École nationale
de la statistique
et de l'analyse
de l'information

DANG, Ba Khuong

Non-confidential

GRADUATION INTERNSHIP REPORT

Centre Inria de l'Université Grenoble Alpes

LONGITUDINAL UNSUPERVISED ANOMALY DETECTION (UAD) WITH DIFFUSION MODELS

INTERNSHIP LOCATION :

CITY: Montbonnot COUNTRY: France

Academic Year 2024-2025

Internship Supervisor: Florence FORBES

ENSAI Internship Advisor: François PORTIER

Longitudinal Unsupervised Anomaly Detection (UAD) with diffusion models

Ba-Khuong Dang
ENSAI
INRIA Grenoble

Acknowledgements

I would like to express my sincere gratitude for the opportunity to work on this project with Statify team at Inria Grenoble. Special thanks to my internship supervisors Dr. Florence Forbes for her constructive feedback and advice throughout the entire project. Her expertise and insightful guidance were instrumental in addressing some of our key challenges. Additionally, I extend my appreciation to my ENSAI professors, specially my internship advisor professor Francois Portier, for their amazing academic guidance.

Contents

List of Acronyms

AD Anomaly Detection

AUROC Area Under the Receiver Operating Characteristic curve

AUPRC Area Under the Precision-Recall curve

DDIMs Denoising implicit models

DDPMs Denoising Probabilistic Diffusion Models

DMs Diffusion Models

FA Feature Attention

FAM Feature Attention Module

FN False Negative

FP False Positive

GAN Generative Adversarial Networks

GMs Generative Models

LAF Longitudinal Attention Fusion

LAFM Longitudinal Attention Fusion Module

SDM Spatial Diffusion Model

TDM Temporal Diffusion Model

TN True Negative

TP True Positive

UAD Unsupervised Anomaly Detection

VAE Variational Autoencoders

List of Notations / Symbols

$\mathbf{X}^i_{1\leq i\leq N}$	Set of observed images of patient i through time
N	Number of patients
$\mathbf{x}^i_l$	Observation of patient i at time point l. We often omit superscript i and subscript l for brevity.
$l \in [1 \dots L]$	Longitudinal time point (time indexed)
$\mathbf{x}$	Original image (potential anomalous) $\in \mathbb{R}^{h\times w}$
$\hat{\mathbf{x}}$	Reconstructed image (can be psuedo-healthy) $\in \mathbb{R}^{h\times w}$
D_p	Pixel anomaly score map / pixel distance $\in \mathbb{R}^{h\times w}$
D_f	Feature distance $\in \mathbb{R}^{h\times w}$
$q(\mathbf{x})$	True probability distribution of $\mathbf{x}$ that we want to learn and sample from.
$p_\theta(\mathbf{x})$	Estimated distribution of $\mathbf{x}$ from diffusion model.
ϵ_θ	Output of diffusion model, with parameter ϵ.
$\mu_\theta(t), \Sigma_\theta(t)$	Mean and variance of predicted posterior of diffusion model.
Enc_ϕ	Semantic encoder with parameter ϕ
$\mathbf{y}$ ($\mathbf{y}_{\text{sem}}$)	Semantic representation of $\mathbf{x} \in \mathbb{R}^d$
β_t	Noise schedule of diffusion model.
$t \in [1 \dots T]$	Time step of diffusion model
Φ (FE_Φ)	Feaure extractor network from FAM, with parameter Φ
λ_{DL}	Hyperparameter of Feature Extract network controls the strength of distillation losses
v	Hyperparameter of FAM that controls the importance of pixel error
$\mathbf{s}$, D_{anomaly}	Anomaly score map $\in \mathbb{R}^{h\times w}$ from FAM
$\mathbf{S}_i$	Set of anomaly score map for L time point for 1 patient i
$\mathbf{s}_{\text{ref}}$	Reference anomaly score map $\in \mathbb{R}^{(L-1)\times h\times w}$ for 1 target anomaly score map (LAFM module)
$\mathbf{s}_{\text{temp}}$	Updated anomaly score map from LAFM after temporal smoothing operator
$\mathbf{S}_{\text{temp}}$	Set of updated anomaly score map from LAFM $\in \mathbb{R}^{L\times h\times w}$
β, γ	Hyperparameters of LAFM that control the correction of false positive and false negative
k_{mf}	Kernal size of median filter of LAFM
$\mathbf{X}^{\mathcal{O}}_i$	Set of observed data for patient i. We often omit subscript i for brevity
$\mathbf{X}^{\mathcal{M}}_i$	Set of missing data for patient i. We often omit subscript i for brevity
Enc_ω	Semantic encoder of TDM (RRDB net)
$\mathbf{Z}$	List of semantic representation from semantic encoder RRDB in TDM
$\mathbf{z}_{\text{sem},i} \in \mathbf{Z}$	Semantic representation from block i of RRDB $\in \mathbb{R}^{h\times w}$

List of Figures

List of Tables

Chapter 1

Introduction

Contents

Anomaly Detection (AD) and anomaly localization in medical images refer to the process of identifying abnormal areas or regions within images of various modalities, one of which is Magnetic Resonance Imaging (MRI) scans. Fully supervised deep learning approaches have demonstrated robust performance in assisting radiologists with MRI interpretation and automating tasks such as anomaly segmentation. For example, UNet-based algorithms have significantly contributed to advancements in automatic lesion segmentation. However, these methods require large datasets with image-level labels and pixel-level annotations for training, which are typically expensive and time consuming to obtain. Furthermore, the subtle nature of certain anomalies makes them difficult to detect, even for human experts. Consequently, rare and subtle anomalies may be underrepresented or entirely absent from training datasets, reducing the robustness of supervised approaches. An alternative is Unsupervised Anomaly Detection (UAD), which relies solely on healthy data rather than annotated pathological examples. The objective is to learn the underlying distribution of healthy brain MRI scans, and then to detect anomalies as deviations from this learned distribution. In recent study, [1] categorize UAD methods into four groups: image reconstruction-based, feature-modeling, attention-based and self-supervised anomaly detection methods. In this project, we will focus on the reconstruction-based method, in particular using diffusion models. Additionally, longitudinal data in MRI scans provide repeated measurements of the same subject over time, offering valuable insights into disease progression and temporal patterns. In the context of UAD, most Diffusion Models (DMs) treat each image independently (spatial dimension), without considering longitudinal dependencies (temporal dimension). Our goal is to investigate the viability of leveraging additional information provided by longitudinal data. Our source code is available at https://github.com/khuongdb/spatio-temporal-DM[1].

1.1 Reconstruction-based UAE with diffusion models

In reconstruction-based UAD, a model is trained exclusively on healthy data and, during inference, attempts to generate a pseudo-healthy version of the input image. The core assumption is that since the model has only seen healthy examples during training, it will fail to accurately reconstruct

[1] At the time of writing, we are still developing and expanding the project. The source code is not yet fully refined.

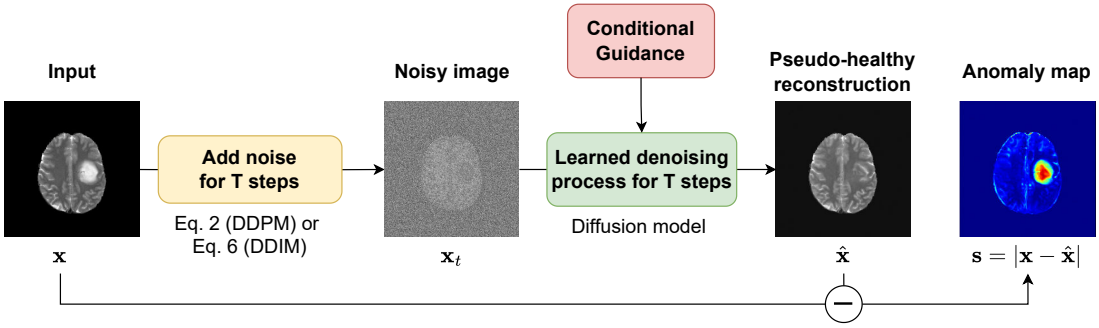


Figure 1.1: Overview of reconstruction-based anomaly segmentation with diffusion models. During training, the model learns the distribution of healthy dataset. At inference, we translate an input image to its pseudo-healthy counterpart $\hat{\mathbf{x}}$. The pixel-wise anomaly score map is given by the difference between input and output $D_p = |\mathbf{x} - \hat{\mathbf{x}}|$. Figure from [2]

pathological regions while still producing realistic healthy structures. The discrepancy between the input and its pseudo-healthy reconstruction is used to generate a residual map, where high error values indicate potential anomalies. The residual error map is often quantified using the pixel-wise $l1$-error: $D_p = |\mathbf{x} - \hat{\mathbf{x}}|$. Generative Models (GMs) known for their impressive ability to synthesize realistic images, have become prominent tools in this domain. In particular, Variational Autoencoders (VAE) [3] and Generative Adversarial Networks (GAN) [4] have been explored, as well as their variants. Recently, DMs have shown promise as generative models for UAD, due to their ability to produce high resolution synthesized images with accurate inpainting [5, 6, 7]. Fig. 1.1 shows an overview of anomaly segmentation task with diffusion models. Nevertheless, reconstruction-based UAD methods in general and diffusion models in particular face some critical challenges, mainly:

Reconstruction quality : even though generative models can produce very high quality samples, it is very unlikely that we can achieve near-exact reconstruction images, even those of healthy subjects. This introduces many false positive regions in our anomaly map, specially small errors in the background and around sharp edges of the object. This underlines the importance of developing models that can distinguish between residual errors caused by anomalies and those caused by model limitations.

Structure preservation : in diffusion model, we gradually add noise to input image (*forward process*) and remove them to recover healthy image (*denoise process*). The quantity and quality of noise added to image plays a crucial role in diffusion process, and this aspect is well studied in the literature. [7] highlight that a key challenge is to limit the reconstruction to healthy brain anatomy. On the one hand, we want to avoid models with limited reconstruction accuracy, which produce imperfect reconstructions everywhere. On the other hand, models that generate highly accurate reconstructions tend to perform a "copy task", resulting in unhealthy structures still reflected in the pseudo-healthy reconstructions. [8] refer to this as the *noise paradox*, which states that adding too much noise can destroy not only the anomalous regions but also the healthy structures of the brain. However, adding too little noise can preserve the structure but fail to corrupt the anomaly region, resulting in pseudo-healthy reconstructions that still contain unhealthy structures.

Thresholding anomaly score : Using a trained model, we can produce residual maps that can be interpreted as anomaly scores at both the pixel level and the image level. For anomaly segmentation task, a threshold is required to binarize the score map. At the image level, the

goal is to determine a threshold that separates healthy subjects from anomalous ones based on their anomaly scores. However, selecting an appropriate threshold remains an ongoing challenge in UAD. Most existing methods in the literature employ greedy algorithms to search for the optimal theoretical threshold based on evaluation metrics (e.g., AUROC, AUPRC). This approach, however, requires a test dataset with fully labeled images and ground-truth segmentations, which violates the principle of unsupervised learning. Other approaches either rely on fixed, predefined threshold values or use quantiles (e.g., the 95th or 97th percentile) of the training or test dataset. In general, selecting an appropriate threshold is challenging and can vary between datasets and applications.

1.2 Longitudinal data modeling

In the context of UAD in MRI images, longitudinal data represents repeated MRI scans of a patient at different time points. Unlike timeseries, the number of observations of a single patient is relatively small, and their frequency can be sparse due to patient dropouts, irregular follow-up intervals, and varying lengths of observation periods. Moreover, the complexity of patient metadata, such as age, sex, and disease stage, also affects the progression of disease in different ways. Therefore, we need a solution that can effectively capture both cohort-level (population-wide) and subject-level (individual) trajectories. In recent study, [9] show that spatio-temporal diffusion models have achieved state-of-the-art results across many modalities, including images, speech, video, and time series. However, the application of diffusion models to longitudinal data analysis, particularly for MRI anomaly detection, remains underexplored.

Existing literature on longitudinal diffusion models has largely focused on data imputation. However, these approaches rarely exploit the temporal dimension to strengthen spatial anomaly detection. We aim to analyze image sequences across different time points and fuse their information to improve detection and localization of anomalies. This approach can help identify anomalies at earlier stages (reducing false negatives) while filtering out nonpersistent anomalies (reducing false positives). Furthermore, we aim to predict missing observations using the available data and use these imputed images as additional information to improve anomaly localization in future scans.

Our project is inspired by recent advances in longitudinal modeling. [10] successfully combined a linear mixed-effects model with a VAE. In their approach, temporal dependencies are imposed in the latent space of the VAE. The mixed-effects model is used to capture the progression of biomarkers in Alzheimer's patients, while spatial variance is learned through the standard VAE. By jointly training these two components, their framework demonstrates the ability to model both spatial and temporal aspects in a unified manner. One major difference between these models and diffusion models is that, in VAE and normalizing flow frameworks, each observation $\mathbf{x}$ corresponds to a single latent variable $\mathbf{z}$. Furthermore, this latent variable is typically chosen to have a lower dimensionality, $d \ll D$. In contrast, in diffusion models, each $\mathbf{x}$ is associated with a sequence of latent variables $\mathbf{z}_t$ that have the same dimensionality as $\mathbf{x}$, making it more challenging to impose a temporal structure.

1.3 Problem setting

In this section, we present our problem setting and notations, and we outline the main challenges that we will try to address in this internship. Let $\mathbf{X}^i_{1\leq i\leq N}$ be our set of observed N individuals through time, assumed sampled *i.i.d.* from an unknown distribution p that does not depend on i. Each patient $i = 1, \ldots, N$ is a sequence of observations through time: $\mathbf{x}^i = (\mathbf{x}^i_1, \ldots, \mathbf{x}^i_{L_i})$, where, for each $j, \mathbf{x}^i_l \in \mathcal{X} := R^{H\times W\times D}$ is a high dimensional data (2D images or 3D MRI scans) and

L_i is the number of observations for a given individual. We note that $L_i \neq L_j$ for any $i \neq j$, which reflects our sparse longitudinal setting: different patients will have different number of observations at different time points. We use notation L to denote longitudinal time point to distinguish from the time step T of diffusion models.

Given a training set $\mathcal{H} = \{\mathbf{X}^{\mathbf{H,i}}\}$ of healthy subjects, our goal is to learn the underlying distribution $q(\mathbf{x}_j^{H,i})$ of each individual sample at any given time. At inference, an anomaly sample $\mathbf{x}_j^{A,i}$ is input to our model to produce a pseudo-healthy reconstruction $\hat{\mathbf{x}}_j^{PH,i}$ that follows healthy distribution $q_\theta(\mathbf{x}^{H,i})$ the network estimates. A residual map is calculated as the difference between original image and its reconstruction $|\mathbf{x} - \hat{\mathbf{x}}|$. We will refer to this initial residual map as our pixel-level distance D_p (which can be used for anomaly segmentation). Later on, we will introduce an additional feature distance D_f that focuses more on perceptual difference between two images.

Our project aims to address some main challenges outlined in Sec. 1.1 and Sec. 1.2. Our main contributions in this project are summarized as follows:

- Reconstruction quality: we improve the generative power of the diffusion model by enhancing the quality of the conditioning information injected into it. This helps guide the denoising process closer to the true distribution of the input. Secondly, by applying the appropriate noise level, we seek to preserve as much structural detail as possible while ensuring that anomaly regions are fully corrected.

- Pixel anomaly score: we introduce feature attention module to combine raw pixel distance with feature distance to further refine our residual map, and eliminate false positive caused by the model errors.

- We employ an automatic thresholding algorithm to remain fully consistent with the unsupervised learning framework.

- We incorporate the temporal structure of data to improve the reliability of the spatial diffusion model. Our temporal smoothing module is used to fuse and capture information across different time points.

- When considering the temporal dimension separately, we apply a temporal-aware diffusion model that captures the progression of healthy subjects and generates follow-up images from any given baseline image.

Chapter 2

Background

This chapter provides an overview of the mathematical background of diffusion models. We first discuss denoising probabilistic diffusion models in Sec. 2.1, which form the theoretical foundation for modern diffusion-based generation. Next, we introduce denoising diffusion implicit models (DDIMs) in Sec. 2.2, that uses a non-Markovian process to accelerate the sampling process of the original diffusion models. Finally, we cover the principle of guided diffusion models in Sec. 2.3, which uses conditions or contexts to steer the generation process, enabling more controlled and targeted outputs.

Contents

2.1 Denoising Probabilistic Diffusion Models

Denoising Probabilistic Diffusion Models (DDPMs) [11] are generative models aim to learn a probability distribution $p_\theta(\mathbf{x})$ that closely resembles the real data distribution $q(\mathbf{x})$. In diffusion-based generative models learn, we gradually add noise to our original clean image, each step will slightly corrupt the image until it becomes almost featureless distribution. Normally, we choose the noise so that at the end of the diffusion process, we end up with a normal distribution $\mathcal{N}(0, \mathbf{I})$, which is easy to sample from in our generative process. DDPMs is composed of two processes: *forward process* and *backward process* (generative process).

Forward process In the forward process, we gradually add noises to our input $\mathbf{x} \sim q(\mathbf{x})$ to generate a series of noisy images $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$ for many time steps T. The noise level t of an image $\mathbf{x}_t$ is predefined by a noise schedule $\beta_t \quad t \in [1, T]$, which is originally implemented as a linear schedule going from $\beta_0 = 10^{-4}$ to $\beta_T = 0.02$ in [11]. The forward process is a non-homogenous Markov chain process, with the following kernel transition:

$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \tag{2.1}$$

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t;\ \sqrt{1-\beta_t}\,\mathbf{x}_{t-1},\ \beta_t I\right) \quad \text{for } t = 1, \ldots, T \tag{2.2}$$

[11] defines $\alpha = 1 - \beta_t$ and $\bar{\alpha} = \prod_{s=1}^{t} \alpha_t$, and the forward process has a special property that

we can use to jump directly from $\mathbf{x}_0$ to any noisy $\mathbf{x}_t$ given a time step t [1]:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{2.3}$$

Using the reparameterization trick, we can express $\mathbf{x}_t$ as:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}}\mathbf{x}_0 + \sqrt{\bar{\alpha}_t}\epsilon \quad \text{with} \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}) \tag{2.4}$$

We see that with large enough T (normally $T = 1000$), $\bar{\alpha}_t \to 0$, and $\mathbf{x}_t$ converge to a Gaussian distribution $\mathcal{N}(0, \mathbf{I})$. With a learned reversed process, we can sample from this normal distribution to generate new image from the original distribution.

