DIFFKANFORMER: DIFFUSION KAN TRANSFORMER FOR GENERAL TIME SERIES ANALYSIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series analysis tasks such as forecasting, imputation, anomaly detection, and classification are crucial for applications spanning climate science, financial domain, retail, and cloud infrastructure. We present DiffKANformer, a conditional diffusion model that integrates Kolmogorov-Arnold Networks (KAN) for feature projection and a Diffusion KAN Transformer architecture for denoising, specifically engineered for time series analysis. DiffKANformer introduces two key innovations: *(i)* a KAN-based projection mechanism in the forward diffusion process that captures complex correlation between features, and *(ii)* a Diffusion KAN Transformer architecture that effectively models complex long-term dependencies through adaptive univariate functions. Our model achieves superior performance across four fundamental time series analysis tasks, significantly outperforming existing prominent models in forecasting (eight datasets), imputation (six datasets), classification (ten datasets) and anomaly detection (five datasets). Comprehensive ablation studies across all tasks validate the utility of each DiffKANformer component, demonstrating the model's robustness in diverse time series challenges.

1 INTRODUCTION

Time series analysis stands at the heart of countless critical applications across diverse domains. From financial market forecasting (Kim, 2003) and economic modeling (Henrique et al., 2019) to transportation planning (Huang et al., 2023), energy management (Dumas et al., 2022), and climate prediction (Li et al., 2024a; Rasul et al., 2022), the ability to understand temporal patterns drives decision-making in our most vital systems. Beyond forecasting, time series analysis enables missing data imputation in domains such as data mining (Friedman, 1962), IT infrastructure monitoring (Qu et al., 2024), satellite telemetry, oil and gas operations, and manufacturing (Zhan et al., 2021). Other critical tasks include anomaly detection for industrial maintenance (Xu et al., 2021) and time series classification for applications such as trajectory-based action recognition (Franceschi et al., 2019).

The field of time series analysis has evolved dramatically from conventional statistical methods such as ARIMA and state-space models to sophisticated deep neural network architectures including NLinear (Zeng et al., 2023), recurrent neural networks (Hewamalage et al., 2021), convolutional neural networks (Yue et al., 2022), and transformers (Vaswani et al., 2017). Given its immense practical value, time series analysis has attracted substantial research attention from deep learning researchers (Wen et al., 2022; Lim & Zohren, 2021). Appendix B has a review of major time series models.

Recently, diffusion models have emerged as a powerful paradigm in generative modeling, achieving unprecedented performance across diverse domains. Their remarkable success in image synthesis (Ho et al., 2020; Dhariwal & Nichol, 2021b), video generation (Harvey et al., 2022; Blattmann et al., 2023), and cross-modal applications (Saharia et al., 2022) has naturally sparked interest in applying their generative power to time series analysis. This has led to several promising conditional diffusion-based frameworks designed specifically for time series forecasting (Tashiro et al., 2021; Li et al., 2024b; Cao et al., 2024; Shen & Kwok, 2023; Shen et al., 2024; Lopez Alcaraz & Strodthoff, 2023). These conditional diffusion models rely on sophisticated conditioning networks and employ various architectural advances to model multi-resolution and nonlinear relations inherent in time series data. For instance, TimeDiff (Shen & Kwok, 2023) incorporates future mixup conditioning for forecasting, TimeGrad (Rasul et al., 2021) combines diffusion with RNN hidden states as condition-

ing information, CSDI (Tashiro et al., 2021) employs self-supervised masking to guide conditioning for non-autoregressive imputation, and CnDiff (Rishi et al., 2025) adapts the Diffusion Transformer (DiT) architecture with specialized conditioning networks for forecasting. However, despite their effectiveness and sophisticated conditioning mechanisms, these methods exhibit two critical limitations: they fail to fully exploit the inherent properties of diffusion models, and none provides a unified solution capable of addressing all aspects of time series analysis.

A deeper examination reveals that current diffusion-based time series models predominantly rely on transformer architectures, which have become popular in computer vision (Dosovitskiy et al., 2020a) and natural language processing (Vaswani et al., 2017). Although extensive research has explored alternatives to attention mechanisms (Liu et al., 2021; 2022d), these variants still use multilayer perceptrons (MLPs) as their core computational building blocks. Surprisingly, relatively few efforts (Shazeer, 2020) have focused on enhancing MLPs themselves, despite their fundamental role.

Addressing this issue is critical for time series modeling. MLPs, while theoretically capable of approximating any function if they have enough neurons (Hornik et al., 1989), encounter practical difficulties with complex temporal patterns. Specifically, MLPs that utilize ReLU-like activations have trouble capturing the periodic functions (Yu et al., 2024; Yang & Wang, 2024) that are common in time series tasks. Additionally, when gradient descent is applied to MLPs, it converges slowly for high-frequency components (Rahaman et al., 2019; Basri et al., 2020), which are widespread in real-world time series data.

To address these architectural limitations, we turn to Kolmogorov-Arnold Networks (KANs), which have recently emerged as a powerful alternative to traditional MLPs. KANs offer superior theoretical parameter efficiency, requiring fewer parameters to model complex functions (Liu et al., 2024c), and demonstrate particular strength in mathematical and symbolic regression tasks (Yu et al., 2024; Liu et al., 2024b). Their key innovation lies in the learnable basis functions employed at each neuron, typically parameterized by B-spline curves (Unser et al., 2002; Gordon & Riesenfeld, 1974). This design enables KANs to approximate intricate functions through spline basis summation, providing enhanced expressiveness for capturing complex temporal dynamics that traditional MLPs struggle to represent.

Building upon these insights, this paper introduces the DiffKANformer architecture that uses a Diffusion Kolmogorov-Arnold Transformer model with a learnable KAN-based data projection in the forward process. These modifications reduce the gap between the true negative log-likelihood and its variational approximation, leading to more effective time series modeling across diverse tasks. The main contributions of this paper are as follows.

1. We introduce the Diffusion KAN transformer framework for time series analysis, which effectively captures long-term dependencies by utilizing adaptive univariate functions. Integrating a KAN-based projection within the forward diffusion process helps to learn complex dependencies between features for enhanced expressiveness in the diffusion process.
2. Experimental results demonstrate that, on average, our model exceeds the performance of other leading time series models across all tasks at the time of writing.
3. This is the first diffusion-based model that shows superior performance in all time series analysis tasks, providing a unified solution for forecasting, imputation, classification, and anomaly detection.

2 METHODOLOGY

Consider a time series represented as $x_{0:H} = \{x_0, x_1, x_2, \ldots, x_H\}$, where each vector x_i consists of one or more variables that can exhibit significant correlations at time i Liu et al. (2023a). This time series can be applied to several tasks. In forecasting, the objective is to predict the future series $x_{H:H+T}$ based on observed data $x_{0:H}$. In the imputation task, certain data points in $x_{0:H}$ might be absent, represented as $\tilde{x}_{0:H} = \{\tilde{x}_0, \tilde{x}_1, \ldots, \tilde{x}_H\}$. The corresponding mask vector $m_{0:H} = \{m_0, m_1, \ldots, m_H\}$ indicates which entries are observed ($m_i = 1$ if observed, 0 if not). The time series imputation task is to estimate these missing values in $\tilde{x}_{0:H}$ using the available data and the mask, generating a complete series $\hat{x}_{0:H}$ that closely mirrors the original $x_{0:H}$. Time series classification involves categorizing multiple series $x_{0:H}$ into defined classes. Anomaly detection in time series data focuses on identifying unusual or unexpected patterns within the series. We define

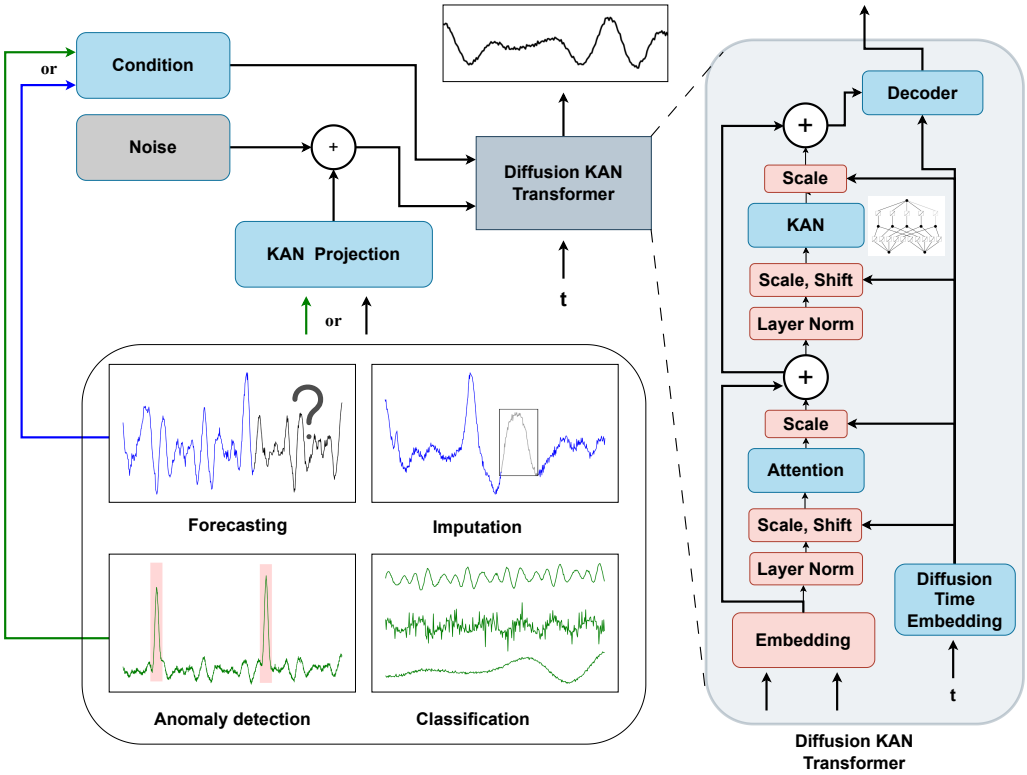


Figure 1: DiffKANformer integrates KAN projections with diffusion KAN transformers for various time series analysis. The architecture handles four key tasks through conditional inputs: forecasting (conditioning on past observations with noise added to future predictions), imputation (conditioning on masked inputs with noise applied only to masked regions), anomaly detection (learning robust representations for anomaly score), and classification (utilizing learned representations that are sent to a classification head). The core Diffusion KAN Transformer processes noised inputs through attention mechanisms, layer normalization, and embedding layers, enabling unified multi-task learning for time series analysis tasks.

output of our model as x_{output}, where x_{output} is the future values to be predicted in the forecasting task, missing values to be predicted in the imputation task and the representation that is fed into the classification head and anomaly detection scores for those tasks.For ease of notation, we replace x_{output} as $\mathbf{x}$.

Current conditional diffusion models for time series, where the condition is past observations for forecasting, masked time series for imputation, and full series for anomaly detection and classification, convert these correlated data distributions into an isotropic Gaussian prior by adjusting data points and progressively adding fixed linear Gaussian noise to generate latent variables (Appendix C.1 and C.2). This method presents two issues: first, it confines diffusion models during the forward process, rendering them fixed and untrainable, and second, the MLP in the denoiser architecture won't be able to capture correlations between highly nonlinear complex relations between time series (Han et al., 2024b).

To address these challenges, we propose DiffKANformer, a nonlinear KAN-based projection framework that learns time-dependent distributions in the latent space during the forward process and employs the Diffusion KAN transformer block as the denoising architecture. Firstly, we discuss our diffusion loss formulation with KAN projection and subsequently delve into the diffusion KAN transformer architecture.

2.1 KAN DIFF FORMULATION AND OBJECTIVE

Let us define the nonlinear time-dependent KAN projection of data for marginal distributions as

$$q_\phi(\mathbf{x}^t|\mathbf{x}) = \mathcal{N}\left(\mathbf{x}^t; \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t), (1-\bar{\alpha}_t)I\right),$$

where $KAN_\phi(\mathbf{x}, t) : \mathbb{R}^{d\times H} \times [0, T] \mapsto \mathbb{R}^{d\times H}$ is a nonlinear KAN (Appendix C.3) parameterized by ϕ that applies a time-dependent projection to the data point $\mathbf{x}$ which helps to learn complex correlations between time series data. We now introduce a learnable condition to the forward process, denoted c [c = ConditionNetwork(input), where input varies by task], inspired by a similar formulation used in image diffusion Pandey et al. (2022). Consequently, the marginal distribution is

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

For $t = T$ along with an appropriately regulated noise schedule α_t, $\bar{\alpha}_T \approx 0$ results in $q_\phi(\mathbf{x}^T|\mathbf{x}, c) \approx \mathcal{N}(c, I)$. Simply put, the Gaussian $\mathcal{N}(c, I)$ serves as our learnable prior distribution and inference requires executing our reverse process on it.

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

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

Consequently, the corresponding loss formulation of DiffKANformer is represented by the following (Appendix D.2).

$$\begin{aligned}\mathcal{L}_{\text{DiffKANformer}} = \mathbb{E}_{q_\phi}\Bigg[&\underbrace{D_{\text{KL}}\left(q_\phi\left(\mathbf{x}^T|\mathbf{x}, c\right) \| p\left(\mathbf{x}^T|c\right)\right)}_{\mathcal{L}_{\text{prior}}} - \underbrace{\sum_{t=1}^{T} \log p_\theta\left(\mathbf{x}|\mathbf{x}^t, c\right)}_{\mathcal{L}_{\text{rec}}} \\ &+ \underbrace{\sum_{t=1}^{T} D_{\text{KL}}\left(q_\phi\left(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c\right) \| p_\theta\left(\mathbf{x}^{t-1}|\mathbf{x}^t, c\right)\right)}_{\mathcal{L}_{\text{diff}}} \Bigg].\end{aligned} \tag{2}$$

The details of each loss term and the detailed derivation are provided in the Appendix D.3.

2.2 DIFFUSION KAN TRANSFORMER

Beyond modifying the diffusion formulation through KAN projection, we fundamentally redesign the denoising architecture to better capture temporal dependencies in time series data. Traditional diffusion models employ convolutional U-Net architectures as denoisers (Ho et al., 2020), with various architectural modifications proposed, including adaptive normalization layers for conditional information injection Dhariwal & Nichol (2021a); Perez et al. (2018). Recently, transformer architectures have gained prominence in diffusion models through the introduction of Diffusion Transformers (DiT) Peebles & Xie (2023); Esser et al. (2024), demonstrating superior scalability and performance in computer vision tasks.

However, standard transformer architectures face critical limitations when applied to complex time series data Han et al. (2024a). At their core, transformers rely on two fundamental components: attention mechanisms and multi-layer perceptrons (MLPs). Although extensive research has explored alternatives to conventional attention mechanisms Liu et al. (2021); Wang et al. (2021), these variants predominantly depend on traditional MLPs as their computational backbone. This dependence becomes problematic for time series modeling, where MLPs face several inherent limitations despite their theoretical universal approximation capabilities Hornik et al. (1989).

Interestingly, comparatively few efforts are made to improve MLPs themselves in denoising architectures (Shazeer, 2020). We developed the Diffusion KAN Transformer (Figure 1), where we change the MLP in DiT with KAN along with the adaptive layer norm (adaLN) block (Peebles & Xie, 2023) where we replace the standard layer in the Diffusion KAN transformer blocks with the adaptive layer norm. Our scale and shift operations learn dimension-wise scale and shift parameters to provide fine-grained control over feature magnitudes and enable more stable training dynamics. Appendix E has more details with equations for our diffusion KAN transformer. The procedure for both training and inference is detailed in Appendix F.

3 EXPERIMENTS

We conducted extensive experiments to evaluate the performance of DiffKANformer on forecasting, imputation, classification, and anomaly detection tasks.

Implementation Details Table 1 provides an overview of the benchmarks. Further information about the datasets is available in Appendix G. Our DiffKANformer model is trained with the Adam optimizer, utilizing a learning rate of 10^{-4}. The training procedure uses a batch size of 64, employs early stopping with a patience of 10, and continues for 100 epochs. Diffusion steps are implemented with a quadratic variance schedule, starting at $\beta_1 = 10^{-4}$ and escalating to $\beta_T = 10^{-1}$. All experiments run on a single Nvidia RTX A5000 GPU with 24GB. Figure 1 illustrates the architecture used in the DiffKANformer model. Additional information about the baselines and hyperparameters can be found in Appendix H and Appendix I.

Table 1: Summary of dataset benchmarks.

Tasks	Benchmarks	Metrics	Series Length
Forecasting	Norpool, Casio, Traffic, Electricity, Weather, Exchange, ETTh1, ETTm1	MSE, MAE	96 to 720
Imputation	ETT (4 subsets), Electricity, Weather	MSE, MAE	96
Classification	UEA (10 subsets)	Accuracy	29 to 1751
Anomaly Detection	SMD, MSL, SMAP, SWaT, PSM	Precision, Recall, F1-Score	100

Main quantitative results are provided in the following. Appendix contains full quantitative results (Appendix M), qualitative results (Appendix N) and statistical significance analysis (Appendix L).

