Conditional Diffusion Model with Nonlinear Data Transformation for Time Series Forecasting

J Rishi [*1] **GVS Mothish** [*1] **Deepak Subramani** [1]

Abstract

Time-series forecasting finds application across domains such as finance, climate science, and energy systems. We introduce the Conditional Diffusion with Nonlinear Data Transformation Model (CN-Diff), a generative framework that employs novel nonlinear transformations and learnable conditions in the forward process for time series forecasting. A new loss formulation for training is proposed, along with a detailed derivation of both forward and reverse process. The new additions improve the diffusion model's capacity to capture complex time series patterns, thus simplifying the reverse process. Our novel condition facilitates learning an efficient prior distribution. This also reduces the gap between the true negative log-likelihood and its variational approximation. CN-Diff is shown to perform better than other leading time series models on nine real-world datasets. Ablation studies are conducted to elucidate the role of each component of CN-Diff.

1. Introduction

Time series forecasting has become an essential element in modern data-driven decision-making processes, spanning a wide range of fields including financial pricing analysis (Kim, 2003), economics (Henrique et al., 2019), transportation planning (Huang et al., 2023), energy (Dumas et al., 2022) and various other fields (Li et al., 2024a; Rasul et al., 2022). There has been substantial advancement in time series modeling, beginning with traditional statistical techniques such as ARIMA and state-space models, which initially dominated forecasting efforts. Later, various architectures of deep neural networks, such as recurrent neural networks (Hewamalage et al., 2021), convolutional neural networks (Yue et al., 2022), and transformers (Vaswani, 2017a) have been used for time-series modeling.

In addition to traditional deep learning architectures, diffusion models have recently gained prominence as effective tools for generative tasks. Their outstanding performance has been evidenced in various domains, exceeding the capabilities of conventional generative approaches in image generation (Ho et al., 2020; Dhariwal & Nichol, 2021), video synthesis (Harvey et al., 2022; Blattmann et al., 2023), robotics (Mothish et al., 2024) and cross-modal applications (Saharia et al., 2022). Given this impressive success, there is a growing research interest in leveraging the potential of diffusion models for time series forecasting, leading to the development of several diffusion-based frameworks for time series modeling. Each of these frameworks is specifically designed to tackle the complexities and difficulties that arise in the context of temporal forecasting (Tashiro et al., 2021; Li et al., 2024b; Cao et al., 2024; Shen & Kwok, 2023; Shen et al., 2024; Lopez Alcaraz & Strodthoff, 2023). These approaches largely rely on conditional diffusion models. For instance, TimeDiff (Shen & Kwok, 2023) employs future mixup as a condition-guided diffusion model. TimeGrad (Rasul et al., 2021) merges a standard diffusion model with the recurrent neural network's hidden states. CSDI (Tashiro et al., 2021) uses self-supervised masking to steer a non-autoregressive denoising process. While effective, these time-series diffusion models do not completely exploit diffusion model properties.

Conventional training for conditional diffusion models involves gradually adding a fixed linear Gaussian noise at each forward stage. The reverse process is then optimized to closely replicate the forward process but with conditions only applied to the reverse process. Firstly, this approach confines diffusion models in the non-learnable forward process. Secondly, simplifying into an isotropic Gaussian prior complicates the data generation process in the reverse process for time series applications.

This paper introduces the Conditional Diffusion with Nonlinear Data Transformation Model (CN-Diff) (Refer to Figure 1), a framework incorporating a nonlinear time-dependent learnable data transformation into the forward process along-

[*]Equal contribution [1]Department of Computational and Data Sciences, Indian Institute of Science, Bengaluru, India. Correspondence to: J Rishi <rishij@iisc.ac.in>, GVS Mothish <mothishg@iisc.ac.in>, Deepak Subramani <deepakns@iisc.ac.in>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

side the learnable condition. This modification results in a non-Markovian series of latent variables, each formed by transforming the data and subsequently adding noise, which is learned through the reverse process. Code is publicly available at https://github.com/quest-lab-iisc/CNDiff

The main contributions of the present paper are as follows.

1. We introduce a nonlinear, time-dependent data transformation along with a learnable condition for data generation in the forward process for time series forecasting.

2. We develop an objective function based on a non-Markovian learnable reverse generative process to train CN-Diff.

3. Experimental results demonstrate that our model surpasses or matches the performance of other leading time series forecasting models at the time of writing.

2. Preliminaries

2.1. Diffusion Model

Diffusion models are generative frameworks employing latent variables. For a data sample $x \sim q(x)$, a forward noising process produces latent variables $x^0, x^1, x^2, \ldots, x^T$. The reverse process aims to reconstruct x by generating the same latent variables in reverse order.

In the standard diffusion model, the forward process follows a linear Gaussian Markov chain (Sohl-Dickstein et al., 2015; Ho et al., 2020). In denoising diffusion probabilistic models (DDPM), at step t, x^t is generated by modifying the previous state x^{t-1} (multiplied by $\sqrt{\alpha_t}$) with zero-mean Gaussian noise and variance $(1-\alpha_t)$, that is, $q(x^t|x^{t-1}) = \mathcal{N}\left(x^t; \sqrt{\alpha_t}x^{t-1}, (1-\alpha_t)I\right)$. We derive the marginal distribution as $q(x^t|x) = \mathcal{N}(x^t; \sqrt{\bar{\alpha}_t}x, (1-\bar{\alpha}_t)I)$, where $\bar{\alpha}_t$ is defined as $\prod_{s=1}^{t} \alpha_s$. Hence, $x^t = \sqrt{\bar{\alpha}_t}x + \sqrt{1-\bar{\alpha}_t}\epsilon$, with $\epsilon \sim \mathcal{N}(0, I)$. Leveraging these marginal distributions, the joint distribution for the latent variables $x^0, x^1, x^2, \ldots, x^T$ is

$$q(x^{0:T}|x) = \prod_{t=1}^{T} q(x^t \mid x^{t-1})\,.$$

The forward process is typically fixed, lacking trainable parameters, and it is designed so that $q(x^0|x) \approx \delta(x^0 - x)$ and $q(x^T|x) \approx \mathcal{N}(x^T; 0, I)$. If accessing $q(x^{t-1}|x^t)$ were feasible, we could sample from $x^T \sim \mathcal{N}(x^T; 0, I)$ and reverse it to yield $x^0 \sim q(x^0) \approx q(x)$. The distribution $q(x^{t-1}|x^t)$ depends implicitly on $q(x)$, forming a complex relationship. Hence, we approximate the reverse process through a Markov chain adopting the form

$$p_\theta(x^{0:T}) = p(x^T)\prod_{t=1}^{T} p_\theta(x^{t-1}|x^t),$$

with $p(x^T) = \mathcal{N}(x^T; 0, I)$. The integration of the forward process q with the reverse process p_θ is akin to a (hierarchical) variational autoencoder (Kingma, 2013a; Rezende et al., 2014). During training, the standard variational bound of the negative log-likelihood is minimized. In the case of DDPM (Ho et al., 2020), the loss to be minimized is

$$\mathcal{L} = \mathbb{E}_q\Bigg[\underbrace{D_{\mathrm{KL}}\left(q(x^T|x) \,\|\, p(x^T)\right)}_{\mathcal{L}_{\text{prior}}} - \underbrace{\log p_\theta(x|x^0)}_{\mathcal{L}_{\text{rec}}} + \sum_{t=1}^{T}\underbrace{D_{\mathrm{KL}}\left(q(x^{t-1}|x^t, x) \,\|\, p_\theta(x^{t-1}|x^t)\right)}_{\mathcal{L}_{\text{diff}}}\Bigg].$$

Given the fixed nature of process q and distribution $p_\theta(x^T) = p(x^T)$, the prior term $\mathcal{L}_{prior}$ can be disregarded as it does not depend on parameters θ. Since $\log p_\theta(x|x^0)$ is often modeled by a Gaussian distribution with low variance, the reconstruction term $\mathcal{L}_{rec}$ also remains unaffected by θ. Consequently, the diffusion term $\mathcal{L}_{diff}$ is the only part that the model parameters θ influence. $\mathcal{L}_{diff}$ is the sum of Kullback–Leibler (KL) divergences between the posterior distribution in the forward process $q(x^{t-1}|x^t, x)$ and the assumed normal distribution $p_\theta(x^{t-1}|x^t) = \mathcal{N}(x^{t-1}; \mu_\theta(x^t, t), \Sigma_\theta(x^t, t))$ in the reverse process. Here, the variance $\Sigma_\theta(x^t, t)$ is set to $\sigma_t^2 I$, while the mean $\mu_\theta(x^t, t)$ is learned by a neural network with parameters θ. This process is typically viewed as a noise estimation or data prediction problem (Benny & Wolf, 2022). To estimate noise, the network ϵ_θ forecasts the noise in the diffused input x^t and then computes $\mu_\theta(x^t, t)$ using the formula $\left(\frac{1}{\sqrt{\alpha_t}}x^t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\epsilon_\theta(x^t, t)\right)$. The parameter θ is learned by minimizing the loss function $\mathcal{L}_\epsilon = \mathbb{E}_{x,\epsilon,t}\left[\left\|\epsilon - \epsilon_\theta(x^t, t)\right\|^2\right]$. Alternatively, in the data prediction approach, a denoising network x_θ is employed to derive an estimate $x_\theta(x^t, t)$ of the clean data x^0 from x^t, and then set to

$$\mu_\theta(x^t, t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x^t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_\theta(x^t, t)\,.$$

Here, the parameter θ is learned by minimizing the loss

$$\mathcal{L}_x = \mathbb{E}_{x,\epsilon,t}\left\|x - x_\theta(x^t, t)\right\|^2.$$

For time series diffusion models, forecasting x_θ has been found to be more effective than predicting ϵ_θ, as shown in (Feng et al., 2024; Shen & Kwok, 2023). Our work adopts this method.

2.2. Conditional Diffusion models for time series forecasting

In time series forecasting, the aim is to predict future values $x_{1:H} \in \mathbb{R}^{d\times H}$ based on past observations $x_{-L+1:0} \in \mathbb{R}^{d\times L}$. Here, L is the lookback window length, H is the forecast window length, and d is the count of variables, with superscripts and subscripts indicating diffusion time steps and time series values, respectively. When using conditional diffusion models for time series prediction, the following distribution is considered.

$$p_\theta(x^{0:T}_{1:H}|c) = p_\theta(x^T_{1:H}) \prod_{t=1}^{T} p_\theta(x^{t-1}_{1:H}|x^t_{1:H}, c).$$

where $x^T_{1:H} \sim \mathcal{N}(0, I)$ and c represent the condition, which changes depending on the specific models used (Rasul et al., 2021; Shen & Kwok, 2023; Shen et al., 2024; Li et al., 2024b). The procedure for denoising at step t is

$$p_\theta(x^{t-1}_{1:H}|x^t_{1:H}, c) = \mathcal{N}\left(x^{t-1}_{1:H}; \mu_\theta(x^t_{1:H}, t|c), \sigma_t^2 I\right)$$

During the inference process, the generated sample linked to $x^t_{1:H}$ is denoted as $\hat{x}^t_{1:H}$. We start by setting $\hat{x}^T_{1:H}$ as a noise vector sampled from $\mathcal{N}(0, I)$. By repeatedly executing the denoising step outlined in the above equation until reaching $t = 1$, we derive the final generated sample, $\hat{x}_{1:H}$.

3. Methodology

Consider time series data $x_{0:H} = x_0, x_1, x_2, \ldots, x_H$, where each x_i represents a vector with one or several variables with significant correlation between them at time i (Liu et al., 2023). Current conditional diffusion models for time series convert these correlated data distributions into an isotropic Gaussian prior by adjusting data points and progressively adding linear Gaussian noise to generate latent variables. Conditional generation is performed only during the reverse process to help in learning these latent variables. This method presents two issues: first, it confines diffusion models during the forward process, rendering them fixed and untrainable. Second, converting to a simple isotropic Gaussian prior complicates data generation during the reverse process. To address these challenges, we propose CN-Diff, a nonlinear data transformation framework that learns time-dependent distributions of latent space and incorporates conditional distributions in the forward process.

In what follows, first, we describe the nonlinear, time-dependent data transformation devised for the forward process. By integrating conditional information into the forward formulation, we create a diffusion model specifically designed for time series forecasting. Subsequently, we derive the variational bound loss pertinent to this model. We hypothesize that incorporating a more adaptable distribution

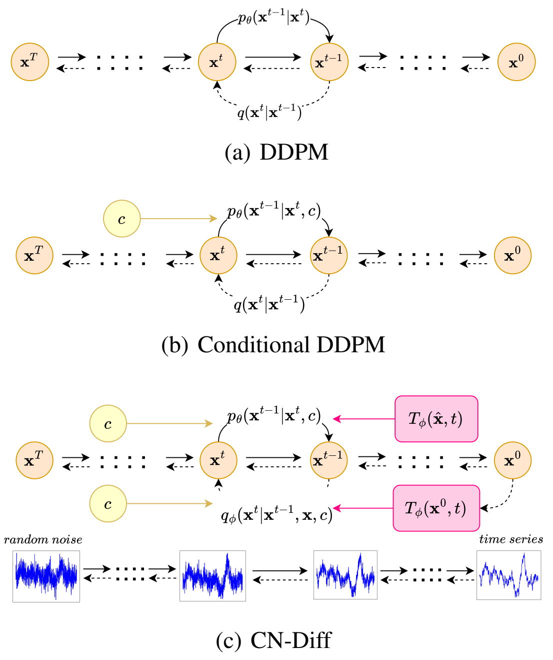
$p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t)$
$\mathbf{x}^T$
$\mathbf{x}^t$
$\mathbf{x}^{t-1}$
$\mathbf{x}^0$
$q(\mathbf{x}^t|\mathbf{x}^{t-1})$
(a) DDPM
c
$p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t, c)$
$\mathbf{x}^T$
$\mathbf{x}^t$
$\mathbf{x}^{t-1}$
$\mathbf{x}^0$
$q(\mathbf{x}^t|\mathbf{x}^{t-1})$
(b) Conditional DDPM
c
$p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t, c)$
$T_\phi(\hat{\mathbf{x}}, t)$
$\mathbf{x}^T$
$\mathbf{x}^t$
$\mathbf{x}^{t-1}$
$\mathbf{x}^0$
c
$q_\phi(\mathbf{x}^t|\mathbf{x}^{t-1}, \mathbf{x}, c)$
$T_\phi(\mathbf{x}^0, t)$
random noise
time series
(c) CN-Diff

Figure 1. Directed graphical models of (a) DDPM (b) Conditional DDPM for time series and (c) CN-Diff

in the forward process can markedly decrease the disparity between the log-likelihood and the variational bound, thereby enabling the model to more effectively capture temporal dynamics and variable interrelationships. Furthermore, embedding conditional information in the formulation aids in synchronizing the prior distribution with the intrinsic temporal dynamics seen in real-world data.

