Latent Diffusion Transformer for Probabilistic Time Series Forecasting

Shibo Feng[1,2,3], **Chunyan Miao**[1,2,3], **Zhong Zhang**[4], **Peilin Zhao**[4*]

[1]School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore
[2]Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY), NTU, Singapore
[3]Webank-NTU Joint Research Institute on Fintech, NTU, Singapore
[4]Tencent AI Lab, Shenzhen, China
{shibo001, ascymiao}@ntu.edu.sg, {todzhang, masonzhao}@tencent.com

Abstract

The probability prediction of multivariate time series is a notoriously challenging but practical task. This research proposes to condense high-dimensional multivariate time series forecasting into a problem of latent space time series generation, to improve the expressiveness of each timestamp and make forecasting more manageable. To solve the problem that the existing work is hard to extend to high-dimensional multivariate time series, we present a latent multivariate time series diffusion framework called **L**atent **D**iffusion **T**ransformer (LDT), which consists of a symmetric statistics-aware autoencoder and a diffusion-based conditional generator, to implement this idea. Through careful design, the time series autoencoder can compress multivariate timestamp patterns into a concise latent representation by considering dynamic statistics. Then, the diffusion-based conditional generator is able to efficiently generate realistic multivariate timestamp values on a continuous latent space under a novel self-conditioning guidance which is modeled in a non-autoregressive way. Extensive experiments demonstrate that our model achieves state-of-the-art performance on many popular high-dimensional multivariate time series datasets.

Introduction

Forecasting time series data is crucial across various sectors, including finance (Sezer, Gudelek, and Ozbayoglu 2020), energy (Cao et al. 2020), traffic (Liu et al. 2016; Feng et al. 2023) and human identification (Rao and Miao 2022; Rao et al. 2021). Multivariate forecasting, prevalent in practical applications, is more important and popular in industrial fields. For example, power companies analyze billions of data points from numerous clients to monitor electricity consumption, reflecting the complexity and significance of this task.

Latent diffusion models (Rombach et al. 2022), a simple and efficient way to significantly improve both the training and sampling efficiency of denoising diffusion models (Ho, Jain, and Abbeel 2020) without degrading their quality. This particular class of latent generative models has gained significant recognition and accomplishments in recent times, particularly in processing high-dimensional types of data such as high-resolution images (Ho, Jain, and Abbeel 2020; Takagi and Nishimoto 2023), natural languages (Li et al. 2022a; Yuan et al. 2022), and audios (Huang et al. 2022; Ruan et al. 2023).

Multivariate time series forecasting seeks to predict future trends accurately but faces challenges due to its complexity and computational demands. The common approach of using deep, auto-regressive (Woo et al. 2022; Liu et al. 2022; Wu et al. 2020) models to predict future timestamps is hindered by the high dimensionality of the data and the model's structure. This leads to two main issues: significant computational resource requirements, limiting scalability, and the accumulation of errors in forecasts, particularly in high-dimensional series. Therefore, there's a pressing need for an innovative forecasting framework that can efficiently and effectively predict future trends with reduced computational load and increased speed.

Latent-space generation, an efficient alternative in time series forecasting, employs a pre-trained autoencoder to mitigate data redundancy, transferring generation from the time to a latent domain. The primary challenge is the distribution shift problem, as statistical properties like mean and variance vary over time (Fan et al. 2023; Kim et al. 2021). Traditional models struggle with numerical inaccuracies when using historical timestamps as the inputs of the autoencoder. Our novel approach dynamically updates statistical parameters in pre-training, ensuring high-quality, accurate latent representations for each timestamp.

Existing multivariate time-series diffusion models face two major issues. First, the autoregressive structure (Rasul et al. 2021) leads to poor long-range prediction, error accumulation, and slow inference. Second, most models excel in low-dimensional series (Tashiro et al. 2021; Alcaraz and Strodthoff 2022; Shen and Kwok 2023) but falter in high dimensions. To overcome these challenges, our approach emphasizes a non-autoregressive, resource-efficient denoising network for forecasting. We introduce a self-conditioning-based transformer denoising structure that effectively denoises time variables in a continuous latent space, incorporating covariant features akin to strategies in image generation (Chen, Zhang, and Hinton 2022; Yang et al. 2022; Ho and Salimans 2022). This Transformer diffusion module significantly reduces computational complexity, resource use,

*Corresponding author

and increases sampling speed compared to autoregressive models.

In this work, we introduce a novel two-stage, non-autoregressive diffusion architecture for multivariate probabilistic forecasting. Our experiments across various real-world datasets demonstrate that this model has surpassed existing state-of-the-art generative models in high-dimensional multivariate time series forecasting. The key contributions of our work are:

- Introduction of the LDT model, a new approach in multivariate time series forecasting, leveraging latent space representations for high-accuracy predictions in high-dimensional scenarios.
- Development of a practical LDT structure featuring a unique self-conditioning mechanism and a non-autoregressive transformer, enabling constrained self-conditioned predictions.
- We perform extensive experiments with multiple multivariate forecasting datasets, demonstrating LDT′s superior performance compared with the recent state-of-the-art forecast methods, for multivariate time series probabilistic predictions.

Background

Diffusion Models (Ho, Jain, and Abbeel 2020) diffusion models are probability generative models proposed to generate the target data distribution $p(x)$ by iterative denoising a normally distributed variable. Diffusion probabilistic models are composed of the fixed forward process and the learnable reverse process, which is a Markov Chain of length T.

Forward Process It is a transition and fixed diffusion process, from the data distribution to a Gaussian distribution. Given a data sample $\mathbf{x} \in \mathbb{R}^d \sim p(\mathbf{x})$ and some latent variables $\{\mathbf{z}_0, \mathbf{z}_1, \cdots, \mathbf{z}_T\}$, which interpolate between the data distribution and a Gaussian distribution with the diffusion steps increase. The forward process can be formally described as a Markov chain parameterized with a series of variances β_t and $\alpha_t := 1 - \beta_t$.

$$\begin{aligned} q\left(\mathbf{z}_{1:T}|\mathbf{z}_0\right) &= \prod_{t=T}^{1} q\left(\mathbf{z}_t|\mathbf{z}_{t-1}\right), \\ \text{where } q\left(\mathbf{z}_t|\mathbf{z}_{t-1}\right) &\sim \mathcal{N}\left(\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I}\right), \end{aligned} \tag{1}$$

Since the more steps of the diffusion process, the more noise added, $q\left(z_t|x\right)$ has a closed-form solution, which can be described by a general form.

$$\begin{aligned} \mathbf{z}_t &\sim q\left(\mathbf{z}_t|\mathbf{x}\right) = \mathcal{N}\left(\sqrt{\overline{\alpha}_t}\mathbf{x}, \left(1-\overline{\alpha}_t\right)\mathbf{I}\right), \\ \mathbf{z}_t &= \sqrt{\overline{\alpha}_t}\mathbf{x} + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \end{aligned} \tag{2}$$

where $\overline{\alpha}_t = \prod_{i=1}^{t}\left(1-\beta_i\right) \in (0,1)$, $z_0 = x$ and $z_T \sim \mathcal{N}(0, \mathbf{I})$ As the diffusion steps increases, the latent variable z_t become noisier until the z_T is approximately a Gaussian variable, which is independent of the starting point x.

