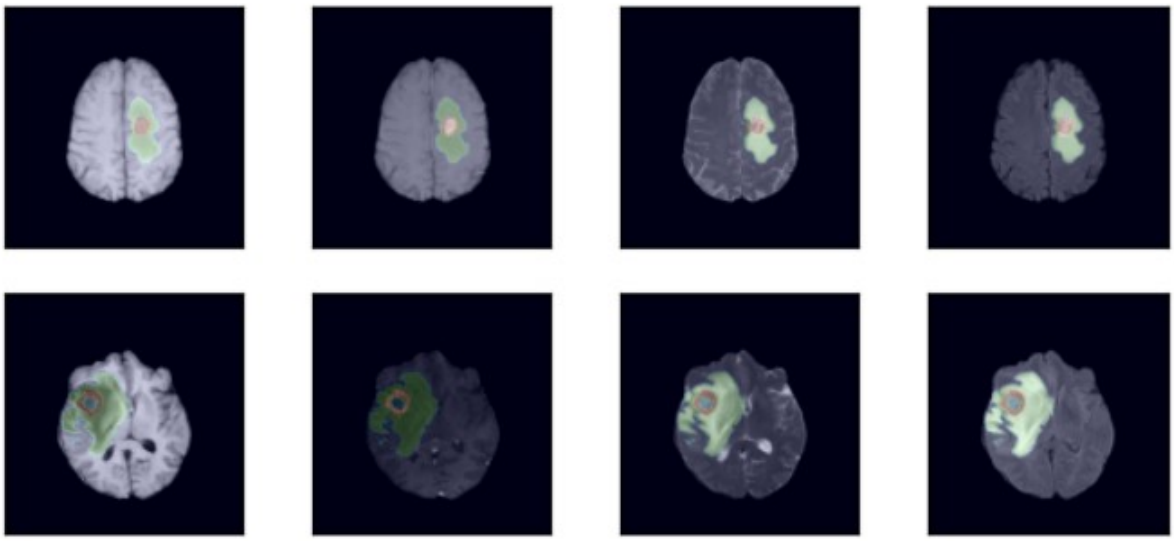


Generate Realistic Image Segmentation Pair Using Diffusion Probabilistic Model

A Data Augmentation Approach for Biomedical Segmentation

Master's thesis in Biomedical Engineering

WENYIN ZHOU

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis 2023

Generate Realistic Image and Segmentation Pair using Diffusion Probabilistic Model

A Data Augmentation Approach for Biomedical Segmentation

WENYIN ZHOU

CHALMERS
UNIVERSITY OF TECHNOLOGY

A thesis
for the degree of
Master of Science in Biomedical Engineering
Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering
Computer Vision and Medical Image Analysis Group
Chalmers University of Technology
Gothenburg, Sweden 2023

Generate Realistic Image Segmentation Pair using Diffusion Probabilistic Model
A Data Augmentation Approach for Biomedical Segmentation
Wenyin Zhou

Supervisor: Roman Naeem, Department of Electrical Engineering
Examiner: Fredrik Kahl, Department of Electrical Engineering

Master's Thesis 2023
Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering
Computer Vision and Medical Image Analysis Group
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Generate Realistic Image Segmentation Pair using Diffusion Probabilistic Model.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2023

Declaration

I affirm that I am the sole author of this thesis and that it has not been previously submitted, either in its entirety or in part, for any other degree application.

Abstract

Diffusion Models (DMs) [22, 54, 55] are newly emerged deep generative models which produce data via progressive denoising and have achieved state-of-art results for image synthesis. Medical image segmentation, a dense prediction task that assigns a pixel-wise classification to the medical images, severely lacks data to achieve great performance in the context of deep learning. In this thesis, we seek to improve the segmentation model using generative modeling, particularly the DMs. We systematically review the DMs from the perspective of variational inference and several key neural network architectures for both the DMs and biomedical segmentation. Based upon the classical DMs, we propose a training scheme of joint modeling for paired data generation and conditional sampling. To that end, the Joint Diffusion Model (JDM) learns mutual dependence among multimodality data and is orthogonal to various accelerating sampling methods and inverse problem solvers using the DMs. The qualitative results present the great visual quality and diversity of the generated samples, showcasing the capability of the JDM in modeling the joint distribution even with a limited number of data samples. The quantitative results suggest that the appropriate use of the synthetic data can improve the discriminative model in recognizing the fine-grained image features, leading to a performance increase for biomedical segmentation. More specifically, we first pretrain the model on synthetic data using supervised learning and then finetune the model on the real data. The number of synthetic data is ablated. The experiments are conducted on five publicly available polyp segmentation and one nuclei segmentation datasets, and show on average +2.5% increase in terms of mean dice score on polyps segmentation and set the new state-of-art results on MoNuSeg [31] using nnU-Net [25].

Keywords: generative model, diffusion model, medical image segmentation, data augmentation.

Acknowledgements

First and foremost, I want to deliver thanks to my parents for their eternal love and permanent support for my cause.

I want to thank my supervisor Roman Naeem for his great guidance and introduction to this interesting topic. This work is a long journey starting from the discussion in October 2022 and is impossible without his preconditioning, patience and devotion.

I want to thank David Hagerman Olzon, Professor Lennart Svensson, and Professor Fredrik Kahl for the useful discussions.

Wenyin Zhou, Gothenburg, July 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

CDM	Conditional Diffusion Model
CNN	Convolutional Neural Network
DMs	Diffusion Models
DDPM	Denoising Diffusion Probabilistic Model
FID	Fréchet Inception Distance
GANs	Generative Adversarial Networks
JDM	Joint Diffusion Model
mDice	Mean Dice Coefficient
mIoU	Mean Intersection over Union
KL	Kullbeck Leibler
VAEs	Variational Autoencoders

Contents

List of Acronyms viii

List of Figures xi

List of Tables xv

1 Introduction 1
- 1.1 Main Contribution . . . 4
- 1.2 Outline . . . 4

2 Background 6
- 2.1 Diffusion Models . . . 6
  - 2.1.1 Generative Modeling and Diffusion Process . . . 7
  - 2.1.2 Training Objectives . . . 10
  - 2.1.3 Simplified Training Objective . . . 10
  - 2.1.4 Learned Variance . . . 12
  - 2.1.5 Reverse Sampling . . . 13
  - 2.1.6 Diffusion Network . . . 13
- 2.2 Posterior sampling using Diffusion Models . . . 15
  - 2.2.1 Conditional Sampling . . . 15
  - 2.2.2 Conditional Model . . . 16
- 2.3 Diffusion Model for Zero-shot Data Editing . . . 16
- 2.4 Medical Segmentation Network . . . 18
  - 2.4.1 U-Net . . . 19
  - 2.4.2 nnU-Net . . . 20

3 Joint Diffusion Model for Paired Data Synthesis 23
- 3.1 Joint Diffusion Model . . . 23
- 3.2 Conditional Sampling . . . 24

4 Experiments 27

4.1 Experimental Setups . . . 27
4.2 Datasets . . . 28
4.3 Hyperparameters . . . 29
4.3.1 Joint Diffusion Model (JDM) . . . 29
4.4 Metrics . . . 30
4.5 Baselines . . . 31

5 Results and Discussion **33**
5.1 Visual Comparison . . . 33
5.1.1 Pair Data Synthesis . . . 33
5.1.2 Mask Conditioned Synthesis . . . 35
5.1.3 Discussion . . . 35
5.2 Medical Segmentation on Synthetic Data . . . 39
5.2.1 Medical Segmentation on Polyps . . . 39
5.2.2 Medical Segmentation on Nuclei . . . 42
5.2.3 Discussion . . . 43
5.3 Transfer Learning on Synthetic Data . . . 45
5.3.1 Transfer Learning on Polyps . . . 45
5.3.2 Transfer Learning on Nuclei . . . 46
5.3.3 Discussion . . . 49

6 Conclusion and Limitations **50**
6.0.1 Limitations and Future Work . . . 51

Bibliography **53**

A Appendix 1
Properties of the Diffusion Model **61**
A.1 Proofs of Equation (2.4) and (2.5) . . . 61
A.2 A Proof of Equation (2.6) . . . 62
A.3 A proof of Equation (2.9) . . . 63
A.4 A proof of Equation (2.10) . . . 64
A.5 Variational bound . . . 65

B Appendix 2
More Generated Samples **67**

List of Figures

1.1 Medical Image Segmentation with Swin UNETR [19] on BRATS2021 [2]. The segmentation network produces segmentation masks for the Enhancing Tumor, Whole Tumor and Tumor Core. From left to right: T1, T1c, T2 and FLAIR MRI axial slice. Image from [19]. 2

2.1 The directed graphical model of DDPM. The latent variables along the sampling path share the same dimension and are hypothesized to be Markovian. The noise is gradually added to the data in the forward diffusion process, while the reverse process recovers the lost semantic information and pushes one step closer to the clean data once the network evaluation. Demo image from [22]. 8

2.2 The Residual block used in the diffusion models. 14

2.3 The overall network design of the diffusion network. RB: residual block, Attn: efficient self-attention block. The residual Block is the basic unit for image downsampling and upsampling with the skip connection introduced. See text for more description. Efficient self-attention block is applied at the deepest level and improve feature extraction via QK product with linear complexity. We refer the reader to [48] for more details about efficient attention. 15

2.4 SegDiff [1]: a diffusion-based segmentation method. The two separate encoders (G & F) are employed for denoising noisy segmentation mask the extracting features of the image. The encodings are integrated by the direct sum in the latent space. 17

2.5 Qualitative results on ImageNet 512×512 for thin mask from Repaint [35]. 18

2.6 The U-Net Architecture: A powerful deep learning framework for semantic segmentation tasks in medical imaging. It comprises a U-shaped network structure with both contracting and expanding paths. The contracting path consists of convolutional and pooling layers to capture context, while the expanding path uses transposed convolutions for precise localization. Skip connections facilitate the integration of multi-scale features, enabling different size of receptive fields. . 19
2.7 nnU-Net's interdependence among the hyperparameter setting, data pre-post processing, and model selection. 21

3.1 The gradual denoising procedure of the Joint Diffusion Model (JDM). A paired sample of medical image and segmentation mask is produced with a single reverse run. 26

5.1 Random generated samples of nuclei and polyp images and their associated segmentation masks. All the images and masks are resized to 256 for display. 36
5.2 Random generated samples and closest train data regarding mutual information concerning segmentation mask. From left to right: synthetic image, associated synthetic segmentation mask, closest real image, closest real segmentation mask. 37
5.3 Synthetic medical images guided by the training mask using the joint diffusion model. The significant difference in contrast to the real image demonstrates the sample diversity and model distribution. . . . 38
5.4 mDice vs. the number of synthetic data. The dashed line denotes the result of training on real data. From left to right and top to down: Kvasir, Clinic-DB, Colon-DB, CVC-300, ETIS. 41
5.5 mDice vs. the number of synthetic data on MoNuSeg. The dashed line denotes the result of training on real data. 44
5.6 Quliative results. The red box highlights the improvement of the pre-trained segmentation model on synthetic data. The results are taken from the best case on ETIS [49] and MoNuSeg [31]. 47

A.1 Variational Bound vs. diffusion steps. Image from [41] 66

B.1 Kvasir generated samples. 68
B.2 Glas generated samples. 69
B.3 MoNuSeg generated samples. 70
B.4 MoNuSeg generated samples. 71
B.5 MoNuSeg generated samples. 72
B.6 Kvasir mask conditioned generated samples. From top to bottom: real image, real segmentation mask, four generated samples. 73

B.7 MoNuSeg mask conditioned generated samples. From top to bottom: real image, real segmentation mask, four generated samples 74

List of Tables

4.1 The hyperparameter setting of the JDM. (*) The number of residual blocks is set to 2 and 3 for Kvasir and MoNuSeg, respectively. 30

5.1 Results on validation sets of polyp segmentation. (# Synthetic - %) represents the number of synthetic data used in training the model. The first and second best results are in **bold** and underlined, respectively. The number in blue quantifies the performance difference in contrast to the result on real data. 40

5.2 Results on generalization sets of polyp segmentation. (# Synthetic - %) represents the number of synthetic data used in training the model. The first and second best results are in **bold** and underlined, respectively. The number in blue quantifies the performance difference compared to the result on real data. 42

5.3 Results on MoNuSeg of nuclei segmentation. (# Synthetic - %) represents the number of synthetic data used in training the model. The first and second best results are in **bold** and underlined, respectively. The number in blue quantifies the performance difference in contrast to the result on real data. 43

5.4 Results on the validation set pretrain on a different number of synthetic data. (# Syn. Pretrain) represents the number of synthetic data used pretraining the model. The first and second best results are in **bold** and underlined, respectively. The number in blue quantifies the performance difference in contrast to the result on real data. . . . 46

5.5 Results on the generalization set pretrained on a different number of synthetic data. (# Syn. Pretrain) represents the number of synthetic data used pretraining the model. The first and second best results are in **bold** and underlined, respectively. The number in blue quantifies the performance difference in contrast to the result on real data. . . . 47

5.6 Results on MoNuSeg [31]. (# Syn. Pretrain) represents the number of synthetic data used pretraining the model. The first and second best results are in **bold** and underlined, respectively. The number in blue quantifies the performance difference in contrast to the result on real data. 48

1

Introduction

Medical Image Segmentation is a discipline that partitions a medical image into multiple, non-overlapping segments, each of which represents a specific anatomical structure or tissue type (See Fig. 1.1). This process plays an important role in modern healthcare and is embedded into many clinical decision workflows, such as diagnosis, treatment, image-guided surgery, etc. Automated segmentation aims to provide a segmentation solution to medical images by designing computer or machine learning algorithms[1] without significant manual efforts. Compared with manual segmentation, the automated counterpart has the advantage of prediction speed, which might potentially alleviate the heaven burden of medical experts and improve diagnostic efficiency.

With the availability of a large amount of data, advanced network architectures, the increasing computing power of the graphical processing unit, and many other important reasons, deep learning [17] has become the de facto choice of machine learning methods in the last decade. In deep learning, neural networks are designed with multiple layers of interconnected nodes, known as artificial neurons or units. Each layer processes and transforms the input data and passes it on to the next layer, gradually extracting higher-level features and representations. These interconnected nodes usually referred to as parameters are updated through stochastic gradient descent. It has been demonstrated that neural networks such as convolutional neural

[1] In this thesis, we refer to machine learning algorithms as a mathematical model or computer procedure to learn the pattern of the input data and make predictions without explicit programming [36].

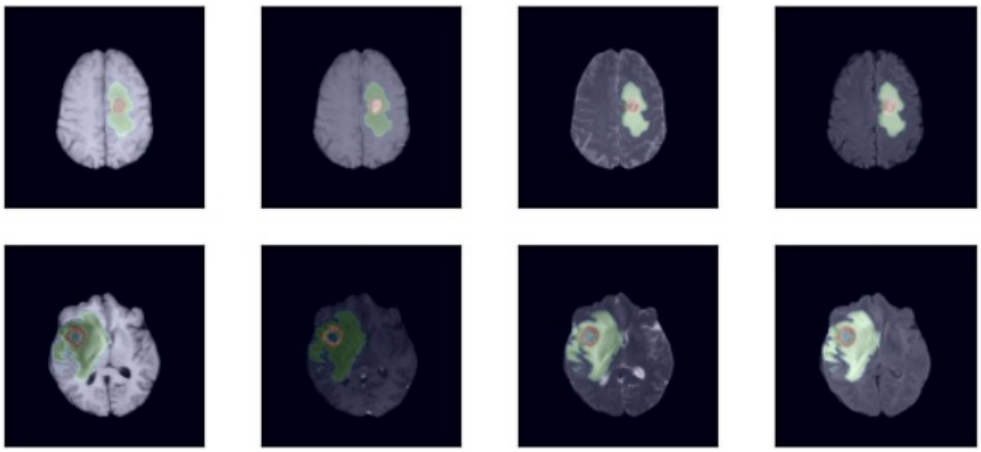


Figure 1.1: Medical Image Segmentation with Swin UNETR [19] on BRATS2021 [2]. The segmentation network produces segmentation masks for the Enhancing Tumor, Whole Tumor and Tumor Core. From left to right: T1, T1c, T2 and FLAIR MRI axial slice. Image from [19].

networks[2] (CNN) can model complex representations for image classification [20], object detection [6], semantic segmentation [7], etc.

As for automated medical segmentation, most developed methods rely on supervised learning based on deep learning, in which both the data point and label are accessible during model training. Empirically, the deep learning based approaches heavily demand a huge amount of heterogeneous data to obtain generalizable results. However, in contrast to the natural image, sizable medical imaging datasets are more difficult to collect, owing to the specialized medical equipment, the requirement of medical experts' devotion, strict regulation of patient data privacy, and many other reasons preventing active data sharing across multiple institutes.

In this thesis, we consider approaching this issue from the perspective of data augmentation using generative modeling. Data augmentation is a common tactic to improve the generalization skill of the discriminative model. In practice, it can be accomplished by simple transformations (e.g. rotations or random flips) or synthetic images from the generative model. While simple transformations can be effective in most scenarios, it is limited in generating diverse and realistic samples without modeling data distribution. On the other hand, based on the observations from the

[2]In this work, CNN is referred to as a type of deep learning neural network designed for visual data analysis, usually utilizing convolutional layers to extract features and pooling layers to downsample the data.

training set, a generative model can be constructed to directly approximate either the underlying data distribution p_{data} or the data generation process, from which the novel data point can be produced. Two well-known representatives are likelihood based approaches [16, 59, 13, 39] and generative adversarial networks (GANs) [18].

GANs support efficient sampling and high sample quality with carefully designed architectures, but training GANs can be unstable and sometimes result in training failure. It can be largely attributed to the imbalance between the generator and the discriminator. For example, in the case of the discriminator being much better than the generator, the generator might not make progress by fooling the discriminator. Besides, the mode covering issues and demand for a multitude of data are also major drawbacks of GANs, preventing them from broader applications. As suggested by [51], GANs' effectiveness as a source of medical imaging data was found to not always be reliable, even if the generated images are nearly indistinguishable from real data.

Models that have a tractable likelihood, such as autoregressive models [16, 59] or flow-based models [13, 39], give competitive likelihoods, potentially leading to great diversity. However, these likelihood based approaches have limitations based on certain assumptions of the model distribution. For instance, autoregressive models assume the factorization of multiple conditional distributions, and flow-based models interpret the generation process as an invertible transformation from a base distribution to data density. Other cases such as Variational Autoencoders (VAEs) [28] having incomplete likelihoods assumes the data is projected to a tractable distribution in the latent space, and the model is trained for projection and recovery. These imposed restrictions about the model distribution can be unnatural and limit the capability of neural networks, which might partially explain why these methods have not established the same sample quality as GANs.

Recently, a newly emerged class of generative model called the diffusion models (DMs) or score-based generative models [52, 22, 54, 55] has attracted much interest in image synthesis and inverse problem solving. The produced images of the DMs achieve great sample quality and diversity on several benchmarks, such as CIFAR-10 [29], and Celeb-A [33]. Owing to its substantial generation capability, the DMs may potentially serve as an efficacious tool for medical images, a domain severely suffering from data sparsity problems. It may also promote active data sharing across the institutes and improve the model performance on downstream tasks. Furthermore, since the DMs also support the exact likelihood computation, it may help to enhance the explainability in the context of clinical practice.

In this thesis, we study the most prevailing instance of the DMs, denoising diffusion probabilistic model (DDPM) [22] as an out-of-box tool for 2d biomedical segmentation. Applying DDPM to the medical images, one can generate novel data samples that might fill in the gap of the training data manifold.

1.1 Main Contribution

The main contribution of this thesis is to give a thorough review of DMs for practitioners to apply this technique to medical problems. The technical novelty is summarized as follows:

1. We propose a new training scheme of joint modeling for the DMs with paired data other than condition modeling. We prove that joint modeling implicitly supports conditional sampling and is orthogonal to various inverse problem solvers.

2. We show that applying the Joint Diffusion Model (JDM) to 2d biomedical segmentation is capable of generating meaningful and unlimited number of paired samples. The generated samples deviate from the training data significantly.

3. Empirical results on supervised learning and transfer learning suggest that the generated samples are useful for improving the segmentation model even if the training set is severely sparse.

1.2 Outline

In Chapter 2, we review the background that is concerned with this thesis, encompassing the DMs and medical segmentation network.

In Chapter 3, we elaborate on how to construct a joint diffusion probabilistic model to generate the paired data. This chapter also includes conditional sampling through joint modeling.

In Chapter 4, we present the experimental setups of this work, including the working environment, hyperparameter setting, performance metrics, and compared baselines.

In Chapter 5, we visually check the generated samples and investigate the effect of data augmentation using synthetic data. This encompasses evaluating the segmentation model trained on purely synthetic data, and pretrained on synthetic data

then finetuned on the real data.

