

TimeLDM: Latent Diffusion Model for Unconditional Time Series Generation

Jian Qian¹, Bingyu Xie², Biao Wan¹, Minhao Li¹, Miao Sun³ and Patrick Yin Chiang¹

Abstract—Time series generation is a crucial research topic in the area of decision-making systems, which can be particularly important in domains like autonomous driving, healthcare, and, notably, robotics. Recent approaches focus on learning in the data space to model time series information. However, the data space often contains limited observations and noisy features. In this paper, we propose TimeLDM, a novel latent diffusion model for high-quality time series generation. TimeLDM is composed of a variational autoencoder that encodes time series into an informative and smoothed latent content and a latent diffusion model operating in the latent space to generate latent information. We evaluate the ability of our method to generate synthetic time series with simulated and real-world datasets and benchmark the performance against existing state-of-the-art methods. Qualitatively and quantitatively, we find that the proposed TimeLDM persistently delivers high-quality generated time series. For example, TimeLDM achieves new state-of-the-art results on the simulated benchmarks and an average improvement of 55% in Discriminative score with all benchmarks. Further studies demonstrate that our method yields more robust outcomes across various lengths of time series data generation. Especially, for the Context-FID score and Discriminative score, TimeLDM realizes significant improvements of 80% and 50%, respectively. The code will be released after publication.

I. INTRODUCTION

Time series generation holds a pivotal role across numerous applications, such as robotics [1], [2], autonomous driving [3], [4], and healthcare [5], [6]. Additionally, generating time series can be a valuable approach to solving the complex challenges associated with data privacy concerns. It enables agents to learn a wealth of information without containing any actual sensitive data, providing a safer framework for model training and development.

Numerous studies have used various architectures of deep neural networks for synthetic realistic time series data, including Variational Autoencoder (VAE) based methods [7], [8], Generative Adversarial Network (GAN) based methods [9], [10], [11], [12], and Diffusion-based methods [13], [14]. Typically, Diffusion-based methods have gained plenty of attention from researchers. For instance, DiffTime [13] adopts future mix-up and autoregressive initialization as a condition to generate time information. Diffusion-TS [14] combines the interpretability component, such as trend and multiple seasonality, to model time series using denoising diffusion models. Those methods have emerged as a superior learning architecture in generative modeling to others. However, existing approaches often apply learning models directly in

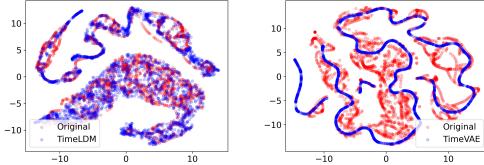


Fig. 1: t-SNE visualization on the stocks dataset, TimeLDM shows better overlap between the generated data and original data than TimeVAE.

the data space, which typically consists of *limited information and noisy features*. Therefore, we are interested in searching for a more flexible framework for time series modeling.

Latent space generation, an efficient alternative model in generative architectures, adopts a pre-trained autoencoder to transfer the generation tasks from the input space to a greater flexible latent domain. In this paper, inspired by the success of the diffusion model on latent space [15], [16], we propose an efficiently synthesized time series method to overcome the above limitations by adopting a *smoother and informative latent presentation*, named **TimeLDM** (**T**ime **L**atent **D**iffusion **M**odel). As shown in Figure 2 (a), We first transform the raw time series data into an embedding space and train the encoder and decoder network for the VAE. The well-studied VAE converts the time series data into the latent space. After that, we apply the latent information as the target of the latent diffusion model (LDM), which is designed with a denoising MLP. During inference, we generate the latent vectors from the LDM and then apply the VAE decoder to synthesize the time series.

We validate the performance of our proposed approach for different benchmarks, including simulated and real-world time series datasets. Qualitatively and quantitatively, we find that the proposed TimeLDM persistently delivers high-quality generated time series (see Figure~5). In Table II and III, the Discriminative scores of TimeLDM consistently outperform current state-of-the-art benchmarks. Furthermore, Table IV demonstrates that TimeLDM presents better performance on different lengths of time series data generation. The main contributions of this paper are summarized as follows:

- We propose TimeLDM, a latent diffusion-based method that leverages the high-fidelity image synthesis ability into unconditional time series generation. To the best of our knowledge, this is the first work to explore the potential of LDM for unconditional time series generation.

¹ Fudan University, {jqian20, pchiang}@fudan.edu.cn.

² Carnegie Mellon University, vxie@andrew.cmu.edu.

³ Nanyang Technological University, miao.sun@ntu.edu.sg.