Reverse Process The learnable reverse process is defined by the inverted Markov chain $p_\theta\left(\mathbf{z}_{0:T}\right) = p\left(\mathbf{z}_T\right)\prod_{t=t}^{T} p_\theta\left(\mathbf{z}_{t-1}|\mathbf{z}_t\right)$, where $p\left(\mathbf{z}_T\right) = \mathcal{N}(0, \mathbf{I})$ is known. The $p_\theta\left(\mathbf{z}_{t-1}|\mathbf{z}_t\right)$ can be approximately driven from the following equation, $q\left(\mathbf{z}_{t-1} \mid \mathbf{z}_t, \mathbf{x}\right) = \mathcal{N}\left(\mu_t\left(\mathbf{z}_t, \mathbf{x}\right), \sigma_t^2\mathbf{I}\right)$, where $\mu_t\left(\mathbf{z}_t, \mathbf{x}\right)$ has a closed-form solution and σ_t is a hyperparameter. To get the $p_\theta\left(\mathbf{z}_{t-1}|\mathbf{z}_t\right)$, we train a denoising network θ to approximate the x given noisy latent z_t and the timestep t

$$L = \mathbb{E}_{\mathbf{x}\sim p(x), t\sim\mathcal{U}\{1,\cdots,T\}, \mathbf{z}_t}\left[\left\|\hat{\mathbf{x}}_\theta\left(\mathbf{z}_t, t\right) - \mathbf{x}\right\|_2^2\right], \tag{3}$$

where $z_t \sim q(z_t|x)$ and $p_\theta\left(\mathbf{z}_{t-1}|\mathbf{z}_t\right)$ is an approximation of the $q\left(\mathbf{z}_{t-1} \mid \mathbf{z}_t, \hat{\mathbf{x}}_\theta\left(\mathbf{z}_t, t\right)\right)$, which enables us to sample from a closed-form and denoise the latent variable by sampling z_{t-1} until we get the $z_0 = x \sim p(x)$.

$$\mathbf{z}_{t-1} \sim p_\theta\left(\mathbf{z}_{t-1}|\mathbf{z}_t\right) = q\left(\mathbf{z}_{t-1} \mid \mathbf{z}_t, \hat{\mathbf{x}}_\theta\left(\mathbf{z}_t, t\right)\right), \tag{4}$$

For sampling from the trained diffusion model, we utilize the inference distribution from Song et al. (2020) and therefore derive the sampling from $q(z_{t-1} \mid z_t, x) = \mathcal{N}\left(\mu_q\left(\boldsymbol{z}_t, \boldsymbol{x}\right), \Sigma_q(t)\mathbf{I}\right)$,

$$q\left(z_{t-1} \mid z_t, x\right) \propto N\left(z_{t-1}; \bar{\alpha}_t^1 z_t + \bar{\alpha}_t^2 x, \bar{\beta}_t I\right), \tag{5}$$

while setting $\Sigma_q(t) = 0$ gives the deterministic DDIM sampler, $\bar{\alpha}_t^1 = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$, $\bar{\alpha}_t^2 = \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}$ and $\bar{\beta}_t = \frac{(1-\alpha_t)(1-\bar{\alpha}_t-1)}{1-\bar{\alpha}_t}$. The detailed training and sampling procedures are shown in the Method section.

Method

Our approach to high-dimensional multivariate time series forecasting involves a two-stage process: a statistics-based time autoencoder and a Latent Diffusion Transformer (LDT) generator. The autoencoder dynamically updates global statistics during training for accurate future timestamp reconstruction. The LDT generator then produces latent conditions using self-conditions and a guidance mechanism, incorporating relevant covariates. This method efficiently captures the inherent dynamics and correlations in the time series data, as LDT is shown in Fig. 1. The specific algorithm details are shown in Algorithms 1 and 2.

Symmetric Time Series Compression

In order to ensure the generality and effectiveness of our model to generative high-power latent embedding, we constructed a simple and accurate autoencoder structure. Statistical properties such as mean and variance often change over time in time series, previous work "RevIN (Kim et al. 2021), DIT-sh (Fan et al. 2023)" had claimed that the discrepancy between different input sequences can significantly degrade the model performance. We found that in non-stationary multivariate time series, different batch samples that were randomly sampled will have high variance deviation, which will degrade the stability and efficacy of autoencoder training. Therefore, we propose a simple yet effective symmetric autoencoder structure with adaptive variance updating normalization layer (*VN*).

More precisely, given the look-back window data $X \in \mathbb{R}^{T\times d}$ and target $Y \in \mathbb{R}^{t\times d}$ in the time-space, the normalization layer *VN* normalizes the target Y to the $\hat{Y} =$