3.1. CN-Diff formulation and objective

Let us define the nonlinear time dependent data transformation of data for marginal distributions as

$$q_\phi(x^t_{1:H}|x_{1:H}) = \mathcal{N}\left(x^t_{1:H}; \sqrt{\bar{\alpha}_t} T_\phi(x_{1:H}, t), (1-\bar{\alpha}_t)I\right),$$

where $T_\phi(x_{1:H}, t) : \mathbb{R}^{d\times H} \times [0, T] \mapsto \mathbb{R}^{d\times H}$ is a nonlinear neural operator parameterized by ϕ that applies a time-dependent transformation to the data point $x_{1:H}$. For ease of notation, we replace $x_{1:H}$ with $\mathbf{x}$ in the following discussion.

We now introduce a learnable condition to the forward process, denoted c, inspired by a similar formulation used in image diffusion (Pandey et al., 2022). Consequently, the marginal distribution is (Figure 1):

$$\begin{aligned} q_\phi(\mathbf{x}^t|\mathbf{x}, c) = \mathcal{N}(\mathbf{x}^t; &\sqrt{\bar{\alpha}_t} T_\phi(\mathbf{x}, t) \\ &+ (1-\sqrt{\bar{\alpha}_t})c, (1-\bar{\alpha}_t)I). \end{aligned} \tag{1}$$

For $t = T$ along with an appropriately regulated noise schedule α_t, $\bar{\alpha}_T \approx 0$ results in $q_\phi(\mathbf{x}^T|\mathbf{x}, c) \approx \mathcal{N}(c, I)$. Simply

Algorithm 1 Training

```
Input: $\mathbf{x}(x_{1:H}), x_{-L+1:0}$
repeat
    $\epsilon \sim \mathcal{N}(0, I), t \sim U[1, T]$
    $\mathbf{x}^t \sim q_\phi(\mathbf{x}^t|\mathbf{x}, c)$
    Compute the loss in Eq. (4)
    Take numerical optimization step on: $\nabla \mathcal{L}_{\text{CN-Diff}}$
until until converged
```

Algorithm 2 Inference

```
$\mathbf{x}^T \sim \mathcal{N}(c, I)$
for $t = T$ to $1$ do
    $\hat{\mathbf{x}} = \hat{\mathbf{x}}_\theta(\mathbf{x}^t, t, c)$
    $\mathbf{x}^{t-1} = \zeta_1 \mathbf{x}^t + \zeta_2 c - T_\phi(\hat{\mathbf{x}}, t)\zeta_1\sqrt{\bar{\alpha}_t} + T_\phi(\hat{\mathbf{x}}, t-1)(\zeta_1\sqrt{\bar{\alpha}_t} + \zeta_0) + \sigma^2_{t-1|t}\epsilon$
end for
$\hat{\mathbf{x}} \sim p(\mathbf{x}|\mathbf{x}_0, c)$
```

put, the Gaussian $\mathcal{N}(c, I)$ serves as our learnable prior distribution and inference requires executing our reverse process on it.

We get the following posterior distribution $\big(q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c)\big)$ that satisfies Eq.(1):

$$\begin{aligned}\mathbf{x}^{t-1} &= \zeta_1 \mathbf{x}^t + \zeta_2 c \\ &- T_\phi(\mathbf{x}, t)\zeta_1\sqrt{\bar{\alpha}_t} \\ &+ T_\phi(\mathbf{x}, t-1)(\zeta_1\sqrt{\bar{\alpha}_t} + \zeta_0) + \sigma^2_{t-1|t}\epsilon\end{aligned}$$

See Appendix A.1 for the detailed derivation.

Given that our forward process exhibits non-Markovian characteristics(Eqn.7), employing a Markovian reverse process is inadequate for accurately reconstructing the forward process. We therefore define a framework that takes the reverse process as non-Markovian, which is one of the many that could be taken. Our non-Markovian trainable reverse generative process $p_\theta(\mathbf{x}^{0:T})$ is as follows:

$$p_\theta(\mathbf{x}^{0:T}) = p(\mathbf{x}^T)\prod_{t=1}^{T} p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t)p_\theta(\mathbf{x}^0|\mathbf{x}^t), \tag{2}$$

$$\text{where } p(\mathbf{x}^T) = \mathcal{N}(\mathbf{x}^T; c, I). \tag{3}$$

For this choice, our **Lemma** A.1 shows that, this approach results in a training objective analogous to that of DDPM even without non linear transform (Proof in Appendix A.1).

Consequently, the corresponding variational objective of CN-Diff is represented by following form (Appendix A.2).

$$\begin{aligned}\mathcal{L}_{\text{CN-Diff}} = \mathbb{E}_{q_\phi}\Bigg[&\underbrace{D_{\text{KL}}\left(q_\phi\left(\mathbf{x}^T|\mathbf{x}, c\right) \| p\left(\mathbf{x}^T|c\right)\right)}_{\mathcal{L}_{\text{prior}}} \\ &- \underbrace{\sum_{t=1}^{T} \log p_\theta\left(\mathbf{x}|\mathbf{x}^t, c\right)}_{\mathcal{L}_{\text{rec}}} \\ + \underbrace{\sum_{t=1}^{T} D_{\text{KL}}\left(q_\phi\left(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c\right) \| p_\theta\left(\mathbf{x}^{t-1}|\mathbf{x}^t, c\right)\right)}_{\mathcal{L}_{\text{diff}}}&\Bigg].\end{aligned} \tag{4}$$

Note that forward process is parametrized by ϕ by the transformation $T_\phi(.)$, making the objective different from (Ho et al., 2020). Consequently, the prior and reconstruction terms depend on ϕ, and therefore must be included in the optimization process.

In the reverse process we approximate posteriors as, $p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t, c) \approx q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t, \hat{\mathbf{x}}_\theta(x^t, t), c)$. Thus, $\mathcal{L}_{diff}$ and $\mathcal{L}_{prior}$ are

$$\begin{aligned}\mathcal{L}_{diff} &= D_{KL}\left(q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c) \| p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t, c)\right) \\ &= \frac{1}{2\sigma^2_{t-1|t}}\Big\|\zeta_1\sqrt{\bar{\alpha}_t}(T_\phi(\mathbf{x}, t) - T_\phi(\hat{\mathbf{x}}_\theta(\mathbf{x}^t, t), t)) - \\ &\quad (\zeta_1\sqrt{\bar{\alpha}_t} + \zeta_0)\left(T_\phi(\hat{\mathbf{x}}_\theta(\mathbf{x}^t, t-1), t) - T_\phi(\mathbf{x}, t-1)\right)\Big\|_2^2, \\ \mathcal{L}_{prior} &= D_{KL}\left(q_\phi(\mathbf{x}^T|\mathbf{x}) \| p(\mathbf{x}^T)\right) \\ &= \frac{1}{2}\bar{\alpha}_T \left\|T_\phi(\mathbf{x}, T) - c\right\|_2^2.\end{aligned}$$

The complete derivation is given in Appendix A.3. It is essential to distinguish between the objectives of DDPM and CN-Diff. For DDPM, the objective is to accurately predict only the original data point x. On the other hand, CN-Diff is designed to predict the transformed data point $T_\phi(x, t)$ along with the original data point x. Despite this modification, the optimization process for CN-Diff remains simulation-free, thereby facilitating its efficient training through KL divergence computed on samples at all time steps. The procedures for both training and inference are detailed in Algorithms 1 and 2.

4. Experiments

This section presents time series forecasting experiments using CN-Diff and compares its performance against recent deep neural models across nine popular real-world datasets. We begin by describing the datasets, architectural framework, and hyperparameters. We then present results benchmarked against existing forecasting outcomes. An ablation study follows to assess the unique components of our CN-Diff model.

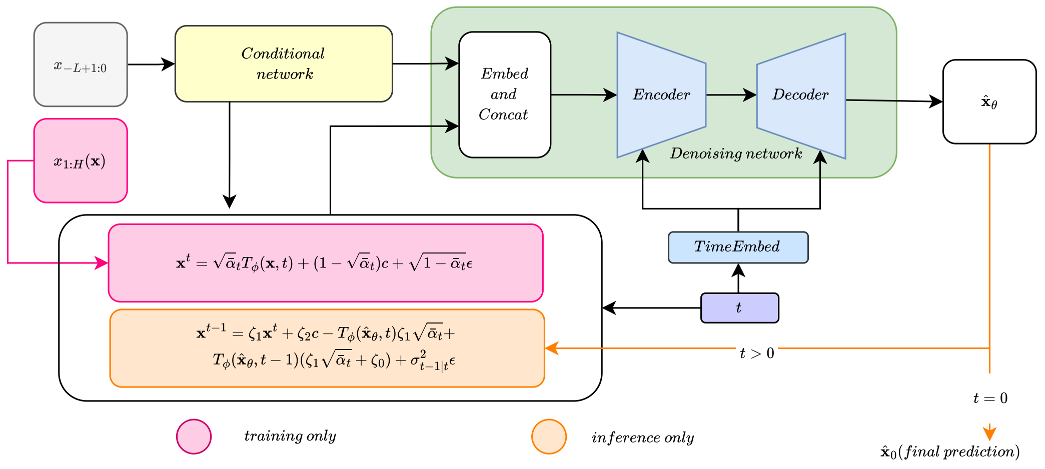
$x_{-L+1:0}$
Conditional network
Embed and Concat
Encoder
Decoder
Denoising network
$\hat{\mathbf{x}}_\theta$
$x_{1:H}(\mathbf{x})$
$\mathbf{x}^t = \sqrt{\bar{\alpha}_t} T_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \bar{\alpha}_t}\epsilon$
$\mathbf{x}^{t-1} = \zeta_1 \mathbf{x}^t + \zeta_2 c - T_\phi(\hat{\mathbf{x}}_\theta, t)\zeta_1\sqrt{\bar{\alpha}_t} + T_\phi(\hat{\mathbf{x}}_\theta, t-1)(\zeta_1\sqrt{\bar{\alpha}_t} + \zeta_0) + \sigma^2_{t-1|t}\epsilon$
TimeEmbed
t
$t > 0$
$t = 0$
training only
inference only
$\hat{\mathbf{x}}_0$(*final prediction*)

Figure 2. An illustration of the CN-Diff architecture. During the training phase, historical data ($x_{-L+1:0}$) is processed through a conditional network, concatenated with noised data, and subsequently passed through the denoising network. During the inference phase, the model initiates from $\mathcal{N}(c, I)$ and performs denoising to obtain ($\hat{x}_0$).

4.1. Experimental Setup

4.1.1. Dataset

We performed an experimental study on nine real-world time series datasets for daily, weekly, and monthly forecasts (Shen et al., 2024; Fan et al., 2022; Zhou et al., 2021; Wu et al., 2021). These datasets are *(1)* Wind[1] - wind power data for 2020-2021 every 15 minutes; *(2)* Caiso[2] - hourly electricity loads over eight years from various California regions; *(3)* Traffic[3] - hourly road occupancy rates from San Francisco Bay area sensors; *(4)* Electricity[4]- hourly electricity use of 321 clients over two years; *(5)* Weather[5] - 10-minute meteorological data for 2020-2021; *(6)* NorPool[6] - eight years of hourly energy production data from various European countries; *(7)* Exchange[7] - daily exchange rates for eight countries: Australia, United Kingdom, Canada, Switzerland, China, Japan, New Zealand and Singapore (Lai et al., 2018); *(8)* ETTh1 and *(9)* ETTm1[8] are benchmarks for China's electricity transformer temperature data, over two years, ETTh1 reported hourly and ETTm1 every 15 minutes (Zhou et al., 2021). Additional details regarding the dataset's characteristics are available in Table 2.

Following (Shen et al., 2024), we adopt a train:validation:test ratio of 7:1:2 for Exchange, Weather, Wind, Traffic and Electricity data sets, and 6:2:2 for ETTh1 and Ettm1 data sets. For Norpool, the training data consists of observations before April 1, 2020; validation data spans from April 1 to October 1, 2020; and testing data is after October 1, 2020. For Caiso, the training set includes data before January 1, 2020, the validation period extends from January 1 to October 1, 2020, and the test set is post-October 1, 2020. Given that data sets exhibit different sampling intervals, we focus on prediction tasks with suitable prediction lengths based on the characteristics of the dataset (Shen & Kwok, 2023; Shen et al., 2024).

