# Diffusion Models and their applications

Milán Szabó

17. Dec. 2023

## 1 Introduction

Diffusion Models were first proposed by Sohl-Dickstein et al. [17] in 2015, inspired by non-equilibrium statistical physics. They are a type of generative model that has been used in several popular deep-learning models, such as DALL-E 2 [13], Stable Diffusion [14], Google Imagen [16] and GLIDE [11], not only because of their ability to produce diverse and high-quality samples, but also because of their flexibility and tractability. The primary purpose of diffusion models is to map training data to a latent space using a Markov chain. This process gradually adds noise to the data, resulting in an asymptotically transformed image, that is Gaussian distributed in nature. The ultimate goal of this method is to learn its reverse, which enables us to generate new data by producing a Gaussian image and traversing the reverse process. Diffusion models have a wide range of applications, including text simplification, question generation, text-to-image generation, paraphrasing, and more. The purpose of this project is to apply diffusion models to computer vision tasks, namely image segmentation.

In the previous semester, I studied the theoretical foundations of Diffusion Models and their applications in Image Synthesis and Image Segmentation. I looked at the repositories belonging to the papers presented in last semester's summary and studied the way they were implemented. I trained the models on the same dataset to compare them and to experiment with different sampling methods.

My main goal this semester was to create an environment in which I could train and test diffusion models. To achieve this I had to extend the already existing experiment pipeline used by the Computer Vision Group. I used and modified open-source implementations, as well as wrote code to achieve a flexible training environment. Since Diffusion Models are a kind of generative model, injecting it into the experimental pipeline, which was not made to test generative models, was a complex task. First I had to study the architecture of the pipeline and the existing implementations of diffusion models, and then I had to come up with a way to make them compatible. Finally, I made modifications that made training and modifying models easier. For the remaining part of the semester, I have spent running experiments for unconditional image synthesis and image segmentation, both on the COVID-QU[2] dataset. This came with a new set of challenges, like finding the optimal model, architecture, and hyperparameters.

## 2 Theoretical foundations

Diffusion Models are a type of latent variable model, which maps our dataset to a latent space and back. This mapping to the latent space is done by gradually adding Gaussian noise to our original data, finally obtaining a pure Gaussian image. Our goal is to learn to reverse this process, this way we can generate images first by sampling from the latent space and passing it through the estimate of the reverse process, obtaining a new image.

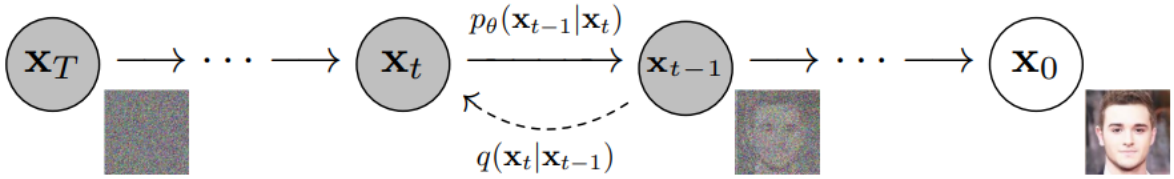


Figure 1: The diffusion process [7]

Specifically, the noising process is a Markov chain, which evolves according to:

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

Where $\mathcal{N}(x; m, \sigma)$ is the probability density function of the $m$ mean $\sigma$ variance normally distributed variable, $x_0$ is our original sample, $x_1, \ldots, x_T$ are our latents, and $\beta_1, \ldots, \beta_T$ is a variance schedule. Under good settings of of $T$ and $\beta_1, \ldots, \beta_T$, $q(x_T)$ is nearly Gaussian. Sampling at a given $t$ can be simplified by writing:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\overline{\alpha_t}}x_0, (1-\overline{\alpha_t})I), \tag{2}$$

$$\alpha_t = 1 - \beta_t, \overline{\alpha}_t = \prod_{s=1}^{t} \alpha_t. \tag{3}$$

