Flow+Diff: Unifying Flow and Diffusion Representations for Multivariate Time Series Anomaly Detection

Van Kwan Zhi Koh **Songnan Lin** **Zhiping Lin** **Bihan Wen***

School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore

Abstract

Many existing multivariate time series anomaly detection methods formulate anomaly detection as a reconstruction problem, where detections are derived from reconstruction errors in the time domain. While effective for large deviations, reconstruction-based scoring struggles with subtle anomalies where only a small subset of features deviates slightly. In such cases, most features are well reconstructed, causing the overall reconstruction error to remain low and indistinguishable from normal data. To address these challenges, we propose Flow+Diff, a novel framework that detects anomalies via discrepancies in a latent space, where two complementary representations, induced by distinct modeling paradigms, diverge under anomalous inputs. One representation is obtained via a normalizing flow, which provides an invertible mapping that preserves temporal dependencies and captures inter-feature dependencies. The other is produced by a diffusion model that learns the distribution of normal samples in the latent space via a stochastic denoising process. When given anomalous inputs, the normalizing flow maps the inputs to latent representations that lie in low-probability regions of the learned latent distribution, while the diffusion model generates representations aligned with normal data distributions, resulting in a pronounced discrepancy between the two. We compute the anomaly score based on this latent space discrepancy, thereby shifting anomaly detection from the time domain to the latent space. Finally, the invertibility of the normalizing flow enables reconstruction in the time domain, facilitating interpretation of anomalous features. Extensive experiments across six benchmark datasets show that Flow+Diff achieves state-of-the-art performance on four datasets against competitive baselines. The code is publicly available at `https://github.com/vanerkz/FlowPlusDiff`

1 Introduction

Industries such as utilities [1, 2], transportation [3], and manufacturing machinery [4] generate vast amounts of high-dimensional time series data, contributing to the increasing availability of large-scale multivariate datasets. Consequently, multivariate time series anomaly detection (MTSAD) serves as a critical tool for identifying abnormal segments, enabling end-users to deploy preventive measures. Despite its importance, MTSAD remains highly challenging, as anomalies are often subtle and entangled within high-dimensional data with complex inter-feature dependencies [5–7]. Beyond

*Corresponding author. Email: bihan.wen@ntu.edu.sg

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

detection accuracy, identifying the features contributing to each anomalous segment is crucial for downstream analysis [8].

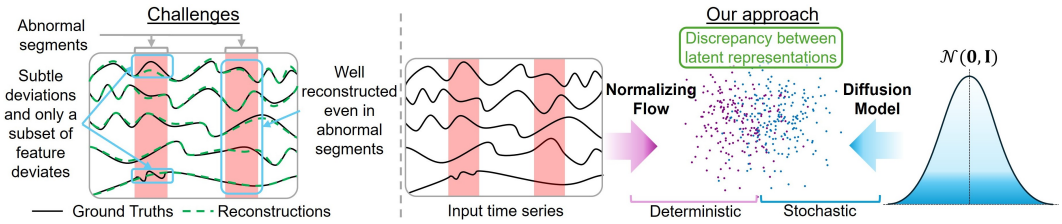


Figure 1: Challenges of reconstruction-based methods (left) versus our approach (right). Subtle anomalies, such as small deviations and sparse feature anomalies, yield low reconstruction errors and are difficult to detect. Moreover, reconstructions may remain accurate in anomalous segments, resulting in reconstruction errors indistinguishable from normal data. In contrast, our approach quantifies anomaly scores by transforming input time series into a shared latent space via normalizing flow and diffusion models, where the resulting representations diverge for anomalous inputs.

Deep learning methods have gained significant interest for MTSAD due to their ability to capture complex dynamics in large-scale multivariate data. In most approaches, models are trained solely on normal data, with anomalies observed only during inference. Consequently, many methods rely on reconstruction-based objectives with inductive biases that encourage learning normal patterns, leading to higher reconstruction errors for anomalous inputs [9–12]. As shown in Fig. 1 (left), relying solely on reconstruction errors for anomaly scoring is limited in capturing subtle anomalies. These include small or sparsely occurring feature deviations, where individual feature reconstructions remain accurate, resulting in low reconstruction errors indistinguishable from normal samples. Prior methods introduce weighting schemes that emphasize potentially anomalous segments [10, 11, 9, 13], and employ multi-granularity reconstruction objectives [11] and multi-scale reconstruction objectives [12], but they still rely on reconstruction errors, failing to address these challenges.

To address these challenges, we propose Flow+Diff, a framework for detecting anomalous time steps by quantifying discrepancies between latent representations learned by a normalizing flow and a diffusion model, as illustrated in Fig. 1 (right). The normalizing flow (NF) provides a bijective, likelihood-preserving representation through deterministic invertible mappings, where anomalous inputs are expected to lie in low-probability regions of the learned distribution. In contrast, the diffusion model performs iterative stochastic denoising conditioned on the input data, producing a representation shaped by the distribution of normal samples. To improve representation quality, we introduce Local Mean Augmentation, which decomposes the input into a temporally computed local mean component and a residual component, explicitly separating them to produce more locally consistent latent representations. The resulting representation is processed by a normalizing flow with a temporal-preserving feature-wise masking strategy that masks a subset of input features according to a predefined ratio while preserving temporal structure to encourage feature dependency learning. Given anomalous inputs, the normalizing flow and diffusion model produce divergent representations, which serve as the basis for anomaly scoring. Subtle anomalies can be detected through divergence between a deterministic latent transformation and a stochastic diffusion-based generative process, resulting in inconsistent representations under anomalous inputs. Finally, the invertibility of the NF enables reconstruction in the time domain, allowing anomalous features to be localized via reconstruction discrepancies and providing interpretability at the feature level.

- We propose Flow+Diff, a latent-space anomaly detection framework that identifies anomalies via discrepancies between representations learned by a normalizing flow and a diffusion model, improving sensitivity to subtle anomalies.
- We introduce a local mean augmentation strategy and a temporal-preserving feature-wise masking mechanism to enhance representation learning in the normalizing flow, resulting in latent spaces that exhibit local consistency and capture feature interdependencies.
- We conduct extensive experiments across multiple benchmark datasets to validate the effectiveness of our method and provide qualitative visualizations demonstrating the interpretability of detected anomalies.

2 Related Works

MTSAD Methods Classical methods, such as LOF [14], OC-SVM [15], and IF [16], are widely used but often struggle with large-scale or complex data. Deep learning approaches have significantly advanced MTSAD by capturing intricate temporal dynamics and inter-feature dependencies. Reconstruction-based methods, such as MEMTO [9], AT [10], TranAD [17], NSPR [11], and CrossAD [12], incorporate architectural constraints to encourage learning of normal patterns. Anomalies are then detected via reconstruction errors between inputs and their reconstructions, often enhanced by weighting mechanisms that emphasize anomalous regions. Density-based methods such as DAGMM [18] use latent representations and probabilistic density estimation for anomaly detection. Alternatively, deep one-class methods, such as THOC [19], DCdetector [20], and COUTA [21], learn latent representations of normal time series and detect anomalies as deviations from the learned normality. However, these approaches rely on a single latent modeling paradigm, which may cause subtle anomalies to remain close to normal patterns in the learned latent space after transformation.

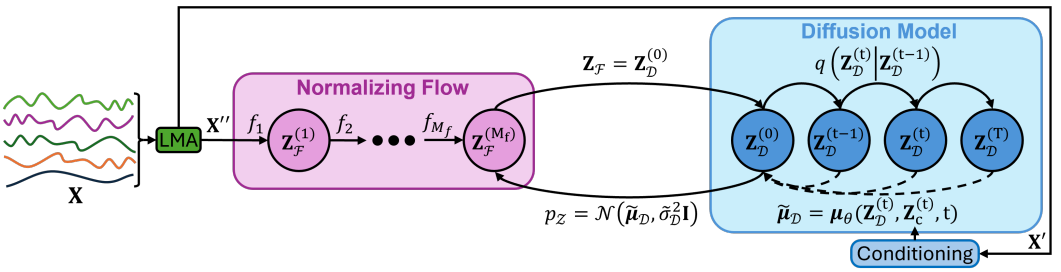
Normalizing flow and diffusion model MTSAD Generative models have gained interest because of their ability to capture complex data distributions, making them particularly suitable for MTSAD [22–26]. NFs, such as GANF [22] and MTGFlow [23], learn expressive distributions while modeling feature inter-dependencies through graphs, self-attention, and recurrent neural network embeddings. Similar to deep one-class approaches, NF-based methods rely on a single modeling paradigm, which may limit separability when anomalies are subtle and overlap with normal patterns in the latent space. In contrast, diffusion models stochastically map temporal features to latent space and reconstruct them to compute reconstruction errors for MTSAD [27–29, 6]. Examples include ImDiffusion [28], which uses masked inputs to learn temporal dependencies, TimeADDM [6], which applies diffusion in latent space, and TWavefussion [29], which combines transformer encoders with wavelet-based decomposition. DiffAD [26] reframes detection as an imputation task to handle dense anomalies. However, these methods rely on reconstruction errors to identify anomalous segments, thereby inheriting the limitations of reconstruction-based approaches. Different from prior methods, we replace the NF's base distribution with a data-adaptive distribution learned via a diffusion model. This yields complementary latent representations, where the NF produces a deterministic representation and the diffusion model produces a stochastic representation, each capturing distinct characteristics such as feature dependencies and distributional shifts in the time series, enabling the detection of anomalous segments.

3 Method

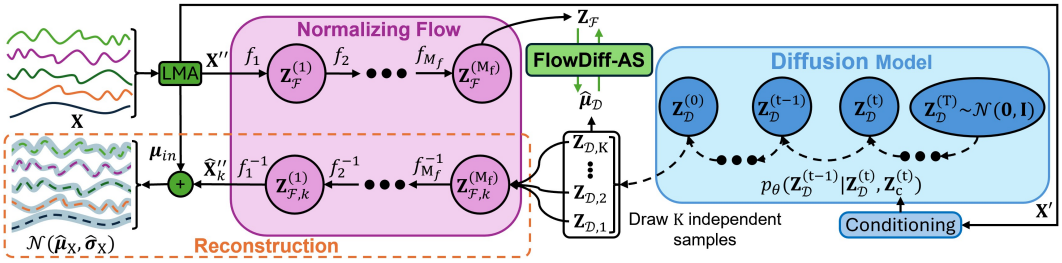
Given a multivariate time series of length I, we segment it into overlapping windows of length L with a stride of st, which serve as inputs to our model. The resulting segmented time series is denoted as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L]^\top \in \mathbb{R}^{L \times d_x}$, where each $\mathbf{x}_t \in \mathbb{R}^{d_x}$ and d_x denote the number of features. Each segmented window is associated with a label vector $\mathbf{Y} = [y_1, y_2, \ldots, y_L]$, where $y_t \in \mathbb{Z}_2$ indicates whether the t-th time step is anomalous ($y_t = 1$) or normal ($y_t = 0$). The model objective is to derive the anomaly score $\boldsymbol{\mathcal{A}} \in \mathbb{R}^L$ that predicts the label vector $\hat{\mathbf{Y}} \in \mathbb{R}^L$. Predictions are computed for all segmented windows of the time series, resulting in an anomaly score vector $\boldsymbol{\mathcal{A}}_{full} \in \mathbb{R}^I$ for all the time steps.

3.1 Model Overview

An overview of our model is shown in Fig. 2. Flow+Diff harnesses NF and diffusion model to capture complementary latent representations for MTSAD. First, **Localized Mean Augmentation (LMA)** subtracts the local mean $\boldsymbol{\mu}_{\text{in}}$ computed along the temporal dimension from the input to obtain the centered input time series $\mathbf{X}^{'}$, which is then concatenated with $\boldsymbol{\mu}_{in}$ to form the augmented input $\mathbf{X}''$. The **NF** then transforms $\mathbf{X}''$ into a flow-derived latent representation $\mathbf{X}'' \overset{\mathcal{F}}{\longmapsto} \mathbf{Z}_{\mathcal{F}}$, where $\mathcal{F} = f_{M_f} \circ \cdots \circ f_1$ denotes a composition of M_f invertible transformations. Each f_i is implemented using REALNVP [30], yielding a bijective mapping that enables exact inverse reconstruction. The **diffusion model** learns to reconstruct $\mathbf{Z}_{\mathcal{F}}$, producing a diffusion-derived latent representation $\mathbf{Z}_{\mathcal{D}}$. During training (Fig. 2a), the mean $\tilde{\boldsymbol{\mu}}_{\mathcal{D}}$ of the NF base distribution is approximated via the reparameterized forward diffusion process $q(\cdot)$ using the predicted diffusion noise $\hat{\boldsymbol{\epsilon}}$ in Eq. (8), and combined with a data-independent isotropic variance $\tilde{\sigma}^2_{\mathcal{D}}\mathbf{I}$. During inference (Fig. 2b), K



(a) Training procedure



(b) Inference procedure

Figure 2: Overview of the training (a) and inference (b) procedures. The NF maps the augmented input time series $\mathbf{X}''$ to a latent representation $\mathbf{Z}_{\mathcal{F}}$. **Training**: the base distribution is parameterized by a diffusion-estimated mean $\tilde{\boldsymbol{\mu}}_{\mathcal{D}}$ and a learnable isotropic variance $\tilde{\sigma}^2_{\mathcal{D}}$. **Inference**: the diffusion model is sampled to obtain $\{\mathbf{Z}_{\mathcal{D},k}\}_{k=1}^{K}$, and the mean latent representation $\hat{\boldsymbol{\mu}}_{\mathcal{D}}$ is computed. The FlowDiff-AS module measures the discrepancy between $\mathbf{Z}_{\mathcal{F}}$ and $\hat{\boldsymbol{\mu}}_{\mathcal{D}}$. Finally, the reconstruction distribution is estimated by passing $\{\mathbf{Z}_{\mathcal{D},k}\}_{k=1}^{K}$ through the inverse normalizing flow.

diffusion-derived latent samples $\{\mathbf{Z}_{\mathcal{D},k}\}_{k=1}^{K}$ are generated via the reverse diffusion process p_θ, conditioned on the centered input time series $\mathbf{X}^{'}$. **Flow+Diffusion Anomaly Scoring (FlowDiff-AS)** measures the discrepancy between $\mathbf{Z}_{\mathcal{F}}$ and $\{\mathbf{Z}_{\mathcal{D},k}\}_{k=1}^{K}$ to compute anomaly scores for identifying anomalous segments. Finally, inverting the diffusion-derived latent representations $\{\mathbf{Z}_{\mathcal{D},k}\}_{k=1}^{K}$ through the NF and adding back $\boldsymbol{\mu}_{in}$, we recover the original input scale to estimate the mean $\hat{\boldsymbol{\mu}}_{\mathbf{X}}$ and standard deviation $\hat{\boldsymbol{\sigma}}_{\mathbf{X}}$ of the reconstructed time series.

3.2 Localized Mean Augmentation

Typical NFs require the input and latent dimensionality to be identical to preserve bijectivity. To operate under this constraint while improving representation quality, we propose Local Mean Augmentation (LMA), which reparameterizes the input into local mean and residual components. This decomposition separates local temporal variations while preserving the local window scale, leading to more locally consistent latent representations. The LMA module is defined as follows:

$$\boldsymbol{\mu}_{in} = \frac{1}{L}\sum_{t=1}^{L}\mathbf{x}_t, \qquad \mathbf{x}'_t = \mathbf{x}_t - \boldsymbol{\mu}_{in}, \quad \forall t = 1, \ldots, L. \tag{1}$$

The resulting matrix $\mathbf{X}' = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_L\}$ is then concatenated with $\boldsymbol{\mu}_{in}$ repeated along the L times, obtaining

$$\mathbf{X}'' = \left[\mathbf{X}',\ [\boldsymbol{\mu}_{in}, \boldsymbol{\mu}_{in}, \ldots, \boldsymbol{\mu}_{in}]^T\right] \in \mathbb{R}^{L \times 2d_x}, \tag{2}$$

where $[\cdot, \cdot]$ represents feature-wise concatenation.

3.3 Normalizing Flow

The NF defines a bijective mapping that transforms the time series into a latent representation while remaining invertible for reconstruction, which is useful for interpreting feature-level contributions during anomalous segments. Given the augmented input, the density is

$p_{\mathcal{X}''}(\mathbf{X}'') = p_{\mathcal{Z}}\left(\mathcal{F}(\mathbf{X}'')\right)\left|\det J_{\mathcal{F}}(\mathbf{X}'')\right|$, where $p_{\mathcal{Z}}$ is the base distribution. Standard NFs assume a fixed Gaussian prior, this limits flexibility when aligning with complementary latent representations induced by diffusion models. We instead define $p_{\mathcal{Z}}(\mathbf{z}) = \mathcal{N}\big(\tilde{\boldsymbol{\mu}}_{\mathcal{D}}, \tilde{\sigma}^2_{\mathcal{D}}\mathbf{I}\big)$, where $\tilde{\boldsymbol{\mu}}_{\mathcal{D}} \in \mathbb{R}^{L\times 2d_x}$ is estimated from the diffusion model. The covariance $\tilde{\sigma}^2_{\mathcal{D}}\mathbf{I}$ is learnable and parameterized exponentially to ensure positivity, providing adaptive scaling to account for the stochasticity of diffusion-based sampling. For the NF transformation $\mathcal{F} = f_{M_f} \circ \cdots \circ f_1$, with M_f layers, we follow the REALNVP [30] procedure. In our approach, affine coupling layers are applied along the feature dimension with masking while preserving the temporal structure.

Temporal-preserving feature-wise masking At each transformation layer, the latent representation is masked along the feature dimension to encourage feature dependency learning, while the temporal dimension remains fully preserved. The affine coupling is defined as:

$$\begin{aligned}\mathbf{Z}^{(l_f)}_{\mathcal{F},A} &= \mathbf{Z}^{(l_f-1)}_{\mathcal{F}} \odot \mathbf{M}^{(l_f)} \\ \mathbf{Z}^{(l_f)}_{\mathcal{F},B} &= \mathbf{Z}^{(l_f-1)}_{\mathcal{F}} \odot \left(1 - \mathbf{M}^{(l_f)}\right)\end{aligned} \tag{3}$$

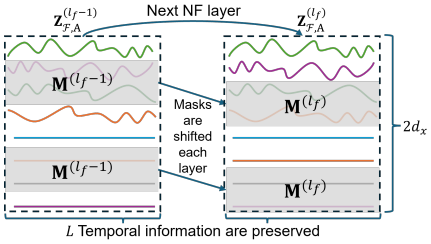


Figure 3: NF layer masking. Feature-wise masking while preserving temporal information.

where $\odot$ denotes element-wise multiplication, l_f denotes the index of the NF transformation layer and $\mathbf{M}^{(l_f)}$ denotes the masking at the l_f layer. As illustrated in Fig. 3, the mask $\mathbf{M}^{(l_f)} \in \{0,1\}^{L\times 2d_x}$ is constructed by applying a masking ratio ρ along the feature dimension $2d_x$, with masked feature positions shifted across layers to expose different feature subsets, thereby providing controllable expressiveness. The alternating masking pattern ensures that each transformation layer operates only on a subset of features at a time.