4.1.2. Implementation details

Our CN-Diff model is trained using the Adam optimizer with a learning rate of 10^{-3}. The training employs a batch size of 64 and incorporates early stopping (maximum of 100 epochs). We use $T = 100$ diffusion steps with a quadratic variance schedule beginning at $\beta_1 = 10^{-4}$ and progressing to $\beta_T = 10^{-1}$. The length of the look-back window is selected from the set {96, 192, 336, 720, 1440}, determined by performance evaluations on the validation dataset averaged over ten runs. All experiments are executed on a single Nvidia RTX A6000 GPU with 48GB of vRAM. Figure 2 shows the architecture employed for the CN-Diff model. For condition c, a single dense layer is used. For T_ϕ, two dense layers with tanh activation are used, one layer handling the feature dimension and the other handling the

[1] https://github.com/PaddlePaddle/PaddleSpatial/tree/main/paddlespatial/datasets/WindPower

[2] https://www.energyonline.com/Data/

[3] https://pems.dot.ca.gov/

[4] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

[5] https://www.bgc-jena.mpg.de/wetter/

[6] https://data.nordpoolgroup.com/power-system/production/

[7] https://github.com/laiguokun/multivariate-time-series-data

[8] https://github.com/zhouhaoyi/ETDataset

Table 1. Multivariate prediction of MSEs on nine real-world time series datasets(subscripts is the rank). CSDI runs out of memory on Traffic and Electricity. Results of all baselines are from (Shen et al., 2024)

Method	NorPool	Caiso	Traffic	Electricity	Weather	Exchange	ETTh1	ETTm1	Wind	Rank
Ours	$\mathbf{0.531}_{(1)}$	$\mathbf{0.094}_{(1)}$	$\underline{0.374}_{(2)}$	$\mathbf{0.145}_{(1)}$	$\mathbf{0.296}_{(1)}$	$\mathbf{0.016}_{(1)}$	$\mathbf{0.405}_{(1)}$	$\underline{0.340}_{(2)}$	$0.988_{(5)}$	1.7
mr-Diff	$0.645_{(3)}$	$0.127_{(4)}$	$0.474_{(7)}$	$0.155_{(4)}$	$\mathbf{0.296}_{(1)}$	$\mathbf{0.016}_{(1)}$	$0.411_{(4)}$	$\underline{0.340}_{(2)}$	$\mathbf{0.881}_{(1)}$	3.0
TimeDiff	$0.665_{(5)}$	$0.136_{(7)}$	$0.564_{(8)}$	$0.193_{(6)}$	$\underline{0.311}_{(3)}$	$0.018_{(7)}$	$\underline{0.407}_{(2)}$	$\mathbf{0.336}_{(1)}$	$\underline{0.896}_{(2)}$	4.6
TimeGrad	$1.152_{(21)}$	$0.258_{(20)}$	$1.745_{(23)}$	$0.736_{(22)}$	$0.392_{(15)}$	$0.079_{(21)}$	$0.993_{(23)}$	$0.874_{(22)}$	$1.209_{(22)}$	21.0
CSDI	$1.011_{(20)}$	$0.253_{(19)}$	-	-	$0.356_{(10)}$	$0.077_{(20)}$	$0.497_{(8)}$	$0.529_{(18)}$	$1.066_{(10)}$	15.0
SSSD	$0.872_{(13)}$	$0.195_{(11)}$	$0.642_{(12)}$	$0.255_{(13)}$	$0.349_{(9)}$	$0.061_{(17)}$	$0.726_{(19)}$	$0.464_{(14)}$	$1.188_{(20)}$	14.2
D^3VAE	$0.745_{(10)}$	$0.241_{(18)}$	$0.928_{(18)}$	$0.286_{(16)}$	$0.375_{(12)}$	$0.200_{(23)}$	$0.504_{(10)}$	$0.362_{(9)}$	$1.118_{(16)}$	14.7
CPF	$1.613_{(24)}$	$0.383_{(22)}$	$1.625_{(22)}$	$0.793_{(23)}$	$1.390_{(24)}$	$\mathbf{0.016}_{(1)}$	$0.730_{(20)}$	$0.482_{(16)}$	$1.140_{(18)}$	18.9
PSA-GAN	$1.501_{(23)}$	$0.510_{(24)}$	$1.614_{(21)}$	$0.535_{(21)}$	$1.220_{(22)}$	$0.018_{(7)}$	$0.623_{(18)}$	$0.537_{(19)}$	$1.127_{(17)}$	19.1
N-Hits	$0.716_{(8)}$	$0.131_{(5)}$	$0.386_{(3)}$	$0.152_{(3)}$	$0.323_{(5)}$	$\underline{0.017}_{(6)}$	$0.498_{(9)}$	$0.353_{(7)}$	$1.033_{(7)}$	5.9
FiLM	$0.723_{(9)}$	$0.179_{(9)}$	$0.628_{(11)}$	$0.210_{(9)}$	$0.327_{(6)}$	$\mathbf{0.016}_{(1)}$	$0.426_{(6)}$	$0.347_{(5)}$	$0.984_{(4)}$	6.7
Depts	$0.662_{(4)}$	$0.106_{(3)}$	$1.019_{(20)}$	$0.319_{(18)}$	$0.761_{(20)}$	$0.020_{(10)}$	$0.579_{(14)}$	$0.380_{(11)}$	$1.082_{(13)}$	12.6
NBeats	$0.832_{(11)}$	$0.141_{(8)}$	$\mathbf{0.373}_{(1)}$	$0.269_{(14)}$	$1.344_{(23)}$	$\mathbf{0.016}_{(1)}$	$0.586_{(16)}$	$0.391_{(12)}$	$1.069_{(11)}$	10.8
Scaleformer	$0.983_{(16)}$	$0.207_{(14)}$	$0.618_{(10)}$	$0.195_{(7)}$	$0.462_{(17)}$	$0.036_{(13)}$	$0.613_{(17)}$	$0.481_{(15)}$	$1.359_{(23)}$	14.7
PatchTST	$0.851_{(12)}$	$0.193_{(10)}$	$0.831_{(17)}$	$0.225_{(11)}$	$0.782_{(21)}$	$0.047_{(15)}$	$0.526_{(12)}$	$0.372_{(10)}$	$1.077_{(12)}$	13.3
FedFormer	$0.873_{(14)}$	$0.205_{(12)}$	$0.591_{(9)}$	$0.238_{(12)}$	$0.342_{(8)}$	$0.133_{(22)}$	$0.541_{(13)}$	$0.426_{(13)}$	$1.113_{(15)}$	13.1
Autoformer	$0.940_{(15)}$	$0.226_{(16)}$	$0.688_{(16)}$	$0.201_{(8)}$	$0.360_{(11)}$	$0.056_{(16)}$	$0.516_{(11)}$	$0.565_{(20)}$	$1.083_{(14)}$	14.1
Pyraformer	$1.008_{(19)}$	$0.273_{(21)}$	$0.659_{(13)}$	$0.273_{(15)}$	$0.394_{(16)}$	$0.032_{(12)}$	$0.579_{(14)}$	$0.493_{(17)}$	$1.061_{(9)}$	15.1
Informer	$0.985_{(17)}$	$0.251_{(17)}$	$0.664_{(14)}$	$0.298_{(17)}$	$0.385_{(13)}$	$0.073_{(19)}$	$0.775_{(22)}$	$0.673_{(21)}$	$1.168_{(19)}$	17.7
Transformer	$1.005_{(18)}$	$0.206_{(13)}$	$0.671_{(15)}$	$0.328_{(19)}$	$0.388_{(14)}$	$0.062_{(18)}$	$0.759_{(21)}$	$0.992_{(23)}$	$1.201_{(21)}$	18.0
SCINet	$\underline{0.613}_{(2)}$	$\underline{0.095}_{(2)}$	$0.434_{(6)}$	$0.171_{(5)}$	$0.329_{(7)}$	$0.036_{(13)}$	$0.465_{(7)}$	$0.359_{(8)}$	$1.055_{(8)}$	6.4
NLinear	$0.707_{(7)}$	$0.135_{(6)}$	$0.430_{(5)}$	$\underline{0.147}_{(2)}$	$0.313_{(4)}$	$0.019_{(9)}$	$0.410_{(3)}$	$0.349_{(6)}$	$0.989_{(5)}$	5.2
DLinear	$0.670_{(6)}$	$0.461_{(23)}$	$0.389_{(4)}$	$0.215_{(10)}$	$0.488_{(18)}$	$0.022_{(11)}$	$0.415_{(5)}$	$0.345_{(4)}$	$0.899_{(3)}$	9.3
LSTMa	$1.481_{(22)}$	$0.217_{(15)}$	$0.966_{(19)}$	$0.414_{(20)}$	$0.662_{(19)}$	$0.403_{(24)}$	$1.149_{(24)}$	$1.030_{(24)}$	$1.464_{(24)}$	21.2

Table 2. Dataset characteristics

dataset	dim	#observations	freq.	H (steps)
NorPool	18	70,128	1 hour	1 month (720)
Caiso	10	74,472	1 hour	1 month (720)
Weather	21	52,696	10 mins	1 week (672)
ETTm1	7	69,680	15 mins	2 days (192)
Wind	7	48,673	15 mins	2 days (192)
Traffic	862	17,544	1 hour	1 week (168)
Electricity	321	26,304	1 hour	1 week (168)
ETTh1	7	17,420	1 hour	1 week (168)
Exchange	8	7,588	1 day	2 weeks (14)

forecast window dimension. More details of the code and hyperparameters are in the Appendix B.

4.2. Main Results

Table 1 presents the mean squared errors (MSEs) in various multivariate time series datasets. The data reveal that the proposed CN-Diff method achieves superior performance in 6 out of 9 datasets. In particular, the enhancement is particularly pronounced in more complex datasets, namely Norpool and ETTh1. In the remaining three datasets, CN-Diff has Rank 2 in two cases, but only with a very narrow margin with respect to Rank 1 for those data sets. For Wind, CN-Diff achieves the fourth rank. The wind data set represents the highly volatile wind power data. The superior performance of the multiresolution diffusion (mrDiff (rank 1 for wind)) model suggests that wind patterns need to be analyzed at multiple time scales simultaneously. We believe that incorporating CN-Diff at multiple resolutions to build multiresolution CN-Diff could help us to model such datasets better.

Overall, the average ranking of the CN-Diff model is 1.7 showing its state-of-the-art performance. Section 5 has brief descriptions of the other leading models.

Table 3 displays the mean squared errors (MSEs) for multiple univariate time series datasets. The findings indicate that the CN-Diff excels in four of nine datasets. In the other five datasets, it secures second place in two instances. Overall, the average ranking of the CN-Diff surpasses all other leading models. Additional results for the mean absolute error (MAEs) are provided in Appendix D. Appendix F provides a qualitative comparison with various models and includes visualizations for different data sets. The visualizations show that CN-Diff is capable of capturing seasonal patterns without explicitly being trained to do so.

4.3. Qualitative analysis

Figure 3 illustrates the prediction results on ETTh1 generated by CN-Diff alongside three competitive models: Informer, DLinear, and FiLM. It is evident that CN-Diff yields predictions of superior quality compared to the other models.

Table 3. Univariate prediction of MSEs on nine real-world time series datasets (subscript is rank). Results of all other models are from (Shen et al., 2024).