Using Bayes' theorem we can prove that the, $q(x_{t-1}|x_t, x_0)$ posteriors are also Gaussian [6]:

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}, \tilde{\mu}(x_{t-1}, x_0), \tilde{\beta}_t \mathbb{I}), \tag{4}$$

with a mean that depends on the data. This means that the reverse transitions depend on the whole data distribution, and we have to estimate them for the sampling process. So to sample from $q(x_0)$, first we would have to sample from $q(x_T)$ and then sample from the estimated reverse steps until we reach $x_0$. By choosing $T$ and a variance schedule such that $q(x_T)$ is nearly Gaussian, sampling from this distribution is trivial.

Let our model that estimates the reverse transitions $p_\theta(x_{t-1}|x_t)$ be of the following form:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_t; \mu_\theta(x_{t-1}, t), \Sigma_\theta(x_{t-1}, t)), \tag{5}$$

where $\mu_\theta(x_{t-1}, t)$ and $\Sigma_\theta(x_{t-1}, t)$ are some kind of neural networks. The goal of training is to find such weights for these neural networks, which maximize the log-likelihood of our training data.

While Sohl-Dickstein et al. [17] proposed Diffusion Models in 2015, it wasn't until 2020 that Ho et al. [7] could produce high-quality samples, while achieving state-of-the-art sample quality results. The main contribution of their paper is the improved loss function. First, they assume, that $\Sigma_\theta(x_{t-1}, t) = \sigma_t \mathbf{I}$, where $\sigma_t = \beta_t$. Then they reparametrize $\mu_\theta(x_{t-1}, x_0)$ with $\varepsilon_\theta(x_t, t)$, a noise predictor. In this setting the loss, that is in the original paper of Sohl-Dickstein et al. [17] is the variational upper bound of the log-likelihood, becomes

$$\mathbb{E}_{x_0,\varepsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\overline{\alpha_t})}||\varepsilon - \varepsilon_\theta(\sqrt{\overline{\alpha_t}}x_0 + \sqrt{1-\overline{\alpha_t}}\varepsilon, t)||^2\right]. \tag{6}$$

Now their neural network only needs to approximate the noise at each diffusion step. The actual loss function that they use in the paper is

$$L_{\text{simple}}(\theta) = \mathbb{E}_{x_0,\varepsilon,t}\left[||\varepsilon - \varepsilon_\theta(\sqrt{\overline{\alpha_t}}x_0 + \sqrt{1-\overline{\alpha_t}}\varepsilon, t)||^2\right]. \tag{7}$$

This is a reweighed version of the previous loss, and they observed, that it works better in practice than simply using the variational upper bound.

### 2.1 Alternative formulations of Diffusion Models

In addition to the Denoising Diffusion Probabilistic Models presented in the previous section, there are multiple alternative frameworks of diffusion models, which might be worth mentioning.

Noise Conditioned Score Networks [19] are one of these frameworks, they aim to estimate the gradient of the data density. Formally, $\sigma_1 < \sigma_2 < \ldots < \sigma_T$ is a sequence of Gaussian noise scales, $p_{\sigma_1}(x) \sim p(x_0), p_{\sigma_1}(x) \sim \mathcal{N}(0, I)$, and $p_{\sigma_t}(x_t|x) \sim \mathcal{N}(x_t; x, \sigma_t I)$. Their aim is to estimate $\nabla_{x_t} p_{\sigma_t}(x_t)$, we know that $\nabla_{x_t} p_{\sigma_t}(x_t|x) = \dfrac{x_t - x}{\sigma_t}$, so they minimize the following loss function, where $s_\theta(x_t, \sigma_t)$ is a neural network.

$$\frac{1}{T}\sum_{t=1}^{T} \lambda(\sigma_T) E_{p(x)} E_{x_t \sim p_{\sigma_t}(x_t|x)} ||s(x_t, \sigma_t) - \frac{x_t - x}{\sigma_t}||$$

The estimated gradient is then used to iteratively denoise a sample from $\mathcal{N}(0, I)$, given the timestep. Namely, the sampling algorithm is the Langevin dynamics algorithm [19].