Affine coupling layer To capture temporal dependencies, we adopt a Transformer encoder [31], denoted as TransEnc$(\cdot)$, composed of n_f multi-head self-attention layers with $atthead_f$ attention heads. The output of TransEnc$(\cdot)$ is subsequently processed by MLPs to parameterize the scale and translation functions $\mathbf{S}^{(l_f)}$ and $\mathbf{T}^{(l_f)}$. Thereafter, the unmasked component $\mathbf{Z}^{(l_f)}_{\mathcal{F},A}$ is used to compute the translation and scale parameters as follows:

$$h^{(l_f)}_{\mathcal{F}} = \text{TransEnc}\left(\mathbf{Z}^{(l_f)}_{\mathcal{F},A}\right), \quad \mathbf{T}^{(l_f)} = \text{MLP}\left(h^{(l_f)}_{\mathcal{F}}\right), \quad \mathbf{S}^{(l_f)} = \tanh\left(\text{MLP}\left(h^{(l_f)}_{\mathcal{F}}\right)\right). \tag{4}$$

where the MLP denotes a two-layer multilayer perceptron with ReLU activation. Following the REALNVP framework [30], the translation parameter of the l_f-th layer, $\mathbf{T}^{(l_f)} \in \mathbb{R}^{L\times 2d_x}$, and the scale parameter $\mathbf{S}^{(l_f)} \in \mathbb{R}^{L\times 2d_x}$, are used to parameterize the affine coupling transformation. Then, the flow-derived latent representation is obtained via a sequence of NF transformations:

$$\mathbf{Z}^{(M_f)}_{\mathcal{F}} = \text{Asig}\left(\mathcal{F}(\mathbf{X}'')\right) = \text{Asig} \circ f_{M_f} \circ f_{M_f-1} \circ \cdots \circ f_1(\mathbf{X}''). \tag{5}$$

Here, the affine-sigmoid function is defined as $\text{Asig}(z) = 2\sigma(z) - 1 = \tanh(z/2)$, where $\sigma(\cdot)$ denotes the sigmoid function. The Asig function constrains $\mathbf{Z}_{\mathcal{F}}$ to $(-1, 1)$, aligning the latent spaces of the NF and diffusion model.

3.4 Diffusion Model

The diffusion model [32] is introduced to provide a more expressive characterization of the NF's base distribution. It consists of a forward process that progressively injects Gaussian noise into the input and a reverse denoising process parameterized by a denoising network $\boldsymbol{\epsilon}_\theta(\cdot)$ to reconstruct the original representation. Detailed descriptions of the diffusion process are provided in Appendix B.2.

Conditioning the diffusion process Time series are inherently non-stationary, so conditioning the diffusion model is crucial to maintaining representations consistent with the present temporal context [28, 26]. Therefore, we condition the diffusion process on a conditional representation $\mathbf{Z}^{(t)}_c$, which is derived using a transformer encoder [31], defined as:

$$\mathbf{Z}^{(t)}_c = \text{Asig}(\text{TransEncTime}(\mathbf{X}', t)). \tag{6}$$

Here, $\mathrm{TransEncTime}(\mathbf{X}', t)$ denotes the Transformer encoder used to condition the diffusion process, consisting of n_d layers and $atthead_d$ attention heads, where the diffusion step t is first mapped to a sinusoidal embedding and added to the input embedding. The Asig function is used to ensure that the scale of the conditional representation is consistent with the diffusion-derived latent representation $\mathbf{Z}_{\mathcal{D}}^{(t)}$. The representations are concatenated and fed into a one-dimensional U-Net, with diffusion step embeddings injected into each residual block [33]. This network, denoted as the denoising network $\boldsymbol{\epsilon}_\theta(\cdot)$, predicts the denoising noise $\hat{\boldsymbol{\epsilon}}$, which is required for the reverse denoising process:

$$\hat{\boldsymbol{\epsilon}} = \boldsymbol{\epsilon}_\theta(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}, t) = \mathrm{UNet}\big([\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}], t\big), \tag{7}$$

where $[\cdot, \cdot]$ denotes concatenation along the feature dimension, resulting in the feature dimension from $2d_x$ to $4d_x$.

Approximating the mean of the NF's base distribution Typically, the denoising network in the diffusion model is trained to minimize the reconstruction losses between the sampled noise $\boldsymbol{\epsilon}$ and the predicted noise $\hat{\boldsymbol{\epsilon}}$. However, in our setting, the diffusion model is used to provide a more expressive modeling of the NF's base distribution. Therefore, based on the forward diffusion process, $\mathbf{Z}_{\mathcal{D}}^{(t)} = \sqrt{\bar{\alpha}_t}\,\mathbf{Z}_{\mathcal{F}} + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\bar{\alpha}_t = \prod_{s=0}^{t}(1-\beta_s)$. The noise schedule $\{\beta_s\}_{s=0}^{T}$ is defined by a linear scheduler ranging from $\beta_{\min}$ to $\beta_{\max}$. Rearranging the equation and replacing $\boldsymbol{\epsilon}$ with its prediction $\hat{\boldsymbol{\epsilon}}$ yields an estimate of $\mathbf{Z}_{\mathcal{F}}$:

$$\tilde{\boldsymbol{\mu}}_{\mathcal{D}} = \boldsymbol{\mu}_\theta\Big(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}, t\Big) = \frac{\mathbf{Z}_{\mathcal{D}}^{(t)} - \sqrt{1-\bar{\alpha}_t}\,\hat{\boldsymbol{\epsilon}}}{\sqrt{\bar{\alpha}_t}} \approx \mathbf{Z}_{\mathcal{F}}. \tag{8}$$

3.5 Training Objective

The NF and diffusion model are jointly optimized, where the diffusion model parameterizes the prior distribution over the NF latent space. The NF learns an invertible mapping from the input to the latent space, while the diffusion model learns the probability distribution of these latent representations through conditional denoising. Thus, the NF determines the latent representation of each sample, whereas the diffusion model estimates its probability density. In Eq. (9), while the learned diffusion prior increases the flexibility of the latent likelihood term $\log p_{\mathcal{Z}}(\mathcal{F}(\mathbf{X}''))$, any attempt to collapse multiple inputs into similar latent representations would reduce the local volume change, driving $|\det J_{\mathcal{F}}(\mathbf{X}'')|$ towards zero. Consequently, $\log|\det J_{\mathcal{F}}(\mathbf{X}'')|$ becomes increasingly negative, penalizing local volume contraction and potentially counteracting gains in latent likelihood. Therefore, the Jacobian term explicitly penalizes local volume contraction, discouraging trivial latent collapse.

Our model extends the standard NF objective by replacing the fixed base distribution with a diffusion-parameterized Gaussian prior. Specifically, we maximize the expected log-likelihood of $p_{\mathcal{X}''}(\mathbf{X}'')$:

$$\mathcal{L} = \mathbb{E}_{\mathbf{X}'' \sim p_{\mathcal{X}''}}\Big[\log p_{\mathcal{Z}}(\mathcal{F}(\mathbf{X}'')) + \log\Big|\det J_{\mathcal{F}}(\mathbf{X}'')\Big|\Big], \tag{9}$$

where the base distribution is defined as a diffusion-conditioned Gaussian: $p_{\mathcal{Z}}(\mathbf{z}) = \mathcal{N}\big(\tilde{\boldsymbol{\mu}}_{\mathcal{D}}, \tilde{\sigma}_{\mathcal{D}}^2\mathbf{I}\big)$. The mean $\tilde{\boldsymbol{\mu}}_{\mathcal{D}}$ is obtained from the denoising network $\boldsymbol{\mu}_\theta(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{X}', t)$, where $t \sim \mathcal{U}(0, T)$ denotes a uniformly sampled diffusion step. The second term in (9) incorporates an additional log-determinant term corresponding to the Asig transformation in the final invertible layer. The derivation is provided in Appendix C.1. Overall, the parameters of the normalizing flow $\mathcal{F}$ and the denoising network $\boldsymbol{\mu}_\theta$ are jointly optimized by maximizing the objective in (9).

3.6 Flow+Diffusion Anomaly Scoring

Instead of relying on temporal reconstruction errors, we propose Flow+Diffusion Anomaly Scoring (FlowDiff-AS), which measures the discrepancy between the latent representations produced by the NF and diffusion models. Their complementary learning objectives provide distinct views of the normal data: NF learns an invertible mapping that enables exact likelihood evaluation, while diffusion learns the normal data distribution through denoising. Following the standard unsupervised TSAD setting, both models are trained exclusively on normal data [11, 9]. We therefore expect their

latent representations to be consistent for normal samples. For anomalous samples, NF preserves the input-specific latent representation, whereas diffusion drives the representation toward the learned normal distribution. The resulting discrepancy serves as the anomaly score.

Generating diffusion-derived latent representations We adopt Denoising Diffusion Implicit Models (DDIM) [32] for efficient sampling, as they skip intermediate steps and enable substantially faster inference. We draw K stochastic samples $\{\mathbf{Z}_{\mathcal{D},k}\}_{k=1}^{K}$ of the diffusion-derived latent representation $\mathbf{Z}_{\mathcal{D}}$ via the reverse diffusion process. This procedure is performed iteratively from $t = T$ to $t = 0$ with a stride N to obtain a single diffusion-derived latent sample $\mathbf{Z}_{\mathcal{D},k}^{(0)}$. Details are provided in Appendix B.2. Similar to the final layer of the NF, an Asig function is applied to the latent representation, $\mathbf{Z}_{\mathcal{D},k} = \mathrm{Asig}\left(\mathbf{Z}_{\mathcal{D},k}^{(0)}\right)$, to ensure bounded values. The mean of these samples is computed as:

$$\hat{\boldsymbol{\mu}}_{\mathcal{D}} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{Z}_{\mathcal{D},k}, \tag{10}$$

where $\mathbf{Z}_{\mathcal{D},k} \in \mathbb{R}^{L\times 2d_x}$ denotes the k-th sample. The resulting $\hat{\boldsymbol{\mu}}_{\mathcal{D}}$ has the same dimensionality and is obtained via element-wise averaging across the K samples.

Computing anomaly scores With the estimated mean $\hat{\boldsymbol{\mu}}_{\mathcal{D}}$, we compare it with the flow latent representation $\mathbf{Z}_{\mathcal{F}} \in \mathbb{R}^{L\times 2d_x}$. The element at time index i and feature index j is denoted as $z_{\mathcal{F},i,j}$. The resulting discrepancy score is defined as:

$$\text{DiscrepancyScore}_i = \sum_{j}\left(z_{\mathcal{F},i,j} - \hat{\mu}_{\mathcal{D},i,j}\right)^2. \tag{11}$$

Here, DiscrepancyScore $\in \mathbb{R}^L$ denotes the score vector for a single window. Applying this procedure to all segments yields anomaly scores for the full time series of length I. These scores are subsequently smoothed via a rolling average with window size L, producing the final anomaly score vector $\boldsymbol{\mathcal{A}}_{full} \in \mathbb{R}^I$.

Estimating the reconstruction distribution Each sampled diffusion-derived latent representation $\mathbf{Z}_{\mathcal{D},k}$ is passed through the inverse flow $\mathcal{F}^{-1}$ for all K samples, producing reconstructed inputs whose distribution is modeled as a Gaussian with mean $\hat{\boldsymbol{\mu}}_X$ and standard deviation $\hat{\boldsymbol{\sigma}}_X$. Details of the reconstruction process are provided in Appendix C.2.

4 Experiments

4.1 Experimental Setup

Setting We conduct five independent runs with different random seeds and report the average performance. Experiments are performed on six publicly available MTSAD datasets, including SMD [34], SMAP and MSL [35], SWaT [36], PSM [37], and WADI [38]. Prior studies have shown that discrete variables can negatively affect anomaly detection performance [39, 12]. Following [12], we exclude discrete features in the SMAP and MSL datasets and preserve feature dimensionality by filling the corresponding entries with zeros. For completeness, we report the mean, standard deviation, and additional evaluation metrics in Appendix E.1.

Metrics In the literature, a common evaluation strategy is point-adjustment [10, 9], under which many models report strong performance. However, recent work shows that even simple adversarial algorithms can perform well under this protocol [39, 40]. Therefore, we report evaluation metrics without point adjustment, including the Area Under the Receiver Operating Characteristic Curve (ROC) and the F1-score (F1), where thresholds are selected to maximize the F1-score [11]. We further incorporate a recent robust metric, the Volume Under the Surface for the Precision–Recall curve (V-PR) [40, 41], which demonstrates improved robustness and reduced sensitivity to detection delays. Additional evaluation metrics are reported in Appendix E.1 to provide a comprehensive comparison.

Baselines To demonstrate the effectiveness of our model, we compare it against a comprehensive set of baselines, ranging from classical methods to recent state-of-the-art approaches. Classical methods include IF [16], LOF [14], and OC-SVM [15], which are widely used for multivariate time series anomaly detection (MTSAD). Deep one-class methods include THOC [19] and DCdetector [20]. Density-based methods include DAGMM [18]. Reconstruction-based approaches include AT [10], MEMTO [9], NSPR [11], and CrossAD [12]. We also consider generative approaches such as GANF [22] and DiffAD [26].

4.2 Main Results

Table 1 presents the performance comparison between the baseline methods and the proposed method. The best results are highlighted in **bold**, while the second-best results are underlined. The proposed model achieves the best ROC and F1 scores on four of six datasets and the best V-PR scores on three, demonstrating overall effectiveness. Classical methods, including IF, OC-SVM, and LOF, represent earlier approaches and remain competitive with more recent state-of-the-art techniques. Deep one-class methods performance gains remain modest. Models such as DCdetector, AT, MEMTO, and DiffAD are typically evaluated using point-adjustment protocols, which favor segment-level detection over precise anomaly localization. NSPR achieves performance comparable to the proposed model, while our method consistently outperforms NSPR on the MSL, SMAP, PSM, and WADI datasets. Overall, the proposed model consistently achieves higher ROC and F1 scores on four of six datasets.

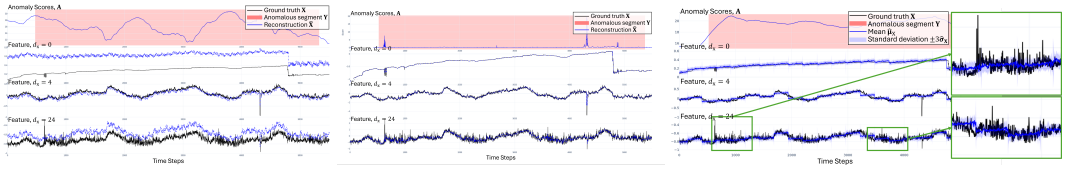
Table 1: Results of our proposed model and baseline methods.

	SMD			MSL			SMAP			SWaT			PSM			WADI		
Methods	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR
IF	0.664	0.182	0.126	0.604	0.253	0.159	0.634	0.305	0.166	0.843	0.741	0.511	0.684	0.493	0.467	0.734	0.313	0.183
OC-SVM	0.590	0.171	0.100	0.526	0.243	0.164	0.377	0.227	0.102	0.884	0.788	**0.677**	0.565	0.460	0.356	0.482	0.120	0.059
LOF	0.645	0.124	0.089	0.556	0.217	0.143	0.623	0.284	0.174	0.888	0.780	0.610	0.728	0.583	0.432	0.513	0.261	0.113
THOC	0.651	0.213	0.136	0.579	0.230	0.163	0.421	0.230	0.112	0.834	0.760	0.668	0.684	0.503	0.457	0.488	0.121	0.058
DCdetector	0.498	0.080	0.062	0.479	0.227	0.132	0.495	0.227	0.135	0.494	0.217	0.124	0.501	0.435	0.295	0.497	0.109	0.061
DAGMM	0.714	0.215	0.151	0.583	0.235	0.160	0.474	0.243	0.128	0.861	0.744	0.546	0.651	0.473	0.456	0.505	0.196	0.089
AT	0.407	0.080	0.054	0.472	0.191	0.119	0.585	0.257	0.164	0.493	0.216	0.124	0.470	0.434	0.293	0.497	0.109	0.061
MEMTO	0.502	0.080	0.063	0.508	0.196	0.140	0.505	0.227	0.135	0.656	0.401	0.245	0.503	0.434	0.307	0.430	0.109	0.054
GANF	0.565	0.184	0.127	0.601	0.307	0.217	0.462	0.261	0.148	0.842	0.759	0.527	0.603	0.534	0.415	0.507	0.262	0.105
DiffAD	0.586	0.126	0.330	0.520	0.213	0.169	0.384	0.227	0.220	0.329	0.217	0.143	0.557	0.448	0.367	0.358	0.109	0.087
NSPR	**0.834**	**0.375**	**0.369**	0.750	0.409	**0.327**	0.590	0.278	0.187	**0.913**	0.826	0.663	0.803	0.634	0.499	0.507	0.150	0.065
CrossAD	0.758	0.240	0.218	0.782	0.338	0.312	0.560	0.277	0.153	0.847	0.717	0.570	0.646	0.465	0.550	0.444	0.111	0.056
Ours	0.745	0.224	0.105	**0.811**	**0.413**	0.304	**0.701**	**0.349**	**0.232**	0.907	**0.836**	0.617	**0.837**	**0.665**	**0.557**	**0.809**	**0.629**	**0.524**

Feature interpretability NSPR and CrossAD leverage multiple reconstructions to obtain robust anomaly scores. Figure 4 compares reconstructions for a subset of features across the baselines and our method. Although the PSM dataset contains 25 features, anomalies may be localized to only a small subset of dimensions. In the highlighted segment, our method assigns higher anomaly scores despite subtle deviations from the ground truth. In contrast, NSPR produces larger reconstruction errors, but many features exhibit a consistent offset across the sequence, which can uniformly inflate reconstruction errors and reduce their discriminative value. Such errors may arise either from model mismatch or from genuinely anomalous patterns, making reconstruction error alone difficult to interpret. Our method estimates the reconstruction distribution, enabling feature-level deviations to be examined beyond the magnitude of reconstruction error. For interpretation, anomalous segments are first identified from elevated anomaly scores. Within these segments, the ground-truth feature values are compared against their estimated reconstruction distributions, where larger deviations indicate potential contributing features. Following this procedure, Figure 4c shows elevated anomaly scores around time steps 1250–1750, 2750–3000, and from approximately 3750 onward. Within these anomalous segments, feature 108 exhibits the largest deviation from its estimated reconstruction distribution and is therefore identified as a potential contributing feature. Reconstructions for all features are provided in Appendix E.3.