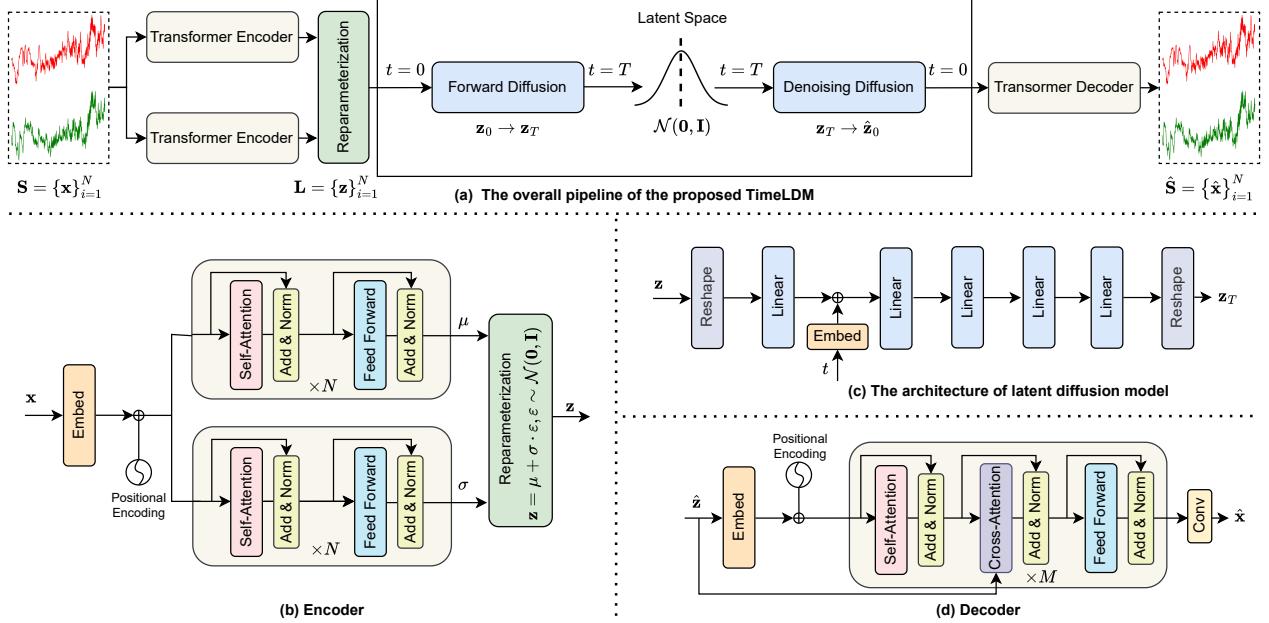


Fig. 2: Structure of our proposed TimeLDM. (a) shows the components of TimeLDM, consisting of the transformer encoder, reparameterization, diffusion process, reverse process, and transformer decoder. (b) shows the details of the transformer encoder and reparameterization. (c) shows the architecture of the latent diffusion model. (d) shows the details of the transformer decoder.

- We evaluate the ability of our method to synthetic time series with simulated and real-world datasets. Empirically, TimeLDM shows better performance than existing generation methods both qualitatively and quantitatively. The ablation study presents the proposed loss function of VAE, which plays a crucial role in improving the capability of our method.
- Furthermore, we evaluate TimeLDM with different lengths of time series data, which presents better performance on the proposed benchmark datasets compared with current state-of-the-art methods.

II. RELATED WORKS

Time Series Generation. Deep generative models have demonstrated their ability to generate high-quality samples across a wide array of fields, where generating time series stands as a particularly challenging endeavor within limited information and noisy features. Early methods based on GANs [17] have been extensively investigated for time series generation. For example, TimeGAN [9] applies an embedding function and supervised loss to the original GAN for capturing the temporal dynamics of data throughout time. Cot-GAN [12] incorporates a specialized loss function based on a regularized Sinkhorn distance, which originates from the principles of causal optimal transport theory. While VAEs [18] also drew the attention of researchers. For instance, TimeVAE [7] implements an interpretable temporal structure and achieves reasonable results on time series synthesis. Recent research [13], [14] has been exploring the use of diffusion models [19] to generate time series, developing

on the successes of forward and reverse processing in other areas such as images [20], video [21], text [22], and audio [23]. Among them, DiffTime [13] approximated the diffusion function based on CSDI [24] where they remove the side information provided as embedding. Diffusion-TS [14] combines the interpretability component, such as trend and multiple seasonality, to model time series using denoising diffusion models.

Generative Modeling in the Latent Space. Although generative models in the data space have achieved significant success, the latest emerging LDMs [15], [16] have demonstrated several advantages, including more compact and disentangled representations, robustness to noise, and greater flexibility in controlling generated styles. LDMs have achieved great success in image generation as they exhibit better scaling properties and expressivity than the vanilla diffusion models in the data space. The success of the LDM in image generation has also inspired their applications in video [25], audio [26], tabular [27], and text [28] domains. In this paper, we explore the application of the LDM for unconditional time series generation tasks.

III. METHOD

TimeLDM, as shown in Figure 2, consists of an encoder-decoder module for VAE, a reparameterization trick for latent information sampling, and the LDM. In this section, we formulate the time series generation task first. Then, we introduce the details of VAE and LDM from network architecture to mathematical formulation. Finally, we summarize the training and sampling procedures.

A. Problem Statement

Let $\mathbf{x}_{1:\tau} = (x_1, \dots, x_\tau) \in \mathbb{R}^{\tau \times d}$ be the original time series, where τ denotes time steps, d is the dimension of observed signals. Given the time series dataset $\mathbf{S} = \{\mathbf{x}\}_{i=1}^N$, the aim of TimeLDM is to learn parameterized generative model $p_\theta(\mathbf{S})$, which can accurately synthesize diverse and realistic time series data $\hat{\mathbf{x}} \in \hat{\mathbf{S}}$ without condition.

B. Time Series Autoencoding

To overcome the weakness of data domain generation, we are focusing on presenting the time series signals $\mathbf{S} = \{\mathbf{x}\}_{i=1}^N$ into an informative and smoothed latent space. The latent representation is $\mathbf{L} = \{\mathbf{z}\}_{i=1}^N$, where $\mathbf{z}_{1:\tau} = (z_1, \dots, z_\tau) \in \mathbb{R}^{\tau \times m}$ denotes the latent feature and m is the dimension of representation. The framework of VAE is shown in Figure 2 (b) and (d). As we can see, it designs with encoder and decoder module, $\text{VAE} = (\mathcal{E}_\phi(\mathbf{x}), \mathcal{D}_\xi(\mathbf{z}))$, where the encoder \mathcal{E}_ϕ learn the latent variable $\mathbf{z} = \mathcal{E}_\phi(\mathbf{x})$ and the decoder \mathcal{D}_ξ decode latent feature \mathbf{z} back to data domain $\hat{\mathbf{x}} = \mathcal{D}_\xi(\mathbf{z})$. Here we adopt β -VAE [29], the coefficient β adaptively balances the reconstruction loss and KL-divergence loss for effective training.