2

Background

2.1 Diffusion Models

Diffusion Models (DMs) are newly emerged deep generative models [22, 52, 54, 55], which produce sample by reversing the data corruption process via iterative refinement. DMs have seen wide applications in image [54, 55, 22, 45, 41, 14], video [32, 8], music [37], molecular synthesis [65], and various image-to-image translation tasks such as super-resolution [46], inpainting [55, 35, 10], etc. In the forward diffusion process, the noise is gradually injected into the data, transforming the complex data distribution into a tractable prior distribution. As an adequate amount of noise is added, the perturbed data can be equivalently considered as a sample from a prior distribution (e.g. Gaussian distribution) that is easy to sample from. For reverse sampling, one can train a deep neural network[1] [17] that progressively eliminates noise from the Gaussian sample to produce data.

Since the DMs can be further generalized to be continuous [55], there are extensive design options available for the DMs. This work primarily concentrates on the DDPM [22], a particular discretization of DMs, building upon the advancements introduced in recent work - improved DDPM [41]. The objective is to enhance the mode coverage and stationary training of the DDPM framework. The following

[1]In this work, we consider deep neural network as a type of computer model inspired by the human brain that learns to recognize patterns in data by building multiple layers of interconnected artificial neurons, allowing it to understand complex information and make accurate predictions.

subsections introduce the DMs in detail. The derivation and proof of the section are presented in a manner similar to that in [22].

2.1.1 Generative Modeling and Diffusion Process

Generative modeling is a machine learning problem with many applications and is usually accomplished by density estimation, a statistical technique used to estimate the underlying probability distribution of a set of data points. It enables us to gain insights into the data's underlying structure, identify patterns, and make informed decisions in various fields, such as machine learning, signal processing, and data analysis. However, it is commonly assumed that the data is governed by a complex data distribution, which might be difficult to approximate directly. To circumvent this, one can leverage the tractable prior $p(z)$ in nature (e.g. Gaussian Distribution) and move the sample from the prior distribution to the target data density. In this sense, the data density is approximated by a posterior distribution given the prior: $p_{\text{data}}(x) \approx p_{\text{model}}(x)^2 = p(x|z)$. Here, we denote the $q(z|x)$ and $p(x|z)$ as forward and backward mapping.

Fortunately, the backward mapping can be modeled using a deep neural network by minimizing the Kullback–Leibler (KL) divergence between the posterior of forward mapping and backward mapping $D_{\text{KL}}(q(x|z) \parallel p(x|z))$. This has led to a series of works posing different assumptions on the tractable prior and model distribution. For example, VAEs [28] assume the tractable prior as a low dimensional Gaussian and the forward and backward mapping is achieved by an autoencoder.

In this context, DMs can be viewed as an instance where the prior is a standard Gaussian distribution sharing the same dimension as the data distribution and the forward mapping (diffusion process) is a gradual noise injection process (See Fig. 2.1).

Specifically, DDPM consists of a chain of latent variables of Markov property and can be decomposed into the multiplication of a series of conditional distributions of the same dimension (See Fig. 2.1). The forward joint distribution or forward mapping can be described as follows:

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1}). \tag{2.1}$$

where $\{x_t\}_{t=1\ldots T}$ are the intermediate latent variables along the forward diffusion

[2]In this work, we refer model distribution $p_{\text{model}}(x)$ as the approximation to the data density $p_{\text{data}}(x)$.

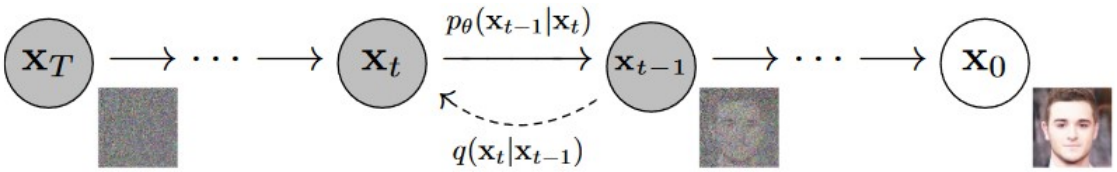


Figure 2.1: The directed graphical model of DDPM. The latent variables along the sampling path share the same dimension and are hypothesized to be Markovian. The noise is gradually added to the data in the forward diffusion process, while the reverse process recovers the lost semantic information and pushes one step closer to the clean data once the network evaluation. Demo image from [22].

process.

The latent variables x_t depend only on x_{t-1} and are perturbed by a pre-defined noise injection process:

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z, \quad z \sim \mathcal{N}(0, I). \tag{2.2}$$

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I). \tag{2.3}$$

where $\{\beta_t\}_{t=1\ldots T}$ is the variance schedule controlling the noise level.

Theoretically, $\{\beta_t\}_{t=1\ldots T}$ can be designed in many ways, tailoring to the speed of forward data corruption. In this work, we only examine the linear schedule proposed in [22], in accordance with the most existing work, where $\{\beta_t\}_{t=1\ldots T} = \frac{T-t}{T-1}\beta_1 + \frac{t-1}{T-1}\beta_T$, with T set to 1000, so that the $\{\beta_t\}_{t=1\ldots T}$ is linearly interpolated between β_1 and β_T.

It should be noted that the mean of the forward diffusion $q(x_t|x_{t-1})$ is scaled by $\sqrt{1-\beta_t}$ to bound the variance so that the x_T approximates to standard Gaussian as T goes to infinity.

According to Eq. (2.2), it's non-trivial to show that any noisy intermediate variables can be obtained via direct sampling (See Section A.1):

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z, \quad z \sim \mathcal{N}(0, I). \tag{2.4}$$

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I). \tag{2.5}$$

where $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{t=1}^{T} \alpha_t$.

Again, as shown by the Eq. (2.5), as $\bar{\alpha}_t$ approaches to 0 with large T, the Gaussian perturbation kernel progressively transforms the data into a standard Gaussian distribution denoted by x_T.

Interestingly, the posterior of forward diffusion $q(x_{t-1}|x_t, x_0)$ under this circumstance is tractable and can be computed using the Bayes rule (See Section A.2):

$$\begin{aligned} q(x_{t-1}|x_t, x_0) &= \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t \mathrm{I}). \\ \tilde{\beta}_t &:= \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t. \\ \tilde{\mu}_t(x_t, x_0) &:= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t. \end{aligned} \tag{2.6}$$

where $\tilde{\mu}_t, \tilde{\beta}_t$ are the posterior mean and variance of the forward diffusion process.

From Eq. (2.6), it is noted that the posterior of forward diffusion $q(x_{t-1}|x_t, x_0)$ shares the same analytical form of Gaussian distribution. Therefore, it is natural to assume the intermediate variables of the reverse process to be also Gaussian, with a tiny amount of noise being removed at each timestep. Practically, the reverse process is approximated by a deep neural network $p_\theta(x_t|x_{t-1})$.

$$p_\theta(x_{t-1}|x_t) \sim \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)). \tag{2.7}$$

where $\mu_\theta(x_t, t), \Sigma_\theta(x_t, t)$ denote the mean and variance of the denoised distribution of the reverse process, and θ represents the model parameters.

Heuristically, a neural network is trained to denoise the image blurred with Gaussian noise with multiple noise levels, and thus the network captures both coarse and fine-grained image features. The reverse mean $\mu_\theta(x_t, t)$ is parameterized as the model output, while the reverse variance $\Sigma_\theta(x_t, t)$ can either be learned or approximated by the posterior variance of the forward process $\tilde{\beta}_t$ or the noise schedule β_t.

In the same spirit of the forward diffusion, the joint distribution of the reverse process can also be depicted as a series of transitions starting from $x_T \sim \mathcal{N}(x_T; 0, I)$:

$$p_\theta(x_{0:T}) := p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t). \tag{2.8}$$

2.1.2 Training Objectives

DDPM trains to optimize the variational bound of the negative log-likelihood of data distribution (See Section A.3):

$$\begin{aligned}\mathbb{E}[-\log p_\theta(x_0)] &\leq \mathbb{E}_q\left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right] \\ &= \mathbb{E}_q\left[-\log p(x_T) - \sum_{t\geq 1}\log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right] := L_{vlb.}\end{aligned} \tag{2.9}$$

The upper bound L_{vlb} in the Eq. (2.9) can be further reduced to the tractable KL divergence [30] (See Section A.4 for the complete derivation):

$$\begin{aligned}\mathbb{E}_q\Big[D_{\mathrm{KL}}(q(x_T|x_0) \parallel p(x_T)) + \sum_{t>1} D_{\mathrm{KL}}(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t)) \\ - \log p_\theta(x_0|x_1)\Big].\end{aligned} \tag{2.10}$$

where $D_{\mathrm{KL}}(q(x_T|x_0) \parallel p(x_T))$ is denoted as L_T, $D_{\mathrm{KL}}(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t)$ as L_{t-1}, and $-\log p_\theta(x_0|x_1)$ as L_0.

L_T is free from the model parameters and can be neglected, as the distribution $q(x_T|x_0)$ converges to the Gaussian $p(x_T)$ when T is large enough. Optimizing the term L_{t-1} pulls the denoised distribution to the ideal posterior of forward diffusion. In this sense, L_0 can also be seen as a special case of L_{t-1} as shown in Eq. (2.11), where the $q(x_0)$ is known and does not depend on the model.

$$D_{\mathrm{KL}}(q(x_0) \parallel p_\theta(x_0|x_1)) = \int q(x_0)\log \frac{q(x_0)}{p_\theta(x_0|x_1)} dx_0. \tag{2.11}$$

2.1.3 Simplified Training Objective

This section mainly discusses the reparameterization of reverse diffusion. Noted that Eq. (2.4) provides an efficient way of directly sampling noisy data at arbitrary steps. Consequently, the training can be performed by uniformly sampling t from [0, 1, ..., T] and corrupting the data to x_t to compute the L_{vlb}.

To model the denoised distribution of $p_\theta(x_{t-1}|x_t)$, there are many ways to elucidate the model output. The most apparent choice is to estimate the $\mu_\theta(x_t, t)$ directly, or one can predict the noise ϵ and leverage the reparameterization to obtain x_{t-1}. In [22], Ho., et.al compare the ϵ-prediction and μ_{t-1}-prediction and argues that both

achieve relatively equal sample quality, while predicting noise ϵ allows for the simplified objective and resemble the denoising score matching [54], a theoretical link is bridged to the score-based models [54, 55]. Additionally, one can also predict the clean image x_0, but is found to lead to degraded sample quality in [22]. Therefore, to simplify the overall design choice, we follow [22, 41] to train a noise estimation network $\epsilon_\theta(x_t, t)$. Nonetheless, recent work [3] proposes to train a dual-output network to predict both the ϵ and x_0, and its linear combination improves the generation quality, especially with a limited number of sampling steps is considered.

Below, we show how the training objective term L_{t-1} in Eq. (2.10) boils down to the final simplified objective. We first derive the objective for the mean of the reverse process, supposing the fixed variance schedule, and then deal with the learned variance in Section 2.1.4. Noted that the gradient is detached from each other when both the mean and variance of the reverse process are estimated by the network.

Based on the result of KL-divergence between two Gaussian, the L_{t-1} can be written as:

$$L_{t-1} = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \parallel \tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t) \parallel^2 \right] + C. \tag{2.12}$$

where C is a constant with respect to θ, σ_t^2 is the variance of the reverse process, $\tilde{\mu}_t$ is the posterior mean of the forward process, and μ_θ is the mean of the reverse process.

From Eq. (2.12), we can again visualize that the most obvious option is to predict the mean of reverse process μ_θ. Yet, the parameterization ($x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t(x_0, \epsilon) - \sqrt{1-\bar{\alpha}_t}\epsilon$) from Eq. (2.2) and Eq. (2.6) can be leveraged to further simplify as:

$$L_{t-1} - C = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \parallel \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon \right) - \mu_\theta(x_t, t) \parallel^2 \right]. \tag{2.13}$$

Eq. (2.13) reveals that $\tilde{\mu}_t$ is accessible given x_t. With x_t set to be the network input, a similar reparameterization of the reverse mean can be considered as:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right). \tag{2.14}$$

where the network output $\epsilon_\theta(x_t, t)$ is the approximator of the sampled noise.

Then we plug in the Eq. (2.14) into Eq. (2.13) and achieve the true variational bound:

$$L_{t-1} - C = \mathbb{E}_{x_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha_t})} \parallel \epsilon - \epsilon_\theta(x_t, t) \parallel^2\right]. \tag{2.15}$$

Practically, Ho., et.al [22] find out that simply dropping the coefficient in the Eq. (2.15) is beneficial to the sample quality and stabilizes the training when predicting the noise term $\epsilon_\theta(x_t, t)$, while predicting $\mu_\theta(x_t, t)$ only works when training with the true variation bound (2.15). The simplified objective $L_{simples}$ is as follows:

$$L_{simple} := \mathbb{E}_{t,x_t,\epsilon}\left[\parallel \epsilon - \epsilon_\theta(x_t, t) \parallel^2\right]. \tag{2.16}$$

The simplified objective in Eq. (2.16) can be perceived as a reweighted variational bound that emphasizes the training of different noise levels. With the small t being down-weighted, the denoising quality of intermediate and large t can be improved significantly. Besides, as the small t mainly concentrates on recovering the imperceptible details, the overall sample quality is not severely deteriorated with the training period being shortened. Nevertheless, as the beginning step (small t) also massively contributes to the true variational bound, the log likelihood can be negatively impacted, potentially causing mode covering issues [41] (See Section A.5).

2.1.4 Learned Variance

Reverse variance is crucial for density estimation to achieve a competitive log likelihood, since both the reverse mean and variance determine the denoised distribution $p_\theta(x_{t-1}|x_t)$. It is generally convinced that the likelihood performance relates to how well the model captures the modes of data distribution [43]. In other words, sample diversity is greatly tied to the performance of the likelihood. Ho., etc [22] adopts the fixed variance schedule, where the same variance being applied to inject the noise in the forward diffusion serves as the reverse variance. As mentioned earlier, bad mode coverage might be caused by fixing variance and simplified objective, and thereby we may improve the log-likelihood by learning the $\Sigma_\theta(x_t, t)$.

Empirically, the possible range of $\Sigma_\theta(x_t, t)$ is very small since only a tiny amount of noise is removed at each iteration, and thus predicting $\Sigma_\theta(x_t, t)$ directly by the network might be suboptimal even in the log domain. To solve this, in [41], a linear interpolation is done between β_t and $\tilde{\beta}$, with the network only deciding the weighting term v. The computation is performed in the log domain so that there is no constraint on network output for v:

$$\Sigma_\theta(x_t, t) = \exp(v \log \beta_t + (1 - v) \log \tilde{\beta}_t). \tag{2.17}$$

Here we kindly remind that the β_t is the linear variance schedule (See Sec. 2.1.1) and $\tilde{\beta}_t$ is the variance of the posterior of the forward mapping in Eq. (2.6).

Then we can define a new training objective combined the L_{simple} and L_{vlb}. The former only glues to the reverse mean, while the latter contributes to the true variational bound for likelihood improvement:

$$L_{hybrid} = L_{simple} + \lambda L_{vlb.} \tag{2.18}$$

where the λ weights the multiple losses. Following [41], λ is set to 0.001 and the gradient of L_{vlb} is detached from the output of $\mu_\theta(x_t, t)$ to prevent L_{vlb} from overwhelming L_{simple}, and thus the sample diversity is traded off with the sample quality.

2.1.5 Reverse Sampling

After training a noise estimation network, samples can be generated by iterative denoising through $p_\theta(x_{t-1}|x_t)$, from T to 0, where the $x_T \sim \mathcal{N}(x_T; 0, \mathbf{I})$. Specifically, the sampling formula via parameterization can be written as follows:

$$x_{t-1} = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t * z. \tag{2.19}$$

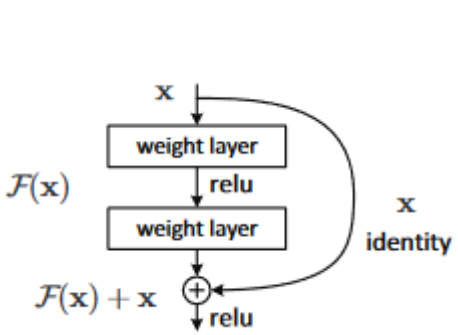
where the σ_t corresponds to the standard deviation of the reverse variance $\Sigma_\theta(x_t, t)$, and z is the sampled noise.

2.1.6 Diffusion Network

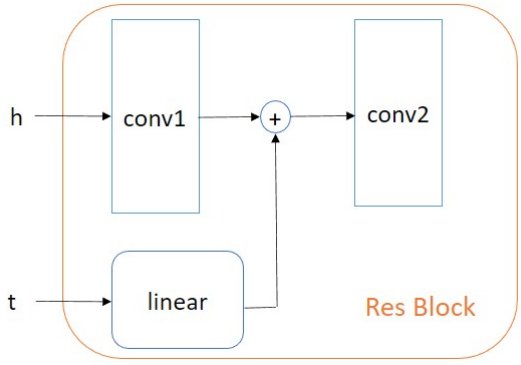
The primary obstacle in the diffusion model lies in developing a robust parameter-sharing network capable of effectively denoising images with varying levels of noise. One commonly employed network in the diffusion model is the U-Net [44] architecture, originally designed for image segmentation, due to the task similarity of image understanding and the same spatial resolution of the network's input and output. It has been emphasized that without utilizing sophisticated architectures (e.g. U-Net), the performance of the diffusion model may hugely deteriorate [41, 55].

Considering the importance of the diffusion network, in this section, we review the network architecture Res-UNet proposed in [41]. Since the U-Net [44] is the most commonly used framework for medical image segmentation, we refer the reader to Section 2.4.1 for the U-Net [44].

Res-UNet modifies the U-Net architecture by adding a skip connection for each convolutional block (See Fig. 2.2a), which is commonly helpful for feature aggregation



(a) A residual block; block input x is fused with the block output $F(x)$ via direct sum. Image from [20].



(b) The residual block with time encoding t (Skip connection omitted).

Figure 2.2: The Residual block used in the diffusion models.

and gradient flow. First, the fusion of feature maps at multiple layers enhances the learned representation and prevents the gradient exploding issue. Second, skip connections also allows multiple paths for gradient propagation, which might accelerate convergence and enhance training stability.

The exact residual block design follows PixelCNN++ [47], consisting of Group Normalization [63], 3×3 convolutional block and GeLU activation function [21]. The overall network is visible in Fig. 2.3, taking both x_t and t as input. Each time step is projected into a unique sinusoidal encoding, passing as a Fourier features for aggregation:

$$\begin{aligned} sin(t) :=& [sin(2w_1 t), sin(2w_2 t), ..., sin(2w_d t), \\ & cos(2w_1 t), cos(2w_2 t), ..., cos(2w_d t)]. \end{aligned} \tag{2.20}$$

where w_1, w_2, ..., and w_d are the pre-defined frequencies free from the parameters, and d is the number of dimension of the projected vector.

Fig. 2.2b shows that the sinusoidal time encoding t goes through a linear time module to adjust the channels for the feature maps of varying depths. Eventually, the time encoding is fused with the feature maps between the first and second conv blocks at each residual block via direct sum.

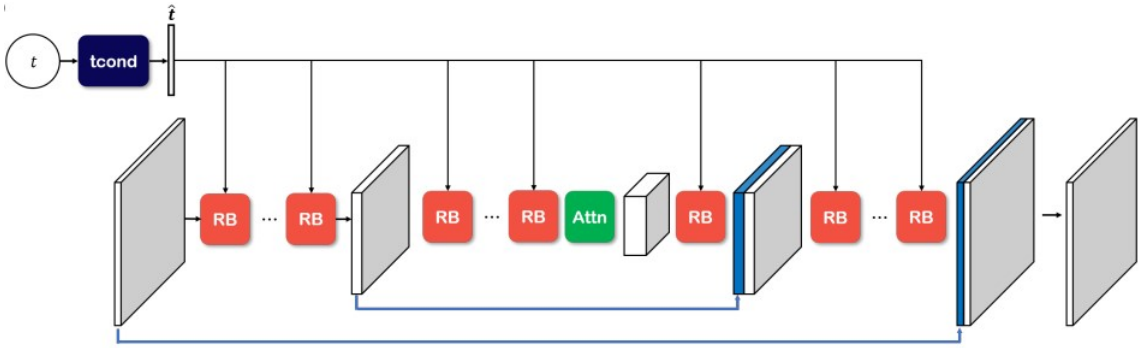


Figure 2.3: The overall network design of the diffusion network. RB: residual block, Attn: efficient self-attention block. The residual Block is the basic unit for image downsampling and upsampling with the skip connection introduced. See text for more description. Efficient self-attention block is applied at the deepest level and improve feature extraction via QK product with linear complexity. We refer the reader to [48] for more details about efficient attention.

2.2 Posterior sampling using Diffusion Models

As the impressiveness of DDPM is revealed for image generation, a natural next step would be investigating its performance of controllable generation. In this section, we roughly review two lines of work for posterior sampling $p(y \mid x)$ using DMs.