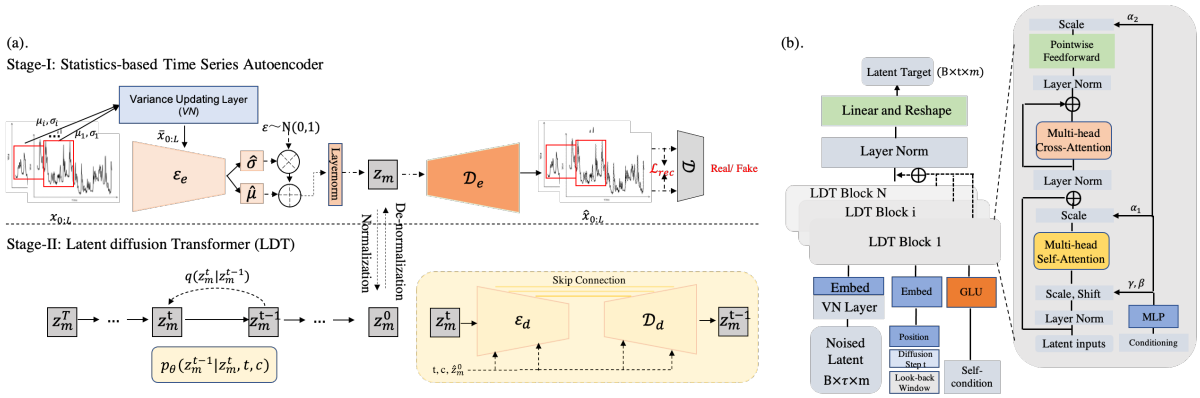

Figure 1: The framework of our proposed LDT (a). During the training process, the stage-I VAE is first trained to construct time series latent with reconstruction task, while LDT is trained to generate the future targets conditioned by self-condition, covariates, and diffusion step $\hat{z}_m^0$, t, c in the second stage. During the sampling process, the time series latent first be generated by LDT, and then input to decoder $\mathcal{D}_e$ to get the future targets. The black dashed lines stand for operations only involved in the training process. The details of the stage-II adaptive layernorm Transformer structure $(\mathcal{E}_d, \mathcal{D}_d)$ are shown in (b) and the structures of stage-I are shown in Appendix.

$VN([X, Y])$. Then, the encoder $\mathcal{E}$ encodes $\hat{Y}$ into a latent representation $Z = \mathcal{E}(\hat{Y})$, and the decoder $\mathcal{D}$ reconstructs the target time series from the latent, giving $Y = \mathcal{D}(Z) = \mathcal{D}(\mathcal{E}(\hat{Y}))$, where $Z \in \mathbb{R}^{t \times m} (m << d)$. Importantly, the encoder downsamples the time series by a factor $f = d/m$, and we use the factor $f \approx 2^m$, with $m \in \mathbb{N}$. The reason for choosing the size of f here is to reduce the training difficulty of the noise-added high-dimensional multivariate time series in the diffusion model. The framework overview of our autoencoder is demonstrated in Fig.1.

As illustrated in the figure, we first use instance normalization (Ulyanov, Vedaldi, and Lempitsky 2016) to calculate the instance-specific mean and standard deviation for scaling every input $W^i = [X^i, Y^i] \in \mathbb{R}^{\tau \times d}$ and $\tau = T + t$, which are described as $\mathbb{E}[W^i] = \frac{1}{\tau}\sum_{j=1}^{\tau} W_j^i$, $\mathrm{Var}[W^i] = \frac{1}{\tau}\sum_{j=1}^{\tau}\left(W_j^i - \mathbb{E}[W^i]\right)^2$ and $\hat{\mathbb{E}}^{n+1}[W^i] = \frac{1}{n}(\mathbb{E}^{n+1}[W^i] + \hat{\mathbb{E}}^n[W^i] \times (n-1))$, where n is the number of batches and the adaptive updated function of variances $\hat{Var}^{n+1}$ is the x same as $\hat{\mathbb{E}}^{n+1}$, we normalize the target Y^i through these updated statistics as $\hat{Y}^i = \gamma_d\left(\frac{Y^i - \hat{\mathbb{E}}^{n+1}[W^i]}{\sqrt{\hat{\mathbb{V}}^{n+1}[W^i] + \epsilon}}\right) + \beta_d$, where $\gamma_d, \beta_d \in \mathbb{R}^d$ are the learnable affine parameters. We gradually update the instance variance and mean that were utilized for regularization. On the one hand, the non-stationary information in the target sequence can be weakened, making it easier to train the autoencoder. Also, the generative results from the autoencoder make the diffusion model training in the second stage more stable and accurate.

Specifically, our autoencoder structure is a symmetrical model, and the specific modules are shown in Appendix due to the limited space. Also, to avoid arbitrarily high-variance latent spaces, we follow the regularization strategy proposed in the "Latent diffusion Model" which imposed a KL-penalty (i.e. weight the $\mathbb{KL}$ term by a factor 10^{-8}) towards a standard normal on the learned latent that preserves details of Y better and train all our autoencoder models in an adversarial manner, such that a timestamp-based discriminator $\mathcal{D}_\eta$ is optimized to differentiate original target time series from reconstructions $\mathcal{D}(\mathcal{E}(\hat{Y}))$. The full objective training loss function L of our autoencoder reads:

$$L = \min_{\mathcal{E},\mathcal{D}} \max_{\eta} \Big(L_{\text{rec}}(Y, \mathcal{D}(\mathcal{E}(Y))) - L_{\text{adv}}(\mathcal{D}(\mathcal{E}(Y))) + \log D_\eta(Y) + L_{\text{reg}}(Y; \mathcal{E}, \mathcal{D}) \Big),$$

where L_{reg} is a regularizing loss term which is to regularize the latent Z to be zero-centered and obtain a small variance. We found that different time series always have large variances, which may lead to the extremely unstable training of the following latent diffusion model. Detailed explanations are described in the experiment.

Latent Diffusion Transformer

Generative Modeling of Latent Representations Compared with applying the diffusion model directly in the time domain of high-dimensional multivariate time series, we introduce the trained time compression models consisting of $\mathcal{E}$ and $\mathcal{D}$ that take the efficient and low-dimensional time series representations to the following denoising network.

Unlike previous work that relied on autoregressive generative models in the time-space (Min et al. 2022b; Yi et al. 2023), we take advantage of the attention-based transformer models (Min et al. 2022a; Xu et al. 2021) to establish a non-autoregressive denoising network structure that includes adaptive normalization layer (adaLN) (Park et al. 2019), transformer encoder-decoder block and self-condition guidance block. The details of our non-autoregressive denoising network can be found in Fig.1(b). The training objective

within our latent diffusion model now reads:

$$L_{LDM} := \mathbb{E}_{\mathcal{E}(x),\mathbf{x}\sim p(x),t}\left[\|\mathcal{E}(x) - \hat{\mathbf{x}}_\theta\left(\mathbf{z}_t, c, t\right)\|_2^2\right], \quad (6)$$

where the denoising backbone $\hat{\mathbf{x}}_\theta$ of our model is a self-guidance transformer structure and c is conditions like look-back window data and covariates. Since the forward process is fixed, z_t can be efficiently obtained from trained $\mathcal{E}$ and we found that training x improves the generative performance compared to ϵ as the denoising target. And finally the samples from $p(x)$ can be directly decoded to the time-space with a trained decoder $\mathcal{D}$.