Stochastic Differential Equations [20] based data synthesis is another interesting approach. Just like the previous two methods, here the data is also transformed into noise. However, the diffusion process is considered to be continuous, it can be defined to be the solution of a SDE. The SDE describing the diffusion process is the following,

$$\frac{\partial x}{\partial t} = f(x, t) + \sigma(t)\omega_t \iff \partial x = f(x, t) \cdot \partial t + \sigma(t) \cdot \partial \omega,$$

where $\omega_t$ are standard normal variables, $\sigma$ is a time-dependent function that computes the diffusion coefficient, and f computes the drift coefficient. To have a diffusion process as a solution, the drift coefficient should be designed such that $x_t$ gradually becomes pure noise. Now the aim is just like before, to reverse this process. The reverse is defined as,

$$\partial x = [f(x, t) - \sigma(t)^2 \cdot \nabla_x \log p_t(x)] \cdot \partial t + \sigma(t) \cdot \partial \hat{\omega},$$

where $\hat{\omega}$ is the time reversed Brownian motion. The job of our neural network is to learn $\nabla_x \log p_t(x)$, just like before. For sampling, we can use any numerical SDE solver.

In the last couple of years, DDPMs have been researched extensively, while other formulations have not. Future research should focus on these two underexplored areas.

## 3 Unconditional Image Synthesis

To be able to apply Diffusion Models to image segmentation one first has to implement them for their primary function, which is unconditional image synthesis. In this section, I am going to present the dataset and the model architecture that was used to generate unconditional samples.

### 3.1 The Dataset

In all of the experiments, the COVID-QU [2] dataset was used. The dataset contains 33,920 chest X-ray of which 11,956 has COVID-19, 11,263 has non-COVID infections (Viral or Bacterial Pneumonia), and 10,701 are X-rays of normal lungs. Additionally, the whole dataset

includes segmentation masks for lungs. The infection dataset is a smaller subset of the data, it contains 1,456 Normal and 1,457 non-COVID-19 chest X-rays with corresponding lung masks, plus 2,913 COVID-19 chest X-rays with corresponding lung mask from the COVID-QU-Ex [2] dataset and corresponding infection masks from the QaTaCov19 [5] dataset.

Since training and sampling diffusion models is a memory-heavy process, all of the training and sampling in the experiments was done on images of size $64 \times 64 \times 1$.

## 3.2 Model Architecture

During training the model used 1000 diffusion steps. A linear noising schedule was employed with $\beta_0 = 0.0001$ and $\beta_{1000} = 0.02$, this controls how much noise we add at each diffusion step. These hyperparameters have remained fixed throughout our experiments.

In last semester's experiments, we concluded that the DDIM [18] sampling technique was superior to the classical approach. During sampling, we used this approach with 50 DDIM steps.

The model architecture described below is based on Rombach et al. [14] with the exception that we didn't use a latent encoder or decoder and that the input and output channels of the U-Net were fit to our dataset. The input of the noise-predicting U-Net model is an image of size $64 \times 64$ with one channel and the timestep embedding. The output is a denoised image of size $64 \times 64$ with one channel. First, the input image is transformed with a convolutional layer to an image of the same spatial size, but with 192 channels. After the first step, the input passes through the three segments of the U-Net [15], an encoder, a middle, and a decoder part. To define the architecture, one must first describe the two other major building blocks of our U-Net: Residual Blocks and Linear Attention.