4.3 Model analysis

Model ablation study We analyze the contribution of each component via ablation studies (Table 2). Details of the model variant settings, along with additional experimental results and visualizations, are



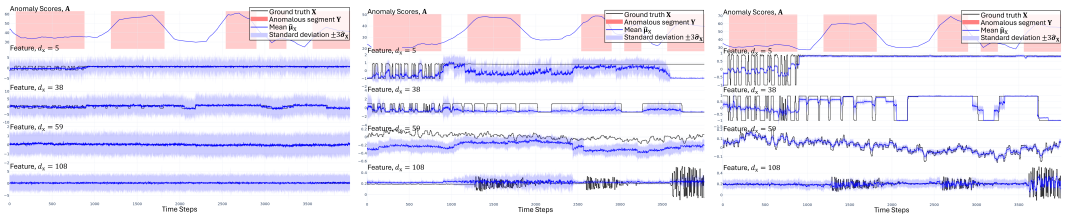
(a) NSPR (b) CrossAD (c) Ours

Figure 4: Potential subtle anomaly patterns in PSM

provided in Appendix F.1. The **Flow-Only** variant removes the diffusion model and uses a standard Gaussian base distribution, achieving competitive ROC performance. However, it lacks meaningful distributional estimates, thereby limiting interpretability (Fig. 5a). **Flow+Enc** replaces the diffusion model with a transformer encoder, resulting in reduced performance and a loss of the ability to model predictive uncertainty due to its deterministic nature. **MLP-Flow** variant replaces the transformer with an MLP, resulting in degraded performance, underscoring the importance of the model in preserving temporal structure. **Without Asig (wo-Asig)** shows the largest drop, demonstrating the necessity of bounded transformations for stable training and reliable scoring. **Without Conditioning (wo-Cond)** removes the conditioning input from the denoising network, resulting in only a slight performance drop. One possible intuition is that the bijective NF avoids many-to-one mappings between inputs and latent representations, allowing anomalous inputs to induce informative discrepancies between NF and diffusion representations for anomaly scoring. Nevertheless, conditioning further improves detection performance and stability, as evidenced by the lower standard deviations across random seeds in the Appendix (Table 9). **Without LMA (wo-LMA)** removes the LMA module, improving performance on WADI but degrading reconstruction quality (Fig. 5b), highlighting a trade-off between detection performance and reconstruction quality.

Table 2: Ablation study of model and anomaly scoring variants

		Model variants						Anomaly scoring variants				
Dataset	Metric	Flow-Only	Flow+Enc	MLP-Flow	wo-Asig	wo-Cond	wo-LMA	MSE($\hat{\mathbf{X}}$)	wo-Avg	NF-LL	Lat-LL	Ours
MSL	ROC	0.740	0.689	0.627	0.468	0.765	0.707	0.793	0.704	0.809	0.808	**0.811**
	F1	0.344	0.316	0.314	0.201	0.370	0.367	0.390	0.307	0.405	0.405	**0.413**
	V-PR	0.250	0.203	0.180	0.114	0.270	0.233	0.288	0.227	0.302	0.302	**0.304**
PSM	ROC	0.835	0.834	0.741	0.569	0.830	0.750	0.604	0.831	0.835	**0.837**	**0.837**
	F1	0.646	0.643	0.542	0.460	0.648	0.564	0.471	0.622	0.654	0.655	**0.665**
	V-PR	0.578	0.567	0.515	0.378	0.566	0.498	0.464	**0.688**	0.557	0.565	0.557
WADI	ROC	0.808	0.798	0.795	0.626	0.803	0.808	0.726	0.804	0.808	**0.817**	0.809
	F1	0.587	0.631	0.583	0.283	0.618	**0.638**	0.249	0.628	0.621	0.631	0.626
	V-PR	0.481	0.522	0.447	0.145	0.494	0.534	0.183	**0.567**	0.528	0.534	0.524

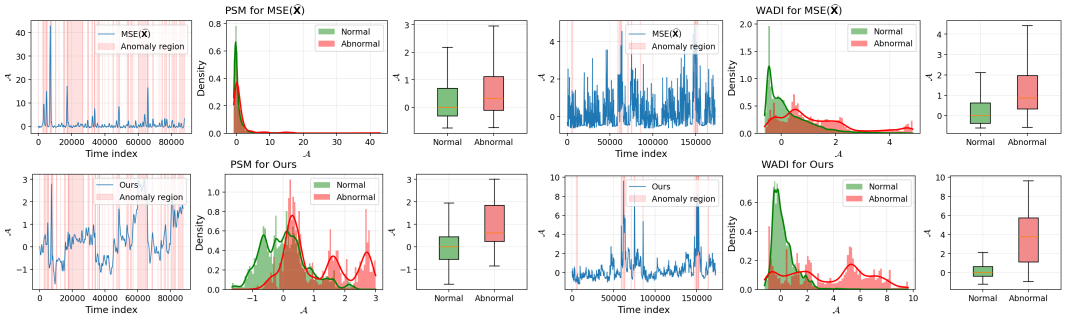


(a) Flow-Only (b) wo-LMA (c) Ours

Figure 5: Comparison of Flow-Only, wo-LMA, and our proposed model on the WADI dataset.

Anomaly scoring ablation study We present results for different anomaly scoring criteria in Fig. 6 and Table 2. Implementation details of the anomaly scoring variants, as well as visualizations of the anomaly scores, are provided in the appendix F.2. The **MSE**($\hat{\mathbf{X}}$) relies on temporal reconstruction errors, resulting in weaker performance compared to anomaly scores that incorporate latent representations. **Without averaging (wo-Avg)** removes rolling-window smoothing, leading to noisier

anomaly scores and degraded detection performance. **NF likelihood (NF-LL)**, **Latent likelihood (Lat-LL)**, and **Ours** are closely related. NF-LL computes exact likelihood via change-of-variables, while Lat-LL evaluates latent negative log-likelihood by omitting the Jacobian term. Our method instead uses a squared L2-norm in the latent space. Lat-LL and NF-LL share the same invertible mapping and therefore exhibit similar behavior, with results indicating that incorporating latent variance remains competitive with a squared L2-norm objective. Nonetheless, latent-space anomaly scoring (Ours, Lat-LL, NF-LL) consistently outperforms reconstruction error $\mathrm{MSE}(\hat{\mathbf{X}})$. Figure 6 further illustrates this on the PSM and WADI datasets, showing a clearer separation between normal and anomalous samples. The anomaly score distributions of NSPR and CrossAD are presented in Appendix E.3 for comparison.



(a) PSM Results (b) WADI Results

Figure 6: Anomaly score distributions of reconstruction error $\mathrm{MSE}(\hat{\mathbf{X}})$ and our proposed method.

Inference with a Precomputed Mean Latent Representation To determine whether gains stem from inference-time diffusion sampling or learned latent representations, we replaced the diffusion-generated latent samples $\hat{\boldsymbol{\mu}}_{\mathcal{D}}$ with a precomputed mean latent representation estimated from the trained diffusion model's outputs using the training and validation data. As shown in Table 3, performance remains nearly identical, with minimal variation across five random seeds. This suggests that the gains primarily stem from the jointly learned latent representations rather than diffusion sampling during inference. Thus, diffusion sampling can be omitted when reconstruction distributions are not required, reducing computational cost without compromising anomaly detection performance.

Table 3: Comparison of precomputed and diffusion-generated latent representations.

Dataset	Method	ROC	F1	V-PR
MSL	Precomputed	0.810 ± 0.011	0.413 ± 0.027	0.304 ± 0.014
	Diffusion-generated	0.811 ± 0.013	0.413 ± 0.030	0.304 ± 0.015
PSM	Precomputed	0.837 ± 0.009	0.657 ± 0.021	0.567 ± 0.012
	Diffusion-generated	0.837 ± 0.011	0.665 ± 0.016	0.557 ± 0.011
WADI	Precomputed	0.812 ± 0.007	0.613 ± 0.008	0.510 ± 0.013
	Diffusion-generated	0.809 ± 0.006	0.626 ± 0.009	0.524 ± 0.019

5 Conclusion

This paper introduces Flow+Diff, a time series anomaly detection framework that identifies anomalies through discrepancies between complementary latent representations learned by normalizing flows and diffusion models, rather than relying solely on temporal reconstruction errors. By coupling a deterministic normalizing flow with a stochastic diffusion-based model, Flow+Diff learns complementary latent representations, whose discrepancies become more pronounced under anomalous inputs. The bijective nature of normalizing flows enables time-domain reconstruction, facilitating the identification of features contributing to detected anomalies. Extensive experiments demonstrate that Flow+Diff consistently outperforms strong baselines, achieving state-of-the-art performance on four datasets. These results demonstrate the effectiveness of shared latent representations for accurate and interpretable anomaly detection in time series data.

Acknowledgments and Disclosure of Funding

This work was supported by the Rapid-Rich Object Search (ROSE) Lab, School of Electrical and Electronic Engineering, Nanyang Technological University.

References

[1] Simona-Vasilica Oprea, Adela Bâra, Florina Camelia Puican, and Ioan Cosmin Radu. Anomaly detection with machine learning algorithms and big data in electricity consumption. *Sustainability*, 13(19), 2021. ISSN 2071-1050. doi: 10.3390/su131910963. URL https://www.mdpi.com/2071-1050/13/19/10963.

[2] Van Kwan Zhi Koh, Ye Li, Xing Yong Kek, Ehsan Shafiee, Zhiping Lin, and Bihan Wen. A review of recent hybridized machine learning methodologies for time series forecasting on water-related variables. *Journal of Hydrology*, 656:132909, 2025.

[3] Xinggan Peng, Yuxuan Lin, Qi Cao, Yigang Cen, Huiping Zhuang, and Zhiping Lin. Traffic anomaly detection in intelligent transport applications with time series data using informer. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3309–3314, 2022. doi: 10.1109/ITSC55140.2022.9922142.

[4] Moussab Orabi, Kim Phuc Tran, Philipp Egger, and Sébastien Thomassey. Anomaly detection in smart manufacturing: An adaptive adversarial transformer-based model. *Journal of Manufacturing Systems*, 77:591–611, 2024. ISSN 0278-6125. doi: https://doi.org/10.1016/j.jmsy.2024.09.021. URL https://www.sciencedirect.com/science/article/pii/S0278612524002255.

[5] Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5):4027–4035, May 2021. doi: 10.1609/aaai.v35i5.16523. URL https://ojs.aaai.org/index.php/AAAI/article/view/16523.

[6] Rongyao Hu, Xinyu Yuan, Yan Qiao, BenChu Zhang, and Pei Zhao. Unsupervised anomaly detection for multivariate time series using diffusion model. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 9606–9610. IEEE, 2024.

[7] Jie Cao, Jiawei Miao, Haicheng Tao, Youquan Wang, Jia Wu, Zidong Wang, and Xindong Wu. Generative models for time series anomaly detection: A survey. *IEEE Transactions on Artificial Intelligence*, pages 1–21, 2025. doi: 10.1109/TAI.2025.3614213.

[8] Zahra Zamanzadeh Darban, Geoffrey I. Webb, Shirui Pan, Charu Aggarwal, and Mahsa Salehi. Deep learning for time series anomaly detection: A survey. *ACM Comput. Surv.*, 57(1), October 2024. ISSN 0360-0300. doi: 10.1145/3691338. URL https://doi.org/10.1145/3691338.

[9] Junho Song, Keonwoo Kim, Jeonglyul Oh, and Sungzoon Cho. Memto: Memory-guided transformer for multivariate time series anomaly detection. *Advances in Neural Information Processing Systems*, 36, 2024.

[10] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *International Conference on Learning Representations*, 2022.

[11] Chih-Yu (Andrew) Lai, Fan-Keng Sun, Zhengqi Gao, Jeffrey H Lang, and Duane Boning. Nominality score conditioned time series anomaly detection by point/sequential reconstruction. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 76637–76655. Curran Associates, Inc., 2023.

[12] Beibu Li, Qichao Shentu, Yang Shu, Hui Zhang, Ming Li, Ning Jin, Bin Yang, and Chenjuan Guo. CrossAD: Time series anomaly detection with cross-scale associations and cross-window modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=fkYFDxiLpi.

[13] Jinyu Cai, Yuan Xie, Glynnis Lim, Yifang Yin, Roger Zimmermann, and See-Kiong Ng. Self-perturbed anomaly-aware graph dynamics for multivariate time-series anomaly detection. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=hJJnwcvE2M.

[14] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[15] David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54:45–66, 2004.

[16] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[17] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. Tranad: deep transformer networks for anomaly detection in multivariate time series data. *Proc. VLDB Endow.*, 15(6):1201–1214, February 2022. ISSN 2150-8097. doi: 10.14778/3514061.3514067. URL https://doi.org/10.14778/3514061.3514067.

[18] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJJLHbb0-.

[19] Lifeng Shen, Zhuocong Li, and James Kwok. Timeseries anomaly detection using temporal hierarchical one-class network. *Advances in neural information processing systems*, 33:13016–13026, 2020.

[20] Yiyuan Yang, Chaoli Zhang, Tian Zhou, Qingsong Wen, and Liang Sun. Dcdetector: Dual attention contrastive representation learning for time series anomaly detection. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, page 3033–3045, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701030. doi: 10.1145/3580305.3599295. URL https://doi.org/10.1145/3580305.3599295.

[21] Hongzuo Xu, Yijie Wang, Songlei Jian, Qing Liao, Yongjun Wang, and Guansong Pang. Calibrated one-class classification for unsupervised time series anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):5723–5736, 2024. doi: 10.1109/TKDE.2024.3393996.

[22] Enyan Dai and Jie Chen. Graph-augmented normalizing flows for anomaly detection of multiple time series. In *International Conference on Learning Representations*, 2022.

[23] Qihang Zhou, Jiming Chen, Haoyu Liu, Shibo He, and Wenchao Meng. Detecting multivariate time series anomalies with zero known label. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4963–4971, 2023.

[24] Yongping He, Tijin Yan, Yufeng Zhan, Zihang Feng, and Yuanqing Xia. Sgfm: Conditional flow matching for time series anomaly detection with state space models. *IEEE Internet of Things Journal*, 11(22): 36979–36990, 2024. doi: 10.1109/JIOT.2024.3439672.

[25] Siwei Guan, Zhiwei He, Shenhui Ma, and Mingyu Gao. Conditional normalizing flow for multivariate time series anomaly detection. *ISA Transactions*, 143:231–243, 2023. ISSN 0019-0578. doi: https://doi.org/10.1016/j.isatra.2023.09.002. URL https://www.sciencedirect.com/science/article/pii/S0019057823004020.

[26] Chunjing Xiao, Zehua Gou, Wenxin Tai, Kunpeng Zhang, and Fan Zhou. Imputation-based time-series anomaly detection with conditional weight-incremental diffusion models. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, page 2742–2751, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701030. doi: 10.1145/3580305.3599391. URL https://doi.org/10.1145/3580305.3599391.

[27] Victor Livernoche, Vineet Jain, Yashar Hezaveh, and Siamak Ravanbakhsh. On diffusion modeling for anomaly detection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=lR3rk7ysXz.

[28] Yuhang Chen, Chaoyun Zhang, Minghua Ma, Yudong Liu, Ruomeng Ding, Bowen Li, Shilin He, Saravan Rajmohan, Qingwei Lin, and Dongmei Zhang. Imdiffusion: Imputed diffusion models for multivariate time series anomaly detection. *Proceedings of the VLDB Endowment*, 17(3):359–372, 2023.

[29] Hongjun Sheng, Xinggan Peng, Van Kwan Zhi Koh, Bihan Wen, and Zhiping Lin. Twavefussion: Wavelet-based diffusion with transformer for multivariate time series anomaly detection. In *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, 2025. doi: 10.1109/ISCAS56072.2025.11043617.

[30] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

[31] 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.

[32] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

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

[34] 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*, pages 2828–2837, 2019.

[35] 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*, pages 387–395, 2018.

[36] 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)*, pages 31–36, 2016. doi: 10.1109/CySWater.2016.7469060.

[37] 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*, pages 2485–2494, 2021.

[38] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P. Mathur. Wadi: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*, CySWATER '17, page 25–28, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349758. doi: 10.1145/3055366.3055375. URL https://doi-org.remotexs.ntu.edu.sg/10.1145/3055366.3055375.

[39] Chengsen Wang, Zirui Zhuang, Qi Qi, Jingyu Wang, Xingyu Wang, Haifeng Sun, and Jianxin Liao. Drift doesn't matter: Dynamic decomposition with diffusion reconstruction for unstable multivariate time series anomaly detection. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=aW5bSuduF1.

[40] Qinghua Liu and John Paparrizos. The elephant in the room: Towards a reliable time-series anomaly detection benchmark. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 108231–108261. Curran Associates, Inc., 2024. doi: 10.52202/079017-3437. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/c3f3c690b7a99fba16d0efd35cb83b2c-Paper-Datasets_and_Benchmarks_Track.pdf.

[41] John Paparrizos, Paul Boniol, Themis Palpanas, Ruey S Tsay, Aaron J Elmore, and Michael J Franklin. Volume under the surface: A new accuracy evaluation measure for time-series anomaly detection. *Proc. VLDB Endow.*, 15(11):2774–2787, 2022.

A Appendix Overview

This appendix is organized as follows:

- Appendix B: Preliminaries, including normalizing flows and diffusion models.
- Appendix C: Methodological details, including derivations, the affine coupling layer in the normalizing flow, and the reconstruction procedure.
- Appendix D: Experimental setup, including implementation details, hyperparameter settings, and dataset descriptions.
- Appendix E: Additional experiments, including extended evaluation metrics and qualitative visualizations.
- Appendix F: Additional ablation experiments, model variants, alternative anomaly scoring strategies, and hyperparameter analyzes.
- Appendix G: Broader impacts.
- Appendix H: Limitations.

B Preliminaries

This section presents the preliminaries underlying our model, including normalizing flows and diffusion models.

B.1 Normalizing Flow

NF models enable tractable density estimation by constructing a sequence of invertible transformations with computable likelihoods [30]. Based on the change-of-variables formula, the probability density of the observed data $p_{\mathcal{X}}(\mathbf{X})$ is given by

$$p_{\mathcal{X}}(\mathbf{X}) = p_{\mathcal{Z}}\big(\mathcal{F}(\mathbf{X})\big)\,\left|\det J_{\mathcal{F}}(\mathbf{X})\right|, \tag{12}$$

where $p_{\mathcal{Z}}(\mathbf{Z})$ denotes the base distribution, which is typically assumed to be a multivariate standard Gaussian distribution, and $|\det J_{\mathcal{F}}(\mathbf{X})|$ denotes the absolute value of the Jacobian determinant. Specifically, $\mathcal{F}$ is defined as a composition of M_f invertible transformations, $\mathcal{F} = f_{M_f} \circ f_{M_f-1} \circ \cdots \circ f_1$, where $\circ$ denotes function composition. Each transformation f_{l_f}, for $l_f = 1, \ldots, M_f$, is parameterized by learnable parameters and has a tractable Jacobian determinant. Therefore, the model can be trained by maximizing the likelihood in (12). However, for general transformations, computing the full Jacobian determinant is often computationally expensive or intractable. To overcome this limitation, expressive transformations with tractable Jacobian determinants, such as REALNVP [30], have been proposed, which we incorporate into our model.