Self-Conditioning Guidance

Firstly, conditional diffusion model can be simply described as $\hat{\mathbf{x}}_\theta\left(\mathbf{z}_t, c, t\right)$. To train the latent diffusion model in a classifier-free guidance manner, we choose to train an unconditional denoising diffusion model $p_\theta(z)$ parameterized through a score estimator $x_\theta(z_t, t)$ together with the conditional model $x_\theta(z_t, c, t)$. We use a single neural network to parameterize both models, for the unconditional model we only treat the look-back window conditions as the missing value when training the denoising network, i.e. $x_\theta(z_t, t) = x_\theta(z_t, c = \varnothing, t)$. We jointly train these two models simply by setting the look-back window data conditions as $\varnothing$ with some probability p_u which is set as a hyperparameter. When the conditional latent diffusion model is trained, we then perform sampling using a simple but effective linear combination score estimates,

$$\hat{\mathbf{x}}_\theta\left(\mathbf{z}_t, c, t\right) = (1 + w)x_\theta(z_t, c, t) - wx_\theta(z_t, t), \quad (7)$$

where w is the guidance strength, and when $w = 0$, the equation becomes the standard conditional diffusion model. When $w > 0$, the updated gradient of the denoising network will be more offset to the first term and deviate from the latter. Specifically, if we have access to the exact predicted scores $x^*_\theta\left(z_t, c, t\right)$ and $x^*_\theta\left(z_t, t\right)$, then the gradient of this denoising network structure would be $\nabla_{z_t} \log p\left(c|z_t\right) = -\sigma_t\left[x^*_\theta\left(z_t, c, t\right) - x^*_\theta\left(z_t, t\right)\right]$.

Moreover, we introduced a self-condition mechanism that can be seen as to direct condition on previously generated samples of its own during the iterative sampling process. Specifically, our conditional latent diffusion model $\hat{x}_\theta\left(z_t, c, t\right)$ is replaced by the slightly different denoising network $\hat{x}_\theta\left(z_t, \hat{z}_0, c, t\right)$ where the $\hat{z}_0$ is the previously estimated and updated iteratively. In our setting, we concatenate z_t with previously estimated $\hat{z}_0$ which is obtained from the earlier prediction of the denoising network in the sampling chain. During the training phase, with some probability (e.g., 60%), we set $\hat{z}_0 = 0$ which falls back to modeling without Self-Conditioning. Apart from this, we first predict $\hat{z}_0 = \hat{x}_\theta\left(z_t, 0, c, t\right)$ and then use it for self-conditioning. Note that we do not backpropagate through the estimated $\hat{z}_0$.

Latent Diffusion Transformer Network

The complete denoising network is shown in Figure 1(b). For the time series forecasting in a non-autoregressive way, we need to cover how to process time series inputs (look-back window data, target) and the architecture of $\hat{x}_\theta$.

First, we describe how we process time series data as inputs for the training of denoising networks. As defined in Section 4.1, $\hat{\mathbb{E}}_t\left[W^i\right]$ and $\hat{\mathrm{Var}}\left[W^i\right]$ come from the complete time series includes look-back window data and forecasting target. We first normalize the look-back window conditions $X \in \mathbb{R}^{T\times d}$ using $\hat{X} = \frac{X_t^i - \hat{\mathbb{E}}\left[W^i\right]}{\sqrt{\hat{\mathrm{Var}}[W^i]+\epsilon}} \in \mathbb{R}^{T\times d}$ and rescale the latent representation $Z = \mathcal{E}(Y)$ using $\hat{Z} \leftarrow \frac{Z}{\hat{\sigma}} = \frac{\mathcal{E}(X)}{\hat{\sigma}}$ where $\hat{\sigma}^2 = \frac{1}{btm}\sum_{b,t,m}\left(z^{b,t,m} - \hat{\mu}\right)^2$, from the updated results from each training batch, where b is batch size, t is prediction length, m is hidden size and $\hat{\mu} = \frac{1}{btm}\sum_{b,t,m} Z^{b,t,m}$, to obtain the input of the denoising network $\hat{Z} \in \mathbb{R}^{t\times m}$. We then obtain the embedding $\hat{X}^{emb} \in \mathbb{R}^{T\times m}$of the $\hat{X}$ and $\hat{Z}^{emb} \in \mathbb{R}^{t\times m}$ of the latent representation $\hat{Z}$ by an input projection block consisting of two multilayer perceptron layers. In our denoising network, we introduce time embedding of $s^{emb} = [s_{1:\tau}]$ to learn the temporal dependency which is obtained by single MLP layer and Position embedding $p^{emb} = [p_{1:\tau}]$ that is defined in (Vaswani et al. 2017). Also, the diffusion-step embedding $t^{emb} \in \mathbb{R}^{n\times 1}$(n=4m) is encoded as a sinusoidal positional embedding to guide the adaptive layer norm in transformer-based residual layers, which replaces the standard layernorm and defined as

$$\gamma_{i,c} = f_c\left(\boldsymbol{x}\right), \quad \beta_{i,c} = h_c\left(\boldsymbol{x}\right), \quad (8)$$

where $\boldsymbol{x}$ indicates arbitrary vector inputs, $\gamma_{i,c}$ and $\beta_{i,c}$ modulate a neural network's activations $Y_{i,c}$, whose subscripts refer to the i^{th} input's c^{th} feature map, via a feature-wise affine transformation:

$$adaLN\left(\gamma_{i,c}, Y_{i,c}, \beta_{i,c}\right) = \gamma_{i,c}Y_{i,c} + \beta_{i,c}, \quad (9)$$

f_c and h_c can be arbitrary functions such as neural networks, and in our practice, it is easier to refer to f_c and h_c as a single function that outputs single vector $(\gamma \in \mathbb{R}^m, \beta \in \mathbb{R}^m)$. In our residual layers, we learn the dimension-wise scale and shift parameters $\gamma_{i,c}$ and $\beta_{i,c}$ through the diffusion step embedding t^{emb}.

Training The overall structure of our LDT denoising network is shown in Fig.1, and the training objective can refer to Eq.6 in the Method section. Besides, the specific training and inference procedure is shown in the Algorithm 1, 2.

