The experiments are repeated 5 times and the average and variance of the predictions are reported. We use the Continuous Ranked Probability Score (CRPS) [68] and Mean Squared Error (MSE) as the evaluation metrics. For both metrics, the lower, the better. In particular, CRPS is used to evaluate the similarity of two distributions and is equivalent to Mean Absolute Error (MAE) when two distributions are discrete.

### 3.2 Main Results

Two different prediction lengths, i.e., $l_y \in \{8, 16\}$ ($l_x = l_y$), are evaluated. The results of longer prediction lengths are available in Appendix D.

**Toy Datasets.** In Table 1, we can observe that D$^3$VAE achieves SOTA performance most of the time, and achieves competitive CRPS in $D_2$ for prediction length 16. Besides, VAEs outperform VARs and GP on $D_1$, but VARs achieve better performance on $D_2$, which demonstrates the advantage of VARs in learning complex temporal dependencies.

**Real-World Datasets.** As for the experiments on real-world data, D$^3$VAE achieves consistent SOTA performance except for the prediction length 16 on the Wind dataset (Table 2). Particularly, under the input-8-predict-8 setting, D$^3$VAE can provide remarkable improvements in Traffic, Electricity, Wind, ETTm1, ETTh1 and Weather w.r.t. MSE reduction (90%, 71%, 48%, 43%, 40% and 28%). Regarding the CRPS reduction, D$^3$VAE achieves a 73% reduction in Traffic, 31% in Wind, and 27% in Electricity under the input-8-predict-8 setting, and a 70% reduction in Traffic, 18% in Electricity, and 7% in Weather under the input-16-predict-16 setting. Overall, D$^3$VAE gains the averaged 43% MSE reduction and 23% CRPS reduction among the above settings. More results under longer prediction-length settings and on full datasets can be found in Appendix D.1.

**Uncertainty Estimation.** The uncertainty can be assessed by estimating the noise of the outcome series when doing the prediction (see Section 2.3). Through scale parameter $\omega$, the generated distribution space can be adjusted accordingly (results on the effect of $\omega$ can be found in Appendix D.3). The showcases in Fig. 3 demonstrate the uncertainty estimation of the yielded series in the Traffic dataset, where the last six dimensions are treated as target variables. We can find that noise estimation can quantify the uncertainty effectively. For example, the estimated uncertainty grows rapidly when extreme values are encountered.

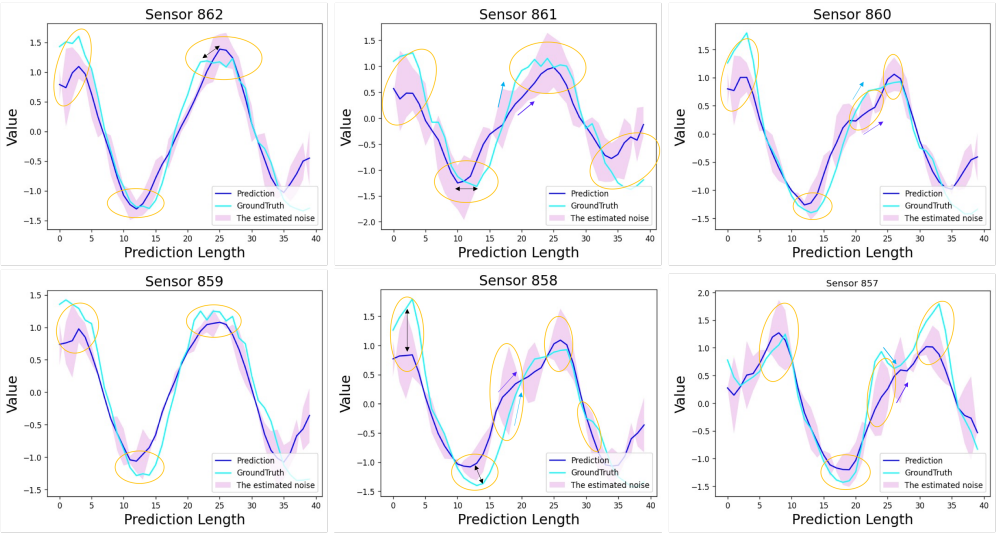



Figure 3: The uncertainty estimation of the prediction of the last six dimensions of the Traffic dataset, and the colored envelope denotes the estimated uncertainty.

Table 3: Ablation study of the coupled diffusion probabilistic model w.r.t. MSE and CSPR.

| Dataset | Traffic | | Electricity | |
|---|---|---|---|---|
| | 16 | 32 | 16 | 32 |
| $\text{D}^3\text{VAE}_{-\widetilde{Y}}$ | $0.122_{\pm.006}$ $0.126_{\pm.013}$<br>$0.250_{\pm.008}$ $0.261_{\pm.017}$ | | $0.350_{\pm.043}$ $0.422_{\pm.012}$<br>$0.480_{\pm.032}$ $0.551_{\pm.012}$ | |
| $\text{D}^3\text{VAE}_{-\widetilde{Y}-\text{DSM}}$ | $0.096_{\pm.006}$ $0.092_{\pm.008}$<br>$0.217_{\pm.010}$ $0.220_{\pm.013}$ | | $0.331_{\pm.023}$ $0.502_{\pm.079}$<br>$0.450_{\pm.021}$ $0.584_{\pm.053}$ | |
| $\text{D}^3\text{VAE}_{-\widetilde{X}}$ | $0.123_{\pm.003}$ $0.117_{\pm.007}$<br>$0.256_{\pm.006}$ $0.253_{\pm.013}$ | | $0.351_{\pm.047}$ $0.420_{\pm.056}$<br>$0.481_{\pm.036}$ $0.540_{\pm.046}$ | |
| $\text{D}^3\text{VAE}_{-\text{CDM}}$ | $0.123_{\pm.004}$ $0.118_{\pm.008}$<br>$0.255_{\pm.007}$ $0.252_{\pm.015}$ | | $0.365_{\pm.025}$ $0.439_{\pm.014}$<br>$0.498_{\pm.018}$ $0.561_{\pm.016}$ | |
| $\text{D}^3\text{VAE}_{-\text{CDM}-\text{DSM}}$ | $0.123_{\pm.003}$ $0.119_{\pm.003}$<br>$0.255_{\pm.003}$ $0.253_{\pm.005}$ | | $0.338_{\pm.041}$ $0.448_{\pm.062}$<br>$0.467_{\pm.029}$ $0.555_{\pm.041}$ | |
| $\text{D}^3\text{VAE}$ | $\mathbf{0.081}_{\pm.009}$ $\mathbf{0.091}_{\pm.007}$<br>$\mathbf{0.200}_{\pm.014}$ $\mathbf{0.216}_{\pm.012}$ | | $\mathbf{0.308}_{\pm.030}$ $\mathbf{0.410}_{\pm.075}$<br>$\mathbf{0.437}_{\pm.020}$ $\mathbf{0.534}_{\pm.058}$ | |

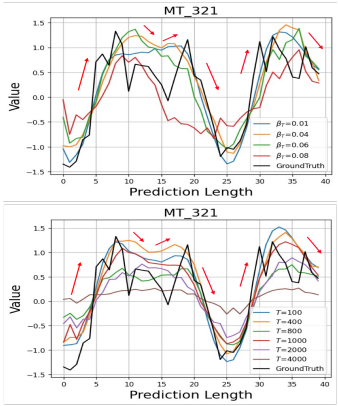



Figure 4: Comparisons of predictions with different $\beta_T$ and varying $T$ on the Electricity dataset.

**Disentanglement Evaluation.** For time series forecasting, it is difficult to label disentangled factors by hand, thus we take different dimensions of $Z$ as the factors to be disentangled: $z_i = [z_{i,1}, \cdots, z_{i,m}]$ ($z_i \in Z$). We build a classifier to discriminate whether an instance $z_{i,j}$ belongs to class $j$ such that the disentanglement quality can be assessed by evaluating the classification performance. Besides, we adopt the Mutual Information Gap (MIG) [15] as a metric to evaluate the disentanglement in a more straightforward way. Due to the space limit, the evaluation of disentanglement with different factors can be found in Appendix E.

### 3.3 Model Analysis

**Ablation Study of the Coupled Diffusion and Denoising Network.** To evaluate the effectiveness of the coupled diffusion model (CDM), we compare the full versioned $\text{D}^3\text{VAE}$ with its three variants: i) $\text{D}^3\text{VAE}_{-\widetilde{Y}}$, i.e. $\text{D}^3\text{VAE}$ without diffused $Y$, ii) $\text{D}^3\text{VAE}_{-\widetilde{X}}$, i.e. $\text{D}^3\text{VAE}$ without diffused $X$, and iii) $\text{D}^3\text{VAE}_{-\text{CDM}}$, i.e. $\text{D}^3\text{VAE}$ without any diffusion. Besides, the performance of $\text{D}^3\text{VAE}$ without denoising score matching (DSM) is also reported when the target series is not diffused, which are denoted as $\text{D}^3\text{VAE}_{-\widetilde{Y}-\text{DSM}}$ and $\text{D}^3\text{VAE}_{-\text{CDM}-\text{DSM}}$. The ablation study is carried out on Traffic and Electricity datasets under the settings of input-16-predict-16 and input-32-predict-32. In Table 3, we can find that the diffusion process is effective in augmenting the input or the target. Moreover, when the target is not diffused, the denoising network would be deficient since the noise level of the target cannot be estimated by then.

**Variance Schedule $\boldsymbol{\beta}$ and The Number of Diffusion Steps $T$.** To reduce the effect of the uncertainty while preserving the informative temporal patterns, the extent of the diffusion should be configured properly. Too small a variance schedule or inadequate diffusion steps will lead to a meaningless diffusion process. Otherwise, the diffusion could be out of control [5]. Here we analyze the effect of the variance schedule $\boldsymbol{\beta}$ and the number of diffusion steps $T$. We set $\beta_1 = 0$ and change the value of $\beta_T$ in the range of $[0.01, 0.1]$, and $T$ ranges from 100 to 4000. As shown in Fig. 4, we can see that the prediction performance can be improved if proper $\boldsymbol{\beta}$ and $T$ are employed.

## 4 Discussion

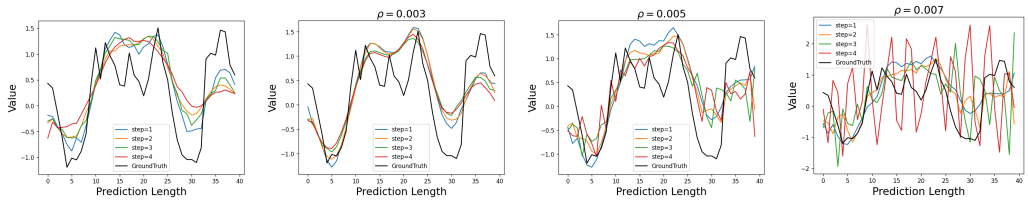
**Sampling for Generative Time Series Forecasting.**
The Langevin dynamics has been widely applied to the sampling of energy-based models (EBMs) [21],

$$Y_k = Y_{k-1} - \frac{\rho}{2}\nabla_Y E_\phi(Y_{k-1}) + \rho^{\frac{1}{2}}\mathcal{N}(0, I_d)\,, \tag{15}$$

[5] An illustrative showcase can be found in Appendix F.

where $k \in \{0, \cdots, K\}$, $K$ denotes the number of sampling steps, and $\rho$ is a constant. With $K$ and $\rho$ being properly configured, samples with high quality can be generated. The Langevin dynamics has been successfully applied to image reconstruction and natural language processing [56, 20].