3.1 FORECASTING

Setup We performed experiments on eight real-world time series datasets for daily, weekly, and monthly forecasts Shen et al. (2024); Fan et al. (2022); Zhou et al. (2021); Wu et al. (2021). The length of the look-back window is selected from the set {96, 192, 336, 720, 1440}, determined by performance evaluations on the validation dataset averaged over ten runs with a fixed prediction length (Table 8). Here, the condition is the time series in the look-back window and the prediction is the future time series.

Results Table 2 presents the mean squared errors (MSEs) for multivariate forecasting. We see that our DiffKANformer has a rank of 1 in 4 out of the 8 datasets. Notable improvements are observed, particularly in complex datasets like Weather and ETTh1. In the other four datasets, DiffKANformer consistently holds Rank 2, with a very narrow margin from Rank 1 in each case. On average, the DiffKANformer model achieves a ranking of 1.5, demonstrating its state-of-the-art performance in comparison to other leading models in forecasting tasks. The results with mean absolute error (MAE) metrics are presented in Table 17 of the Appendix M.

3.2 IMPUTATION

Setup For imputation experiments, we use six datasets from the electricity and weather domains as benchmarks, including ETT (Zhou et al., 2021), Electricity [1], and Weather [2], all of which frequently encounter missing data issues. To evaluate model performance under varying levels of missing data, we randomly mask time points with ratios of 12.5%, 25%, 37.5%, 50%. In this context, masked time series is used as the condition for DiffKANformer and the processes of adding noise and denoising are restricted to the masked portions in the diffusion process.

Results Table 3 presents the results for the imputation task with the evaluation metric as the average MSE over various random masking ratios. Our model consistently delivers top-tier performance, recording the lowest error metrics across all datasets, with notably marked improvements on datasets such as ECL, ETTh1, ETTh2, ETTm1, and ETTm2. For example, in ETTm2, our method achieves an error rate of 0.016, significantly exceeding the nearest competitor. Overall, DiffKANformer secures an average rank of 1.5, clearly outperforming robust baselines like Nonstationary, TimesNet,

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

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

Table 2: Multivariate prediction of MSEs on eight real-world time series datasets (subscripts are the rank). CSDI and TiDE run out of memory on Traffic and Electricity. Results of all baselines are from (Shen et al., 2024). Baselines with * are run using TSLib (Wang et al., 2024b; Wu et al., 2022)

Method	NorPool	Caiso	Traffic	ECL	Weather	Exchange	ETTh1	ETTm1	Rank
Ours	0.544 (2)	**0.089** (1)	0.374 (2)	**0.144** (1)	**0.293** (1)	0.016 (2)	**0.401** (1)	0.338 (2)	1.5
CnDiff*	**0.531** (1)	0.094 (2)	0.374 (2)	0.145 (2)	0.296 (2)	0.016 (2)	0.405 (2)	0.340 (3)	2.0
mr-Diff	0.645 (5)	0.127 (5)	0.474 (11)	0.155 (7)	0.296 (2)	0.016 (2)	0.411 (5)	0.340 (3)	5.0
TimeDiff	0.665 (7)	0.136 (11)	0.564 (17)	0.193 (16)	0.311 (4)	0.018 (15)	0.407 (3)	**0.336** (1)	9.2
SSSD	0.872 (23)	0.195 (25)	0.642 (23)	0.255 (27)	0.349 (19)	0.061 (30)	0.726 (33)	0.464 (26)	25.8
CSDI	1.010 (34)	0.253 (34)	–	–	0.356 (22)	0.077 (34)	0.497 (19)	0.529 (32)	29.2
TimeGrad	1.152 (36)	0.258 (35)	1.745 (37)	0.736 (37)	0.392 (27)	0.079 (35)	0.993 (38)	0.874 (36)	35.1
D3VAE	0.745 (13)	0.241 (32)	0.928 (31)	0.286 (30)	0.375 (24)	0.200 (37)	0.504 (21)	0.362 (13)	25.1
CPF	1.613 (39)	0.383 (37)	1.625 (36)	0.793 (38)	1.390 (39)	0.016 (2)	0.730 (34)	0.482 (29)	31.8
PSA-GAN	1.501 (38)	0.510 (39)	1.614 (35)	0.535 (36)	1.220 (37)	0.018 (15)	0.623 (32)	0.537 (33)	33.1
N-Hits	0.716 (11)	0.131 (7)	0.386 (4)	0.152 (5)	0.323 (10)	0.017 (10)	0.498 (20)	0.353 (10)	9.6
FiLM	0.723 (12)	0.179 (21)	0.628 (22)	0.210 (20)	0.327 (12)	0.016 (2)	0.426 (10)	0.347 (7)	13.2
NBeats	0.832 (17)	0.141 (12)	**0.373** (1)	0.269 (28)	1.344 (38)	0.016 (2)	0.586 (29)	0.391 (21)	18.5
Depts	0.662 (6)	0.106 (4)	1.018 (34)	0.319 (32)	0.761 (35)	0.020 (18)	0.579 (27)	0.380 (19)	21.9
TimeXer*	0.715 (10)	0.145 (14)	0.435 (9)	0.153 (6)	0.315 (7)	0.017 (12)	0.424 (9)	0.345 (6)	9.1
Crossformer*	0.833 (19)	0.127 (6)	0.491 (13)	0.149 (4)	0.321 (8)	0.069 (32)	0.452 (15)	0.348 (8)	13.1
PAttn*	0.832 (18)	0.162 (18)	0.492 (14)	0.183 (15)	0.346 (17)	**0.015** (1)	0.427 (11)	0.363 (14)	13.5
MultiPatchFormer*	0.836 (20)	0.155 (17)	0.441 (10)	0.167 (10)	0.347 (18)	0.017 (14)	0.418 (7)	0.369 (16)	14.0
iTransformer*	0.864 (22)	0.176 (20)	0.429 (6)	0.160 (8)	0.354 (21)	0.017 (11)	0.440 (14)	0.384 (20)	15.2
TimesNet*	0.827 (15)	0.146 (15)	0.595 (19)	0.179 (14)	0.337 (15)	0.021 (20)	0.460 (16)	0.376 (18)	16.5
Nonstationary*	0.576 (3)	0.134 (9)	0.671 (27)	0.179 (13)	0.352 (20)	0.023 (22)	0.575 (26)	0.463 (25)	18.1
TiDE*	0.947 (28)	0.236 (31)	–	0.203 (19)	0.459 (29)	0.016 (8)	0.435 (12)	0.404 (22)	24.8
FedFormer	0.873 (24)	0.205 (26)	0.591 (18)	0.238 (25)	0.342 (16)	0.133 (36)	0.541 (24)	0.426 (24)	24.1
PatchTST	0.851 (21)	0.193 (24)	0.831 (30)	0.225 (22)	0.782 (36)	0.047 (28)	0.526 (23)	0.372 (17)	25.1
Scaleformer	0.983 (29)	0.207 (28)	0.618 (21)	0.195 (17)	0.462 (30)	0.036 (25)	0.613 (31)	0.481 (28)	26.1
Autoformer	0.940 (26)	0.226 (30)	0.688 (28)	0.201 (18)	0.360 (23)	0.056 (29)	0.516 (22)	0.565 (34)	26.2
ETSformer*	1.011 (35)	0.180 (22)	0.955 (32)	0.241 (26)	0.473 (31)	0.025 (23)	0.599 (30)	0.501 (31)	28.8
Pyraformer	1.008 (33)	0.273 (36)	0.659 (24)	0.273 (29)	0.394 (28)	0.032 (24)	0.579 (27)	0.493 (30)	28.9
Transformer	1.004 (32)	0.206 (27)	0.671 (26)	0.328 (33)	0.388 (26)	0.062 (31)	0.759 (35)	0.992 (38)	31.0
Informer	0.985 (30)	0.251 (33)	0.664 (25)	0.298 (31)	0.385 (25)	0.073 (33)	0.775 (36)	0.673 (35)	31.0
Reformer*	0.907 (25)	0.166 (19)	0.713 (29)	0.332 (34)	0.682 (34)	0.674 (39)	0.953 (37)	0.874 (37)	31.8
NLinear	0.707 (9)	0.135 (10)	0.430 (7)	0.147 (3)	0.313 (5)	0.019 (17)	0.410 (4)	0.349 (9)	8.0
SCINet	0.613 (4)	0.095 (3)	0.434 (8)	0.171 (12)	0.329 (13)	0.036 (25)	0.465 (17)	0.359 (12)	11.8
MICN*	0.770 (14)	0.133 (8)	0.517 (16)	0.168 (11)	0.322 (9)	0.017 (13)	0.435 (13)	0.357 (11)	11.9
TimeMixer*	0.828 (16)	0.183 (23)	0.499 (15)	0.164 (9)	0.337 (14)	0.016 (9)	0.420 (8)	0.366 (15)	13.6
DLinear	0.670 (8)	0.461 (38)	0.389 (5)	0.215 (21)	0.488 (32)	0.022 (21)	0.415 (6)	0.345 (5)	17.0
LightTS*	0.943 (27)	0.151 (16)	0.601 (20)	0.225 (23)	0.324 (11)	0.021 (19)	0.465 (18)	0.411 (23)	19.6
TSMixer*	0.990 (31)	0.144 (13)	0.476 (12)	0.225 (24)	0.314 (6)	0.037 (27)	0.550 (25)	0.478 (27)	20.6
LSTMa	1.481 (37)	0.217 (29)	0.966 (33)	0.414 (35)	0.662 (33)	0.403 (38)	1.149 (39)	1.030 (39)	35.4

and Crossformer, which hold ranks of 2.2, 3.0, and 7.3, respectively. These findings emphasize our model's superior ability to manage missing data and accurately reconstruct masked time series portions with diverse temporal dynamics. The results with MSE for individual masking rations and MAE as the metric are provided in Tables 18 and 19 in the appendix M.

3.3 CLASSIFICATION

Setup For classification experiments, we perform sequence-level classification using DiffKANformer to learn the representations that are fed into a classification head. The benchmark datasets are chosen to be 10 multivariate datasets from the UEA Time Series Classification Archive (Bagnall et al., 2018), which include tasks such as gesture, action, and audio recognition, as well as medical diagnosis through heartbeat monitoring, among other practical applications. Subsequently, we pre-process these datasets according to the guidelines provided in (Zerveas et al., 2021), noting that different subsets exhibit varying sequence lengths. The time series that must be classified is used as a condition for diffusion, and noise is added to this series, allowing the DiffKANformer to learn representations which are fed to the classification head. Training is performed end-to-end, including the classification head.

Results Table 4 presents accuracy as the metric for the classification task. Our DiffKANformer consistently attains the highest average accuracy of 0.738 along with the best average ranking of

Table 3: In the context of the Imputation task, we randomly took different proportions of time points specifically 12.5%, 25%, 37.5%, and 50% within 96-length time series and evaluated the performance using MSE. Average outcomes across four distinct masking ratios are presented, indicated by subscripts denoting ranks. For detailed outcomes, please refer to 18 and 19, which present the full results and the MAE metrics. Results of all baselines are run using TSLib (Wang et al., 2024b; Wu et al., 2022).

Models	ECL	ETTh1	ETTh2	ETTm1	ETTm2	Weather	Rank
Ours	**0.075** (1)	**0.065** (1)	**0.037** (1)	0.024 (2)	**0.016** (1)	0.032 (3)	1.5
Nonstationary	0.088 (2)	0.075 (2)	0.049 (2)	0.026 (3)	0.021 (2)	0.030 (2)	2.2
TimesNet	0.094 (3)	0.089 (4)	0.051 (3)	0.027 (4)	0.022 (3)	**0.029** (1)	3.0
Crossformer	0.097 (4)	0.148 (12)	0.151 (5)	0.060 (9)	0.079 (5)	0.043 (9)	7.3
LightTS	0.108 (6)	0.159 (14)	0.152 (6)	0.068 (10)	0.076 (4)	0.046 (10)	8.3
iTransformer	0.099 (5)	0.148 (13)	0.139 (4)	0.071 (11)	0.082 (6)	0.051 (12)	8.5
SSSD	0.208 (15)	0.078 (3)	0.410 (17)	**0.023** (1)	0.152 (13)	0.032 (4)	8.8
Reformer	0.162 (11)	0.101 (5)	0.226 (12)	0.033 (5)	0.165 (14)	0.039 (7)	9.0
Pyraformer	0.189 (14)	0.116 (8)	0.201 (10)	0.037 (7)	0.128 (10)	0.034 (5)	9.0
Transformer	0.165 (12)	0.108 (6)	0.256 (14)	0.034 (6)	0.203 (15)	0.036 (6)	9.8
TiDE	0.129 (8)	0.169 (15)	0.163 (8)	0.090 (14)	0.101 (7)	0.052 (13)	10.8
DLinear	0.128 (7)	0.170 (16)	0.162 (7)	0.090 (15)	0.101 (8)	0.052 (14)	11.2
Informer	0.171 (13)	0.114 (7)	0.366 (16)	0.049 (8)	0.245 (16)	0.042 (8)	11.3
ETSformer	0.136 (10)	0.138 (10)	0.234 (13)	0.073 (13)	0.142 (12)	0.048 (11)	11.5
FiLM	0.129 (9)	0.174 (17)	0.169 (9)	0.090 (16)	0.102 (9)	0.057 (15)	12.5
FEDformer	0.215 (16)	0.128 (9)	0.224 (11)	0.072 (12)	0.129 (11)	0.073 (16)	12.5
Autoformer	0.231 (17)	0.147 (11)	0.266 (15)	0.740 (17)	0.862 (17)	0.210 (17)	15.7

1.9, surpassing all other models. In most datasets, DiffKANformer achieves the highest accuracy (marked in bold) or ranks among the top performers. Notably, it achieves the best accuracy on datasets such as FaceDet (0.692), Handwriting (0.382), PEMS-SF (0.898), SpokenArab (0.994), and UWave (0.884). Although competing models such as TimesNet and Heartbeat deliver commendable results, particularly Heartbeat with a notable accuracy of 0.809 on the Heartbeat dataset, DiffKANformer stands out with its superior average ranking and overall accuracy. This indicates that the new method is more effective and robust for the time series classification task compared to existing state-of-the-art models.

Table 4: Classification task with accuracy as the metric. **to be read as former (e.g., In** is Informer). FiLM runs out of memory for the PEMS-SF dataset.

Method	Ethanol	FaceDet	Handwriting	Heartbeat	Japanese	PEMS-SF	SCP1	SCP2	SpokenArab	UWave	Acc	Rank
Ours	0.316 (2)	**0.692** (1)	**0.382** (1)	0.750 (8)	**0.978** (1)	**0.898** (1)	0.914 (2)	**0.578** (1)	**0.994** (1)	**0.884** (1)	0.738	1.9
TimesNet	0.304 (4)	0.674 (5)	0.327 (3)	**0.809** (1)	**0.978** (1)	0.884 (2)	0.914 (2)	0.555 (3)	0.988 (3)	0.881 (2)	0.731	2.6
Trans*	0.277 (10)	0.686 (2)	0.377 (2)	0.780 (2)	**0.978** (1)	0.832 (7)	0.907 (7)	0.533 (9)	0.989 (2)	0.865 (4)	0.722	4.7
In*	0.258 (13)	0.669 (7)	0.317 (5)	0.780 (2)	0.970 (6)	0.826 (8)	0.914 (2)	0.561 (2)	0.986 (5)	0.862 (5)	0.714	5.6
Re*	0.281 (8)	0.681 (3)	0.316 (6)	0.775 (4)	0.970 (6)	0.815 (9)	0.897 (8)	0.533 (9)	0.985 (6)	0.868 (3)	0.712	6.0
iTrans*	0.269 (12)	0.669 (7)	0.271 (8)	0.756 (6)	0.975 (5)	0.872 (3)	**0.921** (1)	0.544 (5)	0.980 (7)	0.859 (6)	0.712	6.1
FED*	0.292 (6)	0.670 (6)	0.235 (10)	0.765 (5)	**0.978** (1)	0.843 (6)	0.597 (13)	0.527 (11)	0.987 (4)	0.587 (13)	0.648	7.5
LightTS	0.304 (4)	0.665 (9)	0.205 (12)	0.751 (7)	0.962 (12)	0.867 (4)	0.911 (6)	0.538 (7)	0.977 (8)	0.831 (9)	0.701	7.8
Pyra*	0.311 (3)	0.555 (14)	0.325 (4)	0.663 (13)	0.970 (6)	0.554 (13)	0.802 (11)	0.538 (7)	0.968 (11)	0.825 (10)	0.651	9.2
PatchTST	0.273 (11)	0.679 (4)	0.265 (9)	0.678 (11)	0.959 (13)	0.861 (5)	0.849 (10)	0.516 (12)	0.966 (12)	0.850 (7)	0.690	9.6
FiLM	0.281 (8)	0.645 (10)	0.136 (14)	0.731 (9)	0.913 (14)	-	0.890 (9)	0.550 (4)	0.977 (8)	0.781 (12)	0.656	9.8
Cross*	**0.395** (1)	0.625 (12)	0.276 (7)	0.546 (14)	0.970 (6)	0.809 (10)	0.754 (12)	0.461 (14)	0.949 (14)	0.834 (8)	0.662	9.9
DLinear	0.254 (14)	0.628 (11)	0.221 (11)	0.678 (11)	0.967 (10)	0.793 (11)	0.914 (2)	0.544 (5)	0.964 (13)	0.821 (11)	0.678	10.1
Auto*	0.285 (7)	0.586 (13)	0.185 (13)	0.721 (10)	0.964 (11)	0.786 (12)	0.552 (14)	0.511 (13)	0.975 (10)	0.518 (14)	0.608	11.8

3.4 Anomaly Detection

Setup Experiments are performed for unsupervised anomaly detection in time series, aiming to pinpoint abnormal time instances. Five popular anomaly detection benchmarks are used: SMD (Su et al., 2019), MSL (Hundman et al., 2018a), SMAP (Hundman et al., 2018b), SWaT (Mathur & Tippenhauer, 2016), and PSM (Abdulaal et al., 2021), which encompass applications in service monitoring, space and earth exploration, and water treatment. Using the pre-processing techniques of Anomaly Transformer (Xu et al., 2021), we segment the data set into successive non-overlapping

portions using a sliding window approach. In prior research, reconstruction has served as a standard method for unsupervised point-wise representation learning, where reconstruction error acts as a natural anomaly indicator. To ensure a fair evaluation, we modify only the base model used for reconstruction with our model, while maintaining the classical reconstruction error as the universal anomaly criterion across all experiments.