Inference For each time step t in the reverse process, a learned denoising distribution p_θ parameterized by θ generates samples z_{t-1} conditioned on the former noisier samples z_t. After the reverse denoising process reaches T = 0, we round each timestamp of the generated z_0 to its nearest value in the embedding space and obtain the final target by the trained decoder $\mathcal{D}$:

$$z^{t-1} = \hat{\alpha}z_t + \hat{\gamma}x_\theta\left(z^t, t \mid X\right) + \sigma_t\varepsilon, \quad (10)$$

$$Y = \mathcal{D}(z^0), \quad (11)$$

where $\hat{\alpha} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$, $\hat{\gamma} = \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}$, $\sigma_t = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$ and $\epsilon \sim \mathcal{N}\left(0, 1\right)$. Note that in the whole diffusion process of training and sampling, we apply x_θ rather than ϵ_θ. In the experiment, we found that it is difficult to complete the multivariate time series forecasting with ϵ_θ.

Algorithm 1: Training of LDT

Input: Sample $x^0_{1:T}$ (History) and x^0_τ (Target) from training set; Number of diffusion steps K; Encoder $\mathcal{E}$ in pretrained autoencoder. **Output:** Trained denoising function x_θ.

Repeat;
$k \sim \text{Uniform}(\{1, 2, \dots, K\})$;
$\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
Generate latent embedding z^0_τ by $\mathcal{E}(x^0_{1:T}, x^0_\tau)$;
Generate noised latent $\mathbf{z}^k_\tau = \sqrt{\bar{\alpha}_k}\mathbf{z}^0_\tau + \sqrt{1-\bar{\alpha}_k}\epsilon$;
Obtain diffusion step k's embedding p^{emb} using sinusoidal positional embedding.;
$x^0_{1:T} \leftarrow \varnothing$ with probability p_{uncond};
Initialize the self cond $\hat{z}^0_\tau$= zeros_like(z^k_τ);
if $\text{Uniform}(\mathbf{0}, \mathbf{I}) > 0.5$ **then**
    $z^0_{pred} = x_\theta(z^k_\tau, x^0_{1:T}, \hat{z}^0_\tau, k)$;
    z^0_{pred} = Stop_gradient(z^0_{pred});
end if
Use the denoising network to generate denoised sample z^0 by $x_\theta(z^0_{pred}, x^0_{1:T}, z^k_\tau, t)$;
Obtainz^0 by Eq.7, c= $\varnothing$ if $x^0_{1:T} \leftarrow \varnothing$;
Calculate the loss $\mathcal{L}_k(\theta)$ by Eq.6;
Take gradient descent step on $\nabla_\theta L_k(\theta)$;
Until Converged.

Algorithm 2: Generating of LDT

Input: Trained denoising network x_θ, Decoder in pretrained autoencoder $\mathcal{D}$ and Sample $x^0_{1:T}$ (History), guidance strength w. **Output:** Generated corresponding future targets $\hat{x}^0_\tau$.

$z^K_\tau \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
$\hat{z}^0_\tau$= zeros_like(z^K_τ);
$x^{\varnothing}_{1:T}$ = zeros_like($x^0_{1:T}$);
for $k \leftarrow K$ to 1 **do**
    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, if $k > 1$, else $\epsilon = 0$;
    Obtain diffusion step k's embedding p^{emb} using sinusoidal positional embedding.;
    Obtain the self-cond $\hat{z}^0_\tau = x_\theta(z\tau^k, x^{\varnothing}_{1:T}, \hat{z}^0_\tau, p^{emb})$;
    Obtain the target $z^0 = x_\theta(z^k_\tau, x^0_{1:T}, \hat{z}^0_\tau, p^{emb})$;
    Obtain the guidance-based target z^0 with $x^{\varnothing}_{1:T}$ by the Eq.7,
    Estimate z^{k-1}_τ by Eq.10.;
end for
Return $\hat{x}^0_\tau$

Quantitative Experiments

Datasets We extensively evaluate the proposed LDT on five real-world benchmarks, covering the mainstream multivariate time series probabilistic forecasting applications, Energy: Solar (Lai et al. 2018) (137 dimensions) and Electricity (370 dimensions), Traffic (963 dimensions) and Taxi (1214 dimensions), Wikipedia (2000 dimensions). The properties of the datasets used in experiments can refer to the previous works (Rasul et al. 2021; Tashiro et al. 2021) and shown in Appendix C.

Evaluation Metrics For probabilistic estimates, we report both the continuously ranked probability score across summed time series CRPS-sum ((Matheson and Winkler 1976; Jordan, Krüger, and Lerch 2017)) and MSE (mean square error) error metrics, to measure the overall joint distribution pattern fit and fit of joint distribution central tendency, respectively. Due to limited space, the specific form of the metrics is shown in Appendix B.

Baselines We include several baseline methods. For the classical settings and competitive multivariate time series baselines probabilistic models: Gaussian process model(GP) (Roberts et al. 2013), KVAE (Krishnan, Shalit, and Sontag 2017), Vec-LSTM-ind-scaling, GP-scaling, and GP-Copula (Salinas et al. 2019). For the time series diffusion models, including TimeGrad (Rasul et al. 2021), CSDI (Tashiro et al. 2021), SSSD (Alcaraz and Strodthoff 2022), D^3VAE (Li et al. 2022b) as the competitive auto-regressive baselines. Moreover, for the non-autoregressive modeling and flow-based structures, we select TLAE (Nguyen and Quanz 2021) and HMGT (Ding et al. 2020), LSTM-Real-NVP and LSTM-MAF (Rasul et al. 2020) in our work.

Implementation Details In our autoencoder structure of the first stage, both the encoder and the decoder utilized 3 Transformer encoder layers with 4 heads of the attention, and we use one layer Transformer encoder layer with 4 heads of the attention mechanism in the discriminator. The maximum look-back window data is 4 times the predicted target which is the same setting as in (Rasul et al. 2020), with embedding dimension $m \approx [1/4, 1/8]$ data features, diffusion steps T = [50, 100, 200, 300], square-root noise schedule(Li et al. 2022a) and quad variance schedule $\beta_1 = 10 - 4\,\text{till}\,\beta_T = 0.1$. In our denoising network structure, we use a 3-layer transformer structure with 8 attention heads and embedding dim=[32, 64, 128, 256]. Our method is dependent upon the ADAM (Kingma and Ba 2014) optimizer with an initial learning rate of $1e^{-3}$, and the batch size is 64. All experiments are repeated more than five times, implemented in PyTorch (Paszke et al. 2019) and GluonTS (Alexandrov et al. 2020). The specific experimental hyperparameters corresponding to different datasets are shown in Appendix C.