B.2 Diffusion Model

The diffusion model consists of a forward and a reverse process. To accelerate inference, we adopt Denoising Diffusion Implicit Models (DDIM) [32], which enables efficient sampling with fewer denoising steps than standard DDPM [33]. The forward process gradually perturbs the input data $\mathbf{Z}_{\mathcal{D}}^{(0)}$ into a noise distribution through a sequence of latent variables $\{\mathbf{Z}_{\mathcal{D}}^{(t)}\}_{t=0}^{T}$, which can be simplified by reparameterization in closed form [33]:

$$q(\mathbf{Z}_{\mathcal{D}}^{(t)} \mid \mathbf{Z}_{\mathcal{D}}^{(0)}) = \mathcal{N}\Big(\mathbf{Z}_{\mathcal{D}}^{(t)}; \sqrt{\bar{\alpha}_t}\,\mathbf{Z}_{\mathcal{D}}^{(0)},\, (1-\bar{\alpha}_t)\mathbf{I}\Big), \tag{13}$$

where $\bar{\alpha}_t = \prod_{s=0}^{t}(1-\beta_s)$ and $\{\beta_s\}_{s=0}^{T}$ are the predefined minimum and maximum noise schedules, denoted by $\beta_{\min}$ and $\beta_{\max}$, respectively. Therefore, each forward sample at any t can be computed as follows:

$$\mathbf{Z}_{\mathcal{D}}^{(t)} = \sqrt{\bar{\alpha}_t}\,\mathbf{Z}_{\mathcal{D}}^{(0)} + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{14}$$

For conditioning, the reverse process incorporates an additional conditional input $\mathbf{Z}_c^{(t)}$ into the denoising network, i.e., $\boldsymbol{\epsilon}_\theta(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}, t)$. In the reverse process, DDIM evolves the latent variable from diffusion step t to $t-1$ via a non-Markovian implicit update parameterized by the denoising network $\boldsymbol{\epsilon}_\theta$:

$$p_\theta\Big(\mathbf{Z}_{\mathcal{D}}^{(t-1)} \mid \mathbf{Z}_{\mathcal{D}}^{(t)}\Big) = \mathcal{N}\Big(\mathbf{Z}_{\mathcal{D}}^{(t-1)};\, \mu_\theta(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}, t),\, \sigma_t^2\mathbf{I}\Big), \tag{15}$$

where the mean is given by

$$\mu_\theta(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}, t) = \sqrt{\bar{\alpha}_{t-1}}\,\hat{\mathbf{Z}}_{\mathcal{D}}^{(0)} + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\,\boldsymbol{\epsilon}_\theta(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}, t). \tag{16}$$

The optional stochasticity coefficient σ_t is defined as:

$$\sigma_t = \eta\sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}}\sqrt{1-\frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}}, \tag{17}$$

where $\eta = 0$ yields a deterministic sampler, while $\eta > 0$ introduces stochasticity. The estimated clean latent representation is obtained as:

$$\hat{\mathbf{Z}}_{\mathcal{D}}^{(0)} = \frac{\mathbf{Z}_{\mathcal{D}}^{(t)} - \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_\theta(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}, t)}{\sqrt{\bar{\alpha}_t}}. \tag{18}$$

By iteratively applying the update from $t = T$ down to $t = 0$ with a subsampling stride N, the reverse process produces the final latent representation $\mathbf{Z}_{\mathcal{D}}^{(0)}$, offering a more efficient alternative to DDPM. The sampling update is given by:

$$\mathbf{Z}_{\mathcal{D}}^{(t-1)} = \sqrt{\bar{\alpha}_{t-1}}\,\hat{\mathbf{Z}}_{\mathcal{D}}^{(0)} + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\,\boldsymbol{\epsilon}_\theta(\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}, t) + \sigma_t\mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{19}$$

C Methodology Details

This section provides a detailed discussion of the proposed methodology, derivation of the log-determinant of the final layer of the normalizing flow, the reconstruction procedure, an illustration of the normalizing flow transformation, and the algorithms.

C.1 Proof of the Log-Determinant for Masked Affine + Asig Layers

Consider a normalizing flow layer that applies: 1) Masked affine transformation (REALNVP-style) [30]; 2) Element-wise Asig activation, defined as $\mathrm{Asig}(x) = 2\sigma(x) - 1$. For simplicity, we exclude the temporal dimension.

Let $\mathbf{x} \in \mathbb{R}^d$ be the input and $\mathbf{M} \in \{0,1\}^d$ be a binary mask. The affine coupling is defined as:

$$\mathbf{y}_{\text{masked}} = \mathbf{x}_{\text{masked}}, \quad \mathbf{y}_{\text{unmasked}} = \mathbf{x}_{\text{unmasked}} \odot \exp(\mathbf{s}) + \mathbf{t},$$

where $\mathbf{s}, \mathbf{t}$ are scale and translation functions. The Jacobian is triangular:

$$J_{\text{affine}} = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} I_{\text{masked}} & 0 \\ * & \mathrm{diag}(\exp(\mathbf{s}_{\text{unmasked}})) \end{bmatrix}.$$

Hence, the log-determinant is:

$$\log|\det J_{\text{affine}}| = \sum_i (1 - M_i)\, s_i.$$

After the affine coupling, apply element-wise:

$$z_i = \mathrm{Asig}(y_i) = 2\sigma(y_i) - 1$$

with derivative:

$$\frac{dz_i}{dy_i} = 2\sigma(y_i)(1 - \sigma(y_i)).$$

Expressing in terms of z_i: $\sigma(y_i) = \frac{z_i+1}{2}$, we get

$$\frac{dz_i}{dy_i} = 2 \cdot \frac{z_i + 1}{2}\left(1 - \frac{z_i + 1}{2}\right) = \frac{1 - z_i^2}{2}.$$

Thus, the log-determinant contribution of Asig is:

$$\log\left|\det J_{\text{Asig}}\right| = \sum_i \log \frac{dz_i}{dy_i} = \sum_i \left[\log(1 - z_i^2) - \log 2\right].$$

Combining the masked affine layer and the Asig activation:

$$\begin{aligned}\log|\det J_{\mathcal{F}}| &= \log|\det J_{\text{affine}}| + \log|\det J_{\text{Asig}}| \\ &= \sum_i (1 - M_i)\, s_i + \sum_i \left[\log(1 - z_i^2) - \log 2\right].\end{aligned}$$

Based on the derivation above, the second term in (9) is formulated as follows:

$$\log\left|\det J_{\mathcal{F}}(\mathbf{X}'')\right| = \underbrace{\sum_{l_f=1}^{M_f}\sum_{i=1}^{L}\sum_{j=1}^{2d_x}\left(1 - m_{i,j}^{(l_f)}\right)s_{i,j}^{(l_f)}}_{\text{Affine coupling layer}} + \underbrace{\sum_{i=1}^{L}\sum_{j=1}^{2d_x}\left[\log\left(1 - (z_{\mathcal{F},i,j}^{(M_f)})^2\right) - \log(2)\right]}_{\text{Asig}}$$

where $\mathbf{M}^{(l_f)}$ is a binary mask for the l_f-th affine coupling layer, and $m_{i,j}^{(l_f)} \in \{0,1\}$ denotes its (i,j)-th entry. The term $\mathbf{S}^{(l_f)}$ denotes the scale parameter produced by the affine coupling network, and $s_{i,j}^{(l_f)}$ denotes its (i,j)-th entry.

C.2 Reconstruction

Reconstructing the input $\hat{\mathbf{X}}$ enables interpretation of affected features by analyzing deviations in the reconstructed time series. This is enabled by the bijective nature of the NF, which allows latent representations sampled from the diffusion model to be inversely mapped back to the input space. Given a diffusion-derived latent sample $\mathbf{Z}_{\mathcal{D},k}$, the corresponding input is recovered via the inverse affine-sigmoid followed by the inverse flow:

$$\hat{\mathbf{X}}_k'' = f_1^{-1} \circ \cdots \circ f_{M_f}^{-1} \circ \mathrm{Asig}^{-1}(\mathbf{Z}_{\mathcal{D},k}),$$

where $\mathrm{Asig}^{-1}(y) = \log\frac{1+y}{1-y}$. Since the input is mean-centered, the original input is reconstructed as

$$\hat{\mathbf{X}}_k = \left[\hat{\mathbf{x}}_{k,1}'', \ldots, \hat{\mathbf{x}}_{k,d_x}''\right] + \mathbf{1}_L \boldsymbol{\mu}_{in},$$

where $\mathbf{1}_L \in \mathbb{R}^{L\times 1}$ denotes a vector of ones. This reconstruction is repeated for all K samples, from which the mean $\hat{\boldsymbol{\mu}}_X$ and standard deviation $\hat{\boldsymbol{\sigma}}_X$ of the reconstructed inputs are computed.

C.3 An illustration of normalizing flow transformation

For clarity, we provide an illustration of the normalizing flow transformation in Fig. 7.

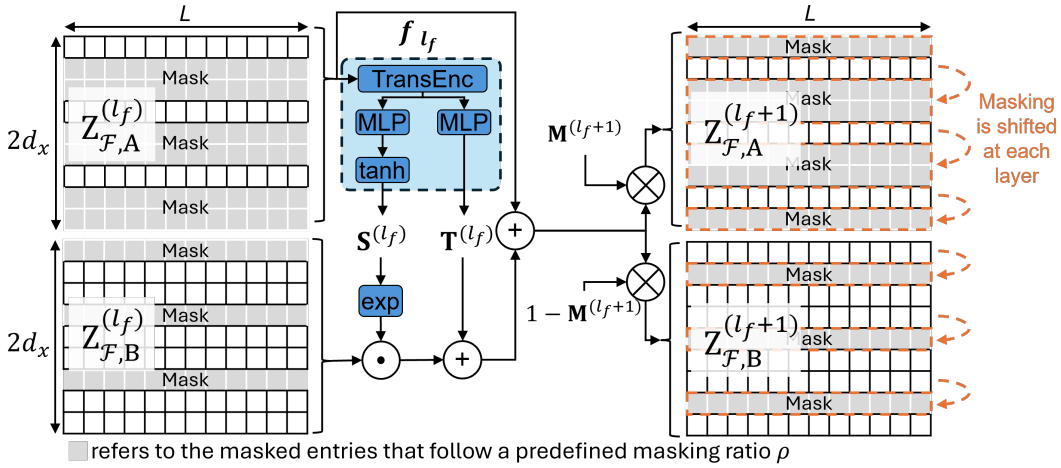


Figure 7: An illustration of the NF transformation.

C.4 Algorithms

To improve clarity and reproducibility, we provide the training procedure in Algorithm 1, and the inference and reconstruction procedures for anomaly scoring in Algorithm 2.

Algorithm 1 Training Procedure

Require: Input sequence $\mathbf{X} \in \mathbb{R}^{L \times d_x}$, diffusion step T
Initialize learnable parameter $\log \sigma_{\mathcal{D}}^2$
repeat
  $(\mathbf{X}^{'}, \mathbf{X}^{''}, \boldsymbol{\mu}_{in}) = \mathrm{LMA}(\mathbf{X})$ (Eqs. 1-2)
  Normalizing Flow:
  $\Big(\mathbf{Z}_{\mathcal{F}}, \log \Big| \det J_{\mathcal{F}}(\mathbf{X}^{''})\Big|\Big) = \mathcal{F}(\mathbf{X}^{''})$
  Diffusion Process:
  Sample $t \sim \mathcal{U}(0, T), \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$
  $\mathbf{Z}_{\mathcal{D}}^{(t)} = \sqrt{\bar{\alpha}_t}\mathbf{Z}_{\mathcal{F}} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$
  Noise Prediction:
  $\mathbf{Z}_c^{(t)} = \mathrm{Asig}\Big(\mathrm{TransEncTime}(\mathbf{X}^{'}, t)\Big)$
  $\hat{\boldsymbol{\epsilon}} = \mathrm{UNet}\big([\mathbf{Z}_{\mathcal{D}}^{(t)}, \mathbf{Z}_c^{(t)}], t\big)$ (Eq. 7)
  Gaussian Parameter Estimation:
  $\tilde{\boldsymbol{\mu}}_{\mathcal{D}} = \frac{\mathbf{Z}_{\mathcal{D}}^{(t)} - \sqrt{1-\bar{\alpha}_t}\,\hat{\boldsymbol{\epsilon}}}{\sqrt{\bar{\alpha}_t}}$ (Eq. 8)
  $\hat{\boldsymbol{\sigma}}_{\mathcal{D}} = \mathrm{clip}\big(\exp(0.5 \cdot \log \sigma^2), 0.1, 5.0\big)$ (prevents variance explosion)
  Loss:
  $\mathcal{L} = \mathbb{E}_{\mathbf{X}'' \sim p_{\mathcal{X}''},\, t,\, \boldsymbol{\epsilon}}\Big[\log \mathcal{N}\big(\mathcal{F}(\mathbf{X}''); \tilde{\boldsymbol{\mu}}_{\mathcal{D}}, \tilde{\sigma}_{\mathcal{D}}^2\mathbf{I}\big) + \log \Big|\det J_{\mathcal{F}}(\mathbf{X}'')\Big|\Big]$ (Eq. 9)
  Update parameters using gradient descent
until $\mathcal{L}$ convergence

D Experimental Setup

This section presents the implementation details, hyperparameter settings, and dataset details.

D.1 Implementation Details

All experiments are conducted on a server equipped with a single RTX 3090 GPU using the PyTorch framework. The models are trained with the AdamW optimizer at a learning rate of 1×10^{-4} for 50 epochs with a batch size of 64. Early stopping is applied if the validation loss does not decrease for three consecutive evaluations. For the main experiments, we conduct five independent runs with random seeds of 1000-1004 and report the average performance.

Algorithm 2 Anomaly Scoring & Reconstruction Procedure

Require: Input sequence $\mathbf{X} \in \mathbb{R}^{L \times d_x}$, number of samples K, DDIM stride N, diffusion step T

Local Mean Augmentation:

$(\mathbf{X}^{'}, \mathbf{X}^{''}, \boldsymbol{\mu}_{in}) = \mathrm{LMA}(\mathbf{X})$ (Eqs. 1-2)

Normalizing Flow:

$\mathbf{Z}_{\mathcal{F}} = \mathcal{F}(\mathbf{X}^{''})$

Define DDIM stride:

for $k = 1, \ldots, K$ **do**

  Initialize $\mathbf{Z}_{\mathcal{D},k}^{(T)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

  for $t = T, T-N, T-2N, \ldots, 0$ **do**

    $\mathbf{Z}_c^{(t)} = \mathrm{Asig}(\mathrm{TransEncTime}(\mathbf{X}^{'}, t))$

    $\hat{\boldsymbol{\epsilon}} = \mathrm{UNet}\big([\mathbf{Z}_{\mathcal{D},k}^{(t)}, \mathbf{Z}_c^{(t)}], t\big)$

    $\mathbf{Z}_{\mathcal{D},k}^{(0)} = \frac{\mathbf{z}_{\mathcal{D},k}^{(t)} - \sqrt{1-\bar{\alpha}_t}\hat{\boldsymbol{\epsilon}}}{\sqrt{\bar{\alpha}_t}}$

    $\mathbf{Z}_{\mathcal{D},k}^{(t-N)} = \sqrt{\bar{\alpha}_{t\text{-N}}}\mathbf{Z}_{\mathcal{D},k}^{(0)} + \sqrt{1-\bar{\alpha}_{t\text{-N}} - \sigma_t^2}\hat{\boldsymbol{\epsilon}} + \sigma_t \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$

  end for

  $\mathbf{Z}_{\mathcal{D},k} = \mathrm{Asig}(\mathbf{Z}_{\mathcal{D},k}^{(0)})$

  $\hat{\mathbf{X}}_k^{''} = \mathcal{F}^{-1}(\mathbf{Z}_{\mathcal{D},k})$ (Eq. 10)

end for

FlowDiff-AS:

$\hat{\boldsymbol{\mu}}_{\mathcal{D}} = \frac{1}{K}\sum_{k=1}^{K} \mathbf{Z}_{\mathcal{D},k}$

$\text{DiscrepancyScore}_i = \sum_j \left(z_{\mathcal{F},i,j} - \hat{\mu}_{\mathcal{D},i,j}\right)^2$ (Eq. 11)

Reconstuction:

$\hat{\mathbf{X}}_k = [\hat{\mathbf{x}}_{k,1}^{''}, \ldots, \hat{\mathbf{x}}_{k,d_x}^{''}] + \mathbf{1}_L \boldsymbol{\mu}_{in}$ adding back the input mean

$\hat{\boldsymbol{\mu}}_X = \frac{1}{K}\sum_{k=1}^{K} \hat{\mathbf{X}}_k$

$\hat{\boldsymbol{\sigma}}_X = \sqrt{\frac{1}{k}\sum_{k=1}^{K}\left(\hat{\mathbf{X}}_k - \hat{\boldsymbol{\mu}}_X\right)^2}$

Return: $\hat{\boldsymbol{\mu}}_X \in \mathbb{R}^{L \times d_x}$, $\hat{\boldsymbol{\sigma}}_X \in \mathbb{R}^{L \times d_x}$, **DiscrepancyScore** $\in \mathbb{R}^L$

We re-implemented all baseline methods following the methodological descriptions in their original papers, with reference to the official or publicly available implementations listed below. Minor differences may arise from implementation details, random initialization, and training environments:

- IF [16], LOF [14], OC-SVM [15], GANF [22], and DAGMM [18]: https://github.com/OpsPAI/MTAD
- THOC [19]: https://github.com/carrtesy/THOC-Pytorch
- DCdetector [20]: https://github.com/DAMO-DI-ML/KDD2023-DCdetector
- AT [10]: https://github.com/thuml/Anomaly-Transformer
- MEMTO [9]: https://github.com/gunny97/MEMTO
- NSPR [11]: https://github.com/andrewlai61616/NPSR
- CrossAD [12]: https://github.com/decisionintelligence/CrossAD
- DiffAD [26]: https://github.com/ChunjingXiao/DiffAD

D.2 Hyperparameter Setting

We empirically identify a subset of hyperparameters that generalize well across datasets, thereby avoiding extensive dataset-specific tuning. The hidden dimension is set to $h = 256$, and the number of normalizing flow layers is $M_f = 4$. The affine coupling network, TransEnc$(\cdot)$, consists of $n_f = 4$ layers with $attheadf = 4$ attention heads. For the diffusion model, all datasets use $T = 100$ diffusion steps with a sampling stride of $N = 10$, and a linear noise schedule ranging from $\beta_{\min} = 1 \times 10^{-3}$ to $\beta_{\max} = 1 \times 10^{-1}$. The diffusion conditional network, TransEncTime$(\cdot)$, consists of $n_d = 4$ layers with $attheadd = 4$ attention heads. The U-Net denoising network uses channel multipliers $U_{\mathcal{D}} = (1, 2, 4, 8)$. During inference, $K = 30$ stochastic samples are generated.