Method	NorPool	Caiso	Traffic	Electricity	Weather	Exchange	ETTh1	ETTm1	Wind	Rank
Ours	$\underline{0.606}_{(2)}$	$\underline{0.118}_{(2)}$	$0.126_{(4)}$	$0.234_{(3)}$	$\mathbf{0.002}_{(1)}$	$\mathbf{0.015}_{(1)}$	$\mathbf{0.066}_{(1)}$	$\mathbf{0.038}_{(1)}$	$2.171_{(3)}$	2.0
mr-Diff	$0.667_{(5)}$	$0.122_{(4)}$	$\mathbf{0.119}_{(1)}$	$0.234_{(3)}$	$\mathbf{0.002}_{(1)}$	$\underline{0.016}_{(2)}$	$\mathbf{0.066}_{(1)}$	$0.039_{(3)}$	$2.182_{(5)}$	2.8
TimeDiff	$0.636_{(3)}$	$0.122_{(4)}$	$\underline{0.121}_{(2)}$	$\underline{0.232}_{(2)}$	$\mathbf{0.002}_{(1)}$	$0.017_{(5)}$	$\mathbf{0.066}_{(1)}$	$0.040_{(6)}$	$2.407_{(14)}$	4.2
TimeGrad	$1.129_{(23)}$	$0.325_{(23)}$	$1.223_{(24)}$	$0.920_{(24)}$	$\mathbf{0.002}_{(1)}$	$0.041_{(21)}$	$0.078_{(11)}$	$0.048_{(12)}$	$2.530_{(19)}$	17.6
CSDI	$0.967_{(22)}$	$0.192_{(15)}$	$0.393_{(21)}$	$0.520_{(19)}$	$\mathbf{0.002}_{(1)}$	$0.071_{(24)}$	$0.083_{(16)}$	$0.050_{(16)}$	$2.434_{(16)}$	16.7
SSSD	$1.145_{(24)}$	$0.176_{(13)}$	$0.151_{(9)}$	$0.370_{(12)}$	$0.004_{(12)}$	$0.023_{(17)}$	$0.097_{(21)}$	$0.049_{(14)}$	$3.149_{(23)}$	16.1
D^{3}VAE	$0.964_{(21)}$	$0.521_{(24)}$	$0.151_{(9)}$	$0.535_{(20)}$	$0.003_{(10)}$	$0.019_{(13)}$	$0.078_{(11)}$	$0.044_{(10)}$	$2.679_{(21)}$	15.4
CPF	$0.855_{(16)}$	$0.260_{(22)}$	$0.279_{(19)}$	$0.609_{(22)}$	$\mathbf{0.002}_{(1)}$	$\underline{0.016}_{(2)}$	$0.080_{(14)}$	$0.041_{(7)}$	$2.430_{(15)}$	13.1
PSA-GAN	$0.658_{(4)}$	$0.150_{(8)}$	$0.250_{(18)}$	$0.273_{(8)}$	$0.035_{(22)}$	$0.020_{(14)}$	$0.084_{(17)}$	$0.051_{(17)}$	$2.510_{(18)}$	14.0
N-Hits	$0.739_{(10)}$	$0.170_{(12)}$	$0.147_{(8)}$	$0.346_{(9)}$	$\mathbf{0.002}_{(1)}$	$0.017_{(5)}$	$0.089_{(18)}$	$0.043_{(9)}$	$2.406_{(13)}$	9.4
FiLM	$0.707_{(8)}$	$0.185_{(14)}$	$0.198_{(14)}$	$0.260_{(6)}$	$0.007_{(15)}$	$0.018_{(10)}$	$0.070_{(4)}$	$\mathbf{0.038}_{(1)}$	$\underline{2.143}_{(2)}$	8.2
Depts	$0.668_{(6)}$	$\mathbf{0.107}_{(1)}$	$0.151_{(9)}$	$0.380_{(15)}$	$0.024_{(20)}$	$0.020_{(14)}$	$0.070_{(4)}$	$0.046_{(11)}$	$3.457_{(24)}$	11.6
NBeats	$0.768_{(12)}$	$0.125_{(6)}$	$0.142_{(7)}$	$0.378_{(14)}$	$0.137_{(23)}$	$\underline{0.016}_{(2)}$	$0.095_{(20)}$	$0.048_{(12)}$	$2.434_{(16)}$	12.4
Scaleformer	$0.778_{(13)}$	$0.232_{(17)}$	$0.286_{(20)}$	$0.361_{(10)}$	$0.009_{(18)}$	$0.035_{(20)}$	$0.150_{(23)}$	$0.078_{(23)}$	$2.646_{(20)}$	18.2
PatchTST	$\mathbf{0.595}_{(1)}$	$0.193_{(16)}$	$0.177_{(13)}$	$0.450_{(18)}$	$0.026_{(21)}$	$0.020_{(14)}$	$0.106_{(22)}$	$0.052_{(19)}$	$2.698_{(22)}$	16.2
FedFormer	$0.891_{(17)}$	$0.164_{(10)}$	$0.173_{(12)}$	$0.376_{(13)}$	$0.005_{(13)}$	$0.050_{(23)}$	$0.076_{(8)}$	$0.065_{(22)}$	$2.351_{(12)}$	14.4
Autoformer	$0.946_{(20)}$	$0.248_{(18)}$	$0.473_{(22)}$	$0.659_{(23)}$	$0.003_{(10)}$	$0.041_{(21)}$	$0.081_{(15)}$	$0.051_{(17)}$	$2.349_{(11)}$	17.4
Pyraformer	$0.933_{(19)}$	$0.165_{(11)}$	$0.136_{(5)}$	$0.389_{(16)}$	$0.020_{(19)}$	$0.017_{(8)}$	$0.076_{(8)}$	$0.054_{(20)}$	$2.279_{(7)}$	12.6
Informer	$0.804_{(14)}$	$0.250_{(19)}$	$0.213_{(16)}$	$0.363_{(11)}$	$0.007_{(15)}$	$0.023_{(17)}$	$0.076_{(8)}$	$0.049_{(14)}$	$2.297_{(8)}$	13.6
Transformer	$0.928_{(18)}$	$0.250_{(19)}$	$0.238_{(17)}$	$0.430_{(15)}$	$0.007_{(15)}$	$0.018_{(10)}$	$0.092_{(19)}$	$0.058_{(21)}$	$2.306_{(10)}$	16.0
SCINet	$0.746_{(11)}$	$0.154_{(9)}$	$0.212_{(15)}$	$0.272_{(7)}$	$\mathbf{0.002}_{(1)}$	$0.018_{(10)}$	$0.071_{(7)}$	$0.039_{(3)}$	$\mathbf{2.063}_{(1)}$	7.1
NLinear	$0.708_{(9)}$	$0.147_{(7)}$	$0.124_{(3)}$	$\mathbf{0.231}_{(1)}$	$\mathbf{0.002}_{(1)}$	$0.017_{(5)}$	$0.070_{(4)}$	$0.039_{(3)}$	$2.193_{(6)}$	4.3
DLinear	$0.671_{(7)}$	$\underline{0.118}_{(2)}$	$0.139_{(6)}$	$0.244_{(5)}$	$0.168_{(24)}$	$0.017_{(5)}$	$0.078_{(11)}$	$0.041_{(7)}$	$2.171_{(3)}$	7.8
LSTMa	$0.836_{(15)}$	$0.253_{(21)}$	$1.032_{(23)}$	$0.596_{(21)}$	$0.005_{(13)}$	$0.031_{(19)}$	$0.167_{(24)}$	$0.091_{(24)}$	$2.299_{(9)}$	18.8

4.4. Ablation Study

To evaluate the individual effects of each component of the CN-Diff model, we conducted various experiments. Initially, we illustrate the impact of the introduction of the condition c (Eqn.1), followed by an examination of the effects of the nonlinear transformation.

In the condition ablation study (Table 4), we present the forecast performance in terms of MSE and MAE when using only the diffusion model (DDPM (Ho et al., 2020)) and when applying the condition only in the backward process, which represents the standard conditional diffusion setting for time series analysis). These numbers are compared with the results with our CN-Diff conditional method, where the condition is used in both forward and backward processes. The MSE and MAE improve by 8% and 6% for ETTh1 (feature dimension of 7), 16% and 9% for electricity (feature dimension of 321) and 21% and 16% for traffic (feature dimension of 862). This clearly demonstrates that with an increase in the feature dimension, the effect of our conditioning is enhanced while maintaining the same forecast window. We can infer that, as the feature dimension increases, learning from an isotropic Gaussian prior becomes considerably more challenging and CN-Diff solves this challenge.

For the ablation study on the time-dependent nonlinear data transformation T_ϕ, we consider four variations (Table 5): *(i)* CN-Diff without introducing any nonlinear transformation; nonlinear transformations *(ii)* only along the feature dimension, *(iii)* only along the look-back window dimension, and *(iv)* along both dimensions. From variation *(i)* to *(ii)*, the MSE and MAE improved by 54% and 36% for ETTh1, 72% and 52% for Electricity, 57% and 35% for Caiso and 51% and 30% for Wind. From *(i)* to *(iii)*, the improvements are similar at 54% and 38% for ETTh1, 70% and 49% for Electricity, 25% and 8% for Caiso and 48% and 27% for Wind. From *(i)* to *(iv)*, an additional improvement is observed at 57% and 40% for ETTh1, 73% and 53% for Electricity, 61% and 38% for Caiso and 53% and 31% for Wind. Clearly, the incorporation of nonlinear data transformation in both dimensions effectively captures the significant correlations present within both the sequence and feature dimensions of time series data.

We present a joint ablation study in Appendix E.1, examining the contributions due to variations of nonlinear transformation and CN-Diff condition. Additional ablation studies for different diffusion parameters are found in Appendix E.3, and for various look-back windows in Appendix E.2. The results indicate a notable enhancement with each component added to the diffusion model. Optimal performance with varying diffusion parameters is attained at diffusion

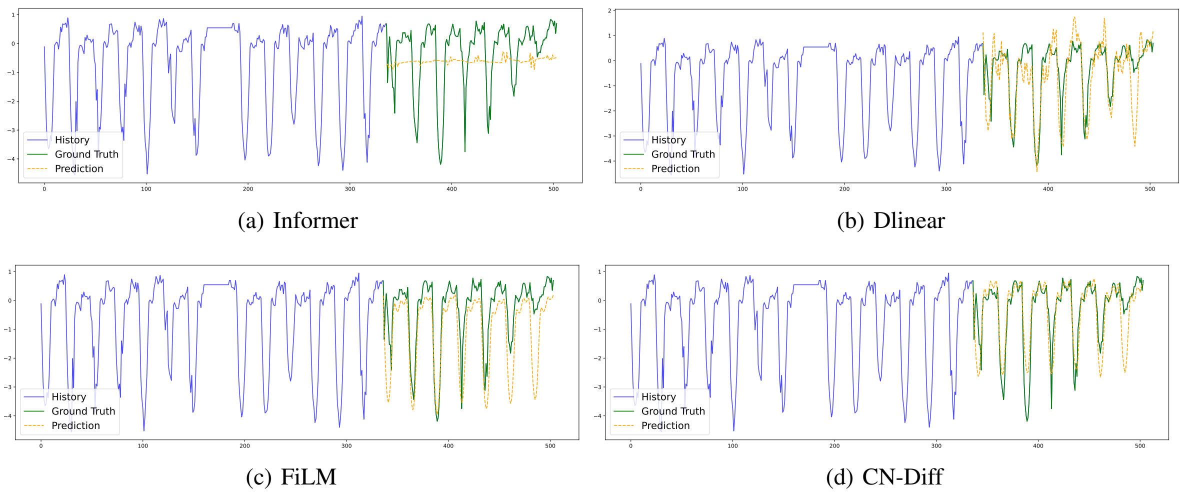

Figure 3. Visualizations of ETTh1 dataset predictions by (a) Informer(Zhou et al., 2021) (b) Dlinear(Zeng et al., 2023) (c) FiLM(Zhou et al., 2022a) and (d) CN-Diff

Table 4. Predicting MSEs and MAEs for DDPM, conditional DDPM and CN-Diff condition

Methods	Diff		Cond Diff		CN-Diff Cond	
Datasets	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	1.748	0.927	1.030	0.746	**0.943**	**0.704**
Electricity	1.784	1.018	0.637	0.564	**0.536**	**0.512**
Traffic	1.794	0.885	1.071	0.573	**0.848**	**0.479**

noise steps $T = 100$ and an upper variance schedule of $\beta_T = 0.1$.

5. Related work

Time series forecasting is utilized in numerous aspects of daily life, ranging from essential applications like traffic and electricity monitoring, and healthcare (Liu et al., 2022b), to more intricate uses such as predicting stock prices, weather, and wind patterns (Shen & Kwok, 2023). Due to its wide significance, research has advanced from traditional state space models to contemporary deep learning methodologies.

Basis expansion has been utilized by several models for time-series forecasting. FiLM (Zhou et al., 2022a) employed Fourier analysis and low-rank matrix approximation to reduce noise, along with Legendre polynomial projection to maintain historical data representations. NBeats (Oreshkin et al., 2019) is an interpretable architecture that combines polynomial trend modeling with Fourier techniques for detecting seasonality. Depts (Fan et al., 2022) builds on NBeats by adding a periodicity module for periodic series, while N-Hits (Challu et al., 2023) enhances the approach using multi-scale hierarchical interpolation.

Models such as SCINet (Liu et al., 2022a) utilize a recursive strategy involving downsampling, convolution, and interaction, to harness temporal dependencies in downsampled subsequences. NLinear (Zeng et al., 2023) implements a basic approach by normalizing the time series before using a linear layer for prediction. DLinear (Zeng et al., 2023) employs a seasonal-trend decomposition akin to Autoformer (Wu et al., 2021).

Several models utilized the transformer (Vaswani, 2017b) and its variations for time series prediction. Informer (Zhou et al., 2021) used sparse attention to minimize computational load and employed a generative-style decoder for rapid long-sequence forecasts in one forward pass. Autoformer (Wu et al., 2021) substituted traditional self-attention with an autocorrelation layer, while Fedformer (Zhou et al., 2022b) incorporated frequency domain mapping through a frequency-enhanced module. Pyraformer (Liu et al., 2022b) introduced pyramidal attention for multi-resolution representation, and Scaleformer (Shabani et al., 2023) adopted a shared-weight multilevel forecasting approach, from broad to fine scales. Inspired by vision transformers (Dosovitskiy et al., 2020), PatchTST (Nie et al., 2023) partitions time series data into subseries patches and uses self-supervised pre-training to extract local semantic features, thus improving long-term prediction.

Table 5. Predicting MSEs and MAEs by applying Nonlinear Transformations across different configurations

Methods	Without T_ϕ		Feature dim		Forecast window		Both	
Datasets	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Electricity	0.536	0.512	0.150	0.246	0.160	0.260	**0.145**	**0.243**
ETTh1	0.943	0.704	0.429	0.447	0.430	0.439	**0.405**	**0.421**
Caiso	0.242	0.307	0.104	0.200	0.182	0.283	**0.094**	**0.191**
Wind	2.090	1.023	1.015	0.715	1.067	0.743	**0.988**	**0.701**

Recently, diffusion models have emerged as a promising method for time series forecasting. TimeGrad (Rasul et al., 2021) pioneered the use of conditional diffusion models, leveraging autoregressive prediction informed by RNN hidden states. CSDI (Tashiro et al., 2021), applies non-autoregressive generation through self-supervised masking but relies on dual transformers, facing boundary inconsistencies and computational challenges with large datasets. SSSD (Lopez Alcaraz & Strodthoff, 2023) aims to reduce CSDI's computational load using structured state-space models, but it continues with masking-based conditioning, preserving boundary issues. TimeDiff (Shen & Kwok, 2023) introduced future mixup and autoregressive initialization within an encoder-decoder scheme for denoising. TMDM (Li et al., 2024b) integrates transformers with a diffusion process for probabilistic multivariate time series forecasting. TimeDiT (Cao et al., 2024) is a foundational model for time series, which employed a transformer type architecture to capture temporal dependencies and employs diffusion processes to generate samples. mr-Diff(Shen et al., 2024) is a recent study utilizing a multi-resolution strategy, incorporating fine-to-coarse patterns as latent variables to aid in denoising.

One step diffusion (Frans et al., 2024), a family of generative models that use a single network and training phase to produce high-quality samples in a single or multiple sampling steps in image diffusion. NDM(Bartosh et al., 2024) applies a non-linear time-dependent transformation for image diffusion models with the Markovian reverse process. Polydiffuse (Chen et al., 2023) uses a guided diffusion model for polygonal shape reconstruction, where the prior is learned via guidance networks $\mu(x, t, i)$ and $\sigma(x, t, i)$, which are independent of the condition. Also they have not incorporated the condition in the forward process (ref. Fig. 3 [(Chen et al., 2023)]). In contrast, our approach incorporates a condition in forward formulation results in condition-dependent prior. And our training procedure and loss function differs significantly, as polydiffuse uses separate stages for prior learning and denoising. PriorGrad (Lee et al., 2021) is diffusion-based generative model for speech synthesis, which introduces a prior distribution based on data statistics rather than a learned prior. While their reverse process starts by sampling from $\mathcal{N}(0, \Sigma)$, our method samples from $\mathcal{N}(c, \Sigma)$ as we include a learnable condition in the forward process. These fundamental differences set our work apart from the above.