Results Table 5 summarizes the results of anomaly detection tasks using F1 score as evaluation metric. DiffKANformer excels as the leading method in most of the datasets. It achieves a notable margin over other leading models for MSL and SMAP datasets. For the other datasets, the F1 score are within top 4 ranks. Overall, DiffKANformer achieves the highest average F1 score of 90.26% with an impressive average rank of 2.4.

Table 5: Anomaly detection task. We calculate the F1-score (presented as a percentage) for each dataset. A higher F1-score signifies enhanced performance. For comprehensive results, refer to Table 20. All Baselines were executed with TSLib (Wang et al., 2024b; Wu et al., 2022).

Model	MSL	PSM	SMAP	SMD	SWaT	Avg F1	Rank
Ours	**90.38** (1)	95.76 (4)	**91.84** (1)	83.79 (4)	<u>90.16</u> (2)	90.39	2.4
TimesNet	81.80 (7)	**97.40** (1)	69.35 (8)	84.59 (3)	**92.48** (1)	85.12	4.0
SSSD	<u>85.55</u> (2)	92.97 (9)	70.47 (5)	**88.88** (1)	88.05 (5)	85.18	4.4
MICN	79.41 (14)	95.93 (3)	70.52 (4)	<u>84.82</u> (2)	89.85 (3)	84.11	5.2
DLinear	81.87 (6)	<u>96.63</u> (2)	68.97 (15)	83.00 (7)	85.73 (7)	83.24	7.4
iTransformer	72.53 (16)	94.32 (5)	69.31 (9)	81.41 (9)	89.24 (4)	81.36	8.6
AnomalyTrans^{+}	83.59 (3)	92.68 (10)	70.15 (6)	78.50 (11)	79.24 (17)	80.83	9.4
FiLM	63.88 (18)	93.70 (7)	69.01 (14)	83.25 (5)	85.72 (8)	79.11	10.4
LightTS	80.99 (11)	93.41 (8)	68.92 (17)	82.70 (8)	81.74 (9)	81.55	10.6
Transformer	80.90 (12)	90.64 (16)	73.26 (3)	71.19 (14)	79.53 (10)	79.10	11.0
KANAD	81.07 (10)	90.53 (17)	69.10 (12)	83.09 (6)	79.30 (11)	80.62	11.2
Reformer	81.10 (8)	92.33 (11)	69.26 (11)	71.04 (16)	79.30 (11)	78.61	11.4
Autoformer	82.59 (4)	88.23 (18)	<u>74.01</u> (2)	71.16 (15)	79.19 (18)	79.04	11.4
Pyraformer	77.19 (15)	92.24 (13)	69.41 (7)	78.97 (10)	79.29 (14)	79.42	11.8
Informer	81.08 (9)	92.32 (12)	69.31 (9)	71.04 (16)	79.29 (14)	78.61	12.0
Crossformer	80.61 (13)	94.30 (6)	69.06 (13)	74.50 (13)	79.28 (16)	79.55	12.2
FEDformer	82.22 (5)	91.40 (14)	68.96 (16)	69.93 (18)	79.30 (11)	78.36	12.8
ETSformer	68.34 (17)	91.09 (15)	68.57 (18)	76.15 (12)	86.76 (6)	78.18	13.6

4 Model Analysis

4.1 KAN Projection

In Table 6, we have shown empirical that DiffKANformer performs better than the ablation run without KAN Projection. To understand what is learned by KAN Projection, we take a trained model and explore the correlation between the features in the learned latent representation space of KAN Projection at different diffusion model timesteps. We observed that there are increasing correlations between the features in learned latent space at different timesteps as shown in Figure 2. Thus, we hypothesize that this increased correlation and time-dependent adaptability perhaps facilitate a more effective diffusion process for time series analysis. Similar observations of correlation in latent space that help diffusion models have been made in image and video diffusion works (Ge et al., 2023). In the space of diffusion models for time series, Tashiro et al. (2021) has noted that learning the correlation between feature and temporal space is necessary for time series imputation tasks.

4.2 KAN vs MLP for time series analysis

Table 7 demonstrate that KANs consistently outperform traditional MLPs across multiple tasks and evaluation metrics. Most notably, KANs show substantial improvements in anomaly detection capabilities, with the DiTKAN model achieving an F1 score of 90.16 compared to 80.80 for the MLP projected DiT model representing a significant 11.6% improvement in detection performance. Sim-

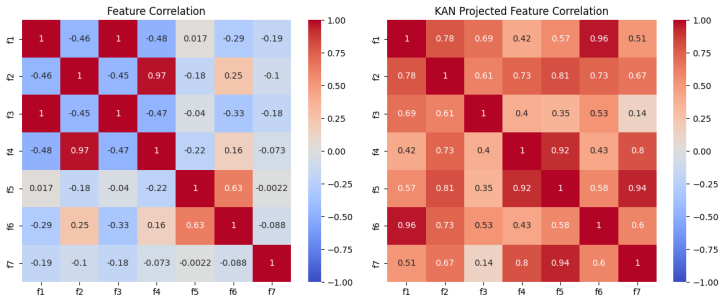


Figure 2: Correlation Matrix of Features for Actual Data and KAN Projection Data in ETTm1 for the Forecasting Task.

Table 6: Comparison of with and without kan projection across a variety of tasks. Various evaluation metrics are utilized, such as MSE for Imputation(25 % mask) and Forecasting, Accuracy for Anomaly Detection and classification.

Task	Dataset	w KAN Projection	w/o KAN Projection
Forecasting	ETTh1	**0.4015**	0.9495
	Weather	**0.2930**	0.6960
Imputation	ETTh1	**0.0520**	0.0649
	ETTm1	**0.0210**	0.0454
	ETTm2	**0.0140**	0.0309
Classification	Handwriting	**0.3822**	0.2115
	PEMS	**0.8984**	0.8359
	SCP2	**0.5781**	0.4922
Anomaly Detection	MSL	**0.9791**	0.9765
	PSM	**0.9783**	0.9725
	SMAP	**0.9793**	0.9797
	SMD	**0.9876**	0.9863
	SwAT	**0.9783**	0.9703

Table 7: Ablation study comparing MLP and KAN architectures across different tasks. Results show forecasting performance (MSE), anomaly detection capability (F1 score), and classification accuracy for DiffKANformer with different neural network components

	Model Architecture Combinations			
Task Dataset, Metric	DiT + MLP	DiT + KAN	DiTKAN + MLP	DiTKAN + KAN
Forecasting Weather, MSE	0.325	0.320	0.308	**0.293**
Imputation ETTm2, MSE	0.035	0.028	0.034	**0.013**
Anomaly Detection SwAT, F1	80.80	88.33	87.35	**90.16**
Classification PEMS-SF, Acc.	0.828	0.838	0.844	**0.898**

ilarly, for classification tasks, KANs deliver superior accuracy with DiTKAN and KAN projection reaching 89.8% accuracy versus 82.8% for the DiT model. While the improvements in forecasting tasks are more modest, KANs still maintain competitive or slightly better performance in MSE metrics for both weather forecasting and ETTm1 imputation tasks. These results suggest that KANs' learnable activation functions and enhanced representational capacity provide meaningful advantages over traditional MLPs for time series analysis.

In addition to these ablation studies, we also evaluated the efficiency and performance of the model through computational time analysis (Appendix K) and statistical significance analysis (Appendix L). The sensitivity analysis for different diffusion hyperparameters is in Appendix J.1, and for various look-back windows is in Appendix J.2. Results show competitive performance and robustness.

5 Conclusion

This paper introduces DiffKANformer, a diffusion-based framework that advances time series analysis through two key innovations: KAN-based projection in the forward diffusion process and the Diffusion KAN Transformer architecture. DiffKANformer achieves state-of-the-art performance in forecasting (8 datasets), imputation (6 datasets), classification (10 datasets), and anomaly detection (5 datasets), representing the first diffusion model to excel across all major time series analysis tasks. The integration of Kolmogorov-Arnold Networks addresses fundamental limitations of traditional MLPs in capturing complex temporal patterns, while our learnable KAN projection enables more effective modeling of complex correlation between features. These architectural advances, validated through extensive ablation studies, establish a new paradigm for time series modeling that bridges the gap between generative modeling and time series analysis.

REFERENCES

Ahmed Abdulaal, Zhuanghua Liu, and Tomer Lancewicki. Practical approach to asynchronous multivariate time series anomaly detection and localization. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 2485–2494, 2021.

Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

Ronen Basri, Meirav Galun, Amnon Geifman, David Jacobs, Yoni Kasten, and Shira Kritchman. Frequency bias in neural networks for input of non-uniform density. In *International conference on machine learning*, pp. 685–694. PMLR, 2020.

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

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten 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*, pp. 22563–22575, 2023.

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

Cristian Challu, Kin G Olivares, Boris N Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. Nhits: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 6989–6997, 2023.

Si-An Chen, Chun-Liang Li, Sercan O Arik, Nathanael Christian Yoder, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecast-ing. *Transactions on Machine Learning Research*, 2023.

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023a.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*, 2023b.

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

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020a.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020b.

Jonathan Dumas, Antoine Wehenkel, Damien Lanaspeze, Bertrand Cornélusse, and Antonio Sutera. A deep generative model for probabilistic energy forecasting in power systems: normalizing flows. *Applied Energy*, 305:117871, 2022.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.

Wei Fan, Shun Zheng, Xiaohan Yi, Wei Cao, Yanjie Fu, Jiang Bian, and Tie-Yan Liu. Depts: Deep expansion learning for periodic time series forecasting. In *International Conference on Learning Representations*, 2022.

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

Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32, 2019.

Milton Friedman. The interpolation of time series by related series. *Journal of the American Statistical Association*, 57(300):729–757, 1962.

Songwei Ge, Seungjun Nah, Guilin Liu, Tyler Poon, Andrew Tao, Bryan Catanzaro, David Jacobs, Jia-Bin Huang, Ming-Yu Liu, and Yogesh Balaji. Preserve your own correlation: A noise prior for video diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22930–22941, 2023.

William J Gordon and Richard F Riesenfeld. B-spline curves and surfaces. In *Computer aided geometric design*, pp. 95–126. Elsevier, 1974.

Xiao Han, Xinfeng Zhang, Yiling Wu, Zhenduo Zhang, and Zhe Wu. Are kans effective for multivariate time series forecasting? *arXiv preprint arXiv:2408.11306*, 2024a.

Xiao Han, Xinfeng Zhang, Yiling Wu, Zhenduo Zhang, and Zhe Wu. Are kans effective for multivariate time series forecasting? *arXiv preprint arXiv:2408.11306*, 2024b.

William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos. *Advances in Neural Information Processing Systems*, 35: 27953–27965, 2022.

Robert Hecht-Nielsen. Kolmogorov's mapping neural network existence theorem. In *Proceedings of the international conference on Neural Networks*, volume 3, pp. 11–14. IEEE press New York, NY, USA, 1987.

Bruno Miranda Henrique, Vinicius Amorim Sobreiro, and Herbert Kimura. Literature review: Machine learning techniques applied to financial market prediction. *Expert Systems with Applications*, 124:226–251, 2019.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

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

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 387–395, 2018a.

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 387–395, 2018b.

Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

Paul Jeha, Michael Bohlke-Schneider, Pedro Mercado, Shubham Kapoor, Rajbir Singh Nirwan, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Psa-gan: Progressive self attention gans for synthetic time series. In *The Tenth International Conference on Learning Representations*, 2022.

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

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

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

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

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.

Lizao Li, Robert Carver, Ignacio Lopez-Gomez, Fei Sha, and John Anderson. Generative emulation of weather forecast ensembles with diffusion models. *Science Advances*, 10(13):eadk4489, 2024a.

Yan Li, Xinjiang Lu, Yaqing Wang, and Dejing Dou. Generative time series forecasting with diffusion, denoise, and disentanglement. *Advances in Neural Information Processing Systems*, 35: 23009–23022, 2022.

Yuxin Li, Wenchao Chen, Xinyue Hu, Bo Chen, Mingyuan Zhou, et al. Transformer-modulated diffusion models for probabilistic multivariate time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024b.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022a.

Qihao Liu, Zhanpeng Zeng, Ju He, Qihang Yu, Xiaohui Shen, and Liang-Chieh Chen. Alleviating distortion in image generation via multi-resolution diffusion models and time-dependent layer normalization. *Advances in Neural Information Processing Systems*, 37:133879–133907, 2024a.

Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *# PLACEHOLDER_PARENT_METADATA_VALUE#*, 2022b.

Yiming Liu, Huadong Guo, Lu Zhang, Dong Liang, Qi Zhu, Xuting Liu, Zhuoran Lv, Xinyu Dou, and Yiting Gou. Research on correlation analysis method of time series features based on dynamic time warping algorithm. *IEEE Geoscience and Remote Sensing Letters*, 20:1–5, 2023a.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in neural information processing systems*, 35: 9881–9893, 2022c.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023b.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022d.

Ziming Liu, Pingchuan Ma, Yixuan Wang, Wojciech Matusik, and Max Tegmark. Kan 2.0: Kolmogorov-arnold networks meet science. *arXiv preprint arXiv:2408.10205*, 2024b.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint arXiv:2404.19756*, 2024c.

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

Aditya P Mathur and Nils Ole Tippenhauer. Swat: A water treatment testbed for research and training on ics security. In *2016 international workshop on cyber-physical systems for smart water networks (CySWater)*, pp. 31–36. IEEE, 2016.

Vahid Naghashi, Mounir Boukadoum, and Abdoulaye Banire Diallo. A multiscale model for multivariate time series forecasting. *Scientific Reports*, 15(1):1565, 2025.

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

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

Kushagra Pandey, Avideep Mukherjee, Piyush Rai, and Abhishek Kumar. Diffusevae: Efficient, controllable and high-fidelity generation from low-dimensional latents. *Transactions on Machine Learning Research*, 2022.

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

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Xinji Qu, Zhuo Liu, Chase Q Wu, Aiqin Hou, Xiaoyan Yin, and Zhulian Chen. Mfgan: multimodal fusion for industrial anomaly detection using attention-based autoencoder and generative adversarial network. *Sensors*, 24(2):637, 2024.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International conference on machine learning*, pp. 5301–5310. PMLR, 2019.

Syama Sundar Rangapuram, Shubham Kapoor, Rajbir Singh Nirwan, Pedro Mercado, Tim Januschowski, Yuyang Wang, and Michael Bohlke-Schneider. Coherent probabilistic forecasting of temporal hierarchies. In *International Conference on Artificial Intelligence and Statistics*, pp. 9362–9376. PMLR, 2023.

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

Kashif Rasul, Young-Jin Park, Max Nihlén Ramström, and Kyung-Min Kim. Vq-ar: Vector quantized autoregressive probabilistic time series forecasting. *arXiv preprint arXiv:2205.15894*, 2022.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pp. 1278–1286. PMLR, 2014.

J Rishi, GVS Mothish, and Deepak Subramani. Conditional diffusion model with nonlinear data transformation for time series forecasting. In *Forty-second International Conference on Machine Learning*, 2025.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

J Taylor Sean and J Taylor. Forecasting at scale. *Am. Stat*, 72(1):37–45, 2018.

Mohammad Amin Shabani, Amir H Abdi, Lili Meng, and Tristan Sylvain. Scaleformer: Iterative multi-scale refining transformers for time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

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

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

Robert H Shumway, David S Stoffer, Robert H Shumway, and David S Stoffer. Arima models. *Time series analysis and its applications: with R examples*, pp. 75–163, 2017.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2828–2837, 2019.

Mingtian Tan, Mike Merrill, Vinayak Gupta, Tim Althoff, and Tom Hartvigsen. Are language models actually useful for time series forecasting? *Advances in Neural Information Processing Systems*, 37:60162–60191, 2024.

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

Michael Unser, Akram Aldroubi, and Murray Eden. B-spline signal processing. i. theory. *IEEE transactions on signal processing*, 41(2):821–833, 2002.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The eleventh international conference on learning representations*, 2023.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024a.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 568–578, 2021.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Mingsheng Long, and Jianmin Wang. Deep time series models: A comprehensive survey and benchmark. 2024b.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. Timexer: Empowering transformers for time series forecasting with exogenous variables. *Advances in Neural Information Processing Systems*, 37:469–498, 2024c.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Etsformer: Exponential smoothing transformers for time-series forecasting. *arXiv preprint arXiv:2202.01381*, 2022.

