Latent Diffusion Transformer for Probabilistic Time Series Forecasting

Shibo Feng\textsuperscript{1,2,3}, Chunyan Miao\textsuperscript{1,2,3}, Zhong Zhang\textsuperscript{4}, Peilin Zhao\textsuperscript{4}\textsuperscript{*}
\textsuperscript{1}School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore
\textsuperscript{2}Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY), NTU, Singapore
\textsuperscript{3}Webank-NTU Joint Research Institute on Fintech, NTU, Singapore
\textsuperscript{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 Latent Diffusion Transformer (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; Yuan et al. 2022), and multivariate 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,
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 $x \in \mathbb{R}^d \sim p(x)$ and some latent variables $\{z_0, z_1, \cdots, z_T\}$, 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 $\beta_t$ and $\alpha_t := 1 - \beta_t$.

$$q(z_{1:T}|z_0) = \prod_{t=0}^{T} q(z_t|z_{t-1}),$$

where $q(z_t|z_{t-1}) \sim \mathcal{N}(\sqrt{1 - \beta_t}z_{t-1}, \beta_t I)$, (1)

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

$$z_t \sim q(z_t|x) \sim \mathcal{N}(\sqrt{1 - \alpha_t}x, \alpha_t I),$$

$$z_t = \sqrt{\alpha_t}x + \sqrt{1 - \alpha_t}e, e \sim \mathcal{N}(0, I),$$

where $\alpha_t = \prod_{i=1}^{t}(1 - \beta_i) \in (0, 1)$, $z_0 = x$ and $z_T \sim \mathcal{N}(0, I)$ 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 $p_\theta(z_{0:T}) = p(z_T)\prod_{t=1}^{T} p_\theta(z_{t-1}|z_t)$, where $p(z_T) = \mathcal{N}(0, I)$ is known. The $p_\theta(z_{t-1}|z_t)$ can be approximately driven from the following equation, $q(z_{t-1} | z_t, x) = \mathcal{N}(\mu_t(z_t, x), \sigma_t^2 I)$, where $\mu_t(z_t, x)$ has a closed-form solution and $\sigma_t$ is a hyperparameter. To get the $p_\theta(z_{t-1}|z_t)$, we train a denoising network $\theta$ to approximate the $x$ given noisy latent $z_t$ and the timestep $t$.

$$L = \mathbb{E}_{x \sim p(x), t \sim \mathcal{U}\{1, \ldots, T\}, z_t} \left[\|\tilde{x}_\theta(z_t, t) - x\|^2_2\right],$$

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

$$z_{t-1} \sim p_\theta(z_{t-1}|z_t) = q(z_{t-1} | z_t, \tilde{x}_\theta(z_t, t)), (4)$$

For sampling from the trained diffusion model, we utilize the reference distribution from Song et al. (2020) and therefore derive the sampling from $q(z_{t-1} | z_t, x) = \mathcal{N}(\mu_q(z_t, x), \Sigma_q(t)I)$,

$$q(z_{t-1} | z_t, x) \propto \mathcal{N}(z_{t-1}; \alpha_t^2 z_t + \bar{\alpha}_t^2 x, \bar{\beta}_t I),$$

while setting $\Sigma_q(t) = 0$ gives the deterministic DDIM sampler, $\alpha_t^2 = \frac{\sqrt{\sigma_t^2(1 - \alpha_{t-1})}}{1 - \alpha_t^2}$, $\bar{\alpha}_t^2 = \frac{\sqrt{\sigma_t^2(1 - \alpha_{t-1})}}{1 - \alpha_t}$ and $\bar{\beta}_t = \frac{(1 - \alpha_t)(1 - \alpha_{t-1})}{1 - \alpha_t}$. 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 the inference distribution from Song et al. (2020) and this model has surpassed existing state-of-the-art generative models in high-dimensional 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 $\hat{Y} = \frac{Y - \mu}{\sigma}$.
VN([X, Y]). Then, the encoder E encodes \( \hat{Y} \) into a latent representation \( Z = E(\hat{Y}) \), and the decoder D reconstructs the target time series from the latent, giving \( Y = D(Z) = D(E(\hat{Y})) \), where \( Z \in \mathbb{R}^{1 \times m} (m << d) \). 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 \( W^i = [X^i, Y^i] \in \mathbb{R}^{T \times d} \) and \( \tau = T + t \), which are described as \( E[\hat{W}^i]\) = \( \frac{1}{T} \sum_{j=1}^{T} W_{j}^{i} \). Var [\( \hat{W}^{i} \)] = \( \frac{1}{T} \sum_{j=1}^{T} (W_{j}^{i} - E[\hat{W}^{i}])^{2} \) and \( \hat{E}^{n+1}[\hat{W}^{i}] = \frac{1}{n} \text{Var}^{n+1}[\hat{W}^{i}] + \hat{E}^{n}[\hat{W}^{i}] \times (n - 1) \), where \( n \) is the number of batches and the adaptive updated function of variances \( \text{Var}^{n+1}[\hat{W}^{i}] \) is the same as \( \hat{E}^{n+1} \), we normalize the target \( Y^i \) through these updated statistics as \( \hat{Y}^i = \gamma_d \frac{(Y^i - \hat{E}^{n+1}[\hat{W}^{i}])}{\sqrt{\text{Var}^{n+1}[\hat{W}^{i}] + \varepsilon}} + \beta_d \), 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 KLL 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 \( D_{T} \) is optimized to differentiate original target time series from reconstructions \( D(E(\hat{Y})) \). The full objective training loss function \( L \) of our autoencoder reads:

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

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 \( E \) and \( D \) 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 (adLN) (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

![Figure 1](https://example.com/figure1.png)
within our latent diffusion model now reads:

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

(6)

where the denoising backbone $x_\theta$ 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 $x$ and we found that training $x$ improves the generative performance compared to $e$ as the denoising target. And finally the samples from $p(x)$ can be directly decoded to the time-space with a trained decoder $D$.

**Self-Conditioning Guidance**

Firstly, conditional diffusion model can be simply described as $x_\theta(z_t, c, t)$. 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. $x_\theta(z_t, t) = x_\theta(z_t, c = \emptyset, t)$. We jointly train these two models simply by setting the look-back window data conditions as $\emptyset$ with some probability $p_\theta$, 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,

$$x_\theta(z_t, c, t) = (1 + w)x_\theta(z_t, c, t) - wx_\theta(z_t, t),$$

(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 $x_\gamma^*(z_t, c, t)$ and $x_\gamma^*(z_t, t)$, then the gradient of this denoising network structure would be $\nabla_{z_t} \log p(c|z_t) = -\sigma_t [x_\gamma^*(z_t, c, t) - x_\gamma^*(z_t, t)]$.

Moreover, we introduced a self-conditioning 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 $x_\theta(z_t, c, t)$ is replaced by the slightly different denoising network $x_\theta(z_t, \hat{z}_0, c, t)$ 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 $\hat{z}_0 = x_\theta(z_t, 0, c, t)$ 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 Fig.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 $x_\theta$.

First, we describe how we process time series data as inputs for the training of denoising networks. As defined in Section 4.1, $\mathbb{E}_t [W^i]$ and $\mathbb{V}[W^i]$ 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 $X = \frac{X - \bar{X}}{\sqrt{\mathbb{V}[W^i] + \epsilon}} \in \mathbb{R}^{T \times d}$ and rescale the latent representation $Z = \mathcal{E}(Y)$ using $Z \leftarrow \frac{Z}{\hat{\sigma}} = \frac{\mathcal{E}(X)}{\hat{\sigma}}$ where $\hat{\sigma}^2 = \frac{1}{\mathbb{m}_t} \sum_{b=1}^{\mathbb{m}_t} (\hat{z}_b - \hat{\mu})^2$, from the updated results from each training batch, where $b$ is batch size, $t$ is prediction length, $\mathbb{m}$ is hidden size and $\hat{\mu} = \frac{1}{\mathbb{m}_t} \sum_{b=1}^{\mathbb{m}_t} \hat{z}_b$, to obtain the input of the denoising network $\bar{Z} \in \mathbb{R}^{t \times \mathbb{m}}$. We then obtain the embedding $X_{emb} \in \mathbb{R}^{T \times \mathbb{m}}$ of the $X$ and $Z_{emb} \in \mathbb{R}^{t \times \mathbb{m}}$ of the latent representation $Z$ 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 $\tilde{p}_{emb} = [\tilde{p}_{i, t}]$ that is defined in (Vaswani et al. 2017). Also, the diffusion-step embedding $\tilde{p}_{emb} \in \mathbb{R}^{\mathbb{n} \times \mathbb{m}(4m)}$ 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

$$\gamma_{i, c} = f_c(x), \ \beta_{i, c} = h_c(x),$$

(8)

where $x$ 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:

$$\text{adaLN} (\gamma_{i, c}, Y_{i, c}, \beta_{i, c}) = \gamma_{i, c}Y_{i, c} + \beta_{i, c},$$

(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 $\tilde{p}_{emb}$.