6. Conclusion and Future work

This paper presents CN-Diff, a new framework for time series forecasting that combines non-linear data transformation with conditional information in the forward process. We have derived the variational loss for the diffusion model from this framework. Our experiments indicate that CN-Diff matches or surpasses the performance of top time series models across nine real-world datasets. The avenues of future work include adding an explicit seasonal trend diffusion model, exploring different conditions using varied resolutions, and metadata-based conditional generation. We also plan to extend our model for imputation, telemetry-based anomaly detection, and other engineering applications.

Acknowledgment

This research is partially supported by a research grant (MoES/36/OOIS/Extra/84/2022) from the Ministry of Earth Sciences, Government of India. We are also grateful for the TCS PhD Fellowhip to J Rishi.

Impact Statement

This article presents new research exploring a novel diffusion-based framework for modeling time series data. The proposed model exhibits excellent performance when compared to existing diffusion-based time series models for forecasting tasks. Our model shows great promise for improving forecast accuracy in diverse applications such as environmental, finance, climate, and industrial domains. Our research is centered on forecasting time series data, which on its own does not have any major ethical concerns. The downstream application developers must exercise caution to ensure that no ethical concerns arise when using our model for societal applications.

References

Bahdanau, D., Cho, K. H., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

Bartosh, G., Vetrov, D., and Naesseth, C. A. Neural diffusion models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 3073–3095, 2024.

Benny, Y. and Wolf, L. Dynamic dual-output diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11482–11491, 2022.

Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., Kim, S. W., Fidler, S., and Kreis, K. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22563–22575, 2023.

Cao, D., Ye, W., and Liu, Y. Timedit: General-purpose diffusion transformers for time series foundation model. In *ICML 2024 Workshop on Foundation Models in the Wild*, 2024.

Challu, C., Olivares, K. G., Oreshkin, B. N., Ramirez, F. G., Canseco, M. M., and Dubrawski, A. Nhits: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 6989–6997, 2023.

Chen, J., Deng, R., and Furukawa, Y. Polydiffuse: Polygonal shape reconstruction via guided set diffusion models. *Advances in Neural Information Processing Systems*, 36: 1863–1888, 2023.

Das, A., Kong, W., Leach, A., Mathur, S., Sen, R., and Yu, R. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023a.

Das, A., Kong, W., Sen, R., and Zhou, Y. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*, 2023b.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Dumas, J., Wehenkel, A., Lanaspeze, D., Cornélusse, B., and Sutera, A. A deep generative model for probabilistic energy forecasting in power systems: normalizing flows. *Applied Energy*, 305:117871, 2022.

Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.

Fan, W., Zheng, S., Yi, X., Cao, W., Fu, Y., Bian, J., and Liu, T.-Y. Depts: Deep expansion learning for periodic time series forecasting. In *International Conference on Learning Representations*, 2022.

Feng, S., Miao, C., Zhang, Z., and Zhao, P. Latent diffusion transformer for probabilistic time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11979–11987, 2024.

Frans, K., Hafner, D., Levine, S., and Abbeel, P. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.

Ge, S., Nah, S., Liu, G., Poon, T., Tao, A., Catanzaro, B., Jacobs, D., Huang, J.-B., Liu, M.-Y., and Balaji, Y. Preserve your own correlation: A noise prior for video diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22930–22941, 2023.

Harvey, W., Naderiparizi, S., Masrani, V., Weilbach, C., and Wood, F. Flexible diffusion modeling of long videos. *Advances in Neural Information Processing Systems*, 35: 27953–27965, 2022.

Henrique, B. M., Sobreiro, V. A., and Kimura, H. Literature review: Machine learning techniques applied to financial market prediction. *Expert Systems with Applications*, 124: 226–251, 2019.

Hewamalage, H., Bergmeir, C., and Bandara, K. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1):388–427, 2021.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Huang, X., Wu, D., and Boulet, B. Metaprobformer for charging load probabilistic forecasting of electric vehicle charging stations. *IEEE Transactions on Intelligent Transportation Systems*, 24(10):10445–10455, 2023.

Jeha, P., Bohlke-Schneider, M., Mercado, P., Kapoor, S., Nirwan, R. S., Flunkert, V., Gasthaus, J., and Januschowski, T. Psa-gan: Progressive self attention gans for synthetic time series. In *The Tenth International Conference on Learning Representations*, 2022.

Jia, Y., Anaissi, A., and Suleiman, B. Resnls: An improved model for stock price forecasting. *Computational Intelligence*, 40(1):e12608, 2024.

Kim, K.-j. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

Kingma, D. P. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013a.

Kingma, D. P. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013b.

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

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lee, S.-g., Kim, H., Shin, C., Tan, X., Liu, C., Meng, Q., Qin, T., Chen, W., Yoon, S., and Liu, T.-Y. Priorgrad: Improving conditional denoising diffusion models with data-dependent adaptive prior. *arXiv preprint arXiv:2106.06406*, 2021.

Li, L., Carver, R., Lopez-Gomez, I., Sha, F., and Anderson, J. Generative emulation of weather forecast ensembles with diffusion models. *Science Advances*, 10(13):eadk4489, 2024a.

Li, Y., Lu, X., Wang, Y., and Dou, D. Generative time series forecasting with diffusion, denoise, and disentanglement. *Advances in Neural Information Processing Systems*, 35: 23009–23022, 2022a.

Li, Y., Lu, X., Wang, Y., and Dou, D. Generative time series forecasting with diffusion, denoise, and disentanglement. *Advances in Neural Information Processing Systems*, 35: 23009–23022, 2022b.

Li, Y., Chen, W., Hu, X., Chen, B., Zhou, M., et al. Transformer-modulated diffusion models for probabilistic multivariate time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024b.

Liu, M., Zeng, A., Chen, M., Xu, Z., Lai, Q., Ma, L., and Xu, Q. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022a.

Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., and Dustdar, S. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *# PLACEHOLDER_PARENT_METADATA_VALUE#*, 2022b.

Liu, Y., Guo, H., Zhang, L., Liang, D., Zhu, Q., Liu, X., Lv, Z., Dou, X., and Gou, Y. Research on correlation analysis method of time series features based on dynamic time warping algorithm. *IEEE Geoscience and Remote Sensing Letters*, 20:1–5, 2023.

Lopez Alcaraz, J. M. and Strodthoff, N. Diffusion-based time series imputation and forecasting with structured atate apace models. *Transactions on machine learning research*, pp. 1–36, 2023.

Mothish, G., Tayal, M., and Kolathaya, S. Birodiff: Diffusion policies for bipedal robot locomotion on unseen terrains. In *2024 Tenth Indian Control Conference (ICC)*, pp. 385–390. IEEE, 2024.

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. N-beats: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*, 2019.

Pandey, K., Mukherjee, A., Rai, P., and Kumar, A. Diffusevae: Efficient, controllable and high-fidelity generation from low-dimensional latents. *Transactions on Machine Learning Research*, 2022.

Peebles, W. and Xie, S. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Rangapuram, S. S., Kapoor, S., Nirwan, R. S., Mercado, P., Januschowski, T., Wang, Y., and Bohlke-Schneider, M. Coherent probabilistic forecasting of temporal hierarchies. In *International Conference on Artificial Intelligence and Statistics*, pp. 9362–9376. PMLR, 2023.

Rasul, K., Seward, C., Schuster, I., and Vollgraf, R. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, pp. 8857–8868. PMLR, 2021.

Rasul, K., Park, Y.-J., Ramström, M. N., and Kim, K.-M. Vq-ar: Vector quantized autoregressive probabilistic time series forecasting. *arXiv preprint arXiv:2205.15894*, 2022.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pp. 1278–1286. PMLR, 2014.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35: 36479–36494, 2022.

Shabani, M. A., Abdi, A. H., Meng, L., and Sylvain, T. Scaleformer: Iterative multi-scale refining transformers for time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023.

Shen, L. and Kwok, J. Non-autoregressive conditional diffusion models for time series prediction. In *International Conference on Machine Learning*, pp. 31016–31029. PMLR, 2023.

Shen, L., Chen, W., and Kwok, J. Multi-resolution diffusion models for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Taga, E. O., Ildiz, M. E., and Oymak, S. Timepfn: Effective multivariate time series forecasting with synthetic data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20761–20769, 2025.

Tashiro, Y., Song, J., Song, Y., and Ermon, S. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.

Vaswani, A. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017a.

Vaswani, A. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017b.

Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

Yue, Z., Wang, Y., Duan, J., Yang, T., Huang, C., Tong, Y., and Xu, B. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8980–8987, 2022.

Zeng, A., Chen, M., Zhang, L., and Xu, Q. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 2023.

Zhang, Z., Meng, L., and Gu, Y. Sageformer: Series-aware framework for long-term multivariate time-series forecasting. *IEEE Internet of Things Journal*, 11(10):18435–18448, 2024.

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

Zhou, T., Ma, Z., Wen, Q., Sun, L., Yao, T., Yin, W., Jin, R., et al. Film: Frequency improved legendre memory model for long-term time series forecasting. *Advances in neural information processing systems*, 35:12677–12690, 2022a.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., and Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pp. 27268–27286. PMLR, 2022b.

A. Formal Derivations

A.1. Forward Posterior

Given:

$$q_\phi(\mathbf{x}^t|\mathbf{x}, c) = \mathcal{N}(\mathbf{x}^t; \sqrt{\bar{\alpha}_t}T_\phi(\mathbf{x}, t) + (1-\sqrt{\bar{\alpha}_t})c, (1-\bar{\alpha}_t)I) \tag{5}$$

From (5), We can write,

$$\begin{aligned}
\mathbf{x}^t &= \sqrt{\bar{\alpha}_t}T_\phi(\mathbf{x}, t) + (1-\sqrt{\bar{\alpha}_t})c + \sqrt{1-\bar{\alpha}_t}\epsilon \\
&= \sqrt{\bar{\alpha}_t}T_\phi(\mathbf{x}, t) + (1-\sqrt{\bar{\alpha}_t})c + \sqrt{1-\alpha_t\bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t}T_\phi(\mathbf{x}, t) + (1-\sqrt{\bar{\alpha}_t})c + \sqrt{1-\alpha_t}\epsilon + \sqrt{\alpha_t - \alpha_t\bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t}T_\phi(\mathbf{x}, t) + (1-\sqrt{\bar{\alpha}_t})c + \sqrt{1-\alpha_t}\epsilon + \sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t}T_\phi(\mathbf{x}, t) + (1-\sqrt{\bar{\alpha}_t})c + \sqrt{1-\alpha_t}\epsilon + \sqrt{\alpha_t}\left(\mathbf{x}^{t-1} - \sqrt{\bar{\alpha}_{t-1}}T_\phi(\mathbf{x}, t-1) - (1-\sqrt{\bar{\alpha}_{t-1}})c\right) \\
&= \sqrt{\alpha_t}\mathbf{x}^{t-1} + \sqrt{\bar{\alpha}_t}\left(T_\phi(\mathbf{x}, t) - T_\phi(\mathbf{x}, t-1)\right) + (1-\sqrt{\alpha_t})c + \sqrt{1-\alpha_t}\epsilon
\end{aligned} \tag{6}$$

By introducing nonlinear, time-dependent transformation of data, along with condition, results in a forward process that is non-Markovian as:

$$q_\phi(\mathbf{x}^t|\mathbf{x}^{t-1}, \mathbf{x}, c) = \mathcal{N}(\mathbf{x}^t; \sqrt{\alpha_t}\mathbf{x}^{t-1} + \sqrt{\bar{\alpha}_t}\left(T_\phi(\mathbf{x}, t) - T_\phi(\mathbf{x}, t-1)\right) + (1-\sqrt{\alpha_t})c, (1-\alpha_t)I) \tag{7}$$

From (7 and 5), Posterior distribution can be derived as:

$$\begin{aligned}
q_\phi\left(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c\right) &\propto \frac{q_\phi(\mathbf{x}^{t-1}, \mathbf{x}^t, \mathbf{x}, c)}{q_\phi(\mathbf{x}^t, \mathbf{x}, c)} \\
&\propto \frac{q_\phi(\mathbf{x}^t|\mathbf{x}^{t-1}, \mathbf{x}, c)}{q_\phi(\mathbf{x}^t, \mathbf{x}, c)} q_\phi(\mathbf{x}^{t-1}, \mathbf{x}, c) \\
&\propto q_\phi(\mathbf{x}^t|\mathbf{x}^{t-1}, \mathbf{x}, c) q_\phi(\mathbf{x}^{t-1}|\mathbf{x}, c)
\end{aligned} \tag{8}$$

$$\begin{aligned}
&\propto \exp\left(-\frac{1}{2}\left(\frac{\left(\mathbf{x}^t - \sqrt{\alpha_t}\mathbf{x}^{t-1} - \sqrt{\bar{\alpha}_t}(T_\phi(\mathbf{x}, t) - T_\phi(\mathbf{x}, t-1)) - (1-\sqrt{\alpha_t})c\right)^2}{1-\alpha_t}\right.\right. \\
&\left.\left. + \frac{\left(\mathbf{x}^{t-1} - \sqrt{\bar{\alpha}}_{t-1}T_\phi(\mathbf{x}, t-1) - (1-\sqrt{\bar{\alpha}_{t-1}})c\right)^2}{1-\bar{\alpha}_{t-1}}\right)\right) \\
&\propto \exp\left(-\frac{1}{2}\left(\left(\frac{\alpha_t}{1-\alpha_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right)(\mathbf{x}^{t-1})^2\right.\right. \\
&- \frac{\left(2\sqrt{\alpha_t}\mathbf{x}^{t-1}(\mathbf{x}^t - \sqrt{\bar{\alpha}}_t(T_\phi(\mathbf{x}, t) - T_\phi(\mathbf{x}, t-1)) - (1-\sqrt{\alpha_t})c\right)}{1-\alpha_t} \\
&\left.\left. - \frac{2\mathbf{x}^{t-1}}{1-\bar{\alpha}_{t-1}}\left(\sqrt{\bar{\alpha}}_{t-1}T_\phi(\mathbf{x}, t-1) + (1-\sqrt{\bar{\alpha}_{t-1}})c\right)\right)\right)
\end{aligned} \tag{9}$$