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

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. *arXiv preprint arXiv:2110.02642*, 2021.

Xingyi Yang and Xinchao Wang. Kolmogorov-arnold transformer. *arXiv preprint arXiv:2409.10594*, 2024.

Runpeng Yu, Weihao Yu, and Xinchao Wang. Kan or mlp: A fairer comparison. *arXiv preprint arXiv:2407.16674*, 2024.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8980–8987, 2022.

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

George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 2114–2124, 2021.

Peng Zhan, Shaokun Wang, Jun Wang, Leigang Qu, Kun Wang, Yupeng Hu, and Xueqing Li. Temporal anomaly detection on iiot-enabled manufacturing. *Journal of Intelligent Manufacturing*, 32 (6):1669–1678, 2021.

T Zhang, Y Zhang, W Cao, J Bian, X Yi, S Zheng, and J Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. arxiv 2022. *arXiv preprint arXiv:2207.01186*, 2022.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*, 2023.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

Quan Zhou, Changhua Pei, Fei Sun, Jing Han, Zhengwei Gao, Dan Pei, Haiming Zhang, Gaogang Xie, and Jianhui Li. Kan-ad: time series anomaly detection with kolmogorov-arnold networks. *arXiv preprint arXiv:2411.00278*, 2024.

Tian Zhou, Ziqing Ma, Qingsong Wen, Liang Sun, Tao Yao, Wotao Yin, Rong Jin, et al. Film: Frequency improved legendre memory model for long-term time series forecasting. *Advances in neural information processing systems*, 35:12677–12690, 2022a.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pp. 27268–27286. PMLR, 2022b.

A LLM USAGE

LLM based grammar check tool has been used to polish language after it was written by the authors.

B RELATED WORK

Classical time series methods include ARIMA (Shumway et al., 2017), Holt-Winter (Hyndman & Athanasopoulos, 2018), and Prophet (Sean & Taylor, 2018). Real-world time series often exhibit a complexity that exceeds these predefined frameworks, which restricts the effectiveness of these conventional approaches in practical applications. Thus several deep learning methods have been proposed for time series modeling.

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

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

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

Diffusion Based Models: TimeGrad (Rasul et al., 2021) pioneered the use of conditional diffusion models, leveraging autoregressive prediction informed by RNN hidden states. CSDI (Tashiro et al., 2021), applies non-autoregressive generation through self-supervised masking but relies on dual transformers, facing boundary inconsistencies and computational challenges with large datasets. SSSD (Lopez Alcaraz & Strodthoff, 2023) aims to reduce CSDI's computational load using structured state-space models, but it continues with masking-based conditioning, preserving boundary issues. TimeDiff (Shen & Kwok, 2023) introduced future mixup and autoregressive initialization within an encoder-decoder scheme for denoising. TMDM (Li et al., 2024b) integrates transformers with a diffusion process for probabilistic multivariate time series forecasting. TimeDiT (Cao et al., 2024) is a foundational model for time series, which employed a transformer type architecture to capture temporal dependencies and employs diffusion processes to generate samples. mr-Diff(Shen et al., 2024) is a recent study utilizing a multi-resolution strategy, incorporating fine-to-coarse patterns as latent variables to aid in denoising. CN-Diff (Rishi et al., 2025) introduced a generative framework that employs novel non-linear transformations and learnable conditions in the forward process for time series forecasting.

C PRELIMINARIES

C.1 DIFFUSION MODEL FORMULATION

Diffusion models are generative frameworks that progressively corrupt data through a forward noising process and then learn to reverse this process to generate realistic samples. For a data point

$x \sim q(x)$, the forward process produces increasingly noisy latent variables $x^0, x^1, \ldots, x^T$, until x^T resembles pure Gaussian noise. The generative model is trained to reverse this chain, starting from noise and reconstructing $x^0 \approx x$.

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

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

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

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

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

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

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

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

Here, the parameter θ is learned by minimizing the loss..

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

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

C.2 Conditional Diffusion models for time series analysis

In time series analysis, the prediction target x_{output} depends on the specific task at hand. For forecasting, the goal is to predict future values $x_{1:H}$ given past observations $x_{-L+1:0} \in \mathbb{R}^{d\times L}$, where L is the lookback window, H is the forecast horizon, and d is the number of variables. For imputation, the aim is to reconstruct missing regions within the input sequence. For classification, the model learns a latent representation of the input time series that is subsequently mapped to class labels through a classification head. For anomaly detection, the objective is to identify irregular or abnormal patterns within the sequence.

When employing conditional diffusion models for these tasks, the generative distribution is expressed as $p_\theta(x^{0:T}_{\text{output}}|c) = p_\theta(x^T_{\text{output}}) \prod_{t=1}^T p_\theta(x^{t-1}_{\text{output}}|x^t_{\text{output}}, c)$, where $x^T_{\text{output}} \sim \mathcal{N}(0, I)$ and c denotes the condition, which varies according to the task (e.g., past observations for forecasting, masked inputs for imputation, the full sequence for classification and anomaly detection) (Rasul et al., 2021; Shen & Kwok, 2023; Shen et al., 2024; Li et al., 2024b).

At each step t, the denoising transition is modeled as $p_\theta(x^{t-1}_{\text{output}}|x^t_{\text{output}}, c) = \mathcal{N}\left(x^{t-1}_{\text{output}}; \mu_\theta(x^t_{\text{output}}, t|c), \sigma_t^2 I\right)$. During inference, generation begins with $\hat{x}^T_{\text{output}} \sim \mathcal{N}(0, I)$. By iteratively applying the denoising step until $t = T$, we obtain the final prediction $\hat{x}_{\text{output}}$, which corresponds to future values in forecasting, completed sequences in imputation, class-discriminative features in classification, or anomaly-indicative representations in detection tasks.

C.3 Kolmogrov-Arnold Networks

The Kolmogorov-Arnold representation theorem Hecht-Nielsen (1987) states that any multivariate continuous function f, defined on a bounded domain, can be expressed as a finite composition of continuous univariate functions and addition. Specifically, for a smooth function $f : [0, 1]^n \to \mathbb{R}$, it can be represented as:

$$f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \Phi_{q,p}(x_p) \right).$$

Here, each function $\Phi_{q,p} : [0, 1] \to \mathbb{R}$ and $\Phi_q : \mathbb{R} \to \mathbb{R}$ are continuous. This means that the $(2d + 1)(d + 1)$ univariate functions Φ_q and $\phi_{q,p}$ are enough for an exact representation of a d-variate function. Unlike traditional multi-layer perceptrons (MLPs), which employ fixed activation functions at each node, (Liu et al., 2024c) define a generalized Kolmogorov-Arnold layer which replaces s each weight parameter with a univariate function. The resulting functional form for deeper KAN can be expressed as : $\text{KAN}(x) = (\Phi_{L-1} \circ \Phi_{L-2} \circ \cdots \circ \Phi_0)(x)$ where Φ_l, $l \in [0, 1, \cdots, L-1]$ is a KAN layer, and the output dimension of each KAN layer can be expressed as: $[n_0, n_1, \cdots, n_{L-1}]$. Therefore, the transform process of j-th feature in l-th layer can be formed as:$\mathbf{x}_{l,j} = \sum_{i=1}^{n_{l-1}} \Phi_{l-1,j,i}(\mathbf{x}_{l-1,i}), \quad j = 1, \cdots, n_l$. where Φ consists of two parts: the spline function and the residual activation function with learnable parameters w_a, w_b: $\Phi(x) = w_a \,\text{SiLU}(x) + w_b \,\text{Spline}(x)$. where $\text{Spline}(\cdot)$ is a linear combination of B-spline functions $\text{Spline}(x) = \sum_i c_i B_i(x)$.

D Formal Derivations

D.1 Forward Posterior

Given:

$$q_\phi(\mathbf{x}^t|\mathbf{x}, c) = \mathcal{N}(\mathbf{x}^t; \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c, (1 - \bar{\alpha}_t)I) \tag{3}$$

From (3), We can write,

$$\begin{aligned}
\mathbf{x}^t &= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \bar{\alpha}_t}\epsilon \\
&= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \alpha_t\bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \alpha_t}\epsilon + \sqrt{\alpha_t - \alpha_t\bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \alpha_t}\epsilon + \sqrt{\alpha_t}\sqrt{1 - \bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \alpha_t}\epsilon + \sqrt{\alpha_t}\left(\mathbf{x}^{t-1} - \sqrt{\bar{\alpha}_{t-1}} KAN_\phi(\mathbf{x}, t-1) - (1 - \sqrt{\bar{\alpha}_{t-1}})c\right) \\
&= \sqrt{\alpha_t}\mathbf{x}^{t-1} + \sqrt{\bar{\alpha}_t}\left(KAN_\phi(\mathbf{x}, t) - KAN_\phi(\mathbf{x}, t-1)\right) + (1 - \sqrt{\alpha_t})c + \sqrt{1 - \alpha_t}\epsilon
\end{aligned} \tag{4}$$

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

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

From (5 and 3), Posterior distribution can be derived as:

$$\begin{aligned}
q_\phi\left(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c\right) &\propto \frac{q_\phi(\mathbf{x}^{t-1}, \mathbf{x}^t, \mathbf{x}, c)}{q_\phi(\mathbf{x}^t, \mathbf{x}, c)} \\
&\propto \frac{q_\phi(\mathbf{x}^t|\mathbf{x}^{t-1}, \mathbf{x}, c)}{q_\phi(\mathbf{x}^t, \mathbf{x}, c)} q_\phi(\mathbf{x}^{t-1}, \mathbf{x}, c) \\
&\propto q_\phi(\mathbf{x}^t|\mathbf{x}^{t-1}, \mathbf{x}, c) q_\phi(\mathbf{x}^{t-1}|\mathbf{x}, c)
\end{aligned} \tag{6}$$

$$\begin{aligned}
&\propto \exp\left(-\frac{1}{2}\left(\frac{\left(\mathbf{x}^t - \sqrt{\alpha_t}\mathbf{x}^{t-1} - \sqrt{\bar{\alpha}_t}(KAN_\phi(\mathbf{x}, t) - KAN_\phi(\mathbf{x}, t-1)) - (1 - \sqrt{\alpha_t})c\right)^2}{1 - \alpha_t}\right.\right. \\
&\left.\left. + \frac{\left(\mathbf{x}^{t-1} - \sqrt{\bar{\alpha}_{t-1}} KAN_\phi(\mathbf{x}, t-1) - (1 - \sqrt{\bar{\alpha}_{t-1}})c\right)^2}{1 - \bar{\alpha}_{t-1}}\right)\right)
\end{aligned} \tag{7}$$

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

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

$$q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c) = \mathcal{N}(\mathbf{x}^{t-1}; \mu_{t-1|t}, \sigma^2_{t-1|t} I) \tag{8}$$

Where,

$$\begin{aligned}
\mu_{t-1|t} = \left(\frac{\sqrt{\alpha_t}\mathbf{x}^t - \sqrt{\bar{\alpha}_t}\sqrt{\alpha_t}(KAN_\phi(\mathbf{x}, t) - KAN_\phi(\mathbf{x}, t-1)) - \sqrt{\alpha_t}(1 - \sqrt{\alpha_t})c}{1 - \alpha_t} + \right. \\
\left. \frac{\sqrt{\bar{\alpha}_{t-1}} KAN_\phi(\mathbf{x}, t-1) + (1 - \sqrt{\bar{\alpha}_{t-1}})c}{1 - \bar{\alpha}_{t-1}}\right) \sigma^2_{t-1|t}
\end{aligned} \tag{9}$$

$$\sigma^2_{t-1|t} = \frac{(1 - \alpha_t)(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \tag{10}$$

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

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

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

Where,

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

D.2 Loss Formulation

Although our intention was not to explicitly make the forward process non-Markovian, our formulation has resulted in a non-Markovian forward process Eq.5. So, we can make use of joint distribution from DDIM (Song et al., 2020) as:

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

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

$$p_\theta(x^{0:T}) = p(x^T)\prod_{t=1}^{T} p_\theta(x^{t-1}|x^t)p_\theta(x^0|x^t), \tag{12}$$

$$\text{where } p(x^T) = \mathcal{N}(x^T; 0, I). \tag{13}$$

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

Our definition of trainable reverse generative process (13) is a result of introducing a KAN projection in the marginal distribution, which makes it challenging to learn from the reverse Markovian process. However, even without non-linearity, the resulting loss will be similar to that of DDPM,

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

$$\begin{aligned}\mathbb{E}_{q_\phi}\Bigg[& \underbrace{D_{\mathrm{KL}}\left(q_\phi\left(\mathbf{x}^T|\mathbf{x},c\right)\|p\left(\mathbf{x}^T|c\right)\right)}_{\mathcal{L}_{\text{prior}}} - \underbrace{\sum_{t=1}^{T}\log p_\theta\left(\mathbf{x}|\mathbf{x}^t,c\right)}_{\mathcal{L}_{\text{rec}}} \\ & + \underbrace{\sum_{t=1}^{T} D_{\mathrm{KL}}\left(q_\phi\left(\mathbf{x}^{t-1}|\mathbf{x}^t,\mathbf{x},c\right)\|p_\theta\left(\mathbf{x}^{t-1}|\mathbf{x}^t,c\right)\right)}_{\mathcal{L}_{\text{diff}}}\Bigg].\end{aligned} \tag{14}$$

D.3 Variational Objective

To calculate the diffusion term $\mathcal{L}_{diff}$ of the objective (2), we need to compute the KL divergence between the forward posterior distribution $q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t,\mathbf{x},c)$ and the reverse distribution $p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t,c)$.

Since we use parameterization $p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t, c) = q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t, \hat{\mathbf{x}}_\theta(x^t, t), c)$, both of these distributions are normal distributions with the same variance, so we can evaluate the KL divergence between them analytically as follows:

$$D_{KL}\left(q_\phi(\mathbf{x}^{t-1}|\mathbf{x}^t, \mathbf{x}, c)\|p_\theta(\mathbf{x}^{t-1}|\mathbf{x}^t, c)\right) =$$

$$\frac{1}{2\sigma^2_{t-1|t}}\left\|\zeta_1\sqrt{\bar{\alpha}_t}(KAN_\phi(\mathbf{x}, t) - KAN_\phi(\hat{\mathbf{x}}_\theta(\mathbf{x}^t, t), t)) - (\zeta_1\sqrt{\bar{\alpha}_t} + \zeta_0)\left(KAN_\phi(\hat{\mathbf{x}}_\theta(\mathbf{x}^t, t-1), t) - KAN_\phi(\mathbf{x}, t-1)\right)\right\|_2^2. \tag{15}$$

We can compute the prior term as follows:

$$\begin{aligned} D_{KL}\left(q_\phi(\mathbf{x}^T|\mathbf{x}, c)\|p(\mathbf{x}^T|c)\right) &= \frac{1}{2}\left[\log\frac{|I|}{\sigma_T^2 I} - d + \mathrm{Tr}\{I^{-1}\sigma_T^2 I\} + \left\|\sqrt{\bar{\alpha}_T}KAN_\phi(\mathbf{x}, T) + (1-\sqrt{\bar{\alpha}_T})c - c\right\|_2^2\right] \\ &= \frac{1}{2}\left[-d\log\sigma_T^2 + d\sigma_T^2 + \left\|\sqrt{\bar{\alpha}_T}KAN_\phi(\mathbf{x}, T) - \sqrt{\bar{\alpha}_T}c\right\|_2^2\right] \\ &= \frac{1}{2}\left(d\left(\sigma_T^2 - \log\sigma_T^2 - 1\right) + \bar{\alpha}_T\left\|KAN_\phi(\mathbf{x}, T) - c\right\|_2^2\right). \\ &= \frac{1}{2}\bar{\alpha}_T\left\|KAN_\phi(\mathbf{x}, T) - c\right\|_2^2. \end{aligned} \tag{16}$$

The reconstruction term is considered for all latent variables similar to VAE(Kingma, 2013b). This formulation is inspired from Rishi et al. (2025).

E Diffusion KAN Transformer Block Equations

Our Diffusion KAN Transformer replaces MLPs in the standard DiT architecture with KAN layers. The other key components of the architecture are as follows.

Adaptive Layer Normalization with KAN Integration: Following the success of adaLN-Zero in DiT (Peebles & Xie, 2023), we employ adaptive layer normalization (adaLN) for injecting conditional information. The conditioning vector, derived from timestep t and condition c, modulates both the normalization parameters and the KAN activations:

$$\text{adaLN}(\mathbf{h}, \mathbf{c}) = \gamma_\mathbf{c} \odot \frac{\mathbf{h} - \mu(\mathbf{h})}{\sigma(\mathbf{h})} + \beta_\mathbf{c} \tag{17}$$

where $\gamma_\mathbf{c}$ and $\beta_\mathbf{c}$ are regression outputs from the conditioning vector $\mathbf{c}$.

Temporal Conditioning Mechanism: For time series analysis, we enhance the conditioning mechanism to specifically handle temporal dependencies. The timestep embedding t undergoes sinusoidal encoding followed by

$$\mathbf{e}_t = (\text{SinusoidalEmbed}(t)) \tag{18}$$

Scale and Shift Operations: We incorporate additional scale and shift operators within each transformer block, similar to recent advances in diffusion transformers (Liu et al., 2024a). These operations provide fine-grained control over feature magnitudes and enable more stable training dynamics as mentioned in Section 2.2.