2.2.1 Conditional Sampling

Using Bayes rule, one can easily show that posterior sampling is equivalent with the multiplication of prior and forward model:

$$p(y_{t-1}|y_t, x) \sim p(y_{t-1}|y_t)p(x|y_t). \tag{2.21}$$

In the context of the DMs as Eq. (2.21), in order to sample from the posterior $p(y_{t-1}|y_t, x)$, conditional sampling concentrates on leveraging a pre-trained diffusion model to capture the generative prior $p(y_{t-1}|y_t)$, and incorporate the conditional information during sampling. The key insight is to "hijack" the intermediate latent variables y_t or y_{t-1} and enforce consistency with the guidance signal x.

In practice, the conditional information is only integrated during sampling through the forward model $p(x|y_t)$, a reverse mapping contrary to our target. If the forward model is linear (e.g. $y = Ax$), it is not hard to estimate without training. This has led to a series of works leveraging pre-trained diffusion models for solving linear inverse problems in an unsupervised manner [9, 55, 56]. However, if the forward

model is nonlinear, a certain approximation is inevitable. For example, Dhariwal, P. and Alex N. [14] trains a separate classifier to perform class-conditional sampling referred to as classifier guidance, in which the gradient of the forward model is incorporated for consistency. The result balances sample diversity and fidelity.

2.2.2 Conditional Model

Another line of work focuses on training a conditional diffusion model (CDM), once the paired data $\{(x_i, y_i)\}_{i=1}^N$ is collected. In this scenario, conditional distribution $p(y|x)$ needs to be modeled given a dataset of paired data $D = \{(x_i, y_i)\}_{i=1}^N$, where N is the number of data points. Specifically, the target distribution $p(y|x)$ is a one-to-many mapping that multiple target data y can align with the source information x.

To achieve this, the denoising network takes both the noisy input y_t and conditional input x to predict the sampled noise. This leaves much room for the design of a conditional strategy to build an effective CDM. Existing work [46] for super resolution employs simple concatenation along the channel dimension between noisy input and low resolution image ($\text{net}(\text{cat}[y_t, x], t)$), obtaining great results on the recovery of fine-grained details. Since CDM is based upon a more general relation between the condition and target, it is much easier to apply directly to the case of multimodal generation. For instance, the latent diffusion model [45] can be applied for class, segmentation map and text-guided synthesis via a separate encoder. In this case, a conditional encoder is leveraged to encode the features of x and then aggregate the representation with the denoising network [1] (See Fig. 2.4).

2.3 Diffusion Model for Zero-shot Data Editing

A remarkable property of DMs is that zero-shot inverse problem solving can be attained without problem-specific training. As the forward diffusion process is non-parametric, a mild initialization can be optimized by adding noise and applying the diffusion denoiser. For instance, for image colorization [55, 11, 9], the grayscale image can be tripled along the channel dimension and then being injected noise to a certain intermediate timestep for the denoiser. In this section, we review how the pretrained diffusion model performs zero-shot image inpainting and will see later how the same logic generalizes to image level that training a joint diffusion model with pair data resembles conditional sampling in Section 3.1.

The initial idea of inpainting is proposed by Yang, S., etc [54] to illustrate the representation learned by the DMs, and trailed by a dozen of work [55, 9, 35, 27]

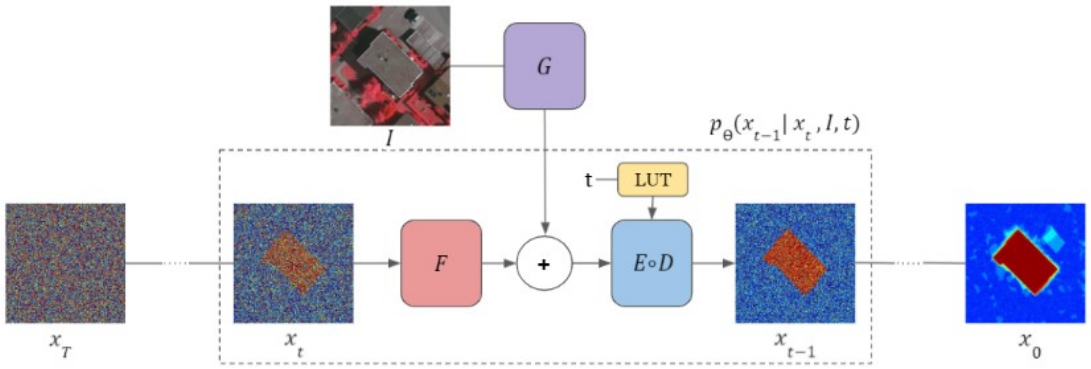


Figure 2.4: SegDiff [1]: a diffusion-based segmentation method. The two separate encoders (G & F) are employed for denoising noisy segmentation mask the extracting features of the image. The encodings are integrated by the direct sum in the latent space.

to explore a robust sampling algorithm, which gives the most harmonious example visually and highest sample quality quantitatively. As for the vanilla sampling method, the image is subdivided into known regions for conditioning and unknown regions for generation. The intermediate noisy image sources from two directions: the forward diffusion of the known regions, and the reverse diffusion of the unknown regions:

$$\begin{aligned} x_{t-1}^{known} &\sim \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I). \\ x_{t-1}^{unknown} &\sim \mathcal{N}(\mu_\theta(x_t,t), \Sigma_\theta(x_t,t))s. \\ x_{t-1} &= m \odot x_{t-1}^{known} + (1-m) \odot x_{t-1}^{unknown}. \end{aligned} \tag{2.22}$$

where the m is the inpainting mask, x_{t-1}^{known} samples from the forward process and $x_{t-1}^{unknown}$ samples from the reverse process.

The combined output via masking enforces harmony between the known and unknown regions for the next step's evaluation. The pseudocode[3] and examples can be seen in Alg. 1 and Fig. 2.5. We refer the reader to more illustrations and examples to Repaint [35].

[3]In the Repatint [35], an additional resampling strategy is introduced to enhance the quality of the output. This strategy brings about certain differences in the algorithms in contrast to Alg. 1, while the core ideas behind the algorithms remain unchanged.

Algorithm 1 Diffusion Model for Image Inpainting

Initialization:
$x_T \sim \mathcal{N}(0, I)$
for t = T, ..., 1 **do**
  $\epsilon \sim \mathcal{N}(0, I)$ if t > 1 else $\epsilon = 0$
  $x_{t-1}^{known} = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$
  $z \sim \mathcal{N}(0, I)$ if t > 1 else $z = 0$
  $x_{t-1}^{unknown} = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t * z$
  $x_{t-1} = m \odot x_{t-1}^{known} + (1 - m) \odot x_{t-1}^{unknown}$
end for
Output: x_0

Figure 2.5: Qualitative results on ImageNet 512 × 512 for thin mask from Repaint [35].

2.4 Medical Segmentation Network

The medical segmentation network is a special type of neural network $f_\theta(x)$ for partitioning the image pixels into desired categories. The network projects the image into a probability map of targeted classes, and then the class with the maximum probability is taken as the final output:

$$y = \text{argmax}(f_\theta(X), \text{dim} = 1). \tag{2.23}$$

where $X \in (H, W, 3)$, $f_\theta(X) \in (C, H, W)$, $y \in (H, W)$ and C is the number of class.

The Eq. (2.23) can be applied to both 2D and 3D biomedical segmentation tasks, while the key distinction lies in the additional spatial dimension present in 3D data. It is also apparent that directly training a 3D network is more computationally expensive than that of 2d. As the neural network is key to the medical segmentation

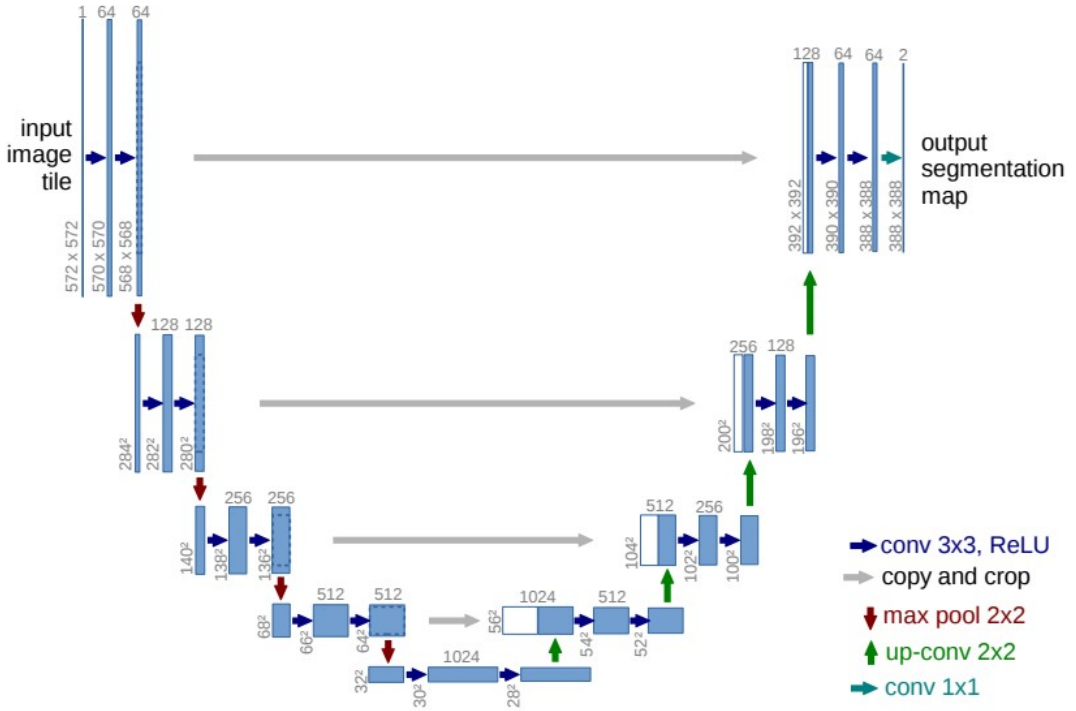


Figure 2.6: The U-Net Architecture: A powerful deep learning framework for semantic segmentation tasks in medical imaging. It comprises a U-shaped network structure with both contracting and expanding paths. The contracting path consists of convolutional and pooling layers to capture context, while the expanding path uses transposed convolutions for precise localization. Skip connections facilitate the integration of multi-scale features, enabling different size of receptive fields.

performance, in this section, we review the biomedical segmentation network architecture. It is also noted that the same network is also utilized to train the DDPM with coincidence due to task similarity.

2.4.1 U-Net

U-Net (See Fig. 2.6) proposed by Olaf R., etc [44] is a CNN architecture originally designed by biomedical segmentation, and soon being extended to many applications where the model's input and output share the same spatial resolution. The name "U-Net" originates from the entire U-shaped design of the architecture, which resembles an encoder-decoder framework. Currently, the encoder and decoder are not only limited to the CNN structure but general feature extractor and interpreter.

The encoder-decoder structure is highly effective for tasks where pixel-wise understanding is required for the output, such as semantic segmentation, denoising, etc. Overally, the entire architecture consists of two main parts: the contracting path for downsampling and the expanding path for reconstruction.

The contracting path resembles a typical CNN encoder and is responsible for capturing context and extracting features from the input image. It composes of a series of convolutional and pooling layers, which progressively reduce the spatial dimensions of the input while increasing the number of feature channels. The pooling operations are used for downsampling, contracts the image into a lower resolution feature map. These operations, including maxpooling, average pooling or stride convolutional kernel, capture the context and extract higher-level features.

The expanding path serves as the decoder and reconstructs the segmented image from the lower-resolution feature map. It consists of a series of up-convolutional layers, also known as transposed convolutions or deconvolution [42]. The up-convolutional layers perform upsampling implemented by either parameter-free extrapolation or convolutional kernel, increasing the spatial dimensions of the feature map while reducing the number of feature channels. To compensate for the information loss during dowansampling and fuse the multi-scale features maps for different spatial levels of features, the skip connection is adopted to aggregate the upsampled features with the corresponding feature maps from the contracting path. The skip connection is critical for recognizing objects of various sizes as multi-scale features being fused have different receptive fields, and also supports the active gradient flow with more paths for backpropagation. In the original design [44], the skip connection is implemented by simple concatenation. Then, the concatenated features are further processed through convolutional layers to refine the segmentation. The contracting and expanding paths are symmetric, with skip connections between them.

Overall, the U-Net architecture has proven to be very successful for various image segmentation tasks, particularly in the medical domain. Its ability to learn from limited training data and produce accurate pixel-level predictions has made it a popular choice in the field of image analysis.

2.4.2 nnU-Net

nnU-Net [25] (See Fig. 2.7) is a deep learning framework specifically designed for volumetric medical segmentation tasks. It is purely based on the U-Net [44] architecture, but over-engineered with a multitude of training recipes built upon the data format, exact memory constraint, etc.

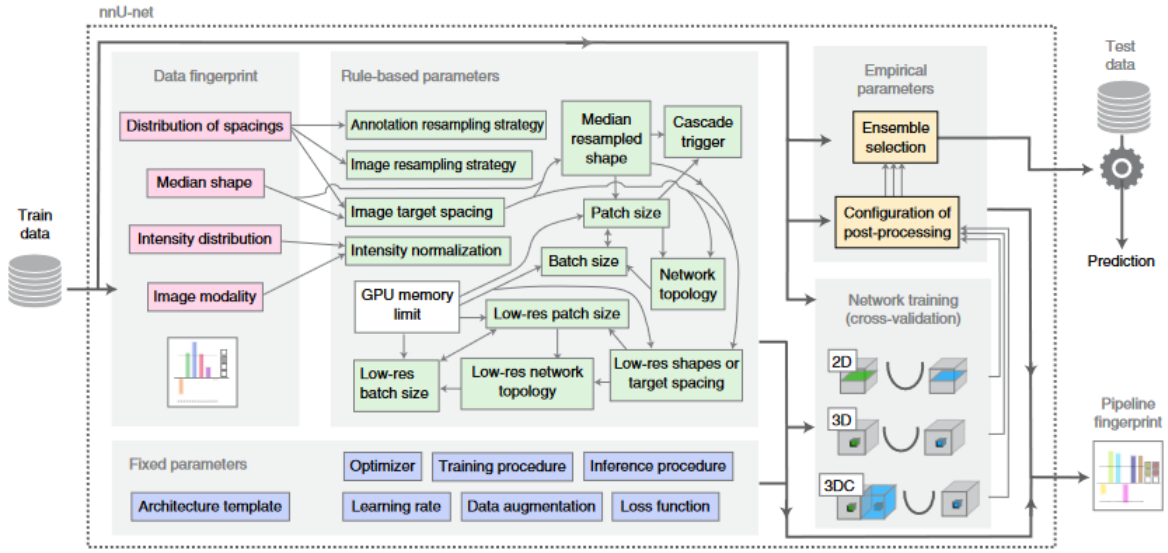


Figure 2.7: nnU-Net's interdependence among the hyperparameter setting, data pre-post processing, and model selection.

This results in a highly generalized and self-configured framework for practical training decisions via heuristic rules in terms of hyperparameters and data processing. nnU-Net achieves state-of-art on a majority of tasks without detailed network architecture design, but systematizing the complex process of manual method configuration, which was previously addressed either by cumbersome manual tuning or purely empirical approaches with practical limitations [25].

nnU-Net is a comprehensive toolbox with high flexibility, robustness, making it a valuable tool and baseline for medical imaging researchers and practitioners. Because of this, it can also be an appropriate training instance for to assess the effectiveness of new plug-and-play methods (e.g. data augmentation) for volumetric segmentation. The whole design of distilling the domain knowledge emcompasses three parts of parameter groups: fixed, rule-based and empirical parameters. In the rest of this section, we roughly review the core design. The full details are available in [25].

The fixed parameters, involving architecture, optimizer, learning rate, etc (See Fig. 2.7), are optimized for joint configuration across multiple datasets. This part is determined manually by the designer and stay fixed regardless of the datasets being applied to.

The rule-based parameters manage the data pre-processing and the setting of some parts of hyperparameters. This decision is a set of heuristic if-else rules merely inspired by the experience of the designer. For instance, the distribution of spacing decides the image resampling strategy (See Fig. 2.7): if the collected data is anisotrpic[4], a cubic spline interpolation is employed within the image plane and nearest neighbor interpolation for the outer plane, while a cubic spline interpolation is applied to all planes if the opposite.

The empirical parameters mainly decide the data post-processing and the model inference. For example, the best configured model among 2D, 3D, and 3D cascade model will be chosen for inference according to their validation performance.

Besides, nnU-Net utilizes an ensembling approach during the inference phase to further boost performance. It combines the predictions from multiple models trained on different training sets or with different settings. This ensemble of models helps to capture diverse information and improve the segmentation results.

[4]According to [5], anisotropic refers to a situation where the pixel or voxel dimensions are not equal in all directions. In the context of medical imaging, it means that the spacing between pixels or voxels differs along different axes.

3

Joint Diffusion Model for Paired Data Synthesis

In this chapter, we illustrate the Joint Diffusion Model (JDM), a generative framework for pair data synthesis in parallel with the classical DMs, and CDM. We show how a joint model is trained and sampled to produce data, and how to leverage a joint model for conditional sampling, inspired by the zero-shot data edition using pretrained DMs (See Section 2.3). Generally speaking, the JDM still follows the training diagram of the DMs, however, instead of injecting noise and predicting the sampled noise for the single modality input, the same operation is performed on the multiple modalities simultaneously. To that end, joint modeling incorporates the marginal distribution of single modality data and conditional sampling, resembling the image inpainting using pretrained DMs [35]. Section 3.1 discusses the framework of the JDM in detail. Section 3.2 further illustrates the theory of the condition sampling.

3.1 Joint Diffusion Model

Supposing paired data is collected as $\{(x_{i1}, x_{i2}, ..., x_{im})\}_{i=1}^{N}$, where N is the number of data points and m is the number of modality (e.g. text image pair corresponds to m equals to 2). Rather than model the conditional distribution with parts of data adopted for conditioning, a JDM can be trained to model the joint distribution. Under this circumstance, extending from the classical DMs, the target data

distribution switches from uni-modal x to the joint variable $(x_1, x_2, ..., x_m)$:

$$p(x) \Rightarrow p(x_1, x_2, ..., x_m). \tag{3.1}$$

In spite of the huge shift from unimodal to multi-modal data, the mathematical framework remains the same as the DDPM [22], because we can simply define a joint variable and this results in the same mathematical derivation of the DMs:

$$x =: (x_1, x_2, ..., x_m). \tag{3.2}$$

3.2 Conditional Sampling

There are several advantages of considering pair data entirely as the target in the context of DMs. First, joint modeling incorporates the marginal distribution. If the joint distribution $p(x_1, x_2, ..., x_m)$ is correctly modeled, we can easily sample the arbitrary unimodal data $\{x_i\}_{i=1}^m$ by simply dropping the remaining modalities $\{x_\tau\}_{\tau \neq i}$. Second, it also permits condition sampling given an arbitrary number of modalities. For instance, a joint distribution is modeled for CT, MRI images for the same patient, one can use either CT or MRI to induce the generation of the other. Theoretically, assuming the number of available modalities is k, we reorder the multimodal data as $(x_1, x_2, ..., x_{m-k})$ and $(x_{m-k+1}, x_{m-k+2}, ..., x_m)$, with the former for sampling and the latter for conditioning. The conditional distribution can be decomposed by the multiplication rule:

$$p(x_1, x_2, ..., x_{m-k} | x_{m-k+1}, x_{m-k+2}, ..., x_m) = \frac{p(x_1, x_2, ..., x_m)}{p(x_{m-k+1}, x_{m-k+2}, ..., x_m)}. \tag{3.3}$$

where m is the number of modalities.

In the diffusion setting, the distribution is characterized as the progressive denoised distribution. Therefore, the practical distribution for sampling is:

$$\frac{p(x_1, x_2, ..., x_m)}{p(x_{m-k+1}, x_{m-k+2}, ..., x_m)} \Rightarrow \frac{p(x_{t-1,1}, x_{t-1,2}, ..., x_{t-1,m} | x_{t,1}, x_{t,2}, ..., x_{t,m})}{p(x_{t-1,m-k+1}, x_{t-1,m-k+2}, ..., x_{t-1,m})}. \tag{3.4}$$

where t is the timestep across the gradual denoising process.

Note that this results in two easy-to-sample distributions in the practical context. For the denominator in Eq. (3.4), we can use the forward diffusion to corrupt the data as the $(x_{m-k+1}, x_{m-k+2}, ..., x_m)$ being known. For the nominator in Eq. (3.4),

the denoised joint distribution can be employed for reverse sampling with a trained joint model on hand:

$$\begin{aligned} p(x_{t-1,1}, x_{t-1,2}, ..., x_{t-1,m}|x_{t,1}, x_{t,2}, ..., x_{t,m}) &\sim \mathcal{N}(\mu_\theta(x_t, t)), \Sigma_\theta(x_t, t)) \\ p(x_{t-1,m-k+1}, x_{t-1,m-k+2}, ..., x_{t-1,m}) &\sim \mathcal{N}(\sqrt{\bar{\alpha_{t-1}}}x_0, (1-\bar{\alpha_{t-1}})I) \end{aligned} \tag{3.5}$$

where θ is the model parameters and $\bar{\alpha_{t-1}}$ is the pre-defined noise schedule (See Section 2.1.1).