The remaining hyperparameters are dataset-specific. We set $\rho = 0.5$ for all datasets except PSM and WADI, where $\rho = 0.25$ is used. The input window size is chosen based on the typical anomaly segment length (see dataset details in Appendix D.3), with shorter anomalies requiring smaller windows. Accordingly, we set $L = 64$ for SMD and MSL, $L = 128$ for SWaT and WADI, and $L = 256$ for SMAP and PSM. The segmentation stride is set to $st = L/4$ for training, while a non-overlapping window ($st = L$) is used during inference.

D.3 Datasets

In our experiments, we utilize six publicly available datasets for MTSAD. The details of these datasets are as follows:

1. Server Machine Dataset (SMD) [34] is collected from a large internet company and consists of multivariate server performance metrics, including CPU load, network usage, and memory utilization.

2. Soil Moisture Active Passive (SMAP) satellite and the Mars Science Laboratory (MSL) rover datasets consist of telemetry data collected by NASA, including radiation levels, temperature, power consumption, and computational metrics [35]. Following [12, 39], discrete values are found to be undesirable, as they introduce discontinuities that hinder effective modeling. We therefore set these features to zero.

3. Secure Water Treatment (SWaT) [36] is the network traffic data of a Water treatment plant, recorded from 51 sensors and actuators. The data recorded consists of 7 days of normal operation and 5 days of attack scenarios. Following [11], features with indices 5 and 10 are set to zero.

4. Pooled Server Metrics (PSM) [37] are data recorded from multiple application server nodes on eBay, in which the types of features are omitted due to anonymization requirements.

5. Water Distribution (WADI) [38] is a 16-day cyber-physical systems dataset collected from a water distribution network testbed, consisting of 14 days of normal operation and 2 days of controlled attacks. It contains continuous measurements from 127 sensors and actuators.

Datasets containing multiple entities are concatenated along the time dimension. For example, after preprocessing, the MSL dataset yields training data of size $58{,}317 \times 27$. Both training and testing data are rescaled using the minimum and maximum values computed from the training set. All datasets are normalized using min–max normalization to the range $[-1, 1]$, with values clipped to $[-4, 4]$ based on training data statistics. The training data are further split into training and validation sets using an 80% / 20% ratio.

Table 4: Details of the datasets used in this study.

Datasets	SMD	MSL	SMAP	SWaT	PSM	WADI
Number of features	38	55	25	51	25	123
Entities	28	27	55	1	1	1
Training steps	708,405	58,317	140,825	495,000	132,481	1,209,601
Testing steps	708,420	73,729	444,035	449,919	87,841	172,801
$\frac{\text{Abnormal steps}}{\text{Test steps}}$	4.16%	10.48%	12.83%	12.14%	27.76%	5.71%
Number of anomalous segments	327	36	69	35	72	14
Average anomaly length	90	216	826	1561	339	710

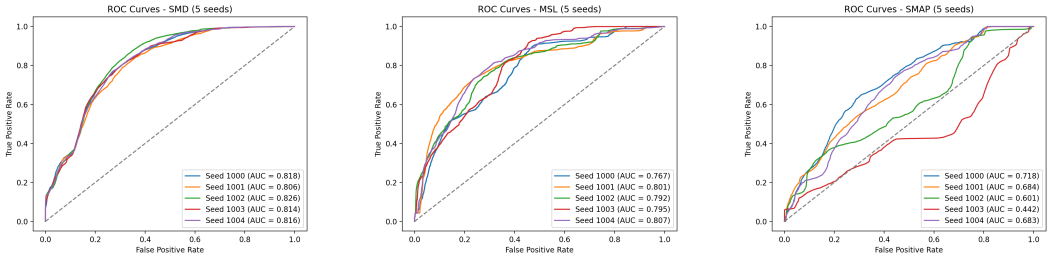
E Supplementary Experiments

E.1 Additional Metrics of the Main Results

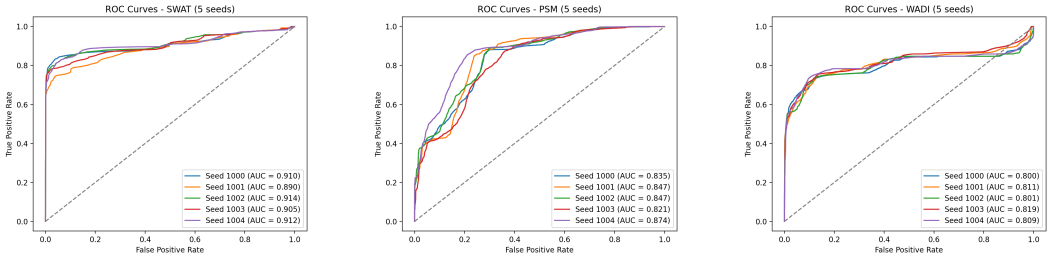
To comprehensively evaluate anomaly detection performance, we report six complementary metrics for each dataset: V-ROC, AUC-PR, Point-Adjustment F1-Score (PA-F1), Affiliation-F1 (Aff-F1), R-ROC, and R-PR [40]. Detailed definitions of these metrics are provided in [40]. The quantitative results are presented in Tables 6–7, while the corresponding ROC and PR curves are shown in Figures 8 and 9, respectively.

Table 5: Results of our proposed model and baseline methods.

	SMD			MSL			SMAP			SWaT			PSM			WADI		
Methods	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR
IF	0.664±0.0050	0.182±0.0049	0.126±0.0045	0.604±0.0101	0.253±0.0083	0.159±0.0042	0.634±0.0131	0.305±0.0101	0.166±0.0050	0.843±0.0021	0.741±0.0045	0.511±0.0080	0.684±0.0275	0.493±0.0204	0.467±0.0267	0.734±0.0063	0.313±0.0183	0.183±0.0087
OC-SVM	0.590±0.0000	0.171±0.0000	0.100±0.0000	0.526±0.0000	0.243±0.0000	0.164±0.0000	0.377±0.0000	0.227±0.0000	0.102±0.0000	0.884±0.0000	0.788±0.0000	**0.677±0.0000**	0.565±0.0000	0.460±0.0000	0.356±0.0000	0.482±0.0000	0.120±0.0000	0.059±0.0000
LOF	0.645±0.0000	0.124±0.0000	0.089±0.0000	0.556±0.0000	0.217±0.0000	0.143±0.0000	0.623±0.0000	0.284±0.0000	0.174±0.0000	0.888±0.0000	0.780±0.0000	0.610±0.0000	0.728±0.0000	0.583±0.0000	0.432±0.0000	0.513±0.0000	0.261±0.0000	0.113±0.0000
THOC	0.651±0.0049	0.213±0.0054	0.136±0.0041	0.579±0.0052	0.230±0.0035	0.163±0.0011	0.421±0.0050	0.230±0.0006	0.112±0.0009	0.834±0.0049	0.760±0.0017	0.668±0.0114	0.684±0.0072	0.503±0.0043	0.457±0.0161	0.488±0.0069	0.121±0.0005	0.058±0.0029
DCdetector	0.498±0.0014	0.080±0.0000	0.062±0.0001	0.479±0.0212	0.227±0.0000	0.132±0.0034	0.495±0.0000	0.227±0.0000	0.135±0.0000	0.494±0.0010	0.217±0.0000	0.124±0.0003	0.501±0.0027	0.435±0.0000	0.295±0.0006	0.497±0.0053	0.109±0.0001	0.061±0.0005
DAGMM	0.714±0.0094	0.215±0.0215	0.151±0.0205	0.583±0.0185	0.235±0.0113	0.160±0.0047	0.474±0.0590	0.243±0.0194	0.128±0.0161	0.861±0.0205	0.744±0.0393	0.546±0.0846	0.651±0.0260	0.473±0.0105	0.456±0.0420	0.505±0.0555	0.196±0.0601	0.089±0.0345
AT	0.407±0.0249	0.080±0.0000	0.054±0.0021	0.472±0.0083	0.191±0.0000	0.119±0.0026	0.585±0.0179	0.257±0.0096	0.164±0.0077	0.493±0.0016	0.216±0.0000	0.124±0.0004	0.470±0.0048	0.434±0.0000	0.293±0.0035	0.497±0.0101	0.109±0.0000	0.061±0.0007
MEMTO	0.502±0.0012	0.080±0.0000	0.063±0.0001	0.508±0.0417	0.196±0.0046	0.140±0.0101	0.505±0.0027	0.227±0.0000	0.135±0.0005	0.656±0.0978	0.401±0.1703	0.245±0.1092	0.503±0.0012	0.434±0.0000	0.307±0.0003	0.430±0.0053	0.109±0.0000	0.054±0.0007
GANF	0.565±0.0394	0.184±0.0175	0.127±0.0145	0.601±0.0067	0.307±0.0158	0.217±0.0052	0.462±0.0826	0.261±0.0143	0.148±0.0238	0.842±0.0269	0.759±0.0270	0.527±0.0886	0.603±0.0188	0.534±0.0116	0.415±0.0099	0.507±0.0218	0.262±0.0053	0.105±0.0020
DiffAD	0.586±0.0000	0.126±0.0000	0.330±0.0000	0.520±0.0000	0.213±0.0000	0.169±0.0000	0.384±0.0000	0.227±0.0000	0.220±0.0000	0.329±0.0000	0.217±0.0000	0.143±0.0000	0.557±0.0000	0.448±0.0000	0.367±0.0000	0.358±0.0000	0.109±0.0000	0.087±0.0000
NSPR	**0.834±0.0063**	**0.375±0.0080**	**0.369±0.0073**	0.750±0.0201	0.409±0.0095	**0.327±0.0096**	0.590±0.0252	0.278±0.0140	0.187±0.0198	**0.913±0.0030**	0.826±0.0068	0.663±0.0308	0.803±0.0108	0.634±0.0162	0.499±0.0083	0.507±0.0534	0.150±0.0209	0.065±0.0083
CrossAD	0.758±0.0016	0.240±0.0004	0.218±0.0016	0.782±0.0011	0.338±0.0002	0.312±0.0002	0.560±0.0016	0.277±0.0019	0.153±0.0001	0.847±0.0017	0.717±0.0054	0.570±0.0119	0.646±0.0145	0.465±0.0151	0.550±0.0248	0.444±0.0006	0.111±0.0003	0.056±0.0000
Ours	0.745±0.0062	0.224±0.0125	0.105±0.0035	**0.820±0.0079**	**0.446±0.0244**	0.317±0.0094	**0.700±0.0113**	**0.348±0.0039**	**0.232±0.0026**	0.907±0.0050	**0.836±0.0037**	0.617±0.0273	**0.838±0.0096**	**0.659±0.0213**	**0.567±0.0120**	**0.812±0.0101**	**0.619±0.0107**	**0.515±0.0179**

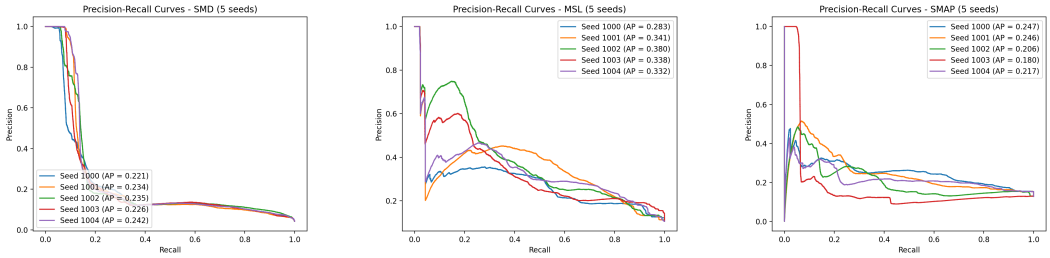


(a) SMD (b) MSL (c) SMAP

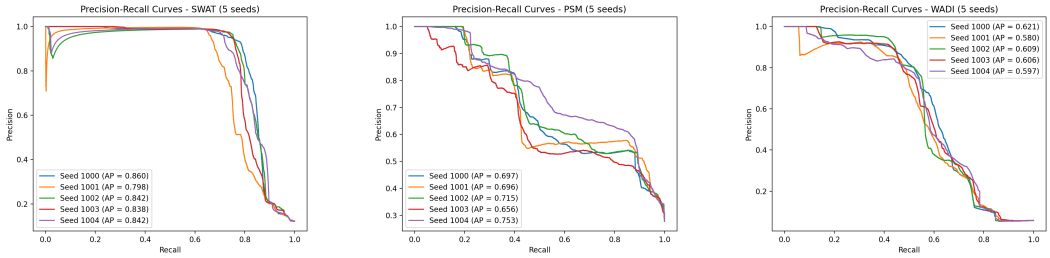


(d) SWaT (e) PSM (f) WADI

Figure 8: ROC curves across five random seeds.



(a) SMD (b) MSL (c) SMAP



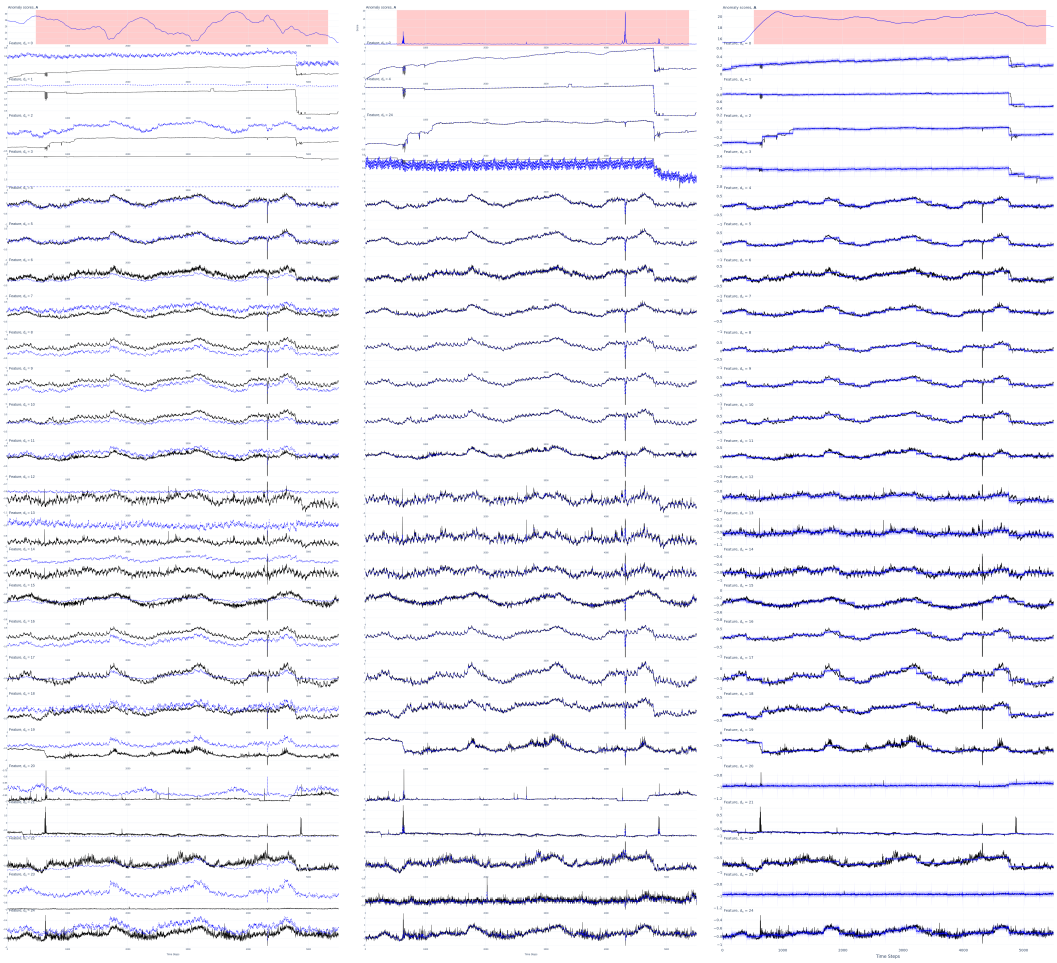
(d) SWaT (e) PSM (f) WADI

Figure 9: Precision and recall curves across five random seeds.

E.2 Inference Cost and Computational Complexity Analysis

Our method generates K DDIM samples for each input window. Since these samples are generated independently, they can be produced in parallel, resulting in a practical inference overhead that is substantially lower than the naive K × increase in computation time. To quantify the computational cost, we benchmarked the training time per epoch and inference latency against CrossAD and NSPR for SWAT dataset using a batch size of 32.

We acknowledge the computational overhead of our design and will explore further model reduction in future work. Nevertheless, our model achieves improved performance while providing feature-wise interpretability. The diffusion model dominates the complexity, accounting for approximately 230M of the 250M parameters,



(a) NSPR (PSM) (b) CrossAD (PSM) (c) Ours (PSM)

Figure 10: Feature reconstruction comparison on PSM.