In this work, we employ a single-step gradient denoising jump to generate the target series. The experiments hat were carried out demonstrate the effectiveness of such single-step sampling. To investigate whether it is worth taking more sampling steps for further performance improvement of time series forecasting, we conduct an extra empirical study. We showcase the prediction results under different sampling strategies in Fig. 5. By omitting the additive noise in Langevin dynamics, we employ the multi-step denoising for $D^3$VAE to generate the target series and plot the generated results in Fig. 5a. Then, with the standard Langevin dynamics, we can implement a generative procedure instead of denoising and compare the generated target series with different $\rho$ (see Figs. 5b to 5d). We can observe that, more sampling steps might not be helpful in improving prediction performance for generative time series forecasting (Fig. 5a). Besides, larger sampling steps would lead to high computational complexity. On the other hand, different configurations of Langevin dynamics (with varying $\rho$) cannot bring indispensable benefits for time series forecasting (Figs. 5b to 5d).




(a) Multi-step denoising. (b) $\rho = 0.003$. (c) $\rho = 0.005$. (d) $\rho = 0.007$.

Figure 5: The prediction showcases in the Electricity dataset with different sampling strategies.

**Limitations.**
With the coupled diffusion probabilistic model, the aleatoric uncertainty of time series can be reduced though, new bias is brought into the series to mimic the distribution of the input and target. However, as a common issue in VAEs that any introduced bias in the input will result in bias in the generated output [92], the diffusion steps and variance schedule need to be chosen cautiously, such that this model can be applied to different time series tasks smoothly. The proposed model is devised for general time series forecasting, it should be used properly to avoid the potential negative societal impacts, such as any kind of illegal applications.

In time series predictive analysis, disentanglement of the latent variables has been of great importance for interpreting the prediction to provide more reliance. Due to the lack of prior knowledge of the entangled factors in generative time series forecasting, only unsupervised disentanglement learning can be done, which has been proven theoretically feasible for time series [64]. Despite this, for boarder applications of disentanglement and better performance, it is still worth exploring the ways of labeling the factors of time series in the future. Moreover, because of the uniqueness of time series data, it is also a promising direction to explore more generative and sampling methods for the time series generation task.

## 5 Conclusion

In this work, we propose a generative model with the bidirectional VAE as the backbone. To further improve the generalizability, we devise a coupled diffusion probabilistic model for time series forecasting. Then a scaled denoising network is developed to guarantee the prediction accuracy. Afterward, the latent variables are further disentangled for better model interpretability. Extensive experiments on synthetic data and real-world data validate that our proposed generative model achieves SOTA performance compared to existing competitive generative models.

## Acknowledgement

We thank Longyuan Power Group Corp. Ltd. for supporting this work.

## References


[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021.

[2] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations*, 2018.

[3] Adebiyi A Ariyo, Adewumi O Adewumi, and Charles K Ayo. Stock price prediction using the ARIMA model. In *UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, pages 106–112. IEEE, 2014.

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

[5] Mauricio Barahona and Chi-Sang Poon. Detection of nonlinear dynamics in short, noisy time series. *Nature*, 381(6579):215–217, 1996.

[6] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[7] Mikolaj Binkowski, Gautier Marti, and Philippe Donnat. Autoregressive convolutional neural networks for asynchronous time series. In *International Conference on Machine Learning*, pages 580–589. PMLR, 2018.

[8] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. Conditional time series forecasting with convolutional neural networks. *STAT*, 1050:16, 2017.

[9] George EP Box and Gwilym M Jenkins. Some recent advances in forecasting and control. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 17(2):91–109, 1968.

[10] Sofiane Brahim-Belhouari and Amine Bermak. Gaussian process for nonstationary time series prediction. *Computational Statistics & Data Analysis*, 47(4):705–712, 2004.

[11] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, et al. Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in Neural Information Processing Systems*, 33:17766–17778, 2020.

[12] Li-Juan Cao and Francis Eng Hock Tay. Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on Neural Networks*, 14(6):1506–1518, 2003.

[13] Shiyu Chang, Yang Zhang, Wei Han, Mo Yu, Xiaoxiao Guo, Wei Tan, Xiaodong Cui, Michael Witbrock, Mark A Hasegawa-Johnson, and Thomas S Huang. Dilated recurrent neural networks. *Advances in Neural Information Processing Systems*, 30, 2017.

[14] Sotirios P Chatzis. Recurrent latent variable conditional heteroscedasticity. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2711–2715. IEEE, 2017.

[15] Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in Neural Information Processing Systems*, 31, 2018.

[16] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *Advances in Neural Information Processing Systems*, 28, 2015.

[17] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

[18] Emmanuel de Bézenac, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, and Tim Januschowski. Normalizing kalman filters for multivariate time series analysis. *Advances in Neural Information Processing Systems*, 33:2995–3007, 2020.

[19] Ankur Debnath, Govind Waghmare, Hardik Wadhwa, Siddhartha Asthana, and Ankur Arora. Exploring generative data augmentation in multivariate time series forecasting: Opportunities and challenges. *Solar-Energy*, 137:52–560, 2021.

[20] Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. Residual energy-based models for text generation. In *International Conference on Learning Representations*, 2019.

[21] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *Advances in Neural Information Processing Systems*, 32, 2019.

[22] Stephen Ellner and Peter Turchin. Chaos in a noisy world: New methods and evidence from time-series analysis. *The American Naturalist*, 145(3):343–375, 1995.

[23] Amirreza Farnoosh, Bahar Azari, and Sarah Ostadabbas. Deep switching auto-regressive factorization: Application to time series forecasting. *arXiv preprint arXiv:2009.05135*, 2020.

[24] Konstantinos Fokianos, Anders Rahbek, and Dag Tjøstheim. Poisson autoregression. *Journal of the American Statistical Association*, 104(488):1430–1439, 2009.

[25] Germain Forestier, François Petitjean, Hoang Anh Dau, Geoffrey I Webb, and Eamonn Keogh. Generating synthetic time series to augment sparse datasets. In *IEEE International Conference on Data Mining*, pages 865–870. IEEE, 2017.

[26] Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. GP-VAE: Deep probabilistic time series imputation. In *International Conference on Artificial Intelligence and Statistics*, pages 1651–1661. PMLR, 2020.

[27] Vincent Fortuin, Matthias Hüser, Francesco Locatello, Heiko Strathmann, and Gunnar Rätsch. SOM-VAE: Interpretable discrete representation learning on time series. In *International Conference on Learning Representations*, 2019.

[28] WR Foster, F Collopy, and LH Ungar. Neural network forecasting of short, noisy time series. *Computers & Chemical Engineering*, 16(4):293–297, 1992.

[29] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[30] John Cristian Borges Gamboa. Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*, 2017.

[31] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv:2107.03342*, 2021.

[32] Felix A Gers, Nicol N Schraudolph, and Jürgen Schmidhuber. Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, 3(Aug):115–143, 2002.

[33] C Lee Giles, Steve Lawrence, and Ah Chung Tsoi. Noisy time series prediction using recurrent neural networks and grammatical inference. *Machine Learning*, 44(1):161–183, 2001.

[34] Laurent Girin, Simon Leglaive, Xiaoyu Bie, Julien Diard, Thomas Hueber, and Xavier Alameda-Pineda. Dynamical variational autoencoders: A comprehensive review. *arXiv preprint arXiv:2008.12595*, 2020.

[35] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, 2016.

[36] Vinayaka Gude, Steven Corns, and Suzanna Long. Flood prediction and uncertainty estimation using deep learning. *Water*, 12(3):884, 2020.

[37] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *AAAI Conference on Artificial Intelligence*, volume 33, pages 922–929, 2019.

[38] Michaela Hardt, Alvin Rajkomar, Gerardo Flores, Andrew Dai, Michael Howell, Greg Corrado, Claire Cui, and Moritz Hardt. Explaining an increase in predicted risk for clinical alerts. In *ACM Conference on Health, Inference, and Learning*, pages 80–89, 2020.

[39] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. 2016.

[40] Steven Craig Hillmer and George C Tiao. An ARIMA-model-based approach to seasonal adjustment. *Journal of the American Statistical Association*, 77(377):63–70, 1982.

[41] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[42] Siu Lau Ho and Min Xie. The use of ARIMA models for reliability forecasting and analysis. *Computers & Industrial Engineering*, 35(1-2):213–216, 1998.

[43] Aapo Hyvärinen, Jarmo Hurri, and Patrik O Hoyer. Estimation of non-normalized statistical models. In *Natural Image Statistics*, pages 419–426. Springer, 2009.

[44] Aya Abdelsalam Ismail, Mohamed Gunady, Hector Corrada Bravo, and Soheil Feizi. Benchmarking deep learning interpretability in time series predictions. *Advances in Neural Information Processing Systems*, 33:6441–6452, 2020.

[45] Aya Abdelsalam Ismail, Mohamed Gunady, Luiz Pessoa, Hector Corrada Bravo, and Soheil Feizi. Input-cell attention reduces vanishing saliency of recurrent neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[46] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019.

[47] Dulakshi SK Karunasinghe and Shie-Yui Liong. Chaotic time series prediction with a global model: Artificial neural network. *Journal of Hydrology*, 323(1-4):92–105, 2006.

[48] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in Neural Information Processing Systems*, 30, 2017.

[49] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *International Conference on Machine Learning*, pages 2668–2677. PMLR, 2018.

[50] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International Conference on Machine Learning*, pages 2649–2658. PMLR, 2018.

[51] Kyoung-jae Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

[52] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. The (un) reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 267–280. Springer, 2019.

[53] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[54] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in Neural Information Processing Systems*, 29, 2016.

[55] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

[56] Rithesh Kumar, Sherjil Ozair, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models. *arXiv preprint arXiv:1901.08508*, 2019.

[57] Naoto Kunitomo and Seisho Sato. A robust-filtering method for noisy non-stationary multivariate time series with econometric applications. *Japanese Journal of Statistics and Data Science*, 4(1):373–410, 2021.

[58] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 95–104, 2018.

[59] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.

[60] Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks: A unified approach to action segmentation. In *European Conference on Computer Vision*, pages 47–54. Springer, 2016.

[61] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems*, 32, 2019.

[62] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.

[63] Yingzhen Li and Richard E Turner. Rényi divergence variational inference. *Advances in Neural Information Processing Systems*, 29, 2016.

[64] Yuening Li, Zhengzhang Chen, Daochen Zha, Mengnan Du, Denghui Zhang, Haifeng Chen, and Xia Hu. Learning disentangled representations for time series. *arXiv preprint arXiv:2105.08179*, 2021.

[65] Zengyi Li, Yubei Chen, and Friedrich T Sommer. Learning energy-based models in high-dimensional spaces with multi-scale denoising score matching. *arXiv preprint arXiv:1910.07762*, 2019.