A **Residual Block** has two inputs, an image, and the timestep embedding. Given a timestep $t$, first, a sinusoidal positional embedding is calculated, it is a vector of size 192, and then it is transformed with linear layers and SiLU activation to a vector of size $4 \cdot 192$. Then, the input image is normalized using Group Normalization, and it is passed through a SiLU activation. If the Residual Block is used for down or upsampling, average pooling or nearest interpolation is applied to the output of the previous step. The output is then passed through a convolutional layer. Then the timestep embedding is passed through SiLU and a linear layer and added together with the output of the previous step, the output is then passed through a Group Normalization layer, a SiLU activation and a convolutional layer. The output of the Residual Block is the sum of the previous step and the original input that was down or upsampled so the shapes fit.

A **Linear Attention Layer** first flattens the spatial dimensions of the input. After normalizing the input, $Q, V, K$ matrices are calculated with a one-dimensional convolutional layer. Then the rows of the matrices are cut up to equal pieces of size 64. After this the following formula is calculated: $softmax(\frac{QK^T}{\sqrt{D}})V$, where $Q, V, K$ are now the submatrices and $D$ is a scaling parameter, the outputs are then concatenated and transformed back to the shape of the input. Finally, the output is given by the sum of the input and the output of the previous step.

Now we can define the **Encoder**. The encoder has multiple levels, and each level has two Residual Blocks and a downsampling Residual Block. The first Residual Block grows the image channels, the second keeps the image dimensions the same the third RB downsamples the image spatial dimensions by a factor of two. The number of channels on each level is controlled by the *channel_mult* hyperparameter. In our case, the first level has 192, then $2 \times 192$, $3 \times 192$, and finally $4 \times 192$. Additionally, Attention layers are applied at levels of spatial resolution: $32 \times 32, 16 \times 16, 8 \times 8$, after each non-downsampling Residual Block.

The **Middle** part of the U-Net is made up of two Residual Blocks and a Linear Attention

Layer in the middle. These Residual Blocks keep the channel size constant.

The structure of the **Decoder** part matches the encoder, but instead of downsampling on each level, the Residual Block upsamples by a factor of two, and the input of each level is the concatenation of the output of the level of the Encoder with the same spatial resolution and the output of the previous Decoder level. The final output is then obtained by passing the output of the Decoder through a Group Normalization Layer, a SiLU activation, and a convolutional layer which transforms the image back to its original shape.

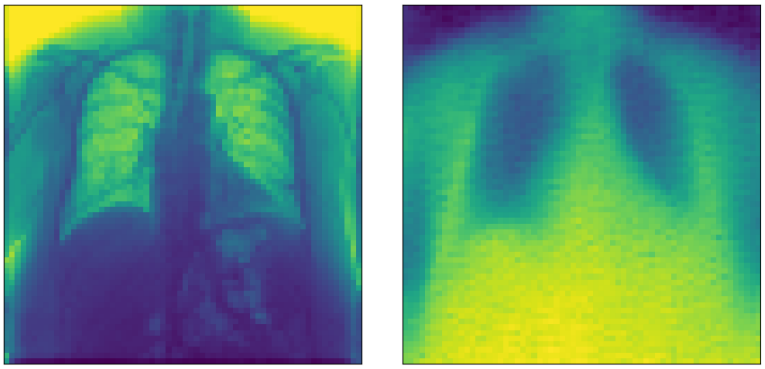


Figure 2: Example of an image and the image after adding noise and passing it through the U-Net. As you can imagine, one U-Net pass has removed some noise. During vanilla sampling, 1000 passes would be completed.

# 4 Conditional Image Synthesis

In conditional image synthesis, we aim to generate data from the conditional densities of our dataset. With diffusion models, this is usually achieved by adding the information about the conditioning to the noise estimating model, which will be in the form of $\varepsilon_\theta(x, y, t)$. One major architectural question is how we should add information about the conditioning. A usual approach is to simply concatenate, add it to the input of the U-Net or use cross-attention to add it to intermediate levels of the U-Net. In the latter part of this section, we are going to present the results of our experiments which have compared the effectiveness of the first two methods. This area of Diffusion Model research seems underexplored thus, future experimentation might be useful.