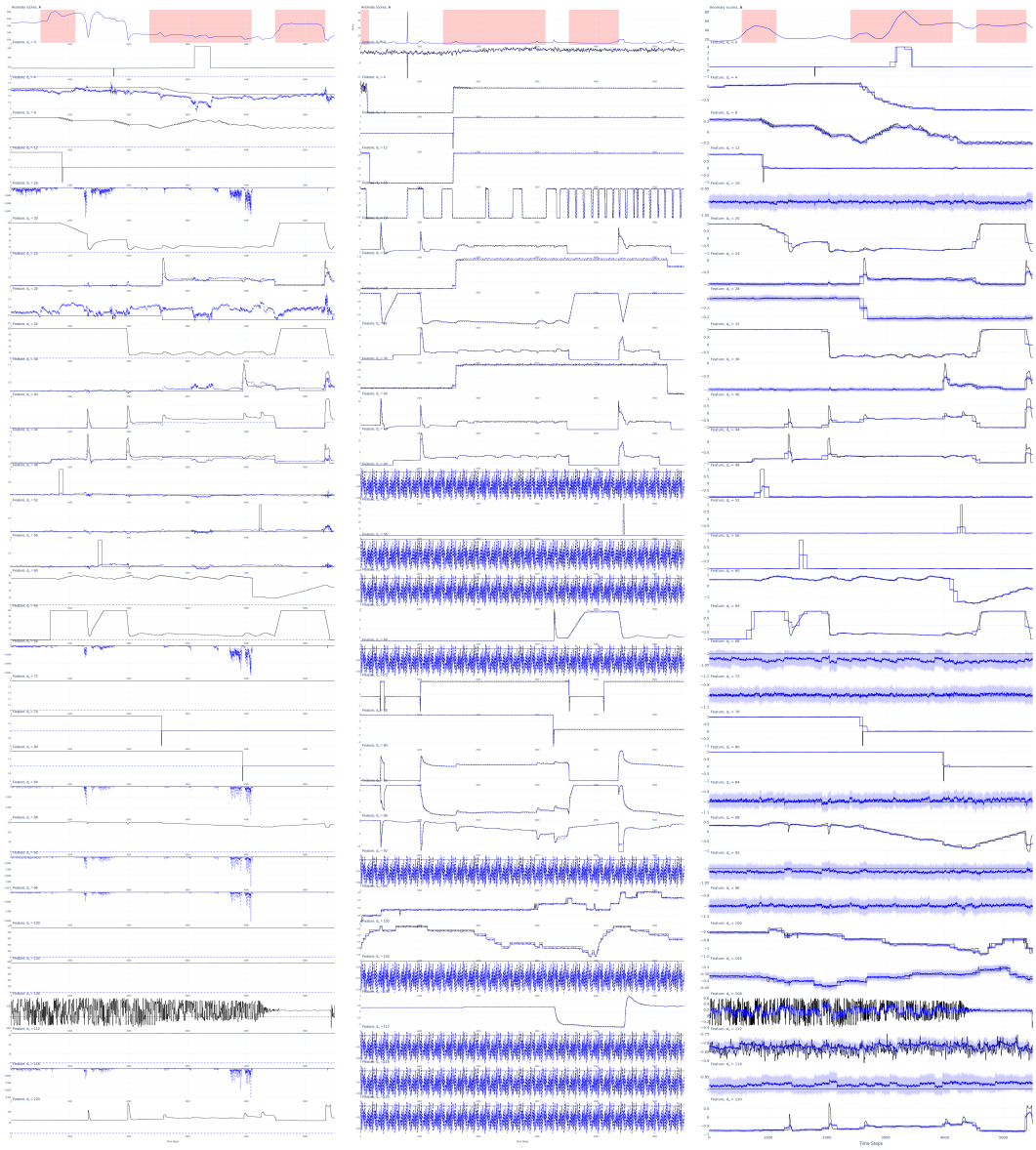
Table 6: Results on SMD, MSL, and SMAP using V-ROC, AUC-PR, Aff-F1, PA-F1, R-ROC, and R-PR.

	SMD						MSL						SMAP					
Methods	V-ROC	AUC-PR	Aff-F1	PA-F1	R-ROC	R-PR	V-ROC	AUC-PR	Aff-F1	PA-F1	R-ROC	R-PR	V-ROC	AUC-PR	Aff-F1	PA-F1	R-ROC	R-PR
IF	0.737±0.0037	0.152±0.0098	0.743±0.0070	0.611±0.0219	0.796±0.0039	0.153±0.0047	0.655±0.0089	0.139±0.0056	0.700±0.0083	0.856±0.0229	0.701±0.0083	0.182±0.0049	0.610±0.0095	0.159±0.0054	0.690±0.0024	0.702±0.0004	0.622±0.0095	0.173±0.0051
OC-SVM	0.650±0.0000	0.110±0.0000	0.758±0.0000	0.675±0.0000	0.727±0.0000	0.123±0.0000	0.578±0.0000	0.156±0.0000	**0.679±0.0000**	0.495±0.0000	0.629±0.0000	0.191±0.0000	0.393±0.0000	0.097±0.0000	0.680±0.0000	0.697±0.0000	0.414±0.0000	0.108±0.0000
LOF	0.699±0.0000	0.067±0.0000	0.750±0.0000	0.500±0.0000	0.754±0.0000	0.111±0.0000	0.595±0.0000	0.120±0.0000	**0.679±0.0000**	0.405±0.0000	0.656±0.0000	0.167±0.0000	0.566±0.0000	0.174±0.0000	0.680±0.0000	0.370±0.0000	0.588±0.0000	0.181±0.0000
THOC	0.737±0.0053	0.128±0.0092	0.725±0.0028	0.766±0.0136	0.808±0.0053	0.168±0.0050	0.633±0.0048	0.138±0.0018	0.715±0.0015	0.897±0.0061	0.685±0.0042	0.190±0.0014	0.439±0.0048	0.106±0.0008	0.680±0.0000	0.694±0.0019	0.459±0.0045	0.118±0.0009
DCdetector	0.507±0.0013	0.041±0.0001	0.665±0.0015	0.855±0.0044	0.530±0.0079	0.082±0.0002	0.495±0.0486	0.100±0.0060	0.688±0.0063	**0.938±0.0048**	0.534±0.0476	0.137±0.0111	0.484±0.0203	0.125±0.0029	0.704±0.0129	**0.973±0.0046**	0.492±0.0192	0.138±0.0038
DAGMM	0.791±0.0083	0.128±0.0249	0.762±0.0116	0.773±0.0712	0.845±0.0092	0.183±0.0240	0.636±0.0159	0.137±0.0050	0.713±0.0181	0.886±0.0119	0.686±0.0137	0.187±0.0050	0.479±0.0590	0.122±0.0154	0.680±0.0000	0.696±0.0068	0.497±0.0580	0.133±0.0164
AT	0.461±0.0196	0.039±0.0011	0.714±0.0039	0.903±0.0016	0.521±0.0146	0.068±0.0033	0.523±0.0083	0.101±0.0022	0.672±0.0032	0.934±0.0094	0.575±0.0086	0.137±0.0029	0.600±0.0175	0.156±0.0076	0.689±0.0016	0.964±0.0007	0.615±0.0173	0.171±0.0078
MEMTO	0.509±0.0008	0.048±0.0003	0.666±0.0024	0.779±0.0139	0.517±0.0007	0.082±0.0001	0.569±0.0401	0.120±0.0083	**0.679±0.0000**	0.462±0.2715	0.621±0.0390	0.163±0.0117	0.507±0.0028	0.128±0.0006	0.699±0.0043	0.907±0.0232	0.510±0.0031	0.142±0.0006
GANF	0.504±0.0416	0.146±0.0420	0.692±0.0090	0.325±0.0482	0.552±0.0405	0.146±0.0153	0.559±0.0124	0.222±0.0089	0.694±0.0047	0.477±0.1147	0.590±0.0130	0.239±0.0064	0.414±0.0597	0.151±0.0290	0.683±0.0024	0.520±0.0675	0.431±0.0567	0.155±0.0237
DiffAD	0.859±0.0000	0.062±0.0000	0.701±0.0000	0.415±0.0000	0.870±0.0000	0.516±0.0000	0.584±0.0000	0.160±0.0000	**0.679±0.0000**	0.757±0.0000	0.646±0.0000	0.201±0.0000	**0.612±0.0000**	0.099±0.0000	0.681±0.0000	0.698±0.0000	**0.744±0.0000**	**0.340±0.0000**
NSPR	**0.899±0.0043**	**0.346±0.0063**	**0.836±0.0016**	**0.869±0.0119**	**0.929±0.0045**	**0.412±0.0082**	0.758±0.0185	**0.392±0.0045**	0.678±0.0000	0.260±0.0607	0.794±0.0168	**0.361±0.0099**	0.604±0.0230	0.182±0.0225	**0.784±0.0146**	0.855±0.0324	0.621±0.0224	0.194±0.0203
CrossAD	0.871±0.0015	0.154±0.0005	0.674±0.0000	0.170±0.0000	0.914±0.0011	0.272±0.0020	**0.808±0.0001**	0.289±0.0007	**0.679±0.0000**	0.191±0.0000	**0.832±0.0001**	0.351±0.0005	0.595±0.0006	0.134±0.0003	0.763±0.0007	0.739±0.0021	0.626±0.0006	0.170±0.0002
Ours	0.627±0.0100	0.160±0.0237	0.693±0.0046	0.303±0.0171	0.656±0.0108	0.126±0.0044	0.749±0.0061	0.357±0.0132	**0.679±0.0000**	0.595±0.0724	0.772±0.0035	0.345±0.0099	0.605±0.0218	**0.241±0.0105**	0.715±0.0205	0.566±0.0353	0.637±0.0198	0.256±0.0024

Table 7: Results on SWaT, PSM, and WADI using V-ROC, AUC-PR, Aff-F1, PA-F1, R-ROC, and R-PR.

	SWaT						PSM						WADI					
Methods	V-ROC	AUC-PR	Aff-F1	PA-F1	R-ROC	R-PR	V-ROC	AUC-PR	Aff-F1	PA-F1	R-ROC	R-PR	V-ROC	AUC-PR	Aff-F1	PA-F1	R-ROC	R-PR
IF	0.736±0.0072	0.738±0.0037	0.711±0.0014	0.848±0.0072	0.760±0.0072	0.539±0.0084	0.694±0.0231	0.452±0.0299	0.733±0.0128	0.913±0.0075	0.722±0.0210	0.497±0.0263	0.716±0.0110	0.192±0.0090	0.723±0.0038	0.663±0.0183	0.737±0.0119	0.197±0.0098
OC-SVM	0.818±0.0000	0.812±0.0000	0.786±0.0000	0.868±0.0000	0.835±0.0000	0.694±0.0000	0.509±0.0000	0.374±0.0000	0.697±0.0000	0.776±0.0000	0.546±0.0000	0.382±0.0000	0.446±0.0000	0.052±0.0000	0.715±0.0000	0.157±0.0000	0.461±0.0000	0.063±0.0000
LOF	0.772±0.0000	0.798±0.0000	0.771±0.0000	0.890±0.0000	0.791±0.0000	0.637±0.0000	0.654±0.0000	0.421±0.0000	0.744±0.0000	0.760±0.0000	0.677±0.0000	0.454±0.0000	0.466±0.0000	0.204±0.0000	0.773±0.0000	0.273±0.0000	0.482±0.0000	0.117±0.0000
THOC	0.831±0.0090	0.738±0.0069	0.773±0.0067	0.893±0.0132	0.838±0.0092	0.673±0.0118	0.681±0.0066	0.447±0.0198	0.727±0.0064	0.975±0.0077	0.730±0.0066	0.494±0.0187	0.491±0.0110	0.055±0.0046	0.717±0.0012	0.397±0.1423	0.512±0.0120	0.062±0.0032
DCdetector	0.496±0.0008	0.121±0.0003	0.688±0.0016	**0.970±0.0013**	0.498±0.0008	0.127±0.0003	0.503±0.0025	0.278±0.0006	0.667±0.0051	0.976±0.0011	0.518±0.0021	0.311±0.0007	0.504±0.0021	0.057±0.0005	0.715±0.0047	0.937±0.0088	0.510±0.0091	0.064±0.0004
DAGMM	0.792±0.0274	0.641±0.1007	0.798±0.0245	0.893±0.0235	0.802±0.0265	0.559±0.0864	0.671±0.0240	0.448±0.0485	**0.768±0.0052**	0.938±0.0128	0.703±0.0233	0.486±0.0438	0.464±0.0348	0.135±0.0757	0.733±0.0204	0.362±0.1586	0.479±0.0355	0.094±0.0370
AT	0.495±0.0017	0.120±0.0004	0.698±0.0071	0.968±0.0011	0.497±0.0020	0.127±0.0004	0.485±0.0057	0.270±0.0030	0.672±0.0093	**0.979±0.0016**	0.523±0.0066	0.314±0.0047	0.499±0.0093	0.057±0.0007	0.712±0.0101	**0.901±0.0427**	0.502±0.0088	0.064±0.0008
MEMTO	0.658±0.0981	0.244±0.1052	0.714±0.0095	0.947±0.0107	0.660±0.0980	0.248±0.1090	0.511±0.0008	0.281±0.0004	0.670±0.0154	0.953±0.0062	0.522±0.0010	0.332±0.0003	0.397±0.0047	0.051±0.0013	0.711±0.0023	0.507±0.0238	0.409±0.0046	0.057±0.0007
GANF	0.734±0.0791	0.739±0.0353	0.719±0.0234	0.846±0.0184	0.746±0.0782	0.537±0.0903	0.469±0.0151	0.425±0.0122	0.710±0.0000	0.606±0.0280	0.496±0.0136	0.436±0.0095	0.435±0.0201	0.211±0.0029	0.732±0.0000	0.283±0.0000	0.445±0.0186	0.109±0.0020
DiffAD	0.465±0.0000	0.111±0.0000	0.693±0.0000	0.682±0.0000	0.554±0.0000	0.181±0.0000	0.535±0.0000	0.398±0.0000	0.699±0.0000	0.810±0.0000	0.584±0.0000	0.402±0.0000	0.470±0.0000	0.041±0.0000	0.713±0.0000	0.214±0.0000	0.527±0.0000	0.121±0.0000
NSPR	0.806±0.0295	**0.843±0.0085**	**0.789±0.0248**	0.885±0.0092	0.818±0.0286	**0.674±0.0314**	0.671±0.0183	0.561±0.0443	0.705±0.0119	0.768±0.0041	0.694±0.0190	0.524±0.0072	0.383±0.0243	0.068±0.0170	0.713±0.0003	0.340±0.1457	0.400±0.0239	0.069±0.0088
CrossAD	**0.838±0.0009**	0.632±0.0261	0.716±0.0012	0.870±0.0019	**0.859±0.0008**	0.598±0.0144	**0.730±0.0149**	0.443±0.0141	0.694±0.0000	0.861±0.0162	**0.781±0.0150**	0.616±0.0315	0.437±0.0005	0.047±0.0000	0.713±0.0000	0.109±0.0000	0.465±0.0005	0.063±0.0000
Ours	0.744±0.0148	0.842±0.0099	0.762±0.0183	0.896±0.0067	0.752±0.0156	0.626±0.0283	0.657±0.0130	**0.697±0.0195**	0.712±0.0048	0.810±0.0244	0.700±0.0143	**0.626±0.0138**	**0.796±0.0179**	**0.596±0.0167**	**0.790±0.0043**	0.718±0.0114	**0.830±0.0179**	**0.560±0.0194**

whereas the NF contributes only 20M. To evaluate the trade-off between complexity and anomaly detection



(a) NSPR (WADI) (b) CrossAD (WADI) (c) Ours (WADI)

Figure 11: Feature reconstruction comparison on WADI.

performance, we reduce the latent dimensions of both components from 256 to 64. This decreases the model size by 92% (250M→20M parameters), training time by 68.4% (54.8s→17.3s), and inference latency by 48.9% (235ms→120ms), while only marginally reducing ROC by 0.2% (0.907→0.905), F1 by 2.5% (0.836→0.815), and V-PR by 2.4% (0.617→0.602). In addition, as shown in Table 3, replacing diffusion sampling during inference with a precomputed mean latent representation from the training and validation data achieves comparable detection performance. This eliminates the need for inference-time sampling, thereby reducing computational overhead. These results demonstrate the potential to improve inference efficiency without compromising detection performance.

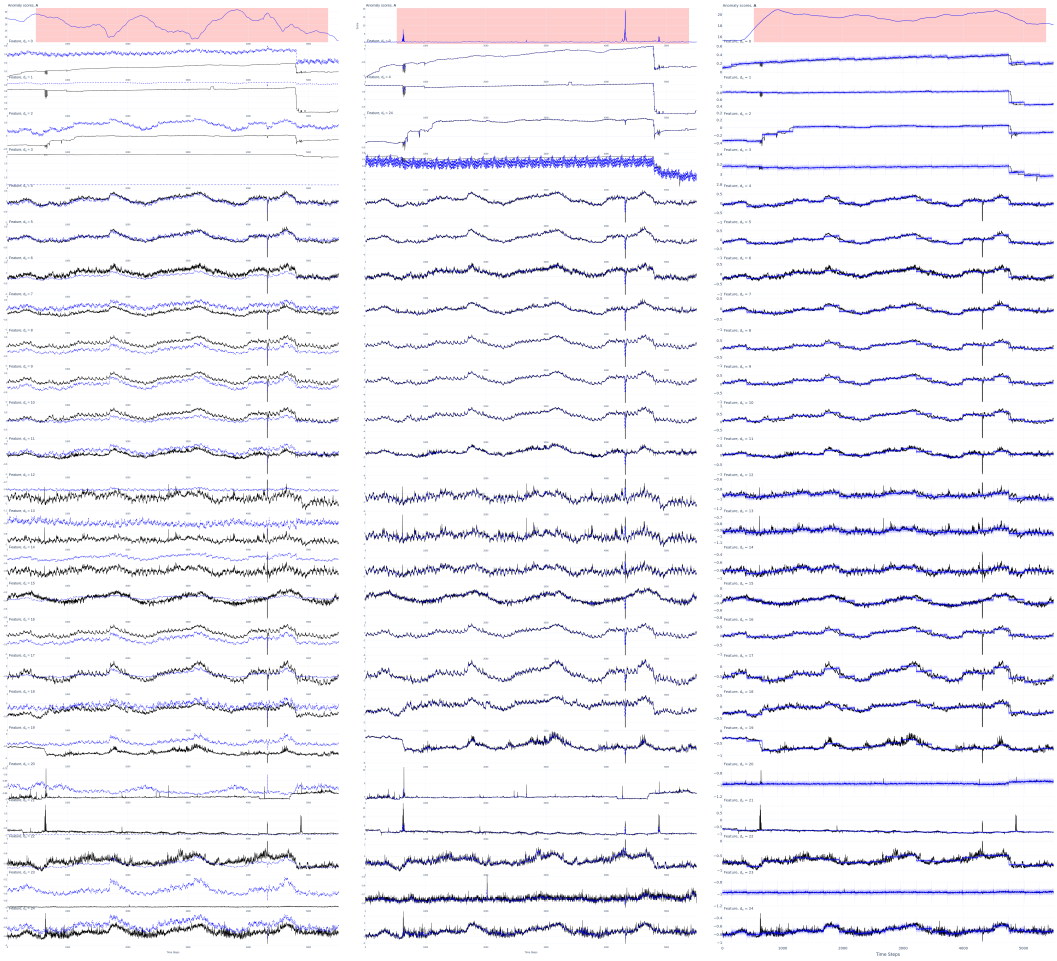
E.3 Visualization

For a comprehensive analysis, we provide full feature reconstruction results for our model and all baselines. Since NSPR produces both sequential and point-wise reconstructions, we report the sequence-wise variant for consistency. For CrossAD, which operates across multiple temporal scales, we present the original window-scale reconstruction. For PSM (Figure 12), all features are shown, whereas for WADI (Figure 13), we visualize a

Table 8: Comparison of training and inference efficiency.