Main Results

Real-World Datasets Results We compare the test time prediction of our LDT to the above baselines with CRPS, CRPS-sum, and MSE. The results for probabilistic forecasting in a multivariate setting are shown in Table 1. Compared with the other generative models, we observe that LDT achieves the state-of-the-art (to the best of our knowledge) CRPS-sum on almost all benchmarks. Notably, our model has shown a significant CRPS-sum reduction in Electricity $16\%(0.025 \rightarrow 0.021)$, in Traffic $14.8\%(0.047 \rightarrow 0.040)$, in Taxi $4\%(0.130 \rightarrow 0.125)$. Also, in terms of MSE metric, we obtain $22\%(2.1e5 \rightarrow 1.6e5)$, $8\%(4.5e-4 \rightarrow 4.1e-4)$ and $4\%(2.2e \rightarrow 2.3e)$ of improvement in the above three datasets.

Uncertainty Estimation The uncertainty can be assessed by estimating the noise of the outcome series when making the prediction. We found that our model showed obvious uncertainty estimation in two types of datasets. As shown in Fig-

	SOLAR		ELECTRICITY		TRAFFIC		TAXI		WIKIPEDIA	
Method	C-S	MSE	C-S	MSE	C-S	MSE	C-S	MSE	C-S	MSE
GP	0.828(.010)	-	0.947(.016)	-	2.198(.774)	-	0.425(.199)	-	0.93(.003)	-
KVAE	0.340(.025)	-	0.051(.019)	-	0.100(.005)	-	-	-	0.095(.012)	-
VLIS	0.391(.017)	9.3e2	0.025(.001)	2.1e5	0.087(.041)	6.3e-4	0.506(.005)	7.3e	0.133(.002)	7.2e7
GP-scaling	0.368(.012)	1.1e3	0.022(.000)	1.8e5	0.079(.000)	5.2e-4	0.183(.395)	2.7e	1.483(1.034)	5.5e7
GP-Copula	0.337(.024)	9.8e2	0.024(.001)	2.4e5	0.078(.002)	6.9e-4	0.208(.183)	3.1e	0.086(.004)	4.0e7
LSRP	0.331(.020)	9.1e2	0.024(.001)	2.5e5	0.078(.001)	6.9e-4	0.175(.001)	2.6e	0.078(.001)	4.7e7
LSTM-MAF	0.315(.032)	9.8e2	0.023(.000)	1.8e5	0.069(.002)	4.9e-4	0.161(.002)	2.4e	0.067(.002)	3.8e7
HMGT	0.327(.013)	9.4e2	0.022(.003)	2.1e5	0.052(.002)	4.4e-4	0.158(.042)	2.4e	0.074(.011)	3.0e7
TLAE	**0.124(.014)**	8.3e2	0.040(.001)	2.7e5	0.069(.005)	5.0e-4	0.130(.010)	2.6e	0.241(.012)	3.8e7
TimeGrad	0.317(.020)	9.9e2	0.025(.001)	2.1e5	0.050(.006)	4.6e-4	0.137(.013)	2.4e	0.064(.003)	3.1e7
CSDI	0.298(.004)	9.4e2	0.029(.002)	2.4e5	0.053(.009)	4.4e-4	-	-	-	-
SSSD	0.275(.004)	**5.4e2**	0.026(.001)	2.3e5	0.047(.002)	4.5e-4	0.133(.006)	2.3e	0.065(.001)	2.99e7
D^3VAE	0.332(.002)	9.2e2	0.030(.000)	2.4e5	0.049(.001)	4.5e-4	0.130(.011)	2.4e	0.069(.004)	3.2e7
LDT	0.253(.002)	7.7e2	**0.021**(.001)	**1.6e5**	**0.040**(.000)	**4.1e-4**	**0.125**(.007)	**2.2e**	**0.061**(.002)	**2.92e7**

Table 1: The Test set CRPS-sum(C-S) and MSE comparison(lower is better) of models from the baselines and our model LDT, with - are runs failed with numerical issues, and (*) indicates the experimental variance. VLIS, LSRP are the abbreviations for the Vec-LSTMind-scaling and LSTM- Real-NVP respectively. - in CSDI is out of memory.

	SOLAR		ELECTRICITY		TRAFFIC		TAXI		WIKIPEDIA	
Strategy	C-S	MSE	C-S	MSE	C-S	MSE	C-S	MSE	C-S	MSE
ϵ_θ	0.528(.006)	1.4e3	0.044(.007)	3.0e5	0.074(.012)	6.4e-4	0.218(.012)	3.2e	0.079(.010)	4.1e7
x_θ	**0.253**(.002)	**7.7e2**	**0.021**(.001)	**1.6e5**	**0.040**(.000)	**4.1e-4**	**0.125**(.007)	**2.2e**	**0.061**(.002)	**2.92e7**

Table 2: The Test set CRPS-sum(C-S) and MSE comparison(lower is better) of models from ϵ_θ strategy and x_θ denoising strategy.

	Solar		Electricity	
H	24	48	24	48
TimeGrad	104.51(.73)	203.16(.90)	302.61(.35)	615.02(.06)
SSSD	80.36(.28)	132.39(.71)	176.23(.82)	295.75(.54)
CSDI	92.23(.53)	147.52(.17)	203.52(.74)	314.23(.76)
D^3VAE	87.53(.29)	153.43(.11)	198.61(.19)	304.76(.82)
LDT	**13.72**(.25)	**14.03**(.37)	**22.13**(.14)	**25.29**(.18)

Table 3: multivariate datasets Solar and Electricity with two different forecasting lengths H=(24, 48).

	Solar		Electricity		Traffic	
Method	C-S	MSE	C-S	MSE	C-S	MSE
LDT-g	0.301(.001)	8.9e2	0.024(.000)	2.1e5	0.050(.003)	4.3e-4
LDT-c	0.264(.004)	8.0e2	0.023(.003)	1.8e5	0.047(.004)	4.5e-4
LDT	**0.253**(.002)	**7.7e2**	**0.021**(.001)	**1.6e5**	**0.040**(.000)	**4.1e-4**

Table 4: The Test set CRPS-sum(C-S) and MSE comparison (lower is better) of models in Ablation studies and our model LDT.

ure 2, for the Solar dataset, although the data had a strong periodicity, there were great differences in both the numerical amplitude periodicity change and the length of periodicity in the dataset that will cause two similar look-back window data to result in two different forecasting targets. Also, in the Taxi dataset which is with high stochasticity, we found that the estimated uncertainty grows rapidly when extreme values are encountered in our model.