Notably, this resembles the zero-shot image inpainting using a pretrained diffusion model in Section 2.3, in which the image is subgrouped into the known and unknown regions. Literally, training a JDM and performing conditional sampling spread the target from pixel level to image level. Hence, JDM is also orthogonal to various zero-shot inverse problem solvers using the DMs [55, 10, 9].

Empirically, the sampling can be achieved by replacing the generated intermediate noisy data of known modality with the known data through the forward data corruption.

However, extending from unimodal data to higher dimensional paired data inevitably poses a huge challenge to the neural network. This may possibly lead to failure of mode capturing, unstable training, etc., as the network shoulders the extra burden of learning the mutual dependence among the multimodal data rather than the unilateral dependence in the conditional model. Hence, in this thesis, we study the simple case where $m = 2$ and train the JDM of medical image and segmentation mask, a task that usually suffers from data sparsity problems. A trained JDM can ideally produce countless numbers of data. Given the great sample quality and diversity, we expect the JDM can be leveraged as a data augmentation tool to improve the performance of medical image segmentation.

As for mask-conditioned sampling, we can add an extra step of substituting the generated mask with the true mask injected by noise, which only results in a few lines of code difference (See Alg. 2). Theoretically, a trained model can also be used for medical segmentation, but it is beyond the interest of this work.

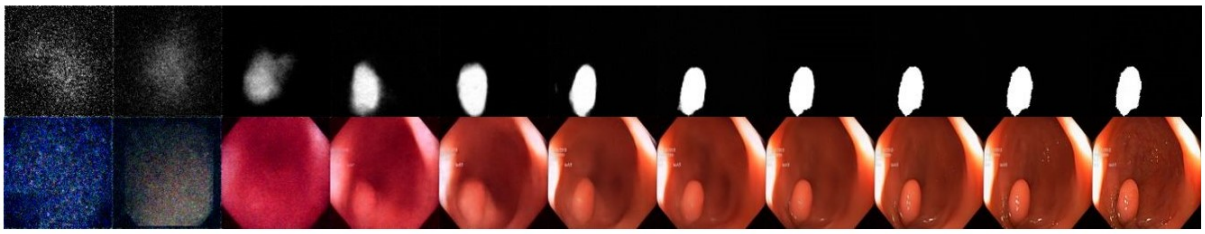


Figure 3.1: The gradual denoising procedure of the Joint Diffusion Model (JDM). A paired sample of medical image and segmentation mask is produced with a single reverse run.

Algorithm 2 Conditional Sampling

Initialization:
$x_T \sim \mathcal{N}(0, I)$, $m_0 :=$ mask_true
for t = T, ..., 1 **do**
  $z \sim \mathcal{N}(0, I)$ if t > 1 else $z = 0$
  $x_{t-1} \sim \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t)), \Sigma_\theta(x_t, t))$
  $\text{img}_{t-1}, \text{mask}_{t-1} = \text{chunk}(x_{t-1}, \text{dim} = 1)$
  $\text{mask}_{t-1} \sim \mathcal{N}(m_{t-1}; \sqrt{\bar{\alpha}_{t-1}} m_0, (1 - \bar{\alpha}_{t-1})I)$
  $x_{t-1} = \text{cat}([\text{img}_{t-1}, \text{mask}_{t-1}], \text{dim} = 1)$
end for
Output: x_0

4

Experiments

In this chapter, we discuss the experimental setups of this work. This involves in the dataset descriptions, experimental environments, setups of the training and evaluation, and performance metrics.

4.1 Experimental Setups

Software Setup:

We use Pytorch [40], an open source deep learning framework that provides a flexible and efficient platform for building and training various types of deep learning models. The experiments of the JDM is based on the code[1] from [64]. For biomedical segmentation, we use the training instance nnU-Net [2]. The code and scripts to reimplement this work is accessible through: https://github.com/CaviarLover/diffAug.

Hardware Setup:

All experiments are executed on two NVIDIA GEFORCE RTX 3090 GPUs and the AMD Ryzen 9 3950X 16-Core Processor.

[1] https://github.com/JuliaWolleb/Diffusion-based-Segmentation
[2] https://github.com/MIC-DKFZ/nnUNet

Transfer Learning on Synthetic Data:

We adopt a two-step process to perform supervised transfer learning on synthetic data:

1. Pretraining with Synthetic Data: We begin by pretraining the segmentation model using supervised learning on the synthetic data samples. This initial training step allows the model to gain insights from the generated data and learn relevant features.

2. Fine-tuning with Real Data: After the pretraining phase, we proceed to finetune the segmentation model using real data. By fine-tuning on real data, the model can adapt and refine its learned features to better align with the characteristics of real-world data.

It is worth noting that an alternative strategy could involve mixing both real and synthetic data during training. However, our preliminary experiments show that the success of this scheme largely depends on the proportion of the data combination and varies significantly across different datasets. Due to the complexities involved and the potential challenges in finding the optimal data mixing ratio, we have decided not to consider this strategy in this work.

By adopting the two-step approach of pretraining with synthetic data and then fine-tuning on real data, we aim to improve the segmentation performance and leverage the benefits of synthetic data in a more controlled and effective manner.

4.2 Datasets

Kvasir dataset is a polyp segmentation dataset composed of consistent video frames of endoscopy. The train test split is adopted from [15], where randomly selected 900 images from Kvasir and 450 images from Clinic-DB formulating the training set.

GlaS dataset comprises microscopic images of slides stained with Hematoxylin and Eosin (H&E). These images and corresponding ground truth annotations by expert pathologists are used for training and testing purposes. The dataset contains 85 images for training and 80 images for testing.

MoNuSeg dataset is created using images captured at 40x magnification of H&E stained tissue. It includes images from multiple organs and patients. The MoNuSeg dataset encompasses 30 training images with approximately 22,000 nuclear bound-

ary annotations and 14 testing images with over 7,000 nuclear boundary annotations.

Testsets:

Kvasir [26], Clinic-DB [4], CVC-ColonDB [57], CVC-300 [60] and ETIS [49] are five polyp segmentation datasets, commonly used to demonstrate the effectiveness of the segmentation model of polyps. Since the generative model is trained on a mixed dataset of Kvasir and Clinic-DB as mentioned earlier, we claim the Kvasir and Clinic-DB as the validation set, and CVC-ColonDB, CVC-300, and ETIS as the generalization set. The testset of MoNuSeg with 14 images is leveraged to evaluate the nuclei segmentation. The segmentation model is trained on all images resized to 256 (polyps) and 512 (nuclei) and inference is performed in the same resolution and upsample to the original resolution via nearest upsampling.

4.3 Hyperparameters

4.3.1 Joint Diffusion Model (JDM)

Key hyperparameters of the JDM are summarized in Table 4.1. Aligned with [41], we train the generative model with the number of residual blocks at each level set to 2. The number of base channels is 128, and the attention block of 4 heads is applied on the U-Net level with the downsampling factor of 8 and 16. We do not use the dropout during training. For more detailed information, we recommend referring to the publication [41] and visiting the GitHub repository at https://github.com/openai/improved-diffusion. These sources provide additional details and resources related to the setting.

Train Period. Since there is no suggested number of steps for training a diffusion model, we train the model for 60k steps on Kvasir, 40k on Monuseg, and 40k on GlaS. After training, the checkpoints after half steps are loaded to visualize the generated samples. We pinpoint the valid setup of the number of steps giving the most harmonious sample between the generated segmentation mask and medical image, while the early checkpoints lack fine-grained image features and are less harmonious. We do not qualitatively check the metric of the sample quality such as "Fréchet Inception Distance" (FID) [23] for picking the checkpoint.

	setting
number of diffusion steps	1000
learned variance	True
dropout	0.0
batch size	10
learning rate	1e-4
number of base channels	128
channel multiplier	1, 1, 2, 2, 4, 4
number of residual blocks	2(3)*
number of input channels	4
number of heads per attention block	4
depth level applied with attention block	16, 8

Table 4.1: The hyperparameter setting of the JDM. (*) The number of residual blocks is set to 2 and 3 for Kvasir and MoNuSeg, respectively.

4.4 Metrics

To evaluate the segmentation results, we adopt two most commonly used metrics mean Dice score (mDice) and mean Intersection over Union (mIoU). mDice also known as the Dice coefficient or Dice similarity coefficient, measures the similarity between the predicted segmentation and the ground truth. It quantifies the overlap between the two by computing the ratio of twice the intersection of the predicted and ground truth regions to the sum of their areas.

mIoU is another widely used metric that evaluates the accuracy of segmentation algorithms. It calculates the ratio of the intersection area to the union area between the predicted and ground truth regions. The mIoU is the average IoU calculated over all the classes in the dataset. The computation of two metrics is as follows:

$$\text{mDice} = \frac{2 \cdot \sum_{i=1}^{c} TP_i}{\sum_{i=1}^{c}(TP_i + FP_i) + \sum_{i=1}^{c}(TP_i + FN_i)} \tag{4.1}$$

$$\text{mIoU} = \frac{1}{C} \sum_{i=1}^{C} \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i + \text{FN}_i} \tag{4.2}$$

where TP_i is the true positives for class i, FP_i is the false positives for class i, FN_i false negatives for class i, and c is the total number of classes.

4.5 Baselines

nnU-Net. The training of nnU-Net [25] strictly adheres to the guidelines presented in the official implementation. For detailed instructions, please refer to the nnU-Net repository on GitHub: https://github.com/MIC-DKFZ/nnUNet.

Polyp Segmentation. Except for the nnU-Net [25], we include some other methods for comparison. PraNet [15] and CaraNet [24] are two methods designed specifically for polyp segmentation. The PraNet set the best performance back in 2020 and CaraNet is known for its ability to effectively detect small medical objects by combining various modules to achieve this capability. The result can be found on the leaderboard of *paperswithcode.com*[3]. This leaderboard showcases the latest state-of-the-art of the benchmarks in the field of computer vision.

Nuclei Segmentation. We have incorporated several state-of-the-art methods for comparison on the MoNuSeg [31] dataset. Each of these methods introduces unique advancements to the U-Net [44] architecture to improve its performance in medical image segmentation. The results are extracted from the leaderboard of *paperswithcode.com*[4].

1. U-Net++ [66]: This method enhances the standard U-Net by deeply supervising the intermediate feature maps. This strategy compels the low-level features to become more discriminative, leading to improved segmentation results.

2. Attention U-Net [38]: By incorporating an attention gate mechanism on the skip connections, this method selectively highlights the salient features of the encoder. This pruning process helps the model focus on relevant information.

3. MedT [58]: This approach combines both transformer and CNN structures, creating two separate branches to process medical images. The transformer enables capturing long-range dependencies, while the CNN branch handles local features, improving the model's ability to capture both global and local context.

4. UCTransUnet [61]: Building upon the U-Net architecture, UCTransUnet introduces a global feature aggregation module. This module effectively fuses representations from different levels of the encoders and decoders, leading to a more comprehensive understanding of the input.

[3] https://paperswithcode.com/sota/medical-image-segmentation-on-kvasir-seg
[4] https://paperswithcode.com/sota/medical-image-segmentation-on-monuseg

5. SMESwin Unet [62]: This method leverages a superpixel segmentation branch to identify and exclude irrelevant features from the original image. By doing so, the Swin Transformer [34] based U-Net is enhanced to focus solely on important regions.

5

Results and Discussion

In this chapter, we present the main results of this thesis. We explore the capability of the Joint Diffusion Model (JDM) to produce medical image segmentation pair data. We further train the segmentation model based upon nnU-Net [25] to investigate the usefulness of the synthetic data.

In Section 5.1, we visualize the generated samples on several datasets, including pair data synthesis and mask-conditioned synthesis. In Section 5.2, we train the segmentation model in supervised manner using purely synthetic data for polyp and nuclei segmentation. In Section 5.3, we further leverage the pretrained checkpoint and finetune the segmentation model on the real data. In both cases, the number of used synthetic data is compared.

5.1 Visual Comparison

In this section, we train the JDM on three medical image segmentation datasets: Kvasir [26], MoNuSeg [31] and GlaS [50]. The generative model is trained from scratch. The full descriptions of the datasets can be found in [26] [50] [31].

5.1.1 Pair Data Synthesis

We highlight some of our generated samples and their associated segmentation masks (See Fig. 5.1). We further qualitatively investigate the similarity between the syn-

thetic sample and the closest sample in the training set in terms of the cross entropy mutual information.

As shown in Fig. 5.1, the synthetic samples demonstrate the effectiveness of the JDM in generating high-quality images with intricate details. It showcases the model's capability to capture the nuances of light contrast in endoscopy and accurately visualize small text descriptions of patients, particularly for polyp images (See last two rows of Fig. 5.1). Furthermore, the generated segmentation masks, together with their corresponding images, exhibit a remarkable visual consistency and are appropriately positioned.

Notably, the results display a significant diversity in the generated samples, despite being trained on a limited number of large-resolution images (512×512 for the MoNuSeg with only 30 images). The first two rows of Fig. 5.1 for the MoNuSeg are particularly distinct from each other in various aspects. These differences include the background, color composition, size of nuclei, and compactness of nuclei structures. This proves the model's ability to produce diverse and varied outputs even when trained on a relatively small set of high-resolution images.

To inspect the true effectiveness of the model distribution in the context of deep learning, it is necessary to verify that the model does not just simply memorize the training set, especially for the dataset with a limited number of data. This verification can be accomplished by first computing the mutual distance between the images used for training and the generated samples. Then, the real image with the closest distance is visually compared with the synthetic one to further check the similarity, so that the validity of the model distribution can be demonstrated if huge visual differences are witnessed.

Following [12], a similar work for retina image and vessel network synthesis, we adopt the Mutual Information Measure to analyze the distance between the training and synthetic segmentation masks. We denote the generated segmentation mask as $\tilde{s}$, and a true mask as s. The metric can be computed as follows:

$$MI(s, \tilde{s}) = \sum_{s_i \in s} \sum_{\tilde{s}_i \in \tilde{s}} p(s_i, \tilde{s}_i) \log \frac{p(s_i, \tilde{s}_i)}{p(s_i) \cdot p(\tilde{s}_i)} \tag{5.1}$$

where s_i and $\tilde{s}_i$ are the pixel values of the s and $\tilde{s}$, the $p(s_i, \tilde{s}_i)$ is the joint histogram of the true and synthetic segmentation masks, and $p(s_i)$, $p(\tilde{s}_i)$ are the marginals.

The closest sample is extracted from the training set for visual comparison. The randomly selected examples are shown in Fig 5.2.

The findings indicate that there are noticeable distinctions between synthetic images and real images, in spite of high structure similarities between the segmentation masks. This is particularly evident for polyp images and histopathological images from the MoNuSeg dataset (e.g. nuclei of different sizes, background color, etc). We will see later that the mask-conditioned synthesis (See Section 5.1.2) gives us a great diversity of images. This further reinforces the observation that synthetic samples differ from real ones.

In the case of the GlaS dataset, the synthetic images primarily differ from real images in terms of style and fine-grained details, while the main content of the images remains relatively consistent. This overall similarity might be partly attributed to the size of the training sets and the low resolution of the training images being used. The limited size of the training sets may overlook the overall interdependence among different regions of the images. Moreover, the low resolution of the training images along with the small size might miss some intricate details.

5.1.2 Mask Conditioned Synthesis

We investigate the outcomes of mask-conditioned synthesis using the JDM (See Section 3.1) for polyp and nuclei images. To demonstrate that our method goes beyond simple memorization of the training set, we apply the training mask for conditional sampling. The qualitative results can be found in Fig. 5.3.

Fig. 5.3a and 5.3b provide visual evidence that the generated images induced by the training mask exhibit significant differences compared to the original training images. The main distinctions lie in context, textures, and contrast. The substantial deviation between the generated images and the training images again indicates the validity of the model distribution, image representation, and mutual consistency.

5.1.3 Discussion

In this section, we visually evaluate the generated samples on both polyp and nuclei segmentation datasets.

The analysis reveals that the synthetic paired data generated by the JDM exhibits visual harmony and demonstrates a high level of consistency between the generated images and their associated segmentation masks. The model is capable of producing paired samples that possess both quality and diversity.

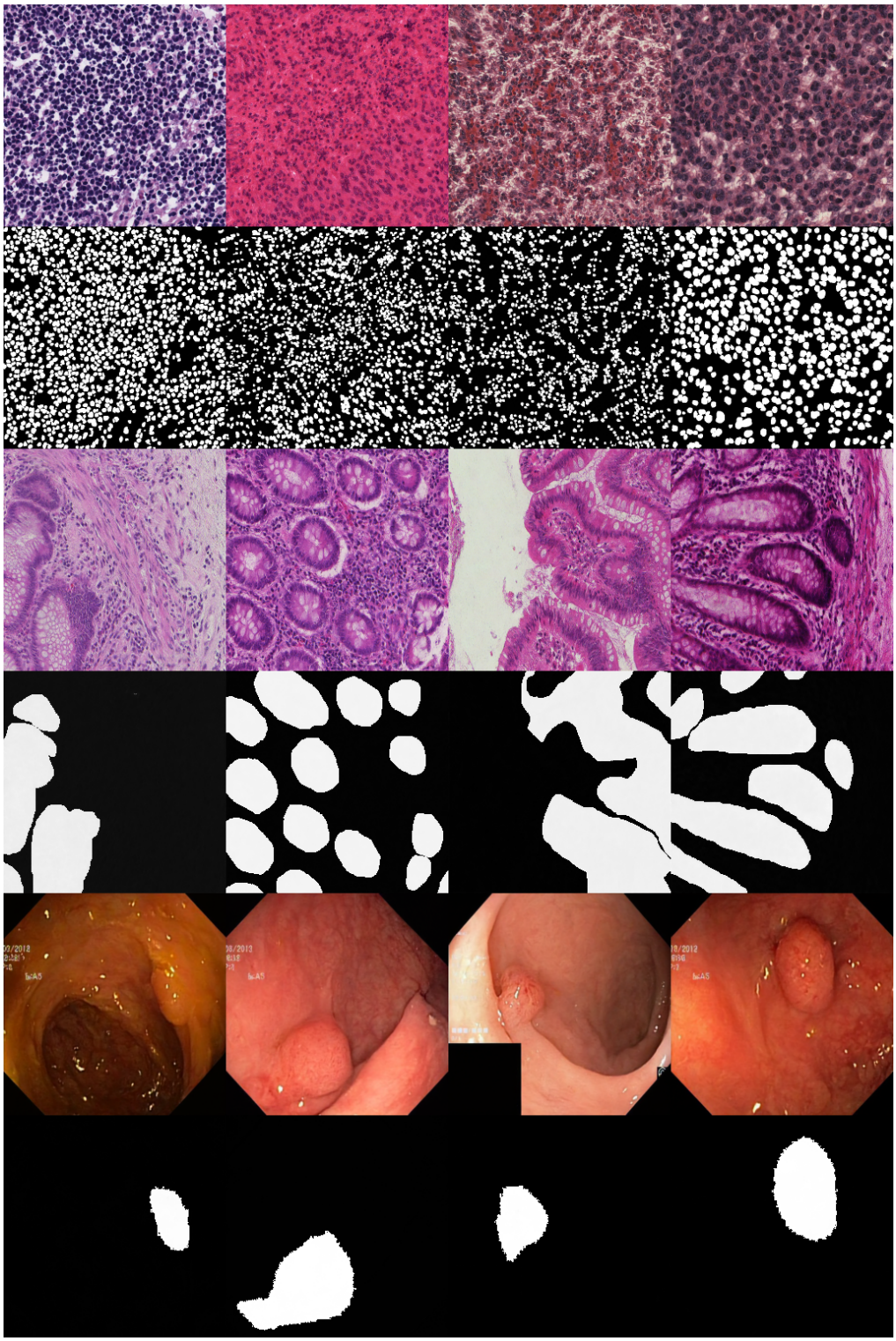


Figure 5.1: Random generated samples of nuclei and polyp images and their associated segmentation masks. All the images and masks are resized to 256 for display.