Generation process for image generation, we start from a random variable $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$ and the generation process is given by:

$$p(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \tag{2.5}$$

$$\text{with} \quad p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)) \tag{2.6}$$

The denoising network p_θ is learned by optimizing model parameters θ to predict $\mathbf{x}_{t-1}$ from $\mathbf{x}_t$, for each $t \in [1...T]$. This is equivalent to gradually remove noise at each time step, so we can generate noise-free image $\hat{\mathbf{x}}$ at $t = 0$. Instead of learning both μ_θ and Σ_θ, [11] show that by conditioning the true posterior on $\mathbf{x}_0$, we can fix $\Sigma_\theta(\mathbf{x}_t, t) = \Sigma(t)$, and the mean μ_θ can be derived using a noise prediction network ϵ_θ [12]:

$$\Sigma_\theta(\mathbf{x}_t, \mathbf{x}_0, t) = \Sigma(t) = \frac{1-\alpha_{t-1}}{1-\alpha_t}\beta_t\mathbf{I} \tag{2.7}$$

$$\mu_\theta(\mathbf{x}_t, \mathbf{x}_0, t) = \frac{1}{\sqrt{\alpha_t}}(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)) \tag{2.8}$$

Learning objective : $\epsilon_\theta(\mathbf{x}_t, t)$ in E.q (2.8) is the output of a time conditioned network, which [11] use UNet architecture to predict the noise added to the original image $\mathbf{x}_0$ at time t. The network is trained with loss function:

$$\mathcal{L}(\theta) = ||\epsilon - \epsilon_\theta(\mathbf{x}_t, t)||^2 = ||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}}\mathbf{x}_0 + \sqrt{\bar{\alpha}_t}\epsilon)||^2 \tag{2.9}$$
$$\text{where} \quad \mathbf{x}_0 \sim q(\mathbf{x}_0), \quad t \sim \text{Unif}[1, T], \quad \epsilon \sim \mathcal{N}(0, \mathbf{I})$$

2.2 Denoising Diffusion Implicit Models

DDIM sampling scheme : one disadvantage of DDPMs is during inference, we have to go through each step in reverse $t \in [T...0]$, and with large T comes with longer inference time. Denoising implicit models (DDIMs) [13] accelerate DDPMs by introducing a non-Markovian sampling process $q_\sigma(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$. DDIMs defines a new variance schedule σ that is chosen to have the same marginal distribution as in E.q (2.3). So the training procedure and objective stay the same as in DDPMs, but only the sampling process is different. Based on E.q (2.4), we can predict noise-free observation $\hat{\mathbf{x}}$ as:

$$\hat{\mathbf{x}}_0 = f_\theta(t, \mathbf{x}_t) = \frac{\mathbf{x}_t - \sqrt{1-\alpha_t} \cdot \epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{\alpha_t}} \tag{2.10}$$

[1] This is referred to as the Nice property of diffusion model. See math derivation from [12]

[13] generalize the DDPMs sampling scheme as:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}}\, f_\theta(t, \mathbf{x}_t) + \sqrt{1 - \alpha_{t-1} - \sigma_t^2}\, \epsilon_\theta(\mathbf{x}_t, t) + \sigma_t\, \epsilon \tag{2.11}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and σ_t controls the stochasticity of the sampling process. For DDPMs, we choose $\sigma_t = \sqrt{(1 - \bar{\alpha}_{t-1})/(1 - \bar{\alpha}_t)}\sqrt{1 - \bar{\alpha}_t/\bar{\alpha}_{t-1}}$, thus DDPMs will always have a stochastic element in its sampling step [2]. On the other hand, if we set $\sigma_t = 0$, we have an extreme case of deterministic DDIM sampling process, which means that $\mathbf{x}_{t-1}$ is fully determined given $\mathbf{x}_0$ and $\mathbf{x}_t$. Formally, the deterministic DDIM sampling scheme is defined as:

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\sqrt{\alpha_{t-1}}\, f_\theta(t, \mathbf{x}_t) + \sqrt{1 - \alpha_{t-1}}\, \epsilon_\theta(\mathbf{x}_t, t), 0) \tag{2.12}$$

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}}\, f_\theta(t, \mathbf{x}_t) + \sqrt{1 - \alpha_{t-1}}\, \epsilon_\theta(\mathbf{x}_t, t) \tag{2.13}$$

DDIM noise encoding : As derived in [13], E.q (2.13) can be viewed as the Euler method to solve an ordinary differential equation (ODE). Consequently, we can reverse the generation process by using the reversed ODE. With enough discretization steps, we can encode $\mathbf{x}_{t+1}$ given $\mathbf{x}_t$ with:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \sqrt{\bar{\alpha}_{t+1}} \left[\left(\sqrt{\frac{1}{\bar{\alpha}_t}} - \sqrt{\frac{1}{\bar{\alpha}_{t+1}}} \right) \mathbf{x}_t + \left(\sqrt{\frac{1}{\bar{\alpha}_{t+1}} - 1} - \sqrt{\frac{1}{\bar{\alpha}_t} - 1} \right) \epsilon_\theta(\mathbf{x}_t, t) \right] \tag{2.14}$$

By applying E.q (2.14) for $t \in \{1, ..., T-1\}$, we can encode a clean image into a noisy image $\mathbf{x}_T$. We refer to this $\mathbf{x}_T$ as *stochastic subcode* of $\mathbf{x}_0$, and [14] show that this stochastic encoding process stores fine-grain detail about $\mathbf{x}_0$. Starting from this stochastic subcode, we can then decode it using E.q (2.11) to have a near-exact reconstruction of $\mathbf{x}_0$ [14, 15, 6].

2.3 Guided diffusion models

Diffusion models described in Sec. 2.1 is designed to generate unconditional images, but many downstream tasks require generating images that adhere to a specific condition. Given an input image $\mathbf{x}$ and its corresponding class or context $\mathbf{y}$, we want to train a diffusion model that can generate images belong to a given class $\mathbf{y}$ [3]. In this section, we briefly discuss several techniques that are used to guide the diffusion process.

Gradient guided diffusion : Dhariwal and Nichol [16] incorporate the class guidance into the denoising process by training a separate classifier network C to distinguish between class labels. The network $C(\mathbf{y} \mid \mathbf{x}_t, t)$ is trained at every noise step t with corresponding noisy image $\mathbf{x}_t$ obtained by E.q (2.4). During DDIM sampling, the predicted noise in E.q (2.13) is modified as:

$$\hat{\epsilon}_\theta = \epsilon_\theta(\mathbf{x}_t, t) - s\sqrt{1 - \bar{\alpha}_t} \nabla_{\mathbf{x}_t} \log C(\mathbf{y} \mid \mathbf{x}_t, t) \tag{2.15}$$

where s is a parameter controls the strength of the guidance, and $\nabla_{\mathbf{x}_t} \log C(\mathbf{y} \mid \mathbf{x}_t, t)$ can be interpreted as log gradient of the likelihood of class $\mathbf{y}$ given a noisy image $\mathbf{x}_t$. From probabilistic model perspective, by taking a step in the opposite direction of this gradient, we can move toward region with higher density, and theoretically that will allow us to arrive at the mode of the true

[2] This is also why we have to perform DDPMs sample step by step instead of deriving $\mathbf{x}_0$ directly from $\mathbf{x}_T$ by reversing E.q (2.4). [12] show both the mathematical and intuitive reasons behind this

[3] I slightly abuse the term "class" here to align with common usage in the literature, even though in the UAD setting we typically do not have access to the true class label of the anomalous subject. Throughout the rest of this report, the terms "class" and "context" will be used interchangeably.

distribution $p(\mathbf{y} \mid \mathbf{x}_t)$. In the context of diffusion sampling process, this steers the generation of $\mathbf{x}_{t-1}$ toward the desire class $\mathbf{y}$. We note that this highlights the connection between diffusion models and score-based models [17], and we can reformulate ϵ_θ in E.q (2.11) to be a score-based function, which estimated the deviation should happen at each time step t to arrive at a clean image $\mathbf{x}_0 \sim q(\mathbf{x})$, as shown in [18, 19]:

$$\nabla_{\mathbf{x}_t} \log p_\theta(\mathbf{x}_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \tag{2.16}$$

Classifier-free guided diffusion : one major disadvantage of gradient guidance method is that we need to train an additional class condition network $C(\mathbf{y} \mid \mathbf{x}_t, t)$, and output quality depends on the performance of the classification network, introducing potential bias [2]. For the network to learn effectively, For the network to learn effectively, we need to train it for each class $\mathbf{y}$, at every noise level$t$, and for every $\mathbf{x}_0$. This approach is computationally expensive. There are some alternative methods that leverage this score-based function without needing to train a classifier network. One approach is to inject the context directly into the UNet to provide guidance for the network. Formally, given a noisy image $\mathbf{x}_t$ at time step t and a context $\mathbf{y}$, we want to update our noise prediction output from E.q (2.8) become $\epsilon_\theta(\mathbf{x}_t, t, \mathbf{y})$ to incorporate this information. Our loss function in E.q (2.9) becomes:

$$\mathcal{L}(\theta) = ||\epsilon - \epsilon_\theta(\mathbf{x}_t, t, \mathbf{y})||^2 \tag{2.17}$$

There are several methods for implementing conditioning mechanisms. Some earlier approaches use the input $\mathbf{x}$ directly as context and concatenate this information with the noisy image $\mathbf{x}_t$. In this implementation, the context is *spatial context*, i.e. $\mathbf{y} \in \mathbb{R}^{h \times w}$. However, this strategy has the potential drawback of revealing the target to the network, since subtracting $\mathbf{x}$ from $\mathbf{x}_t$ yields the exact noise that was added. To address this, more recent approaches use *non-spatial context* $\mathbf{y} \in \mathbb{R}^d$, which is generated by a semantic encoder network. LDM [20] employs a cross-attention module to inject context at every step of the UNet, while cDDPM [7] concatenates the context with the time embedding.

Chapter 3

Proposed Frameworks for Spatial-Temporal UAD

In this section, we introduce our proposed framework for Spatial-Temporal unsupervised anomaly detection, comprises two main modules: Spatial Diffusion Model Spatial Diffusion Model (SDM) (Sec. 3.1) and Anomaly Detection Module (Sec. 3.2). During the training of SDM, we remove all temporal dependencies between samples and treat them as independent. SDM accounts only for the spatial representations of the input. At inference, we add back the temporal dependencies using longitudinal module to increase the accuracy of anomaly segmentation.

Contents

3.1 Spatial Diffusion Model for pseudo-healthy data generation

We now introduce the framework of our proposed SDM, with anomaly detection process during inference. Our SDM comprises four keys blocks: a semantic encoder (Sec. 3.1.1), a denoising UNet (Sec. 3.1.2), a FAM (Sec. 3.2.1) and a LAFM (Sec. 3.2.2). These four components, summarized in Fig. 3.1, collectively can address the challenges we outlined in the introduction. We structure the description of SDM into two parts: this section provides a detailed explanation of diffusion model for image reconstruction (training phase). Sec. 3.2 will discuss anomaly detection during inference phase. In training process, our SDM is designed as a conditional diffusion model that treats each $\mathbf{x}_l^i \quad (l = 1, ..., L_i)$ as independent cross-sectional data. To simplify the notation, we omit the superscript i and subscript l from here onward. During inference, we introduce FAM and LAFM after the denoising process to achieve final anomaly prediction and anomaly segmentation.

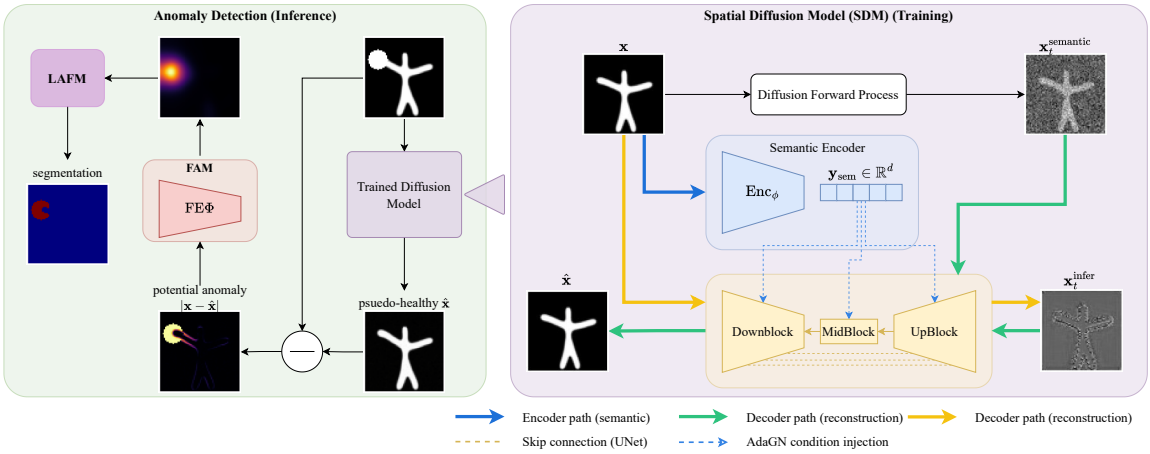


Figure 3.1: Overview of our SDM framework for unsupervised anomaly detection using reconstruction-based method. Our model consists of four keys components: (i) **semantic encoder** extracts a non-spatial representation of input ($\mathbf{x} \rightarrow \mathbf{y}_{\text{sem}}$), (ii) **denoising UNet** comprises a down block, a middle block and up block with skip connection to concatenate information from down blocks to corresponding up blocks, (iii) **Feature Attention Module (FAM)**, and (iv) **Longitudinal Attention Fusion Module (LAFM)**. The semantic representation is injected as conditions into our diffusion process. Conditional diffusion UNet acts as both a stochastic encoder ($\mathbf{x} \rightarrow \mathbf{x}_T^{\text{infer}}$) and a decoder (DDIM denoising step) ($\mathbf{x}_T \rightarrow \widehat{\mathbf{x}}$). During inference, we use trained SDM to reconstruct pseudo-healthy image of input, and potential anomaly map is calculated as residual $|\mathbf{x} - \hat{\mathbf{x}}|$. We use Feature Attention Module (FAM) to clean residual map, and Longitudinal Attention Fusion Module (LAFM) module further fuses information from other time points to have final anomaly segmentation.

3.1.1 Semantic Encoder

The goal of the semantic encoder $\text{Enc}_\phi(\mathbf{x})$ is to summarize an input image into a meaningful non-spatial representation $\mathbf{y}_{\text{sem}} \in \mathbb{R}^d$ with enough information to help the decoder $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{\text{sem}})$ denoise and predict the output image. We employ ResNet-50 [21] as our semantic encoder. ResNet-50 has 50 layers, and is built around residual connections that enable stable training of very deep models. It comprises four main residual blocks operating at progressively lower spatial resolutions that capture hierarchical feature representations of the input. We replace the last classifier layer with an MLP that produces a non-spatial representation of the input. Together with time step t, $\mathbf{y}_{\text{sem}}$ form the condition that we will inject into diffusion UNet.

3.1.2 Diffusion UNet

The core component of our SDM is denoising UNet module ϵ_θ with trainable parameter θ, adapted from [16]. Given a clean image $\mathbf{x}$ and a time step t, we add noise to the image using E.q (2.4). The time step t controls the amount of added noise and is sampled from $t \sim \text{Uniform}(1 \dots T)$. When $t = T$, the image $\mathbf{x}$ is transformed into pure noise $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$. Subsequently, $\mathbf{x}_t$ is passed through the UNet, which predicts the added noise. We condition the backward process on context vector comprises time step and semantic representation $c = (t, \mathbf{y}_{\text{sem}})$. Time step is embedded using sinusoidal positional embeddings, originated from [22], while semantic representation is the output of our semantic encoder. To provide context c into UNet, we use FiLM-style adaptive group norm (AdaGN), following [16]. We formulate our AdaGN operation as:

$$\text{AdaGN}(\mathbf{h}, t, \mathbf{y}_{\text{sem}}) = \mathbf{y}^s_{\text{sem}}(\mathbf{t}_s.\text{GroupNorm}(\mathbf{h}) + \mathbf{t}_b) + \mathbf{y}^b_{\text{sem}} \tag{3.1}$$

where $\mathbf{h}$ denotes the hidden feature map at each level of the UNet. The pairs $[\mathbf{y}^s_{\text{sem}}, \mathbf{y}^b_{\text{sem}}]$ and $[\mathbf{t}_s, \mathbf{t}_b]$ are obtained via linear projections of $\mathbf{y}_{\text{sem}}$ and t, respectively. They represent the *shift* and *scale* parameters applied to the feature maps, effectively adjusting them according to the conditioning signals. This design enables the model to guide the denoising process using high-level semantic codes. Compared to using a cross-attention module as in [20], this approach is simpler, and unlike concatenating $\mathbf{y}$ with t as in [7], it does not increase the hidden dimensionality of $\mathbf{h}$.

The semantic encoder is jointly trained with UNet with updated loss function:

$$\mathcal{L}(\theta, \phi) = ||\epsilon - \epsilon_\theta(\mathbf{x}_t, t, \text{Enc}_\phi(\mathbf{x})||^2 \tag{3.2}$$

We can interpret this training algorithm as forcing the semantic encoder Encϕ to capture as much information about $\mathbf{x}$ as possible. Theoretically, if $\mathbf{y}_{\text{sem}} = \text{Enc}\phi(\mathbf{x}) \in \mathbb{R}^d$ retains all the information of the input, we can achieve exact reconstruction [15].

During inference, our trained model can generate a new sample by denoising process using E.q (2.6), starting from pure noise $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$. We elevate the sampling process using DDIM sampling scheme, using E.q (2.13). We set $\sigma = 0$ to have a deterministic process. It is worth noting that although the training process is performed with $T = 1000$ steps, it is not necessary to reconstruct healthy images from pure noise. In fact, since our main objective is not to generate entirely new images but rather to heal anomalous regions, it is preferable to start decoding from a noise level $t < T$. Using this approach, we preserve some high-level structure of the original image, in contrast to starting from pure noise. The noisy image $\mathbf{x}_t$ is obtained by adding noise to $\mathbf{x}_0$ with E.q (2.4).

3.1.3 Stochastic Encoder and Conditional Decoder

Here, we provide further insights into our encode and decode processes. We refer to our semantic encoder in Sec. 3.1.1 as *semantic encoder*, which map input into its nonspatial representation.

We will refer to $\mathbf{y}_{\text{sem}}$ as our *semantic subcode*. In addition, our denoising network ϵ_θ plays a dual role in our model. Its main purpose is to act as a *conditional decoder* that can generate healthy images from noisy input. On the other hand, it can also serve as a *stochastic encoder* by using deterministic DDIM sampling scheme. Given a clean image $\mathbf{x}$, SDM can encode it to a stochastic subcode using E.q (2.14). [14, 15] demonstrates that this stochastic subcode contains fine grain details about $\mathbf{x}$ that is not captured by semantic encoder. We note that, unlike the standard noising process E.q (2.4), the stochastic subcode does not necessarily follow Gaussian noise, since it retains residual information about the input. We refer to *stochastic subcode* of original image $\mathbf{x}$ as $\mathbf{x}_t^{\text{infer}}$, and normal noisy image as $\mathbf{x}_t^{\text{semantic}}$ (or $\mathbf{x}_t$) [1], with $t \leq T$.

3.2 Unsupervised Anomaly Detection Module

In the simplest case, reconstruction-based method relies on pixel wise difference between the original image $\mathbf{x}$ and psuedo-healthy reconstructed image $\hat{\mathbf{x}}$ to generate the anomaly map, which will be used as our anomaly segmentation. To address the challenges mentioned in Sec. 1.1, we aim to employ techniques that: (i) ignore small outliers in the background, (ii) discard minor reconstruction errors around object edges, and (iii) emphasize differences within the critical anomaly region (i.e., the region of interest). [23] introduce the effectiveness of using image features as perceptual metrics, which are robust to minor pixel-level differences that do not alter the overall image structure. Features are sensitive to changes in texture and shape, but can be robust against slight transformation where pixel level distance might fail. We follow method of DDAD [18] that defines anomaly score as combination of both pixel and feature errors. By utilizing both pixel differences and features differences, we can achieve high precision in anomaly localization. We refer to this process that wraps around the feature extractor network as our FAM. We discuss the methodology of FAM in the section below.

3.2.1 Feature Attention Module

Anomaly score as combination of feature distances and pixel distances Given an input image $\mathbf{x}$, and it's psuedo-healthy reconstruction $\hat{\mathbf{x}}$, we define a pixel-wise distance function D_p and a feature-wise distance function D_f to derive the anomaly heatmap. D_p is calculated based on the $\mathcal{L}_1$ norm in pixel space. At the feature level, we use adaptive average pooling to spatially smooth each individual feature map. Feature distance D_f is then calculated as a cosine-similarity. Formally we have:

$$D_p(\mathbf{x}, \hat{\mathbf{x}}) = \mathcal{L}_1(\mathbf{x} - \hat{\mathbf{x}}) \tag{3.3}$$

$$D_f(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{j \in J} \text{Interp}\left(1 - \cos\left(\Phi_j(\mathbf{x}), \Phi_j(\hat{\mathbf{x}})\right)\right) \tag{3.4}$$

where Φ is a pre-trained feature extractor network and j refers to the level (or stage) that we want to extract feature from, and Interp is the interpolation operator of feature maps to match input's resolution. We note that for each feature level j, the cosine similarity is calculated along channel dimension and for each pixel. Formally, we have $\mathbf{x}, \hat{\mathbf{x}} \in \mathbb{R}^{h \times w}$, $\Phi_j(\mathbf{x}), \Phi_j(\hat{\mathbf{x}})$, and $\cos(.,.) \in \mathbb{R}^{h_j \times w_j}$, with h_j, w_j are dimensions of feature map j.

We utilize our trained semantic encoder Enc_ϕ, and we train a copy of it as feature adaptation step. We give detail about this step in section 3.2.1. We follow the convention with existing

[1] We note that in our report, $\mathbf{x}_t^{\text{semantic}}$ and $\mathbf{x}_t$ are used interchangeably. $\mathbf{x}_t^{\text{semantic}}$ is used in the context of comparison with $\mathbf{x}_t^{\text{infer}}$ to emphasize the differences, whereas $\mathbf{x}_t$ is used elsewhere for the sake of brevity.

literatures for ResNet-like architecture, so that $j \in \{1, 2, 3, 4\}$ indicating the final output of respective spatial resolution blocks. The final anomaly score (pixel wise) will be given as [18]:

$$\mathbf{s} = D_{\text{anomaly}}(\mathbf{x}, \hat{\mathbf{x}}) = \left(v \cdot D_p + D_f \frac{\max(D_p)}{\max(D_f)} \right) \tag{3.5}$$

where v is the parameter to control the importance of the errors at pixel level, $D_{\text{anomaly}}(.,.) \in \mathbb{R}^{h \times w}$ is spatial representation that retains the same height and width as the input, allowing for pixel-wise localization of anomalies. We use notation $\mathbf{s}$ to denote pixel-wise anomaly score map.