Deterministic Estimation In addition to the aforementioned uncertainty estimation approach, our work has revealed that our model exhibits deterministic estimation outcomes when applied to datasets with limited extreme variations, such as Electricity and Traffic. Our one-shot LDT can predict relatively stationary high-dimensional time series more accurately, as shown in Fig. 3. We find that the change of guidance strength in our model will affect the performance of deterministic prediction, which is also shown in Appendix D. We observed that in datasets with deterministic predictions like Electricity and Traffic, a larger guidance w will yield better results, whereas lower guidance w is better in Solar and Taxi. This shows that our model can adapt to different forecasting scenarios by adjusting guid-

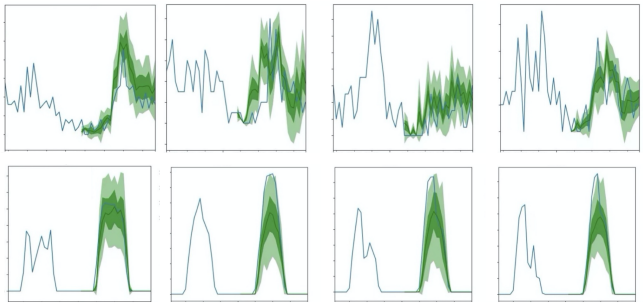

Figure 2: Undeterministic estimation of the prediction of the 8 samples in the Solar and Taxi datasets.

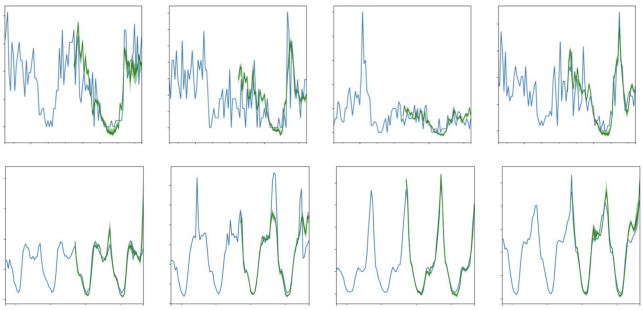

Figure 3: Deterministic estimation of the prediction in the Electricity and Traffic datasets.

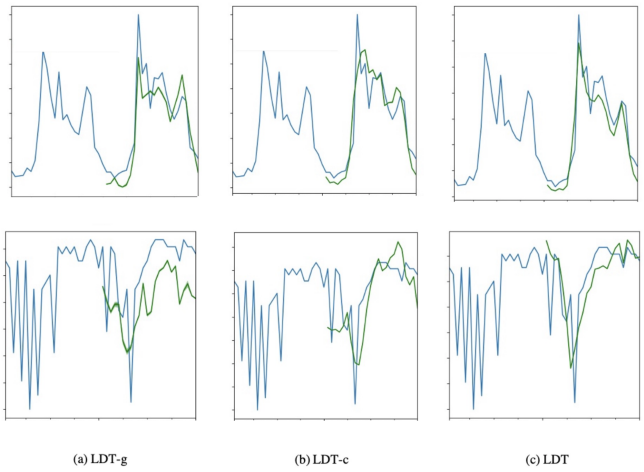

Figure 4: Visualizations on Electricity and Traffic by LDT-g, LDT-c, and the proposed LDT.

ance strength to achieve deterministic and uncertainty predictions for different types of datasets.

Ablation Studies

In this section, we study the effectiveness of the proposed components in our structure. Three representative multivariate datasets are introduced in Table.1: Solar, Electricity, and Traffic, which are non-stationary and high dimensional.

Self-Conditioning Guidance Mechanism In this experiment, we study the effectiveness of the self-conditioning guidance mechanism which is described in section 4.3. We consider the three different settings to verify the effectiveness of our module where LDT-g is without the self-condition to train the denoising network, LDT-c is without the guidance and LDT is proposed in our work. Table 4 shows the results of the two metrics MSE and CRPS-sum with the same guidance strength w =3.0. To verify the role of different parts, we visualized the results of a more complex sample generation in the Electricity dataset. As seen in Figure 4, LDT-g can capture the detailed change pattern of the forecasting targets, but there is a deviation in the accuracy of the numerical value. LDT-c can effectively learn to forecast the interval of the numerical change of the future targets, but the details are not fine enough. And our proposed LDT effectively combines the advantages of these two factors. The guidance factor learns the detailed patterns of the forecasting targets, and the self-condition factor learns to predict the numerical values of the predicted targets.

Predicting x_θ vs. Predicting ϵ_θ In this experiment, we will discuss the different denoising strategies in our work. We compared two different training strategies on five datasets and Table 2 shows our comparative results. We found that the denoising process shows exactly poor performance in ϵ_θ strategy, but we found that the autoregressive diffusion-based method like TimeGrad can use ϵ_θ as the denoising target. We believe that there are two reasons for this result, (1) In the non-autoregressive condition, the target is set to noise, which makes the model ignore the correlation between timestamps. (2) The time series usually contains highly nonlinear noise, which can be easily confused with the noise generated from the diffusion process.

Inference Efficiency In this experiment, we compare the inference efficiency of the proposed LDT with the other time series diffusion model baselines TimeGrad, SSSD, CSDI and D^3VAE. Table 3 shows the inference time on the multivariate datasets Solar and Electricity with two different forecasting lengths (24, 48). In terms of generation efficiency, our one-shot latent structure LDT performs well.

Conclusions

In this study, we introduce a multivariate probabilistic time series forecasting approach that leverages latent space representations. Our method incorporates a self-conditioning guidance mechanism, which combines self-condition bias with condition-based guidance to enhance the denoising process in our latent diffusion model. Furthermore, we develop a one-shot, non-autoregressive Latent Diffusion Transformer (LDT) for high-dimensional multivariate time series prediction. Evaluation of our LDT model on five standard time-series benchmarks sets a new benchmark, outperforming existing generative methods. Ablation studies validate the contribution of each component within our model. We aim to refine denoising structures further for modeling high-dimensional multivariate time series.

Acknowledgements

This research was supported, in part, by the National Research Foundation, Prime Minister's Office, Singapore under its AI Singapore Programme (AISG Award No: AISG-GC-2019-003) and under its NRF Investigatorship Programme (NRFI Award No. NRF-NRFI05-2019-0002). Any opinions, findings conclusions, or recommendations expressed in this material are those of the authors and do not reflect the views of the National Research Foundation, Singapore. Moreover, the authors greatly appreciate the reviewers' suggestions and the editor's encouragement.

References

Alcaraz, J. M. L.; and Strodthoff, N. 2022. Diffusion-based time series imputation and forecasting with structured state space models. *arXiv preprint arXiv:2208.09399*.