[66] Chi-Jie Lu, Tian-Shyug Lee, and Chih-Chou Chiu. Financial time series forecasting using independent component analysis and support vector regression. *Decision Support Systems*, 47(2):115–125, 2009.

[67] Danielle C Maddix, Yuyang Wang, and Alex Smola. Deep factors with Gaussian processes for forecasting. *arXiv preprint arXiv:1812.00098*, 2018.

[68] James E Matheson and Robert L Winkler. Scoring rules for continuous probability distributions. *Management Science*, 22(10):1087–1096, 1976.

[69] Nam Nguyen and Brian Quanz. Temporal latent auto-encoder: A method for probabilistic multivariate time series forecasting. In *AAAI Conference on Artificial Intelligence*, volume 35, pages 9117–9125, 2021.

[70] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[71] Fotios Petropoulos, Rob J Hyndman, and Christoph Bergmeir. Exploring the sources of uncertainty: Why does bagging for time series forecasting work? *European Journal of Operational Research*, 268(2):545–554, 2018.

[72] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *International Joint Conference on Artificial Intelligence*, 2017.

[73] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. *Advances in Neural Information Processing Systems*, 31, 2018.

[74] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, pages 8857–8868. PMLR, 2021.

[75] Geoffrey Roeder, Yuhuai Wu, and David K Duvenaud. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. *Advances in Neural Information Processing Systems*, 30, 2017.

[76] David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank Gaussian copula processes. *Advances in Neural Information Processing Systems*, 32, 2019.

[77] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

[78] Saeed Saremi, Aapo Hyvärinen, et al. Neural empirical Bayes. *Journal of Machine Learning Research*, 2019.

[79] Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. *Advances in Neural Information Processing Systems*, 32, 2019.

[80] Alex Sherstinsky. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020.

[81] Qiquan Shi, Jiaming Yin, Jiajun Cai, Andrzej Cichocki, Tatsuya Yokota, Lei Chen, Mingxuan Yuan, and Jia Zeng. Block hankel tensor ARIMA for multiple short time series forecasting. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 5758–5766, 2020.

[82] Shun-Yao Shih, Fan-Keng Sun, and Hung-yi Lee. Temporal pattern attention for multivariate time series forecasting. *Machine Learning*, 108(8):1421–1441, 2019.

[83] Sameer Singh. Noisy time-series prediction using pattern recognition techniques. *Computational Intelligence*, 16(1):114–133, 2000.

[84] Slawek Smyl and Karthik Kuber. Data preprocessing and augmentation for multiple short time series forecasting with recurrent neural networks. In *International Symposium on Forecasting*, 2016.

[85] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[86] Huan Song, Deepta Rajan, Jayaraman J Thiagarajan, and Andreas Spanias. Attend and diagnose: Clinical time series analysis using attention models. In *AAAI Conference on Artificial Intelligence*, 2018.

[87] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[88] Sana Tonekaboni, Shalmali Joshi, Kieran Campbell, David K Duvenaud, and Anna Goldenberg. What went wrong and when? instance-wise feature importance for time-series black-box models. *Advances in Neural Information Processing Systems*, 33:799–809, 2020.

[89] Arash Vahdat and Jan Kautz. NVAE: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 33:19667–19679, 2020.

[90] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

[91] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(12), 2010.

[92] Julius Von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. Self-supervised learning with data augmentations provably isolates content from style. *Advances in Neural Information Processing Systems*, 34, 2021.

[93] Juntao Wang, Wun Kwan Yam, Kin Long Fong, Siew Ann Cheong, and KY Wong. Gaussian process kernels for noisy time series: Application to housing price prediction. In *International Conference on Neural Information Processing*, pages 78–89. Springer, 2018.

[94] Satosi Watanabe. Information theoretical analysis of multivariate correlation. *Ibm Journal of Research and Development*, 4(1):66–82, 1960.

[95] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053*, 2017.

[96] Mike West. Bayesian forecasting of multivariate time series: scalability, structure uncertainty and decisions. *Annals of the Institute of Statistical Mathematics*, 72(1):1–31, 2020.

[97] Bingzhe Wu, Jintang Li, Chengbin Hou, Guoji Fu, Yatao Bian, Liang Chen, and Junzhou Huang. Recent advances in reliable deep graph learning: Adversarial attack, inherent noise, and distribution shift. *arXiv preprint arXiv:2202.07114*, 2022.

[98] Mike Wu, Michael Hughes, Sonali Parbhoo, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez. Beyond sparsity: Tree regularization of deep models for interpretability. In *AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[99] Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv preprint arXiv:2001.08317*, 2020.

[100] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *International Joint Conference on Artificial Intelligence*, pages 1907–1913, 2019.

[101] Jiehui Xu, Jianmin Wang, Mingsheng Long, et al. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34, 2021.

[102] Tijin Yan, Hongwei Zhang, Tong Zhou, Yufeng Zhan, and Yuanqing Xia. ScoreGrad: Multivariate probabilistic time series forecasting with continuous energy-based generative models. *arXiv preprint arXiv:2106.10121*, 2021.

[103] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[104] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In *International Joint Conference on Artificial Intelligence*, pages 3634–3640, 2018.

[105] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Computation*, 31(7):1235–1270, 2019.

[106] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI Conference on Artificial Intelligence*, 2021.

[107] Yong Zou, Reik V Donner, Norbert Marwan, Jonathan F Donges, and Jürgen Kurths. Complex network approaches to nonlinear time series analysis. *Physics Reports*, 787:1–97, 2019.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 4 for details.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We discussÅ the potential negative societal impacts in Section 4
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] Our problem formulation and theoretical proof are described in Section 2, and all the theoretical results are with reasonable and necessary assumptions.
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We use six real-world datasets, for each of them is provided by URL or citation in Section 3.1.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 3 and Appendix C
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] All the experimental results are repeated five times, the mean and standard deviation of the results are provided in Section 3
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 3.1
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] The datasets and baseline models that used in this paper are all cited in Section 3.1.
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] The newly introduced dataset Wind is provided in the supplementary materials and will be released later.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] The datasets used in this paper are all public datasets, and we didn't use/curate data from other people.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] The data we use doesn't contain any identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

# A Related Work

## A.1 Time Series Forecasting

We first briefly review the related literature of time series forecasting (TSF) methods as below. Complex temporal patterns can be manifested over short- and long-term as the time series evolves across time. To leverage the time evolution nature, existing statistical models, such as ARIMA [9] and Gaussian process regression [10] have been well established and applied to many downstream tasks [40, 42, 3]. Recurrent neural network (RNN) models are also introduced to model temporal dependencies for TSF in a sequence-to-sequence paradigm [35, 13, 95, 58, 67, 73, 77]. Besides, temporal attention [72, 86, 82] and causal convolution [4, 8, 79] are further explored to model the intrinsic temporal dependencies. Recent Transformer-based models have strengthened the capability of exploring hidden intricate temporal patterns for long-term TSF [101, 61, 99, 106]. On the other hand, the multivariate nature of TSF is another topic many works have been focusing on. These works treat a collection of time series as a unified entity and mine the inter-series correlations with different techniques, such as probabilistic models [77, 73], matrix/tensor factorization [79**?** ], convolution neural networks (CNNs) [4, 58], and graph neural networks (GNNs) [37, 62, 104, 100, 11].

To improve the reliability and performance of TSF, instead of modeling the raw data, there exist works inferring the underlying distribution of the time series data with generative models [103, 18]. Many studies have employed a variational auto-encoder (VAE) to model the probabilistic distribution of sequential data [29, 34, 16, 14, 69]. For example, VRNN [16] employs the VAE to each hidden state of RNN such that the variability of highly structured sequential data can be captured. To yield predictive distribution for multivariate TSF, TLAE [69] implements nonlinear transformation by replacing matrix factorization with encoder-decoder architecture and temporal deep temporal latent model. Another line of generative methods for TSF focus on energy-based models (EBMs), such as TimeGrad [74] and ScoreGrad [102]. EBMs do not place a restriction on the tractability of the normalizing constants [102]. Though flexible, the unknown normalizing constant makes the training of EBMs particularly difficult.

This paper focuses on TSF with VAE-based models. Besides, as many real-world time series data are relatively short and small [84], a coupled probabilistic diffusion model is proposed to augment the input series, as well as the output series, simultaneously, such that the distribution space can be enlarged without increasing the aleatoric uncertainty [48]. Moreover, to guarantee the generated target series moving toward the true target, a multi-scaled score-matching denoising network is plugged in for accurate future series prediction. To the best of our knowledge, this is the first work focusing on generative TSF with the diffusion model and denoising techniques.

## A.2 Time Series Augmentation

Both traditional methods and deep learning methods can deteriorate when limited time series data are encountered. Generating synthetic time series is commonly adopted for augmenting short time series [17, 25, 103]. Transforming the original time series by cropping, flipping and warping [46, 19] is another kind of approach dedicated to TSF when the training data is limited. Whereas the synthetic time series may not respect the original feature relationship across time, and the transformation methods do not change the distribution space. Thus, the overfitting issues cannot be avoided. Incorporating the probabilistic diffusion model for TSF differentiates our work from existing time series augmentation methods.

## A.3 Uncertainty Estimation and Denoising for Time Series Forecasting

There exist works aiming to estimate the uncertainty [48] for time series forecasting [71, 96, 36] by epistemic uncertainty. Nevertheless, the inevitable aleatoric uncertainty of time series is often ignored, which may stem from error-prone data measurement, collection, and so forth [97]. Another line of studies focuses on the detection of noise in time series data [66] or devising suitable models for noise alleviation [33]. However, none of the existing works attempts to quantify the aleatoric uncertainty, which further differentiates our work from priors.

It is necessary to relieve the effect of noise in real-world time series data [22]. [5, 57] propose to preprocess the time series with smoothing and filtering techniques. However, such preprocessing methods can only be applied to the noise raised by the irregular data of time series. Neural networks

are also introduced to denoise the time series [28, 83, 33, 47], while these deep networks can only deal with specific types of time series as well.

### A.4 Interpretability of Time Series Forecasting

A number of works put effort to explain the deep neural networks [98, 49, 2] to make the prediction more interpretable, but these methods often lack reliability when the explanation is sensitive to factors that do not contribute to the prediction [52]. Several works have been proposed to increase the reliability of TSF tasks [44, 45]. For multivariate time series, the interpretability of the representations can be improved by mapping the time series into latent space [27]. Besides, recent works have been proposed to disentangle the latent variables to identify the independent factors of the data, which can further lead to improved interpretability of the representation and higher performance [39, 59, 50]. The disentangled VAE has been applied to time series to benefit the generated results [64]. However, the choice of the latent variables is crucial for the disentanglement of time series data. We devise a bidirectional VAE (BVAE) and take the dimensions of each latent variable as the factors to be disentangled.