By reformulating the above propotional form, akin to normal distribution, we can see that,

$$q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c) = \mathcal{N}(\mathbf{x}^{t-1}; \mu_{t-1|t}, \sigma^2_{t-1|t}I) \tag{10}$$

Where,

$$\begin{aligned}
\mu_{t-1|t} = &\left(\frac{\sqrt{\alpha_t}\mathbf{x}^t - \sqrt{\bar{\alpha}}_t\sqrt{\alpha_t}(T_\phi(\mathbf{x}, t) - T_\phi(\mathbf{x}, t-1)) - \sqrt{\alpha_t}(1-\sqrt{\alpha_t})c}{1-\alpha_t} + \right. \\
&\left. \frac{\sqrt{\bar{\alpha}}_{t-1}T_\phi(\mathbf{x}, t-1) + (1-\sqrt{\bar{\alpha}_{t-1}})c}{1-\bar{\alpha}_{t-1}}\right)\sigma^2_{t-1|t}
\end{aligned} \tag{11}$$

$$\sigma^2_{t-1|t} = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \tag{12}$$

Mean($\mu_{t-1|t}$) can be simplified as,

$$\begin{aligned}
\mu_{t-1|t} = \Bigg(& \frac{\sqrt{\alpha_t}\mathbf{x}^t - \sqrt{\bar{\alpha}}_t\sqrt{\alpha_t}(T_\phi(\mathbf{x},t) - T_\phi(\mathbf{x},t-1)) - \sqrt{\alpha_t}(1-\sqrt{\alpha_t})c}{(1-\alpha_t)} \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \\
& + \frac{\sqrt{\bar{\alpha}}_{t-1}T_\phi(\mathbf{x},t-1) + (1-\sqrt{\bar{\alpha}}_{t-1})c}{(1-\bar{\alpha}_{t-1})} \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \Bigg) \\
= & \left(\sqrt{\alpha_t}\mathbf{x}^t - \sqrt{\bar{\alpha}}_t\sqrt{\alpha_t}(T_\phi(\mathbf{x},t) - T_\phi(\mathbf{x},t-1)) - \sqrt{\alpha_t}(1-\sqrt{\alpha_t})c\right) \frac{(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \\
& + \left(\sqrt{\bar{\alpha}}_{t-1}T_\phi(\mathbf{x},t-1) + (1-\sqrt{\bar{\alpha}}_{t-1})c\right) \frac{(1-\alpha_t)}{1-\bar{\alpha}_t}
\end{aligned} \tag{13}$$

$$\mu_{t-1|t} = \zeta_1\mathbf{x}^t + \zeta_2 c - T_\phi(\mathbf{x},t)\zeta_1\sqrt{\bar{\alpha}_t} + T_\phi(\mathbf{x},t-1)(\zeta_1\sqrt{\bar{\alpha}_t} + \zeta_0)$$

Where,

$$\zeta_0 = \frac{1-\alpha_t}{1-\bar{\alpha}_t}\sqrt{\bar{\alpha}_{t-1}}; \qquad \zeta_1 = \frac{(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\sqrt{\alpha_t}; \qquad \zeta_2 = \left(1 + \frac{(\sqrt{\bar{\alpha}_t}-1)(\sqrt{\alpha_t}+\sqrt{\bar{\alpha}_{t-1}})}{1-\bar{\alpha}_t}\right);$$

A.2. Loss Formulation

Although our intention was not to explicitly make the forward process non-Markovian, our formulation has resulted in a non-Markovian forward process Eq.7. So, we can make use of joint distribution from DDIM (Song et al., 2020) as:

$$q(x^{0:T}|x) = q(x^T|x)\prod_{t=1}^{T} q(x^{t-1}|x^t, x),$$

Next, we define Non-Markovian trainable generative process $p_\theta(x_{0:T})$ as:

$$p_\theta(x^{0:T}) = p(x^T)\prod_{t=1}^{T} p_\theta(x^{t-1}|x^t)p_\theta(x^0|x^t), \tag{14}$$

$$\text{where } p(x^T) = \mathcal{N}(x^T; 0, I). \tag{15}$$

This formulation is similar to Equation (55) in DDIM (Song et al., 2020). However, the objective in the referenced work is to minimize the number of sampling time steps; therefore, certain time steps are employed for image generation, and others are included in the variational objective, ultimately demonstrating that the loss formulation aligns with DDPM.

Our definition of trainable reverse generative process (15) is a result of introducing a nonlinear transformation in the marginal distribution, which makes it challenging to learn from the reverse Markovian process. However, even without non-linearity, the resulting loss will be similar to that of DDPM, (Refer Lemma A.1)

Now, by incorporating this into our variational loss term, the variational loss is given as:

$$\begin{aligned}
\mathbb{E}_{q_\phi}\Bigg[& \underbrace{D_{\mathrm{KL}}\left(q_\phi\left(\mathbf{x}^T|\mathbf{x},c\right)\|p\left(\mathbf{x}^T|c\right)\right)}_{\mathcal{L}_{\text{prior}}} \underbrace{- \sum_{t=1}^{T}\log p_\theta\left(\mathbf{x}|\mathbf{x}^t,c\right)}_{\mathcal{L}_{\text{rec}}} \\
& \underbrace{+ \sum_{t=1}^{T} D_{\mathrm{KL}}\left(q_\phi\left(\mathbf{x}^{t-1}|\mathbf{x}^t,\mathbf{x},c\right)\|p_\theta\left(\mathbf{x}^{t-1}|\mathbf{x}^t,c\right)\right)}_{\mathcal{L}_{\text{diff}}} \Bigg].
\end{aligned} \tag{16}$$

Lemma A.1. *For $p_\theta(x^{0:T})$ defined in Eq. 15, variational lower bound results in DDPM training objective.*
Proof. Based on (Ho et al., 2020), the variational objective (J_{x_θ}) can be expressed as follows:

$$\begin{aligned}
J_{x_\theta} &= \mathbb{E}_q\left[\log q(x^{1:T} \mid x_0) - \log p_\theta(x^{0:T})\right] \\
&= \mathbb{E}_q\left[\log q(x^T|x^0) + \sum_{t=1}^{T}\log q(x^{t-1}|x^t,x^0) - \sum_{t=1}^{T}\log p_\theta(x^{t-1}|x^t) - \sum_{t=1}^{T}\log p_\theta(x^0|x^t) - \log p_\theta(x^T)\right] \\
&= \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}\left(q(x^T|x^0)\,\|\,p(x^T)\right)}_{\mathcal{L}_{prior}} + \underbrace{\sum_{t=1}^{T} D_{\mathrm{KL}}\left(q(x^{t-1}|x^t,x^0)\,\|\,p_\theta(x^{t-1}|x^t)\right)}_{\mathcal{L}_{diff}}\right] + \underbrace{\mathbb{E}_q\left[-\sum_{t=1}^{T}\log p_\theta(x^0|x^t)\right]}_{\mathcal{L}_{rec}} \\
&= \mathcal{L}_{prior} + \sum_{t=1}^{T}\mathbb{E}_q\left[\frac{1}{2\bar{\sigma}_t^2}\left\|x^0 - x_\theta(x^t,t)\right\|^2\right] + \sum_{t=1}^{T}\mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\left\|x^0 - x_\theta(x^t,t)\right\|^2\right] \\
&\approx \underbrace{\left\|x^0 - x_\theta(x^t,t)\right\|^2}_{\mathcal{L}_{DDPM}}
\end{aligned} \tag{17}$$

Therefore, When predicting x^0, with a trainable non markovian reverse process defined in Eq. 15 will result in DDPM training objective.

A.3. Variational Objective

To calculate the diffusion term $\mathcal{L}_{diff}$ of the objective (4), we need to compute the KL divergence between the forward posterior distribution $q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t,\mathbf{x},c)$ and the reverse distribution $p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t,c)$. Since we use parameterization $p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t,c) = q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t,\hat{\mathbf{x}}_\theta(x^t,t),c)$, both of these distributions are normal distributions with the same variance, so we can evaluate the KL divergence between them analytically as follows:

$$D_{KL}\left(q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t,\mathbf{x},c)\|p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t,c)\right) =$$
$$\frac{1}{2\sigma_{t-1|t}^2}\left\|\zeta_1\sqrt{\bar{\alpha}_t}(T_\phi(\mathbf{x},t) - T_\phi(\hat{\mathbf{x}}_\theta(\mathbf{x}^t,t),t)) - (\zeta_1\sqrt{\bar{\alpha}_t}+\zeta_0)\left(T_\phi(\hat{\mathbf{x}}_\theta(\mathbf{x}^t,t-1),t) - T_\phi(\mathbf{x},t-1)\right)\right\|_2^2. \tag{18}$$

We can compute the prior term as follows:

$$\begin{aligned}
D_{KL}\left(q_\phi(\mathbf{x}^T|\mathbf{x},c)\|p(\mathbf{x}^T|c)\right) &= \frac{1}{2}\left[\log\frac{|I|}{\sigma_T^2 I} - d + \mathrm{Tr}\{I^{-1}\sigma_T^2 I\} + \left\|\sqrt{\bar{\alpha}_T}T_\phi(\mathbf{x},T) + (1-\sqrt{\bar{\alpha}_T})c - c\right\|_2^2\right] \\
&= \frac{1}{2}\left[-d\log\sigma_T^2 + d\sigma_T^2 + \left\|\sqrt{\bar{\alpha}_T}T_\phi(\mathbf{x},T) - \sqrt{\bar{\alpha}_T}c\right\|_2^2\right] \\
&= \frac{1}{2}\left(d\left(\sigma_T^2 - \log\sigma_T^2 - 1\right) + \bar{\alpha}_T\left\|T_\phi(\mathbf{x},T) - c\right\|_2^2\right). \\
&= \frac{1}{2}\bar{\alpha}_T\left\|T_\phi(\mathbf{x},T) - c\right\|_2^2.
\end{aligned} \tag{19}$$

The reconstruction term is considered for all latent variables similar to VAE(Kingma, 2013b)

B. Implementation Details

The MLP modules within the embeddings of the condition and noised input, as well as the decoder, are from TiDE (Das et al., 2023a). This MLP framework is acknowledged as an effective component for universal time series analysis models (Das et al., 2023b). The encoder employs the diffusion transformer block (DiT), with the diffusion time step adjusted by adaptive layer normalization (Peebles & Xie, 2023; Esser et al., 2024). Hyperparameters are tuned specifically regarding embedding layers, encoder layers, and hidden dimensions while other hyperparameters remain constant as detailed in Section 4.1.2. Optimal hyperparameters are selected via cross-validation and summarized for various datasets in Table B.

Table 6. Best hyperparameters for different datasets

datasets	hidden dimension	embedding layers	encoder layers
NorPool	256	1	2
Caiso	128	1	2
Weather	64	1	1
ETTm1	512	1	1
Wind	512	1	1
Traffic	64	2	2
Electricity	256	2	1
ETTh1	256	2	3
Exchange	64	2	1

C. Baselines

We have thoroughly benchmarked our CN-Diff model against 23 baseline models using a variety of methodologies, from basis expansion techniques to diffusion models. These include: (i) diffusion-based models (Shen et al., 2024; Shen & Kwok, 2023; Rasul et al., 2021; Tashiro et al., 2021; Lopez Alcaraz & Strodthoff, 2023); (ii) basis expansion methods such as those detailed in (Challu et al., 2023; Zhou et al., 2022a; Fan et al., 2022; Oreshkin et al., 2019); (iii) alternative generative techniques like GAN and VAE as noted in (Li et al., 2022a; Rangapuram et al., 2023; Jeha et al., 2022). We also evaluated CN-Diff against transformers (Shabani et al., 2023; Nie et al., 2023; Zhou et al., 2022b; Wu et al., 2021; Liu et al., 2022b; Zhou et al., 2021; Vaswani, 2017b) and other deep learning approaches (Liu et al., 2022a; Zeng et al., 2023; Bahdanau et al., 2015). Note that we don't implement these baseline models. Instead, we rely on the reported numbers in literature.

D. Results of MAE

Tables 8 and 9 present the mean absolute error (MAE) results for multivariate and univariate time series forecasting tasks. It is evident that CN-Diff continues to demonstrate a superior overall performance for MAE as the evaluation metric. The average rank difference for multivariate is significant, but for univariate, it is almost comparable to mr-Diff. Furthermore, acknowledging that the performance of deep models in time series forecasting can be affected by various random initializations, Table 7 illustrates the prediction results for univariate time series in five distinct random runs.

Table 7. Univariate prediction errors of CN-Diff obtained on five runs.