Feature Extractor Network : is a trainable model FE with parameter Φ. Following DDAD method [18], FE is initialized from our semantic encoder Enc_ϕ, which is already jointly trained with conditioned diffusion UNet. Although it is trained specifically on our dataset to capture high semantic structure of input, it serves a slightly different purpose compared to Φ. Enc_ϕ is used to compress and capture as much information as possible about $\mathbf{x}$ to use as guidance in our diffusion process, while Φ needs to be trained to ignore discrepancies between $\mathbf{x}$ and $\hat{\mathbf{x}}$ that are not semantically meaningful. We train an extra step by converging different extracted layers from two nearly identical images.

Given an input $\mathbf{x}$ and a trained denoising model ϵ_θ, we go through a complete inference process of $\mathbf{x} \rightarrow \mathbf{x}_T \rightarrow \hat{\mathbf{x}}$ to create its reconstruction image. We then extract their features from Φ, denoted as $\Phi_j(\mathbf{x})$ and $\Phi_j(\hat{\mathbf{x}})$ respectively. Since we train the network on healthy training dataset only, we can make assumption that $\hat{\mathbf{x}} \sim \mathbf{x}$, thus their features should be similar. Therefore, our network Φ is fine-tuned by minimizing the distance between extracted features. We use cosine-similarity as our loss function, which is calculated for each of the final activation layers of the jth spatial resolution block. Similarly to [18], we incorporate a distillation loss from a frozen feature extractor $\bar{\phi} = \text{Enc}_\phi$ as a safeguard mechanism to prevent the network from losing its generalization it has been trained before. Formally, our loss function can be expressed as:

$$\begin{aligned} \mathcal{L}_{\text{sim}} = & \sum_{j \in J} \left(1 - \cos\left(\Phi_j(\hat{\mathbf{x}}), \Phi_j(\mathbf{x})\right)\right) \\ & + \lambda_{DL} \sum_{j \in J} \left(1 - \cos\left(\Phi_j(\hat{\mathbf{x}}), \bar{\phi}_j(\hat{\mathbf{x}})\right)\right) \\ & + \lambda_{DL} \sum_{j \in J} \left(1 - \cos\left(\Phi_j(\mathbf{x}), \bar{\phi}_j(\mathbf{x})\right)\right) \end{aligned} \tag{3.6}$$

where λ_{DL} controls the strength of our distillation losses. This network Φ will help to transfer the trained semantic encoder Enc_ϕ to feature-distance network Φ. Similar to [18, 24] we set $j \in J = \{2, 3\}$, we only consider middle-sized blocks in our ResNet to retain the generality of the used features.

3.2.2 Longitudinal Attention Fusion Module

Up to now, we have considered each sample as an i.i.d. draw from $p(\mathbf{x})$. After combining feature distances and pixel distances from Sec. 3.2.1, each image $\mathbf{x}_0$ is associated with an anomaly score map $\mathbf{s}^{pixel} \in \mathbb{R}^{h \times w}$ that ideally contains sufficient information to determine which pixels are anomalous. Unfortunately, this is often not the case: if we detect anomalies independently at each time point, the anomaly score map can be noisy and exhibit: (i) false positives that appear at one time and then vanish, and (ii) missing of real anomalies (false negatives), especially when the anomalies are subtle. We assume that, in reality, anomalies usually have temporal persistence:

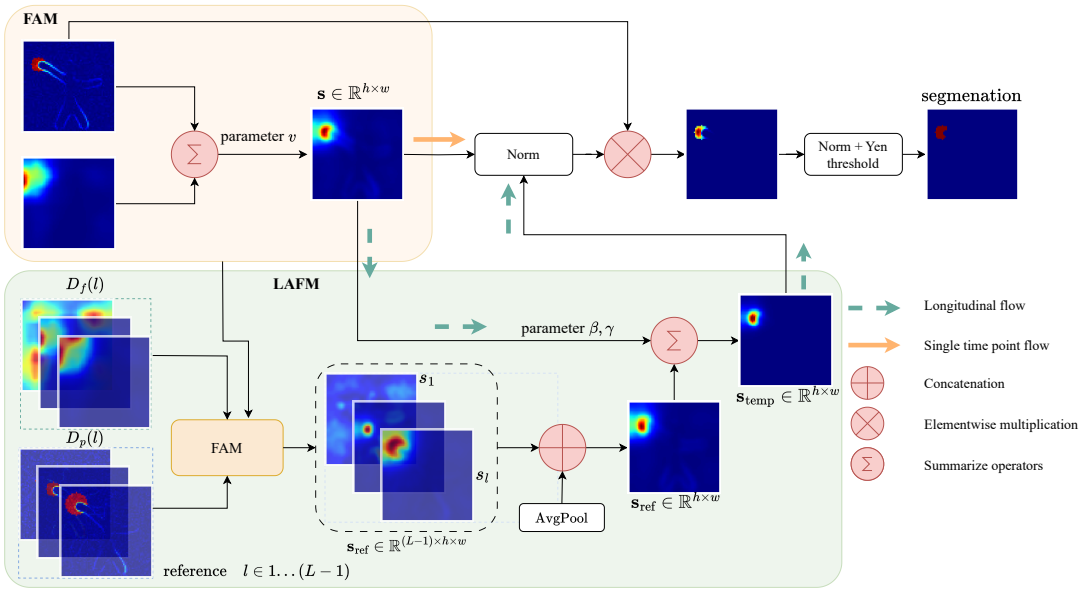


Figure 3.2: The overall structure of Feature Attention Module and Longitudinal Attention Fusion Module. For single time point (orange arrow), anomaly segmentation is calculated directly from the output of FAM. With longitudinal data (green arrow), each time point anomaly score map is processed by FAM, concatenated together and applied the F_{mean} operator with Gaussian weights (E.q (3.7)). The referenced anomaly score map $\mathbf{x}_{\text{ref}}$ is used to update the targeted anomaly score to have $\mathbf{s}_{\text{temp}}$ (E.q (3.8))

.

they do not appear and disappear randomly[2]. Similar to [25], we design our LAFM that acts as a temporal filter, enforcing the persistence of anomalies and smoothing out noise. We note that, in principle, our Longitudinal Attention Fusion (LAF) module is similar to the MAF module in [25], but instead of attending to information from multiple modalities at the same time point, here we concatenate information from a single modality across different time points. Fig. 3.2 shows the overall structure of LAFM.

We consider the full set of images collected for each patient,

$$\mathbf{X}_i = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L\}, \quad \mathbf{x}_l \in \mathbb{R}^{h \times w}, \quad l \in [1, \ldots, L],$$

where $\mathbf{x}_l$ represents an image at a single time point l. Different patients may have a different total number of visits and images acquired at different time points. All images in the set $\mathbf{X}_i$ are processed through our SDM model and FAM as described above, yielding a corresponding set of anomaly score maps,

$$\mathbf{S}_i = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_L\}.$$

We denote $\mathbf{x}$ as the target image and $\mathbf{s}$ as its corresponding anomaly score map at target time point l_{target}. We further define the reference set

$$\mathbf{s}_{\text{ref}} = \mathbf{S}_i \setminus \{\mathbf{s}\}$$

as the set of anomaly score maps from other time points, excluding the target. Note that the length of $\mathbf{s}_{\text{ref}}$ is $L-1$, and $\mathbf{s}_{\text{ref}} \in \mathbb{R}^{(L-1) \times h \times w}$.

[2]This might not be the case for patients undergoing treatment, but this scenario requires a different type of model and is beyond the scope of our experiment.

Similar to MAF from [25], the role of LAFM is to perform the weighted fusion of complementary information from reference time points, and ultimately to obtain the accurate anomaly masks. To fuse key information across time points and align it into a single output, we apply pixel-wise average pooling with a receptive field of 1. This gives us our mean response map F_{mean} To account for the relative importance of each reference time point with respect to the target time point, we weight the reference anomaly score maps using a Gaussian kernel. Formally, we have:

$$\begin{aligned} \mathbf{s}_{\text{ref}} &= \oplus\{\mathbf{s}_l\}, \quad l \in [1, \ldots, L],\ l \neq l_{\text{target}} \\ \mathbf{w} &= \exp\left(-\frac{(\mathbf{l} - l_{\text{target}}\mathbf{I})^2}{2\sigma^2}\right) \\ F_{\text{mean}} &= \text{AvgPool}(\mathbf{s}_{\text{ref}} \odot \mathbf{w}) \in \mathbb{R}^{h \times w} \end{aligned} \tag{3.7}$$

where

- $\mathbf{l} = [\, l_1, l_2, \ldots, l_L \,]^\top$ is the vector of reference time points.
- l_{target} is the target time point.
- $\mathbf{w} = [\, w_1, w_2, \ldots, w_L \,]^\top$ is the vector of Gaussian weights.
- σ controls the temporal decay rate.
- $\oplus$ denotes concatenation and $\odot$ denotes element wise multiplication.

Finally, we combine $\mathbf{s}_{\text{ref}}$ with target anomaly score $\mathbf{s}$ using:

$$\mathbf{s}_{\text{temp}} = \underbrace{\mathbf{s} \times (1 - \beta \times (1 - \text{Norm}(\mathbf{s}_{\text{ref}})))}_{\text{eliminate false positive}} + \underbrace{\gamma \times \mathbf{s}_{\text{ref}}}_{\text{eliminate false negative}} \tag{3.8}$$

We refer to this as *temporal smoothing operation* of $\mathbf{s}$, and it can be interpreted as follows: first, we normalize $\mathbf{s}_{\text{ref}}$ using min-max normalization. This transforms raw anomaly score into the scaled anomaly probability at each pixel. The first expression of RHS of E.q (3.8) is to eliminate false negative in target image. For example, if pixel $p_{i,j}$ is flagged as anomalous in $\mathbf{s}$, we check to see if it is consistent over time period. If this is a false positive, meaning this is pixel error cause by reconstruction process, it will not persist over time and thus its reference value across time will be small, i.e. $p_{i,j}^{\text{ref,norm}} \approx 0$. We negate its value and subtract it from $\mathbf{s}$, efficiently reducing its anomaly score. We note that here we are working with normalized scaled anomaly score $(1 - \text{Norm}(\mathbf{s}_{\text{ref}})) \in [0, 1]$, so the target score will be scaled down according to its reference. Conversely, if the error persists over time, its reference value will be higher $p_{i,j}^{\text{ref,norm}} \approx 1$, and in this case we keep it unchanged. $\beta \in [0, 1]$ is hyperparameter controls how much the reference information influences the final anomaly score. In our setting, we choose $\beta = 1$ to strictly limit false positive, because in our experiments we observed that our model performs well in detecting anomalies, but tends to produce false positives on small pixel errors in healthy data. The second expression on the RHS of E.q (3.8) helps to decrease false negative. In particular, if a pixel $p_{i,j}^{\text{ref,norm}}$ is consistently flagged as anomalous across reference time points, but our model is false to detect in the current anomaly score map $\mathbf{s}$, we want to increase its value $p_{i,j}$. Hyperparameter γ controls how much information is incorporated from $\mathbf{s}_{\text{ref}}$ to $\mathbf{s}$. As note earlier, false negatives are not a major issue in our model, so we set a relatively small $\gamma = 0.1$ as a safeguard mechanism.

3.2.3 Post Processing

Pixel anomaly score : We adapt some common post-processing technique used in UAD domain to further clean the residual map $\mathbf{s}_l$ before binarizing it to obtain anomaly segmentation [7, 26]. First, we apply a median filter with kernel size $k_{mf} = 5 \times 5$ to smooth out the anomaly score map, and to reduce the influence of noise to obtain a more reliable distribution of the anomalies. Next, we apply min–max normalization on a per-patient basis. This differs from other methods (cDDPM [7], EPDiff [25]) where the authors apply normalization at each image/volume individually.

Pixel anomaly segmentation : To obtain the localization of anomalies, a binary map is obtained by thresholding the anomaly score map. In previous methods [8, 7, 18], the optimal threshold for segmentation was found using a greedy search strategy on the validation dataset, by calculating the AUPRC metric and find the best theoretical value. However, ground truth annotations need to be provided in order to perform these search, and we argue that using labeled data will compromise the unsupervised setup (Sec. 1.1). We follow [25] to employ Yen threshold technique [27], which is an automatic image thresholding strategy that can separate background and foreground based on histogram of gray levels in the image. We can formulate our whole post-processing for pixel anomaly score as follows:

$$
\begin{aligned}
\mathbf{S}_{\text{temp}} &= [\mathbf{s}_{\text{temp},1}, \mathbf{s}_{\text{temp},2}, \dots, \mathbf{s}_{\text{temp},L}] \\
\mathbf{S}_{\text{temp}} &= \text{Norm}(\text{MedianFilter}(k_{mf})(\mathbf{S}_{\text{temp}})) \\
\text{thres} &= \text{YenThreshold}(\mathbf{S}) \\
\text{m}_{i,j} &= \begin{cases} 0 & \text{if } s_{i,j} < \text{thres} \\ 1 & \text{if } s_{i,j} \geq \text{thres} \end{cases} \\
\mathbf{D}_p &= \text{m} \odot \mathbf{D}_p \\
\text{p\_thres} &= \text{YenThreshold}(\mathbf{D}_p) \\
\text{p\_m}_{i,j} &= \begin{cases} 0 & \text{if } d_{i,j} < \text{p\_thres} \\ 1 & \text{otherwise} \end{cases} \\
\text{ano\_m} &= \text{m} \cap \text{p\_m}
\end{aligned}
\tag{3.9}
$$

where $\mathbf{S} \in \mathbb{R}^{L \times h \times w}$ is batch of all anomaly score maps (updated from LAFM) for 1 patient i, and $\mathbf{D}_p \in \mathbb{R}^{L \times h \times w}$ is batch of all pixel distance maps. Here we drop the subscript i for brevity. m and p_m are anomaly score segmentation maps and pixel distance segmentation maps, respectively. The final anomaly segmentation maps are defined as intersection between the twos.

Image anomaly score : After post-processing step, we obtain the pixel-level anomaly score $\mathbf{s} \in \mathbb{R}^{h \times w}$ for each individual image. For image-level anomaly score, a common practice in the literature is taking the mean over all pixel values in the anomaly map. [18, 28] use the maximum value across pixels and assign it as the overall anomaly score of the image, with the argument that it should be independent of the anomaly size. [29] combine both technique and design the image-wise anomaly score as the average of the largest S pixel wise anomaly scores in $\mathbf{s}$ to mitigate false positives caused by image noise. We employ several similar techniques, which will be discussed in Sec. 5.2.3.

Chapter 4

Experiment

Contents

4.1 Dataset

We use the `Starmen` dataset as a toy example to test and demonstrate the performance of our model. The `Starmen` dataset is a synthetic longitudinal dataset[1] consisting of $N = 1,000$ subject, each with 10 visits at different time points. This dataset is specifically designed for studying longitudinal models, capturing both spatial and temporal variability. The temporal variability of the population is prescribed by a "starman" raising its left arm, generated according to a diffeomorphism model described in [30]. On the other hand, the spatial variability represents individual heterogeneity are characterized by the location of other four control points: the head, right arm and legs. This way, the effects of time progression, raising the left arm, are (spatially) independent of the inter-variability of the shapes. All subjects raise the left arm but vary in shape with different position of their legs and arms.

All images are gray scaled with 1 channel ($1 \times 64 \times 64$), and pixel values are normalized in the range $[0, 1]$. We use 700 patients for our training set, 150 for validation set and 150 for test set. We note that 1 patient has 10 visits, thus when we consider individual images we have $7000, 1500$ and 1500 samples for train, validation and test set, respectively. We consider all images in these sets are healthy. Since this dataset does not have anomaly structure, we use `cv2` package to generate 3 types of synthetic anomaly to be used in our anomaly detection task: `growing_circle`, `darker_line` and `darker_circle`. For each type of anomaly, we create an anomaly set of 20 patients, corresponding to 200 anomaly images. Fig. 4.1 shows 1 example of healthy patient and 3 synthetic anomaly types.

[1]The dataset is available at https://zenodo.org/records/5081988

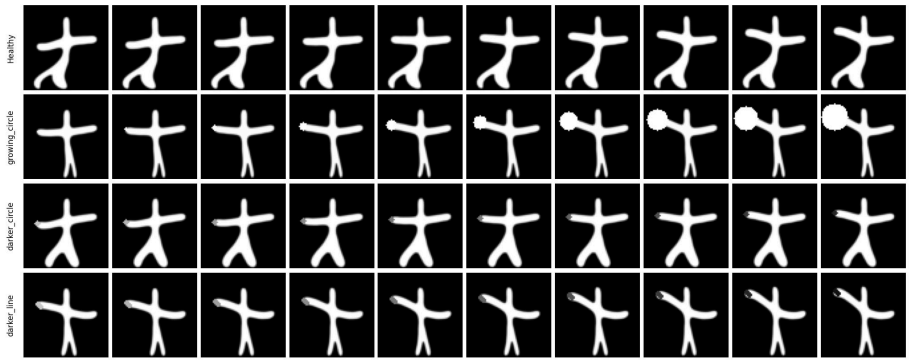


Figure 4.1: Examples of Starmen dataset images. The first row shows a healthy image. The last 3 rows show different anomaly type, from top to bottom: anomaly of type `growing_circle`, anomaly of type `darker_circle` and `darker_line`. Each sample contains 10 sequential images at different time points, through time the mini starmen slowly raises its left hand, while maintaining all other parts of its body (head, legs and right hand). Anomalies are added using `cv2` and places at the left most points of the figure.

4.2 Implementation details

4.2.1 Spatial Diffusion Model

UNet : The baseline diffusion model is based on the DDIM model [13][2] with adaptation from LDAE [15] [3]. The UNet comprises an input block, a middle block and an output block. The input block follows an encoder architecture that encodes and downsamples the original input. The output blocks are symmetric counterparts of the encoder, upsampling the encoded representation back to the original resolution. Each block contains one residual block (ResBlock), with channel multipliers of $[32, 64, 96]$. Each ResBlock begins with a GroupNorm layer and a SiLU activation, followed by a Conv2d layer with kernel size $(3, 3)$ and stride $(1, 1)$. Downsampling is performed with an AvgPool layer of kernel size $(2, 2)$, stride $(2, 2)$, and no padding. Attention layers are added at resolution scales of $[\frac{1}{2}, \frac{1}{4}]$ to capture dependencies across neighboring regions of the image and enhance spatial information exchange between feature maps.

Condition signals : time step condition t is embedded using sinusoidal time step embeddings, originated from [22]. The embedded t is then projected through a linear layer followed by a SiLU activation to obtain a vector of dimension $d_{cond} = 128$. We use ResNet50 as the backbone for the semantic encoder Encϕ. The parameters ϕ are initialized from the pretrained ResNet module in PyTorch. The final classifier layer of ResNet is replaced with a fully connected layer to produce a non-spatial representation of the input, $\mathbf{y}_{sem} = \text{Enc}_\phi(\mathbf{x})$, with latent dimension $d = 512$. This representation is further projected via linear layers to match the dimension of the time step embedding. Finally, the conditioning variables $(\mathbf{y}_{sem}, t)$ are injected into the network using AdaGN.

Training configuration : SDM is trained with a total number of diffusion time steps $T = 1000$ and a linear noise variance from $\beta_1 = 0.0001$ to $\beta_T = 0.02$. We use the Adam optimizer with a learning rate of 2×10^{-5}. Training is performed for 500 epochs with a training batch size of 2 patients, corresponding to an effective batch size of 20 samples. Following common practice in

[2]Available at https://github.com/openai/guideddiffusion
[3]Available at https://github.com/GabrieleLozupone/LDAE

diffusion models [15, 20], we use an exponential moving average (EMA) to stabilize training and improve generalization. EMA decay rate is 0.999, and EMA parameters are updated after every 10 batches. A full list of architectural and training parameters is provided in Tab. 4.1.