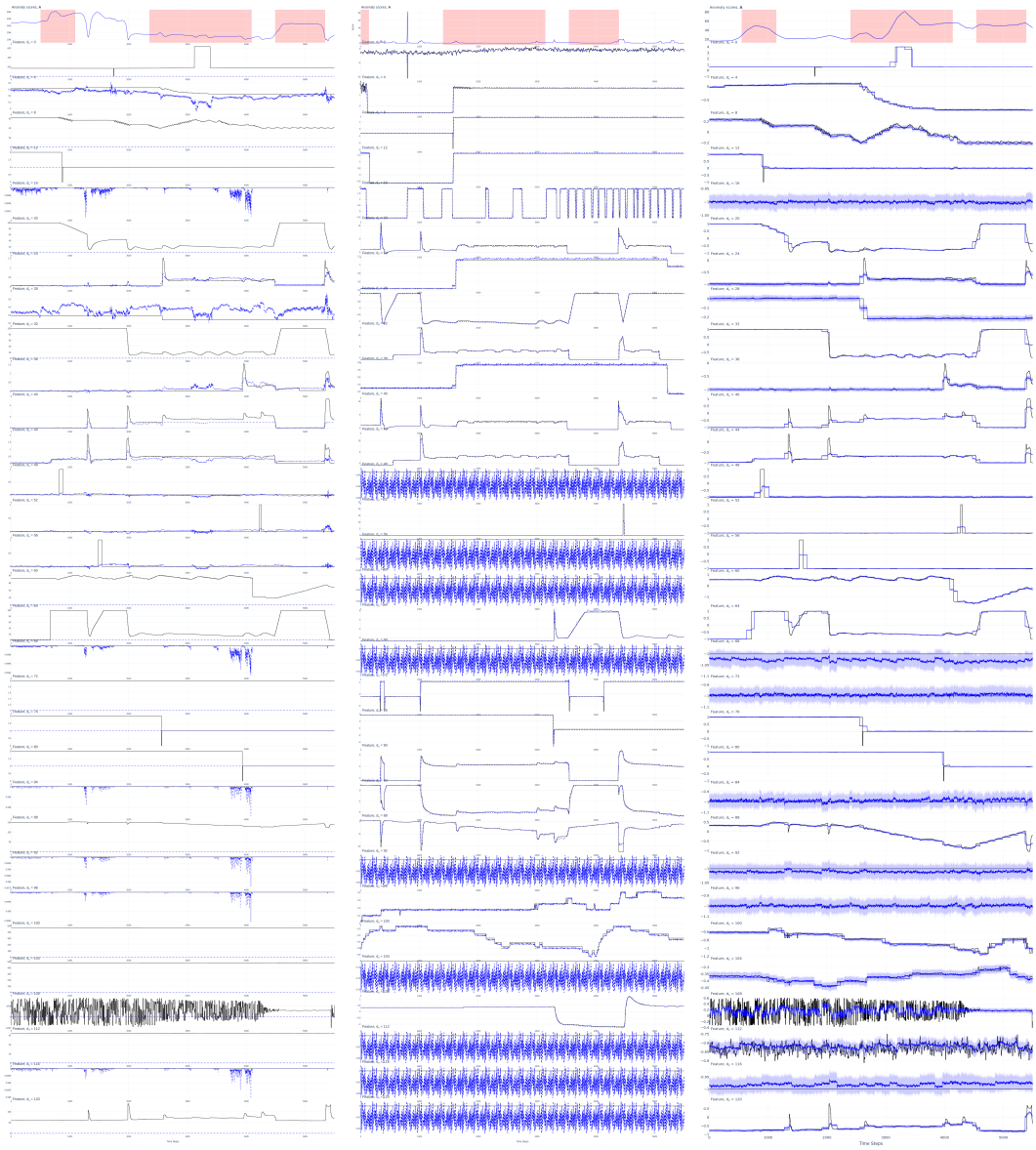
Model	Training Time / Epoch (s)	Inference Time / Window (ms)	Parameters (M)
CrossAD	45.5	46	1.30
NSPR	34.5	18	0.39
Flow+Diff	54.8	235	250

subset (one quarter) by plotting every fourth feature. The first row presents anomaly scores, with anomalous segments highlighted in red. Compared to the baselines, our method produces more stable anomaly scores. NSPR often yields large reconstruction errors with consistent offsets across features, reducing discriminative power by uniformly inflating errors over time. CrossAD achieves accurate reconstructions, making anomalous segments less distinguishable. In contrast, our method produces elevated anomaly scores during anomalous segments while maintaining well-estimated reconstructions, enabling clearer identification of potential anomaly-contributing features. Figure 14 presents the anomaly score distributions of NSPR and CrossAD. In comparison, our method shows better separation between normal and anomalous samples, highlighting the effectiveness of our approach.



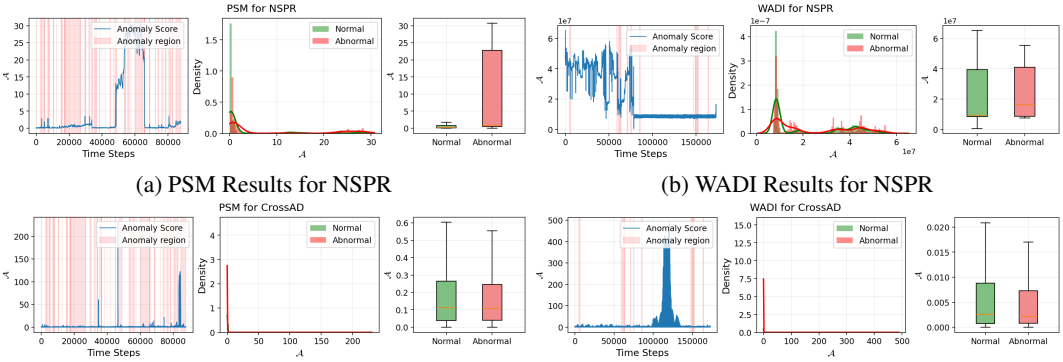
(a) NSPR (PSM) (b) CrossAD (PSM) (c) Ours (PSM)

Figure 12: Feature reconstruction comparison on PSM.



(a) NSPR (WADI) (b) CrossAD (WADI) (c) Ours (WADI)

Figure 13: Feature reconstruction comparison on WADI.



(a) PSM Results for NSPR (b) WADI Results for NSPR

(c) PSM Results for CrossAD (d) WADI Results for CrossAD

Figure 14: Anomaly score distributions of NSPR and CrossAD for PSM and WADI datasets

F Supplementary Ablation Studies

In this section, we provide details of the model and anomaly scoring variants, followed by additional experiments investigating model variations, scoring strategies, model analysis, and hyperparameter sensitivity.

F.1 Model variants

The following explains the design of model variants:

- **Flow-Only**: This variant removes the diffusion component and relies solely on the NF with a multivariate standard Gaussian base distribution, $p_{\mathcal{Z}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, while keeping all other components unchanged. This variant achieves performance comparable to the proposed model.
- **Flow+Enc**: The diffusion model is replaced by a transformer encoder $\mathrm{TransEnc}(\cdot)$ that directly maps the mean offset input $\mathbf{X}'$ to the base distribution of the NF, $p_{\mathcal{Z}} = \mathcal{N}\big(\boldsymbol{\mu}_{\mathrm{Enc}}, \tilde{\sigma}^2_{\mathrm{Enc}}\mathbf{I}\big)$, where $\boldsymbol{\mu}_{\mathrm{Enc}} = \mathrm{TransEnc}(\mathbf{X}')$ and $\tilde{\sigma}^2_{\mathrm{Enc}}\mathbf{I}$ is equivalent to $\tilde{\sigma}^2_{\mathcal{D}}\mathbf{I}$, representing a data-independent learnable parameter that provides an adaptive scale during training.
- **MLP-Flow**: The transformer encoder $\mathrm{TransEnc}(\cdot)$ in the NF is replaced with a three-layer MLP with ReLU activations.
- **wo-Asig**: The bounded function Asig is removed from the final NF transformation layer, and the diffusion inference step is performed without a bounded function.
- **wo-Cond**: The denoising network operates without conditioning. The results demonstrate that conditioning is beneficial, as removing it leads to degraded performance.
- **wo-LMA**: The Local Mean Augmentation (LMA) module has been removed.

F.2 Anomaly scoring criterion analysis

The following presents the design of different anomaly scoring variants:

- **MSE($\hat{\mathbf{X}}$)**: Anomaly scoring is based on the mean squared reconstruction error in the time-domain, aggregated using a rolling window.
- **Without averaging (wo-Avg)**: The rolling window is omitted, and the anomaly score is computed solely using (11).
- **Latent Likelihood (Lat-LL)**: Anomaly scores are computed using the negative log-likelihood of the flow-derived latent representation, parameterized by the empirical mean in (10) and the corresponding standard deviation described as

$$\hat{\boldsymbol{\sigma}}_{\mathcal{D}} = \sqrt{\frac{1}{K}\sum_{k=1}^{K}(\mathbf{Z}_{\mathcal{D},k} - \hat{\boldsymbol{\mu}}_{\mathcal{D}})^2}.$$

  Accordingly, the discrepancy score at time step i is defined as the negative log-likelihood under a multivariate Gaussian distribution,

$$\mathrm{DiscrepancyScore}_i = -\log\mathcal{N}\big(\mathbf{Z}_{\mathcal{F},i}\,;\,\hat{\boldsymbol{\mu}}_{\mathcal{D},i}, \mathrm{diag}\big(\hat{\boldsymbol{\sigma}}^2_{\mathcal{D},i}\big)\big)\,,$$

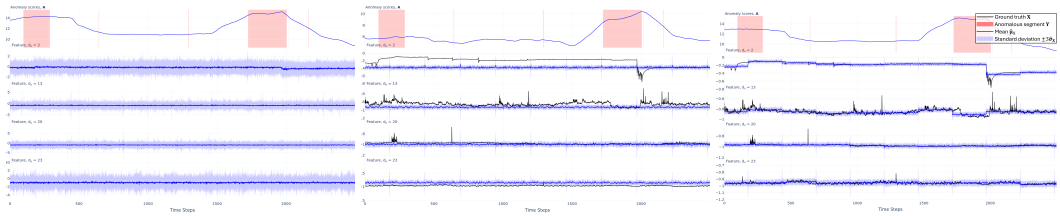
  where $\mathcal{N}(\cdot)$ denotes a multivariate Gaussian distribution with diagonal covariance. Here, $\hat{\boldsymbol{\mu}}_{\mathcal{D},i}$ and $\hat{\boldsymbol{\sigma}}^2_{\mathcal{D},i}$ denote the mean and variance vectors corresponding to the i-th time step. Without incorporating $\hat{\boldsymbol{\sigma}}^2_{\mathcal{D},i}$, this variant reduces to the formulation in (11).
- **NF Likelihood (NF-LL)**: Anomaly scoring is performed using the log-likelihood defined in (9), which is identical to the training loss. This variant consists of a likelihood term and a log-determinant term.

F.3 Additional ablation study results

Table 9 presents additional results from the ablation studies. Although Figure 15 shows comparable anomaly scores across different model variants, our method better preserves the structure of the underlying data distribution during reconstruction. Figure 16 presents anomaly scores for different anomaly scoring variants. We observe that the reconstruction error MSE($\hat{\mathbf{X}}$) is the least sensitive to anomalous segments. Meanwhile, wo-Avg exhibits erratic patterns, indicating that averaging over the window helps produce more discriminative scores. For NF-LL, Lat-LL, and our method, the results are largely consistent, which is expected since all three leverage latent representations for scoring, further supporting the usefulness of latent-space representations for anomaly detection.

Table 9: Ablation study of model component and anomaly scoring variants

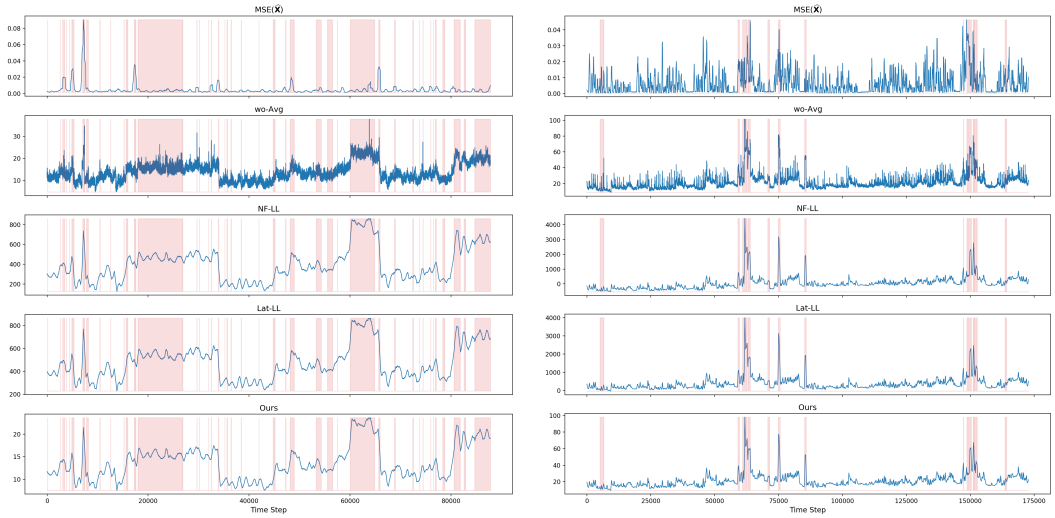
		Model variants						Anomaly scoring criteria				
Dataset	Metric	Flow-Only	Flow+Enc	MLP-Flow	wo-Asig	wo-Cond	wo-LMA	MSE($\hat{\mathbf{X}}$)	wo-Avg	NF-LL	Lat-LL	Ours
MSL	ROC	0.740±0.0647	0.689±0.0780	0.627±0.0264	0.468±0.0400	0.765±0.0792	0.707±0.0787	0.793±0.0004	0.704±0.0113	0.809±0.0128	0.808±0.0129	**0.811±0.0128**
	F1	0.344±0.0082	0.316±0.0518	0.314±0.0427	0.201±0.0032	0.370±0.0724	0.367±0.0256	0.390±0.0025	0.307±0.0207	0.405±0.0247	0.405±0.0247	**0.413±0.0295**
	V-PR	0.250±0.0425	0.203±0.0507	0.180±0.0141	0.114±0.0097	0.270±0.0680	0.233±0.0222	0.288±0.0004	0.227±0.0148	0.302±0.0125	0.302±0.0125	**0.304±0.0148**
SMAP	ROC	**0.708±0.0308**	0.688±0.0204	0.440±0.0129	0.447±0.0393	0.676±0.0595	0.650±0.1242	0.577±0.0017	0.668±0.0099	0.699±0.0114	0.699±0.0113	0.701±0.0122
	F1	**0.354±0.0180**	0.337±0.0270	0.232±0.0030	0.235±0.0051	0.314±0.0271	0.329±0.0639	0.291±0.0034	0.311±0.0083	0.347±0.0038	0.347±0.0036	0.349±0.0042
	V-PR	**0.240±0.0144**	0.223±0.0125	0.138±0.0050	0.135±0.0117	0.220±0.0220	0.217±0.0490	0.171±0.0004	0.245±0.0176	0.231±0.0030	0.231±0.0029	0.232±0.0025
SWaT	ROC	0.897±0.0126	0.908±0.0034	**0.914±0.0036**	0.880±0.0087	0.906±0.0027	0.897±0.0126	0.249±0.0008	0.905±0.0056	0.908±0.0066	0.908±0.0066	0.907±0.0055
	F1	0.823±0.0197	**0.837±0.0135**	0.836±0.0096	0.803±0.0116	0.830±0.0245	0.823±0.0197	0.217±0.0000	0.834±0.0064	0.836±0.0075	0.836±0.0075	0.836±0.0042
	V-PR	0.585±0.0397	0.616±0.0184	**0.646±0.0099**	0.542±0.0294	0.607±0.0285	0.585±0.0397	0.092±0.0011	0.671±0.0311	0.617±0.0364	0.617±0.0366	0.617±0.0306
PSM	ROC	0.835±0.0221	0.834±0.0141	0.741±0.0255	0.569±0.0145	0.830±0.0253	0.750±0.0316	0.604±0.0030	0.831±0.0127	0.835±0.0132	0.837±0.0101	**0.837±0.0105**
	F1	0.646±0.0303	0.643±0.0258	0.542±0.0297	0.460±0.0145	0.648±0.0400	0.564±0.0439	0.471±0.0035	0.622±0.0198	0.654±0.0214	0.655±0.0251	**0.665±0.0164**
	V-PR	**0.578±0.0279**	0.567±0.0162	0.515±0.0184	0.378±0.0061	0.566±0.0246	0.498±0.0176	0.464±0.0034	0.688±0.0147	0.557±0.0104	0.565±0.0130	0.557±0.0111
WADI	ROC	0.808±0.0128	0.798±0.0048	0.795±0.0221	0.626±0.1038	0.803±0.0083	0.808±0.0259	0.726±0.0048	0.804±0.0071	0.808±0.0067	**0.817±0.0079**	0.809±0.0061
	F1	0.587±0.0354	0.631±0.0188	0.583±0.0215	0.283±0.0286	0.618±0.0174	**0.638±0.0349**	0.249±0.0062	0.628±0.0153	0.621±0.0145	0.631±0.0146	0.626±0.0093
	V-PR	0.481±0.0383	0.522±0.0245	0.447±0.0242	0.145±0.0523	0.494±0.0137	0.534±0.0131	0.183±0.0063	**0.567±0.0156**	0.528±0.0159	0.534±0.0126	0.524±0.0189



(a) Flow-Only (b) wo-LMA (c) Ours

Figure 15: Flow-Only, wo-LMA, and our model on PSM.

Local Mean Augmentation Module Analysis Direct mean subtraction centres the input but discards mean-level information, whereas LMA represents the input using both the centred residual and the local mean, preserving this information. Following your suggestion [a], we evaluated whether the performance gain is solely due to the increased embedding dimension by comparing LMA with two variants of the same embedding dimension: (i) mean subtraction and (ii) concatenated mean. As shown below, LMA consistently outperforms both variants, suggesting that the performance gain arises from the proposed representation rather than the increased embedding dimension alone.



(a) PSM (b) WADI

Figure 16: Anomaly scores over time for different scoring variants.

Table 10: Ablation study of mean subtraction and mean concentration.