The complete Diffusion KAN Transformer block can be expressed as:

$$\mathbf{h}' = \text{adaLN}(\mathbf{h}, \mathbf{c}) \tag{19}$$

$$\mathbf{h}_{\text{attn}} = \text{MultiHeadAttention}(\mathbf{h}') + \mathbf{h} \tag{20}$$

$$\mathbf{h}_{\text{norm}} = \text{adaLN}(\mathbf{h}_{\text{attn}}, \mathbf{c}) \tag{21}$$

$$\mathbf{h}_{\text{out}} = \text{KAN}(\mathbf{h}_{\text{norm}}) + \mathbf{h}_{\text{attn}} \tag{22}$$

F Algorithm

The training and inference algorithms used in our paper are as follows.

Algorithm 1 Training

Input: $\mathbf{x}(x_{1:H}), x_{-L+1:0}$
repeat
  $\epsilon \sim \mathcal{N}(0, I), t \sim U[1, T]$
  $\mathbf{x}^t \sim q_\phi(\mathbf{x}^t|\mathbf{x}, c)$
  Compute the loss in Eq. (2)
  Take numerical optimization step on: $\nabla \mathcal{L}_{\text{DiffKANformer}}$
until until converged

Algorithm 2 Inference

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

Algorithm 3 DDPM Inference

1: $\mathbf{x}^T \sim \mathcal{N}(0, I)$
2: **for** $t = T$ **to** 1 **do**
3: $\hat{\mathbf{x}} = \hat{\mathbf{x}}_\theta(\mathbf{x}^t, t)$
4: $\mathbf{x}^{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}^t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\mathbf{x}}\right) + \sigma_t\epsilon$
5: **end for**
6: **return** $\mathbf{x}_0$

G Extended Dataset Details

G.1 Forecasting

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

[3] https://www.energyonline.com/Data/
[4] https://pems.dot.ca.gov/
[5] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014
[6] https://www.bgc-jena.mpg.de/wetter/
[7] https://data.nordpoolgroup.com/power-system/production/
[8] https://github.com/laiguokun/multivariate-time-series-data
[9] https://github.com/zhouhaoyi/ETDataset

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

Table 8: Dataset characteristics for Forecasting

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

G.2 Imputation, Classification and Anomaly Detection

The descriptions of the datasets utilized for Imputation, Classification, and Anomaly Detection are presented in Table 9. These configurations are derived from TSLib (Wu et al., 2022; Wang et al., 2024b).

Table 9: Dataset descriptions. The dataset size is organized in (Train, Validation, Test).

Tasks	Dataset	Dim	Series Length	Dataset Size	Information (Frequency)
Imputation	ETTm1, ETTm2	7	96	(34465, 11521, 11521)	Electricity (15 mins)
	ETTh1, ETTh2	7	96	(8545, 2881, 2881)	Electricity (15 mins)
	Electricity	321	96	(18317, 2633, 5261)	Electricity (15 mins)
	Weather	21	96	(36792, 5217, 10540)	Weather (10 mins)
Classification (UEA)	EthanolConcentration	3	1751	(261, 0, 263)	Alcohol Industry
	FaceDetection	144	62	(5890, 0, 3524)	Face (250Hz)
	Handwriting	3	152	(150, 0, 850)	Handwriting
	Heartbeat	61	405	(204, 0, 205)	Heart Beat
	JapaneseVowels	12	29	(270, 0, 370)	Voice
	PEMS-SF	963	144	(267, 0, 173)	Transportation (Daily)
	SelfRegulationSCP1	6	896	(268, 0, 293)	Health (256Hz)
	SelfRegulationSCP2	7	1152	(200, 0, 180)	Health (256Hz)
	SpokenArabicDigits	13	93	(6599, 0, 2199)	Voice (1025Hz)
	UWaveGestureLibrary	3	315	(120, 0, 320)	Gesture
Anomaly Detection	SMD	38	100	(566724, 141681, 708420)	Server Machine
	MSL	55	100	(14653, 11664, 73729)	Spacecraft
	SMAP	25	100	(108146, 27037, 427617)	Spacecraft
	SWaT	51	100	(396000, 99000, 449919)	Infrastructure
	PSM	25	100	(105984, 26497, 87841)	Server Machine

H Baselines

We have extensively evaluated our DiffKANformer model by benchmarking it against various baseline models using a range of methodologies across different tasks, from basis expansion methods to diffusion techniques. These evaluations include: (i) models based on diffusion (Shen et al., 2024; Shen & Kwok, 2023; Rasul et al., 2021; Tashiro et al., 2021; Lopez Alcaraz & Strodthoff, 2023; Rishi et al., 2025); (ii) basis expansion methods, as described in (Challu et al., 2023; Zhou et al., 2022a; Fan et al., 2022; Oreshkin et al., 2019); (iii) other generative approaches such as GAN and VAE, referenced in (Li et al., 2022; Rangapuram et al., 2023; Jeha et al., 2022). We have also compared DiffKANformer with transformer models (Shabani et al., 2023; Nie et al., 2023; Zhou et al.,

2022b; Wu et al., 2021; Liu et al., 2022b; Zhou et al., 2021; Kitaev et al., 2020; Liu et al., 2022c; Woo et al., 2022; Wang et al., 2024c; Tan et al., 2024; Naghashi et al., 2025; Liu et al., 2023b; Wu et al., 2022; Liu et al., 2022c; Das et al., 2023a; Woo et al., 2022; Kitaev et al., 2020; Zhang & Yan, 2023) and additional deep learning methodologies (Liu et al., 2022a; Zeng et al., 2023; Bahdanau et al., 2015; Zhou et al., 2024; Chen et al., 2023; Wang et al., 2024a; 2023; Zhang et al., 2022; Xu et al., 2021).

I Implementation Details

The MLP modules within the embeddings of the condition and noised input, as well as the decoder, are from TiDE (Das et al., 2023a). This MLP framework is acknowledged as an effective component for universal time series analysis models (Das et al., 2023b). In the diffusion transformer block (DiT), MLP is replaced with KAN, with the diffusion time step adjusted by adaptive layer normalization (Peebles & Xie, 2023; Esser et al., 2024). Hyperparameters are tuned specifically regarding embedding layers, encoder layers, time steps and hidden dimensions while other hyperparameters remain constant as detailed in Section 3. Optimal hyperparameters are selected via cross-validation and summarized for various datasets and tasks in Table 10. In the condition network, for condition c, a dense layer is used for tasks such as forecasting and anomaly detection, while a transformer is utilized for imputation and classification tasks. The implementation of KAN is inspired by Yang & Wang (2024).

Table 10: Best hyperparameters for all our time series tasks.

Task	Dataset	Timesteps	Hidden Dim	Depth	Emb	Heads
Forecasting	weather	100	256	1	4	8
	Caiso	100	128	3	5	8
	ETTh1	100	128	3	5	8
	Exchange	200	64	2	2	4
	ECL	100	256	3	5	8
	Norpool	200	256	2	1	4
	ETTm1	100	128	1	4	8
	Traffic	200	256	3	2	8
Imputation	ETTh1	200	128	2	1	1
	ETTh2	200	128	2	1	1
	ETTm2	200	128	2	1	1
	ECL	200	128	2	1	4
	ETTm1	200	128	2	1	8
	weather	200	32	2	1	4
Classification	EthanolConcentration	200	128	4	4	8
	FaceDetection	200	256	1	4	8
	Handwriting	100	256	1	1	8
	Heartbeat	200	256	4	4	1
	PEMS-SF	200	256	1	1	8
	SelfRegulationSCP2	100	256	4	1	8
	SpokenArabicDigits	200	256	1	1	8
	UWaveGestureLibrary	200	128	2	1	8
	JapaneseVowels	1000	32	1	1	8
	SelfRegulationSCP1	200	256	1	1	8
Anomaly Detection	MSL	100	32	4	1	4
	PSM	100	64	1	1	8
	SMAP	100	64	4	1	4
	SMD	100	32	1	1	1
	SWaT	100	32	4	4	4

J SENSITIVITY ANALYSIS

J.1 DIFFUSION HYPERPARAMETERS

Table 11 presents the metrics for different tasks involving DiffKANformer, dependent on T (diffusion timesteps). In Classification, the accuracy is evaluated, along with Forecasting via MSE and MAE, and for Anomaly Detection, both Accuracy and F1 score are assessed. Notably, setting T = 200 leads to improved outcomes in both classification and anomaly detection. Meanwhile, T = 100 is more effective for forecasting.

Table 11: Ablation Study with varying diffusion time steps

Diffusion time	Classification (Ethanol)	Anomaly Detection (MSL)		Forecasting (ETTm1)	
Datasets	Accuracy	Acc	F1	MSE	MAE
50	0.261	0.969	0.839	0.344	0.380
100	0.261	0.965	0.815	**0.337**	**0.375**
200	**0.316**	**0.980**	**0.903**	0.339	0.377
500	0.234	0.979	0.899	0.351	0.390

A quadratic variance schedule is employed to establish the variance of Gaussian noise β_k. In this study, the parameter β_T is selected from the set $\{0.001, 0.01, 0.1, 0.9\}$, while β_1 remains fixed at 10^{-4}. As shown in Table 12, the optimal setting with $\beta_T = 0.1$ consistently yields improved results.

Table 12: Ablation study by varying variance upper bound (β_T)

Tasks (Datasets)	Classification (Ethanol)	Anomaly Detection (MSL)		Forecasting (ETTm1)	
β_T Metric	Accuracy	Acc	F1	MSE	MAE
0.001	0.246	0.974	0.868	0.345	0.381
0.01	0.261	0.973	0.867	0.339	0.376
0.1	**0.316**	**0.980**	**0.879**	**0.337**	**0.375**
0.9	0.250	0.964	0.806	0.337	0.376

J.2 LOOK BACK WINDOW FOR FORECASTING

Table 13 presents the prediction MSE and MAE for DiffKANformer utilizing various lengths of the lookback window. The values considered are L = {96, 192, 336, 720, 1440}. It is evident that on the datasets ETTh1, and weather (corresponding to 168-step, and 672-step-ahead predictions, respectively), satisfactory performance is achieved when L is set to 720.

Table 13: Predicting MSE and MAE for different Lookback windows (L) for Multivariate forecasting

Look back window	96		192		336		720		1440	
Datasets	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.518	0.487	0.445	0.443	0.428	0.439	**0.401**	**0.425**	0.426	0.443
Weather	0.365	0.368	0.342	0.362	0.331	0.361	**0.307**	**0.336**	0.394	0.412

K COMPUTATIONAL ANALYSIS

We evaluate the computational efficiency of our proposed DiffKANformer in terms of training time, inference time, and model size, with results summarized in Tables 14 and 15. Although the current

KAN implementation is still in its early stage and not fully parallelizable, leading to moderately higher training times compared to the most optimized baselines, this limitation is primarily due to immature implementation rather than deficiencies of the architecture itself. With more efficient and parallelized implementations, the training efficiency of KAN-based models is expected to improve substantially. Importantly, DiffKANformer achieves competitive accuracy while maintaining a significantly smaller parameter footprint (0.5M parameters), which translates into faster inference and lower memory requirements compared to most diffusion-based baselines. These results highlight the potential of DiffKANformer as a lightweight yet effective framework for time series analysis, offering a promising trade-off between accuracy, efficiency, and scalability.

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

	$H = 96$	$H = 168$	$H = 192$	$H = 336$	$H = 720$	MSE
DiffKANformer	0.92	0.90	0.97	0.84	0.71	0.066
CN-Diff	0.21	0.26	0.27	0.31	0.39	0.066
mr-Diff	0.59	0.69	0.71	0.74	0.82	0.066
TimeDiff	0.71	0.75	0.77	0.82	0.85	0.066
TimeGrad	2.11	2.42	3.21	4.22	5.93	0.078
CSDI	5.72	7.09	7.59	10.59	17.21	0.083
SSSD	16.98	19.34	22.64	32.12	52.93	0.097

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

Model	Trainable Params	$H = 96$	$H = 168$	$H = 192$	$H = 336$	$H = 720$
DiffKANformer	0.5M	10.1	9.9	9.8	10.0	9.8
CN-Diff	1.1M	6.2	6.7	6.9	7.8	9.1
mr-Diff	1.4M	12.5	14.3	14.9	16.8	27.5
TimeDiff	1.7M	16.2	17.3	17.6	26.5	34.6
TimeGrad	3.1M	870.2	1620.9	1854.5	3119.7	6724.1
CSDI	10M	90.4	128.3	142.8	398.9	513.1
SSSD	32M	418.6	590.2	645.4	1054.2	2516.9

L Statistical Significance Analysis

Acknowledging that the performance of deep models in time series analysis can be affected by various random initializations. Table 16 presents the outcomes from various runs across several tasks, including forecasting, imputation, anomaly detection, and classification. In the forecasting task (ETTh1) and imputation task (ETTm1), we measure performance using the mean squared error (MSE). For anomaly detection (MSL), we utilize the F1-score for evaluation, and for classification (Ethanol Concentration), accuracy is employed as the metric. We note that classification accuracy and the F1-score for anomaly detection are consistent across different runs, indicating stable and reliable performance reflecting that multiple runs are predicting same categories. Conversely, the MSE for forecasting and imputation exhibits minor fluctuations across runs, highlighting the statistical significance of our model.

M Full Results

M.1 Forecasting

Table 17 illustrates the mean absolute error (MAE) outcomes for multivariate time series prediction tasks, with an average rank of 2.6. It is evident that DiffKANformer continues to demonstrate a superior overall performance for MAE as the evaluation metric.

Table 16: Statistical significance analysis across five different runs over four different task

	Forecasting	Imputation	Anomaly Detection	Classification
Datasets(metric)	ETTh1(MSE)	ETTm1(MSE)	MSL(F1)	Ethanol Conc(Accuracy)
0	0.4016211	0.0214681	0.9038665	0.3822115
1	0.4016621	0.0213752	0.9038665	0.3822115
2	0.4016901	0.0215876	0.9038665	0.3822115
3	0.4016121	0.0209875	0.9038665	0.3822115
4	0.4016319	0.0223561	0.9038665	0.3822115
mean	0.40164346	0.0215549	0.9038665	0.3822115
std deviation	0.0000287742	0.0004482766	0.0	0.0

Table 17: Multivariate prediction of MAEs on eight real-world time series datasets (subscripts are the rank). CSDI and TiDE run out of memory on Traffic and Electricity. Results of all baselines are from Shen et al. (2024). Baselines with * are run using TSLib (Wang et al., 2024b; Wu et al., 2022)