Algorithm 1 Training Algorithm of TimeLDM

```

Input: Time series data  $\mathbf{S} = \{\mathbf{x}\}_{i=1}^N$ 
Output: Encoder  $\mathcal{E}_\phi$ , Decoder  $\mathcal{D}_\xi$ , Denoising Network  $\epsilon_\theta$ 
function TRAIN AUTOENCODER
    Initialize  $\mathcal{E}_\phi, \mathcal{D}_\xi$ 
    while  $\phi, \xi$  have not converged do
        Sample  $\mathbf{x} \in \mathbf{S}$ 
        Get the embedding pattern
        Get the positional pattern
         $\mu, \sigma \leftarrow \mathcal{E}_\phi(\mathbf{x})$ 
         $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
        Reparameterization :  $\mathbf{z} = \mu + \varepsilon \cdot \sigma$ 
         $\hat{\mathbf{x}} \leftarrow \mathcal{D}_\xi(\mathbf{z})$ 
         $\mathcal{L} = \mathcal{L}_{\text{recon}}(\mathbf{x}, \hat{\mathbf{x}}) + \beta \mathcal{L}_{\text{KL}}(\mu, \sigma)$ 
         $\phi, \xi \leftarrow \text{optimizer}(\mathcal{L}; \phi, \xi)$ 
        if  $\ell_{\text{recon}}$  fails to decrease for  $S$  steps then
             $\beta \leftarrow \lambda\beta$ 
        end if
    end while
    return  $\mathcal{E}_\phi, \mathcal{D}_\xi$ 
end function

function TRAIN LATENT DIFFUSION
    Initialize  $\epsilon_\theta$ 
    while  $\theta$  have not converged do
         $\mathbf{z} \sim q_\phi(\mathbf{z} | \mathbf{x})$ 
         $t \sim \mathbf{U}(0, T)$ 
         $\varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$ 
         $\mathbf{z}_t = \mathbf{z}_0 + \varepsilon$ 
         $\ell(\theta) = \|\epsilon_\theta(\mathbf{z}_t, t) - \varepsilon\|_2^2$ 
         $\theta \leftarrow \text{optimizer}(\mathcal{L}_{LDM}; \theta)$ 
    end while
    return  $\epsilon_\theta$ 
end function

 $\mathcal{E}_\phi, \mathcal{D}_\xi \leftarrow \text{TRAIN AUTOENCODER}$ 
Fix parameters  $\phi$  and  $\xi$ 
 $\epsilon_\theta \leftarrow \text{TRAIN LATENT DIFFUSION}$ 
return  $\mathcal{E}_\phi, \mathcal{D}_\xi, \epsilon_\theta$ 

```

VAE’s Encoder. As shown in Figure 2 (b), the VAE’s Encoder \mathcal{E}_ϕ first apply a convolutional neural network to learn an embedding pattern $\mathbf{e} = emb(\mathbf{x}) \in \mathbb{R}^{\tau \times m}$ from the temporal structures $\mathbf{x}_{1:\tau} \in \mathbb{R}^{\tau \times d}$, then a learnable positional encoding $pe \in \mathbb{R}^{\tau \times m}$ equip to the embedding feature for adaptively learning the time series positional information. After that, we train two transformer encoders to learn the mean $\mu \in \mathbb{R}^{\tau \times m}$ and log variance $\sigma \in \mathbb{R}^{\tau \times m}$ from the positional encoding feature $\mathbf{e}_{1:\tau}^{pe} = \mathbf{e}_{1:\tau} + pe$, respectively. Next, we obtain the latent variables $\mathbf{z}_{1:\tau}$ from the reparameterization trick Function 1.

$$\mathbf{z} = \mu + \sigma \cdot \varepsilon, \varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (1)$$

VAE’s Decoder. As shown in Figure 2 (d), the aim of VAE’s Decoder \mathcal{D}_ξ is to minimize the reconstruction error by generating outputs that closely resemble the original time series information. The input to the Decoder \mathcal{D}_ξ consists of latent variables sampled from a typically Gaussian distribution, which is derived from the Encoder using the reparameterization trick. The architect of the VAE decoder incorporates both self-attention and cross-attention mechanisms. The input also respects the learning embedding and positional encoding process. Finally, it generates the realistic samples of time series data $\hat{\mathbf{x}} = \mathcal{D}_\xi(\mathbf{z})$.

Training Loss. The training objective of the VAE consists of the reconstruction loss $\mathcal{L}_{\text{recon}}$ and the KL divergence \mathcal{L}_{KL} . The reconstruction loss is composed of the \mathcal{L}_1 norm, \mathcal{L}_2 norm in the data domain, and the Fast Fourier Transformation (FFT) [30] loss term $\|\mathcal{FFT}(\mathbf{x}), \mathcal{FFT}(\hat{\mathbf{x}})\|$ in the frequency domain [31], which is inspired by HyperTime [32] for accurate time series reconstruction. λ_1, λ_2 , and λ_3 are weights to balance three losses.

$$\mathcal{L}_{\text{recon}} = \lambda_1 \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \lambda_2 \|\mathbf{x} - \hat{\mathbf{x}}\| + \lambda_3 \|\mathcal{FFT}(\mathbf{x}), \mathcal{FFT}(\hat{\mathbf{x}})\| \quad (2)$$

KL divergence loss regularizes the mean and log variance of the latent space. As shown Equation 3, the $q_\phi(\mathbf{z} | \mathbf{x})$ is probabilistic output from the encoder \mathcal{E}_ϕ that represents the approximate posterior of latent variable \mathbf{z} given the input \mathbf{x} ; $\mathcal{N}(\mathbf{z}; \mu, \sigma)$ is the prior on \mathbf{z} . The β is adaptively tuned during training, where $\beta = \lambda\beta, \lambda < 1$. If the $\mathcal{L}_{\text{recon}}$ fails to decrease with defined steps, the β will decrease to encourage the model to pay more attention to the reconstruction term.

$$\mathcal{L}_{\text{KL}} = \beta \text{KL}(q_\phi(\mathbf{z} | \mathbf{x}) \| \mathcal{N}(\mathbf{z}; \mu, \sigma)) \quad (3)$$

Finally, the overall training objective of the VAE is as below. For the adaptive β , we set $\beta_{\max} = 10^{-2}$, $\beta_{\min} = 10^{-5}$, and $\lambda = 0.7$, where the β_{\max} is initial setting, and β_{\min} is the minimum number of the adaptive β .