	MSL			PSM			WADI		
Method	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR
Mean subtraction only	0.785	0.341	0.255	0.623	0.465	0.479	0.773	0.341	0.272
Mean concentration only	0.724	0.310	0.207	0.813	0.619	0.539	0.803	0.621	**0.525**
Ours	**0.811**	**0.413**	**0.304**	**0.837**	**0.665**	**0.557**	**0.809**	**0.626**	0.524

Replacement of NF with encoder Although our framework is not inherently restricted to NFs, alternative latent encoders may be viable when invertibility and latent-space interpretability are not primary objectives. To preliminarily investigate this possibility, we replaced the NF with a Transformer encoder while keeping the remaining components unchanged. As shown in Table 11, this replacement resulted in consistently lower performance across all three datasets. Further investigation of alternative encoder architectures and their integration into the proposed framework is left for future work.

Table 11: Ablation study of flow replacement with a Transformer encoder.

	MSL			PSM			WADI		
Method	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR
Flow replaced with Transformer encoder	0.520	0.201	0.128	0.526	0.438	0.359	0.523	0.117	0.062
Ours	**0.811**	**0.413**	**0.304**	**0.837**	**0.665**	**0.557**	**0.809**	**0.626**	**0.524**

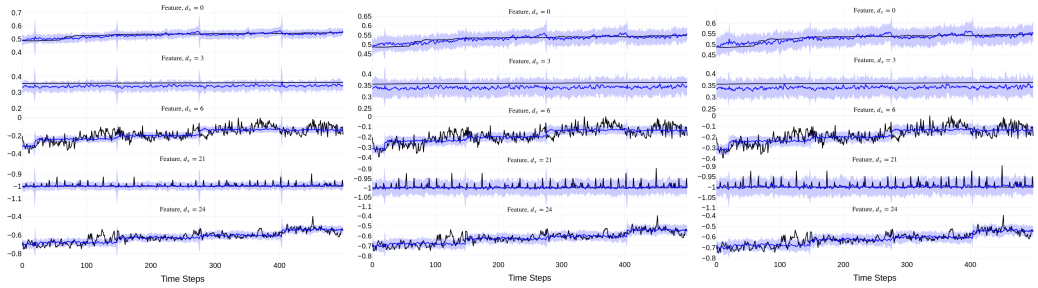
Sensitivity to Normal Training Data Diversity To evaluate the sensitivity of our method to the amount of normal training data, we conducted an additional experiment by randomly sampling 25%, 50%, and 75% of the training set while keeping the validation and test sets unchanged. Overall, the proposed method exhibits limited sensitivity to the amount of normal training data. While additional data improves performance, particularly on MSL, the model remains robust on PSM and WADI even with only 25% of the training data, indicating good data efficiency and generalization.

Table 12: Ablation study of masking ratio.

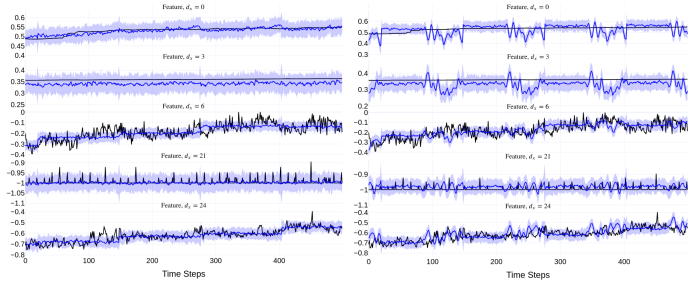
Normal Training Data (%)	25%			50%			75%			100%		
Dataset	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR
MSL	0.596	0.276	0.170	0.676	0.359	0.232	0.780	0.347	0.275	**0.811**	**0.413**	**0.304**
PSM	**0.839**	**0.665**	**0.581**	0.832	0.649	0.571	0.836	0.645	0.567	0.837	**0.665**	0.557
WADI	0.795	0.613	0.495	0.805	0.618	0.512	0.808	0.620	0.508	**0.809**	**0.626**	**0.524**

Threshold Selection for F1-score Computation Following prior work, we report the Best-F1 score, obtained using the threshold that maximizes the F1-score. To assess the practical applicability of our method, we evaluate an unsupervised thresholding strategy based solely on validation anomaly scores. Specifically, we apply K-means clustering ($k = 2$) to partition the scores into normal and high-score clusters, and estimate the threshold as $\tau = \mu_c + 2\sigma_c$, where μ_c and σ_c denote the mean and standard deviation of the lower-score cluster, respectively. The estimated thresholds closely approximate the optimal thresholds on MSL and PSM, achieving F1-scores of 0.390 and 0.637, respectively, compared with the optimal scores of 0.413 and 0.665. However, on WADI, K-means to underestimate the threshold (22.6 vs. 35.2), resulting in a substantially lower F1-score of 0.267 compared with the optimal score of 0.626. This limitation highlights the need for more robust unsupervised threshold estimation methods for certain datasets to achieve practical applicability of the proposed approach.

Discrete inputs Discrete input values have been shown to affect model performance [12, 39]. To systematically examine this effect, we evaluate models with and without discrete inputs under a fixed random seed. Feature analysis reveals that SMAP contains only 1 continuous feature out of 25, with the remainder being discrete. Similar patterns are observed in MSL (1/55 continuous), SWaT (25/51 continuous), and WADI (67/123 continuous). In addition, we conduct an ablation study by removing all discrete inputs. As shown in Table 13, removing discrete features improves performance on datasets dominated by discrete features, such as SMAP and MSL, where discrete signals may violate the assumptions of continuous density modeling. However, this effect varies across datasets: on datasets with a higher proportion of continuous features, such as SWaT and WADI, removing discrete inputs leads to performance degradation. This suggests that discrete features, while potentially harmful in isolation, can still provide complementary information when sufficiently integrated with continuous signals.



(a) $N = 5$ (b) $N = 10$ (c) $N = 14$



(d) $N = 20$ (e) $N = 50$

Figure 17: These plots show a subset of features from the estimated distributions on the PSM dataset for different values of N. The Black line represents the ground truth $\mathbf{X}$, while the blue line and shaded area indicate the estimated mean $\hat{\boldsymbol{\mu}}_X$ and three times of the standard deviation $3\hat{\boldsymbol{\sigma}}_X$, respectively.

Table 13: Results of without and with discrete inputs.

Features	MSL			SMAP			SWaT			WADI		
	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR	ROC	F1	V-PR
wo discreet	**0.811**	**0.413**	**0.304**	**0.688**	**0.336**	**0.220**	0.867	0.796	0.471	0.721	0.434	0.259
All features	0.693	0.287	0.190	0.533	0.244	0.169	**0.907**	**0.836**	**0.617**	**0.809**	**0.629**	**0.524**

Handling discrete inputs in SMAP and MSL Normalizing flows are designed for continuous-valued inputs and may be more challenging to train when a large proportion of the input variables are discrete. This issue is particularly relevant to MSL and SMAP, where most input variables are discrete. We therefore adopt zero masking to retain the original multivariate input while preserving the invertibility of the flow without modifying the network architecture. As shown in Table 14, using the raw inputs results in the lowest performance. Removing the discrete features improves the results, while zero masking achieves the best overall performance.

Table 14: Ablation study of input masking strategies.

Method	MSL			SMAP		
	ROC	F1	V-PR	ROC	F1	V-PR
Raw Input (No Masking)	0.690	0.284	0.187	0.492	0.243	0.151
Remove Discrete Features	0.756	0.338	0.223	0.676	0.314	0.218
Ours (Zero Masking)	**0.811**	**0.413**	**0.304**	**0.701**	**0.349**	**0.232**

Analysis of SMD dataset Although F1 scores on SMD are lower, ROC performance remains competitive. As shown in Fig. 18 and Table 4, SMD anomalies are short and abrupt. The model produces clear local increases in anomaly scores during anomalous segments (green boxes in Fig. 18), indicating effective detection of local deviations. However, F1-based thresholding under a global decision boundary may fail to capture such short segments consistently, highlighting the need for global context-aware score normalization for more reliable segment-level detection.

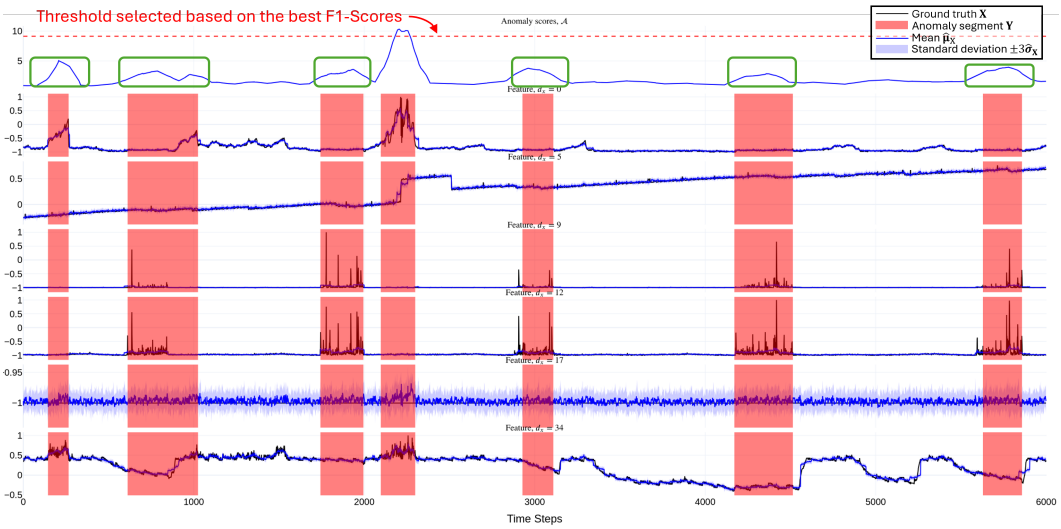


Figure 18: Estimated mean $\hat{\boldsymbol{\mu}}_{\mathbf{X}}$ and standard deviation $3\hat{\boldsymbol{\sigma}}_{\mathbf{X}}$ for SMD dataset.

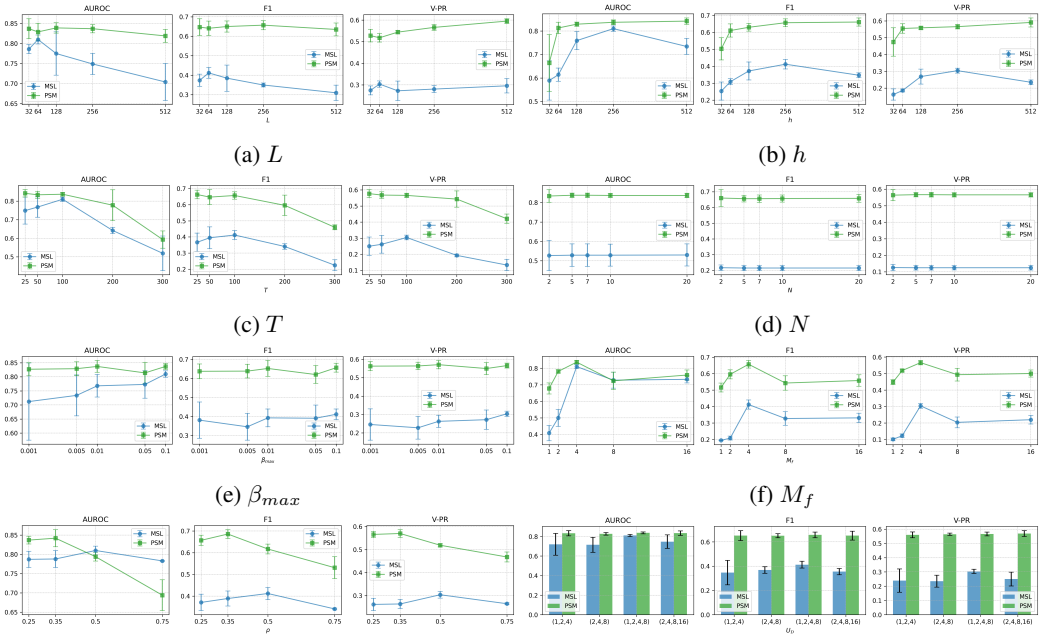
F.4 Hyperparameter Sensitivity Analysis

To investigate the effect of different hyperparameter settings, we analyze their impact on performance by fixing all other hyperparameters and varying one hyperparameter at a time. The predefined search ranges considered for each hyperparameter are summarized as follows:

- **Input window length**: $L \in \{32, 64, 128, 256, 512\}$, with a stride of $st = L/4$.
- **Hidden dimension**: $h \in \{32, 64, 128, 256, 512\}$
- **Number of diffusion steps**: $T \in \{25, 50, 100, 200, 300\}$
- **Number of diffusion strides** $N \in \{5, 10, 14, 20, 50\}$
- **Maximum diffusion noise level**: $\beta_{max} \in \{1 \times 10^{-3}, 5 \times 10^{-3}, 1 \times 10^{-2}, 5 \times 10^{-2}, 1 \times 10^{-1}\}$
- **Number of flow transformation layers**: $M_f \in \{1, 2, 4, 8, 16\}$
- **Masking ratio**: $\rho \in \{0.25, 0.35, 0.5, 0.75\}$
- **U-Net channel multiplier configurations**: $U_{\mathcal{D}} \in \{(1, 2, 4), (2, 4, 8), (1, 2, 4, 8), (2, 4, 8, 16)\}$.

From the plots in Fig. 19, the optimal values of h range between 64 and 256. Since performance varies only marginally within this range, smaller values of h are generally preferred to reduce computational cost, as the our model uses a transformer architecture with a complexity of $\mathcal{O}(L^2h + Lh^2)$, which grows quadratically with the hidden dimension h. The time step $L \in \{64, 128\}$ provides a favorable trade-off between performance and efficiency, with smaller values of L offering reduced computational overhead. The parameter 1×10^{-1} yields improved performance and more stable results across different random seeds, highlighting the benefit of stochasticity. As the number of diffusion steps T increases, performance degrades, with the optimal value achieved at T=100. We also observe that adjusting the diffusion stride N has little impact on detection performance. However, as shown in Fig. 17, varying N can substantially affect the quality of the estimated distribution, which in turn influences the interpretation of affected features when an anomalous segment is detected. Specifically, large values of N result in overly narrow distributions, whereas excessively small values lead to unstable mean estimates. Since adjusting N does not require retraining the model, calibration can be performed efficiently by inspecting the fitness of the estimated distribution. The U-Net channel multiplier $U_{\mathcal{D}}$ exhibits minimal sensitivity to different configurations. Fixing it to $U_{\mathcal{D}} = (1, 2, 4, 8)$ consistently yields stable and optimal performance across datasets. The parameters M_f and p are inherently coupled, as p defines the masking ratio in the normalizing flow. If M_f is too small, the masking cycle is not fully covered, and some features may never be transformed, making the transformation jointly dependent on both parameters. Fig. 19f shows that $M_f = 4$ is optimal: for MSL with $\rho = 0.5$, the masking pattern overlaps twice across layers, while for PSM with $\rho = 0.25$, all features are guaranteed to be transformed at least once.

Window size is dataset-dependent To reduce dataset-specific tuning, we conducted a sensitivity analysis of the window size L. As shown in Table 15, although a window size of 64 achieves slightly higher average ROC and F1 scores, this improvement is primarily driven by MSL, whereas a window size of 128 provides more consistent performance across datasets. Larger windows improve V-PR but may compromise ROC and F1 due



(a) L (b) h

(c) T (d) N

(e) β_{max} (f) M_f

(g) ρ (h) $U_{\mathcal{D}}$

Figure 19: Sensitivity analysis of model performance with respect to hyperparameters.

to reduced anomaly localization precision. Therefore, we recommend a fixed window size of 128 as a robust default.

Table 15: Ablation study on the effect of sequence length L.

	MSL					PSM					WADI				
Metric	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
ROC	0.786	**0.810**	0.775	0.749	0.704	0.837	0.829	**0.839**	0.837	0.819	0.802	0.806	**0.810**	0.800	0.800
F1	0.374	**0.412**	0.385	0.350	0.310	0.647	0.641	0.650	**0.657**	0.635	0.618	**0.627**	0.619	0.595	0.602
V-PR	0.275	**0.304**	0.273	0.281	0.296	0.528	0.518	0.544	0.566	**0.595**	0.483	0.494	0.519	0.544	**0.564**

G Broader Impacts

With the emergence of smart grid technologies, including Internet of Things (IoT) devices deployed across water, gas, and electricity networks, the volume of generated data has increased significantly, making it more challenging to extract meaningful insights. These insights are important for utilities, as they enable early detection of anomalies and support preventive maintenance and operational decision-making. Moreover, digitalization increases the exposure of critical infrastructure to cyber threats due to expanded connectivity and attack surfaces. As a result, detecting anomalies in network traffic can improve cybersecurity and system resilience. In this context, multivariate time series anomaly detection plays a crucial role in monitoring complex, interdependent sensor signals and supporting the reliable operation of critical infrastructure systems. Beyond accurate multivariate time series anomaly detection, our model also provides feature-wise interpretability, enabling the analysis of the features contributing to anomalies.

H Limitation

Although Flow+Diff achieves strong overall performance, it is less effective on the SMD dataset, where anomalies are typically short and exhibit erratic changes (Figure 18). While the model produces elevated anomaly scores over anomalous segments, these increases are insufficient to reliably distinguish anomalies from normal patterns at the global level. This highlights the need for robust score normalization or adaptive thresholding to improve detection reliability. Furthermore, our ablation studies reveal that obtaining appropriate thresholds without labels remains challenging for certain datasets, limiting the reliability and practical applicability of the proposed approach.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly articulate the contributions of our work to address key limitations in existing multivariate time series anomaly detection methods.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discuss our model limitation in Appendix H.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer:[Yes]

   Justification: We append an affine-sigmoid transformation to the final layer of the normalizing flow, yielding an additional log-determinant Jacobian term. The derivation is provided in Appendix C.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.

- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the algorithmic procedure to reproduce our model in Appendix C.4, and the code is publicly available at `https://github.com/vanerkz/FlowPlusDiff`.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All datasets used are publicly available, and the code is publicly available at `https://github.com/vanerkz/FlowPlusDiff`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We include all experimental setup details in Appendix D.2 and Appendix F.4, and dataset details in Appendix D.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We run all experiments with 5 random seeds and report both the mean and standard deviation. These details are included for the main results in Appendix E.1, ablation studies in Appendix F.3, and hyperparameter analysis in Appendix F.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Computational resource details are provided in Appendix D.1.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm that we have read the NeurIPS Code of Ethics and fully comply with it.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Multivariate time series anomaly detection is crucial, as it identifies anomalous data and enables end-users to take preventive measures before significant impacts occur. Further discussion is provided in Appendix G.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our model is designed for time series anomaly detection and is not intended for generative applications; thus, it does not pose any risk of misuse.

    Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original papers corresponding to all datasets used in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Code is publicly available at https://github.com/vanerkz/flowplusdiff.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.