## B Proofs of Lemma 1 and Lemma 2

With the coupled diffusion process and Eqs. (5) and (6), as well as Proposition 1, introduced in the main text, the diffused target series and generated target series can be decomposed as $\widetilde{Y}^{(t)} = \langle \widetilde{Y}^{(t)}, \delta_{\widetilde{Y}}^{(t)} \rangle$ and $\widehat{Y}^{(t)} = \langle \widehat{Y}^{(t)}, \delta_{\widehat{Y}}^{(t)} \rangle$. Then, we can draw the following two conclusions:

**Lemma 1.** *$\forall \varepsilon > 0$, there exists a probabilistic model $f_{\phi,\theta} := (p_\phi, p_\theta)$ to guarantee that $\mathcal{D}_{\mathrm{KL}}(q(\widetilde{Y}_r^{(t)}) || p_\theta(\widehat{Y}_r^{(t)})) < \varepsilon$, where $\widehat{Y}_r^{(t)} = f_{\phi,\theta}(X^{(t)})$.*

*Proof.* According to Proposition 1, $\widehat{Y}_r$ can be fully captured by the model. That is, $\|Y_r - \widehat{Y}_r\| \longrightarrow 0$ where $Y_r$ is the ideal part of ground truth target series $Y$. And, with Eq. (6) (in the main text), $\widetilde{Y}_r^{(t)} = \sqrt{\bar{\alpha}_t'} Y_r$. Therefore, $\|\widetilde{Y}_r^{(t)} - \widehat{Y}_r^{(t)}\| \longrightarrow 0$ when $t \to \infty$. □

**Lemma 2.** *With the coupled diffusion process, the difference between diffusion noise and generation noise will be reduced, i.e., $\lim_{t\to\infty} \mathcal{D}_{\mathrm{KL}}(q(\delta_{\widetilde{Y}}^{(t)}) || p_\theta(\delta_{\widehat{Y}}^{(t)} | Z^{(t)})) < \mathcal{D}_{\mathrm{KL}}(q(\epsilon_Y) || p_\theta(\epsilon_{\widehat{Y}}))$ .*

*Proof.* According to Proposition 1, the noise of $Y$ consists of the estimation noise $\epsilon_{\widehat{Y}}$ and residual noise $\delta_Y$, i.e., $\epsilon_Y = \langle \epsilon_{\widehat{Y}}, \delta_Y \rangle$ where $\epsilon_{\widehat{Y}}$ and $\delta_Y$ are independent of each other, then $q(\epsilon_Y) = q(\epsilon_{\widehat{Y}}) q(\delta_Y)$. Let $\Delta = \mathcal{D}_{\mathrm{KL}}(q(\epsilon_Y) || p_\theta(\epsilon_{\widehat{Y}})) - \mathcal{D}_{\mathrm{KL}}(q(\epsilon_{\widehat{Y}}) || p_\theta(\epsilon_{\widehat{Y}}))$, we have

$$\begin{aligned}
\Delta &= \mathcal{D}_{\mathrm{KL}}(q(\epsilon_{\widehat{Y}}) q(\delta_Y) || p_\theta(\epsilon_{\widehat{Y}})) - \mathcal{D}_{\mathrm{KL}}(q(\epsilon_{\widehat{Y}}) || p_\theta(\epsilon_{\widehat{Y}})) \\
&= \mathcal{D}_{\mathrm{KL}}(q(\epsilon_{\widehat{Y}}) || p_\theta(\epsilon_{\widehat{Y}})) + \mathcal{D}_{\mathrm{KL}}(q(\delta_Y) || p_\theta(\epsilon_{\widehat{Y}})) - \mathcal{D}_{\mathrm{KL}}(q(\epsilon_{\widehat{Y}}) || p_\theta(\epsilon_{\widehat{Y}})) \\
&= \mathcal{D}_{\mathrm{KL}}(q(\delta_Y) || p_\theta(\epsilon_{\widehat{Y}})) > 0 \,,
\end{aligned}$$

which leads to $\mathcal{D}_{\mathrm{KL}}(q(\epsilon_Y) || p_\theta(\epsilon_{\widehat{Y}})) > \mathcal{D}_{\mathrm{KL}}(q(\epsilon_{\widehat{Y}}) || p_\theta(\epsilon_{\widehat{Y}}) > 0$. Moreover, both $\delta_{\widetilde{Y}}^{(t)}$ and $\delta_{\widehat{Y}}^{(t)}$ are Gaussian noises, when $t \to \infty$, $\exists\ \varepsilon' > 0$, we have $\mathcal{D}_{\mathrm{KL}}(q(\delta_{\widetilde{Y}}^{(t)}) || p_\theta(\delta_{\widehat{Y}}^{(t)} | Z^{(t)})) \leq \varepsilon' < \mathcal{D}_{\mathrm{KL}}(q(\epsilon_Y) || p_\theta(\epsilon_{\widehat{Y}}))$. □

## C Extra Implementation Details

### C.1 Experimental Settings

**Datasets Description.** The main descriptive statistics of the real-world datasets adopted in the experiments of this work are demonstrated in Table 5.

**Input Representation.** We adopt the embedding method introduced in [106] and feed it to an RNN to extract the temporal dependency. Then we concatenate them as follows:

$$X_{\text{input}} = \mathrm{CONCAT}(\mathrm{RNN}(\mathcal{E}(X)), \mathcal{E}(X)) \,,$$

Table 5: Statistical descriptions of the real-world datasets.

| Datasets | # Dims. | Full Data | | Sliced Data | | Target Variable | Time Interval |
|---|---|---|---|---|---|---|---|
| | | Time Span | # Points | Pct. of Full Data | # Points | | |
| Traffic | 862 | 2015-2016 | 17544 | 5% | 877 | Sensor 862 | 1 hour |
| Electricity | 321 | 2011-2014 | 18381 | 3% | 551 | MT_321 | 10 mins |
| Weather | 21 | 2020-2021 | 36761 | 2% | 735 | CO2 (ppm) | 10 mins |
| ETTm1 | 7 | 2016-2018 | 69680 | 1% | 697 | OT | 15 mins |
| ETTh1 | 7 | 2016-2018 | 17420 | 5% | 871 | OT | 1 hour |
| Wind | 7 | 2020-2021 | 45550 | 2% | 911 | wind_power | 15 mins |

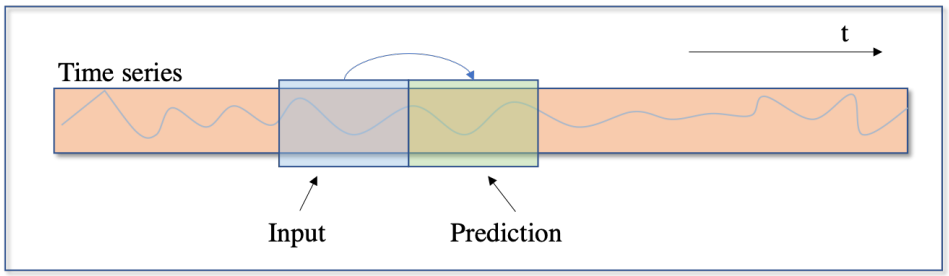



Figure 6: Forecasting process of DeepAR, TimeGrad, and GP-copula. The sliding step is set to 1.

where $X$ is the raw time series data and $\mathcal{E}(\cdot)$ denotes the embedding operation. Here, we use a two-layer gated recurrent unit (GRU), and the dimensionality of the hidden state and embeddings are 128 and 64, respectively.

**Diffusion Process Configuration.** Besides, the diffusion process is configured to be $\beta_t \in [0, 0.1]$ and $T = 100$ for the **Weather** dataset, $\beta_t \in [0, 0.1]$ and $T = 1000$ for the **ETTh1** dataset, $\beta_t \in [0, 0.08]$ and $T = 1000$ for the **Wind** dataset, and $\beta_t \in [0, 0.01]$ and $T = 1000$ for the other datasets.

### C.2 Implementation Details of Baselines

We select previous state-of-the-art generative models as our baselines in the experiments on both synthetic datasets and real-world datasets. Specifically, **1) GP-copula** [76] is a method based on the Gaussian process which is devoted to high dimensional multivariate time series, **2) DeepAR** [77] combines traditional auto-regressive models with RNNs by modeling a probabilistic distribution in an auto-encoder fashion, **3) TimeGrad** [74] is an auto-regressive model for multivariate probabilistic time series forecasting with the help of an energy-based model, **4) Vanilla VAE** (VAE for short) [53] is a classical statistical variational inference method on top of auto-encoder, **5) NVAE** [89] is a deep hierarchical VAE built for image generation using depth-wise separable convolutions and batch normalization, **6) factor-VAE** (f-VAE for short) [50] disentangles the latent variables by encouraging the distribution of representations to be factorial and independent across dimensions, and **7) $\beta$-TCVAE** [15] learns the disentangled representations with total correlation variational auto-encoder algorithm.

To train DeepAR, TimeGrad, and GP-copula in accordance with their original settings, the batch is constructed without shuffling the samples. That is, the instances (sampled with the input-$l_x$-predict-$l_y$ rolling window and $l_x = l_y$, as illustrated in Fig. 6) are fed to the training procedure of these three baselines in chronological order. Besides, these three baselines employ the cumulative distribution function (CDF) for training, so the CDF needs to be reverted to the real distribution for testing.

For f-VAE, $\beta$-TCVAE, and VAE, since the dimensionality of different time series varies, we design a preprocess block to map the original time series into a tensor with the fix-sized dimensionality which can further suit the VAEs well. The preprocess block consists of three nonlinear layers with the sizes of the hidden states to be: $\{128, 64, 32\}$. For NVAE, we keep the original settings suggested in [89] and use Gaussian distribution as the prior. All the baselines are trained using early stopping and the patience is set to 5.

Table 6: Performance comparisons of short-term and long-term TSF in real-world datasets in terms of MSE and CRPS. For MSE and CRPS, the lower, the better. The best results are in boldface.