Method	NorPool	Caiso	Traffic	Electricity	Weather	Exchange	ETTh1	ETTm1	Rank
Ours	0.565 (2)	**0.189** (1)	0.268 (2)	0.243 (2)	0.325 (4)	**0.079** (1)	0.425 (7)	0.372 (2)	2.6
CnDiff*	**0.554** (1)	0.191 (2)	0.270 (6)	0.243 (2)	0.324 (2)	**0.079** (1)	0.421 (3)	0.378 (7)	3.0
mr-Diff	0.604 (5)	0.219 (5)	0.320 (14)	0.252 (6)	0.324 (2)	0.082 (7)	0.422 (4)	0.373 (3)	5.8
TimeDiff	0.611 (6)	0.234 (12)	0.384 (20)	0.305 (18)	**0.312** (1)	0.091 (16)	0.430 (10)	**0.372** (1)	10.5
TimeGrad	0.821 (35)	0.339 (34)	0.849 (37)	0.630 (37)	0.381 (27)	0.193 (34)	0.719 (37)	0.605 (37)	34.8
CSDI	0.777 (32)	0.345 (35)	–	–	0.374 (25)	0.194 (35)	0.438 (12)	0.442 (27)	27.7
SSSD	0.753 (26)	0.295 (25)	0.398 (27)	0.363 (27)	0.350 (16)	0.127 (25)	0.561 (32)	0.406 (21)	24.9
D3VAE	0.692 (14)	0.331 (32)	0.483 (31)	0.372 (29)	0.380 (26)	0.301 (37)	0.502 (26)	0.391 (15)	26.2
CPF	0.889 (37)	0.424 (37)	0.714 (36)	0.643 (38)	0.781 (39)	0.082 (7)	0.597 (35)	0.472 (29)	32.2
PSA-GAN	0.890 (38)	0.477 (38)	0.697 (35)	0.533 (36)	0.578 (37)	0.087 (14)	0.546 (30)	0.488 (33)	32.6
N-Hits	0.643 (10)	0.221 (6)	0.268 (2)	0.245 (4)	0.335 (7)	0.085 (10)	0.480 (20)	0.388 (12)	8.9
FiLM	0.646 (11)	0.278 (22)	0.398 (27)	0.320 (20)	0.336 (8)	**0.079** (1)	0.436 (11)	0.374 (4)	13.0
Depts	0.611 (6)	0.204 (4)	0.568 (34)	0.401 (32)	0.394 (30)	0.100 (18)	0.491 (24)	0.412 (23)	21.4
NBeats	0.832 (36)	0.235 (13)	**0.265** (1)	0.370 (28)	0.420 (31)	0.081 (6)	0.521 (27)	0.409 (22)	20.5
Scaleformer	0.769 (30)	0.310 (27)	0.379 (18)	0.304 (17)	0.438 (32)	0.138 (27)	0.579 (34)	0.475 (30)	26.9
PatchTST	0.710 (20)	0.293 (24)	0.411 (30)	0.348 (26)	0.555 (36)	0.147 (28)	0.489 (23)	0.392 (16)	25.4
FedFormer	0.744 (23)	0.317 (29)	0.385 (21)	0.341 (24)	0.347 (14)	0.233 (36)	0.484 (21)	0.413 (24)	24.0
Autoformer	0.751 (25)	0.321 (30)	0.392 (26)	0.313 (19)	0.354 (17)	0.167 (30)	0.484 (21)	0.496 (34)	25.2
Pyraformer	0.781 (33)	0.371 (36)	0.390 (22)	0.379 (30)	0.385 (28)	0.112 (23)	0.493 (25)	0.435 (26)	27.9
Informer	0.757 (27)	0.336 (33)	0.391 (24)	0.383 (31)	0.364 (21)	0.192 (33)	0.605 (36)	0.542 (35)	30.0
Transformer	0.765 (28)	0.321 (30)	0.410 (29)	0.405 (33)	0.370 (24)	0.178 (32)	0.567 (33)	0.592 (36)	30.6
Crossformer	0.704 (19)	0.225 (8)	0.269 (5)	0.247 (5)	0.369 (23)	0.170 (31)	0.459 (17)	0.390 (14)	15.2
NonStationary*	0.581 (3)	0.232 (11)	0.382 (19)	0.281 (14)	0.356 (18)	0.105 (22)	0.528 (28)	0.443 (28)	17.9
Timexer*	0.647 (12)	0.230 (10)	0.275 (7)	0.252 (8)	0.333 (6)	0.086 (11)	0.429 (9)	0.376 (6)	8.6
Timesnet*	0.696 (15)	0.239 (14)	0.320 (15)	0.281 (15)	0.349 (15)	0.103 (21)	0.456 (16)	0.394 (18)	16.1
Multipatchformer*	0.701 (18)	0.245 (16)	0.292 (8)	0.256 (10)	0.342 (11)	0.086 (13)	0.420 (2)	0.387 (11)	11.1
PAttn*	0.698 (17)	0.248 (17)	0.317 (12)	0.267 (11)	0.341 (10)	0.080 (5)	0.429 (8)	0.384 (10)	11.2
Reformer*	0.717 (22)	0.272 (21)	0.390 (23)	0.409 (34)	0.613 (38)	0.650 (39)	0.740 (38)	0.689 (38)	31.6
iTransformer*	0.711 (21)	0.264 (19)	0.292 (9)	0.252 (7)	0.346 (13)	0.086 (12)	0.439 (13)	0.397 (19)	14.1
ETSFormer*	0.783 (34)	0.291 (23)	0.525 (33)	0.341 (25)	0.500 (34)	0.112 (24)	0.555 (31)	0.484 (32)	29.5
TiDE*	0.745 (24)	0.315 (28)	–	0.291 (16)	0.392 (29)	0.079 (4)	0.424 (6)	0.399 (20)	18.1
SCINet	0.601 (4)	0.193 (3)	0.335 (16)	0.280 (13)	0.344 (12)	0.137 (26)	0.463 (19)	0.389 (13)	13.2
NLinear	0.636 (8)	0.223 (7)	0.293 (10)	**0.239** (1)	0.328 (5)	0.091 (16)	**0.418** (1)	0.375 (5)	6.6
DLinear	0.640 (9)	0.497 (39)	0.268 (2)	0.336 (22)	0.444 (33)	0.102 (19)	0.442 (14)	0.378 (7)	18.1
LSTMa	0.974 (39)	0.305 (26)	0.510 (32)	0.444 (35)	0.501 (35)	0.534 (38)	0.782 (39)	0.699 (39)	35.4
MICN*	0.682 (13)	0.230 (9)	0.306 (11)	0.277 (12)	0.365 (22)	0.089 (15)	0.449 (15)	0.393 (17)	14.2
TSMixer*	0.774 (31)	0.244 (15)	0.369 (17)	0.336 (23)	0.357 (19)	0.149 (29)	0.538 (29)	0.477 (31)	24.2
LightTS*	0.765 (29)	0.253 (18)	0.391 (25)	0.327 (21)	0.362 (20)	0.102 (20)	0.462 (18)	0.418 (25)	22.0
TimeMixer*	0.698 (16)	0.265 (20)	0.319 (13)	0.255 (9)	0.341 (9)	0.084 (9)	0.422 (5)	0.381 (9)	11.2

M.2 IMPUTATION

Table 18 and 19 provides a comprehensive analysis of the MSE and MAE of several imputation models across various datasets with differing levels of missing data (12.5%, 25%, 37.5% and 50%). Our model's excellence is evident in the "Avg Rank" row, where it secures the top position with

a rank of 1.5. These findings collectively underscore the robustness of our imputation method, particularly as the proportion of missing data increases.

Table 18: Full results for the imputation task with MSE metrics. We randomly mask 12.5%, 25%, 37.5% and 50% time points to compare the model performance under different missing degrees. + in the Transformers indicates the name of + former .

Dataset		**Ours**	NS+	TimesNet	Cross+	LightTS	iTrans+	SSSD	Pyra+	Re+	Trans+	TiDE	DLinear	In+	ETS*	FED*	FiLM	Auto+
ECL	0.125	**0.060**	0.078	0.088	0.079	0.077	0.073	0.197	0.177	0.148	0.149	0.085	0.085	0.149	0.117	0.184	0.085	0.196
	0.25	**0.070**	0.085	0.091	0.092	0.099	0.090	0.205	0.184	0.155	0.161	0.114	0.113	0.165	0.127	0.206	0.114	0.215
	0.375	**0.079**	0.091	0.096	0.104	0.119	0.107	0.214	0.190	0.166	0.170	0.142	0.141	0.179	0.141	0.225	0.142	0.242
	0.5	**0.092**	0.099	0.102	0.114	0.137	0.128	0.217	0.204	0.178	0.178	0.174	0.173	0.193	0.158	0.247	0.174	0.269
	Avg	**0.075**	0.088	0.094	0.097	0.108	0.099	0.208	0.189	0.162	0.165	0.129	0.128	0.171	0.136	0.215	0.129	0.231
ETTh1	0.125	0.047	0.047	0.062	0.120	0.114	0.098	**0.046**	0.077	0.060	0.065	0.111	0.111	0.071	0.086	0.072	0.117	0.094
	0.25	**0.052**	0.064	0.080	0.139	0.146	0.125	0.065	0.102	0.084	0.118	0.150	0.150	0.095	0.118	0.103	0.154	0.118
	0.375	**0.077**	0.083	0.097	0.156	0.175	0.156	0.090	0.128	0.113	0.113	0.186	0.188	0.127	0.152	0.140	0.191	0.160
	0.5	**0.083**	0.107	0.116	0.176	0.203	0.212	0.110	0.156	0.146	0.135	0.229	0.229	0.165	0.196	0.197	0.233	0.217
	Avg	**0.065**	0.075	0.089	0.148	0.159	0.148	0.078	0.116	0.101	0.108	0.169	0.170	0.114	0.138	0.128	0.174	0.147
ETTh2	0.125	**0.029**	0.039	0.040	0.126	0.113	0.095	0.184	0.141	0.144	0.182	0.108	0.108	0.279	0.132	0.127	0.110	0.212
	0.25	**0.025**	0.046	0.047	0.143	0.140	0.119	0.336	0.186	0.201	0.252	0.143	0.144	0.354	0.189	0.176	0.150	0.219
	0.375	**0.037**	0.052	0.054	0.155	0.166	0.148	0.477	0.221	0.246	0.281	0.179	0.179	0.402	0.264	0.247	0.188	0.274
	0.5	**0.055**	0.060	0.061	0.181	0.189	0.192	0.642	0.257	0.313	0.308	0.221	0.217	0.430	0.351	0.348	0.227	0.360
	Avg	**0.037**	0.049	0.051	0.151	0.152	0.139	0.410	0.201	0.226	0.256	0.163	0.162	0.366	0.234	0.224	0.169	0.266
ETTm1	0.125	0.017	0.018	0.019	0.051	0.052	0.045	**0.016**	0.027	0.024	0.022	0.056	0.056	0.028	0.042	0.034	0.056	1.103
	0.25	0.021	0.022	0.023	0.054	0.062	0.060	**0.020**	0.034	0.033	0.030	0.076	0.076	0.040	0.061	0.051	0.077	0.811
	0.375	0.026	0.028	0.029	0.060	0.073	0.077	**0.024**	0.040	0.035	0.037	0.099	0.099	0.056	0.084	0.079	0.100	0.561
	0.5	**0.032**	0.034	0.036	0.067	0.086	0.101	0.032	0.049	0.047	0.045	0.128	0.128	0.073	0.104	0.125	0.129	0.484
	Avg	0.024	0.026	0.027	0.060	0.068	0.071	**0.023**	0.037	0.033	0.034	0.090	0.090	0.049	0.073	0.072	0.090	0.740
ETTm2	0.125	**0.011**	0.018	0.019	0.069	0.062	0.052	0.088	0.077	0.105	0.178	0.066	0.066	0.174	0.081	0.060	0.066	1.059
	0.25	**0.014**	0.020	0.021	0.069	0.073	0.070	0.128	0.101	0.142	0.215	0.089	0.088	0.244	0.115	0.090	0.089	0.957
	0.375	**0.018**	0.022	0.023	0.087	0.084	0.090	0.236	0.142	0.182	0.206	0.111	0.111	0.285	0.163	0.131	0.112	0.726
	0.5	**0.022**	0.025	0.026	0.091	0.087	0.117	0.158	0.192	0.230	0.213	0.138	0.138	0.277	0.209	0.237	0.138	0.705
	Avg	**0.016**	0.021	0.022	0.079	0.076	0.082	0.152	0.128	0.165	0.203	0.101	0.101	0.245	0.142	0.129	0.102	0.862
weather	0.125	0.028	0.025	**0.023**	0.039	0.038	0.038	0.036	0.029	0.031	0.030	0.038	0.038	0.036	0.036	0.042	0.041	0.304
	0.25	0.031	**0.027**	0.028	0.042	0.043	0.046	0.029	0.032	0.036	0.034	0.047	0.047	0.039	0.044	0.058	0.050	0.237
	0.375	0.032	0.031	0.031	0.044	0.048	0.055	**0.030**	0.036	0.040	0.040	0.056	0.056	0.044	0.052	0.077	0.062	0.151
	0.5	0.036	0.036	0.034	0.047	0.053	0.067	**0.033**	0.040	0.048	0.044	0.066	0.066	0.050	0.061	0.114	0.076	0.148
	Avg	0.032	0.030	**0.029**	0.043	0.046	0.051	0.032	0.034	0.039	0.036	0.052	0.052	0.042	0.048	0.073	0.057	0.210
Avg MSE		**0.041**	0.048	0.049	0.095	0.102	0.104	0.151	0.118	0.114	0.128	0.117	0.117	0.165	0.129	0.140	0.120	0.408
Avg Rank		**1.5**	2.2	3.0	7.3	8.3	8.5	8.8	9.0	9.0	9.8	10.8	11.2	11.3	11.5	12.5	12.5	15.7

Table 19: Full results for the imputation task with MAE metrics. We randomly mask 12.5%, 25%, 37.5% and 50% time points to compare the model performance under different missing degrees. + in the Transformers indicates the name of + former .

Dataset		**Ours**	NS+	TimesNet	Cross+	iTrans+	LightTS	SSSD	Re+	Pyra+	Trans+	DLinear	TiDE	In+	ETS+	FED+	FiLM	Auto+
ECL	0.125	**0.168**	0.193	0.203	0.199	0.190	0.197	0.304	0.276	0.298	0.277	0.207	0.207	0.276	0.246	0.322	0.207	0.333
	0.25	**0.184**	0.201	0.208	0.217	0.214	0.228	0.308	0.282	0.299	0.285	0.243	0.244	0.288	0.257	0.340	0.243	0.345
	0.375	**0.195**	0.209	0.213	0.231	0.235	0.252	0.312	0.290	0.303	0.291	0.273	0.274	0.297	0.270	0.354	0.274	0.365
	0.5	**0.212**	0.218	0.221	0.244	0.258	0.271	0.313	0.297	0.310	0.296	0.303	0.305	0.306	0.285	0.370	0.304	0.382
	Avg	**0.190**	0.205	0.211	0.223	0.224	0.237	0.309	0.286	0.302	0.287	0.256	0.257	0.292	0.264	0.347	0.257	0.356
ETTh1	0.125	0.152	**0.146**	0.168	0.237	0.219	0.235	0.151	0.172	0.200	0.181	0.232	0.231	0.188	0.209	0.194	0.238	0.222
	0.25	**0.156**	0.169	0.191	0.258	0.249	0.267	0.177	0.205	0.229	0.245	0.269	0.268	0.219	0.245	0.233	0.273	0.249
	0.375	**0.189**	0.192	0.209	0.277	0.279	0.294	0.205	0.239	0.259	0.238	0.302	0.299	0.252	0.277	0.275	0.303	0.288
	0.5	**0.195**	0.217	0.227	0.298	0.326	0.318	0.232	0.273	0.285	0.262	0.332	0.329	0.287	0.315	0.327	0.334	0.337
	Avg	**0.173**	0.181	0.199	0.268	0.268	0.278	0.191	0.222	0.243	0.232	0.284	0.282	0.236	0.261	0.257	0.287	0.274
ETTh2	0.125	**0.121**	0.130	0.131	0.237	0.210	0.227	0.310	0.268	0.276	0.311	0.222	0.222	0.389	0.256	0.242	0.224	0.311
	0.25	**0.111**	0.142	0.144	0.254	0.238	0.256	0.417	0.317	0.328	0.374	0.259	0.258	0.451	0.316	0.290	0.263	0.327
	0.375	**0.136**	0.152	0.155	0.267	0.265	0.280	0.501	0.355	0.350	0.389	0.289	0.289	0.480	0.375	0.340	0.295	0.359
	0.5	0.168	**0.164**	0.165	0.290	0.302	0.300	0.581	0.411	0.377	0.406	0.319	0.321	0.503	0.442	0.399	0.326	0.413
	Avg	**0.134**	0.147	0.149	0.262	0.254	0.266	0.452	0.338	0.333	0.370	0.272	0.273	0.456	0.347	0.318	0.277	0.353
ETTm1	0.125	**0.084**	0.090	0.092	0.158	0.147	0.156	0.088	0.110	0.113	0.103	0.161	0.161	0.117	0.144	0.130	0.162	0.875
	0.25	**0.091**	0.098	0.101	0.164	0.172	0.174	0.095	0.129	0.126	0.121	0.190	0.190	0.141	0.175	0.160	0.190	0.719
	0.375	**0.101**	0.110	0.111	0.171	0.195	0.190	0.103	0.130	0.139	0.134	0.217	0.217	0.168	0.205	0.196	0.217	0.565
	0.5	**0.110**	0.122	0.124	0.181	0.225	0.207	0.119	0.151	0.153	0.148	0.246	0.246	0.194	0.230	0.251	0.246	0.504
	Avg	**0.096**	0.105	0.107	0.171	0.184	0.182	0.101	0.127	0.133	0.126	0.204	0.203	0.155	0.189	0.185	0.204	0.666
ETTm2	0.125	**0.071**	0.078	0.082	0.174	0.151	0.165	0.210	0.230	0.200	0.310	0.171	0.171	0.300	0.199	0.166	0.171	0.819
	0.25	**0.080**	0.083	0.087	0.174	0.178	0.182	0.256	0.272	0.236	0.342	0.199	0.200	0.368	0.241	0.206	0.200	0.748
	0.375	0.091	**0.089**	0.092	0.196	0.203	0.196	0.341	0.318	0.272	0.348	0.225	0.225	0.406	0.294	0.244	0.226	0.630
	0.5	0.102	**0.096**	0.099	0.195	0.232	0.202	0.272	0.359	0.307	0.343	0.252	0.251	0.386	0.331	0.331	0.251	0.587
	Avg	**0.086**	0.087	0.090	0.185	0.191	0.186	0.270	0.295	0.254	0.336	0.212	0.212	0.365	0.266	0.237	0.212	0.696
weather	0.125	0.049	0.046	**0.046**	0.094	0.086	0.089	0.085	0.077	0.065	0.076	0.089	0.089	0.092	0.092	0.102	0.089	0.402
	0.25	0.054	**0.051**	0.052	0.101	0.105	0.102	0.067	0.087	0.075	0.086	0.108	0.108	0.094	0.109	0.133	0.111	0.334
	0.375	**0.056**	0.056	0.057	0.104	0.121	0.108	0.062	0.092	0.082	0.095	0.122	0.122	0.102	0.122	0.163	0.131	0.257
	0.5	0.062	0.064	**0.061**	0.110	0.141	0.114	0.065	0.105	0.090	0.100	0.136	0.137	0.107	0.138	0.209	0.152	0.254
	Avg	0.055	0.054	**0.054**	0.102	0.113	0.103	0.070	0.090	0.078	0.088	0.114	0.114	0.099	0.115	0.152	0.120	0.312
Avg MAE		**0.122**	0.130	0.133	0.201	0.213	0.209	0.232	0.219	0.224	0.229	0.224	0.223	0.267	0.241	0.249	0.226	0.443
Avg Rank		**1.3**	2.2	3.0	7.0	8.2	8.5	8.8	9.2	9.3	9.8	11.0	11.0	11.5	12.0	12.2	12.2	15.8

M.3 Anomaly Detection

The results for the anomaly detection task are summarized in the table 20, which compares the performance of various models across multiple datasets, including MSL, PSM, SMAP, SMD and SWaT. The evaluation metrics used are Accuracy (A), Recall (R), and Precision (P), with higher values indicating superior performance. Our proposed model consistently demonstrates state-of-the-art performance, achieving the highest scores in accuracy, recall, and precision on nearly all datasets. This is further validated by its overall average rank of 2.2, which is significantly better than the next best model, TimesNet, at 3.8. These findings collectively highlight the robust and superior capability of our model in effectively identifying anomalies across diverse time series data.