Table 4.1: Spatial Diffusion Model (SDM): configurations and parameters

Semantic Encoder Enc_ϕ	
Backbone	`ResNet50`[21]
Pretrained Init	ImageNet
Input Modality	$1 \times 64 \times 64$
Output layer	`Linear(in=1000, out=512)`
Output Representation	non-spatial vector $\mathbf{y}_{sem} \in \mathbb{R}^{512}$
Diffusion UNet ϵ_θ	
Input Shape	$(B, 1, 64, 64)$, with batch size first
Channels multipliers	[32, 64, 96]
Residual Blocks per Level	1
Attention Resolutions (factors)	[2, 4]
Number of attention head	1
Conditional injection	AdaGN (scale-shift norm)
Dropout	0.1
Time embedded dimension	$d_{cond} = 128$
Semantic encoder dimension	$d_{sem} = 512$
Timestep	1000
Beta Schedule	Linear, $\beta_t \in [10^{-4}, 2 \times 10^{-2}]$
Training Configuration	
Optimizer	Adam
Learning Rate	2.5×10^{-5}
EMA Decay	0.999
Train Batch Size (effective)	20
Training Duration	500 epochs / $\sim$3 hours
Hardware	2 x Nvidia L40S (45 GiB)

4.2.2 Feature Extractor network of FAM

Our FE network Φ is initialized from trained semantic encoder Enc_ϕ. To learn the similarity between samples, each training batch $\mathbf{x} \in \mathbb{R}^{B\times1\times64\times64}$ is passed through the SDM model to obtain its reconstruction $\widehat{\mathbf{x}} \in \mathbb{R}^{B\times1\times64\times64}$. We employ the DDIM sampling scheme in the denoising process with 100 steps to accelerate training. To improve generalization, we reconstruct $\mathbf{x}$ from different noise levels $[100, 250, 500, 1000]$, selected uniformly at each epoch. Cosine similarity loss is calculated on three layers of ResNet50, each at a different resolution, and the losses are summed to obtain the total loss. Distillation loss is incorporated from frozen Enc_ϕ network, with parameter $\lambda_{DL} = 1$, using E.q (3.6). FE is trained with 100 epochs, using Adam optimizer with learning rate 10^{-4}. Tab. 4.2 shows the details configurations of our FE network.

Table 4.2: Feature Extractor network (FE): configurations and parameters

FE Φ	
Backbone	`ResNet50`[21]
Pretrained Init	Semantic Encoder Enc_ϕ
Architecture	Same as Enc_ϕ
Input Shape	$(B, 1, 64, 64)$, with leading dimension is batch size
Feature layers	[`layer1, layer2, layer3`]
Feature size - `layer1`	[256, 64, 64]
Feature size - `layer2`	[512, 8, 8]
Feature size - `layer3`	[1024, 4, 4]
DDIM samle step	100
Noise level	[100, 250, 500, 1000]
λ_{DL}	0.1
Optimizer	Adam
Learning Rate	1.0×10^{-4}
Train Batch Size (effective)	20
Training Duration	100 epochs / $\sim$4 hours
Hardware	1 x Nvidia L40S (45 GiB)

4.3 Evaluation metrics

The performance of our model is evaluated based on two main criteria: reconstruction quality and anomaly detection score. In this section, we outline the details of our evaluation metrics for both tasks.

4.3.1 Reconstruction quality

For both SDM and TDM, the goal of diffusion model is to reconstruct the ground truth image: it can be the original healthy image (in the case of SDM), or the ground truth missing image (in the case of TDM). To evaluate the overall reconstruction quality, we follow conventional methods in literature [20, 15, 7], and calculate both pixel error metrics and similarity metrics between inputs and reconstructed images. For pixel error, we report both the $l1$ and $l2$ errors. For similarity, we consider the Structural Similarity Index Measure (SSIM), the Peak Signal To Noise Ratio (PSNR) and the Learned Perceptual Image Patch Similarity (LPIPS) as metrics to assess the reconstruction quality. To account for perceptual structures at both coarse scales and fine details scales, we also calculate the Multi Scale Structural Similarity (MSSIM), which is more aligned with human visual perception. For the feature based LPIPs metric, features are extracted by a `squeeze` network. All similarity metrics are implemented in `Monai` package [4].

Furthermore, for anomaly detection task, only healthy anatomy should be reconstructed. Similar to [7], we consider the $l1$-error of healthy and unhealthy anatomy separately, given the synthetic anomaly data sets. Specifically, we calculate the $l1$-error for both healthy and unhealthy anatomy, as indicated by the annotation masks and also calculate an $l1$-ratio as follows:

$$l1\text{-ratio} = \frac{l1_{anomaly}}{l1_{healthy}} \tag{4.1}$$

[4]MONAI is an open-source framework for deep learning in healthcare imaging, available at https://docs.monai.io/en/stable/index.html

For healthy region, we want to have as small error as possible, while for anomaly region it is better to have higher errors, which will help to separate anomaly part from healthy structure. A higher value for $l1$-ratio indicates that the model successfully remove the anomaly region while maintaining correct structure of healthy anatomy, and vice versa.

4.3.2 Anomaly scores

To evaluate the performance of our models in anomaly detection, we use the ground-truth segmentation from our synthetic anomaly dataset. It is important to note that ground-truth labels/annotations are often unavailable in real-life MRI datasets, especially in unsupervised settings. Nevertheless, most common models still rely on datasets with ground-truth labels to assess their performance, as reported in Bercea's study [2].

Image-level anomaly detection: recall from Sec. 3.2.3 that from a spatial anomaly score map, we use several functions to summarize all pixel-level score into 1 single value that we can assign for image level anomaly score. The AUROC and AUPRC are popular metrics [31, 7, 18, 25] due to their threshold independence, so we can assess the model's performance directly from raw anomaly score without having to find a threshold value. However, AUROC is reported to have potential misleading in imbalanced datasets dominated by the majority class [2]. The AUPRC focuses on precision recall trade-offs and offers better evaluations in imbalanced datasets where anomalies are scarce.

Pixel-level anomaly localization: at pixel level, we care about the segmentation of anomaly. Following [7, 25], we use the Dice score metric, which measures the overlap between the predicted and ground truth segmentations, providing a single value that balances false positives and false negatives [2]. The Dice score is given by:

$$\text{DICE} = \frac{2.|A \cap B|}{|A| + |B|} \tag{4.2}$$

where A and B are the anomaly map and the ground truth segmentation, respectively. One disadvantage of this metric is that it requires setting a threshold value from our anomaly score maps to determine whether a pixel is classified as anomalous, which can significantly impact results. As mentioned in Sec. 3.2.3, we will use Yen threshold technique, which can provide automatic threshold value without using any greedy search algorithm.

Chapter 5

Results

Contents

5.1 Reconstruction with SDM

5.1.1 Reconstruction quality

Table 5.1 presents the quantitative results of reconstruction errors for two experiments: denoising from pure random noise and denoising from a stochastic subcode of the input $\mathbf{x}$. The term `semantic` indicates that $\mathbf{x}_T$ is obtained by adding random Gaussian noise, whereas `infer` refers to $\mathbf{x}_T$ generated by applying the reversed DDIM process (stochastic encoding) to $\mathbf{x}$. All results are evaluated by first noising the image to a noise level of 250, followed by denoising using the DDIM sampling scheme with 250 denoising steps. We observe that using $\mathbf{x}_T^{\text{infer}}$ yields much smaller reconstruction errors and significantly higher similarity metrics. This improvement arises because $\mathbf{x}_T^{\text{infer}}$ retains fine-grained details about the input, which are absent in the semantic encoding,

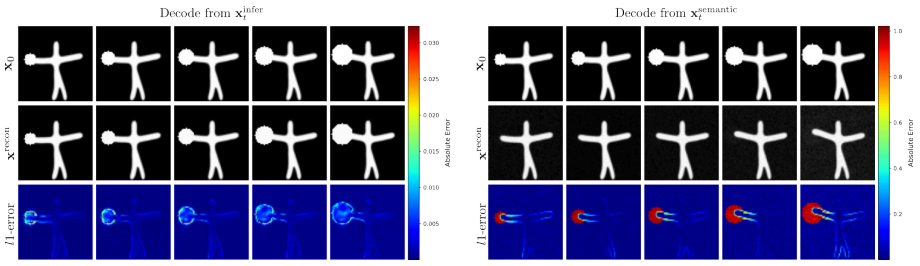


Figure 5.1: Reconstruction error when decoding from $\mathbf{x}_T^{\text{infer}}$ (left) and $\mathbf{x}_T^{\text{semantic}}$ (right). From top to bottom: the original image, the reconstructed image and residual error. Note that the color code is not on the same scale for all images.

and thus better guides the diffusion process toward the true distribution of the original image. For the healthy dataset and subtle anomaly types `darker_line` and `darker_circle`, we observe extreme cases where MSSIM reaches 100%, indicating that perceptually the reconstructed image is indistinguishable from the original to a human observer. On the other hand, this raises a problem of "anomaly preservation", which indicates that the anomaly region is preserved in the reconstructed image. We can see this in the l_1-ratio for `growing_circle`: it is much higher when denoising from $\mathbf{x}_T^{\text{semantic}}$ compared to $\mathbf{x}_T^{\text{infer}}$, because the former completely removes the anomaly, whereas the latter preserves it via information encoded in the stochastic subcode. For the case of very subtle anomalies, $l1$-ratio is higher when denoising from $\mathbf{x}_T^{\text{infer}}$, this is because the l_1-error within the anomaly region is relatively small, which can be attributed to the subtlety of the anomaly. Nevertheless, we can still observe the perceptual effects of anomaly preservation, as well as the qualitative results of different types of reconstruction errors, in Fig. 5.1.

5.1.2 Stochastic subcode from diffusion model

Fig. 5.2 shows the histogram comparing pixel intensities of $\mathbf{x}_T^{\text{infer}}$ and $\mathbf{x}_T^{\text{semantic}}$. We observe that, because $\mathbf{x}_T^{\text{infer}}$ preserves fine-grained details of the original image, its histogram does not follow a strict Gaussian distribution. This effect is more pronounced for anomalous images. This result is in line with the findings of [14, 15], which show that the stochastic subcode from the diffusion model preserves semantic information about the input image. This effect is a result of the DDIM deterministic sampling scheme. As shown in other studies, this characteristic is beneficial not only for reconstruction quality but also for image manipulation. As demonstrated in [15], by modifying the stochastic subcode, we can alter the original image while preserving its overall structure and semantic meaning. However, this requires training an additional network (for example, a classifier) to obtain the decision vector used to modify $\mathbf{x}_T^{\text{infer}}$. The challenges are: (i) the training dataset contains only healthy images, and (ii) $\mathbf{x}_T^{\text{infer}}$ is a spatial representation in $\mathbb{R}^{h \times w}$, which necessitates a patch-based approach or another representation learning network. Without deeper analysis, this presents a challenge in removing subtle anomalies in the context of UAD. We leave the exploration of this direction for future work.

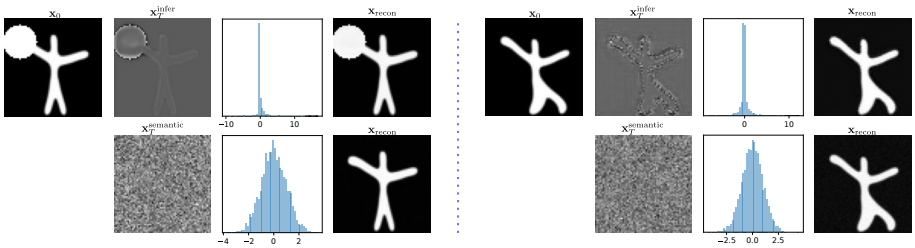


Figure 5.2: The histogram of intensity of $\mathbf{x}_T^{\text{infer}}$ compared to normal noisy $\mathbf{x}_T^{\text{semantic}}$, and their corresponding reconstruction. Left: example from anomaly image. Right: example from healthy image.

5.1.3 Effect of noise level on reconstruction errors

In this section, we investigate the effect of added noise to reconstruction error. Tab. 5.2 presents the reconstruction errors for different noise levels applied to the input image. We test several noise level from 50 to 1000, with 1000 means that the denoising process starts from pure random Gaussian noise. We observe that lower noise levels (50 and 100) yield better reconstruction quality for healthy anatomy, as indicated by lower $l1$-error and higher SSIM and PSNR values. However, these noise levels result in lower anomaly reconstruction errors, which is undesirable

Table 5.1: Comparison of the reconstruction quality of different test datasets. `test_healthy` refers to healthy dataset, while others correspond to various anomaly types. Dataset name follows format `<dataset> (N)`, where N indicates the number of patients (each patient has 10 longitudinal samples). For all metrics, the mean ± standard deviation across whole dataset are reported. The arrows ↑ and ↓ indicate that higher and lower values are favorable, respectively.

	test_healthy (150)		growing_circle (20)	
	semantic	infer	semantic	infer
$l1-$All (e-3) ↓	12.61 ± 23.94	1.04 ± 1.97	32.79 ± 125.79	0.75 ± 2.07
$l1-$Anomaly (e-3) ↑	-	-	743.48 ± 372.27	6.52 ± 6.44
$l1-$Healthy (e-3) ↓	-	-	16.44 ± 33.34	0.62 ± 1.63
$l1-$ratio ↑	-	-	45.23	10.58
$l2-$All (e-3) ↓	0.73 ± 8.26	0.00 ± 0.03	16.90 ± 118.55	0.00 ± 0.08
$l2-$Anomaly (e-3) ↑	-	-	691.34 ± 388.77	0.08 ± 0.45
$l2-$Healthy (e-3) ↓	-	-	1.38 ± 13.98	0.00 ± 0.04
D_f-All (e-3) ↓	2.48 ± 2.02	0.01 ± 0.01	31.50 ± 81.52	0.02 ± 0.05
D_f-Anomaly (e-3) ↑	-	-	306.74 ± 165.98	0.10 ± 0.08
D_f-Healthy (e-3) ↓	-	-	25.17 ± 66.19	0.02 ± 0.05
SSIM ↑	85.70 ± 5.12	99.89 ± 0.02	75.00 ± 14.64	99.86 ± 0.05
MSSIM ↑	98.85 ± 0.50	100.00 ± 0.00	93.76 ± 5.09	99.99 ± 0.00
PSNR ↑	32.66 ± 3.00	53.27 ± 1.32	21.19 ± 6.57	53.19 ± 0.66

(a) Reconstruction metrics for healthy dataset and growing circle anomaly

	darker_line (20)		darker_circle (20)	
	semantic	infer	semantic	infer
$l1-$All (e-3) ↓	14.93 ± 42.74	0.47 ± 1.88	14.60 ± 37.13	0.59 ± 2.09
$l1-$Anomaly (e-3) ↑	400.49 ± 240.64	14.48 ± 10.88	385.45 ± 230.77	19.93 ± 14.25
$l1-$Healthy (e-3) ↓	13.06 ± 28.79	0.41 ± 1.42	13.50 ± 28.58	0.53 ± 1.64
$l1-$ratio ↑	30.66	35.64	28.56	37.65
$l2-$All (e-3) ↓	2.05 ± 27.42	0.00 ± 0.05	1.59 ± 19.48	0.00 ± 0.08
$l2-$Anomaly (e-3) ↑	218.28 ± 216.30	0.33 ± 0.46	201.81 ± 198.32	0.60 ± 0.93
$l2-$Healthy (e-3) ↓	1.00 ± 17.33	0.00 ± 0.03	1.00 ± 12.06	0.00 ± 0.05
D_f-All (e-3) ↓	5.65 ± 9.39	0.02 ± 0.05	3.94 ± 4.49	0.03 ± 0.07
D_f-Anomaly (e-3) ↑	42.74 ± 22.02	0.23 ± 0.07	25.94 ± 7.16	0.36 ± 0.11
D_f-Healthy (e-3) ↓	5.47 ± 8.92	0.02 ± 0.04	3.87 ± 4.32	0.03 ± 0.07
SSIM ↑	84.27 ± 4.92	99.94 ± 0.04	84.53 ± 4.91	99.92 ± 0.04
MSSIM ↑	98.30 ± 0.60	100.00 ± 0.00	98.52 ± 0.48	100.00 ± 0.00
PSNR ↑	27.36 ± 2.02	54.36 ± 0.91	28.43 ± 1.87	53.52 ± 1.40

(b) Reconstruction metrics for darker line and darker circle anomaly

for UAD tasks. Conversely, higher noise levels (750, 800, and 1000) lead to increased anomaly reconstruction errors but also higher reconstruction errors for healthy anatomy. The optimal balance is achieved at a noise level of 250, which provides a good trade-off between accurately reconstructing healthy anatomy and effectively removing anomalies. This result is inline with the *noise paradox* discussed in [8].

Tab. 5.3 provides a detailed breakdown of reconstruction errors for each anomaly type across different noise levels. We observe that the `growing_circle` anomaly (large and more noticeable anomalies) type shows the most significant increase in anomaly reconstruction error as the noise level increases, while the `darker_line` and `darker_circle` (more subtle) anomaly types show more moderate increases. This suggests that the effectiveness of anomaly removal may vary depending on the nature of the anomaly and its contrast with the surrounding healthy tissue. Anomaly with higher intensity needs more noise to be effectively removed, while subtle anomalies require more moderate nosie levels and need more dedicated handling to preserve healthy anatomy.

	Healthy					Anomalies		
Noise level	$l1$-error(1e-3)↓	SSIM(%) ↑	MSSIM(%) ↑	PSNR ↑	LPIPS(1e-3) ↓	$l1$-Ano(1e-3) ↑	$l1$-Healthy(1e-3)↓	$l1$-ratio ↑
50	**10.678**	84.771	98.864	**36.908**	38.093	323.192	**11.737**	27.537
100	11.132	85.222	**98.902**	35.697	35.431	508.632	12.076	42.119
250	12.615	85.694	98.852	32.655	32.041	653.881	13.111	**49.873**
400	13.704	**85.806**	98.751	31.084	30.399	666.405	14.070	47.365
500	14.273	85.586	98.677	30.508	30.457	669.581	14.628	45.775
600	14.642	85.486	98.635	30.162	**30.288**	671.424	14.999	44.763
750	15.250	84.954	98.547	29.850	31.434	672.778	15.598	43.133
800	15.420	84.885	98.513	29.793	31.366	673.050	15.759	42.708
1000	17.953	83.505	97.989	29.539	34.377	**674.115**	18.192	37.056

Table 5.2: The effects of applying different noise levels on reconstruction errors. Arrows ↑ and ↓ indicate that higher and lower values are favorable, respectively. The best results are highlighted in **bold**. Results for the anomalous datasets represent the average over all three types of anomalies.

	growing circle			darker line			darker circle		
Noise level	$l1$-Ano(1e-3) ↑	$l1$-Healthy(1e-3)↓	$l1$-ratio ↑	$l1$-Ano(1e-3) ↑	$l1$-Healthy(1e-3)↓	$l1$-ratio ↑	$l1$-Ano(1e-3) ↑	$l1$-Healthy(1e-3)↓	$l1$-ratio ↑
50	313.516	18.810	16.668	352.260	12.660	27.824	349.308	**11.849**	29.480
100	553.121	19.249	28.735	384.994	**12.228**	**31.486**	372.165	11.993	**31.031**
250	743.802	16.359	45.467	399.816	13.063	30.608	384.890	13.706	28.081
400	759.369	**15.821**	**47.998**	403.404	13.983	28.851	388.863	15.188	25.604
500	763.256	15.957	47.831	404.529	14.598	27.711	389.978	16.020	24.343
600	765.465	16.096	47.557	405.295	14.983	27.051	390.802	16.633	23.495
750	767.102	16.423	46.709	405.784	15.375	26.393	391.408	17.630	22.201
800	767.445	16.445	46.668	**405.840**	15.404	26.346	391.492	17.997	21.754
1000	**768.724**	17.109	44.931	405.403	16.318	24.845	**393.398**	22.926	17.159

Table 5.3: Details of reconstruction errors with different noise levels on anomaly datasets. Best results are highlighted in **bold**

5.2 Unsupervised Anomaly Detection

In this section, we present the result of anomaly detection using psuedo-healthy image from trained SDM model. All reconstructed images are obtained by denoising from $\mathbf{x}_T^{\text{infer}}$ (adding random Gaussian noise to original image). All experiences are done with added noise level of 250, and DDIM sampling is used with 100 denoising steps.