| Model | | D$^3$VAE | NVAE | $\beta$-TCVAE | f-VAE | DeepAR | TimeGrad | GP-copula | VAE |
|---|---|---|---|---|---|---|---|---|---|
| Traffic | 8 | **0.081**$_{\pm.003}$ | 1.300$_{\pm.024}$ | 1.003$_{\pm.006}$ | 0.982$_{\pm.059}$ | 3.895$_{\pm.306}$ | 3.695$_{\pm.246}$ | 4.299$_{\pm.372}$ | 0.794$_{\pm.130}$ |
| | | **0.207**$_{\pm.003}$ | 0.593$_{\pm.004}$ | 0.894$_{\pm.003}$ | 0.666$_{\pm.032}$ | 1.391$_{\pm.071}$ | 1.410$_{\pm.027}$ | 1.408$_{\pm.046}$ | 0.759$_{\pm.07}$ |
| | 16 | **0.081**$_{\pm.009}$ | 1.271$_{\pm.019}$ | 0.997$_{\pm.004}$ | 0.998$_{\pm.042}$ | 4.140$_{\pm.320}$ | 3.495$_{\pm.362}$ | 4.575$_{\pm.141}$ | 0.632$_{\pm.057}$ |
| | | **0.200**$_{\pm.014}$ | 0.589$_{\pm.001}$ | 0.893$_{\pm.002}$ | 0.692$_{\pm.026}$ | 1.338$_{\pm.043}$ | 1.329$_{\pm.057}$ | 1.506$_{\pm.025}$ | 0.671$_{\pm.038}$ |
| | 32 | **0.091**$_{\pm.007}$ | 0.126$_{\pm.013}$ | 1.254$_{\pm0.019}$ | 0.977$_{\pm.002}$ | 4.234$_{\pm.139}$ | 5.195$_{\pm2.26}$ | 3.717$_{\pm.361}$ | 0.735$_{\pm.084}$ |
| | | **0.216**$_{\pm.012}$ | 0.422$_{\pm.012}$ | 0.937$_{0.007}$ | 0.882$_{\pm.001}$ | 1.367$_{\pm.015}$ | 1.565$_{\pm.329}$ | 1.342$_{\pm.048}$ | 0.735$_{\pm.048}$ |
| | 64 | **0.125**$_{\pm.005}$ | 1.263$_{\pm0.014}$ | 0.903$_{\pm.111}$ | 0.936$_{\pm.190}$ | 3.381$_{\pm.130}$ | 3.692$_{\pm1.54}$ | 3.492$_{\pm.092}$ | 0.692$_{\pm.059}$ |
| | | **0.244**$_{\pm.006}$ | 0.940$_{\pm0.005}$ | 0.839$_{\pm.062}$ | 0.829$_{\pm.078}$ | 1.233$_{\pm.027}$ | 1.412$_{\pm0.257}$ | 1.367$_{\pm.012}$ | 0.710$_{\pm.035}$ |
| Electricity | 8 | **0.251**$_{\pm.015}$ | 1.134$_{\pm.029}$ | 0.901$_{\pm.052}$ | 0.893$_{\pm.069}$ | 2.934$_{\pm.173}$ | 2.703$_{\pm.087}$ | 2.924$_{\pm.218}$ | 0.853$_{\pm.040}$ |
| | | **0.398**$_{\pm.011}$ | 0.542$_{\pm.003}$ | 0.831$_{\pm.004}$ | 0.809$_{\pm.024}$ | 1.244$_{\pm.037}$ | 1.208$_{\pm.024}$ | 1.249$_{\pm.048}$ | 0.795$_{\pm.016}$ |
| | 16 | **0.308**$_{\pm.030}$ | 1.150$_{\pm.032}$ | 0.850$_{\pm.003}$ | 0.807$_{\pm.034}$ | 2.803$_{\pm.199}$ | 2.770$_{\pm.237}$ | 3.065$_{\pm.186}$ | 0.846$_{\pm.062}$ |
| | | **0.437**$_{\pm.020}$ | 0.531$_{\pm.003}$ | 0.814$_{\pm.002}$ | 0.782$_{\pm.024}$ | 1.220$_{\pm.048}$ | 1.240$_{\pm.048}$ | 1.307$_{\pm.042}$ | 0.793$_{\pm.029}$ |
| | 32 | **0.410**$_{\pm.075}$ | 1.302$_{\pm0.011}$ | 0.844$_{\pm.025}$ | 0.861$_{\pm.105}$ | 2.402$_{\pm.156}$ | 2.640$_{\pm.138}$ | 2.880$_{\pm.221}$ | 0.841$_{\pm.071}$ |
| | | **0.534**$_{\pm.058}$ | 0.944$_{\pm0.005}$ | 0.808$_{\pm.005}$ | 0.797$_{\pm.037}$ | 1.130$_{\pm.055}$ | 1.234$_{\pm.027}$ | 1.281$_{\pm.054}$ | 0.790$_{\pm.026}$ |
| Weather | 8 | **0.169**$_{\pm.022}$ | 0.801$_{\pm.024}$ | 0.234$_{\pm.042}$ | 0.591$_{\pm.198}$ | 2.317$_{\pm.357}$ | 2.715$_{\pm.189}$ | 2.412$_{\pm.761}$ | 0.560$_{\pm.192}$ |
| | | **0.357**$_{\pm.024}$ | 0.757$_{\pm.013}$ | 0.404$_{\pm.040}$ | 0.565$_{\pm.080}$ | 0.858$_{\pm.078}$ | 0.920$_{\pm.013}$ | 0.897$_{\pm.115}$ | 0.572$_{\pm.077}$ |
| | 16 | **0.187**$_{\pm.047}$ | 0.811$_{\pm.016}$ | 0.212$_{\pm.012}$ | 0.530$_{\pm.167}$ | 1.269$_{\pm.187}$ | 1.110$_{\pm.083}$ | 1.357$_{\pm.145}$ | 0.424$_{\pm.141}$ |
| | | **0.361**$_{\pm.046}$ | 0.759$_{\pm.009}$ | 0.388$_{\pm.014}$ | 0.547$_{\pm.067}$ | 0.783$_{\pm.059}$ | 0.733$_{\pm.016}$ | 0.811$_{\pm.032}$ | 0.503$_{\pm.068}$ |
| | 32 | **0.203**$_{\pm.008}$ | 0.836$_{\pm0.014}$ | 0.439$_{\pm.394}$ | 0.337$_{\pm.086}$ | 2.518$_{\pm.546}$ | 1.178$_{\pm.069}$ | 1.065$_{\pm.145}$ | 0.329$_{\pm.083}$ |
| | | **0.383**$_{\pm.007}$ | 0.777$_{\pm0.007}$ | 0.508$_{\pm.176}$ | 0.461$_{\pm.031}$ | 0.847$_{\pm.036}$ | 0.724$_{\pm.021}$ | 0.747$_{\pm.035}$ | 0.459$_{\pm.045}$ |
| | 64 | **0.191**$_{\pm.022}$ | 0.932$_{0.020}$ | 0.276$_{\pm.026}$ | 0.676$_{\pm.484}$ | 3.595$_{\pm.956}$ | 1.063$_{\pm.061}$ | 0.992$_{\pm.114}$ | 0.721$_{\pm.496}$ |
| | | **0.358**$_{\pm.044}$ | 0.836$_{0.009}$ | 0.463$_{\pm.026}$ | 0.612$_{\pm.176}$ | 0.994$_{\pm.100}$ | 0.696$_{\pm.011}$ | 0.699$_{\pm.016}$ | 0.635$_{\pm.204}$ |
| ETTm1 | 8 | **0.527**$_{\pm.073}$ | 0.921$_{\pm.026}$ | 1.538$_{\pm.254}$ | 2.326$_{\pm.445}$ | 2.204$_{\pm.420}$ | 1.877$_{\pm.245}$ | 2.024$_{\pm.143}$ | 2.375$_{\pm.405}$ |
| | | **0.557**$_{0.048}$ | 0.760$_{\pm.026}$ | 1.015$_{\pm.112}$ | 1.260$_{\pm.167}$ | 0.984$_{\pm.074}$ | 0.908$_{\pm.038}$ | 0.961$_{\pm.027}$ | 1.258$_{\pm.104}$ |
| | 16 | **0.968**$_{\pm.104}$ | 1.100$_{\pm.032}$ | 1.744$_{\pm.100}$ | 2.339$_{\pm.270}$ | 2.350$_{\pm.170}$ | 2.032$_{\pm.234}$ | 2.486$_{\pm.207}$ | 2.321$_{\pm.469}$ |
| | | **0.821**$_{\pm.072}$ | 0.822$_{\pm.026}$ | 1.104$_{\pm.041}$ | 1.249$_{\pm.088}$ | 0.974$_{\pm.016}$ | 0.919$_{\pm.031}$ | 0.984$_{\pm.016}$ | 1.259$_{\pm.132}$ |
| | 32 | **0.707**$_{\pm.061}$ | 1.298$_{\pm.028}$ | 1.438$_{\pm.429}$ | 2.563$_{\pm.358}$ | 4.855$_{\pm.179}$ | 1.251$_{\pm.133}$ | 1.402$_{\pm.187}$ | 2.660$_{\pm.349}$ |
| | | **0.697**$_{\pm.040}$ | 0.893$_{\pm.010}$ | 0.953$_{\pm.173}$ | 1.330$_{\pm.104}$ | 1.787$_{\pm.029}$ | 0.822$_{\pm.032}$ | 0.844$_{\pm.043}$ | 1.367$_{\pm.083}$ |
| ETTh1 | 8 | **0.292**$_{\pm.036}$ | 0.483$_{\pm.017}$ | 0.703$_{\pm.054}$ | 0.870$_{\pm.134}$ | 3.451$_{\pm.335}$ | 4.259$_{\pm1.13}$ | 4.278$_{\pm1.12}$ | 1.006$_{\pm.281}$ |
| | | **0.424**$_{\pm.033}$ | 0.461$_{\pm.011}$ | 0.644$_{\pm.038}$ | 0.730$_{\pm.060}$ | 1.194$_{\pm.034}$ | 1.092$_{\pm.028}$ | 1.169$_{\pm.055}$ | 0.762$_{\pm.115}$ |
| | 16 | **0.374**$_{\pm.061}$ | 0.488$_{\pm.010}$ | 0.681$_{\pm.018}$ | 0.983$_{\pm.139}$ | 1.929$_{\pm.105}$ | 1.332$_{\pm.125}$ | 1.701$_{\pm.088}$ | 0.681$_{\pm.104}$ |
| | | 0.488$_{\pm.039}$ | **0.463**$_{\pm.018}$ | 0.640$_{\pm.008}$ | 0.760$_{\pm.062}$ | 1.029$_{\pm.030}$ | 0.879$_{\pm.037}$ | 0.999$_{\pm.023}$ | 0.641$_{\pm.055}$ |
| | 32 | **0.334**$_{\pm.008}$ | 0.464$_{\pm0.007}$ | 0.477$_{\pm.035}$ | 0.669$_{\pm.092}$ | 6.153$_{\pm.715}$ | 1.514$_{\pm.042}$ | 1.922$_{\pm.032}$ | 0.578$_{\pm.062}$ |
| | | **0.461**$_{\pm.004}$ | 0.543$_{\pm0.004}$ | 0.537$_{\pm.019}$ | 0.646$_{\pm.048}$ | 1.689$_{\pm.112}$ | 0.925$_{\pm.016}$ | 1.068$_{\pm.011}$ | 0.597$_{\pm.035}$ |
| | 64 | **0.349**$_{\pm.039}$ | 0.425$_{\pm.006}$ | 0.418$_{\pm.021}$ | 0.484$_{\pm.051}$ | 2.419$_{\pm.520}$ | 1.150$_{0.118}$ | 1.654$_{\pm.117}$ | 0.463$_{\pm.081}$ |
| | | **0.473**$_{\pm.024}$ | 0.523$_{0.004}$ | 0.517$_{\pm.013}$ | 0.551$_{\pm.027}$ | 1.223$_{\pm.127}$ | 0.835$_{\pm.045}$ | 0.987$_{\pm.036}$ | 0.546$_{\pm.042}$ |
| Wind | 8 | **0.681**$_{\pm.075}$ | 1.854$_{\pm.032}$ | 1.321$_{\pm.379}$ | 1.942$_{\pm.101}$ | 12.53$_{\pm2.25}$ | 12.67$_{\pm1.75}$ | 11.35$_{\pm6.61}$ | 2.006$_{\pm.145}$ |
| | | **0.596**$_{\pm.052}$ | 1.223$_{\pm.014}$ | 0.863$_{\pm.143}$ | 1.067$_{\pm.086}$ | 1.370$_{\pm.107}$ | 1.440$_{\pm.059}$ | 1.305$_{\pm.369}$ | 1.103$_{\pm.100}$ |
| | 16 | 1.033$_{\pm.062}$ | 1.955$_{\pm.015}$ | **0.894**$_{\pm.038}$ | 1.262$_{\pm.178}$ | 13.96$_{\pm.1.53}$ | 12.86$_{\pm2.60}$ | 13.79$_{\pm5.37}$ | 1.138$_{\pm.205}$ |
| | | **0.757**$_{\pm.053}$ | 1.247$_{\pm.011}$ | 0.785$_{\pm.037}$ | 0.843$_{\pm.066}$ | 1.347$_{\pm.060}$ | 1.240$_{\pm.070}$ | 1.261$_{\pm.171}$ | 0.862$_{\pm.092}$ |
| | 32 | **1.224**$_{\pm.060}$ | 1.784$_{\pm.011}$ | 1.266$_{\pm.006}$ | 1.434$_{\pm.126}$ | 5.398$_{\pm.179}$ | 13.10$_{\pm.955}$ | 15.33$_{1.904}$ | 1.480$_{\pm.072}$ |
| | | **0.869**$_{\pm.074}$ | 1.200$_{\pm.007}$ | 0.872$_{\pm.010}$ | 0.920$_{\pm.077}$ | 1.434$_{\pm.013}$ | 1.518$_{\pm.020}$ | 1.614$_{\pm.118}$ | 0.987$_{\pm.010}$ |
| | 64 | 0.902$_{\pm.024}$ | 1.652$_{\pm.010}$ | **0.786**$_{\pm.022}$ | 0.898$_{\pm.095}$ | 4.403$_{\pm.301}$ | 3.857$_{\pm.597}$ | 3.564$_{\pm.293}$ | 1.374$_{\pm1.02}$ |
| | | 0.761$_{\pm.021}$ | 1.167$_{\pm.005}$ | **0.742**$_{\pm.017}$ | 0.789$_{\pm.048}$ | 1.361$_{\pm.021}$ | 1.110$_{\pm.143}$ | 1.152$_{\pm.081}$ | 0.842$_{\pm.215}$ |