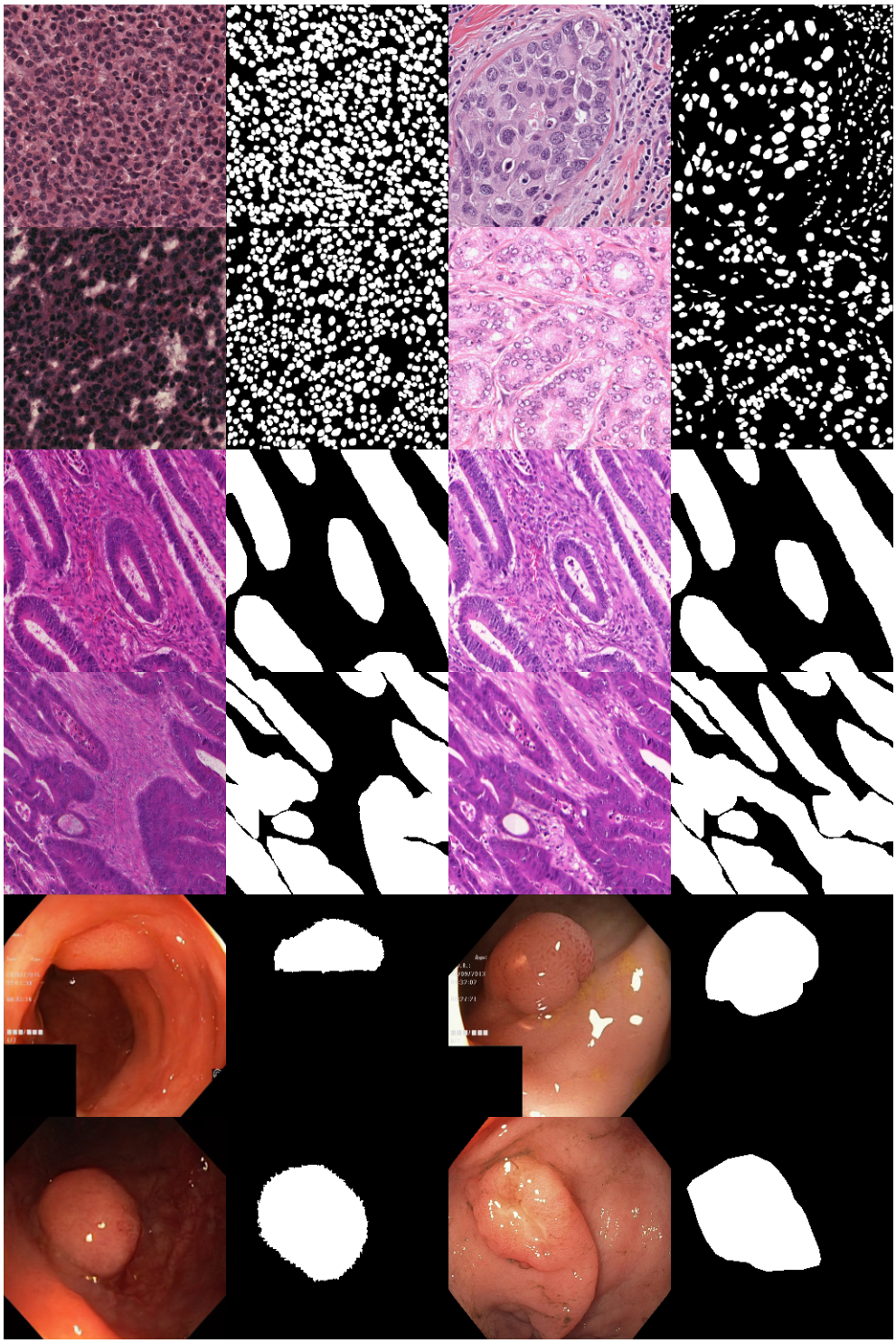
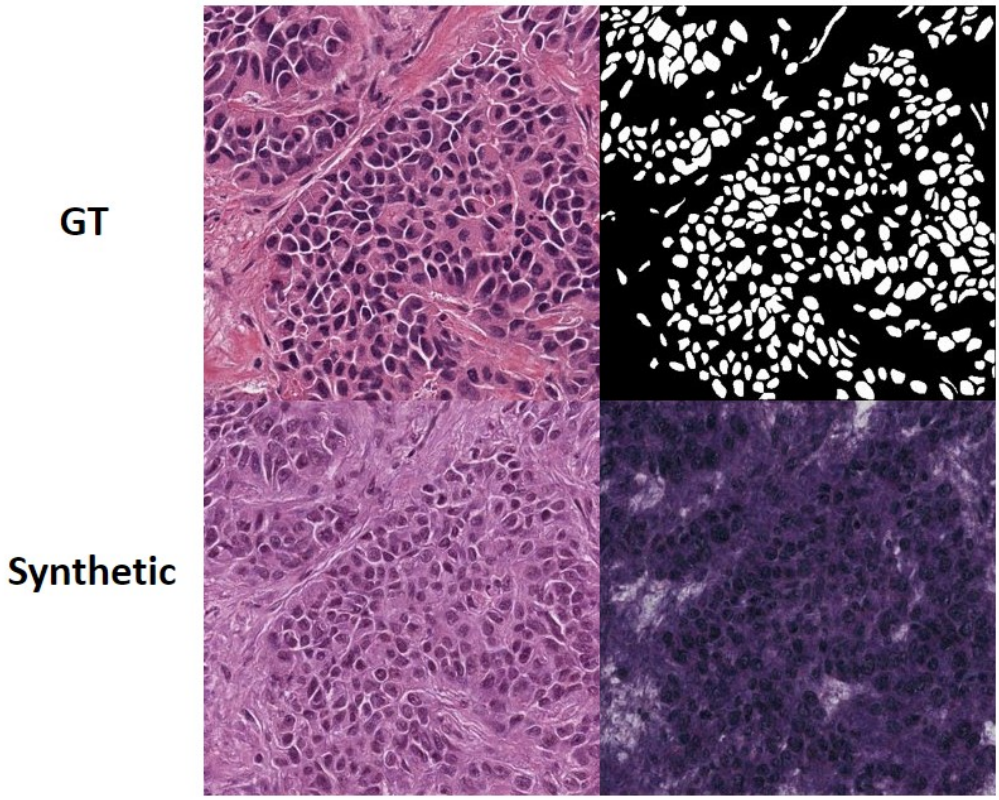
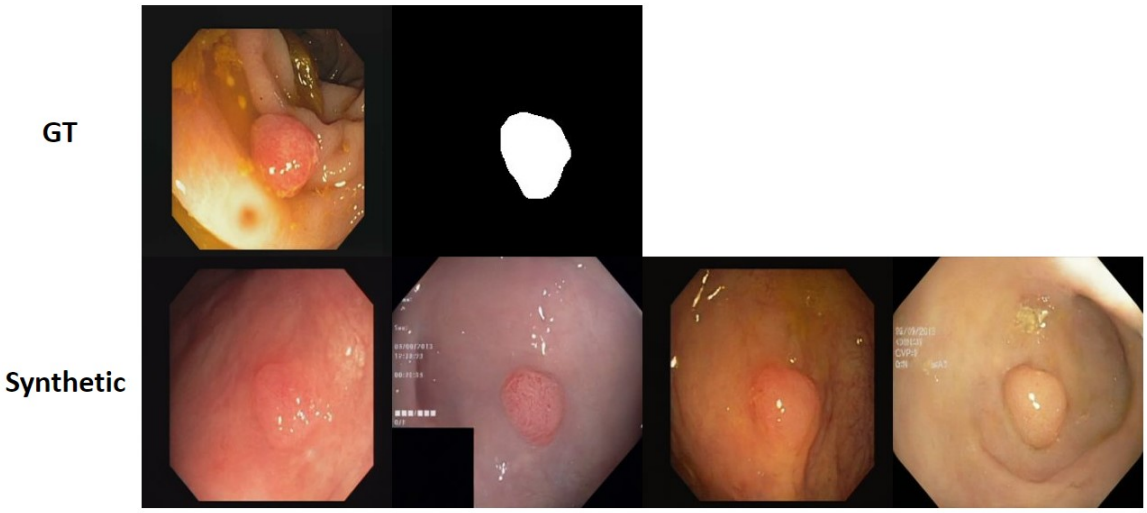


Figure 5.2: Random generated samples and closest train data regarding mutual information concerning segmentation mask. From left to right: synthetic image, associated synthetic segmentation mask, closest real image, closest real segmentation mask.



(a) Synthetic images induced by the training mask of MoNuseg [31].



(b) Synthetic images induced by the training mask of Kvasir [26].

Figure 5.3: Synthetic medical images guided by the training mask using the joint diffusion model. The significant difference in contrast to the real image demonstrates the sample diversity and model distribution.

For polyp segmentation, the generated samples appropriately position the polyps within the images, however, the boundaries of the polyps may not be entirely smooth. The model successfully avoids simple memorization of the training images, as evidenced by studying the closest sample in the training set compared to the synthetic ones. As for the *Glas* dataset, the results exhibit a high degree of structural similarity to real images. This outcome might be partly attributed to the size of the training set and the relatively low resolution of the images used, which might lower the learning difficulty of the generative model, and thus fail to capture the interdependence among different image regions. Hence, we suggest training the DDPM in a large resolution under the situation of data sparsity.

For future work, we plan to explore how the training time will impact the sample quality and diversity quantitatively. In the next section, we will investigate the generated images for training the segmentation model.

5.2 Medical Segmentation on Synthetic Data

In this section, we investigate the segmentation performance of training on purely synthetic data. Given that countless numbers of data can be produced, we compare the number of synthetic images used for training. The results can be an indication of how many numbers of images should be used for transfer learning in Section 5.3. We start from the same number of synthetic data as the original dataset, and then gradually increase the data percentage. The number of synthetic data is quantified as the proportion relative to the training set.

5.2.1 Medical Segmentation on Polyps

We set the number of training epochs to 400, as the validation curve saturates nearly after 300 epochs. The test is split into two subgroups (validation set and generalization set) for Results 1 & 2.

Result 1:

We compare the number of synthetic samples training the segmentation model. The results are presented in Table 5.1 and the first row of Fig. 5.4, in contrast to training on real data.

Note that the results are quite close to that of training on real data, achieving competitive performance on both datasets. It can be observed that both performance metrics vary with different numbers of synthetic images. Two metrics peak when

Training images	Kvasir mDice(%) mIoU(%)	Clinic-DB mDice(%) mIoU(%)
Real	**90.81 85.16**	**88.54 83.91**
Synthetic - 100%	85.87 78.71	77.40 70.66
Synthetic - 150%	86.50 79.76	77.46 70.28
Synthetic - 200%	86.36 79.60	77.69 71.06
Synthetic - 250%	<u>87.08</u>(-3.73) <u>80.41</u>(-4.75)	78.01 71.49
Synthetic - 300%	86.00 78.96	<u>78.21</u>(-10.33) <u>71.70</u>(-12.21)
Synthetic - 400%	84.52 77.28	77.19 70.39
Synthetic - 500%	83.84 76.69	77.93 71.35

Table 5.1: Results on validation sets of polyp segmentation. (# Synthetic - %) represents the number of synthetic data used in training the model. The first and second best results are in **bold** and <u>underlined</u>, respectively. The number in blue quantifies the performance difference in contrast to the result on real data.

250% or 300% of images are leveraged for training and decline afterwards (See the first row of Fig. 5.4). The variation of the number of synthetic data is more sensitive to the Kvasir rather than the Clinic-DB with larger performance drops is witness. Besides, the performance gap between the synthetic and real data is much smaller for Kvasir compared with that of Clinic-DB, with the best case (250% and 300%, respectively) of 3.73% and 10.33% differences in terms of mDice. This suggests that the model distribution is more inclined to approximate the distribution of Kvasir instead of Clinic-DB.

Result 2:

The results of the generalization sets can be found in Table 5.2 and the last two rows of Fig. 5.4.

It is unexpected that the best outcome (Synthetic 100%) is achieved with the same number of real images (100%). Generally, Fig. 5.4 indicates a trend of decline as more synthetic data is used to train the model. All three testsets experience a small bump when the number of data is 250%. Similar to the validation set, both mDice and mIoU decrease as the number of images exceeds 250%. Additionally, the performance drop is huge for ETIS of 8.2% as opposed to 4.4% of Colon-DB and 3.2% of CVC-300.

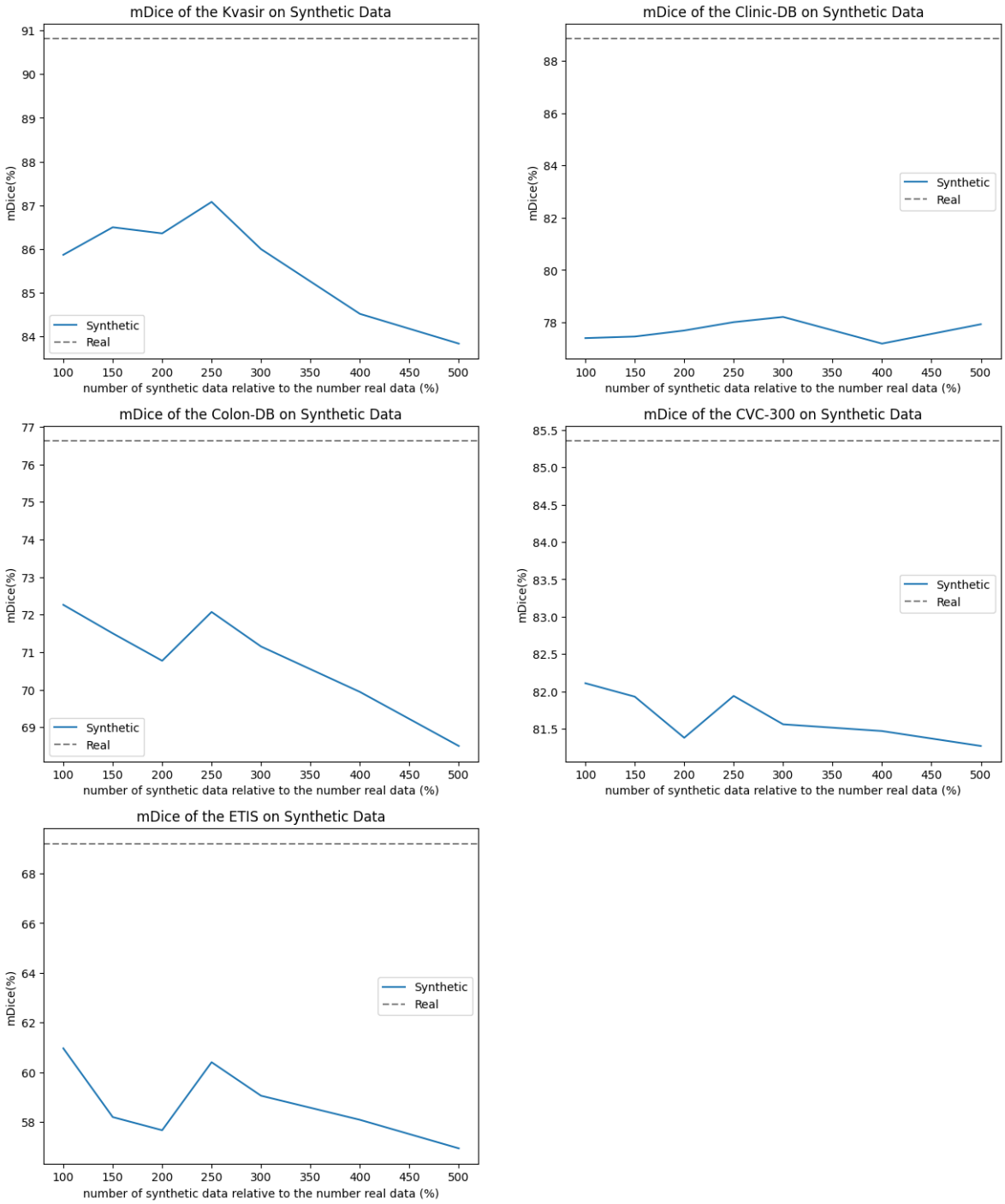


Figure 5.4: mDice vs. the number of synthetic data. The dashed line denotes the result of training on real data. From left to right and top to down: Kvasir, Clinic-DB, Colon-DB, CVC-300, ETIS.

Training Images	Colon-DB mDice(%) mIoU(%)	CVC-300 mDice(%) mIoU(%)	ETIS mDice(%) mIoU(%)
Real	**76.62 69.39**	**85.35 77.84**	**69.21 62.14**
Synthetic - 100%	<u>72.26</u> <u>65.53</u> (-4.36) (-3.86)	<u>82.11</u> <u>74.56</u> (-3.24) (-3.28)	<u>60.96</u> <u>53.28</u> (-8.25) (-8.86)
Synthetic - 150%	71.50 64.77	81.93 74.24	58.19 51.22
Synthetic - 200%	70.77 64.10	81.38 73.63	57.66 50.60
Synthetic - 250%	72.07 65.44	81.94 74.28	60.40 53.18
Synthetic - 300%	71.15 64.25	81.56 73.67	59.05 51.75
Synthetic - 400%	69.94 63.15	81.47 73.16	58.08 50.43
Synthetic - 500%	68.50 62.14	81.27 73.41	56.93 49.14

Table 5.2: Results on generalization sets of polyp segmentation. (# Synthetic - %) represents the number of synthetic data used in training the model. The first and second best results are in **bold** and <u>underlined</u>, respectively. The number in blue quantifies the performance difference compared to the result on real data.

5.2.2 Medical Segmentation on Nuclei

We set the number of epochs by default to be 1000 as the computational cost is bearable and the general trend of the validation curve stabilizes only when the number of training epochs is from 900 to 1000. Other hyperparameter setting is determined by the nnU-Net [25] without checking.

Result:

We perform an ablation study on the number of synthetic data same as the Section 5.2.1. The results of these experiments are presented in Table 5.3 and visualized in Fig. 5.5.

Table 5.3 demonstrates that the segmentation performance is only marginally lower than the real data counterpart (with a difference of 2.44% and 3.32% for mDice and mIoU, respectively) when training with 150% of synthetic data. This finding suggests that the JDM can effectively model complex data distributions even when the available data is limited.

It is further illustrated in Fig. 5.5 about the segmentation model's robustness, showing a slight tendency of performance drop as the number of synthetic data increases, but overall it remains resilient. Notably, it is observed that the mDice for the case of using 100% synthetic data is significantly lower compared to the results obtained on the real data, indicating potential challenges in modeling the

Training images	MoNuSeg mDice(%)	MoNuSeg mIoU(%)
Real	**81.68**	**69.10**
Synthetic - 100%	75.47	61.15
Synthetic - 150%	<u>79.24</u>(-2.44)	<u>65.78</u>(-3.32)
Synthetic - 200%	77.71	63.91
Synthetic - 250%	78.26	64.57
Synthetic - 300%	78.30	64.74
Synthetic - 400%	78.32	64.91
Synthetic - 500%	78.24	64.68
Synthetic - 1000%	77.56	63.77

Table 5.3: Results on MoNuSeg of nuclei segmentation. (# Synthetic - %) represents the number of synthetic data used in training the model. The first and second best results are in **bold** and <u>underlined</u>, respectively. The number in blue quantifies the performance difference in contrast to the result on real data.

small dataset.

5.2.3 Discussion

In this section, we evaluate the segmentation model's performance using various quantities of synthetic images. The experiments are conducted by training the model exclusively on synthetic data for two tasks: polyp segmentation and nuclei segmentation.

When a model is trained using synthetic data, its overall performance is encouraging, surpassing that of a randomly initialized model, with only a minor single-digit percentage difference compared to real data. This again demonstrates the effectiveness of the model distribution, approximating the data distribution. However, as the quantity of synthetic data increases, the model's performance begins to decline, albeit at varying rates depending on the datasets used. This decline can be attributed to the growing deviation between the imperfect model distribution and the true data distribution caused by the increased use of synthetic data. In practice, the optimal "sweet spot" for the number of data points lies within the range of 100% to 300% of the original dataset, and its exact value should be determined empirically.

Regarding the segmentation gap observed between real and synthetic data for polyp segmentation, it is noteworthy that the performance drop is more pronounced on the validation set especially on the Clinic-DB dataset compared to the generalization

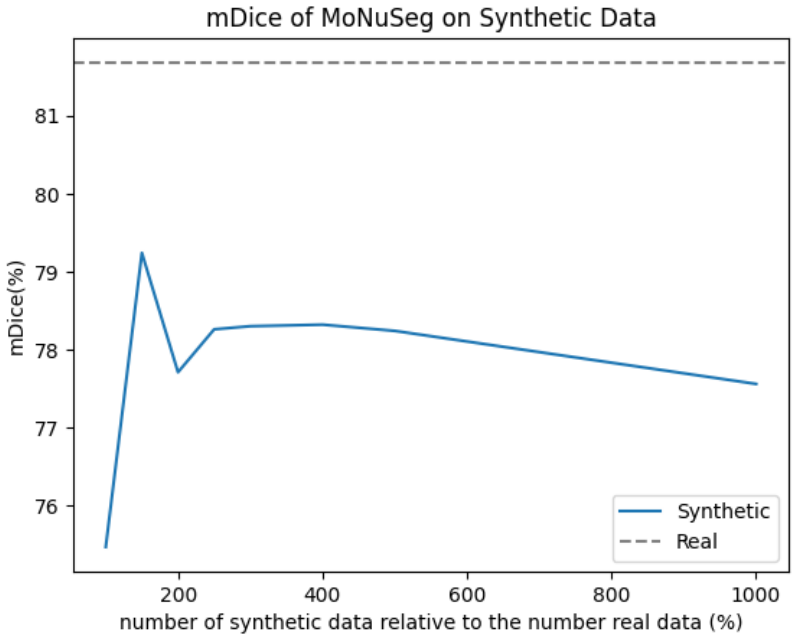


Figure 5.5: mDice vs. the number of synthetic data on MoNuSeg. The dashed line denotes the result of training on real data.

set, even though the generative model is partially constructed using Clinic-DB data. We explain this phenomenon to the fact that the model distribution is actually a mixture of data distributions of the Kvasir and Clinic-DB datasets. However, it leans much closer to the distribution of Kvasir. Despite incorporating Clinic-DB data in the generative model, the synthetic data still exhibits a more significant performance drop due to the dominant influence of the Kvasir data distribution in the mixture model. Furthermore, this mixture model distribution acts as a mitigating factor to some extent, reducing the gap between the validation set and the generalization set, leading to better performance on the generalization set rather than on the Clinic-DB. This finding sheds light on the intricacies of data distributions and their impact on the performance of generative models.