Fig. 5.3 shows examples of residual error ($l1$-error) for healthy and anomaly subjects. Visually, the anomaly part can be detected by the high residual error, relatively compared to other parts

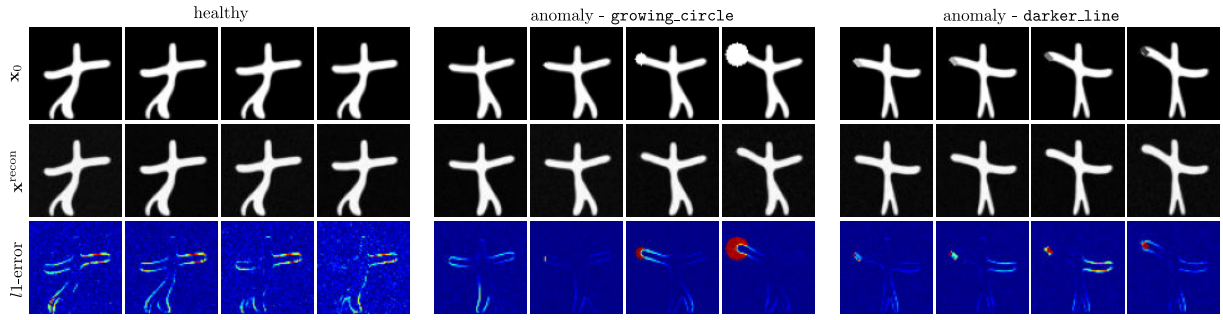


Figure 5.3: Example of residual error $l1$-error for healthy and anomaly subjects. From top to bottom: original image, reconstructed image, and residual error. Left: healthy subject. Middle: anomaly subject with `growing_circle`. Right: anomaly subject with `darker_line`. Note that the color code is not on the same scale for all images.

of the image. However, choosing a proper threshold is proven to be non-trivial, as we can observe high residual error in some healthy parts due to reconstruction quality. This leads to high false positive rate when using simple threshold methods. We can follow some post processing steps from existing literature, such as removing small connected components, but this approach requires prior knowledge about the size of anomalies [7]. In this work, we target subtle anomaly type for Parkinson disease, and we do not want to make any assumption about the anomaly size. On the other hand, we see that at early stage of disease progression, when the anomaly is very small, residual error is hard to distinguish from other healthy parts. This leads to false negative challenge, which can be more detrimental in clinical screening. Fig. 5.7 shows the results of using naive quantile threshold methods: it generates extremely high false positive rate, for both healthy and anomalous subjects.

5.2.1 Anomaly segmentation with Feature Attention Module

To address the shortcomings of the pixel residual map, our FAM module is designed to attend to the information from the feature distance between the original images and their pseudo-healthy reconstructions. Unlike pixel distance, feature distance focuses more on perceptual differences between two images, highlighting errors only when there is a distortion in the overall image structure. FAM learns to ignore minor pixel residuals (reconstruction errors) from our trained FE network Φ (Sec. 3.2.1).

Fig. 5.4 shows the anomaly score maps from FAM as a combination of feature distance D_f and pixel distance D_p. The feature distance correctly captures the overall structural changes in the image. By combining D_f and D_p, FAM reduces reconstruction errors and shifts the focus to regions with high residual and perceptual errors. However, when anomalies are small and subtle (at an earlier stage), we still encounter false negatives—for example, the residual error in the correct region may be relatively small compared to other healthy regions. Note that the color code in Fig. 5.4 are not on the same scale. At time $l = 0$, the red region indicates the highest value in the feature distance map, but the actual feature error there is very small (close to 0), as shown in Fig. 5.5.

Fig. 5.6 shows the effect of using FAM on the pixel distance map. For anomalous subjects, the normalized histogram of residual errors is not a reliable indicator for separating anomalous pixels from healthy ones. By using the feature distance, FAM spreads out the histogram of residual errors, making it easier to apply automatic thresholding or quantile-based methods to localize anomaly regions.

Tab. 5.4 reports the quantitative results of mDICE scores for different thresholding methods.

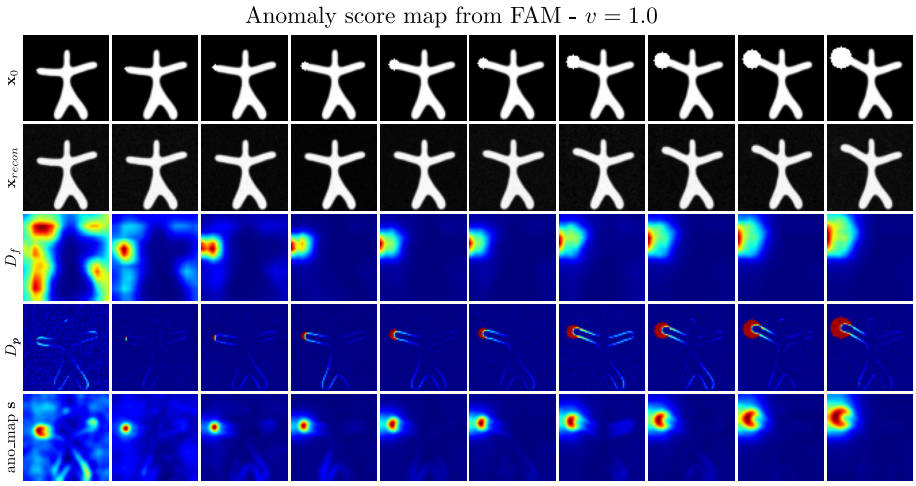


Figure 5.4: Anomaly score map as combination of feature distance and pixel distance. From top to bottom: original image, reconstructed image, pixel distance, feature distance, and anomaly score map as combination of both. The hyperparameter is $v = 1$.

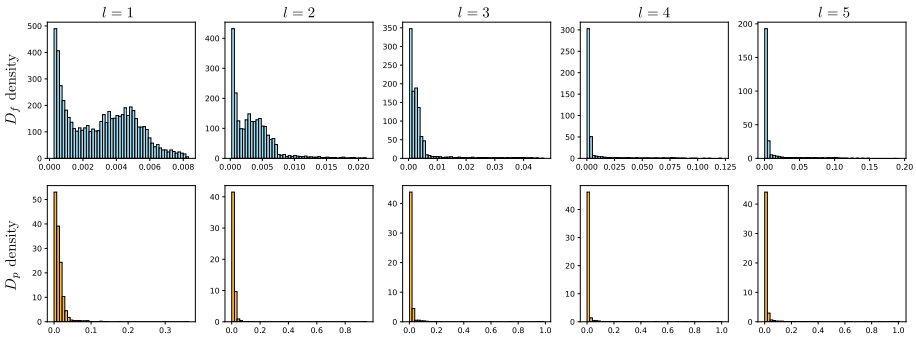


Figure 5.5: Example of normalized histogram of feature distance D_f (top row) and pixel distance D_p (bottom row) for different time points l.

For the healthy dataset, we also report the false positive percentage (FP%), i.e., the proportion of the image area incorrectly marked as anomalous. Pixel-quantile methods compute the quantile for each pixel individually, while quantile methods compute it over the whole image (dataset-wise). Yen method applies Yen automatic threshold technique to the pixel distance map D_p. FAM calculates the anomaly score map by combining feature distance and pixel distance. Quantile methods are applied to the entire test dataset. For the Yen and FAM methods, results are reported for both dataset-wise and patient-wise thresholds. The dataset-wise method computes a single threshold value for the whole test dataset (including both healthy and anomalous images). The patient-wise method computes one threshold per patient, considering all observations for that patient (10 observations per patient).

We observe that the classic quantile methods perform poorly in terms of both false positive rates and mDICE scores. The Yen method achieves strong anomaly segmentation performance, attaining some of the highest mDICE scores, except for the `growing_circle` anomaly type, where the FAM method achieves higher segmentation quality. Moreover, FAM yields the best results in terms of minimizing the false positive rate in healthy regions. The result indicates that Yen method has the best overall mDICE score of 69% only if we use the whole dataset to calculate threshold, which increase almost 10% compared to patient-wise result of 58%. We see similar results for FAM method, where all mDICE score is higher when the threshold is calculated

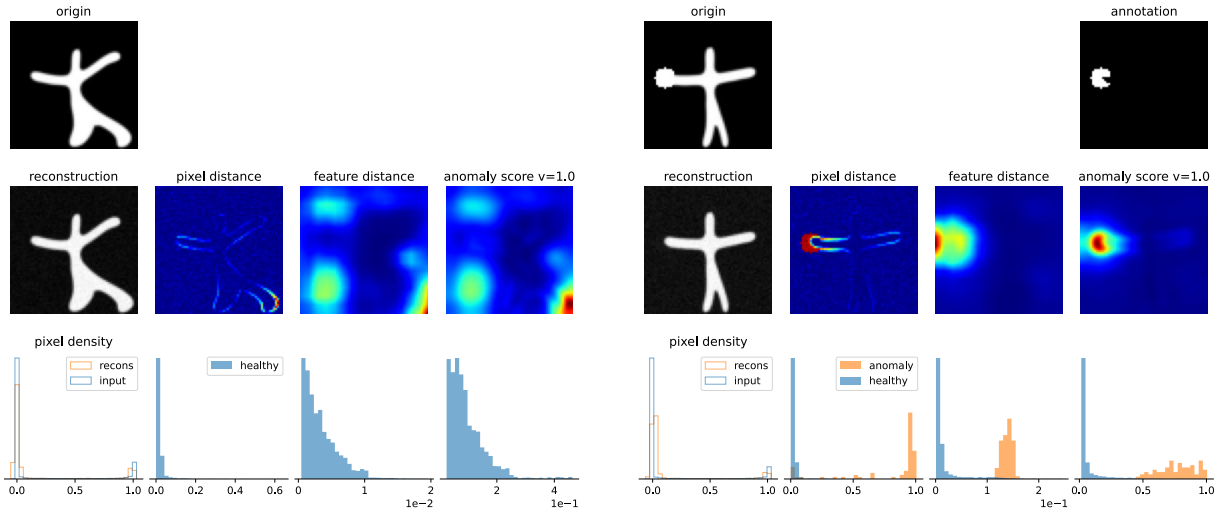


Figure 5.6: The effect of combination of feature distance and pixel distance. Left: example of healthy image. Right: example of anomaly image. The hyperparameter is $v = 1$

dataset-wise. This is expected, as having more data provides better information to separate healthy and anomalous pixels. On the other hand, FP is best with patient-wise method.

Fig. 5.7 displays examples of anomaly localization results for different methods. It is clear that the quantile methods perform poorly, as they fail to disentangle anomaly errors from model reconstruction errors. For healthy subjects, FAM outperforms the Yen method in minimizing false positive segmentations.

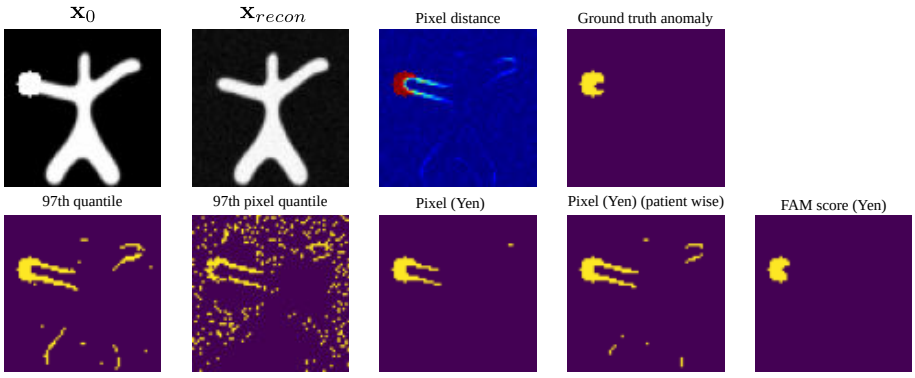
Notably, one drawback of applying thresholding techniques dataset-wise is that the results strongly depend on the composition of the dataset. Our test dataset comprises 150 healthy subjects and 60 anomalous subjects (20 for each type of anomaly), meaning that approximately 70% of the samples are healthy. Because healthy images have smaller residual errors, having more healthy references helps to better highlight anomalous regions. Fig. 5.8 and Tab. 5.5 illustrate the fluctuations in FP and mDICE scores for different dataset compositions. In particular, we vary the number of healthy subjects while fixing the number of anomalous ones.

As expected, mDICE scores increase as the number of healthy references grows. One interesting observation is that, while the Yen method achieves a higher mDICE score than FAM, FAM demonstrates more stable performance, maintaining consistent scores across different numbers of healthy observations. For false positive rates, FAM also exhibits smaller fluctuations than the Yen method, although we do not see the same consistency as for mDICE.

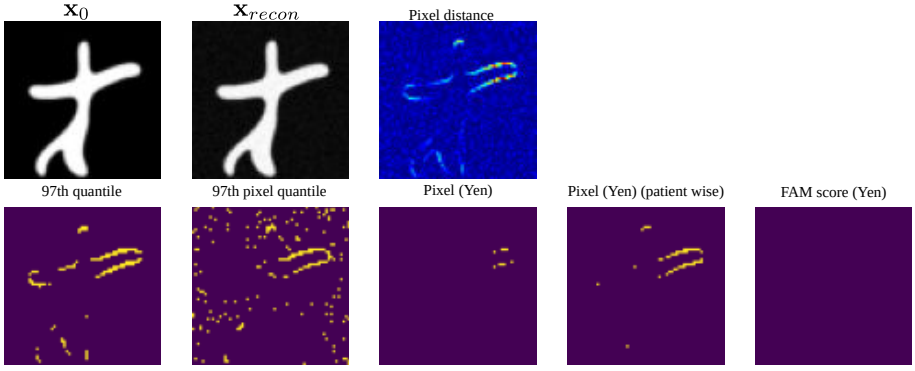
In a real case scenario, we do not know the real composition of our test dataset, and ideally we want to improve our method so that it only depends on information per patient, making it more robust for unbalanced dataset. Also, FAM mainly solves the problem of false positive, but for real anomaly detection its performance is still inferior to normal automatic Yen threshold method. Fig. 5.7c shows 1 example where FAM fails to detect subtle anomaly, while Yen method can detect the anomaly but also include other healthy regions. In the next section, we present our result for anomaly segmentation with our proposed Longitudinal Attention Fusion Module.

5.2.2 Anomaly segmentation with Longitudinal Attention Fusion Module

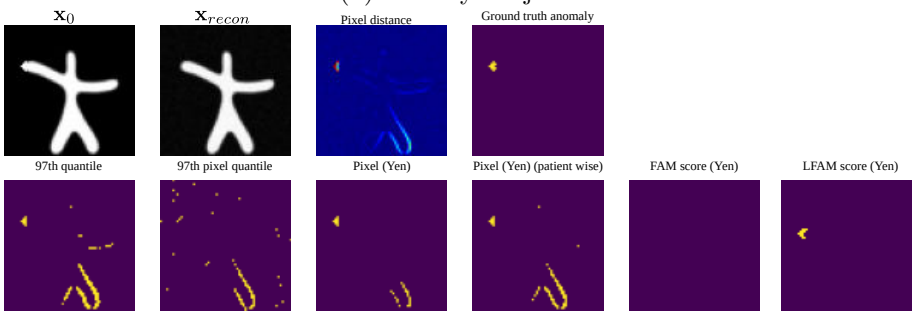
By fussing the information from multiple time points, LAFM aims to improve the accuracy of anomaly detection, both reducing false positives (anomalies that do not persist over time) and false negatives (use anomalies that are detected from future time points to help detect subtle anomalies at earlier time points).



(a) Anomaly subject.



(b) Healthy subject.



(c) Anomaly subject (subtle).

Figure 5.7: Example of anomaly segmentation using different methods. Fig. 5.7a and Fig. 5.7c show examples of anomaly subjects, while Fig. 5.7b shows an example of a healthy subject. The first row: original image, reconstructed image, residual error, and ground truth anomaly segmentation (if any). The second row: segmentation using image-wise 97% quantile, pixel-wise 97% quantile, dataset-wise Yen thresholding, patient-wise Yen thresholding, and anomaly score map with FAM.

Methods		Healthy	Anomaly			
		FP (%) ↓	growing circle mDICE% ↑	darker circle mDICE% ↑	darker line mDICE% ↑	Total mDICE% ↑
Pixel quantile	q90.0	9.081	16.526	6.477	11.751	11.585
	q92.5	6.702	19.513	7.572	14.354	13.813
	q95.0	4.310	23.718	8.039	16.960	16.239
	q97.5	2.041	27.565	7.193	16.743	17.167
	q99.0	0.734	24.889	4.551	10.016	13.152
Quantile	q90.0	9.152	17.125	6.663	11.592	11.793
	q92.5	6.768	20.953	8.592	14.577	14.707
	q95.0	4.435	28.061	12.165	19.784	20.004
	q97.5	2.122	42.638	22.375	33.193	32.736
	q99.0	0.684	60.276	44.588	57.832	54.232
Pixel score (Yen)	dataset wise	0.192	**76.867**	59.600	70.683	**69.050**
	patient wise	1.645	47.369	58.500	**70.872**	58.914
Anomaly score map FAM	dataset wise	0.204	60.531	**65.254**	62.455	62.747
	patient wise	**0.059**	57.782	49.798	44.877	50.819

Table 5.4: DICE scores for the anomaly segmentation task, reported for different methods. For anomaly subjects, `Total` is the average of mDICE scores for all anomaly types. Best results are highlighted in **bold**.

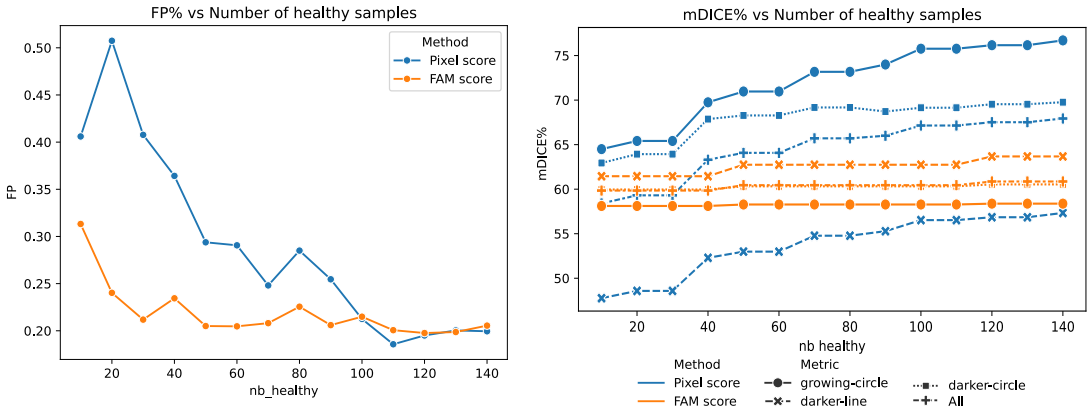


Figure 5.8: FP(%) and mDICE(%) scores for different unbalanced compositions of the test dataset. Left: FP % for healthy patients. Right: mDICE % score for the anomaly segmentation task. The x-axis denotes the number of healthy patients in the dataset, the number of anomalous patients is fixed at 60.

	Pixel distance (Yen threshold)					Feature Attention Module				
nb healthy	healthy FP(%) ↓	growing-circle mDICE% ↑	darker-line mDICE% ↑	darker-circle mDICE% ↑	All mDICE% ↑	healthy mDICE% ↑	growing-circle mDICE% ↑	darker-line mDICE% ↑	darker-circle mDICE% ↑	All mDICE% ↑
10	0.401	65.986	51.072	64.600	60.553	0.240	60.068	61.929	61.460	61.152
20	0.379	66.804	51.798	65.485	61.362	0.213	60.068	61.929	61.460	61.152
30	0.415	68.433	53.291	67.230	62.985	0.247	60.298	63.275	61.945	61.839
40	0.331	68.433	53.291	67.230	62.985	0.178	60.298	63.275	61.945	61.839
50	0.268	70.670	55.110	68.992	64.924	0.186	60.445	64.423	62.263	62.377
60	0.348	71.752	55.728	69.356	65.612	0.209	60.445	64.423	62.263	62.377
70	0.253	71.752	55.728	69.356	65.612	0.171	60.445	64.423	62.263	62.377
80	0.241	74.453	57.774	69.748	67.325	0.220	60.445	64.423	62.263	62.377
90	0.244	75.114	58.066	69.651	67.611	0.211	60.445	64.423	62.263	62.377
100	0.218	75.247	58.504	69.803	67.851	0.204	60.445	64.423	62.263	62.377
110	0.204	76.033	58.883	70.128	68.348	0.194	60.531	65.254	62.455	62.747
120	0.210	76.386	59.171	70.480	68.679	0.203	60.531	65.254	62.455	62.747
130	0.200	76.386	59.171	70.480	68.679	0.211	60.531	65.254	62.455	62.747
140	0.197	76.867	59.600	70.683	69.050	0.201	60.531	65.254	62.455	62.747

Table 5.5: FP(%) and mDICE(%) results for different numbers of healthy patients in the test dataset. The number of anomaly patients is fixed at 60. The left part of the table shows results using Yen threshold on pixel distance, while the right part shows results using the Feature Attention Module (FAM).