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{KL}} \quad (4)$$

C. Latent Diffusion Model

After the preparations mentioned above, a trained VAE allows us to access the latent space $\mathbf{L} = \{\mathbf{z}\}_{i=1}^N$. Figure 2(c) presents the neural network architecture of LDM. First, we reshape the sampling representation into one dimension before

Algorithm 2 Sampling Algorithm of TimeLDM

Input: Decoder network \mathcal{D}_ξ , denoising network ϵ_θ
Output: $\hat{\mathbf{x}} \in \hat{\mathbf{S}}$

```

Sample  $\mathbf{z}_T \sim \mathcal{N}(\mathbf{0}, \sigma^2(T)\mathbf{I})$ ,  $t_{\max} = T$ 
for  $i = \max, \dots, 1$  do
     $\nabla_{\mathbf{z}_{t_i}} \log p(\mathbf{z}_{t_i}) = -\epsilon_\theta(\mathbf{z}_{t_i}, t_i)/\sigma(t_i)$ 
    Get  $\mathbf{z}_{t_{i-1}}$  via solving the reverse process
end for
 $\hat{\mathbf{x}} \sim p_\xi(\hat{\mathbf{x}} | \mathbf{z})$ 
return  $\hat{\mathbf{S}}$ 

```

passing through a linear layer. Next, we transform the time step t into sinusoidal embeddings t_{emb} , and added to the $\text{Linear}(\mathbf{z})$. After that, we apply four linear layers to learn the denoising pattern. Finally, we reshape the latent representation back to the input shape. Following [33], we adopt below forward process Equation 5 and reverse process Equation 6 to obtain noising data and learn to reverse back:

$$\mathbf{z}_t = \mathbf{z}_0 + \sigma(t) \cdot \varepsilon, \varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (5)$$

$$d\mathbf{z}_t = -2\dot{\sigma}(t)\sigma(t)\nabla_{\mathbf{z}_t} \log p(\mathbf{z}_t) dt + \sqrt{2\dot{\sigma}(t)\sigma(t)} d\omega_t \quad (6)$$

where $\mathbf{z}_0 = \mathbf{z}$ is the original latent representation from encoder, \mathbf{z}_t is diffused representation with noise level $\sigma(t)$. While for reverse process, $\nabla_{\mathbf{z}_t} \log p_t(\mathbf{z}_t)$ preset score of the \mathbf{z}_t , ω_t is the standard Wiener process. The training object of LDM is:

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{\mathbf{z}_0 \sim p(\mathbf{z}_0)} \mathbb{E}_{t \sim p(t)} \mathbb{E}_{\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \|\epsilon_\theta(\mathbf{z}_t, t) - \varepsilon\|_2^2 \quad (7)$$

where ϵ_θ is the neural network to project \mathbf{z}_t into Gaussian noise. Following [34], we set the noise level $\sigma(t) = t$, and $\nabla_{\mathbf{z}_t} \log p(\mathbf{z}_t) = -\epsilon_\theta(\mathbf{z}_t, t)/\sigma(t)$.

D. Training and Sampling

With the proposed formulation and practical parameterization, we now introduce the training and sampling schemes for TimeLDM. The training process of TimeLDM can be divided into two steps where the first step is to train the VAE and the second step is to study the LDM on the latent space. The Algorithm 1 presents the overall training procedure. For sampling process includes generative diffusion data on the standard latent states, reversing the original time series with a well-learning decoder. The Algorithm 2 shows the overall sampling process.

IV. EXPERIMENTS

We evaluate TimeLDM for time series generation with five different benchmarks, covering simulated and real-world

Parameter	Sines	MuJoCo	Stocks	ETTh	fMRI
dim(x)	5	14	6	7	50
Attention heads	2	2	2	2	2
Attention head dimension	16	16	16	16	16
Encoder layers	1	1	2	2	1
Decoder layers	2	2	3	3	2
Batch size	1024	1024	512	1024	1024
Hidden dimension of LDM	1024	4096	1024	1024	4096

TABLE I: Hyperparameters of VAE and LDM.

Metric	Methods	Sines	MuJoCo
Context-FID Score (Lower the Better)	TimeLDM	0.004±.001	0.006±.000
	Diffusion-TS	0.006±.000	0.013±.001
	TimeGAN	0.101±.014	0.563±.052
	TimeVAE	0.307±.060	0.251±.015
	DiffTime	0.006±.001	0.188±.028
	Cot-GAN	1.337±.068	1.094±.079
Correlational Score (Lower the Better)	TimeLDM	0.013±.005	0.189±.029
	Diffusion-TS	0.015±.004	0.193±.027
	TimeGAN	0.045±.010	0.886±.039
	TimeVAE	0.131±.010	0.388±.041
	DiffTime	0.017±.004	0.218±.031
	Cot-GAN	0.049±.010	1.042±.007
Discriminative Score (Lower the Better)	TimeLDM	0.006±.005	0.004±.004
	Diffusion-TS	0.006±.007	0.008±.002
	TimeGAN	0.011±.008	0.238±.068
	TimeVAE	0.041±.044	0.230±.102
	DiffTime	0.013±.006	0.154±.045
	Cot-GAN	0.254±.137	0.426±.022
Predictive Score (Lower the Better)	TimeLDM	0.093±.000	0.007±.000
	Diffusion-TS	0.093±.000	0.007±.000
	TimeGAN	0.093±.019	0.025±.003
	TimeVAE	0.093±.000	0.012±.002
	DiffTime	0.093±.000	0.010±.001
	Cot-GAN	0.100±.000	0.068±.009
Original		0.094±.001	0.007±.001

TABLE II: Main results on simulated time series datasets. The best result in each case is **bolded**.

datasets. Our framework demonstrates better performance than existing methods, both qualitatively and quantitatively. Further analysis across various lengths of time series data confirms the robustness of TimeLDM.