Table 20: Full results for the anomaly detection task. The A, R and P represent the accuracy, recall and precision (%) respectively. A higher value of A, R and P indicates a better performance.

Model	MSL			PSM			SMAP			SMD			SWaT			Average	
	A	R	P	A	R	P	A	R	P	A	R	P	A	R	P	A	Rank
Ours	**97.91**	86.85	92.82	97.83	93.34	98.30	**97.93**	90.62	93.09	98.76	76.92	92.01	97.83	82.92	98.78	98.05	2.2
TimesNet	96.47	75.29	89.55	**98.57**	96.30	98.52	93.62	56.41	90.01	98.77	81.54	87.89	**98.15**	93.54	91.45	97.12	3.8
SSSD	97.18	79.35	92.80	96.34	87.14	99.62	93.76	58.23	89.23	**99.12**	84.66	93.53	97.35	80.29	97.48	96.75	4.4
MICN	96.06	71.98	88.56	97.81	93.24	98.79	93.78	58.14	89.61	98.83	78.95	91.65	97.60	87.48	92.35	96.82	5.4
DLinear	96.48	75.30	89.69	98.17	94.69	98.65	93.51	56.38	88.82	98.71	75.84	91.65	96.75	80.36	91.88	96.72	7.4
iTransformer	95.00	62.63	86.15	96.99	90.26	98.77	93.62	56.33	90.07	98.61	73.45	91.31	97.47	86.29	92.39	96.34	8.2
AnomalyT	96.85	76.29	92.43	96.20	86.75	99.48	93.71	57.80	89.22	98.40	70.08	89.22	95.82	65.73	99.75	96.20	9.4
LightTS	96.33	74.08	89.32	96.54	88.54	98.84	93.56	55.80	90.12	98.69	75.41	91.55	96.25	69.16	99.93	96.27	9.4
FiLM	93.88	51.30	84.62	96.68	89.02	98.88	93.52	56.40	88.86	98.72	76.59	91.19	96.75	80.36	91.85	95.91	10.2
KANAD	96.42	72.82	91.43	95.15	83.54	98.81	93.56	56.31	89.43	98.71	76.25	91.27	95.83	65.73	99.95	95.93	10.8
Reformer	96.38	73.71	90.15	96.04	85.97	99.71	93.56	56.76	88.84	97.95	60.39	86.25	95.83	65.73	99.94	95.95	11.4
Informer	96.37	73.69	90.12	96.04	85.90	99.77	93.57	56.79	88.93	97.95	60.37	86.30	95.83	65.73	99.92	95.95	11.4
Crossformer	96.31	72.78	90.32	96.99	89.72	99.36	93.52	56.51	88.77	98.15	64.86	87.52	95.83	65.73	99.89	96.16	11.4
Transformer	96.33	73.66	89.70	95.24	83.15	99.62	94.27	61.37	90.86	97.80	65.24	78.32	95.87	66.10	99.81	95.90	11.6
Autoformer	96.63	75.80	90.71	94.16	78.95	99.99	94.38	62.53	90.65	97.81	65.10	78.45	95.81	65.56	99.96	95.76	11.8
Pyraformer	95.73	68.47	88.46	95.99	86.03	99.41	93.58	56.96	88.84	98.43	71.05	88.87	95.83	65.73	99.90	95.91	11.8
FEDformer	96.57	75.23	90.65	95.58	84.66	99.31	93.51	56.38	88.76	97.89	59.10	85.61	95.84	65.73	99.95	95.88	12.4
ETSformer	94.56	55.68	88.46	95.41	84.65	98.58	93.49	55.49	89.72	98.30	65.29	91.36	96.98	81.60	92.61	95.75	13.6

N Qualitative Results

N.1 Forecasting

The forecasting visualizations in Figures 3 and 4 demonstrate the predictive capabilities of the proposed DiffKANformer model across various datasets. Figure 3 provides a comparative analysis on the ETTh1 dataset, where DiffKANformer's predictions exhibit a high degree of accuracy, closely following the ground truth values and outperforming the forecasting accuracy of other baseline models such as TimesNet, Reformer, and Nonstationary Transformer. These results collectively validate the effectiveness of the DiffKANformer architecture for long-term time series forecasting. Also shown in Figure 4, our model effectively captures the underlying patterns and trends of distinct time series, including electricity consumption (ECL, Caiso, Norpool) and weather data. The close alignment between the ground truth (green line) and the prediction (dashed orange line) in the forecasting window suggests the model's ability of forecasting different datasets accurately.

N.2 Imputation

To evaluate the effectiveness of our proposed method for imputation, we present qualitative comparisons through imputation visualizations across multiple datasets and masking scenarios. Figure 5 demonstrate the imputation performance of various time series models like DLinear, Crossformer, NonStationary Transformer on the Ettm1 dataset and Figure 6 Weather, Etth1, Etth2 and ECL at 50% and 25% mask ratios respectively. These visualizations clearly show that our method achieves superior imputation quality compared to baseline approaches, maintaining better temporal continuity and more accurately capturing the underlying patterns of the original time series.

N.3 Anomaly Detection

Figure 7 demonstrates the superior anomaly detection capabilities of our proposed method on the SMAP dataset across different baseline models including DLinear, Crossformer, and NonStationary Transformer. The visualization clearly shows that our method successfully identifies and highlights

the anomalous regions (indicated by the yellow shaded areas), while the baseline methods fail to detect these critical anomalies. This superior detection performance validates the effectiveness of our method in capturing meaningful temporal dependencies essential for accurate anomaly identification in time series data.

O SAMPLING TIME AND CONVERGENCE ANALYSIS

Our ablation studies (Tables 21, 22, 23) reveal critical insights into the computational overhead and convergence characteristics of KAN-based projections compared to traditional Markov diffusion models (without KAN projection).

Examining the training time complexity, KAN models exhibit substantially reduced training times across all configurations. For instance, at T=50 timesteps (Table 21), KAN requires only 295,227 seconds compared to Markov's 377,415 seconds, representing a 22% reduction in training time. This efficiency gain becomes more pronounced with increased timesteps: at T=400, KAN completes training in 171,007 seconds versus Markov's 315,110 seconds—a remarkable 46% improvement. This superlinear scaling advantage suggests that KAN's learnable activation functions converge more efficiently during the forward diffusion process, requiring fewer gradient updates to reach comparable performance.

The memory efficiency of KAN projections demonstrates consistent advantages, with peak memory consumption remaining stable around 90.2-90.3 MB across different timestep configurations, while Markov models show slight increases from 89.7 MB to 89.7 MB. This memory stability indicates that KAN's computational graph maintains bounded complexity regardless of the diffusion timestep parameter T.

The prediction horizon ablation (Table 22) reveals that KAN architectures maintain superior convergence rates even with extended prediction lengths. At Pred=720, KAN trains in 191,307 seconds versus Markov's 366,121 seconds a 48% reduction while achieving comparable test performance (40,705 vs 32,527 seconds). This indicates that KAN's learned basis functions generalize more effectively across longer temporal dependencies.

Most notably, the sequence length scaling (Table 23) demonstrates near linear computational complexity for KAN. As sequence length increases from 512 to 1024, KAN's training time scales from 184,162 to 175,398 seconds (actually decreasing due to optimization convergence), while Markov scales from 296,145 to 375,446 seconds. This sublinear scaling behavior for KAN suggests superior sample efficiency and faster convergence to optimal diffusion parameters.

These results conclusively demonstrate that KAN projections achieve faster convergence with lower computational overhead in the forward diffusion process. The consistent memory footprint, reduced training times (up to 48% improvement), and favorable scaling properties make KAN-based architectures particularly suitable for large scale diffusion models where training efficiency is paramount. The slight increase in test time for certain configurations is more than offset by the substantial training time reductions, making the overall computational budget significantly more favorable for KAN implementations.

Table 21: Computational complexity analysis for diffusion timestep (T) ablation. KAN projections demonstrate superior training efficiency with 22-46% reduction in training time compared to Markov models across all timestep configurations (T=50 to 400), while maintaining stable memory consumption (90.2 MB). The convergence advantage becomes more pronounced at higher timesteps, indicating KAN's efficient optimization in the forward diffusion process.

Type	T	train time sec	test time sec	train peak mem MB	test peak mem MB
Markov	50	377.4154	20.7349	89.6963	59.4868
Markov	100	349.3379	40.3780	89.6963	59.4868
Markov	200	270.5591	81.2419	89.6978	59.4883
Markov	400	315.1102	157.6585	89.7007	59.4912
KAN	50	295.2273	25.3280	90.2920	59.8442
KAN	100	169.3998	49.8523	90.2920	59.8442
KAN	200	158.0108	98.2846	90.2935	59.8457
KAN	400	171.0078	196.9544	90.2964	59.8486

Table 22: Computational complexity analysis for prediction horizon (Pred) ablation. KAN models exhibit significantly faster convergence across all prediction lengths, achieving up to 48% reduction in training time at Pred=720 (191,307s vs 366,121s). The consistent computational advantage demonstrates KAN's superior generalization capabilities for extended temporal dependencies, with stable memory footprint across varying prediction horizons.

Type	Pred	train time sec	test time sec	train peak mem MB	test peak mem MB
Markov	96	381.2194	42.1023	87.3667	58.1963
Markov	168	353.5391	40.4335	89.6963	59.4868
Markov	720	366.1210	32.3271	107.5386	69.3701
Markov	336	315.5612	38.7018	95.1255	62.4941
KAN	96	193.2493	51.0251	87.5962	58.3340
KAN	168	170.7544	50.0561	90.2920	59.8442
KAN	336	204.5689	47.2504	97.3374	63.8213
KAN	720	191.3073	40.7052	117.5020	75.3521

Table 23: Computational complexity analysis for sequence length (Seq) ablation. KAN projections show favorable scaling properties as sequence length increases from 512 to 1024, with training times remaining near-constant (184,162s to 175,398s) while Markov models exhibit substantial increases (296,145s to 375,446s). This sublinear scaling behavior indicates superior sample efficiency and faster convergence to optimal parameters, making KAN particularly suitable for long-sequence modeling tasks.

Type	Seq	train time sec	test time sec	train peak mem MB	test peak mem MB
Markov	512	296.1450	40.2757	88.7505	59.0869
Markov	720	349.0643	40.4680	89.6963	59.4868
Markov	1024	375.4464	40.5819	91.0786	60.0713
KAN	512	184.1622	49.0772	89.3462	59.4443
KAN	720	172.4618	49.2707	90.2920	59.8442
KAN	1024	175.3989	49.8541	91.6743	60.4287

P Comparision of DiT-KAN and DiT-MLP

The computational complexity analysis presented in Table 24 compares DiT-MLP and DiT-KAN architectures across varying sequence lengths. Our experiments reveal that DiT-KAN consistently outperforms DiT-MLP in terms of training efficiency. Specifically, at a sequence length of 512, DiT-KAN achieves a training time of 184.16 seconds compared to 209.77 seconds for DiT-MLP, representing a reduction of approximately 12%. This efficiency gap widens considerably as sequence length increases; at a sequence length of 1024, DiT-KAN requires only 175.40 seconds for training, whereas DiT-MLP demands 380.60 seconds, yielding a speedup factor of approximately 2.2×.

Notably, DiT-KAN exhibits remarkable stability in test time performance across different sequence lengths, maintaining values between 49.08 and 49.85 seconds regardless of input dimensionality. In contrast, DiT-MLP demonstrates comparable test times ranging from 40.47 to 41.69 seconds. While DiT-MLP shows marginally faster inference, the difference remains relatively modest compared to the substantial training time advantages offered by DiT-KAN.

Regarding memory consumption, both architectures exhibit similar peak memory usage during training and testing phases. DiT-KAN's training peak memory ranges from 89.35 to 91.67 MB, closely matching DiT-MLP's range of 89.32 to 91.65 MB. Similarly, test peak memory remains comparable across both methods, with differences of less than 1 MB in most configurations. These results suggest that DiT-KAN achieves its computational efficiency gains without imposing additional memory overhead, making it a practical and scalable alternative for diffusion transformer applications.

Q DiffKANformer to DDPM

In Appendix D, we explicitly derived each step used to obtain the loss formulation for our proposed method.

Table 24: Computational complexity analysis of DIT-KAN and DiT-MLP for various sequence lengths.

Type	Seq	train time sec	test time sec	train peak mem MB	test peak mem MB
DiT-MLP	512	209.7688	41.6930	89.3168	59.4267
DiT-MLP	720	349.0643	40.4680	89.6963	59.4868
DiT-MLP	1024	380.6010	41.2959	91.6450	60.4111
DiT-KAN	512	184.1622	49.0772	89.3462	59.4443
DiT-KAN	720	172.4618	49.2707	90.2920	59.8442
DiT-KAN	1024	175.3989	49.8541	91.6743	60.4287

From equation 5 in Appendix D.1 we can clearly see.

$$q_\phi(\mathbf{x}^t|\mathbf{x}, c) = \mathcal{N}(\mathbf{x}^t; \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c, (1 - \bar{\alpha}_t)I) \tag{23}$$

From (23), We can write,

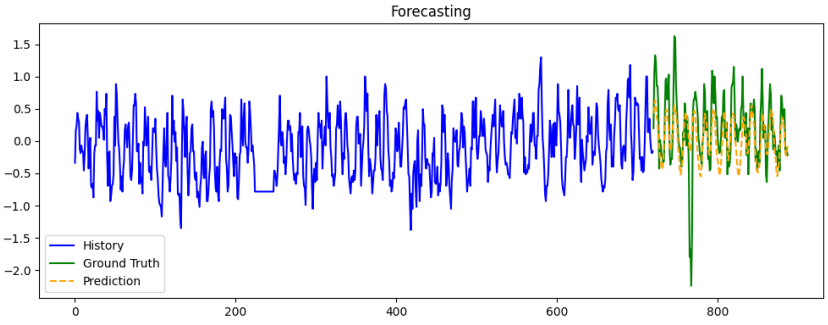
$$\begin{aligned}
\mathbf{x}^t &= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \bar{\alpha}_t}\epsilon \\
&= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \alpha_t\bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \alpha_t}\epsilon + \sqrt{\alpha_t - \alpha_t\bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \alpha_t}\epsilon + \sqrt{\alpha_t}\sqrt{1 - \bar{\alpha}_{t-1}}\epsilon \\
&= \sqrt{\bar{\alpha}_t} KAN_\phi(\mathbf{x}, t) + (1 - \sqrt{\bar{\alpha}_t})c + \sqrt{1 - \alpha_t}\epsilon + \sqrt{\alpha_t}\left(\mathbf{x}^{t-1} - \sqrt{\bar{\alpha}_{t-1}} KAN_\phi(\mathbf{x}, t-1) - (1 - \sqrt{\bar{\alpha}_{t-1}})c\right) \\
&= \sqrt{\alpha_t}\mathbf{x}^{t-1} + \sqrt{\bar{\alpha}_t}\left(KAN_\phi(\mathbf{x}, t) - KAN_\phi(\mathbf{x}, t-1)\right) + (1 - \sqrt{\alpha_t})c + \sqrt{1 - \alpha_t}\epsilon
\end{aligned} \tag{24}$$

Here, if we replace KAN = 1d and c = 0 in the above equation, we get

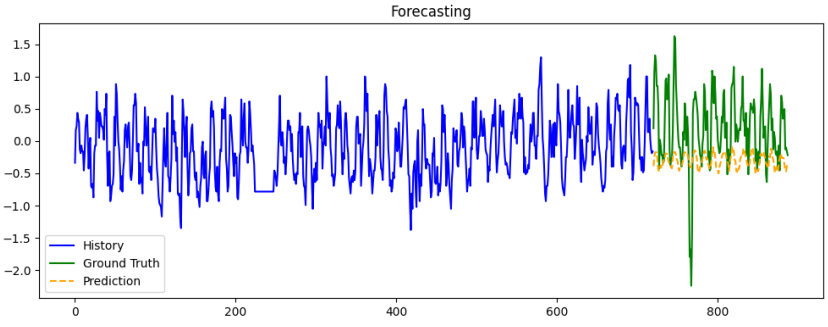
$$\mathbf{x}^t = \sqrt{\alpha_t}\mathbf{x}^{t-1} + \sqrt{1 - \alpha_t}\epsilon \tag{25}$$

$$q_\phi(\mathbf{x}^t|\mathbf{x}, c) = \mathcal{N}(\mathbf{x}^t; \sqrt{\bar{\alpha}_t}\mathbf{x} + (1 - \bar{\alpha}_t)I) \tag{26}$$

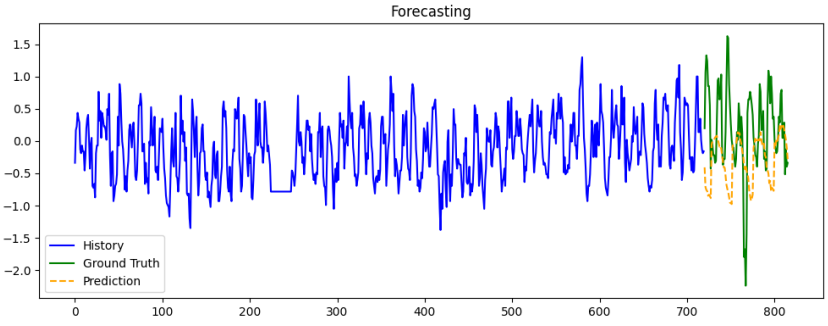
Which is DDPM formulation.