	Healthy	Anomalies			
methods	healthy FP(%) ↓	growing circle mDICE% ↑	darker circle mDICE% ↑	darker line mDICE% ↑	All mDICE% ↑
Pixel score (dataset wise)	0.192	76.867	59.600	70.683	69.050
Pixel score (patient wise)	1.645	47.369	58.500	70.872	58.914
FAM (dataset wise)	0.204	60.531	65.254	62.455	62.747
FAM (patient wise)	**0.059**	57.782	49.798	44.877	50.819
LAFM ($\sigma = 0.0$)	0.169	84.790	81.665	86.924	84.460
LAFM ($\sigma = 0.5$)	0.198	82.441	**82.784**	**87.950**	84.391
LAFM ($\sigma = 1.0$)	0.191	82.893	81.926	87.891	84.237
LAFM ($\sigma = 1.5$)	0.178	83.604	82.682	87.909	84.731
LAFM ($\sigma = 2.0$)	0.207	85.540	81.997	87.634	85.057
LAFM ($\sigma = 2.5$)	0.183	**86.069**	81.981	87.352	**85.134**
LAFM ($\sigma = 3.0$)	0.184	85.995	81.779	87.304	85.026
LAFM ($\sigma = 3.5$)	0.198	85.890	81.648	86.929	84.823
LAFM ($\sigma = 4.0$)	0.194	85.762	81.826	86.893	84.827
LAFM ($\sigma = 4.5$)	0.183	85.783	81.683	86.779	84.748

Table 5.6: mDICE and FP scores for different anomaly segmentation methods. The best results are highlighted in **bold**.

Tab. 5.6 reports anomaly segmentation metrics using LAFM with different variants of Gaussian smoothing σ. Here, $\sigma = 0$ indicates that time points are weighted equally. As discussed in Sec. 3.2.2, σ controls the contribution of anomaly score maps from surrounding time points, depending on their distance from the current target image. The mDICE metric increases to around 86%, compared to 62% and 69% for FAM and Yen, respectively. LAFM only underperforms FAM in terms of the FP metric, but its result is still comparable to the Yen method, with an acceptable rate of around 0.19%. This highlights the trade off between FP and FN: method that achieves the best FP rate processes the residual error more conservatively at the cost of potentially missing some anomalies. On the other hand, method with higher segmentation accuracy will increase the likelihood of anomalies, which leads to more healthy regions being flagged as anomaly.

Fig. 5.9 shows 1 example of anomaly segmentation from LAFM. We see that by using information from other time points, we can refine the anomaly map by adding anomaly regions that persist across time and reducing potential anomalies that do not. This effect is more apparent at earlier stages, for example, in the first two time points.

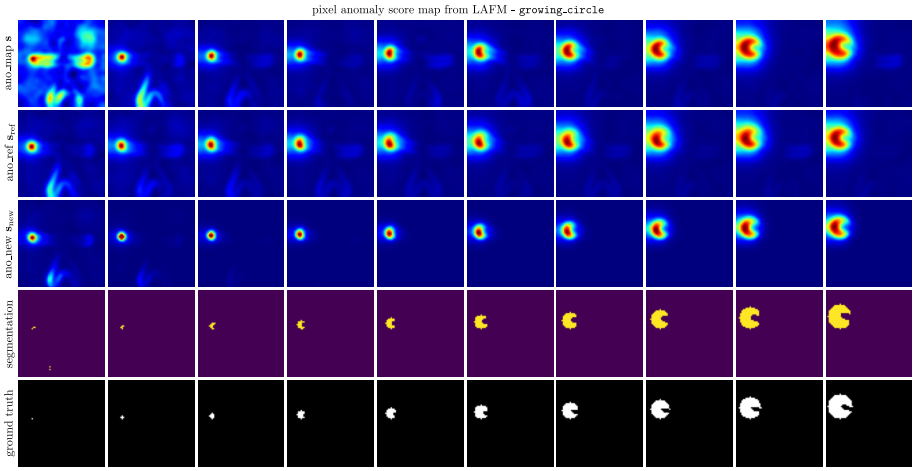


Figure 5.9: Anomaly segmentation from LAFM. From top to bottom: anomaly map (targeted image), anomaly map references from other time points, updated anomaly map based, anomaly segmentation and ground truth annotation.

One disadvantage of LAFM is that its performance depends on the number of observations for each patient. The more observations we have, the more additional information we can use to improve the current prediction. Fig. 5.10 shows the performance of Yen, FAM and LAFM with different numbers of observations (per patient). For each experience, we randomly choose n samples for each patient's longitudinal samples ($\max(n) = 10$). We note that for each n, the interval between time points Δ_l are random. Since the referenced anomaly score maps are weighted by a Gaussian kernel that depends on Δ_l, this randomness can affect the overall performance. To account for this, we run the experience 10 times and reports the average results.

We see that Yen and FAM have small fluctuations with different number of observations, while LAFM's performance increases as we have more observations. mDICE increases significantly from 2 to 4 observations, and from 4 observations onward, LAFM starts outperforming Yen. On the other hand, FP only reaches stable performance when we have at least 8 observations. This confirms that the strength of LAFM lies in its reliability for anomaly segmentation, rather than in minimizing false positives.

Fig. 5.11 displays the results of pixel anomaly score map with varying number of observations from LAFM. We see that with more observation, we have more referenced information to correct residual errors (e.g. reconstruction error at first time point $l = 0$). In the extreme case, in which

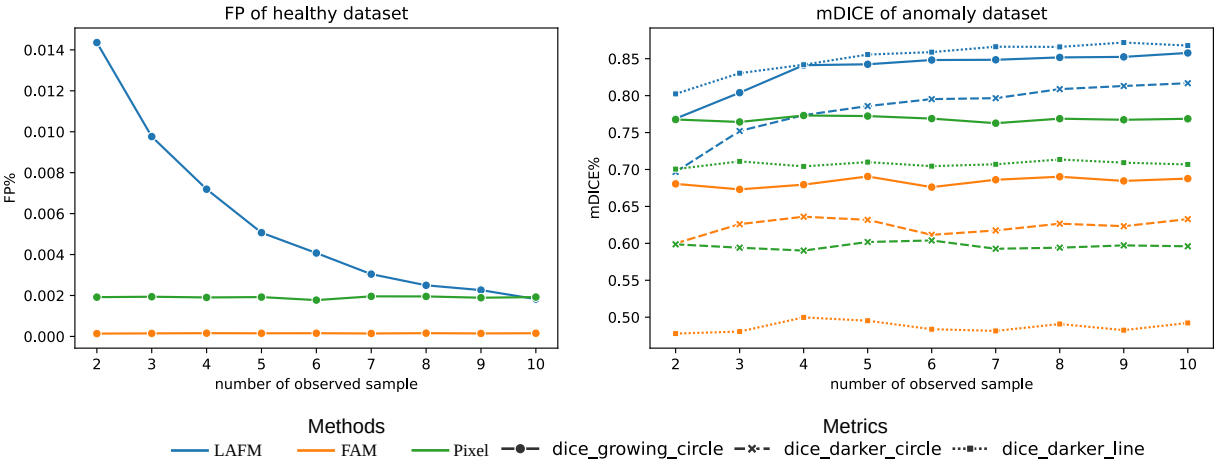


Figure 5.10: Comparison of FP% and mDICE% scores for different numbers of observed time points. Results for LAFM is calculated with $\sigma = 0.5$.

we only have two observations, and they are far away from each other (time points $l = 2$ and $l = 9$), the model exhibits negative effect: wrong information (false positive) from the first time point is used as reference for the other. This dilutes the correct anomaly score and reduces the accuracy of segmentation. However, we argue that this cannot be immediately regarded as a defect of the model. By design, LAFM is intended to attend to as much information from longitudinal data as possible. When we have only two observations, it is hard to choose with one contains the correct information. This holds true specially in real case scenario, when we do not know the ground truth. It can be the case that earlier time point helps correct false positive for later one.

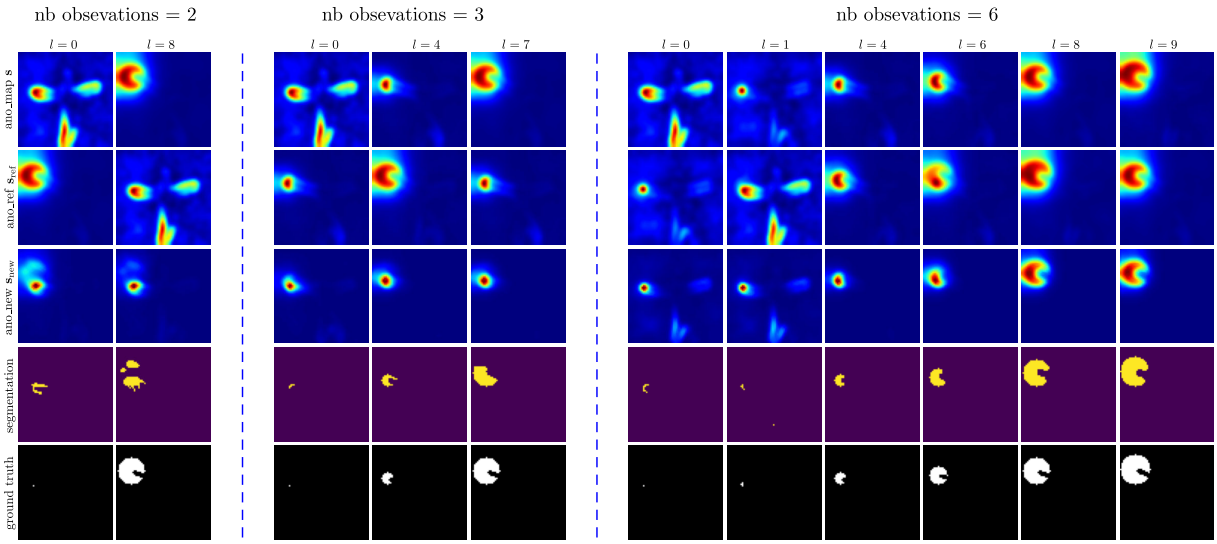


Figure 5.11: Effect of number of observations on output of LAFM.

5.2.3 Image anomaly score

Follow method discussed in Sec. 3.2.3, we assign image level anomaly score as a summary operator of pixel level anomaly score map. We employ several common strategies, namely:

- `max`: use the maximum pixel score as image anomaly score.

- `mean`: assign the average pixel scores as image anomaly score.
- `mean-top`k: calculate the average of top k pixel scores as image anomaly score.
- q`-quantile`: calculate the average of top qth percentile pixel scores as image anomaly score.

Tab. 5.7 reports the AUPRC and AUROC metrics for different methods and strategies. We apply each strategy to different pixel anomaly score maps that we have obtained so far. Pixel, FAM and LAFM refer to the original pixel residual map D_p, pixel anomaly map from FAM and LAFM, respectively. A key observation is that pixel residual error achieves the highest value for both AUROC and AUPRC at 97.5% and 92.9% respectively, while our proposed methods FAM only has performance around 86-90%. We suggest that this may result from the application of several post-processing steps to D_p. FAM and LAFM are good for pixel segmentation, but also it neutralizes some information in the pixel residual map. For example, LAFM smooths the pixel anomaly map by concatenating information from other time points, which may reduce the contrast between healthy and anomalous images.

We also see that using max operator remains the best strategy to assign image anomaly score. On the other hand, mean operators have the worst performance, as it average all pixel scores. By doing so, it dilutes the anomaly information because most of our residual errors are small (background pixel). We can observe this effect clearly in mean-top-k and mean-q-quantile methods, where the performance increases as we decrease the number of pixels that we consider. On the other hand, we see that LAFM has more stable performance than other methods. Even though mean operators still have lower performance, LAFM remains around 90% AUROC and 85%AUPRC, which is better than FAM or Pixel score. Fig. 5.12 and Fig. 5.13 show the AUPRC and AUROC curves using pixel distance D_p, and LAFM anomaly score, respectively.

	AUPRC(%)			AUROC(%)		
	FAM	LAFM	Pixel	FAM	LAFM	Pixel
`max`	*86.714*	*92.074*	**92.889**	*90.017*	*96.589*	**97.509**
`mean`	23.558	74.778	62.905	35.604	83.692	77.458
`mean-top50`	83.040	87.680	77.815	86.630	93.547	89.823
`mean-top100`	79.827	86.380	73.208	83.723	92.484	86.521
`mean-top150`	77.073	85.752	71.114	81.295	91.872	85.002
`mean-top200`	74.347	85.266	70.011	79.067	91.422	84.242
`90.0-quantile`	62.750	83.494	68.585	70.534	90.051	83.256
`92.5-quantile`	68.351	84.307	69.030	74.536	90.662	83.593
`95.0-quantile`	74.068	85.215	69.939	78.842	91.379	84.192
`97.5-quantile`	79.660	86.336	73.036	83.572	92.442	86.398
`99.0-quantile`	83.693	88.184	79.268	87.243	93.861	90.781

Table 5.7: AUPRC and AUROC for image anomaly score. `mean-top`k refers to taking the mean of the top k the highest pixel anomaly scores. q`-quantile` refers to taking the mean of the pixel scores at the qth percentile. The best results for AUPRC and AUROC are highlighted in **bold**, calculated across all methods. Best results for each method are highlighted in *italic*

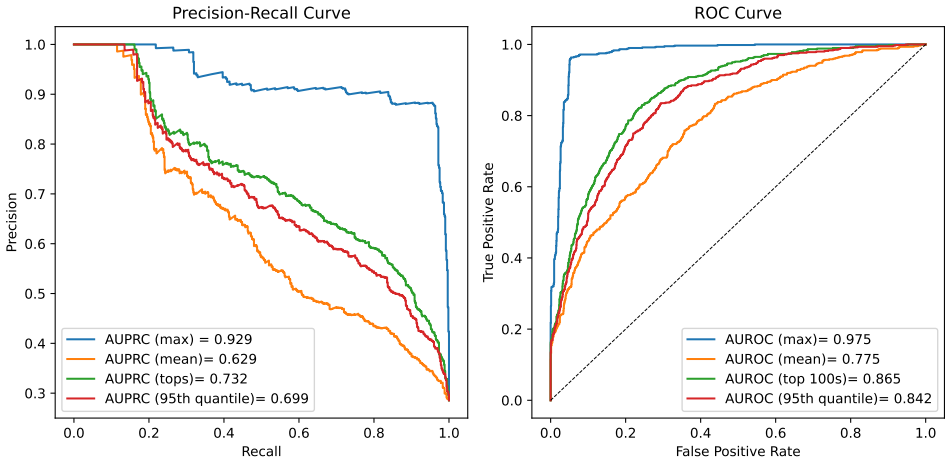


Figure 5.12: AUROC and AUPRC curves for image anomaly score with different methods, apply to pixel distance D_p. Left: AUPRC curves. Right: AUROC curves.

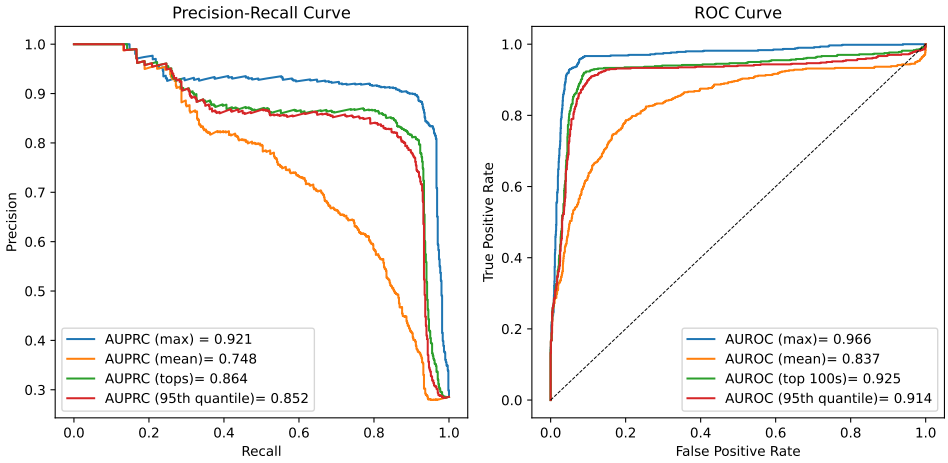


Figure 5.13: AUROC and AUPRC curves for image anomaly score with different methods, apply to LAFM anomaly score. Left: AUPRC curves. Right: AUROC curves.

Chapter 6

Temporal Diffusion Model for data imputation

In this chapter, we introduce our implementation of temporal diffusion model for follow-up data generation. Our model is based on TADM model [32], which uses diffusion process to learn the residual growth of images in the temporal space. We cover the methodology in Sec. 6.1. Sec. 6.2 explains the implementation of our model, and Sec. 6.3 shows the results of our experiment. We also utilize `Starmen` dataset in this experiment.

Contents

6.1 Temporal Diffusion Model for follow up data generation

In this section, we come back to our spatial-temporal setting and introduce our framework to use diffusion model to learn temporal dependencies between samples. As mentioned earlier, one major challenge in unsupervised anomaly detection in MRI scans is the presence of missing scans. Consequently, this raises the important task of imputing missing data based on the available observations. Consequentially, this raises an important domain of imputation of missing data, based on observation. We denote $\mathbf{X}_i^{\mathcal{O}} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{s}_L\}$ as our set of observed data, and $\mathbf{X}_i^{\mathcal{M}} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{s}_M\}$ the set of missing data that we want to impute, for a patient i. Since both training and inference are performed on a per-patient basis, we drop the subscript i for brevity. Our problem setting for temporal diffusion model becomes to learn the conditional distribution of $p(\mathbf{x}_m|\mathbf{x}_l)$ with $m \in [1 \ldots M]$ and $l \in [1 \ldots L]$. Our model operates as an autoregressive model that sequentially reconstructs the follow up sample from the previous one, so we have $m < l$. We note that m and l do not need to be consecutive, so $(l - m) \geq 1$.

To characterize temporal features within sequences, we adopt the implementation of Temporal-Aware Diffusion Model (TADM) [32]. Compared to other existing SOTA models, TADM offers several advantages:

- It provides more flexibility than approaches based on interpolation ([15]), which requires two input images to be able to generate (interpolate) missing images in between. On the

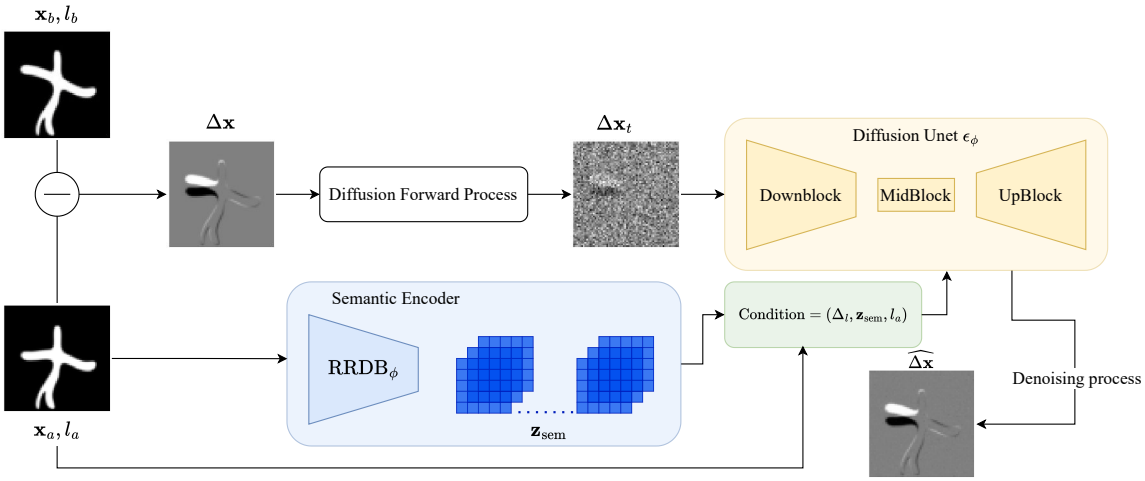


Figure 6.1: Overview of our Temporal Diffusion Model (TDM). Our model takes as input the residual change $\Delta\mathbf{x}$ between 2 time points $l_a < l_b$, the conditions to the model are: the current baseline age l_a, a list of spatial representations extracted from RRDB [33] encoder, the time interval Δ_l. During **training**, the inputs go through normal diffusion forward-backward process and conditional UNet learns to predict the added noise. At **inference**, we sample a Gaussian random noise $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$, use trained UNet to denoise with DDIM sampling scheme to predict the residual change. The target image is constructed by adding predicted residual to baseline image $\widehat{\mathbf{x}}_b = \mathbf{x}_a + \widehat{\Delta\mathbf{x}}$

other hand, TADM has the capacity of generating follow up image in any time point in the future based on current image.

- Instead of learning the whole age-related changes within the image, TADM is trained by learning the residual changes between 2 time points. This reduces the complexity of the problem, and minimize the generation errors.