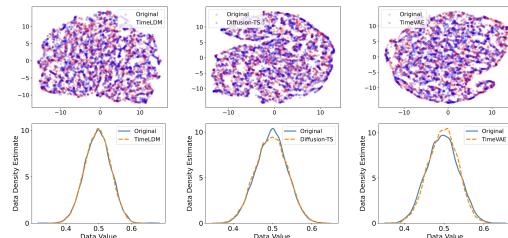


Fig. 3: Visualizations of the simulated MuJoCo dataset, synthesized by TimeLDM, Diffusion-TS and TimeVAE.

A. Experimental Setups

Datasets. We utilize five different datasets to evaluate our model, including: *Sine* is a simulated dataset with 5 features in sinusoidal sequence, and each feature has independent frequencies and phases [9]; *MuJoCo* is the multivariate physics simulation time series data with 14 features [35]; *Stocks* is the Google stock price information from 2004 to 2019, presented daily information and includes 6 features [7]; *ETTh* is built from electricity transformers on 15 minutes basis, including load and oil temperature from July 2016 to July 2018 [36]; *fMRI* serves as a benchmark for causal discovery, featuring simulations that realistically mimic blood-oxygen-level-dependent time series [37].

Baseline. We compare our TimeLDM against five unconditional time series generation methods, including *Diffusion-based* architectures (Diffusion-TS [14] and DiffTime [13]), *GAN-based* models (TimeGAN [9] and Cot-GAN [12]), and *VAE-based* approach (TimeVAE [7]).

Metric	Methods	Stocks	ETTh	fMRI
Context-FID Score (Lower the Better)	TimeLDM	0.032±.007	0.034±.003	0.139±.025
	Diffusion-TS	0.147±.025	0.116±.010	0.105±.006
	TimeGAN	0.103±.013	0.300±.013	1.292±.218
	TimeVAE	0.215±.035	0.805±.186	14.449±.969
	DiffTime	0.236±.074	0.299±.044	0.340±.015
Correlational Score (Lower the Better)	Cot-GAN	0.408±.086	0.980±.071	7.813±.550
	TimeLDM	0.028±.009	0.028±.009	1.036±.025
	Diffusion-TS	0.004±.001	0.049±.008	1.411±.042
	TimeGAN	0.063±.005	0.210±.006	23.502±.039
	TimeVAE	0.095±.008	0.111±.020	17.296±.526
Discriminative Score (Lower the Better)	DiffTime	0.006±.002	0.067±.005	1.501±.048
	Cot-GAN	0.087±.004	0.249±.009	26.824±.449
	TimeLDM	0.017±.011	0.009±.003	0.102±.020
	Diffusion-TS	0.067±.015	0.061±.009	0.167±.023
	TimeGAN	0.102±.021	0.114±.055	0.484±.042
Predictive Score (Lower the Better)	TimeVAE	0.145±.120	0.209±.058	0.476±.044
	DiffTime	0.097±.016	0.100±.007	0.245±.051
	Cot-GAN	0.230±.016	0.325±.099	0.492±.018
	Original	0.036±.001	0.121±.005	0.090±.001

TABLE III: Main results on real-world time series datasets. The best result in each case is **bolded**.

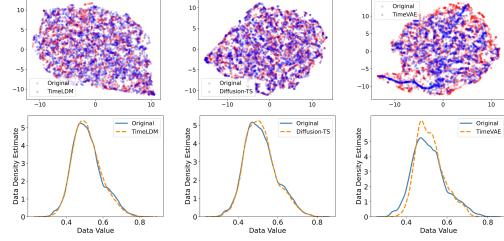


Fig. 4: Visualizations of the real-world ETTh dataset, synthesized by TimeLDM, Diffusion-TS, and TimeVAE.

Setups. In this paper, all the neural networks of TimeLDM are implemented with PyTorch [38] package. For the well-training across all datasets, we tune the limited hyperparameter, as shown in Table I. We proceed with the training in two steps. The first step is to train the β -VAE to obtain latent space information. The second step involves training a diffusion model in the latent space. In the first step, we optimize our network using Adam with default decay rates. The initial learning rate is 10^{-3} . In the second step, We optimize our network using the Adam optimizer with the first and second moment decay rates set to 0.9 and 0.96 respectively. The initial learning rate start is 10^{-4} . The main results are training on an NVIDIA RTX 4080 GPU.

Evaluation Methods. For quantitative analysis, we adopt four different evaluation metrics to evaluate the synthesized time series: (1) *Context-Fréchet Inception Distance (Context-FID) score* assesses the quality of the synthetic time series samples by calculating the difference between representations of time series that fit into the local context [11]; (2) *Correlational score* assesses temporal dependencies by calculating the absolute error between the cross-correlation matrices of real and synthetic data [39]; (3) *Discriminative score* evaluates similarity by employing a classification model to differentiate between original and synthetic data in a

supervised setting [9]; (4) *Predictive score* assesses the utility of synthesized data by training a sequence model post-hoc to predict future temporal vectors using the train-synthesis-and-test-real (TSTR) method [9]. For qualitative analysis, we apply two different data representation methods to evaluate the synthesized time series: (1) *t-SNE* evaluates synthesized time series by projecting both original and synthetic data into a two-dimensional space [40]; (2) *Kernel density estimation* is to draw data distributions to check the alignment between original and synthetic data.