Table 7: Performance comparisons of TSF in 100%-Electricity and 100%-ETTm1 datasets in terms of MSE and CRPS. The best results are highlighted in boldface.

| Model | | D$^3$VAE | NVAE | $\beta$-TCVAE | f-VAE | DeepAR | TimeGrad | GP-copula | VAE |
|---|---|---|---|---|---|---|---|---|---|
| Electricity | 16 | **0.330**$_{\pm.033}$ | 1.408$_{\pm.015}$ | 0.801$_{\pm.001}$ | 0.765$_{\pm.026}$ | 33.93$_{\pm1.85}$ | 46.69$_{\pm3.13}$ | 50.25$_{\pm4.39}$ | 0.680$_{\pm.022}$ |
| | | **0.445**$_{\pm.020}$ | 0.999$_{\pm.006}$ | 0.723$_{\pm.001}$ | 0.710$_{\pm.013}$ | 2.650$_{\pm.030}$ | 2.702$_{\pm.079}$ | 2.796$_{\pm.072}$ | 0.675$_{\pm.008}$ |
| | 32 | **0.336**$_{\pm.017}$ | 1.403$_{\pm.014}$ | 0.802$_{\pm.001}$ | 0.748$_{\pm.033}$ | 46.10$_{\pm2.00}$ | 30.94$_{\pm1.70}$ | 32.13$_{\pm1.96}$ | 0.727$_{\pm.033}$ |
| | | **0.444**$_{\pm.015}$ | 0.997$_{\pm.007}$ | 0.724$_{\pm.001}$ | 0.703$_{\pm.016}$ | 2.741$_{\pm.011}$ | 2.476$_{\pm.042}$ | 2.591$_{\pm.064}$ | 0.692$_{\pm.014}$ |
| ETTm1 | 16 | **0.018**$_{\pm.002}$ | 2.577$_{\pm.047}$ | 0.918$_{\pm.015}$ | 1.285$_{\pm.236}$ | 73.82$_{\pm3.25}$ | 68.26$_{\pm2.04}$ | 66.97$_{\pm2.02}$ | 1.335$_{\pm.156}$ |
| | | **0.102**$_{\pm.003}$ | 1.509$_{0.016}$ | 0.766$_{\pm.005}$ | 0.911$_{\pm.090}$ | 1.136$_{\pm.013}$ | 1.153$_{\pm.019}$ | 1.111$_{\pm.016}$ | 0.923$_{\pm.056}$ |
| | 32 | **0.034**$_{\pm.001}$ | 2.622$_{\pm.057}$ | 0.929$_{\pm.010}$ | 1.420$_{\pm.073}$ | 68.11$_{\pm2.60}$ | 53.47$_{\pm26.1}$ | 63.67$_{\pm1.14}$ | 1.223$_{\pm.213}$ |
| | | **0.144**$_{\pm.006}$ | 1.524$_{\pm.018}$ | 0.770$_{\pm.004}$ | 0.960$_{\pm.021}$ | 1.121$_{\pm.024}$ | 1.083$_{\pm.109}$ | 1.097$_{\pm.008}$ | 0.888$_{\pm.082}$ |

# D Supplementary Experimental Results

## D.1 Comparisons of Predictive Performance for TSF Under Different Settings

**Longer-Term Time Series Forecasting.** To further inspect the performance of our method, we additionally conduct more experiments for longer-term time series forecasting. In particular, by configuring the output length with 32 and 64[6], we compare D$^3$VAE to other baselines in terms of MSE and CRPS, and the results (including short-term and long-term) are reported in Table 6. We can conclude that D$^3$VAE also outperforms the competitive baselines consistently under the longer-term forecasting settings.

**Time Series Forecasting in Full Datasets.** Moreover, we evaluate the predictive performance for time series forecasting in two "full-version" datasets, i.e. 100%-Electricity and 100%-ETTm1. The split of train/validation/test is 7:1:2 which is the same as the main experiments. The comparisons in terms of MSE and CRPS can be found in Table 7. With sufficient data, compared to previous state-of-the-art generative models, the MSE and CRPS reductions of our method are also satisfactory under different settings (including input-16-predict-16 and input-32-predict-32). For example, in Electricity dataset, compared to the second best results, D$^3$VAE achieves 52% (0.680 $\rightarrow$ 0.330) and 54% (0.727 $\rightarrow$ 0.336) MSE reductions, and 34% (0.675 $\rightarrow$ 0.445) and 36% (0.692 $\rightarrow$ 0.444) CRPS reductions, under input-16-predict-16 and input-32-predict-32 settings, respectively.

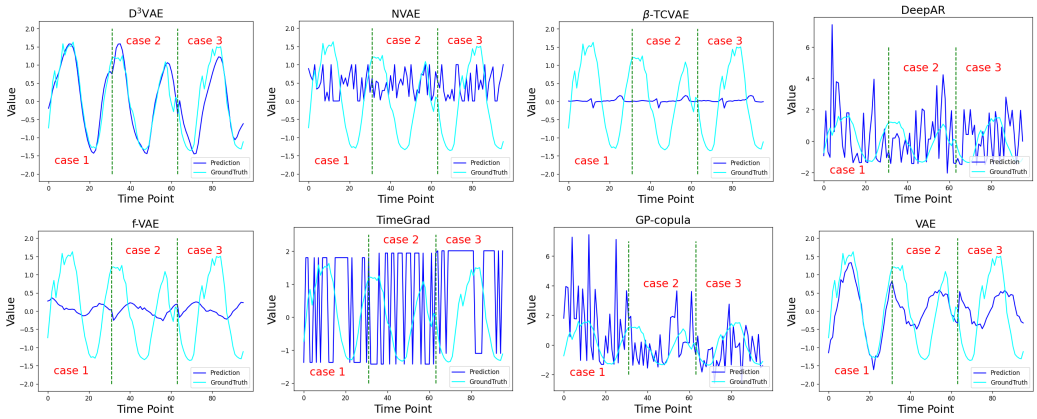



Figure 7: The case study of forecasting results on the Traffic dataset under input-32-predict-32 settings. Only the last dimension is plotted. To demonstrate the forecasting results in a long range, we show the predictions of three cases that are ordered chronologically without overlapping.

[6]The length of input time series is the same as the output time series.

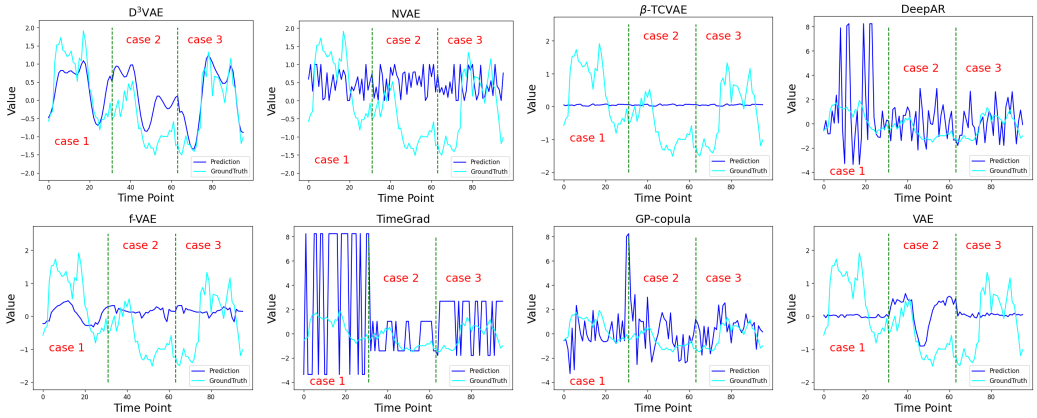



Figure 8: Case study of the forecasting results from the Electricity dataset (same settings as Fig. 7).

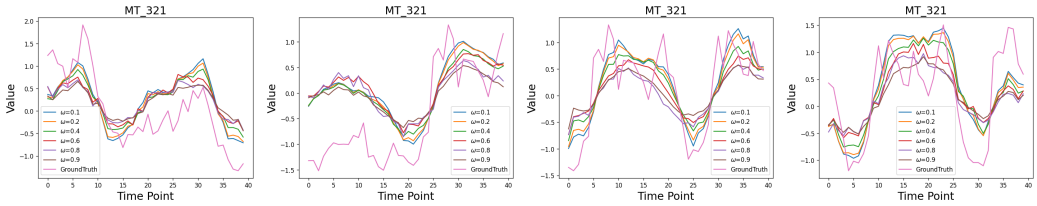



Figure 9: Forecasting results (under the input-40-predict-40 setting) of a case from the Electricity dataset with $\omega$ increasing from 0.1 to 0.9.

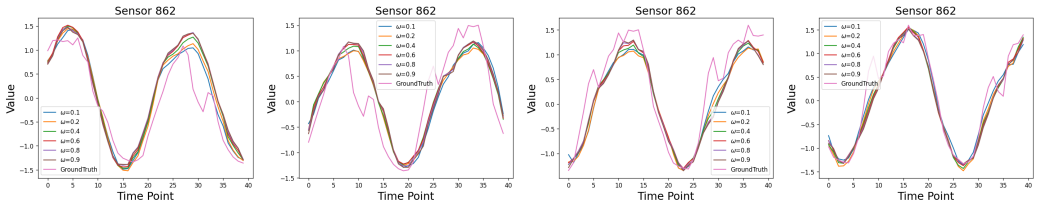



Figure 10: Forecasting results of a case from the Traffic dataset under the input-40-predict-40 setting.