- Consequently, TADM conditions the model on the age gap between the input and output scans rather than directly on the output age. The authors argue that because the same age gap can happen between scans acquired at difference ages, conditioning on age gap avoids the necessity of including samples from every age group in the training set. This is particularly beneficial when the dataset has limited samples in some age groups [32].

By combining age gaps (time interval) and residual changes (spatial difference), TADM is trained to learn the distribution of temporal progression. During training, we use pairs of images denoted as $\mathbf{x}_a$ and $\mathbf{x}_b$, acquired from the same patient at two different time points $l_a < l_b$. These scans are used to compute the residual change $\Delta\mathbf{x} = \mathbf{x}_b - \mathbf{x}_a$, which represents the image growth over the interval $\Delta_l = l_b - l_a$. TADM takes $\Delta\mathbf{x}$ and Δ_l as input and undergoes the standard forward–backward diffusion process to learn the underlying distribution $p(\Delta\mathbf{x} \mid \Delta_l)$. At inference, by performing the denoising process, we can sample $\widehat{\Delta\mathbf{x}}$ given a baseline age l_a and time interval Δ_l. The future image at l_b is then predicted as $\widehat{\mathbf{x}}_b = \mathbf{x}_a + \widehat{\Delta\mathbf{x}}$. Fig. 6.1 shows an overview of our TDM framework.

One major concern with this approach is that different patients may exhibit different rates of change over the same time interval. To account for this, TDM also employs an encoder to extract a non-spatial representation of the input, which is then used as a conditional signal to guide the diffusion process. We make the underlying assumption that the current image contains the most

meaningful information about a patient's state up to this point. In other words, the sequence of images can be viewed as a Markov chain, where future images depend only on the previous one. In addition, we also condition the diffusion process on baseline age l_a, under the assumption that the rate of change varies across different ages. Following [32], our TDM comprises two blocks: i) an encoder to extract semantic representation of baseline image, and ii) an UNet block as diffusion model to predict the noises added. Our TDM differs from TADM in that TADM employs an additional Brain Age Estimation (BAE) module to predict the age gap between the predicted image $\widehat{\mathbf{x}_b}$ and the baseline $\mathbf{x}_a$. It then incorporates the difference between the predicted age gap, $\widehat{\Delta_l} = BAE(\widehat{\mathbf{x}_b}, \mathbf{x}_a)$, and the true age gap Δ_l into its loss function. Conceptually, this can be interpreted as an adversarial loss, similar to that used in GAN models. This requires additional step of pretraining the BAE module, and we omit it in our experiment for simplicity. Our denoising U-Net is also a conditional diffusion model that follows the same methodology as SDM. For the semantic encoder, we employ the Residual-in-Residual Dense Blocks (RRDB) module [33]. Unlike the semantic encoder in SDM (ResNet50), RRDB outputs a list of spatial representations of the input, denoted as $\mathbf{z}_{\text{sem},i} \in \mathbb{R}^{h \times w}$, corresponding to the i-th block of the RRDB.

Training : during diffusion step, TDM is trained to predict the noise ϵ added to the input Δ_l at time step t. The loss function from E.q (2.9) is updated to incorporate patient specific data as follows:

$$\begin{aligned} \mathcal{L}(\theta, \omega) &= ||\epsilon - \epsilon_\theta(\Delta\mathbf{x}_t, t; \Delta_l, l_a, \mathbf{Z}_a||^2 \\ \mathbf{Z}_a &= \text{Enc}_\omega(\mathbf{x}_a)) \end{aligned} \tag{6.1}$$

where $\Delta\mathbf{x}_t = \sqrt{\bar{\alpha}}\Delta\mathbf{x}_0 + \sqrt{\bar{\alpha}_t}\epsilon$ (E.q (2.4)), $t \sim \text{Unif}[1, T]$ is the time step, z_a is latent representation of baseline image from RRDB encoder. Similar to our spatial diffusion model, parameters of UNet and Encoder, θ and ω respectively, are jointly trained through backpropagation.

Inference : our model impute missing scans by an autoregressive process. Given a sequence of observed data $\mathbf{X}_i^{\mathcal{O}} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{s}_L\}$, we impute the missing image at time m by finding the closest image from the observed set $\mathbf{x}_l \in \mathbf{X}^{\mathcal{O}}$ such that $l < m$. Our model takes as inputs baseline image $\mathbf{x}_a$ and time interval with respect to baseline time $\Delta_l = m - l$. The reverse process starts from random Gaussian noise $\Delta_T \sim \mathcal{N}(0; \mathbf{I})$ and progressively denoises through $\epsilon_\theta(\Delta_T, t; \mathbf{x}_a, \Delta_l, z_a)$. The predicted residual change $\widehat{\Delta_l}$ is then added to baseline image to generate missing image $\widehat{\mathbf{x}_m} = \mathbf{x}_a + \widehat{\Delta_l}$.

6.2 TDM implementation

UNet : Our temporal diffusion model is modified from TADM implementation [32] [1]. Similar to base diffusion model, our UNet comprises input blocks (with downsampling layer), a middle block, and out blocks (with upsampling layer) which are the symmetric counterparts of corresponding downsampling blocks. Down and Up blocks contain 4 levels with channel multipliers of $[32, 64, 128, 256]$. Unlike SDM model, each level in TDM has 2 residual blocks (ResBlock), followed by downsampling (upsampling) block. We omit cross-attention layers in the TDM, as the model operates on residual changes $\Delta\mathbf{x}$ rather than the original image $\mathbf{x}$, where spatial information is less prominent. This is similar to the implementation from TADM paper.

[1] Available at https://github.com/MattiaLitrico/TADM-Temporally-Aware-Diffusion-Model-for-Neurodegenerative-Progression-on-Brain-MRI

Condition signals : similar to SDM, time step condition t is embedded using sinusoidal embeddings. The condition dimension is $d_{cond} = 32$. For semantic encoder, we use RRDB [33] as our backbone network, with 8 blocks and each block has 64 channels. The RRDB is not initialized but is trained from scratch jointly with UNet module. Other patient-specific data, such as age l_a and age difference Δl, are projected using a linear layer to match the dimension of the time embedding vector. Unlike the SDM, all conditions are injected into the ResBlocks by being summarized with the model's hidden state. We note that all conditions (except semantic encoded z_a) are represented as vectors, while the hidden states are spatial representations produced by Conv2D layers. Therefore, the summation is performed via broadcasting. Formally we have $h := h + \psi_1(t) + \psi_2(l_a) + \psi_3(\Delta_l) + z_a$, with ψ_1, ψ_2, ψ_3 are trainable projection layers applied to time step, age and time interval, respectively.

Training configuration : TDM is trained with 500 epochs, using Adam optimizer with learning rate 2.5×10^{-4}. Similar to SDM, we employ EMA strategy to smooth out parameters, with EMA decay rate is 0.9999 and EMA is updated after every 10 batches. Tab. 6.1 shows the details configurations of our TDM network.

Table 6.1: Temporal Diffusion Model (TDM): configurations and parameters

Encoder Enc_ω	
Backbone	`RRDB` [33]
Input Modality	$1 \times 64 \times 64$
RRDB number of blocks	8
RRDB number of features	64
Diffusion UNet ϵ_θ	
Input Shape	$(B, 1, 64, 64)$, with batch size first
Channels multipliers	[32, 64, 128, 256]
Residual Blocks per Level	2
Conditional injection	Summarize
Dropout	0.1
Time embedded dimension	$d_{cond} = 32$
Timestep	1000
Beta Schedule	Linear, $\beta_t \in [10^{-4}, 2 \times 10^{-2}]$
Training Configuration	
Optimizer	Adam
Learning Rate	2.5×10^{-4}
EMA Decay	0.999
Train Batch Size (effective)	20
Training Duration	500 epochs / ~4.5 hours
Hardware	1 x Nvidia L40S (45 GiB)

6.3 Results

In this section, we present our results on using a temporal diffusion model to impute missing data. We demonstrate how the model leverages temporal information from previous observations

to generate accurate predictions for the missing values, and we analyze its performance across different scenarios.

6.3.1 Imputing subsequence images

First, we test the reconstruction performance of Temporal Diffusion Model (TDM) in case of predicting images using information from preceding images. In this simple case, the time interval is $\Delta_l \sim 1$. Tab. 6.2 shows the similarity metrics and residual errors of subsequently imputation for different anomaly types. We see that TDM achieves the best performance on the `healthy` dataset, which is expected since, during training, our model only sees normal samples. When TDM encounters anomalous images during inference, even though semantic representations are provided by the RRDB semantic encoder, our model does not know how to process this information and fails to predict how the anomaly will progress over time. Not only that, but the presence of an anomaly distorts the normal semantic representation, causing our model to produce lower-quality follow-up images. This can be observed in Fig. 6.3b. Starting from the 2^{nd} observation (5^{th} time point), the anomaly is present, causing subsequent imputations to deteriorate in quality. The anomaly does not grow but rather remains blurry in the following time points.

For healthy dataset, we see that TDM achieves very high results, both in terms of similarity metrics and residual errors, with MSSIM closed to 100%, and LPIPS is around 26×10^{-3}.

	Similarity metrics				Residual errors	
	SSIM ↑	MSSIM ↑	PSNR ↑	LPIPS (e-3) ↓	$l1$-error(e-3) ↓	$l2$-error(e-3) ↓
`healthy`	77.26	99.17	37.40	26.61	9.90	0.19
`growing_circle`	75.02	97.72	23.97	33.70	17.77	6.31
`darker_circle`	76.68	99.07	33.05	29.19	10.99	0.52
`darker_line`	75.82	98.90	31.73	29.31	11.60	0.71

Table 6.2: Imputation error: using the previous image to predict the next consecutive image. Results are reported for each type of anomaly.

6.3.2 Imputation with random time interval

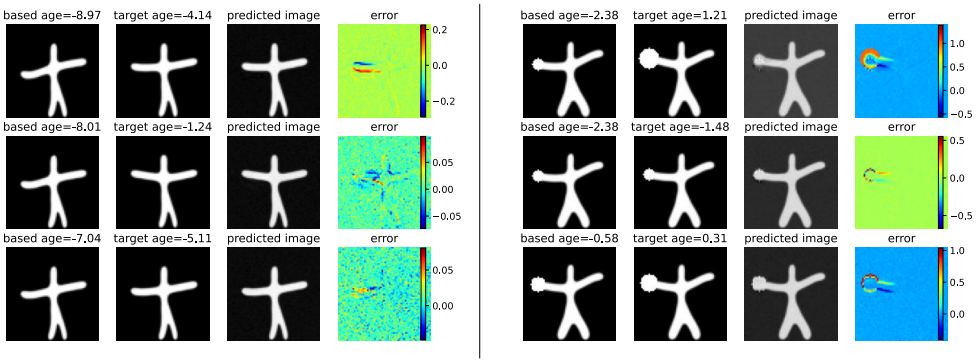


Figure 6.2: Example of imputation with random time interval. Left: healthy sample. Right: anomaly sample. For each image, from left to right: current image, future image, predicted image, and residual error ($l1$-error)

Next, we test the model capacity of generating follow-up data at arbitrarily Δ_l time steps in the future. Tab. 6.3 reports reconstruction errors for healthy dataset. Results are reported as

average of different time gap group. Gap bin $(0-1)$ corresponds to our previous case: imputing subsequence data. We clearly see that the quality decreases as time gap increases, for both similarity metrics and pixel residual errors. The only exception is LPIPS metrics, where we see the best result is obtained at time point really far into the future (time gap bin $(10-11)$), although the differences is small compared to case of subsequent imputation.

This result can be explained by the fact that the residual growth between 2 consecutive time points is relatively smaller. So even though our model is trained with all random pairs of images (thus it is trained with all time differences), its performance is best when we use preceding image as input conditions. Fig. 6.2 shows examples of generating follow-up images with different time gap.

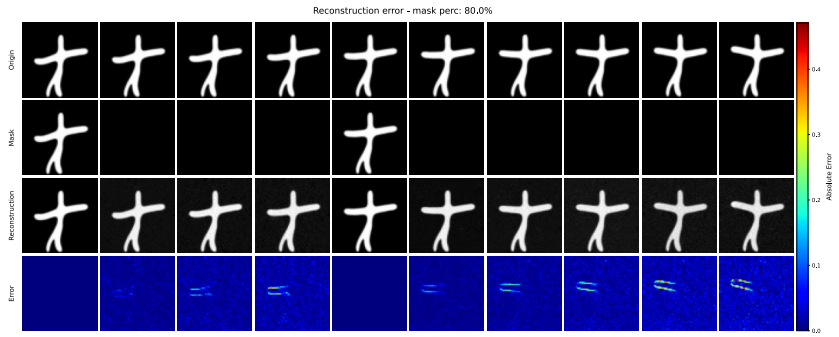
	Similarity metrics				Residual errors	
Δ_l gap group	SSIM ↑	MSSIM ↑	PSNR ↑	LPIPS (e-3) ↓	$l1$-error(e-3) ↓	$l2$-error(e-3) ↓
0-1	**84.882**	**99.158**	**38.266**	25.867	**9.001**	**0.151**
1-2	83.955	99.118	36.778	26.513	9.769	0.226
2-3	83.422	99.037	34.670	28.398	10.835	0.362
3-4	84.150	98.921	32.524	28.877	11.828	0.630
4-5	83.235	98.826	31.843	29.618	12.453	0.774
5-6	82.772	98.744	30.906	31.476	13.255	0.898
6-7	83.273	98.635	30.372	31.266	14.002	1.040
7-8	82.625	98.532	29.580	32.333	14.432	1.215
8-9	83.880	98.564	29.427	29.821	14.433	1.289
9-10	83.127	98.507	29.099	27.434	15.178	1.440
10-11	83.329	98.473	28.870	**25.325**	15.436	1.372
11-12	76.029	98.373	26.884	29.327	17.134	2.078

Table 6.3: Imputation errors: using random pairs with different time interval. Results are reported as the average for each time gap group. Best results are highlighted in **bold**

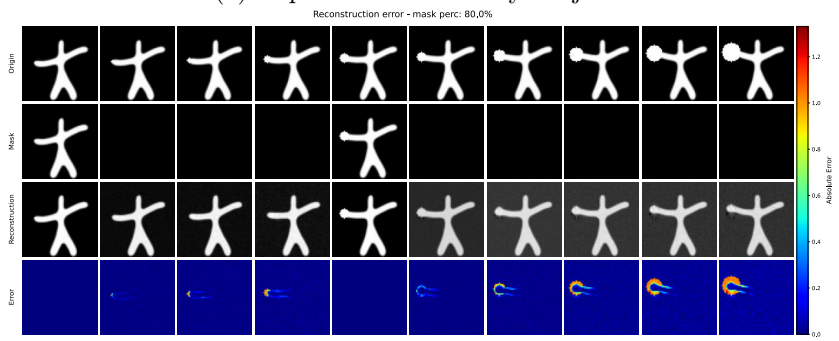
Fig. 6.3 shows examples of imputing all missing images based on observed data. TDM generates images in an autoregressive process. Starting from an observed time point, the model predicts the next succeeding image, and this newly generated image becomes the baseline for the following prediction. The process continues until the next observed time point.

6.3.3 Oversampling

To test the model capacity to generate new samples that follow the same trajectory of observed data, we oversample by using the last observation as condition and generate a number of follow-up images with time interval $\Delta_l = 1$. Fig. 6.4 shows an example of oversampling for healthy subject. All generated images here are completely new and not included in our dataset. From the residual error (compared to the last observation), we see that our model efficiently learn to recognize the moving part of the sequence (the left hand). For each image in the subsequence, it gradually moves this part higher to align with the subject's trajectory.



(a) Imputation for healthy subject.



(b) Imputation for anomaly subject.

Figure 6.3: Example of imputing missing data from TADM model. Fig. 6.3a shows example from healthy subject, Fig. 6.3b shows example for anomaly subject. From top to bottom of each image: original data, masked observed data, imputed data sequence, $l1$-error.

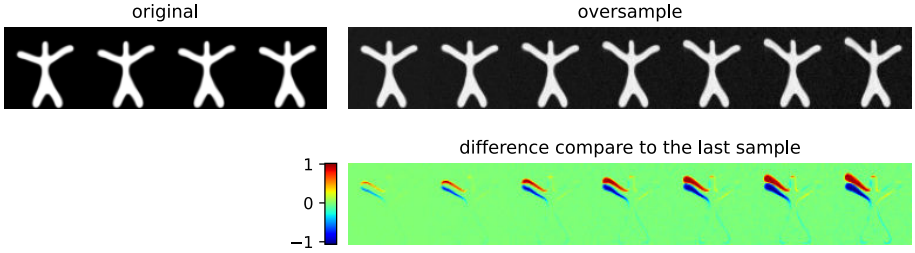


Figure 6.4: Example of oversampling with TDM.

Chapter 7

Conclusion

7.1 Summary

In this internship project, we studied the application of diffusion models for UAD, both in spatial and temporal contexts. Building on recent developments in diffusion models, we investigated the feasibility of addressing some of their critical limitations. First, we aimed to improve reconstruction quality by developing a mechanism that effectively conditions the diffusion process on the input image. To achieve this, we used a semantic encoder to extract a rich representation of the input, which is injected into our model using shift and scale operators. The benefits of our method are twofold. First, by avoiding direct concatenation of the input, we reduce the risk of data leakage, thereby preventing the model from simply memorizing the answer. Second, by employing the AdapGN mechanism, our model remains simpler than alternatives that rely on attention mechanisms or input concatenation. Finally, by choosing an appropriate noise level, our approach can correct anomalous regions while preserving the high-quality structure of healthy areas.

Next, we improved the performance of anomaly segmentation. Specially we focused on how to reduce false positive, which is a big challenge of reconstruction-based UAD methods. We demonstrated that classic quantile methods are not sufficient to deal with subtle anomalies, and their performance highly depends on the composition of the dataset. We introduced two modules: the feature attention module (FAM) and Longitudinal attention fusion module (LAFM) that align with UAD principles: (i) they do not require labeled or ground truth train dataset to perform greedy search for best threshold, and (ii) they are robust to different data compositions. FAM is presented to incorporate structural similarities to eliminate residual error caused by model error. Additionally, LAFM leverages temporal information from multiple time points to further smooth the residual maps. LAFM accounts for the temporal persistence of anomalies, enabling the detection of subtle anomalies at earlier stages, under the assumption that they will also be present in future observations. Our model shows promising results and outperforms all previous quantile methods. Furthermore, both FAM and LAFM exhibit more stable performance across datasets with varying proportions of healthy and anomalous samples.

For longitudinal learning, we presented a temporal diffusion model and its capacities to generate future images based on current observed data. Even though we did not exploit TDM in the context of anomaly detection, our model effectively captures the residual progression of healthy patients across arbitrary time intervals. This validates the ability of diffusion models to learn longitudinal progression, and it serves as an immediate step for future development.

7.2 Limitation and future work

Despite showing promising result, our approach has certain limitations that can be improved in future work. First, our proposed fusion modules (FAM and LAFM) do not form a unified framework. Instead, their performance varies across different test cases and metrics. For example, FAM outperforms the other in reducing false positive segmentation, while LAFM is more effective in accurately segmenting true anomaly. In the case of image wise anomaly score, results suggests that raw pixel distance with automatic Yen threshold is the most effective strategy. We can switch between them to achieve optimal results, or use one as an initial mask for other, but it introduces more complexity into the post processing pipeline. Furthermore, our model is heavily curated for our synthetic dataset, which may introduce bias and risk of overfitting. More extensive evaluation is needed to assess the generalization capabilities of our model.

A key aspect we wish to emphasize is the spatio-temporal modeling, as it is the main target of our project. Even though we utilize temporal information to increase accuracy, we note that this functions more as a filtering/smoothing operation than as a statistical model. While our approach offers the benefit of training free (LAF module), it has major limitation in capturing temporal variability (population level) and spatial inter-variability (personal level). It is important to note that, in our model, each patient is processed independently, and we do not make use of shared information between patients. Our project is inspired by LVAE model of Sauty et al. [10], in which they successfully impose mix effect model on the latent space of standard VAE. By combining longitudinal learning with generative learning, LVAE offers a more powerful solution for progression modeling. One notable example is that LVAE can propagate both to the past or future, while our TDM can only generate future images. Also, LVAE operates similarly to a Gaussian process, it makes use of all available data to impute missing data, but our model currently can only take the most recent image as condition. We aim to apply the same principle of LVAE to diffusion model. There are other models motivated by the same principle. Chen et al. [34] use orthogonal linear transformation to disentangle global and individual trajectory, also apply to VAE model. [35] use normalizing flow to model temporal dependencies. The main challenge for such adaptation is that in VAE, each sample corresponds to only one (normally non-spatial, low dimension) latent variable, while in DMs, each sample (at one time point) has a sequence of spatial latent variables, and each latent variable is of same resolution as input. One avenue for future work is to view DMs as a Hierarchical Variational Autoencoder (HVAE) [19], and impose longitudinal structure for every image at each noise level. It is apparent that this will increase the complexity of the model, and require large number of epochs to train (one full epoch requires the model to see all samples with all noise steps). Nevertheless, it can serve as a starting point for further improvements.