For future work, we propose investigating the potential benefits of training a new generative model based on a different mixture of training data. By leveraging diverse datasets and possibly different combinations, this new generative model could offer valuable insights into improving the usability of synthetic data to generalize better on various datasets. In the next section, we focus on evaluating the effectiveness of synthetic data through the lens of transfer learning.

5.3 Transfer Learning on Synthetic Data

In this section, we delve deeper into finding an effective strategy to enhance segmentation performance using synthetic data. The approach involves first pretraining on the synthetic data and finetuning on the real data (See Section 4.1).

5.3.1 Transfer Learning on Polyps

We compare the number of data for pretraining at a linear rate, using proportions of 100%, 250%, and 500% of synthetic data. The pretrained checkpoints utilized in these experiments are derived from the segmentation model in Section 5.2.1. The fine-tuning process involves training the model on real data for 150 epochs. To determine the optimal learning rate for fine-tuning, we perform a grid search and find that a learning rate of 1e-2 yields the best results. We carry out the training and fine-tuning for all five folds, employing the same pretraining checkpoint. The evaluation of these experiments is presented in Table 5.4 for the validation set and in Table 5.5 for the generalization set. The experiments are conducted five times and compute the average. Again, the results are illustrated in Results 1 and Results 2, respectively.

Result 1:

The noticeable improvements in model performance are evident from Table 5.4, showcasing the benefits of pretraining with synthetic data. In general, the segmentation performance surpasses that of PraNet and narrows the performance gap with CaraNet, highlighting the potential of properly utilizing synthetic data to bridge the differences with task-specific networks.

While the enhancement on the Kvasir [26] is relatively marginal, there is a substantial increase in performance on Clinic-DB [4], with a remarkable improvement of +3.31% and +3.27% observed when leveraging 500% of synthetic data. As anticipated, models pretrained with larger amounts of synthetic data generally tend to perform better.

It is worth noting that models pretrained with 250% of synthetic data perform worse than those pretrained with only 100%. This result suggests that there may be an optimal amount of synthetic data for pretraining, and exceeding that threshold could have an unexpected impact on model performance. Hence, careful consideration of the quantity of synthetic data used for pretraining is essential for achieving optimal results.

Methods	# Syn. Pretrain	Kvasir mDice(%) mIoU(%)	Clinic-DB mDice(%) mIoU(%)
U-Net [44]	-	81.80 74.60	82.30 75.50
ResUNet++	-	81.30 79.30	79.60 79.60
PraNet [15]	-	89.80 84.00	89.90 84.90
CaraNet [24]	-	**91.80 86.50**	**93.60 88.70**
nnU-Net	-	90.81 85.16	88.54 83.91
nnU-Net	100%	91.02 85.60	90.21 85.61
nnU-Net	250%	90.81 85.51	89.31 84.87
nnU-Net	500%	<u>91.22</u>(+0.41) <u>85.89</u>(+0.73)	<u>91.85</u>(+3.31) <u>87.18</u>(+3.27)
nnU-Net	1000%	91.09 85.77	91.55 86.96

Table 5.4: Results on the validation set pretrain on a different number of synthetic data. (# Syn. Pretrain) represents the number of synthetic data used pretraining the model. The first and second best results are in **bold** and <u>underlined</u>, respectively. The number in blue quantifies the performance difference in contrast to the result on real data.

Result 2:

Table 5.5 presents the results of the generalization set, demonstrating that pretraining enhances the segmentation model's ability to generalize. The pretrained nnU-Net outperforms PraNet on all three datasets and performs comparably to CaraNet on the challenging dataset ETIS [49]. Specifically, on Colon-DB [4], the pretrained nnU-Net outperforms the task-specific network, and a notable increase of +5.4% is observed on ETIS when utilizing 500% of pretraining data. The improvement is particularly significant for ETIS, which contains numerous images with small medical objects, showing how pretraining with diverse synthetic datasets enhances the recognition of such small objects. Figure 5.6 visually demonstrates a more accurate segmentation of small polyps after pretraining. Notably, the most substantial increase in performance for CVC-300 [60] is achieved when using a 250% pretraining data increment. These findings further disclose that the improvement achieved through pretraining is influenced by the specific dataset and the quantity of synthetic data employed.

5.3.2 Transfer Learning on Nuclei

Following the section 5.3.1, we conduct the experiments using 100%, 250%, 500% and 1000% percent of synthetic data and checkpoints from section 5.2.2. We set the training epoch to 100. The grid search of the learning rate yields 5e-6 as the

Methods	# Syn. Pretrain	Colon-DB mDice(%) mIoU(%)	CVC-300 mDice(%) mIoU(%)	ETIS mDice(%) mIoU(%)
U-Net [44]	-	51.20 44.40	71.00 62.70	39.80 33.50
PraNet [15]	-	70.90 64.00	87.10 79.71	62.80 56.70
CaraNet [24]	-	77.30 68.90	**90.30 83.80**	**74.70** <u>67.20</u>
nnU-Net	-	76.62 69.39	85.45 77.84	69.21 62.14
nnU-Net	100%	77.58 70.80	87.49 80.81	71.71 65.06
nnU-Net	250%	<u>77.90</u> **71.16**	<u>87.61</u> <u>81.09</u>	72.10 65.21
nnU-Net	500%	**78.19** <u>70.89</u>	86.73 79.93	<u>74.61</u> **67.67**
nnU-Net	1000%	77.65 70.84	87.05 80.46	73.63 66.68
		(+1.57) (+1.50)	(+2.16) (+3.25)	(+5.40) (+5.53)

Table 5.5: Results on the generalization set pretrained on a different number of synthetic data. (# Syn. Pretrain) represents the number of synthetic data used pretraining the model. The first and second best results are in **bold** and <u>underlined</u>, respectively. The number in blue quantifies the performance difference in contrast to the result on real data.

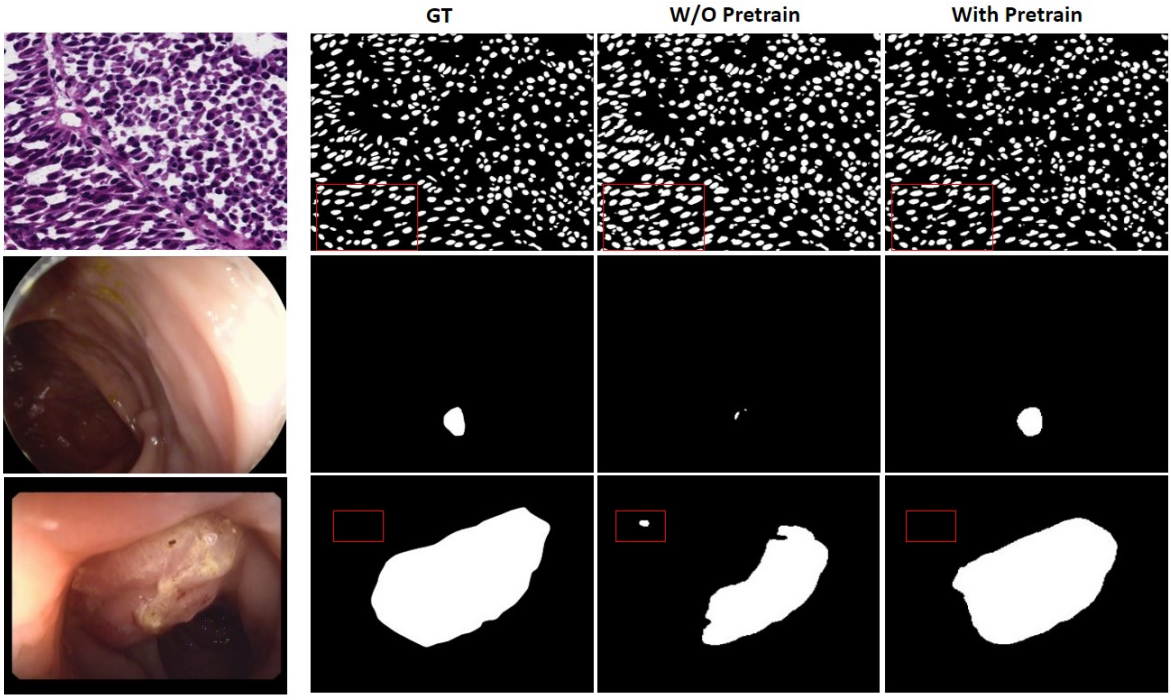


Figure 5.6: Quliative results. The red box highlights the improvement of the pretrained segmentation model on synthetic data. The results are taken from the best case on ETIS [49] and MoNuSeg [31].

Methods	# Syn. Pretrain	MoNuSeg mDice(%)	MoNuSeg mIoU(%)
U-Net++	-	81.17	69.29
Attention U-Net	-	79.74	67.19
MedT	-	79.55	66.17
UCTransUnet	-	80.73	68.42
SMESwin Unet	-	81.13	68.35
nnU-Net	-	81.68	69.09
nnU-Net	100%	81.13	68.36
nnU-Net	250%	81.57	68.98
nnU-Net	500%	<u>81.80</u>	<u>69.30</u>
nnU-Net	1000%	**82.31**(+0.51)	**70.03**(+0.73)

Table 5.6: Results on MoNuSeg [31]. (# Syn. Pretrain) represents the number of synthetic data used pretraining the model. The first and second best results are in **bold** and <u>underlined</u>, respectively. The number in blue quantifies the performance difference in contrast to the result on real data.

optimal value. The experiments are done five times and compute the average. The result can be found in Table 5.6.

Result:

As shown in Table 5.6, nnU-Net has already appeared as one of the top-performing methods compared with other methods on this dataset. Further leveraging the power of pretraining on a much larger dataset comprising ten times more synthetic data, a new state-of-the-art result is achieved with 0.51% and 0.73% increase on mDice and mIoU. Visualization of the first row of Fig. 5.6 shows that the pretrained model excels at capturing fine-grained details present in the medical images, outperforming models trained directly on real data.

However, note that the benefits of pretraining are not universally consistent. When the amount of pretrain synthetic data is kept relatively small (100% and 150%), the performance of the segmentation model actually degrades compared to training on real data alone. We explain this degradation to the scale of the synthetic dataset (only 30 images for 100% as opposed to the 100% of Kvasir consisting of 1450 images). With insufficient data for pretraining, the model fails to learn the representation for recognizing the image features, leading to overfitting on the synthetic data and imperfect initialization.

5.3.3 Discussion

In this section, we further explore the effectiveness of the synthetic data through supervised transfer learning on polyps and nuclei segmentation testsets. We also compare the number of synthetic data used for pretraining.

We found a substantial overall increase in performance following pretraining with synthetic data. Moreover, there is a consistent trend of performance improvement with the utilization of more synthetic data, up to a certain point where further increases show diminishing returns. In essence, pretraining with synthetic data proves to be an effective strategy for enhancing model performance. As the quantity of synthetic data used for pretraining increases, the performance of the model tends to improve steadily. However, this improvement reaches a saturation point, beyond which additional synthetic data does not yield significant gains in performance. Thus, there is an optimal balance to be struck in terms of the amount of synthetic data employed to achieve the best results.

On ETIS, the model is significantly increased by recognizing small medical objects. We explain this by better feature perception and more small objects being seen after pretraining. On MoNuSeg, the model outperforms all state-of-arts in Table 5.6. We believe that the generated samples from the JDM alleviate the problem of imbalanced examples in the training set and enhance the diversity of the training data.

For future work, we aim to delve into more advanced strategies to fully harness the potential of synthetic data, such as a student teacher model to enhance the segmentation model.

6

Conclusion and Limitations

Diffusion models (DMs) have incited the latest trend in image synthesis for their great sample quality and diversity, accomplished by moving a Gaussian sample to the data distribution through progressive denoising. This work explores the possibility of DMs as a data augmentation tool for biomedical segmentation.

A theoretical framework Joint Diffusion Model (JDM) is proposed to synthesize paired samples. It is proved that the joint modeling resembles the classical DMs for conditional sampling and is orthogonal to various methods of solving inverse problems using the DMs, including image inpainting, super-resolution, colorization, and etc. The qualitative results on the medical image segmentation dataset present the realness and diversity of the synthetic samples, demonstrating the capability of the JDM even with only a small amount of data samples. Using the segmentation mask for training to perform conditional sampling and identifying the closest matching sample from the training set, we have confirmed that the JDM effectively learns the underlying data distribution rather than merely memorizing the specific training data.

We further investigate using synthetic data for biomedical segmentation. It is discovered that training with synthetic data can lead to relatively comparable segmentation accuracy to real data. However, increasing the amount of synthetic data may cause performance to decline. We believe this is caused by the growing deviation between the model distribution and the data distribution. This deviation affects segmentation performance differently across various datasets.

For transfer learning, the pipeline involves pretraining on synthetic data and then finetuning on real data. This process improves segmentation performance by emphasizing sample diversity, which helps the model recognize fine-grained image features effectively. Implicit likelihood training of the DMs encourages the model to generate diverse samples, and thus balance the training data, leading to better generalization and robustness of the segmentation model.

6.0.1 Limitations and Future Work

Joint modeling poses a severe challenge to the network architecture as the number of modalities increases. Hence, it still remains unclear whether the method is solid in the context of multi-class biomedical segmentation, which can be a possible direction for future work.

Besides, as the quantity of pretrained data increases, the performance shows gradual improvement but eventually plateaus to some extent. Hence, we cannot judge how much synthetic data should be leveraged for supervised pretraining and need be found out in the practical context. Another disadvantage of the DMs is the sampling speed, as it takes thousands of network evaluations to produce samples. Considering this, we plan first to explore the teacher-student model to fully unleash the potential of synthetic data for biomedical segmentation. To address the issue of slow sampling speed, we plan to explore and implement several accelerated methods, such as TimeStepRespacing [41] and DDIM [53]. These methods will be investigated to evaluate their effectiveness in terms of sample quality and diversity for joint modeling.

Bibliography

[1] Amit, T., Shaharbany, T., Nachmani, E. & Wolf, L. Segdiff: Image segmentation with diffusion probabilistic models. *ArXiv Preprint ArXiv:2112.00390*. (2021)

[2] Baid, U., Ghodasara, S., Mohan, S., Bilello, M., Calabrese, E., Colak, E., Farahani, K., Kalpathy-Cramer, J., Kitamura, F., Pati, S. & Others The rsna-asnr-miccai brats 2021 benchmark on brain tumor segmentation and radiogenomic classification. *ArXiv Preprint ArXiv:2107.02314*. (2021)

[3] Benny, Y. & Wolf, L. Dynamic Dual-Output Diffusion Models. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 11482-11491 (2022)

[4] Bernal, J., Sánchez, F., Fernández-Esparrach, G., Gil, D., Rodrıguez, C. & Vilariño, F. WM-DOVA maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians. *Computerized Medical Imaging And Graphics*. **43** pp. 99-111 (2015)

[5] Bushberg, J. & Boone, J. The essential physics of medical imaging. (Lippincott Williams Wilkins,2011)

[6] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A. & Zagoruyko, S. End-to-end object detection with transformers. *European Conference On Computer Vision*. pp. 213-229 (2020)

[7] Chen, L., Papandreou, G., Schroff, F. & Adam, H. Rethinking atrous convolution for semantic image segmentation. *ArXiv Preprint ArXiv:1706.05587*. (2017)

[8] Chen, N., Zhang, Y., Zen, H., Weiss, R., Norouzi, M. & Chan, W. Wavegrad: Estimating gradients for waveform generation. *ArXiv Preprint*

ArXiv:2009.00713. (2020)

[9] Chung, H., Kim, J., Mccann, M., Klasky, M. & Ye, J. Diffusion posterior sampling for general noisy inverse problems. *ArXiv Preprint ArXiv:2209.14687*. (2022)

[10] Chung, H., Sim, B., Ryu, D. & Ye, J. Improving Diffusion Models for Inverse Problems using Manifold Constraints. *Advances In Neural Information Processing Systems*. (2022)

[11] Choi, J., Kim, S., Jeong, Y., Gwon, Y. & Yoon, S. Ilvr: Conditioning method for denoising diffusion probabilistic models. *ArXiv Preprint ArXiv:2108.02938*. (2021)

[12] Costa, P., Galdran, A., Meyer, M., Niemeijer, M., Abràmoff, M., Mendonça, A. & Campilho, A. End-to-end adversarial retinal image synthesis. *IEEE Transactions On Medical Imaging*. **37**, 781-791 (2017)

[13] Dinh, L., Krueger, D. & Bengio, Y. Nice: Non-linear independent components estimation. *ArXiv Preprint ArXiv:1410.8516*. (2014)

[14] Dhariwal, P. & Nichol, A. Diffusion models beat gans on image synthesis. *Advances In Neural Information Processing Systems*. **34** pp. 8780-8794 (2021)

[15] Fan, D., Ji, G., Zhou, T., Chen, G., Fu, H., Shen, J. & Shao, L. Pranet: Parallel reverse attention network for polyp segmentation. *Medical Image Computing And Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part VI 23*. pp. 263-273 (2020)

[16] Germain, M., Gregor, K., Murray, I. & Larochelle, H. Made: Masked autoencoder for distribution estimation. *International Conference On Machine Learning*. pp. 881-889 (2015)

[17] Goodfellow, I., Bengio, Y. & Courville, A. Deep learning. (MIT press,2016)

[18] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. Generative adversarial networks. *Communications Of The ACM*. **63**, 139-144 (2020)

[19] Hatamizadeh, A., Nath, V., Tang, Y., Yang, D., Roth, H. & Xu, D. Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images. *International MICCAI Brainlesion Workshop*. pp. 272-284 (2021)

[20] He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*. pp. 770-778 (2016)

[21] Hendrycks, D. & Gimpel, K. Gaussian error linear units (gelus). *ArXiv Preprint ArXiv:1606.08415*. (2016)

[22] Ho, J., Jain, A. & Abbeel, P. Denoising diffusion probabilistic models. *Advances In Neural Information Processing Systems*. **33** pp. 6840-6851 (2020)

[23] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B. & Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances In Neural Information Processing Systems*. **30** (2017)

[24] Lou, A., Guan, S., Ko, H. & Loew, M. CaraNet: context axial reverse attention network for segmentation of small medical objects. *Medical Imaging 2022: Image Processing*. **12032** pp. 81-92 (2022) ´

[25] Isensee, F., Petersen, J., Klein, A., Zimmerer, D., Jaeger, P., Kohl, S., Wasserthal, J., Koehler, G., Norajitra, T., Wirkert, S. & Others nnu-net: Self-adapting framework for u-net-based medical image segmentation. *ArXiv Preprint ArXiv:1809.10486*. (2018)

[26] Jha, D., Smedsrud, P., Riegler, M., Halvorsen, P., Lange, T., Johansen, D. & Johansen, H. Kvasir-seg: A segmented polyp dataset. *MultiMedia Modeling: 26th International Conference, MMM 2020, Daejeon, South Korea, January 5–8, 2020, Proceedings, Part II 26*. pp. 451-462 (2020)

[27] Kawar, B., Elad, M., Ermon, S. & Song, J. Denoising diffusion restoration models. *ArXiv Preprint ArXiv:2201.11793*. (2022)

[28] Kingma, D. & Welling, M. Auto-encoding variational bayes. *ArXiv Preprint ArXiv:1312.6114*. (2013)

[29] Krizhevsky, A., Hinton, G. & Others Learning multiple layers of features from tiny images. (Toronto, ON, Canada,2009)

[30] Kullback, S. & Leibler, R. On information and sufficiency. *The Annals Of Mathematical Statistics*. **22**, 79-86 (1951)