B. Unconditional Time Series Generation

Main Results. We follow the previous setup in TimeGAN [9] to analyze the performance of models on the benchmark datasets mentioned above. The quantitative evaluation results of 24-length time series generation, which represents the most common comparison in existing works, are listed in Table II and III. As can be seen, TimeLDM achieves state-of-the-art results on the simulated benchmarks. Compared with the discriminative score, TimeLDM achieves an average improvement of 55% over Diffusion-TS [14] in all benchmarks. Figure 3 and 4 show the qualitative evaluation results of t-SNE and Kernel density. For the t-SNE analysis, where a greater overlap of blue and red dots shows a better distributional similarity between the generated data and original data. Figure 3 and 4 of t-SNE reveals that our methods have better overlap between the generated data and original data. The Kernel density presents the distribution alignment of original data and synthetic information. Based on the figure, our TimeLDM aligns better with the original data than Diffusion-TS and TimeVAE. We also present the generating time series from the fMRI dataset in Figure 5. Compared to the Diffusion-TS [14] and TimeVAE [7], TimeLDM generates time series that more closely resemble the original training set, while TimeVAE [7] struggles to learn features from the fMRI dataset.

Metric	Methods	ETTh-64	ETTh-128
Context-FID Score (Lower the Better)	TimeLDM	0.067±.008	0.169±.015
	Diffusion-TS	0.631±.058	0.787±.062
	TimeGAN	1.130±.102	1.553±.169
	TimeVAE	0.827±.146	1.062±.134
	DiffTime	1.279±.083	2.554±.318
Correlational Score (Lower the Better)	Cot-GAN	3.008±.277	2.639±.427
	TimeLDM	0.034±.005	0.058±.010
	Diffusion-TS	0.082±.005	0.088±.005
	TimeGAN	0.483±.019	0.188±.006
	TimeVAE	0.067±.006	0.054±.007
Discriminative Score (Lower the Better)	DiffTime	0.094±.010	0.113±.012
	Cot-GAN	0.271±.007	0.176±.006
	TimeLDM	0.030±.053	0.080±.044
	Diffusion-TS	0.106±.048	0.144±.060
	TimeGAN	0.227±.078	0.188±.074
Predictive Score (Lower the Better)	TimeVAE	0.171±.142	0.154±.087
	DiffTime	0.150±.003	0.176±.015
	Cot-GAN	0.296±.348	0.451±.080
	TimeLDM	0.115±.010	0.117±.009
	Diffusion-TS	0.116±.000	0.110±.003
(Lower the Better)	TimeGAN	0.132±.008	0.153±.014
	TimeVAE	0.118±.004	0.113±.005
	DiffTime	0.118±.004	0.120±.008
(Lower the Better)	Cot-GAN	0.135±.003	0.126±.001

TABLE IV: Further evaluation on long-term ETTh dataset generation. The best result in each case is **bolded**.

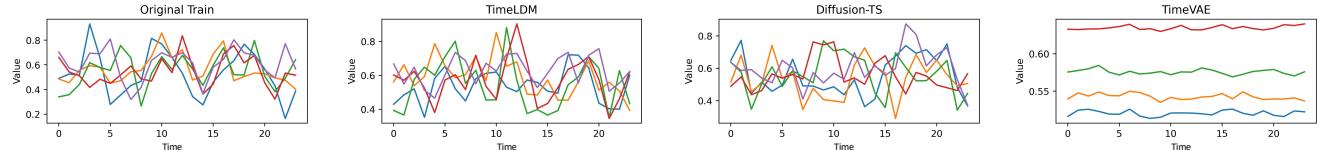


Fig. 5: Examples of generating time series from the fMRI dataset. Our approach yields the closest results to the original training data.

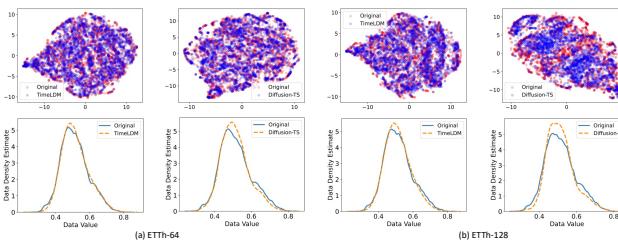


Fig. 6: Visualizations of the real-world ETTh dataset, synthesized by TimeLDM and Diffusion-TS with 64 and 128-time series lengths.

Further Analysis. To further confirm the scalability of our TimeLDM, we evaluate the impact of the different time-series lengths on the generative models for unconditional time series. We examine ETTh data with two different lengths, 64 and 128. For these experiments, we keep all the same hyperparameters with the same metrics to assess the generation quality of different methods. The quantitative results are reported in Table IV. As we can see, our proposed TimeLDM can achieve better performance in most evaluation metrics. Especially on the Context-FID score and Discriminative score, TimeLDM realizes significant state-of-the-art performance with **80%** and **50%** improvement over Diffusion-TS [14]. The qualitative results are depicted in Figure 5. Our TimeLDM shows better alignment than Diffusion-TS with the original data.

C. Ablation Study

In this part, we first assess the adaptive β with the fixed values (β_{\max} , β_{\min}) in the VAE model. Then, we analyze the effectiveness of the reconstruction loss function of VAE. We compare the loss function with its three variants: (1) w/o FFT loss term during training, (2) w/o \mathcal{L}_1 norm term during training, (3) w/o \mathcal{L}_2 norm term during training. The ablation study across all the benchmarks presents the results in Table V and Table VI, respectively.