### D.2 Case Study

We showcase the prediction results of our model and seven baseline models on the Traffic and Electricity datasets in Figs. 7 and 8. Our model can provide the most accurate forecasting results in terms of trends and variations.

### D.3 The Effect of Scale Parameter $\omega$

We demonstrate the forecasting results with different values of $\omega$ on Electricity and Traffic datasets, and the results are plotted in Figs. 9 and 10. It can be depicted that larger or smaller $\omega$ would lead to deviated prediction which is far from the ground truth. Therefore, the value of $\omega$ does affect the prediction performance, which should be tuned properly.

### D.4 Sensitivity Analysis of Trade-off Parameters in Reconstruction Loss $\mathcal{L}$

To examine the effect of the trade-off hyperparameters in loss $\mathcal{L}$, we plot the mean square error (MSE) against different values of trade-off parameters, i.e. $\psi$, $\lambda$ and $\gamma$, in the Traffic dataset. Note that the relative value of MSE is plotted to ensure the difference is distinguishable. This experiment is conducted under different settings: input-8-predict-8, input-16-predict-16, and input-32-predict-32. For $\psi$, the value ranges from 0 to 0.8, $\lambda$ ranges from 0 to 1.6, and $\gamma$ ranges from 0 to 0.5. The results are shown in Fig. 11. We can see that the performance of the model varies slightly as the trade-off parameters take different values, which shows that our model is robust enough against different trade-off parameters.

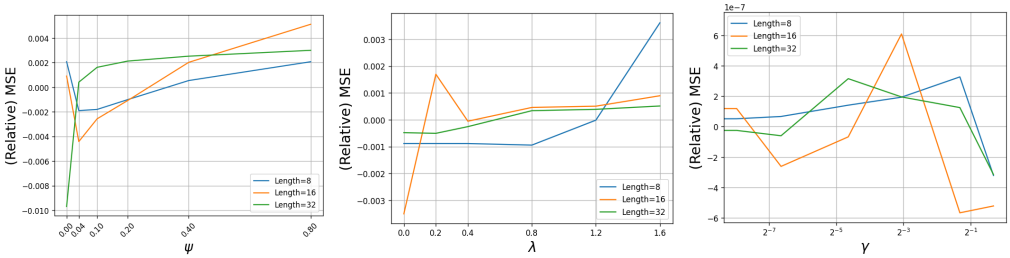



Figure 11: Sensitivity analysis of the trade-off hyperparameters in reconstruction loss $\mathcal{L}$. To highlight the changes in prediction performance against hyperparameters, the relative value of MSE is used.

Table 8: Performance comparisons of D$^3$VAE w.r.t. varying the length of (input and output) time series and the data size. The results are reported on the Electricity dataset.

| Length | Metric | Percentage of Full Electricity Data | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | 100% | 80% | 60% | 40% | 20% | 10% | 5% |
| 8 | MSE | $0.258_{\pm.019}$ | $0.227_{\pm.016}$ | $0.368_{\pm.019}$ | $0.389_{\pm.034}$ | $3.861_{\pm.480}$ | $0.693_{\pm.223}$ | $0.206_{\pm.018}$ |
| | CRPS | $0.383_{\pm.015}$ | $0.355_{\pm.015}$ | $0.453_{\pm.009}$ | $0.504_{\pm.034}$ | $1.728_{\pm.110}$ | $0.673_{\pm.132}$ | $0.352_{\pm.016}$ |
| 16 | MSE | $0.330_{\pm.033}$ | $0.253_{\pm.018}$ | $0.343_{\pm.024}$ | $0.463_{\pm.089}$ | $4.428_{\pm.694}$ | $0.401_{\pm.068}$ | $0.247_{\pm.056}$ |
| | CRPS | $0.445_{\pm.020}$ | $0.373_{\pm.014}$ | $0.433_{\pm.015}$ | $0.562_{\pm.049}$ | $1.858_{\pm.147}$ | $0.496_{\pm.047}$ | $0.378_{\pm.036}$ |
| 32 | MSE | $0.336_{\pm.017}$ | $0.300_{\pm.039}$ | $0.484_{\pm.048}$ | $0.739_{\pm.209}$ | $5.029_{\pm.811}$ | $0.884_{\pm.237}$ | $0.304_{\pm.094}$ |
| | CRPS | $0.444_{\pm.015}$ | $0.413_{\pm.034}$ | $0.537_{\pm.025}$ | $0.693_{\pm.099}$ | $1.989_{\pm.172}$ | $0.723_{\pm.112}$ | $0.418_{\pm.065}$ |

Table 9: Performance comparisons of D$^3$VAE w.r.t. varying the length of (input and output) time series and the data size. The results are reported on the Traffic dataset.

| Length | Metric | Percentage of Full Traffic Data | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | 100% | 80% | 60% | 40% | 20% | 10% | 5% |
| 8 | MSE | $0.370_{\pm.021}$ | $0.215_{\pm.016}$ | $0.063_{\pm.002}$ | $0.062_{\pm.002}$ | $0.054_{\pm.004}$ | $0.210_{\pm.012}$ | $0.081_{\pm.003}$ |
| | CRPS | $0.415_{\pm.013}$ | $0.347_{\pm.015}$ | $0.184_{\pm.003}$ | $0.179_{\pm.005}$ | $0.172_{\pm.008}$ | $0.251_{\pm.005}$ | $0.207_{\pm.003}$ |
| 16 | MSE | $0.272_{\pm.007}$ | $0.189_{\pm.006}$ | $0.063_{\pm.001}$ | $0.058_{\pm.003}$ | $0.056_{\pm.003}$ | $0.178_{\pm.006}$ | $0.081_{\pm.009}$ |
| | CRPS | $0.334_{\pm.009}$ | $0.321_{\pm.008}$ | $0.180_{\pm.002}$ | $0.168_{\pm.006}$ | $0.169_{\pm.005}$ | $0.239_{\pm.007}$ | $0.200_{\pm.003}$ |
| 32 | MSE | $0.307_{\pm.015}$ | $0.197_{\pm.005}$ | $0.064_{\pm.002}$ | $0.063_{\pm.002}$ | $0.056_{\pm.004}$ | $0.191_{\pm.011}$ | $0.091_{\pm.007}$ |
| | CRPS | $0.363_{\pm.008}$ | $0.335_{\pm.004}$ | $0.179_{\pm.002}$ | $0.179_{\pm.003}$ | $0.170_{\pm.005}$ | $0.235_{\pm.008}$ | $0.216_{\pm.012}$ |

### D.5 Scalability Analysis of Varying Time Series Length and Dataset Size

We additionally investigate the scalability of D$^3$VAE against different lengths of the time series and varying amounts of available data. The experiments are conducted on the Electricity and Traffic datasets, and the results are reported in Tables 8 and 9, respectively. We can observe that the predictive performance of D$^3$VAE is relatively stable under different settings. In particular, the longer target series to predict the worse performance might be obtained. Besides, when the amount of available data is shrunk, D$^3$VAE performs more stable than expected. Note that on the 20%-Electricity dataset, the performance of D$^3$VAE is much worse than other subsets of the Electricity dataset, this is mainly because the sliced 20%-Electricity dataset involves more irregular values.

## E Disentanglement for Time Series Forecasting

Fig. 13 illustrates the disentanglement of latent variable $Z$ for time series forecasting. It is difficult to choose suitable disentanglement factors under the unsupervised learning of disentanglement. Therefore, we attempt to inspect the TSF performance against different numbers of factors to be disen-

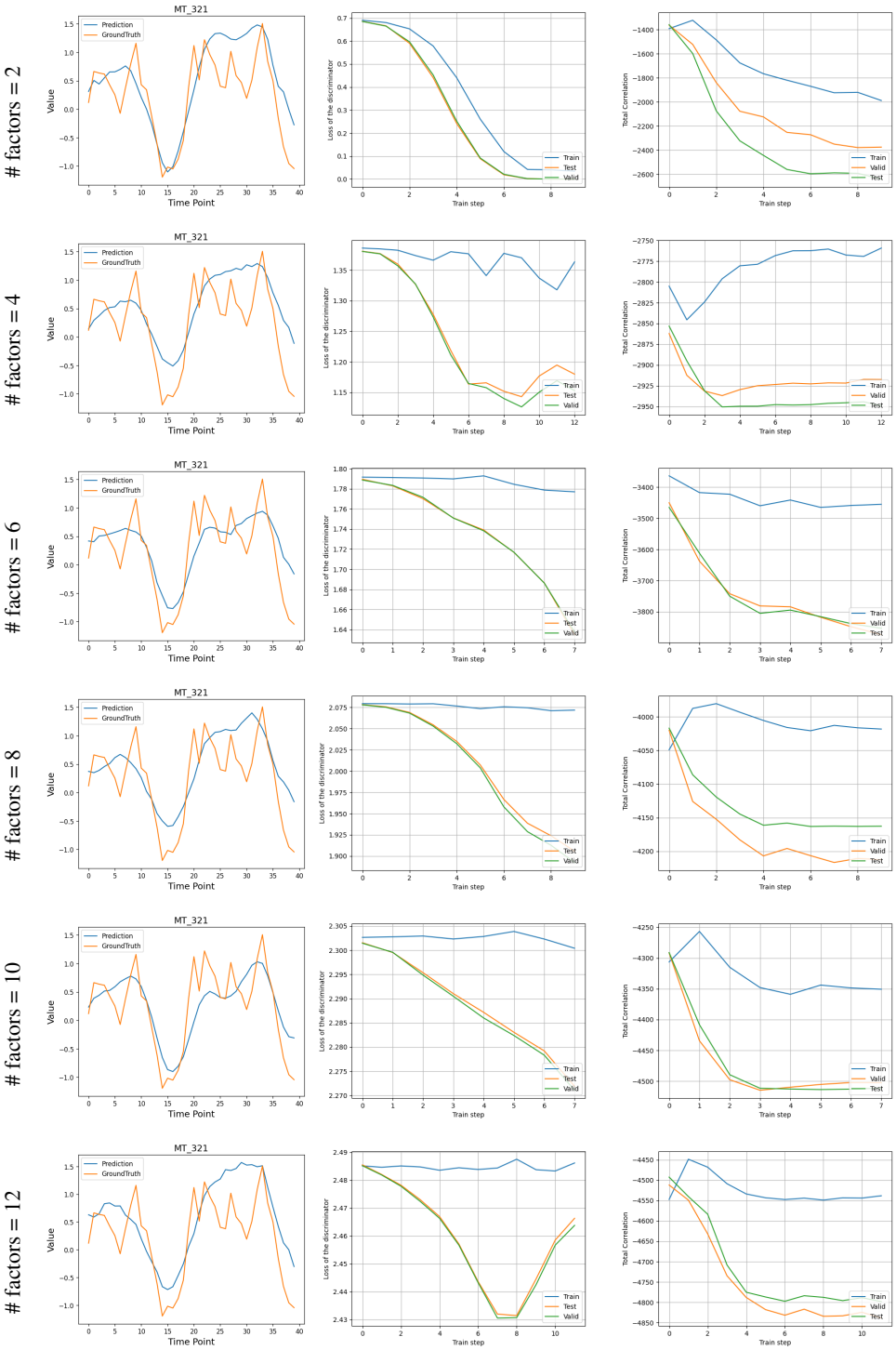



Figure 12: We showcase an instance from the Electricity dataset and demonstrate the results when different numbers of factors in disentanglement are adopted. For each row, from left to right, the prediction result of TSF, the learning curve of the discriminator, and the total correlation are plotted, respectively.