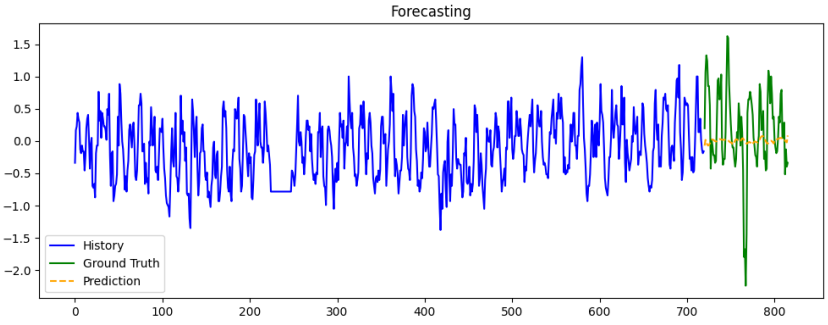
(a) DiffKANformer



(b) TimesNet

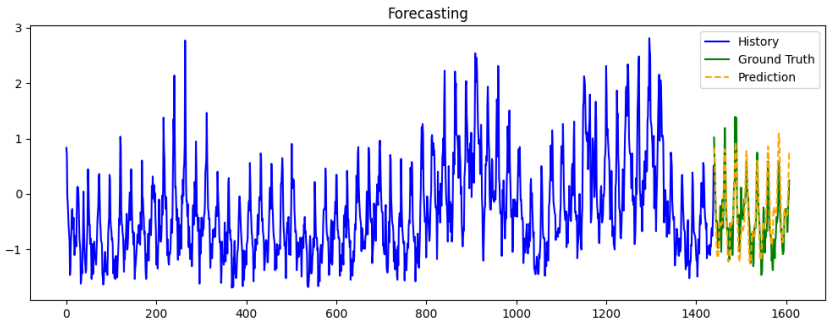


(c) Reformer

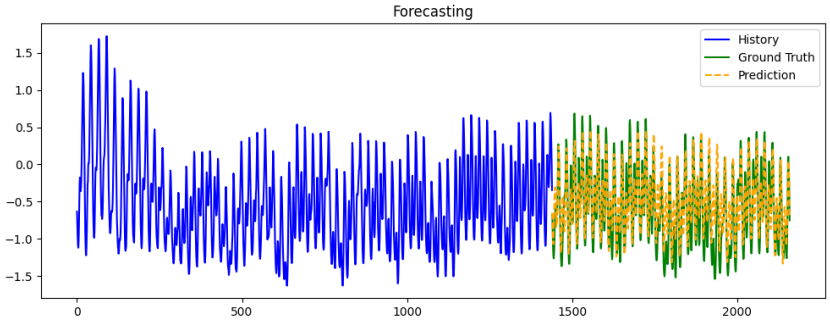


(d) Nonstationary Transformer

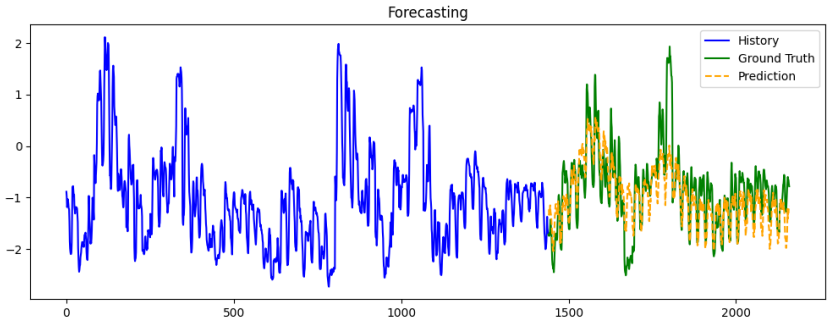
Figure 3: Forecasting visualizations of ETTh1 dataset predictions by (a) DiffKANformer, (b) TimesNet Wu et al. (2022), (c) Reformer Kitaev et al. (2020) and (d) Nonstationary Transformer Liu et al. (2022c)



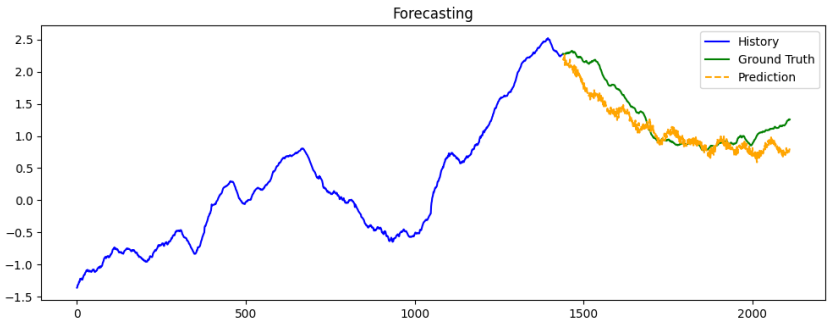
(a) ECL



(b) Caiso

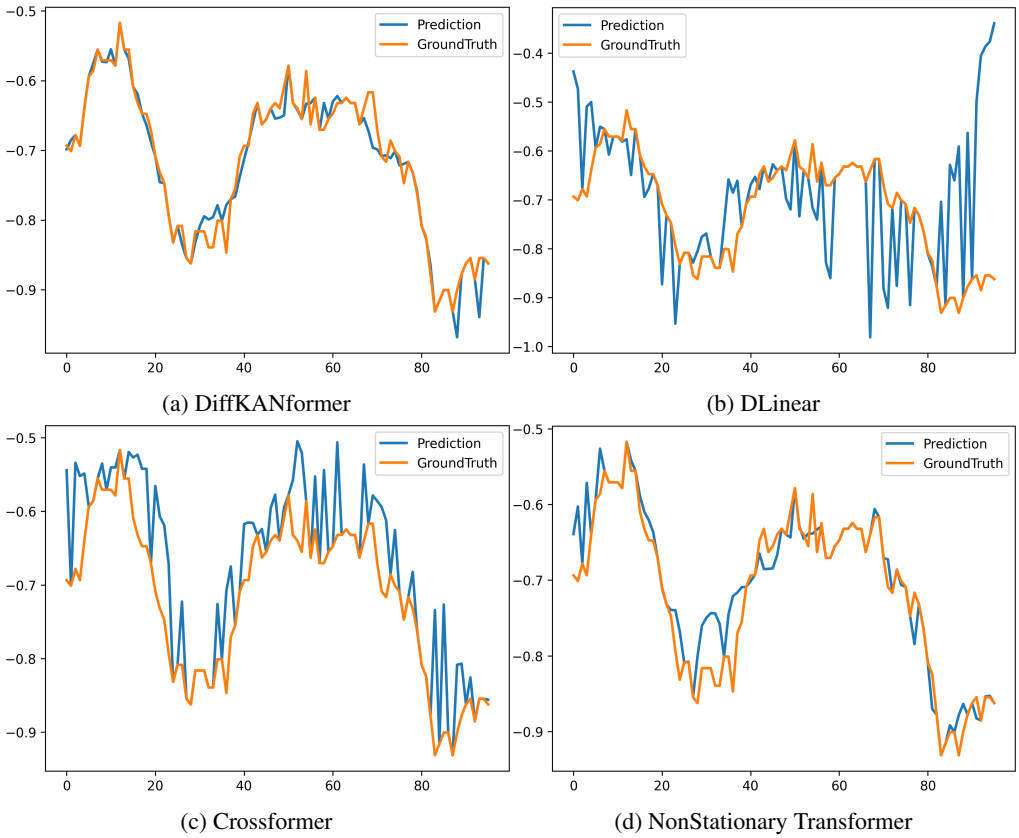


(c) Norpool



(d) Weather

Figure 4: Forecasting visualizations results of different datasets for DiffKANformer



(a) DiffKANformer (b) DLinear

(c) Crossformer (d) NonStationary Transformer

Figure 5: Imputation visualizations for ETTm1 dataset at 50% mask ratio for (a) Ours, (b) Dlinear (Zeng et al., 2023), (c) Crossformer (Zhang & Yan, 2023) and (d) NonStationary Transformer (Liu et al., 2022c)

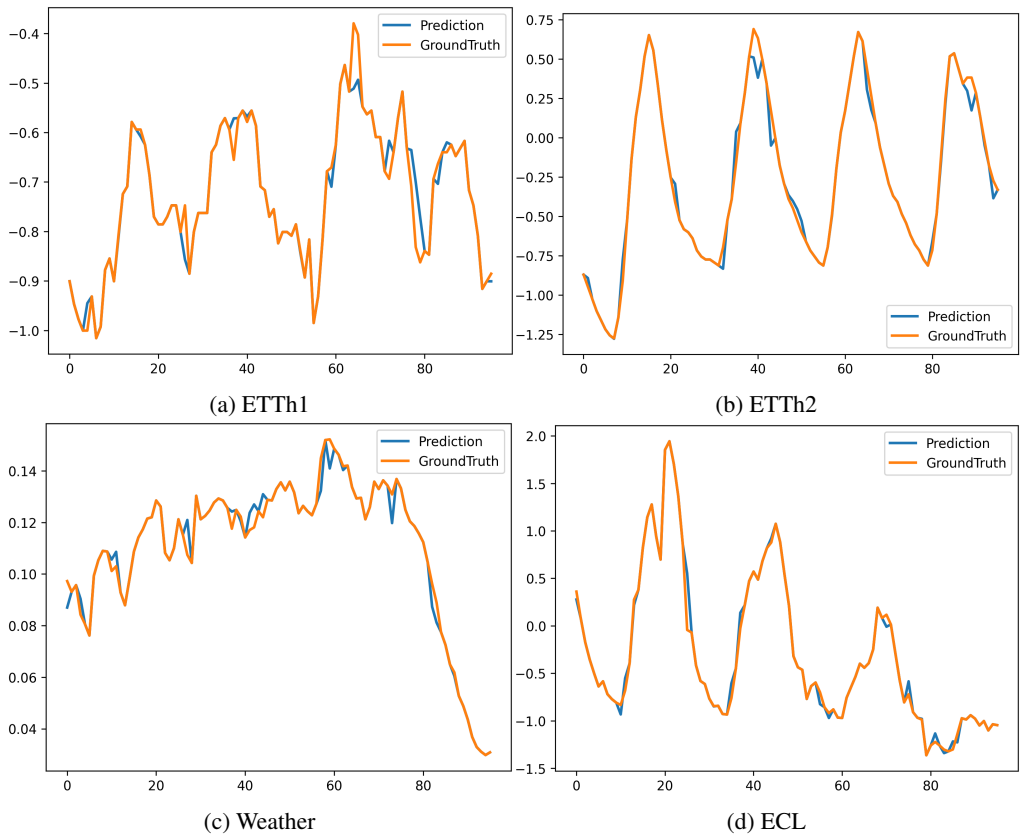
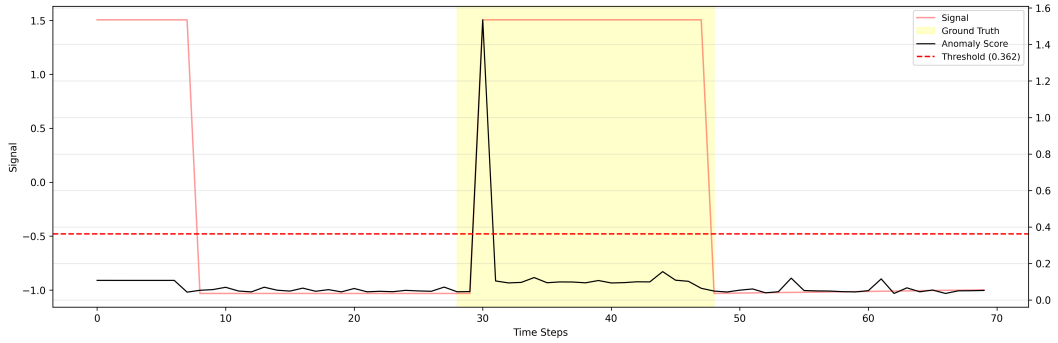
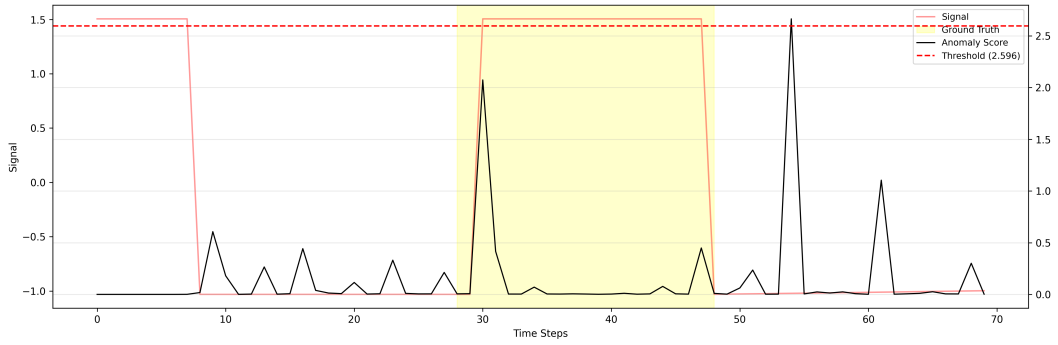


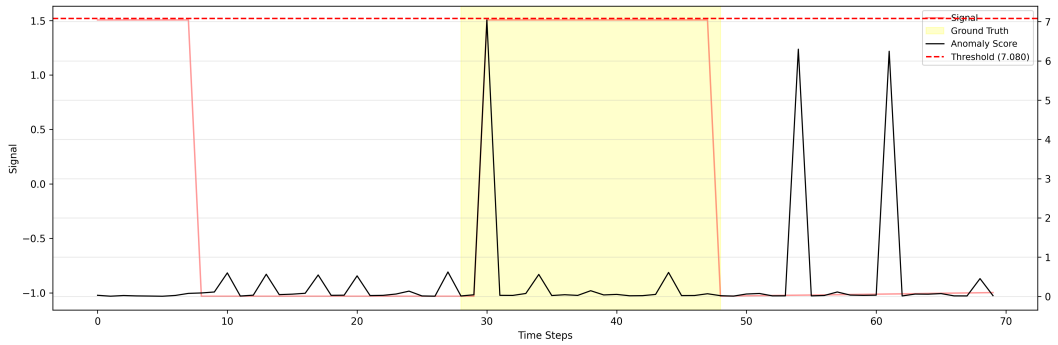
Figure 6: Imputation visualizations of DiffKANformer for ETTh1, ETTh2, Weather and ECL datasets at 25% mask ratio.



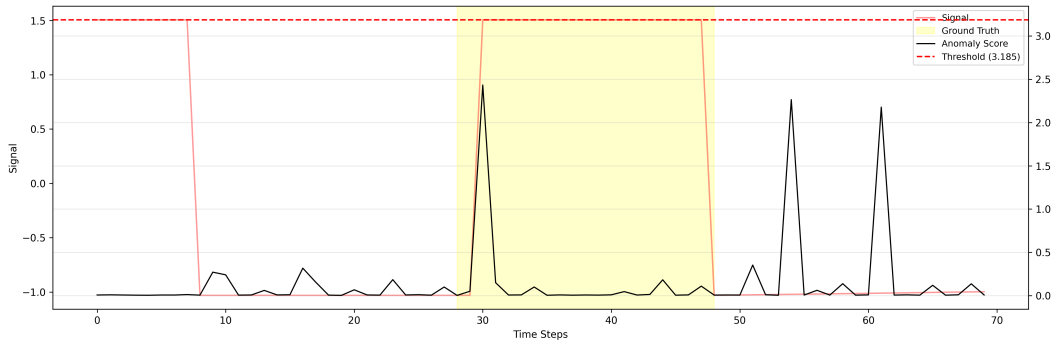
(a) DiffKANformer



(b) DLinear



(c) Crossformer



(d) NonStationary Transformer

Figure 7: Anomaly Detection visualizations for SMAP dataset for (a) Ours, (b) Dlinear (Zeng et al., 2023), (c) Crossformer (Zhang & Yan, 2023) and (d) NonStationary Transformer (Liu et al., 2022c)