Another direction we want to explore is to exploit the stochastic subcodes obtained from the reversed DDIM sampling scheme, which we have shown to contain fine-grained details of the original inputs and to deviate from a strict normal distribution. It would be interesting to investigate whether the divergence between these distributions and a standard Gaussian (e.g., measured via KL divergence) can be leveraged for anomaly detection. One thing to note is that in principle, this is very similar to score-based UAD [25, 5], which is another family of UAD with diffusion models.

We conduct our experiment on synthetic 2D dataset at a low resolution, so the results can be very different from real world 3D MRI scans, specially with anomalies that are very subtle. We aim to adapt and evaluate our model on PPMI dataset [36]. To effectively transfer our model to PPMI dataset, one common approach is to apply 2D diffusion models slice-wise on 3D scans, and then concatenate results together to reconstruct 3D volumes. This comes with limitation of missing 3D context and spatial relationships between slices. Another approach is to compress

original 3D input into latent variable [20, 37, 15], and then train DMs on this latent variable. To best preserve the signal from original 3D volume, the most effective approach would be to implement 3D diffusion models. We can explore the option of training only on patches to address the problem of high memory requirements and computational.

Bibliography

[1] Ioannis Lagogiannis et al. "Unsupervised Pathology Detection: A Deep Dive Into the State of the Art". In: *IEEE Transactions on Medical Imaging* 43.1 (Jan. 2024), pp. 241–252. ISSN: 1558-254X. DOI: 10.1109/TMI.2023.3298093. URL: https://ieeexplore.ieee.org/document/10197302 (cit. on p. 1).

[2] Cosmin I. Bercea et al. *Denoising Diffusion Models for Anomaly Localization in Medical Images*. Oct. 31, 2024. DOI: 10.48550/arXiv.2410.23834. arXiv: 2410.23834 [eess]. URL: http://arxiv.org/abs/2410.23834. Pre-published (cit. on pp. 2, 8, 21).

[3] David Zimmerer et al. *Unsupervised Anomaly Localization using Variational Auto-Encoders*. 2019. arXiv: 1907.02796 [cs.LG]. URL: https://arxiv.org/abs/1907.02796 (cit. on p. 2).

[4] Thomas Schlegl et al. "F-AnoGAN: Fast Unsupervised Anomaly Detection with Generative Adversarial Networks". In: *Medical Image Analysis* 54 (2019), pp. 30–44. ISSN: 1361-8415. DOI: 10.1016/j.media.2019.01.010 (cit. on p. 2).

[5] Walter H. L. Pinaya et al. *Fast Unsupervised Brain Anomaly Detection and Segmentation with Diffusion Models*. 2022. arXiv: 2206.03461 [cs.CV]. URL: https://arxiv.org/abs/2206.03461 (cit. on pp. 2, 44).

[6] Julia Wolleb et al. *Diffusion Models for Medical Anomaly Detection*. 2022. arXiv: 2203.04306 [eess.IV]. URL: https://arxiv.org/abs/2203.04306 (cit. on pp. 2, 7).

[7] Finn Behrendt et al. "Guided Reconstruction with Conditioned Diffusion Models for Unsupervised Anomaly Detection in Brain MRIs". In: *Computers in Biology and Medicine* 186 (Mar. 2025), p. 109660. ISSN: 0010-4825. DOI: 10.1016/j.compbiomed.2025.109660 (cit. on pp. 2, 8, 11, 16, 20, 21, 26).

[8] Cosmin I. Bercea et al. *Mask, Stitch, and Re-Sample: Enhancing Robustness and Generalizability in Anomaly Detection through Automatic Diffusion Models*. May 2023. DOI: 10.48550/arXiv.2305.19643. arXiv: 2305.19643 [cs] (cit. on pp. 2, 16, 25).

[9] Yiyuan Yang et al. *A Survey on Diffusion Models for Time Series and Spatio-Temporal Data*. June 11, 2024. DOI: 10.48550/arXiv.2404.18886. arXiv: 2404.18886 [cs]. URL: http://arxiv.org/abs/2404.18886. Pre-published (cit. on p. 3).

[10] Benoît Sauty and Stanley Durrleman. "Progression Models for Imaging Data with Longitudinal Variational Auto Encoders". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part I*. Singapore, Singapore: Springer-Verlag, 2022, pp. 3–13. DOI: 10.1007/978-3-031-16431-6_1. URL: https://doi.org/10.1007/978-3-031-16431-6_1 (cit. on pp. 3, 44).

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. *Denoising Diffusion Probabilistic Models*. Dec. 2020. DOI: 10.48550/arXiv.2006.11239. arXiv: 2006.11239 [cs] (cit. on pp. 5, 6).

[12] Julio A. Soto. "Appendices for lectures on diffusion models". In: (Nov. 2024). URL: https://julioasotodv.github.io/ie-c4-466671-diffusion-models/Appendices%20for%20lectures%20on%20diffusion%20models.html (cit. on pp. 6, 7).

[13] Jiaming Song, Chenlin Meng, and Stefano Ermon. *Denoising Diffusion Implicit Models*. Oct. 5, 2022. DOI: 10.48550/arXiv.2010.02502. arXiv: 2010.02502 [cs]. URL: http://arxiv.org/abs/2010.02502. Pre-published (cit. on pp. 6, 7, 18).

[14] Konpat Preechakul et al. *Diffusion Autoencoders: Toward a Meaningful and Decodable Representation*. Mar. 10, 2022. DOI: 10.48550/arXiv.2111.15640. arXiv: 2111.15640 [cs]. URL: http://arxiv.org/abs/2111.15640. Pre-published (cit. on pp. 7, 12, 23).

[15] Gabriele Lozupone et al. *Latent Diffusion Autoencoders: Toward Efficient and Meaningful Unsupervised Representation Learning in Medical Imaging*. Apr. 11, 2025. DOI: 10.48550/arXiv.2504.08635. arXiv: 2504.08635 [cs]. URL: http://arxiv.org/abs/2504.08635. Pre-published (cit. on pp. 7, 11, 12, 18–20, 23, 36, 45).

[16] Prafulla Dhariwal and Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*. June 1, 2021. DOI: 10.48550/arXiv.2105.05233. arXiv: 2105.05233 [cs]. URL: http://arxiv.org/abs/2105.05233. Pre-published (cit. on pp. 7, 11).

[17] Yang Song et al. *Score-Based Generative Modeling through Stochastic Differential Equations*. Feb. 10, 2021. DOI: 10.48550/arXiv.2011.13456. arXiv: 2011.13456 [cs]. URL: http://arxiv.org/abs/2011.13456. Pre-published (cit. on p. 8).

[18] Arian Mousakhan, Thomas Brox, and Jawad Tayyub. "Anomaly Detection with Conditioned Denoising Diffusion Models". In: *Pattern Recognition*. Springer Nature Switzerland, 2025, pp. 181–195. ISBN: 9783031851810. DOI: 10.1007/978-3-031-85181-0_12. URL: http://dx.doi.org/10.1007/978-3-031-85181-0_12 (cit. on pp. 8, 12, 13, 16, 21).

[19] Calvin Luo. *Understanding Diffusion Models: A Unified Perspective*. Aug. 25, 2022. DOI: 10.48550/arXiv.2208.11970. arXiv: 2208.11970 [cs]. URL: http://arxiv.org/abs/2208.11970. Pre-published (cit. on pp. 8, 44).

[20] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. arXiv: 2112.10752 [cs.CV]. URL: https://arxiv.org/abs/2112.10752 (cit. on pp. 8, 11, 19, 20, 45).

[21] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV] (cit. on pp. 11, 19, 20).

[22] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762 (cit. on pp. 11, 18).

[23] Richard Zhang et al. *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*. 2018. arXiv: 1801.03924 [cs.CV]. URL: https://arxiv.org/abs/1801.03924 (cit. on p. 12).

[24] Karsten Roth et al. *Towards Total Recall in Industrial Anomaly Detection*. 2022. arXiv: 2106.08265 [cs.CV]. URL: https://arxiv.org/abs/2106.08265 (cit. on p. 13).

[25] Jiazheng Wang et al. "EPDiff: Erasure Perception Diffusion Model for Unsupervised Anomaly Detection in Preoperative Multimodal Images". In: *IEEE Transactions on Medical Imaging* (2025), pp. 1–1. ISSN: 1558-254X. DOI: 10.1109/TMI.2025.3597545. URL: https://ieeexplore.ieee.org/document/11121881/ (cit. on pp. 14–16, 21, 44).

[26] C. Baur et al. "Autoencoders for unsupervised anomaly segmentation in brain MR images: A comparative study". In: *Medical Image Analysis* 69 (2021), p. 101952 (cit. on p. 16).

[27] Jui-Cheng Yen, Fu-Juay Chang, and Shyang Chang. "A new criterion for automatic multilevel thresholding". In: *IEEE Transactions on Image Processing* 4.3 (1995), pp. 370–378. DOI: 10.1109/83.366472 (cit. on p. 16).

[28] Ioannis Lagogiannis et al. "Unsupervised Pathology Detection: A Deep Dive Into the State of the Art". In: *IEEE Transactions on Medical Imaging* 43.1 (Jan. 2024), pp. 241–252. ISSN: 1558-254X. DOI: 10.1109/TMI.2023.3298093. URL: https://ieeexplore.ieee.org/document/10197302 (cit. on p. 16).

[29] Di Wu et al. *Unsupervised Anomaly Detection via Masked Diffusion Posterior Sampling*. Apr. 27, 2024. DOI: 10.48550/arXiv.2404.17900. arXiv: 2404.17900 [cs]. URL: http://arxiv.org/abs/2404.17900. Pre-published (cit. on p. 16).

[30] Alexandre Bône, Olivier Colliot, and Stanley Durrleman. *Learning distributions of shape trajectories from longitudinal datasets: a hierarchical model on a manifold of diffeomorphisms*. 2018. arXiv: 1803.10119 [cs.CV]. URL: https://arxiv.org/abs/1803.10119 (cit. on p. 17).

[31] Di Wu et al. *Unsupervised Anomaly Detection via Masked Diffusion Posterior Sampling*. 2024. arXiv: 2404.17900 [cs.CV]. URL: https://arxiv.org/abs/2404.17900 (cit. on p. 21).

[32] Mattia Litrico et al. *TADM: Temporally-Aware Diffusion Model for Neurodegenerative Progression on Brain MRI*. June 18, 2024. DOI: 10.48550/arXiv.2406.12411. arXiv: 2406.12411 [eess]. URL: http://arxiv.org/abs/2406.12411. Pre-published (cit. on pp. 36–38).

[33] Yulun Zhang et al. *Residual Dense Network for Image Super-Resolution*. 2018. arXiv: 1802.08797 [cs.CV]. URL: https://arxiv.org/abs/1802.08797 (cit. on pp. 37–39).

[34] Ming Chen et al. "Orthogonal Mixed-Effects Modeling for High-Dimensional Longitudinal Data: An Unsupervised Learning Approach". In: *IEEE Transactions on Medical Imaging* 44.1 (Jan. 2025), pp. 207–220. ISSN: 1558-254X. DOI: 10.1109/TMI.2024.3435855. URL: https://ieeexplore.ieee.org/document/10616022 (cit. on p. 44).

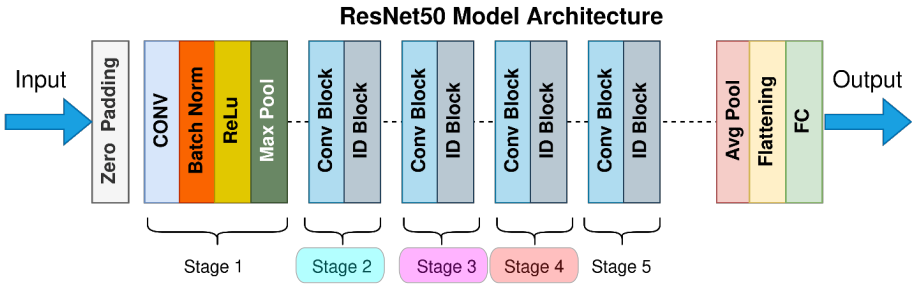
[35] Clément Chadebec and Stéphanie Allassonnière. *Variational Inference for Longitudinal Data Using Normalizing Flows*. Mar. 24, 2023. DOI: 10.48550/arXiv.2303.14220. arXiv: 2303.14220 [stat]. URL: http://arxiv.org/abs/2303.14220 (cit. on p. 44).

[36] Kenneth Marek et al. "The Parkinson's Progression Markers Initiative (PPMI) – Establishing a PD Biomarker Cohort". In: *Annals of Clinical and Translational Neurology* 5.12 (Dec. 2018), pp. 1460–1477. ISSN: 2328-9503, 2328-9503. DOI: 10.1002/acn3.644. URL: https://onlinelibrary.wiley.com/doi/10.1002/acn3.644 (cit. on p. 44).

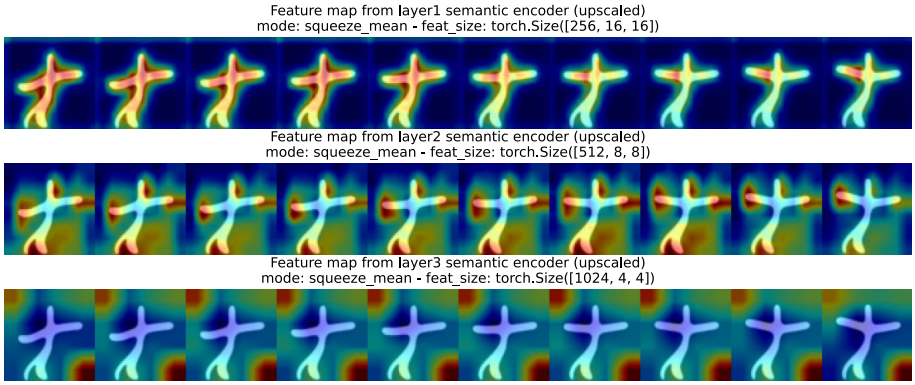
[37] Lemuel Puglisi, Daniel C Alexander, and Daniele Ravi. "Enhancing spatiotemporal disease progression models via latent diffusion and prior knowledge". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2024, pp. 173–183 (cit. on p. 45).

Appendix A

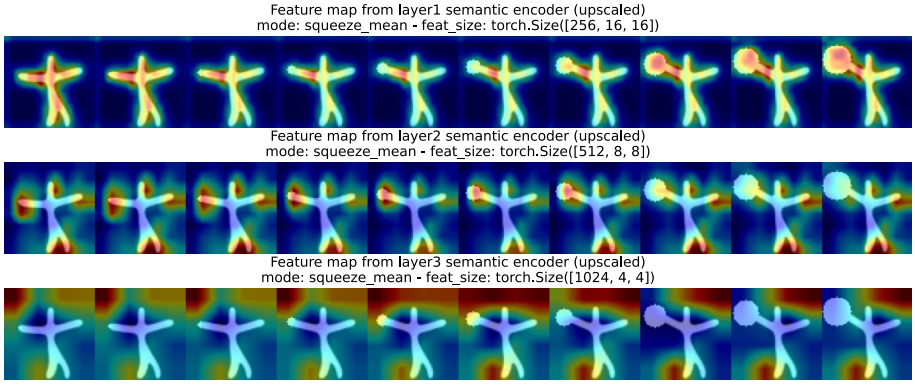
Feature Extractor Network



(a) ResNet-50 architecture



(b) Healthy subject



(c) Anomaly subject

Figure A.1: Example outputs from layers of Feature Extractor network Φ in SDM. Fig. A.1a shows ResNet-50 architecture. Fig. A.1b and Fig. A.1c display example of outputs for healthy and anomalous subject, respectively. Notation: `stage2`, `stage3` and `stage4` from ResNet architecture correspond to `layer1`, `layer2` and `layer3` in our paper, respectively.

Appendix B

More examples of LAFM Anomaly score maps

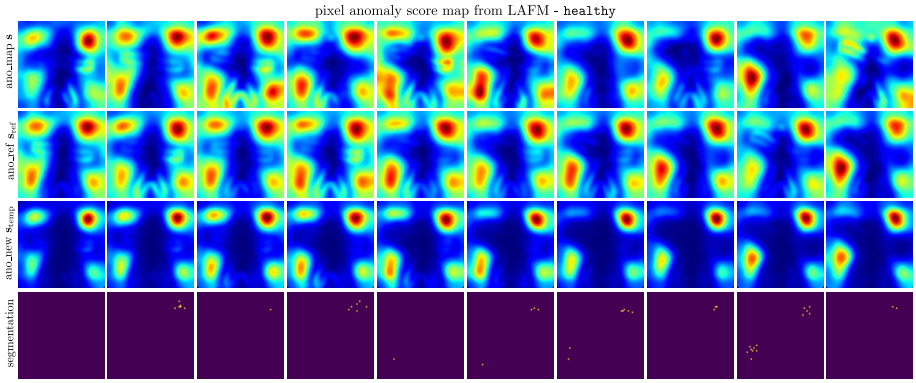


Figure B.1: Example of anomaly score map from LAFM for healthy subject. From top to bottom: input score map (from FAM), anomaly map references, updated score map (LAFM), anomaly segmentation (Yen threshold).

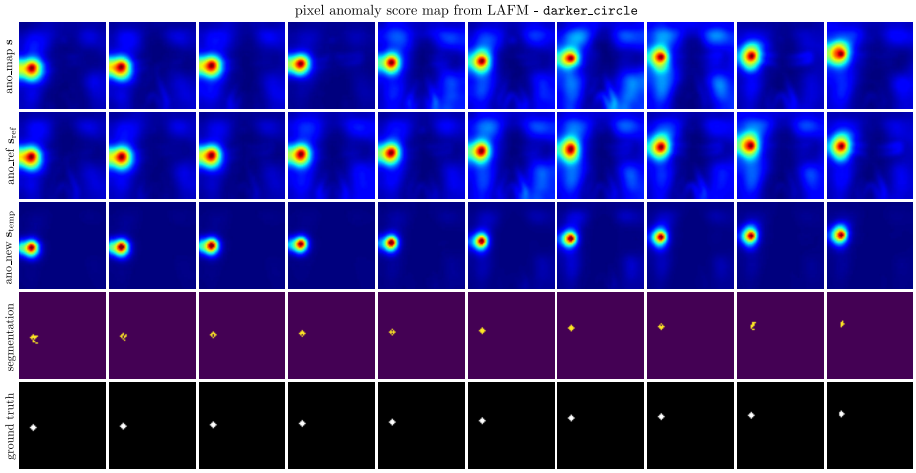


Figure B.2: Example of anomaly score map from LAFM for anomaly `darker_circle` subject. From top to bottom: input score map (from FAM), anomaly map references, updated score map (LAFM), anomaly segmentation (Yen threshold) and ground truth annotation.

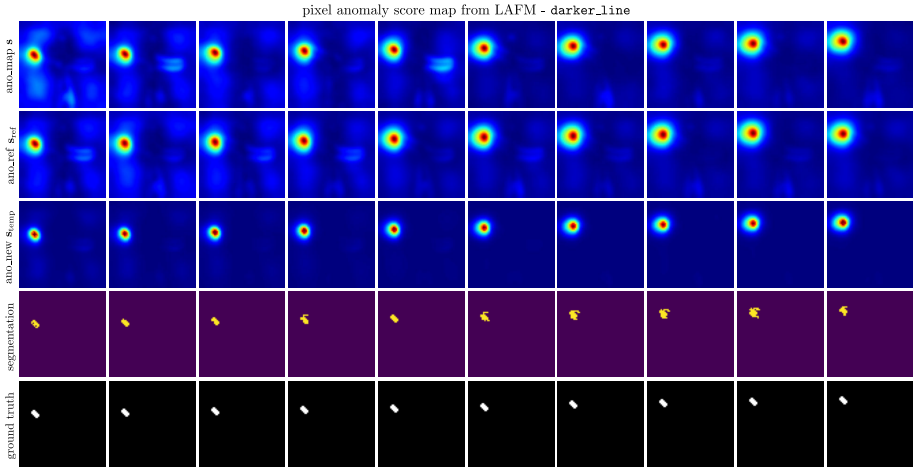


Figure B.3: Example of anomaly score map from LAFM for anomaly `darker_line` subject. From top to bottom: input score map (from FAM), anomaly map references, updated score map (LAFM), anomaly segmentation (Yen threshold) and ground truth annotation.