[31] Kumar, N., Verma, R., Anand, D., Zhou, Y., Onder, O., Tsougenis, E., Chen, H., Heng, P., Li, J., Hu, Z., Wang, Y., Koohbanani, N., Jahanifar, M., Tajeddin, N., Gooya, A., Rajpoot, N., Ren, X., Zhou, S., Wang, Q., Shen, D., Yang, C., Weng, C., Yu, W., Yeh, C., Yang, S., Xu, S., Yeung, P., Sun, P., Mahbod, A., Schaefer, G., Ellinger, I., Ecker, R., Smedby, O., Wang, C., Chidester, B., Ton, T., Tran, M., Ma, J., Do, M., Graham, S., Vu, Q., Kwak, J., Gunda, A., Chunduri, R., Hu, C., Zhou, X., Lotfi, D., Safdari, R., Kascenas, A., O'Neil, A., Eschweiler, D., Stegmaier, J., Cui, Y., Yin, B., Chen, K., Tian, X., Gruening, P., Barth, E., Arbel, E., Remer, I., Ben-Dor, A., Sirazitdinova, E., Kohl, M., Braunewell, S., Li, Y., Xie, X., Shen, L., Ma, J., Baksi, K., Khan, M., Choo, J., Colomer, A., Naranjo, V., Pei, L., Iftekharuddin, K., Roy, K., Bhattacharjee, D., Pedraza, A., Bueno, M., Devanathan, S., Radhakrishnan, S., Koduganty, P., Wu, Z., Cai, G., Liu, X., Wang, Y. & Sethi, A. A Multi-Organ Nucleus Segmentation Challenge. *IEEE Transactions On Medical Imaging*. **39**, 1380-1391 (2020)

[32] Kong, Z., Ping, W., Huang, J., Zhao, K. & Catanzaro, B. Diffwave: A versatile diffusion model for audio synthesis. *ArXiv Preprint ArXiv:2009.09761*. (2020)

[33] Liu, Z., Luo, P., Wang, X. & Tang, X. Deep Learning Face Attributes in the Wild. *Proceedings Of International Conference On Computer Vision (ICCV)*. (2015,12)

[34] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S. & Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. *Proceedings Of The IEEE/CVF International Conference On Computer Vision*. pp. 10012-10022 (2021)

[35] Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R. & Van Gool, L. Repaint: Inpainting using denoising diffusion probabilistic models. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 11461-11471 (2022)

[36] Mitchell, T. Machine Learning. (McGraw Hill,1997)

[37] Mittal, G., Engel, J., Hawthorne, C. & Simon, I. Symbolic music generation

with diffusion models. *ArXiv Preprint ArXiv:2103.16091*. (2021)

[38] Oktay, O., Schlemper, J., Folgoc, L., Lee, M., Heinrich, M., Misawa, K., Mori, K., McDonagh, S., Hammerla, N., Kainz, B. & Others Attention u-net: Learning where to look for the pancreas. *ArXiv Preprint ArXiv:1804.03999*. (2018)

[39] Papamakarios, G., Pavlakou, T. & Murray, I. Masked autoregressive flow for density estimation. *Advances In Neural Information Processing Systems*. **30** (2017)

[40] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L. & Lerer, A. Automatic differentiation in pytorch. (2017)

[41] Nichol, A. & Dhariwal, P. Improved denoising diffusion probabilistic models. *International Conference On Machine Learning*. pp. 8162-8171 (2021)

[42] Noh, H., Hong, S. & Han, B. Learning deconvolution network for semantic segmentation. *Proceedings Of The IEEE International Conference On Computer Vision*. pp. 1520-1528 (2015)

[43] Razavi, A., Oord, A. & Vinyals, O. Generating diverse high-fidelity images with vq-vae-2. *Advances In Neural Information Processing Systems*. **32** (2019)

[44] Ronneberger, O., Fischer, P. & Brox, T. U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing And Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. pp. 234-241 (2015)

[45] Rombach, R., Blattmann, A., Lorenz, D., Esser, P. & Ommer, B. High-resolution image synthesis with latent diffusion models. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 10684-10695 (2022)

[46] Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. & Norouzi, M. Image super-resolution via iterative refinement. *IEEE Transactions On Pattern Analysis And Machine Intelligence*. (2022)

[47] Salimans, T., Karpathy, A., Chen, X. & Kingma, D. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *ArXiv Preprint ArXiv:1701.05517*. (2017)

[48] Shen, Z., Zhang, M., Zhao, H., Yi, S. & Li, H. Efficient attention: Attention with linear complexities. *Proceedings Of The IEEE/CVF Winter Conference On Applications Of Computer Vision*. pp. 3531-3539 (2021)

[49] Silva, J., Histace, A., Romain, O., Dray, X. & Granado, B. Toward embedded detection of polyps in wce images for early diagnosis of colorectal cancer. *International Journal Of Computer Assisted Radiology And Surgery*. **9** pp. 283-293 (2014)

[50] Sirinukunwattana, K., Pluim, J., Chen, H., Qi, X., Heng, P., Guo, Y., Wang, L., Matuszewski, B., Bruni, E., Sanchez, U. & Others Gland segmentation in colon histology images: The glas challenge contest. *Medical Image Analysis*. **35** pp. 489-502 (2017)

[51] Skandarani, Y., Jodoin, P. & Lalande, A. Gans for medical image synthesis: An empirical study. *Journal Of Imaging*. **9**, 69 (2023)

[52] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N. & Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. *International Conference On Machine Learning*. pp. 2256-2265 (2015)

[53] Song, J., Meng, C. & Ermon, S. Denoising diffusion implicit models. *ArXiv Preprint ArXiv:2010.02502*. (2020)

[54] Song, Y. & Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances In Neural Information Processing Systems*. **32** (2019)

[55] Song, Y., Sohl-Dickstein, J., Kingma, D., Kumar, A., Ermon, S. & Poole, B. Score-based generative modeling through stochastic differential equations. *ArXiv Preprint ArXiv:2011.13456*. (2020)

[56] Song, Y., Shen, L., Xing, L. & Ermon, S. Solving inverse problems in medical imaging with score-based generative models. *ArXiv Preprint ArXiv:2111.08005*. (2021)

[57] Tajbakhsh, N., Gurudu, S. & Liang, J. Automated polyp detection in colonoscopy videos using shape and context information. *IEEE Transactions On Medical Imaging*. **35**, 630-644 (2015)

[58] Valanarasu, J., Oza, P., Hacihaliloglu, I. & Patel, V. Medical transformer: Gated axial-attention for medical image segmentation. *Medical Image Computing And Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part I 24*. pp. 36-46 (2021)

[59] Van Den Oord, A., Kalchbrenner, N. & Kavukcuoglu, K. Pixel recurrent neural networks. *International Conference On Machine Learning*. pp. 1747-1756 (2016)

[60] Vázquez, D., Bernal, J., Sánchez, F., Fernández-Esparrach, G., López, A., Romero, A., Drozdzal, M., Courville, A. & Others A benchmark for endoluminal scene segmentation of colonoscopy images. *Journal Of Healthcare Engineering*. **2017** (2017)

[61] Wang, H., Cao, P., Wang, J. & Zaiane, O. Uctransnet: rethinking the skip connections in u-net from a channel-wise perspective with transformer. *Proceedings Of The AAAI Conference On Artificial Intelligence*. **36**, 2441-2449 (2022)

[62] Wang, Z., Min, X., Shi, F., Jin, R., Nawrin, S., Yu, I. & Nagatomi, R. SMESwin Unet: Merging CNN and transformer for medical image segmentation. *International Conference On Medical Image Computing And Computer-Assisted Intervention*. pp. 517-526 (2022)

[63] Wu, Y. & He, K. Group normalization. *Proceedings Of The European Conference On Computer Vision (ECCV)*. pp. 3-19 (2018)

[64] Wolleb, J., Sandkühler, R., Bieder, F., Valmaggia, P. & Cattin, P. Diffusion models for implicit image segmentation ensembles. *International Conference On Medical Imaging With Deep Learning*. pp. 1336-1348 (2022)

[65] Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S. & Tang, J. GeoDiff: A Geometric Diffusion Model for Molecular Conformation Generation. *International Conference On Learning Representations*.

[66] Zhou, Z., Rahman Siddiquee, M., Tajbakhsh, N. & Liang, J. Unet++: A nested u-net architecture for medical image segmentation. *Deep Learning In Medical Image Analysis And Multimodal Learning For Clinical Decision Support: 4th International Workshop, DLMIA 2018, And 8th International Workshop, ML-CDS 2018, Held In Conjunction With MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*. pp. 3-11 (2018)

A

Appendix 1
Properties of the Diffusion Model

We include several key proofs of the diffusion probabilistic model. The detailed discussion is presented using the notation from [22].

Define the variance schedule of the forward diffusion $\{\beta_t\}_{t=1\ldots T}$, and $\alpha_t := 1 - \beta_t$, $\bar{\alpha_t} := \prod_{t=1}^{T} \alpha_t$.

A.1 Proofs of Equation (2.4) and (2.5)

The data perturbation kernel $q(x_t \mid x_{t-1})$ can be parameterized as (same as Eq. (2.2)):

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{\beta_t} z, \quad z \sim \mathcal{N}(0, I). \tag{A.1}$$

Further expanding the formula iteratively, we have:

$$
\begin{aligned}
x_t &= \sqrt{\alpha_t} x_{t-1} + \sqrt{\beta_t} z_{t-1} \\
&= \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \sqrt{\alpha_t (1-\alpha_{t-1})} z_{t-2} + \sqrt{1-\alpha_t} z_{t-1} \\
&= \sqrt{\alpha_t \alpha_{t-1} \alpha_{t-2}} x_{t-3} + \sqrt{\alpha_t \alpha_{t-1} (1-\alpha_{t-2})} z_{t-3} + \sqrt{\alpha_t (1-\alpha_{t-1})} z_{t-2} + \sqrt{1-\alpha_t} z_{t-1} \\
&= ... \\
&= \sqrt{\bar{\alpha}_t} x_0 + \sqrt{\alpha_t \alpha_{t-1} ... \alpha_2 (1-\alpha_1)} z_0 + ... + \sqrt{\alpha_t (1-\alpha_{t-1})} z_{t-2} + \sqrt{1-\alpha_t} z_{t-1}.
\end{aligned} \tag{A.2}
$$

where $z_0, z_1, ..., z_{t-1}$ are independent samples from the standard Gaussian distribution, and x_t is governed by the Gaussian distribution, with mean $\sqrt{\bar{\alpha}_t} x_0$, and the covariance matrix is $(\alpha_t \alpha_{t-1} ... \alpha_2 (1-\alpha_1) + ... + \alpha_t (1-\alpha_{t-1}) + 1 - \alpha_t) I = (1 - \alpha_1 \alpha_2 ... \alpha_t) I = (1 - \bar{\alpha}_t) I$. Hence, the perturbed data distribution is:

$$
q(x_t | x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t) I). \tag{A.3}
$$

This supports the direct sampling of x_t given x_0:

$$
x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} z, \quad z \sim \mathcal{N}(0, I). \tag{A.4}
$$

So far, the proofs have completed for Eq. (2.4) and Eq. (2.5).

Notably, $\bar{\alpha}_t$ approaches 0 as T is very large, indicating that the x_T converges into the standard Gaussian distribution with large T.

A.2 A Proof of Equation (2.6)

The posterior $q(x_{t-1} \mid x_t, x_0)$ can be given by the Bayes rule:

$$
\begin{aligned}
q(x_{t-1} \mid x_t, x_0) &= q(x_t \mid x_{t-1}, x_0)\frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)} \\
&= q(x_t \mid x_{t-1})\frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)} \quad \text{(by Markov assumption of forward diffusion)} \\
&= \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, \beta_t I)\frac{\mathcal{N}(x_{t-1}; \sqrt{\bar{\alpha}_{t-1}}x_0, (1-\bar{\alpha}_{t-1})I)}{\mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)} \quad \text{(by Eq. (2.3) and (2.5))} \\
&\propto \exp\left(-\frac{1}{2}\left(\frac{(x_t - \sqrt{\alpha_t}x_{t-1})^2}{\beta_t} + \frac{(x_{t-1} - \sqrt{\bar{\alpha}_{t-1}}x_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(x_t - \sqrt{\bar{\alpha}_t}x_0)^2}{1-\bar{\alpha}_t}\right)\right) \\
&= \exp\left(-\frac{1}{2}\left((\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})x_{t-1}^2 - (\frac{2\sqrt{\alpha_t}}{\beta_t} + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}x_{t-1}) + \mathrm{c}(x_t, x_0)\right)\right).
\end{aligned}
\tag{A.5}
$$

where the $c(x_t, x_0)$ does not depend on the x_{t-1}.

According to the general formula of the Gaussian distribution $\mathcal{N}(x; \frac{1}{2a}, -\frac{b}{2a}I) \propto \exp(ax^2 + bx + c)$. The posterior mean and variance can be concluded as:

$$
\begin{aligned}
\tilde{\beta}_t &= \frac{1}{\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}} \\
&= \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t. \\
\tilde{\mu}_t(x_t, x_0) &= (\frac{\sqrt{\alpha_t}}{\beta_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}x_0) \quad / \quad (\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}). \\
&= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t.
\end{aligned}
\tag{A.6}
$$

This finalizes the proof of the closed analytic form of $q(x_{t-1} \mid x_t, x_0) \sim \mathcal{N}(x_{t-1} \mid \tilde{\mu}_t, \tilde{\beta_t}I)$

A.3 A proof of Equation (2.9)

In this section, we derive the optimized variational bound on the negative log likelihood. For notation variance, we write $q(x_{1:T} \mid x_0)$ as q:

$$\begin{aligned}
\mathbb{E}_q[-\log p_\theta(x_0)] &\leq \mathbb{E}_q[-\log p_\theta(x_0)] + D_{\mathrm{KL}}(q(x_{1:T} \mid x_0) \parallel p_\theta(x_{1:T} \mid x_0)) \quad \text{(by non-negativity of KL)} \\
&= \mathbb{E}_q[-\log p_\theta(x_0)] + \mathbb{E}_q\left[\log \frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{1:T} \mid x_0)}\right] \quad \text{(by def. of KL)} \\
&= \mathbb{E}_q[-\log p_\theta(x_0)] + \mathbb{E}_q\left[\log \frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})} + \log p_\theta(x_0)\right] \quad \text{(by Bayes rule)} \\
&= \mathbb{E}_q\left[\log \frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})}\right] \quad \text{(by linearity of the expectation)}.
\end{aligned} \tag{A.7}$$

By multiplying the $q(x_0)$ on both sides, the distribution of which the expectation is with respect to can be altered from $q(x_{1:T} \mid x_0)$ to $q(x_{0:T})$. In the rest of the section, we rewrite the $q(x_{0:T})$ as q, maintaining the same notation as in [22].

$$\begin{aligned}
\mathbb{E}_q[-\log p_\theta(x_0)] &\leq \mathbb{E}_q\left[\log \frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})}\right] \\
&= \mathbb{E}_q\left[\log \frac{\prod_{t\geq 1} q(x_t \mid x_{t-1})}{p_\theta(x_T)\prod_{t\geq 1} p_\theta(x_{t-1} \mid x_t)}\right] \quad \text{(by Eq. (2.1) and (2.7))} \\
&= \mathbb{E}_q\left[-\log p(x_T) - \sum_{t\geq 1} \log \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})}\right].
\end{aligned} \tag{A.8}$$

So far, we have derived the variational bound of Eq. (2.9).

A.4 A proof of Equation (2.10)

Starting from Eq. (2.9), demonstrated in the Section A.3, we further derive the final training objective:

$$
\begin{aligned}
\mathbb{E}[-\log p_\theta(x_0)] &= \mathbb{E}_q\Bigg[-\log p(x_T) - \sum_{t\geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} \Bigg] := L_{vlb} \\
&= \mathbb{E}_q\Bigg[-\log p(x_T) - \sum_{t>1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} - \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \Bigg] \\
&= \mathbb{E}_q\Bigg[-\log p(x_T) - \sum_{t>1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} \cdot \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} - \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \Bigg] \\
&= \mathbb{E}_q\Bigg[-\log p(x_T) - \sum_{t>1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} \\
&\qquad - \log \frac{q(x_1|x_0)}{q(x_2|x_0)} \frac{q(x_2|x_0)}{\dots} \frac{\dots}{\dots} \frac{q(x_{T-1}|x_0)}{q(x_T|x_0)} - \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \Bigg] \\
&= \mathbb{E}_q\Bigg[-\log p(x_T) - \sum_{t>1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} - \log \frac{q(x_1|x_0)}{q(x_T|x_0)} - \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \Bigg] \\
&= \mathbb{E}_q\Bigg[\log \frac{q(x_T|x_0)}{p(x_T)} - \sum_{t>1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} - \log p_\theta(x_0|x_1) \Bigg] \\
&= \mathbb{E}_q\Bigg[D_{\mathrm{KL}}(q(x_T|x_0) \parallel p(x_T)) + \sum_{t>1} D_{\mathrm{KL}}(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t)) \\
&\qquad - \log p_\theta(x_0|x_1) \Bigg].
\end{aligned}
\tag{A.9}
$$

Here, we demonstrate the tractable DMs' training objective. We refer the reader to [22] for an alternate version of the proof, which is intractable but showcases the property of progressive decoding.

A.5 Variational bound

Fig. A.1 shows that the beginning step contributes massively to the true variational bound, while the remaining steps are nearly negligible. Hence, simplifying the training objectives claims to have equivalent sample quality, but might implicitly hurt the sample diversity.

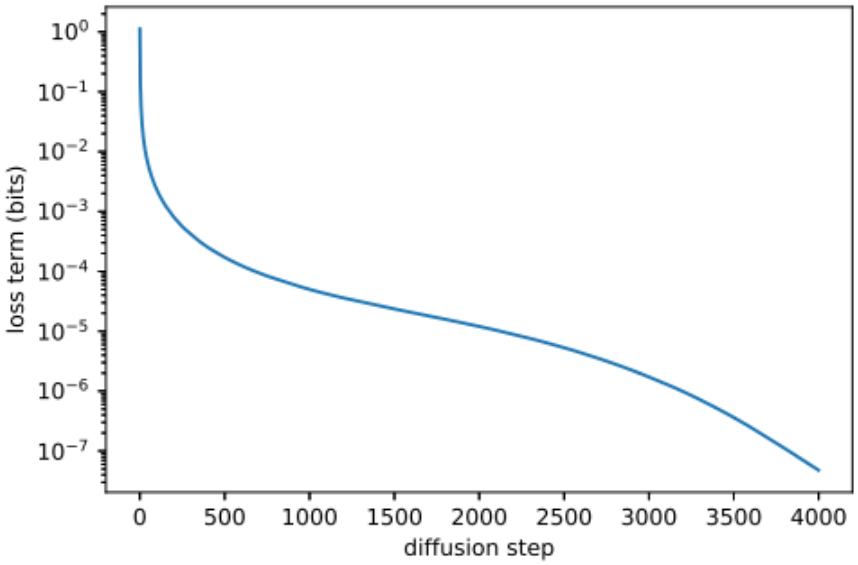


Figure A.1: Variational Bound vs. diffusion steps. Image from [41]

B

Appendix 2
More Generated Samples

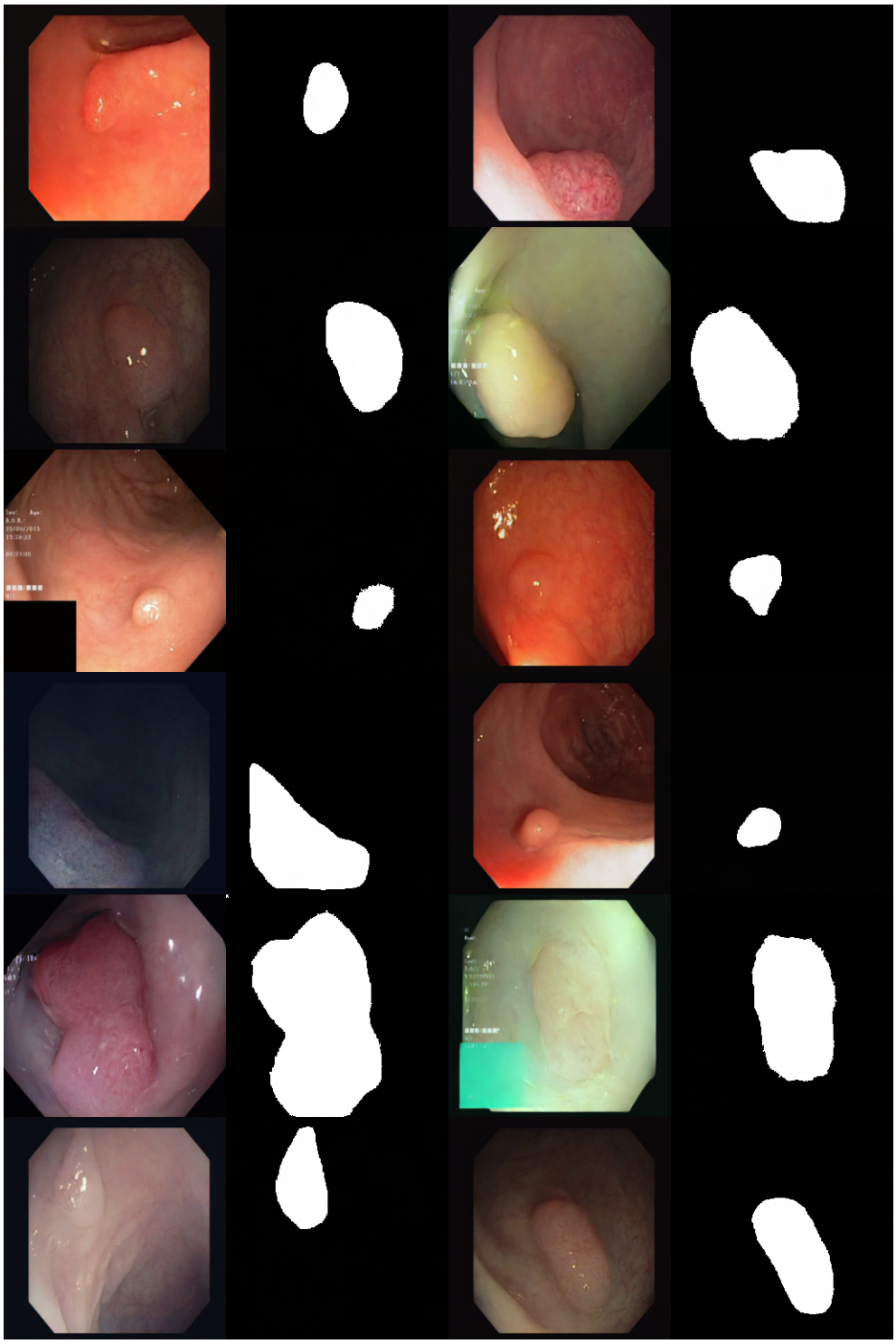


Figure B.1: Kvasir generated samples.