## 4.1 Diffusion Models in Image Segmentation

Image Segmentation is a major problem in computer vision. It involves assigning a label to each pixel, such that pixels with the same label belong to the same kind of object.

Amit et al. [1] tackle image segmentation as a conditional image synthesis task. Here they perform the diffusion steps on the segmentation masks and condition the noise estimating U-Net on the original image. With this method they have been able to achieve state-of-the-art results on the Cityscapes validation set, the Vaihingen building segmentation benchmark and the MoNuSeg dataset. Baranchuk et al. [3] show that the intermediate activations of the denoising U-Net capture semantic information well. They use a classifier on the upsampled activations to obtain a segmentation mask. Pinaya et al. [12] detect and segment anomalies on MRI data. First, they train a Diffusion Model on a healthy dataset. Using an anomalous image as an input will result in large loss values at anomalous regions. With an appropriate threshold, they can create a segmentation mask.

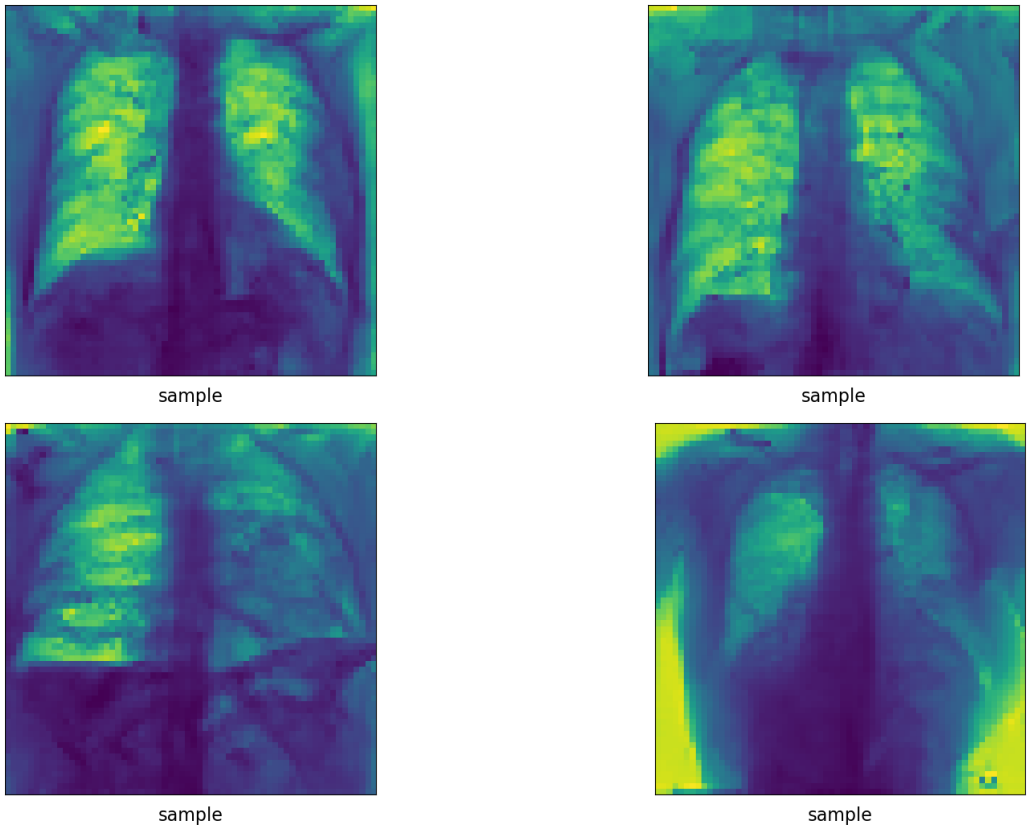


Figure 3: Synthetic Images of Chest X-Rays Generated from a Diffusion Model

Our approach resembles Amit et al. [1]. We aim to generate segmentation masks conditioned on the original image. One of the main questions of this approach is how we should supply the original image to the denoising U-Net. In the last part of this section, we are going to present results that compare different ways of conditioning to state-of-the-art results on the COVID-QU segmentation dataset.