	Exchange		ETTh1		Weather		Electricity	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
0	0.0152	0.0933	0.0660	0.1991	0.0025	0.0342	0.2370	0.3465
1	0.0154	0.0934	0.0649	0.1975	0.0021	0.0341	0.2268	0.3387
2	0.0153	0.0936	0.0656	0.1985	0.0025	0.0343	0.2289	0.3426
3	0.0161	0.0950	0.0670	0.2001	0.0027	0.0349	0.2393	0.3431
4	0.0157	0.0943	0.0644	0.1977	0.0024	0.0346	0.2392	0.3429
mean	0.0155	0.0939	0.0656	0.1986	0.0024	0.0344	0.2342	0.3427
std deviation	0.0004	0.0007	0.0010	0.0010	0.0002	0.0003	0.0059	0.0027

Table 8. Multivariate prediction of MAEs on nine real-world time series datasets(subscripts is the rank). CSDI runs out of memory on Traffic and Electricity. Results of all baselines are from (Shen et al., 2024)

	NorPool	Caiso	Traffic	Electricity	Weather	Exchange	ETTh1	ETTm1	Wind	Rank
Ours	**0.554**$_{(1)}$	**0.191**$_{(1)}$	0.270$_{(4)}$	$\underline{0.243}_{(2)}$	$\underline{0.324}_{(2)}$	**0.079**$_{(1)}$	$\underline{0.421}_{(2)}$	0.378$_{(5)}$	0.701$_{(4)}$	2.4
mr-Diff	0.604$_{(3)}$	0.219$_{(4)}$	0.320$_{(6)}$	0.252$_{(4)}$	$\underline{0.324}_{(2)}$	0.082$_{(4)}$	0.422$_{(3)}$	$\underline{0.373}_{(2)}$	**0.675**$_{(1)}$	3.2
TimeDiff	0.611$_{(4)}$	0.234$_{(7)}$	0.384$_{(9)}$	0.305$_{(7)}$	**0.312**$_{(1)}$	0.091$_{(8)}$	0.430$_{(4)}$	**0.372**$_{(1)}$	0.687$_{(3)}$	4.9
TimeGrad	0.821$_{(20)}$	0.339$_{(19)}$	0.849$_{(23)}$	0.630$_{(22)}$	0.381$_{(15)}$	0.193$_{(20)}$	0.719$_{(23)}$	0.605$_{(23)}$	0.793$_{(22)}$	20.8
CSDI	0.777$_{(18)}$	0.345$_{(20)}$	-	-	0.374$_{(13)}$	0.194$_{(21)}$	0.438$_{(6)}$	0.442$_{(16)}$	0.741$_{(11)}$	15.0
SSSD	0.753$_{(14)}$	0.295$_{(11)}$	0.398$_{(14)}$	0.363$_{(13)}$	0.350$_{(9)}$	0.127$_{(13)}$	0.561$_{(18)}$	0.406$_{(11)}$	0.778$_{(19)}$	13.6
D^3 VAE	0.692$_{(10)}$	0.331$_{(17)}$	0.483$_{(18)}$	0.372$_{(15)}$	0.380$_{(14)}$	0.301$_{(23)}$	0.502$_{(15)}$	0.391$_{(9)}$	0.779$_{(20)}$	15.7
CPF	0.889$_{(22)}$	0.424$_{(22)}$	0.714$_{(22)}$	0.643$_{(23)}$	0.781$_{(24)}$	0.082$_{(4)}$	0.597$_{(21)}$	0.472$_{(17)}$	0.757$_{(16)}$	19.0
PSA-GAN	0.890$_{(23)}$	0.477$_{(23)}$	0.697$_{(21)}$	0.533$_{(21)}$	0.578$_{(23)}$	0.087$_{(7)}$	0.546$_{(17)}$	0.488$_{(19)}$	0.756$_{(14)}$	18.7
N-Hits	0.643$_{(8)}$	0.221$_{(5)}$	$\underline{0.268}_{(2)}$	0.245$_{(3)}$	0.335$_{(5)}$	0.085$_{(6)}$	0.480$_{(9)}$	0.388$_{(7)}$	0.734$_{(9)}$	6.0
FiLM	0.646$_{(9)}$	0.278$_{(9)}$	0.398$_{(14)}$	0.320$_{(9)}$	0.336$_{(6)}$	**0.079**$_{(1)}$	0.436$_{(5)}$	0.374$_{(3)}$	0.717$_{(6)}$	6.9
Depts	0.611$_{(4)}$	0.204$_{(3)}$	0.568$_{(20)}$	0.401$_{(18)}$	0.394$_{(17)}$	0.100$_{(10)}$	0.491$_{(13)}$	0.412$_{(13)}$	0.751$_{(13)}$	12.3
NBeats	0.832$_{(21)}$	0.235$_{(8)}$	**0.265**$_{(1)}$	0.370$_{(14)}$	0.420$_{(18)}$	$\underline{0.081}_{(3)}$	0.521$_{(16)}$	0.409$_{(12)}$	0.741$_{(11)}$	11.6
Scaleformer	0.769$_{(17)}$	0.310$_{(13)}$	0.379$_{(8)}$	0.304$_{(6)}$	0.438$_{(19)}$	0.138$_{(15)}$	0.579$_{(20)}$	0.475$_{(18)}$	0.864$_{(23)}$	15.4
PatchTST	0.710$_{(11)}$	0.293$_{(10)}$	0.411$_{(17)}$	0.348$_{(12)}$	0.555$_{(22)}$	0.147$_{(16)}$	0.489$_{(12)}$	0.392$_{(10)}$	0.720$_{(7)}$	13.0
FedFormer	0.744$_{(12)}$	0.317$_{(14)}$	0.385$_{(10)}$	0.341$_{(11)}$	0.347$_{(8)}$	0.233$_{(22)}$	0.484$_{(10)}$	0.413$_{(14)}$	0.762$_{(17)}$	13.1
Autoformer	0.751$_{(13)}$	0.321$_{(15)}$	0.392$_{(13)}$	0.313$_{(8)}$	0.354$_{(10)}$	0.167$_{(17)}$	0.484$_{(10)}$	0.496$_{(20)}$	0.756$_{(14)}$	13.3
Pyraformer	0.781$_{(19)}$	0.371$_{(21)}$	0.390$_{(11)}$	0.379$_{(16)}$	0.385$_{(16)}$	0.112$_{(12)}$	0.493$_{(14)}$	0.435$_{(15)}$	0.735$_{(10)}$	14.9
Informer	0.757$_{(15)}$	0.336$_{(18)}$	0.391$_{(12)}$	0.383$_{(17)}$	0.364$_{(11)}$	0.192$_{(19)}$	0.605$_{(22)}$	0.542$_{(21)}$	0.772$_{(18)}$	17.0
Transformer	0.765$_{(16)}$	0.321$_{(15)}$	0.410$_{(16)}$	0.405$_{(19)}$	0.370$_{(12)}$	0.178$_{(18)}$	0.567$_{(19)}$	0.592$_{(22)}$	0.785$_{(21)}$	17.6
SCINet	$\underline{0.601}_{(2)}$	$\underline{0.193}_{(2)}$	0.335$_{(7)}$	0.280$_{(5)}$	0.344$_{(7)}$	0.137$_{(14)}$	0.463$_{(8)}$	0.389$_{(8)}$	0.732$_{(8)}$	6.8
NLinear	0.636$_{(6)}$	0.223$_{(6)}$	0.293$_{(5)}$	**0.239**$_{(1)}$	0.328$_{(4)}$	0.091$_{(8)}$	**0.418**$_{(1)}$	0.375$_{(4)}$	0.706$_{(5)}$	4.4
DLinear	0.640$_{(7)}$	0.497$_{(24)}$	$\underline{0.268}_{(2)}$	0.336$_{(10)}$	0.444$_{(20)}$	0.102$_{(11)}$	0.442$_{(7)}$	0.378$_{(5)}$	$\underline{0.686}_{(2)}$	9.8
LSTMa	0.974$_{(24)}$	0.305$_{(12)}$	0.510$_{(19)}$	0.444$_{(20)}$	0.501$_{(21)}$	0.534$_{(21)}$	0.782$_{(24)}$	0.699$_{(24)}$	0.897$_{(24)}$	21.0

Table 9. Univariate prediction of MAEs on nine real-world time series datasets(subscripts is the rank). Results of all baselines are from (Shen et al., 2024)

Method	NorPool	Caiso	Traffic	Electricity	Weather	Exchange	ETTh1	ETTm1	Wind	Rank
Ours	**0.590**$_{(1)}$	$\underline{0.206}_{(2)}$	0.206$_{(3)}$	0.343$_{(4)}$	0.034$_{(4)}$	**0.094**$_{(1)}$	$\underline{0.199}_{(2)}$	**0.149**$_{(1)}$	$\underline{1.168}_{(2)}$	2.2
mr-Diff	0.609$_{(3)}$	0.212$_{(5)}$	$\underline{0.197}_{(2)}$	**0.332**$_{(1)}$	**0.032**$_{(1)}$	**0.094**$_{(1)}$	**0.196**$_{(1)}$	**0.149**$_{(1)}$	$\underline{1.168}_{(2)}$	1.9
TimeDiff	0.613$_{(4)}$	0.209$_{(4)}$	0.207$_{(4)}$	0.341$_{(3)}$	0.035$_{(5)}$	0.102$_{(8)}$	0.202$_{(3)}$	0.154$_{(7)}$	1.209$_{(6)}$	4.9
TimeGrad	0.841$_{(24)}$	0.386$_{(23)}$	0.894$_{(24)}$	0.898$_{(24)}$	0.036$_{(7)}$	0.155$_{(22)}$	0.212$_{(9)}$	0.167$_{(13)}$	1.239$_{(12)}$	17.6
CSDI	0.763$_{(21)}$	0.282$_{(15)}$	0.468$_{(21)}$	0.540$_{(20)}$	0.037$_{(8)}$	0.200$_{(24)}$	0.221$_{(13)}$	0.170$_{(15)}$	1.218$_{(8)}$	16.1
SSSD	0.770$_{(22)}$	0.263$_{(13)}$	0.226$_{(7)}$	0.403$_{(9)}$	0.041$_{(12)}$	0.118$_{(17)}$	0.250$_{(21)}$	0.169$_{(14)}$	1.356$_{(23)}$	15.3
D^{3}VAE	0.774$_{(23)}$	0.613$_{(24)}$	0.237$_{(10)}$	0.539$_{(19)}$	0.039$_{(10)}$	0.107$_{(14)}$	0.221$_{(13)}$	0.160$_{(10)}$	1.321$_{(20)}$	15.9
CPF	0.710$_{(16)}$	0.338$_{(19)}$	0.385$_{(20)}$	0.592$_{(22)}$	0.035$_{(5)}$	**0.094**$_{(1)}$	0.221$_{(13)}$	0.153$_{(6)}$	1.256$_{(13)}$	12.8
PSA-GAN	0.623$_{(6)}$	0.250$_{(9)}$	0.355$_{(18)}$	0.373$_{(7)}$	0.139$_{(23)}$	0.109$_{(15)}$	0.225$_{(17)}$	0.174$_{(17)}$	1.287$_{(17)}$	14.3
N-Hits	0.646$_{(8)}$	0.276$_{(14)}$	0.232$_{(8)}$	0.419$_{(10)}$	$\underline{0.033}_{(2)}$	0.100$_{(6)}$	0.228$_{(18)}$	0.157$_{(9)}$	1.256$_{(13)}$	9.8
FiLM	0.654$_{(10)}$	0.290$_{(16)}$	0.315$_{(15)}$	0.362$_{(6)}$	0.069$_{(15)}$	0.104$_{(11)}$	0.210$_{(7)}$	**0.149**$_{(1)}$	1.189$_{(4)}$	9.4
Depts	0.616$_{(5)}$	**0.205**$_{(1)}$	0.241$_{(11)}$	0.434$_{(13)}$	0.102$_{(20)}$	0.106$_{(13)}$	0.202$_{(3)}$	0.165$_{(11)}$	1.472$_{(24)}$	11.2
NBeats	0.671$_{(11)}$	0.228$_{(6)}$	0.225$_{(6)}$	0.439$_{(14)}$	0.130$_{(22)}$	0.096$_{(4)}$	0.242$_{(19)}$	0.165$_{(11)}$	1.236$_{(10)}$	11.4
Scaleformer	0.687$_{(13)}$	0.320$_{(17)}$	0.375$_{(19)}$	0.430$_{(11)}$	0.083$_{(18)}$	0.148$_{(20)}$	0.302$_{(23)}$	0.210$_{(23)}$	1.348$_{(22)}$	18.4
PatchTST	**0.590**$_{(1)}$	0.260$_{(12)}$	0.269$_{(12)}$	0.478$_{(18)}$	0.098$_{(19)}$	0.111$_{(16)}$	0.260$_{(22)}$	0.174$_{(17)}$	1.338$_{(21)}$	15.3
FedFormer	0.725$_{(18)}$	0.254$_{(10)}$	0.278$_{(13)}$	0.453$_{(15)}$	0.057$_{(14)}$	0.168$_{(23)}$	0.212$_{(9)}$	0.195$_{(21)}$	1.271$_{(15)}$	15.3
Autoformer	0.755$_{(20)}$	0.339$_{(20)}$	0.495$_{(22)}$	0.623$_{(23)}$	0.040$_{(11)}$	0.152$_{(21)}$	0.220$_{(12)}$	0.174$_{(17)}$	1.319$_{(19)}$	18.3
Pyraformer	0.747$_{(19)}$	0.257$_{(11)}$	0.215$_{(5)}$	0.455$_{(16)}$	0.107$_{(21)}$	0.104$_{(11)}$	0.211$_{(8)}$	0.179$_{(20)}$	1.284$_{(16)}$	13.1
Informer	0.698$_{(14)}$	0.345$_{(21)}$	0.308$_{(14)}$	0.433$_{(12)}$	0.069$_{(15)}$	0.118$_{(17)}$	0.212$_{(9)}$	0.172$_{(16)}$	1.236$_{(10)}$	14.1
Transformer	0.723$_{(17)}$	0.345$_{(21)}$	0.336$_{(17)}$	0.469$_{(17)}$	0.071$_{(17)}$	0.103$_{(10)}$	0.247$_{(20)}$	0.196$_{(22)}$	1.212$_{(7)}$	16.4
SCINet	0.653$_{(9)}$	0.244$_{(8)}$	0.322$_{(16)}$	0.377$_{(8)}$	0.037$_{(8)}$	0.101$_{(7)}$	0.205$_{(6)}$	0.150$_{(5)}$	**1.167**$_{(1)}$	7.6
NLinear	0.637$_{(7)}$	0.238$_{(7)}$	**0.192**$_{(1)}$	$\underline{0.334}_{(2)}$	$\underline{0.033}_{(2)}$	0.097$_{(5)}$	0.203$_{(5)}$	**0.149**$_{(1)}$	1.197$_{(5)}$	3.9
DLinear	0.671$_{(11)}$	$\underline{0.206}_{(2)}$	0.236$_{(9)}$	0.348$_{(5)}$	0.310$_{(24)}$	0.102$_{(8)}$	0.222$_{(16)}$	0.155$_{(8)}$	1.221$_{(9)}$	10.2
LSTMa	0.707$_{(15)}$	0.333$_{(18)}$	0.757$_{(22)}$	0.557$_{(21)}$	0.053$_{(13)}$	0.136$_{(19)}$	0.332$_{(24)}$	0.239$_{(24)}$	1.298$_{(18)}$	19.3

E. Ablations

In this section, we present ablation studies for various hyperparameters, beginning with an examination of different components within CN-Diff, as well as the look-back window, diffusion time steps, and Gaussian variance schedule.