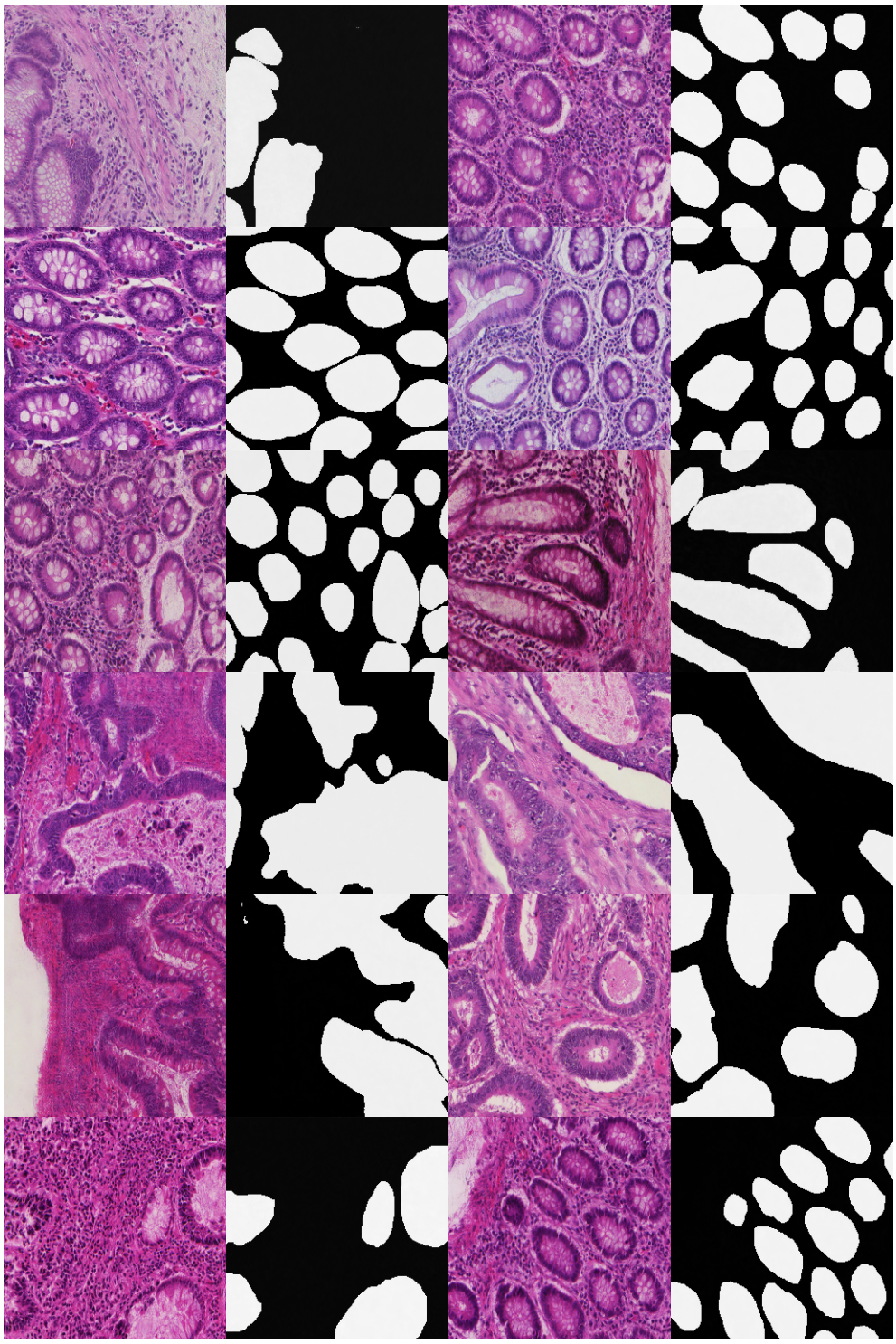


Figure B.2: Glas generated samples.

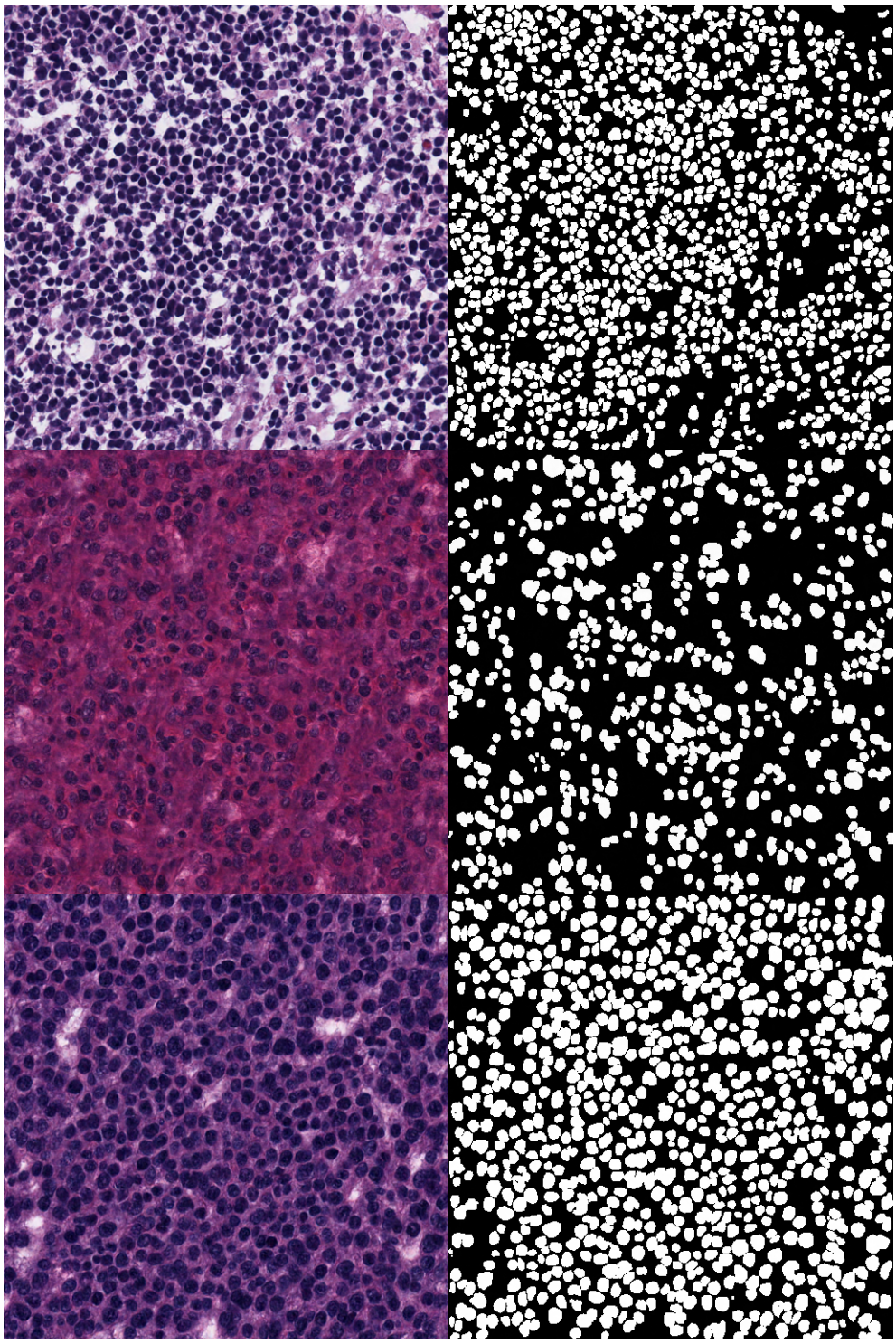


Figure B.3: MoNuSeg generated samples.

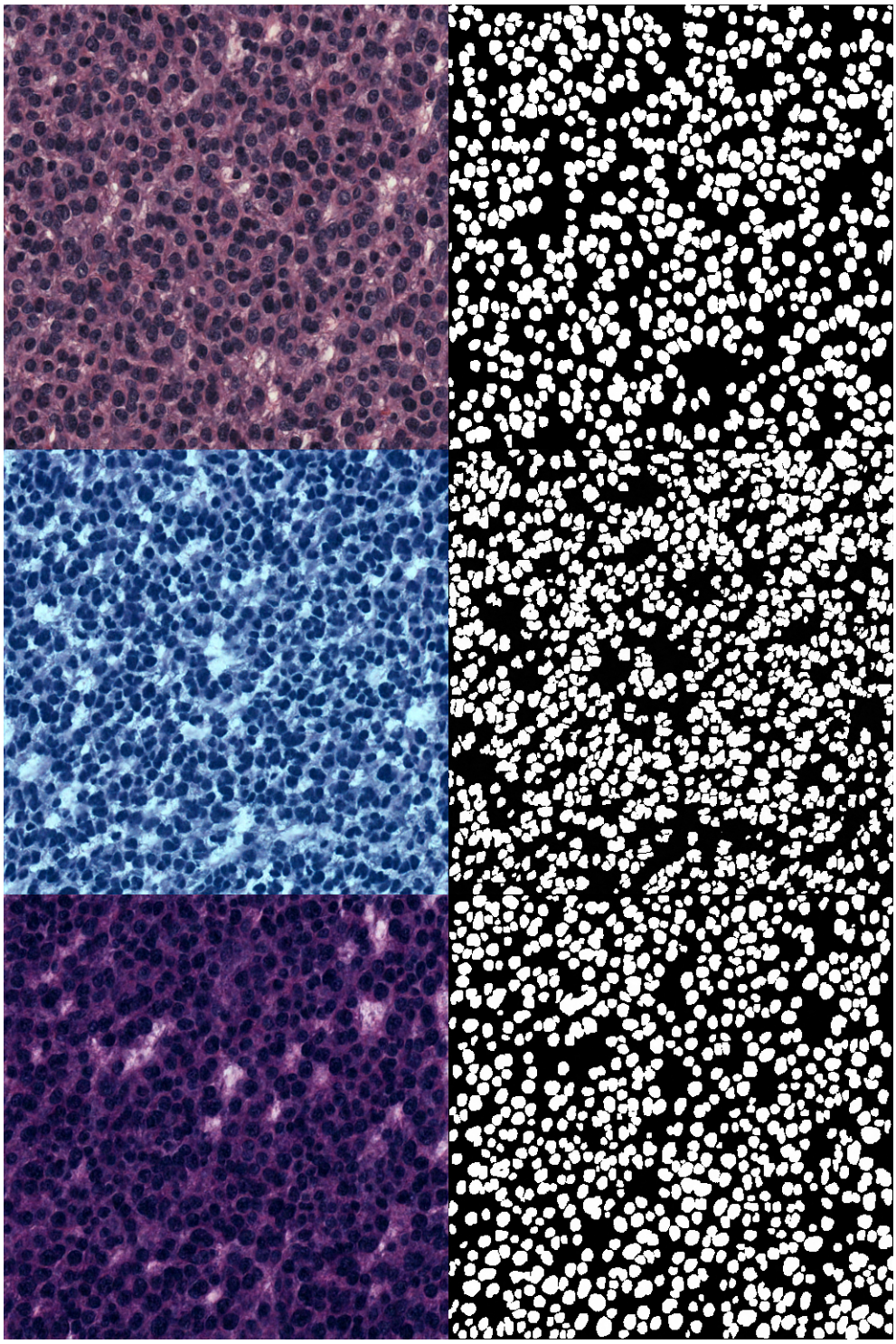


Figure B.4: MoNuSeg generated samples.

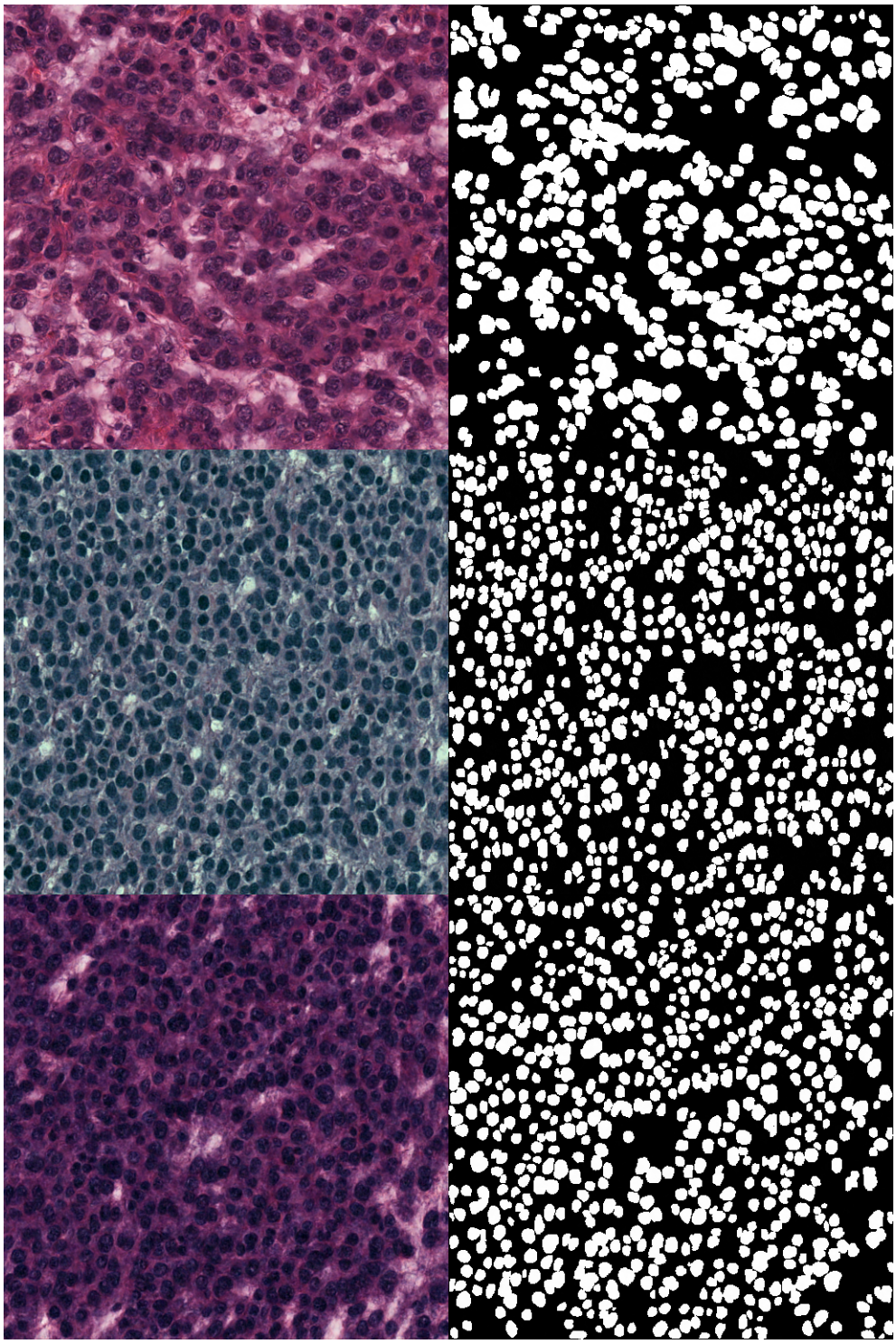


Figure B.5: MoNuSeg generated samples.

Figure B.6: Kvasir mask conditioned generated samples. From top to bottom: real image, real segmentation mask, four generated samples.

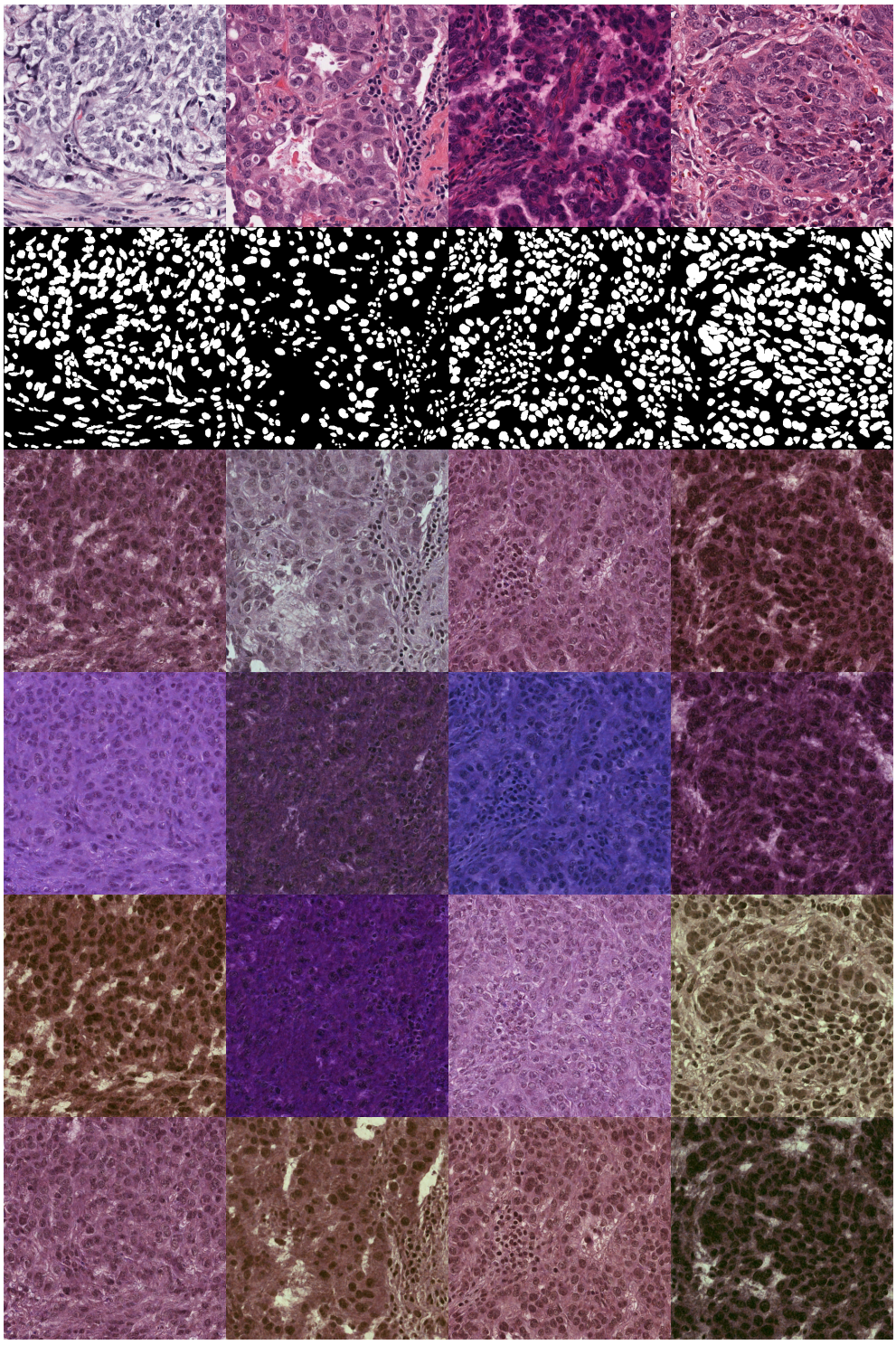


Figure B.7: MoNuSeg mask conditioned generated samples. From top to bottom: real image, real segmentation mask, four generated samples