The effect of adaptive β . We evaluate the adaptive weighting coefficient β in the VAE model. Table V presents the results of adaptive β and constant values (β_{\max} , β_{\min}) on the aforementioned datasets. As can be seen, the difference in performance between the Sines and Stocks benchmarks is insignificant. At the same time, there is a significant performance disparity between the ETTh and MuJoCo benchmarks. The adaptive β improve the effectiveness of TimeLDM, remarkably. This emphasizes the superior performance demonstrated by the adaptive β approach in training the VAE model.

Metric	β	Sines	MuJoCo	Stocks	ETTh	fMRI
Discriminative Score (Lower the Better)	Adaptive	0.006±.005	0.004±.004	0.017±.011	0.009±.003	0.102±.020
	10^{-2}	0.004±.004	0.258±.023	0.015±.015	0.034±.019	0.496±.003
	10^{-5}	0.004±.004	0.007±.008	0.020±.019	0.093±.004	0.216±.021
Predictive Score (Lower the Better)	Adaptive	0.093±.000	0.007±.000	0.037±.000	0.118±.007	0.099±.000
	10^{-2}	0.093±.000	0.013±.001	0.037±.000	0.122±.005	0.100±.000
	10^{-5}	0.093±.000	0.007±.001	0.038±.000	0.123±.005	0.100±.000
(Lower the Better)	Original	0.094±.001	0.007±.001	0.036±.001	0.121±.005	0.090±.001

TABLE V: Ablation study for the adaptive β , which balances the reconstruction loss and KL loss. The best result in each case is **bolded**.

Metric	Methods	Sines	MuJoCo	Stocks	ETTh	fMRI
Discriminative Score (Lower the Better)	TimeLDM	0.006±.005	0.004±.004	0.017±.011	0.009±.003	0.102±.020
	w/o FFT	0.008±.003	0.005±.002	0.022±.028	0.013±.009	0.115±.019
	w/o \mathcal{L}_1	0.007±.005	0.008±.004	0.020±.015	0.014±.014	0.107±.019
Predictive Score (Lower the Better)	w/o \mathcal{L}_2	0.006±.005	0.007±.006	0.014±.010	0.010±.008	0.129±.014
	TimeLDM	0.093±.000	0.007±.000	0.037±.000	0.118±.007	0.099±.000
	w/o FFT	0.093±.000	0.008±.002	0.037±.000	0.118±.006	0.099±.000
(Lower the Better)	w/o \mathcal{L}_1	0.093±.000	0.008±.002	0.037±.000	0.121±.004	0.099±.000
	w/o \mathcal{L}_2	0.093±.000	0.007±.001	0.037±.000	0.121±.006	0.099±.000
	Original	0.094±.001	0.007±.001	0.036±.001	0.121±.005	0.090±.001

TABLE VI: Ablation study for VAE reconstruction loss function. The best result in each case is **bolded**.

The effect of reconstruction loss. We evaluate the effectiveness of reconstruction loss term in the VAE model. Table VI presents the results of three variants on the aforementioned datasets. As can be seen, the performance gap among them for the Predictive score is negligible, the FFT loss term and \mathcal{L}_1 norm term show a crucial role in improving the capability of TimeLDM for the Discriminative score.

V. CONCLUSIONS

In this paper, we propose TimeLDM, a novel latent diffusion model for unconditional time series generation. Particularly, we explore diffusion on the latent space where the original time series is encoded by a variational autoencoder. We evaluate our method on the simulated and real-world datasets and benchmark the performance against existing state-of-the-art methods. Experimental results demonstrate that TimeLDM persistently delivers high-quality generated data both qualitatively and quantitatively. Remarkably, TimeLDM achieves new state-of-the-art results on the simulated benchmarks and an average improvement of 55% in Discriminative score with all benchmarks. Further studies demonstrate that our method yields better performance on different lengths of time series data generation. To the best of our knowledge, this is the first work to explore the potential of the latent diffusion model for unconditional time series generation. We hope that TimeLDM can serve as a robust baseline for generating informative time series tokens for agents learning in the field of physical AI.