**Training**

The overall structure of our LTDN 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 $p_\theta$ parameterized by $\theta$ 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 $D$:

$$z^{t-1} = \hat{\alpha}z_t + \hat{\gamma}x_\theta(z^t, t | X) + \sigma_t \varepsilon,$$

(10)

$$Y = D(z^0),$$

(11)

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

11982
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. 2020) 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), SSD (Alcaraz and Strodthoff 2022), D^4VAE (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 $\beta_T = 10 - 4 \text{till } \beta_T = 0.1$. 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 hyper-parameters 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-
<table>
<thead>
<tr>
<th>Method</th>
<th>C-S MSE</th>
<th>C-S MSE</th>
<th>C-S MSE</th>
<th>C-S MSE</th>
<th>C-S MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>GP</td>
<td>0.828(.101)</td>
<td>-</td>
<td>0.947(.106)</td>
<td>-</td>
<td>2.198(.774)</td>
</tr>
<tr>
<td>KVAE</td>
<td>0.340(.025)</td>
<td>-</td>
<td>0.051(.019)</td>
<td>-</td>
<td>0.100(.005)</td>
</tr>
<tr>
<td>VLIS</td>
<td>0.391(.017)</td>
<td>9.3e2</td>
<td>0.025(.001)</td>
<td>2.1e5</td>
<td>0.087(.041)</td>
</tr>
<tr>
<td>GP-scaling</td>
<td>0.368(.012)</td>
<td>1.1e3</td>
<td>0.022(.000)</td>
<td>1.8e5</td>
<td>0.079(.000)</td>
</tr>
<tr>
<td>GP-Copula</td>
<td>0.337(.024)</td>
<td>9.8e2</td>
<td>0.024(.001)</td>
<td>2.4e5</td>
<td>0.078(.002)</td>
</tr>
<tr>
<td>LSRP</td>
<td>0.331(.020)</td>
<td>9.1e2</td>
<td>0.024(.001)</td>
<td>2.5e5</td>
<td>0.078(.001)</td>
</tr>
<tr>
<td>LSTM-MAF</td>
<td>0.315(.032)</td>
<td>9.8e2</td>
<td>0.023(.000)</td>
<td>1.8e5</td>
<td>0.069(.002)</td>
</tr>
<tr>
<td>LSTM-G</td>
<td>0.327(.013)</td>
<td>9.4e2</td>
<td>0.022(.003)</td>
<td>1.8e5</td>
<td>0.078(.001)</td>
</tr>
<tr>
<td>TLAE</td>
<td>0.124(.014)</td>
<td>8.3e2</td>
<td>0.040(.001)</td>
<td>2.7e5</td>
<td>0.069(.005)</td>
</tr>
<tr>
<td>CSDI</td>
<td>0.298(.004)</td>
<td>9.4e2</td>
<td>0.029(.002)</td>
<td>2.4e5</td>
<td>0.053(.009)</td>
</tr>
<tr>
<td>SSSD</td>
<td>0.275(.004)</td>
<td>9.4e2</td>
<td>0.026(.001)</td>
<td>2.3e5</td>
<td>0.052(.002)</td>
</tr>
<tr>
<td>D^3VAE</td>
<td>0.332(.002)</td>
<td>9.2e2</td>
<td>0.030(.000)</td>
<td>2.4e5</td>
<td>0.049(.001)</td>
</tr>
<tr>
<td>LDT</td>
<td>0.323(.002)</td>
<td>7.7e2</td>
<td>0.021(.001)</td>
<td>1.6e5</td>
<td>0.040(.000)</td>
</tr>
</tbody>
</table>

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-LSTM-ind-scaling and LSTM- Real-NVP respectively. - in CSDI is out of memory.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>SOLAR</th>
<th>ELECTRICITY</th>
<th>TRAFFIC</th>
<th>TAXI</th>
<th>WIKIPEDIA</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \epsilon \theta )</td>
<td>0.528(.006)</td>
<td>1.4e3</td>
<td>0.044(.007)</td>
<td>3.0e5</td>
<td>0.074(.012)</td>
</tr>
<tr>
<td>( x \theta )</td>
<td>0.253(.002)</td>
<td>7.7e2</td>
<td>0.021(.001)</td>
<td>1.6e5</td>
<td>0.040(.000)</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>H</th>
<th>SOLAR</th>
<th>ELECTRICITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>104.51(.73)</td>
<td>203.16(.90)</td>
</tr>
<tr>
<td>48</td>
<td>80.36(.28)</td>
<td>176.23(.82)</td>
</tr>
<tr>
<td>TimeGrad</td>
<td>87.53(.29)</td>
<td>153.43(.11)</td>
</tr>
<tr>
<td>SSSD</td>
<td>92.23(.53)</td>
<td>147.52(.17)</td>
</tr>
<tr>
<td>LSTM-G</td>
<td>113.2(32)</td>
<td>198.61(.19)</td>
</tr>
</tbody>
</table>

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

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-
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 \( x_\theta \) vs. Predicting \( \epsilon_\theta \)
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 \( \epsilon_\theta \) strategy, but we found that the autoregressive diffusion-based method like TimeGrad can use \( \epsilon_\theta \) 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 (NRF 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
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.