E.1. Architecture components

Table E.1 presents an analysis of the various components integrated into CN-Diff. Beginning with DDPM (Ho et al., 2020), we provide results that illustrate the effects of incorporating only non-linearity, the conditional aspect independently, and the combined integration of both into the DDPM, evaluated across multiple datasets.

Table 10. Predicting MSE and MAE for different components of CN-Diff for Multivariate forecasting (improvement relative to Diffusion is denoted within brackets.)

Methods	Diffusion		+ T_ϕ		+ CN-Diff Cond		All (CN-Diff)	
Datasets	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	1.748	0.927	1.320(25%)	0.730(21%)	0.943(46%)	0.704(24%)	**0.405**(77%)	**0.421**(55%)
Exchange	0.090	0.219	0.016(82%)	0.080(63%)	0.021(60%)	0.101(54%)	**0.015**(83%)	**0.079**(64%)
Caiso	0.586	0.553	0.403(31%)	0.433(22%)	0.242(59%)	0.307(44%)	**0.094**(84%)	**0.191**(65%)
Norpool	1.819	1.047	1.441(42%)	0.918(12%)	1.142(37%)	0.814(22%)	**0.531**(71%)	**0.554**(47%)
Electricity	1.824	1.030	1.597(12%)	0.951(8%)	0.536(71%)	0.512(50%)	**0.145**(92%)	**0.243**(76%)
Traffic	1.794	0.885	1.397(22%)	0.799(10%)	0.848(53%)	0.479(46%)	**0.374**(79%)	**0.270**(70%)

E.2. Look back Window

Table E.2 presents the prediction MSE and MAE for CN-Diff utilizing various lengths of the lookback window. The values considered are L = {96, 192, 336, 720, 1440}. It is evident that on the datasets Electricity, ETTh1, and weather (corresponding to 168-step, 168-step, and 672-step-ahead predictions, respectively), satisfactory performance is achieved when L is set to 336 or higher.

Table 11. Predicting MSE and MAE for different Lookback windows (L) for Multivariate forecasting

Look back window	96		192		336		720		1440	
Datasets	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.433	0.437	0.426	0.437	**0.405**	**0.421**	0.434	0.448	0.440	0.452
Weather	0.530	0.503	0.347	0.367	0.339	0.354	0.307	0.335	**0.296**	**0.324**
Electricity	0.167	0.262	0.152	0.248	0.149	0.246	0.148	0.246	**0.145**	**0.243**

E.3. Diffusion Parameters

Diffusion steps, denoted as T, is a critical parameter. Table E.3 presents the MSE and MAE of CN-Diff as a function of T. It is evident that establishing T = 100 results in consistent performance in all three datasets examined. This observation is consistent with the findings in (Li et al., 2022b), which suggest that a minimal value of T may result in incomplete diffusion, while an excessively large K might lead to superfluous computational overhead.(For all datasets analyzed, the best length of the look-back window was considered.)

The variance of Gaussian noise β_k, is formulated through the implementation of a quadratic variance schedule. In the present study, the variable β_T is varied within the set {0.001, 0.01, 0.1, 0.9}, while β_1 is maintained at a constant value of 10^{-4}. As evidenced in Table E.3, optimizing the parameter setting $\beta_T = 0.1$ consistently results in enhanced performance.

Table 12. Predicting MSE and MAE for different diffusion noise steps (T) for Multivariate forecasting

Noise Steps	50		100		200		500	
Datasets	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.415	0.431	**0.405**	**0.421**	0.409	0.426	0.422	0.437
Weather	0.299	0.330	**0.296**	**0.324**	0.325	0.360	0.362	0.378
Electricity	0.145	0.244	**0.145**	**0.243**	0.145	0.243	0.147	0.244

Table 13. Predicting MSE and MAE for different variance upper bound (β_T) for Multivariate forecasting

β_T	0.001		0.01		0.1		0.9	
Datasets	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.807	0.656	0.462	0.463	**0.405**	**0.421**	0.425	0.438
Weather	0.312	0.347	0.340	0.375	**0.296**	**0.324**	0.357	0.373
Electricity	1.745	0.986	0.243	0.345	**0.145**	**0.243**	0.146	0.243

F. Qualitative analysis

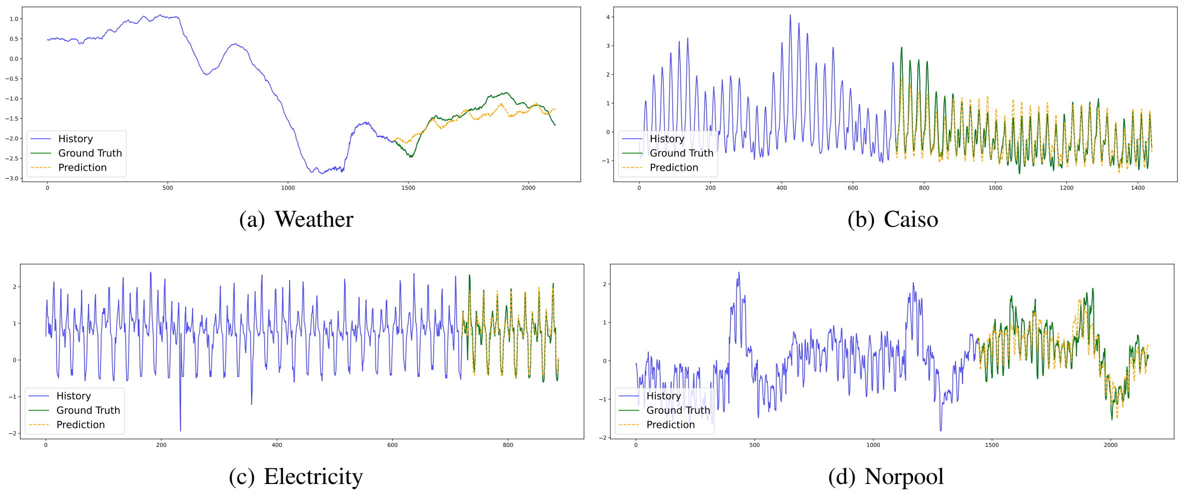

Figure 4. Visualization of CN-Diff forecasts across various datasets

The qualitative analysis presented in Figure 4 demonstrates the forecasting capabilities of our CN-Diff model across four diverse datasets: Weather, Caiso, Electricity, and Norpool. These visualizations provide strong evidence of the model's effectiveness in capturing temporal patterns and generating accurate predictions across varying domains. In Figure 4(a), the Weather dataset results show CN-Diff successfully tracking the overall trend and seasonal patterns, with predictions (green line) closely following the ground truth (blue line). For the Caiso dataset in Figure 4(b), the model effectively captures the high-frequency oscillations that characterize this time series, with particularly strong alignment between the predicted and actual values in the latter portions of the sequence. The Electricity dataset results in Figure 4(c) further illustrate CN-Diff's ability to model periodic patterns with consistent amplitude and frequency matching the ground truth. Finally, Figure 4(d) showcases results on the Norpool dataset, where our model accurately predicts the irregular fluctuations and demonstrates robustness to the dataset's inherent volatility.

G. Computational analysis

We present the computational cost analysis to compare our model with other diffusion-based time series forecasting methods. Refer to the tables for training (Table:14) and inference (Table:15) comparisons (Results for other baselines are taken from (Shen et al., 2024)). Note that, SSSD utilizes structural state-space-based diffusion layers along with multiple dense layers; while TimeGrad leverages RNN hidden states; TimeDiff incorporates future mean and autoregressive initialization; and mr-Diff employs multiple diffusion models. In contrast, our innovation lies in the diffusion formulation rather than in the architectural modifications. Both key components we introduced, the nonlinear transform ($T_\phi(\cdot)$) and the condition network, consist of simple linear layers with an activation function. This design choice allows us to achieve greater computational efficiency compared to existing diffusion models for time series forecasting, and this can be realized by the table below.

Table 14. Training time (ms) for different models and sequence lengths (H) for ETTh1 univariate.

	$H = 96$	$H = 168$	$H = 192$	$H = 336$	$H = 720$
CN-Diff	**0.21**	**0.26**	**0.27**	**0.31**	**0.39**
mr-Diff	0.59	0.69	0.71	0.74	0.82
TimeDiff	0.71	0.75	0.77	0.82	0.85
TimeGrad	2.11	2.42	3.21	4.22	5.93
CSDI	5.72	7.09	7.59	10.59	17.21
SSSD	16.98	19.34	22.64	32.12	52.93

Table 15. Inference time (ms) for different models and sequence lengths (H) for ETTh1 univariate.

Model	Trainable Params	$H = 96$	$H = 168$	$H = 192$	$H = 336$	$H = 720$
CN-Diff	**1.1M**	**6.2**	**6.7**	**6.9**	**7.8**	**9.1**
mr-Diff	1.4M	12.5	14.3	14.9	16.8	27.5
TimeDiff	1.7M	16.2	17.3	17.6	26.5	34.6
TimeGrad	3.1M	870.2	1620.9	1854.5	3119.7	6724.1
CSDI	10M	90.4	128.3	142.8	398.9	513.1
SSSD	32M	418.6	590.2	645.4	1054.2	2516.9

H. Additional Results

H.1. Additional Datasets

In order to further validate our approach, we have carried out supplementary experiments employing multivariate synthetic data. This data utilizes Gaussian process kernels as described by (Taga et al., 2025). Additionally, we incorporated the CSI-300 dataset, which represents the price-weighted average index of 300 prominent companies listed on the Shanghai and Shenzhen Stock Exchanges, as cited in (Jia et al., 2024). The results of these experiments, detailed in Table 16 below, indicate that our model achieves a higher level of performance when benchmarked against other existing models.

Table 16. Performance comparison of CN-Diff across models on Stock and Synthetic datasets using MSE and MAE metrics.

Model	Stock (MSE)	Stock (MAE)	Synthetic (MSE)	Synthetic (MAE)
CN-Diff	**0.020**	**0.109**	**0.722**	**0.308**
Autoformer	0.058	0.190	0.767	0.339
PatchTST	0.028	0.129	0.771	0.345
FiLM	0.040	0.159	0.752	0.319
DLinear	0.044	0.170	0.747	0.311
CSDI	0.037	0.169	0.809	0.506

H.2. Additional Models

In our experiments, we have compared our method with CPF(Rangapuram et al., 2023), a model using Graph Neural Networks. To further strengthen our analysis, we now incorporate a comparison with SageFormer(Zhang et al., 2024) as well, which is a series-aware graph-enhanced Transformer model for time series forecasting. The table below17 demonstrates CN-Diff's superiority over graph-based methods in time series forecasting.

Table 17. Comparison of Model Performance Metrics across different Datasets

Dataset	CN-Diff (MSE/MAE)	Sage Former (MSE/MAE)
ETTm1	**0.340 / 0.378**	0.370 / 0.388
Electricity	**0.145 / 0.243**	0.159 / 0.255
Stock	**0.020 / 0.109**	0.030 / 0.135
Exchange	**0.016 / 0.079**	0.016 / 0.081
ETTh1	**0.405** / 0.421	0.421 / **0.419**
Weather	**0.296 / 0.324**	0.324 / 0.344

H.3. Analysis of Non-linear Transformation($T_\phi(.)$)

In Table 5 and E.1, we have shown empirical evidence through the ablation results presented that CN-Diff performs better than the ablation run without T_ϕ. To understand what is learned by T_ϕ, we take a trained model and explore the correlation between the features in the learned latent representation space of $T_\phi(x, t)$ at different diffusion model timesteps. We observed that there are increasing correlations between the features in learned latent space at different timesteps as shown in tables below. Thus, we hypothesize that this increased correlation and time-dependent adaptability perhaps facilitate a more effective diffusion process for time series forecasting. Similar observations of correlation in latent space that help diffusion models have been made in image and video diffusion works (Ge et al., 2023).

In the space of diffusion models for time series, paper (Tashiro et al., 2021) has noted that learning the correlation between feature and temporal space is necessary for time series imputation tasks. We will add these to the discussion in the ablation study section in the revision.

Table 18. Feature Correlation Matrix for Actual input (ETTh1 dataset)

Features	0	1	2	3	4	5	6
0	**1.00**	0.08	**0.99**	0.07	0.22	-0.20	-0.19
1	0.08	**1.00**	0.08	**0.94**	0.05	0.22	-0.06
2	**0.99**	0.08	**1.00**	0.09	0.12	-0.27	-0.19
3	0.07	**0.94**	0.09	**1.00**	-0.10	-0.06	-0.07
4	0.22	0.05	0.12	-0.10	**1.00**	**0.56**	-0.06
5	-0.20	0.22	-0.27	-0.06	**0.56**	**1.00**	0.11
6	-0.19	-0.06	-0.19	-0.07	-0.06	0.11	**1.00**

Table 19. Feature Correlation Matrix for Transformed Input $T_\phi(.)$ (ETTh1 dataset)

Features	0	1	2	3	4	5	6
0	**1.00**	**-0.52**	**-0.60**	**0.99**	**0.93**	**-0.96**	**-0.56**
1	**-0.52**	**1.00**	-0.31	**-0.58**	**-0.68**	**0.60**	**0.63**
2	**-0.60**	-0.31	**1.00**	**-0.57**	-0.33	**0.55**	-0.08
3	**0.99**	**-0.58**	**-0.57**	**1.00**	**0.94**	**-0.97**	**-0.59**
4	**0.93**	**-0.68**	-0.33	**0.94**	**1.00**	**-0.86**	**-0.80**
5	**-0.96**	**0.60**	**0.55**	**-0.97**	**-0.86**	**1.00**	0.43
6	**-0.56**	**0.63**	-0.08	**-0.59**	**-0.80**	0.43	**1.00**