REFERENCES

- [1] H. Ichikawa, H. Ito, K. Yamamoto, H. Mori, and T. Ogata, “Multimodal time series learning of robots based on distributed and integrated modalities: Verification with a simulator and actual robots,” in *2023 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2023, pp. 9551–9557.
- [2] Y. Hu, X. Jia, M. Tomizuka, and W. Zhan, “Causal-based time series domain generalization for vehicle intention prediction,” in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 7806–7813.
- [3] Y. Deng, T. Zhang, G. Lou, X. Zheng, J. Jin, and Q.-L. Han, “Deep learning-based autonomous driving systems: A survey of attacks and defenses,” *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 7897–7912, 2021.
- [4] Y. Huang, J. Du, Z. Yang, Z. Zhou, L. Zhang, and H. Chen, “A survey on trajectory-prediction methods for autonomous driving,” *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 3, pp. 652–674, 2022.
- [5] A. Afzal, G. Chrysos, V. Cevher, and M. Shoaran, “Rest: Efficient and accelerated eeg seizure analysis through residual state updates,” *arXiv preprint arXiv:2406.16906*, 2024.
- [6] H. J. Choi, S. Das, S. Peng, R. Bajcsy, and N. Figueiroa, “On the feasibility of eeg-based motor intention detection for real-time robot assistive control,” *arXiv preprint arXiv:2403.08149*, 2024.
- [7] A. Desai, C. Freeman, Z. Wang, and I. Beaver, “Timevae: A variational auto-encoder for multivariate time series generation,” *arXiv preprint arXiv:2111.08095*, 2021.
- [8] V. Fortuin, D. Baranchuk, G. Rätsch, and S. Mandt, “Gp-vae: Deep probabilistic time series imputation,” in *International conference on artificial intelligence and statistics*. PMLR, 2020, pp. 1651–1661.
- [9] J. Yoon, D. Jarrett, and M. Van der Schaar, “Time-series generative adversarial networks,” *Advances in neural information processing systems*, vol. 32, 2019.
- [10] H. Pei, K. Ren, Y. Yang, C. Liu, T. Qin, and D. Li, “Towards generating real-world time series data,” in *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021, pp. 469–478.
- [11] P. Jeha, M. Bohlke-Schneider, P. Mercado, S. Kapoor, R. S. Nirwan, V. Flunkert, J. Gasthaus, and T. Januschowski, “Psa-gan: Progressive self attention gans for synthetic time series,” in *International Conference on Learning Representations*, 2021.
- [12] T. Xu, L. K. Wenliang, M. Munn, and B. Acciaio, “Cot-gan: Generating sequential data via causal optimal transport,” *Advances in neural information processing systems*, vol. 33, pp. 8798–8809, 2020.
- [13] A. Coletta, S. Gopalakrishnan, D. Borrajo, and S. Vytenko, “On the constrained time-series generation problem,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [14] X. Yuan and Y. Qiao, “Diffusion-ts: Interpretable diffusion for general time series generation,” *arXiv preprint arXiv:2403.01742*, 2024.
- [15] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10684–10695.
- [16] Y. Li, Y. Mo, L. Shi, and J. Yan, “Improving generative adversarial networks via adversarial learning in latent space,” *Advances in neural information processing systems*, vol. 35, pp. 8868–8881, 2022.
- [17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in neural information processing systems*, vol. 27, 2014.
- [18] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [19] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.
- [20] S. Gu, D. Chen, J. Bao, F. Wen, B. Zhang, D. Chen, L. Yuan, and B. Guo, “Vector quantized diffusion model for text-to-image synthesis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10696–10706.
- [21] Z. Luo, D. Chen, Y. Zhang, Y. Huang, L. Wang, Y. Shen, D. Zhao, J. Zhou, and T. Tan, “Videofusion: Decomposed diffusion models for high-quality video generation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10209–10218.
- [22] P. Yu, S. Xie, X. Ma, B. Jia, B. Pang, R. Gao, Y. Zhu, S.-C. Zhu, and Y. N. Wu, “Latent diffusion energy-based model for interpretable text modeling,” *arXiv preprint arXiv:2206.05895*, 2022.
- [23] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, “Diffwave: A versatile diffusion model for audio synthesis,” *arXiv preprint arXiv:2009.09761*, 2020.
- [24] Y. Tashiro, J. Song, Y. Song, and S. Ermon, “Csdi: Conditional score-based diffusion models for probabilistic time series imputation,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 24804–24816, 2021.
- [25] A. Blattmann, R. Rombach, H. Ling, T. Dockhorn, S. W. Kim, S. Fidler, and K. Kreis, “Align your latents: High-resolution video synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 22563–22575.
- [26] H. Liu, Z. Chen, Y. Yuan, X. Mei, X. Liu, D. Mandic, W. Wang, and M. D. Plumley, “Audiodlm: Text-to-audio generation with latent diffusion models,” *arXiv preprint arXiv:2301.12503*, 2023.
- [27] H. Zhang, J. Zhang, B. Srinivasan, Z. Shen, X. Qin, C. Faloutsos, H. Rangwala, and G. Karypis, “Mixed-type tabular data synthesis with score-based diffusion in latent space,” *arXiv preprint arXiv:2310.09656*, 2023.
- [28] J. Lovelace, V. Kishore, C. Wan, E. Shekhtman, and K. Q. Weinberger, “Latent diffusion for language generation,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [29] I. Higgins, L. Matthey, A. Pal, C. P. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner, “beta-vae: Learning basic visual concepts with a constrained variational framework.” *ICLR (Poster)*, vol. 3, 2017.
- [30] H. J. Nussbaumer, *The Fast Fourier Transform*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1982, pp. 80–111. [Online]. Available: https://doi.org/10.1007/978-3-642-81897-4_4
- [31] R. N. Bracewell and R. N. Bracewell, *The Fourier transform and its applications*. McGraw-Hill New York, 1986, vol. 31999.
- [32] E. Fons, A. Sztrajman, Y. El-Laham, A. Iosifidis, and S. Vytenko, “Hypertime: Implicit neural representation for time series,” *arXiv preprint arXiv:2208.05836*, 2022.
- [33] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, “Score-based generative modeling through stochastic differential equations,” *arXiv preprint arXiv:2011.13456*, 2020.
- [34] T. Karras, M. Aittala, T. Aila, and S. Laine, “Elucidating the design space of diffusion-based generative models,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 26565–26577, 2022.
- [35] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, N. Heess, and Y. Tassa, “dm_control: Software and tasks for continuous control,” *Software Impacts*, vol. 6, p. 100022, 2020.
- [36] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 11106–11115.
- [37] S. M. Smith, K. L. Miller, G. Salimi-Khorshidi, M. Webster, C. F. Beckmann, T. E. Nichols, J. D. Ramsey, and M. W. Woolrich, “Network modelling methods for fmri,” *Neuroimage*, vol. 54, no. 2, pp. 875–891, 2011.
- [38] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” *Advances in neural information processing systems*, vol. 32, 2019.
- [39] S. Liao, H. Ni, L. Szpruch, M. Wiese, M. Sabate-Vidales, and B. Xiao, “Conditional sig-wasserstein gans for time series generation,” *arXiv preprint arXiv:2006.05421*, 2020.
- [40] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.” *Journal of machine learning research*, vol. 9, no. 11, 2008.