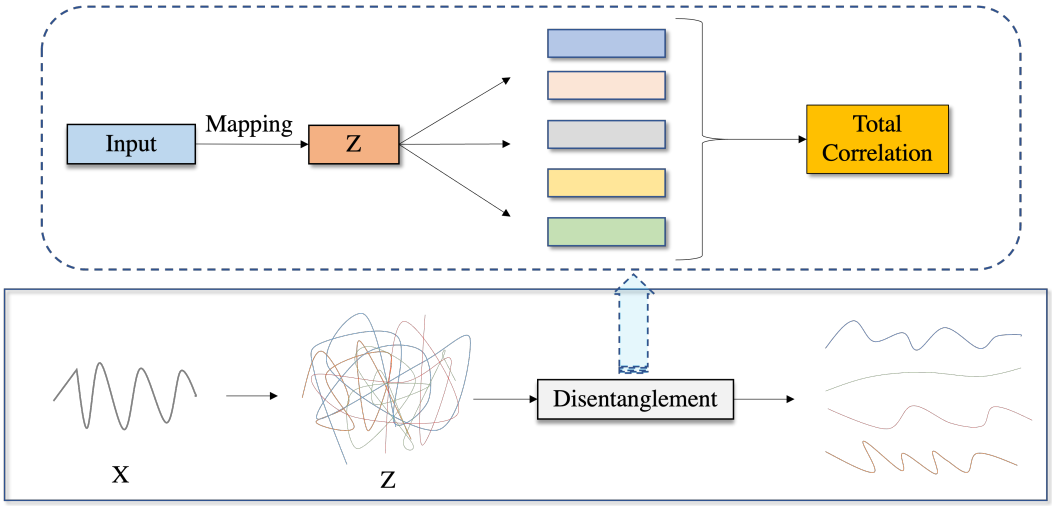



Figure 13: Disentangling latent variable $Z$ of time series. Specifically, the input $X$ is first mapped into $Z$. Then $\forall z_i \in Z$ is decomposed as $z_i = [z_{i,1}, \cdots, z_{i,m}]$ and the metric of total correlation is utilized to minimize the inter-dependencies among "hand-crafted" factors. In this way, the disentangled factors tend to be not only discriminative but also informative.

**Algorithm 3** Train a discriminator for time series disentanglement.

**repeat**
  Initialize the loss of a discriminator $\mathcal{D}_\varphi$: $L(\mathcal{D}_\varphi) = 0$
  Decompose the latent variable generated in Algorithm 1 as $z_i = [z_{i,1}, z_{i,2}, ..., z_{i,m}]$ $(i = 1, \cdots, n)$
  **for** $z_i$ in Z **do**
    $L = L + \sum_{j=1}^{m} \|\mathcal{D}_\varphi(z_{i,j}) - j\|^2$
  **end for**
  Optimize the discriminator: $\varphi \leftarrow \text{argmin}(L)$
**until** Convergence

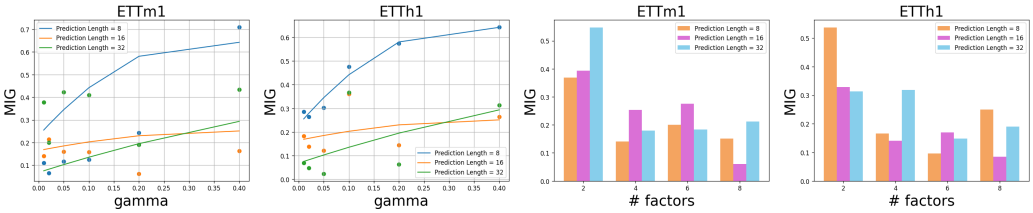
tangled. We implement a simple classifier as a discriminator to further evaluate the disentanglement quality in Fig. 12 (and Algorithm 3 demonstrates the training procedure of the discriminator). To be specific, we take different dimensions of $Z$ as the factors to be disentangled: $z_i = [z_{i,1}, \cdots, z_{i,m}]$ $(z_i \in Z)$, then an instance consisting of factor and label $(z_{i,j}, j)$ is constructed. We shuffle these $m$ examples for each $z_i$ and attempt to classify them with a discriminator, then the disentanglement can be evaluated by measuring the loss of the discriminator. The learning curve of the discriminator can be leveraged to assess the disentanglement, and the discriminator is implemented by an MLP with six nonlinear layers and 100 hidden states. The results of prediction, discriminator loss, and the total correlation w.r.t. different numbers of factors are plotted in Fig. 12, respectively. As shown in Fig. 12, the number of factors does affect the prediction performance, as well as the disentanglement quality. On the other hand, the learning curves can be converged when different number of factors are adopted, which validates that the disentanglement of the latent factors is of high quality.

In addition to the above method evaluating the disentanglement indirectly, we adopt another metric named Mutual Information Gap (MIG) [15] to evaluate the quality of disentanglement in a more straightforward way. Specifically, for a latent variable $z_i \in Z$, the mutual information between $z_{i,j}$, and a factor $v_k \in [1, m]$ can be calculated by

$$I_d(z_{i,j}, v_k) = \mathbb{E}_{q(z_{i,j}, v_k)}[log \sum_{d \in \mathcal{S}_{v_k}} q(z_{i,j}|d)p(d|v_k)] + H(z_{i,j}) \,, \tag{16}$$

where $d$ denotes the sample of $z_{i,j}$ and $\mathcal{S}_{v_k}$ is the support set of $v_k$. Then, for $z_{i,j}$

$$MIG(z_{i,j}) = \frac{1}{m} \sum_{1}^{m} \frac{1}{H(v_k)} (max(I_d(z_{i,j}, v_k)) - submax(I_d(z_{i,j}, v_k))) \,, \tag{17}$$




(a) (b) (c) (d)

Figure 14: We evaluate the quality of disentanglement on ETTm1 and ETTh1 datasets in terms of mutual information gap (MIG). **(a-b)** The scatter plots of the MIG against varying weights $\gamma$ in loss (Eq. (14) in the main text). **(c-d)** MIG v.s. different number of factors.

where $submax$ means the second max value of $I_d(z_{i,j}, v_k)$, then the MIG of $Z$ can be obtained as

$$MIG(Z) = \sum_{i=1}^{n} MIG(z_i), \quad MIG(z_i) = \frac{1}{m} \sum_{j=1}^{m} MIG(z_{i,j}) \,. \tag{18}$$

We evaluate the quality of disentanglement in terms of MIG on ETTm1 and ETTh1 datasets, respectively, which can be seen in Fig. 14. From Figs. 14a and 14b, when the weight of disentanglement (i.e., $\gamma$ in Eq. (14) of the main text) grows, the disentangled factors are of higher quality. In other words, the latent variables can be disentangled with the help of the disentanglement module in $D^3$VAE. In addition, we examine the changes in MIG against different numbers of factors. We can observe that the difficulty of disentanglement climbs up as the number of factors increases.

## F Model Inspection: Coupled Diffusion Process

To gain more insights into the coupled diffusion process, we demonstrate how a time series can be diffused under different settings in terms of variance schedule $\boldsymbol{\beta}$ and the max number of diffusion steps $T$. The examples are illustrated in Fig. 15. It can be seen that when larger diffusion steps or a wider variance schedule is employed, the diffused series deviates far from the original data gradually, which may result in the loss of useful signals, like, temporal dependencies. Therefore, it is important to choose a suitable variance schedule and diffusion steps to ensure that the distribution space is deviated enough without losing useful signals.

## G Necessity of Data Augmentation for Time Series Forecasting

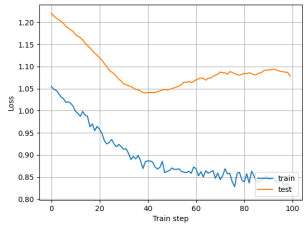
Limited data would result in overfitting and poor performance. To demonstrate the necessity of enlarging the size of data for time series forecasting when deep models are employed, we implement a two-layer RNN and evaluate how many time points are required to ensure the generalization ability. A synthetic dataset is adopted for this demonstration.

According to [23], we generate a toy time series dataset with $n$ time points in which each point is a $d$-dimension variable:

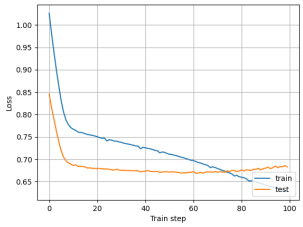
$$w_t = 0.5w_{t-1} + \tanh(0.5w_{t-2}) + \sin(w_{t-3}) + \epsilon \,, \quad X = [w_1, w_2, ..., w_n] * F + v$$

where $w_t \in \mathcal{R}^2$, $F \in \mathcal{R}^{2\times d} \sim \mathcal{U}[-1, 1]$, $\epsilon \sim \mathcal{N}(0, I)$, $v \sim \mathcal{N}(0, 0.5I)$, and $d = 5$. An input-8-predict-8 window is utilized to roll this synthetic dataset. We split this synthetic dataset into training and test sets with a ratio of 7:3. We train the RNN in 100 epochs at most, and the MSE loss of training and testing are plotted in Fig. 16. It can be seen that the inflection points of the loss curves move back gradually and disappear as increasing the size of the dataset. Besides, with fewer time points, like, 400, the model can be overfitted more easily.

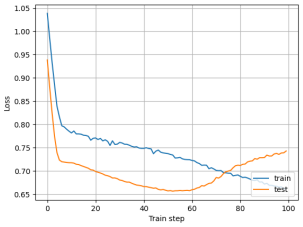
Figure 15: Diffused time series with different variance schedules and diffusion steps. We randomly choose a sample series from the synthetic dataset D2 and plot the original time series data, as well as the diffused series.



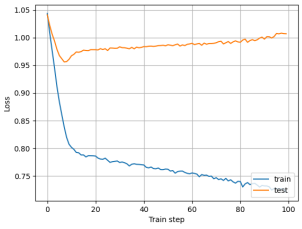

(a) 400 time points.



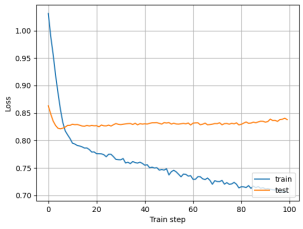

(b) 800 time points.



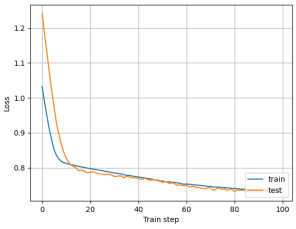

(c) 1000 time points.




(d) 1200 time points.




(e) 1400 time points.




(f) 1600 time points.

Figure 16: The curves of training and testing losses when the available time series data are of different sizes.