Alexandrov, A.; Benidis, K.; Bohlke-Schneider, M.; Flunkert, V.; Gasthaus, J.; Januschowski, T.; Maddix, D. C.; Rangapuram, S.; Salinas, D.; Schulz, J.; et al. 2020. Gluonts: Probabilistic and neural time series modeling in python. *The Journal of Machine Learning Research*, 21(1): 4629–4634.

Cao, D.; Wang, Y.; Duan, J.; Zhang, C.; Zhu, X.; Huang, C.; Tong, Y.; Xu, B.; Bai, J.; Tong, J.; et al. 2020. Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in neural information processing systems*, 33: 17766–17778.

Chen, T.; Zhang, R.; and Hinton, G. 2022. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*.

Ding, Q.; Wu, S.; Sun, H.; Guo, J.; and Guo, J. 2020. Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction. In *IJCAI*, 4640–4646.

Fan, W.; Wang, P.; Wang, D.; Wang, D.; Zhou, Y.; and Fu, Y. 2023. Dish-TS: A General Paradigm for Alleviating Distribution Shift in Time Series Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 7522–7529.

Feng, S.; Miao, C.; Xu, K.; Wu, J.; Wu, P.; Zhang, Y.; and Zhao, P. 2023. Multi-scale attention flow for probabilistic time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*.

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

Ho, J.; and Salimans, T. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.

Huang, R.; Lam, M. W.; Wang, J.; Su, D.; Yu, D.; Ren, Y.; and Zhao, Z. 2022. Fastdiff: A fast conditional diffusion model for high-quality speech synthesis. *arXiv preprint arXiv:2204.09934*.

Jordan, A.; Krüger, F.; and Lerch, S. 2017. Evaluating probabilistic forecasts with scoringRules. *arXiv preprint arXiv:1709.04743*.

Kim, T.; Kim, J.; Tae, Y.; Park, C.; Choi, J.-H.; and Choo, J. 2021. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krishnan, R.; Shalit, U.; and Sontag, D. 2017. Structured inference networks for nonlinear state space models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

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

Li, X.; Thickstun, J.; Gulrajani, I.; Liang, P. S.; and Hashimoto, T. B. 2022a. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35: 4328–4343.

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

Liu, C.; Hoi, S. C.; Zhao, P.; and Sun, J. 2016. Online arima algorithms for time series prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Liu, Y.; Wu, H.; Wang, J.; and Long, M. 2022. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, 35: 9881–9893.

Matheson, J. E.; and Winkler, R. L. 1976. Scoring rules for continuous probability distributions. *Management science*, 22(10): 1087–1096.

Min, E.; Chen, R.; Bian, Y.; Xu, T.; Zhao, K.; Huang, W.; Zhao, P.; Huang, J.; Ananiadou, S.; and Rong, Y. 2022a. Transformer for graphs: An overview from architecture perspective. *arXiv preprint arXiv:2202.08455*.

Min, E.; Rong, Y.; Xu, T.; Bian, Y.; Luo, D.; Lin, K.; Huang, J.; Ananiadou, S.; and Zhao, P. 2022b. Neighbour interaction based click-through rate prediction via graph-masked transformer. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 353–362.

Nguyen, N.; and Quanz, B. 2021. Temporal latent auto-encoder: A method for probabilistic multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 9117–9125.

Park, T.; Liu, M.-Y.; Wang, T.-C.; and Zhu, J.-Y. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2337–2346.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Rao, H.; Hu, X.; Cheng, J.; and Hu, B. 2021. SM-SGE: A self-supervised multi-scale skeleton graph encoding framework for person re-identification. In *Proceedings of the 29th ACM international conference on Multimedia*, 1812–1820.

Rao, H.; and Miao, C. 2022. SimMC: Simple masked contrastive learning of skeleton representations for unsupervised person re-identification. *arXiv preprint arXiv:2204.09826*.

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

Rasul, K.; Sheikh, A.-S.; Schuster, I.; Bergmann, U.; and Vollgraf, R. 2020. Multivariate probabilistic time series forecasting via conditioned normalizing flows. *arXiv preprint arXiv:2002.06103*.

Roberts, S.; Osborne, M.; Ebden, M.; Reece, S.; Gibson, N.; and Aigrain, S. 2013. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984): 20110550.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10684–10695.

Ruan, L.; Ma, Y.; Yang, H.; He, H.; Liu, B.; Fu, J.; Yuan, N. J.; Jin, Q.; and Guo, B. 2023. Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10219–10228.

Salinas, D.; Bohlke-Schneider, M.; Callot, L.; Medico, R.; and Gasthaus, J. 2019. High-dimensional multivariate forecasting with low-rank gaussian copula processes. *Advances in neural information processing systems*, 32.

Sezer, O. B.; Gudelek, M. U.; and Ozbayoglu, A. M. 2020. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, 90: 106181.

Shen, L.; and Kwok, J. 2023. Non-autoregressive Conditional Diffusion Models for Time Series Prediction. *arXiv preprint arXiv:2306.05043*.

Takagi, Y.; and Nishimoto, S. 2023. High-resolution image reconstruction with latent diffusion models from human brain activity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14453–14463.

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

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2016. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Woo, G.; Liu, C.; Sahoo, D.; Kumar, A.; and Hoi, S. 2022. Etsformer: Exponential smoothing transformers for time-series forecasting. *arXiv preprint arXiv:2202.01381*.

Wu, S.; Xiao, X.; Ding, Q.; Zhao, P.; Wei, Y.; and Huang, J. 2020. Adversarial sparse transformer for time series forecasting. *Advances in neural information processing systems*, 33: 17105–17115.

Xu, K.; Zhang, Y.; Ye, D.; Zhao, P.; and Tan, M. 2021. Relation-aware transformer for portfolio policy learning. In *Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence*, 4647–4653.

Yang, L.; Zhang, Z.; Song, Y.; Hong, S.; Xu, R.; Zhao, Y.; Shao, Y.; Zhang, W.; Cui, B.; and Yang, M.-H. 2022. Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796*.

Yi, Y.; Wan, X.; Bian, Y.; Ou-Yang, L.; and Zhao, P. 2023. ETDock: A Novel Equivariant Transformer for Protein-Ligand Docking. *arXiv preprint arXiv:2310.08061*.

Yuan, H.; Yuan, Z.; Tan, C.; Huang, F.; and Huang, S. 2022. Seqdiffuseq: Text diffusion with encoder-decoder transformers. *arXiv preprint arXiv:2212.10325*.