## 4.2 Model Architecture

As mentioned previously, we tested two methods for adding the conditioning to the denoising U-Net. These two are adding conditioning by concatenation, and adding the conditioning to the image input of the U-Net. In both cases the conditioning, in this case the original image, was transformed by a Residual in Residual Dense Block, used by Amit et al. [1].

A **Residual in Residual Dense Block**(RRDB) includes three Residual Dense Blocks(RDB). A Residual Dense Block has 5 convolutional layers, after each layer follows a Leaky ReLU activation. Each of the 5 convolutional layers has the output of the previous layers as input, finally, the input of the RDB and output of the last layer is added. The output of the RRDB is obtained by passing the input through all the RDBs and combining it with the input. In our case, the RRDB transforms the input from $64 \times 64 \times 1$ to $64 \times 64 \times 16$. According to Amit et al. [1] increasing the number of RRDB blocks doesn't affect the mIoU much, however not using a condition embedder yields significantly worse results.

The rest of the U-Net architecture used in our segmentation experiments is the same as in Section 3.2 with the only difference that it has 16 or 17 input channels, depending on whether we add the conditioning with addition or concatenation.

## 4.3 Segmentation Quality Metrics

One of the main disadvantages of diffusion models is their inference speed. Thus evaluating the performance of diffusion models becomes a slow task. For this reason, we calculate the following metrics on 512 random data samples from the validation and train dataset in each epoch, the random seed used for sampling was the same across all measurements, so it makes sense to compare them. Still, the small sample size used for calculating the metrics does introduce some variance. Amit et al. [1] calculated the segmentation mask by inferring multiple times with the same conditioning and averaging the mask. Because of the nondeterministic nature of the DDIM sampling method [18], this approach yields more stable and accurate results, however because of the high inference costs we decided to calculate each mask one time.

Image segmentation can be thought of as the classification of each pixel into various categories. In this project, we only concerned ourselves with the case of binary classification, we used a confidence threshold of 0.5 on all of our experiments. Therefore, the following metrics are calculated from the confusion matrix: DICE-index, Jaccard-index, Accuracy, Sensitivity, Specificity, Balanced Accuracy, Precision, and Matthew's correlation coefficient. The most important metric is the DICE-index, which equals to

$$\frac{2TP}{2TP + FN + FP}.$$

## 4.4 Lung Segmentation on COVID-QU

The previous best result on the lung segmentation COVID-QU dataset achieved by the Computer Vision Group is a 0.99 DICE index with multiple variations of U-Net models.

Our diffusion models were trained for 100 epochs, around 4000 gradient steps, with a learning rate of 0.003 and a batch size of 512. These hyperparameters were carefully optimized since the training process would become unstable otherwise. Table 1 below summarizes our results. The results were not vastly different between the two models, although Amit et al. citeamit2021segdiff found that addition should produce better results in most cases. One reason for not noticing a significant difference between the two methods could be, that the complexity of the task is not high enough. Our models also produce comparable results with the U-Net models.

| | Addition | Concatenation |
|---|---|---|
| DICE index | 0.9529 | 0.9541 |
| Jaccard index | 0.9101 | 0.9123 |
| Accuracy | 0.9792 | 0.9793 |
| Sensitivity | 0.9455 | 0.9502 |
| Specificity | 0.9888 | 0.9878 |
| Balanced Accuracy | 0.9672 | 0.9793 |
| Precision | 0.9604 | 0.9581 |
| MCC | 0.9396 | 0.9408 |

Table 1: Measured metrics on the COVID-QU lung segmentation validation dataset from the last epoch.

## 4.5 Infection Segmentation on COVID-QU

The previous best results on the infection segmentation COVID-QU dataset achieved by the Computer Vision Group are 0.86 with a U-Net model and 0.85 by the Res50AttUNet model.

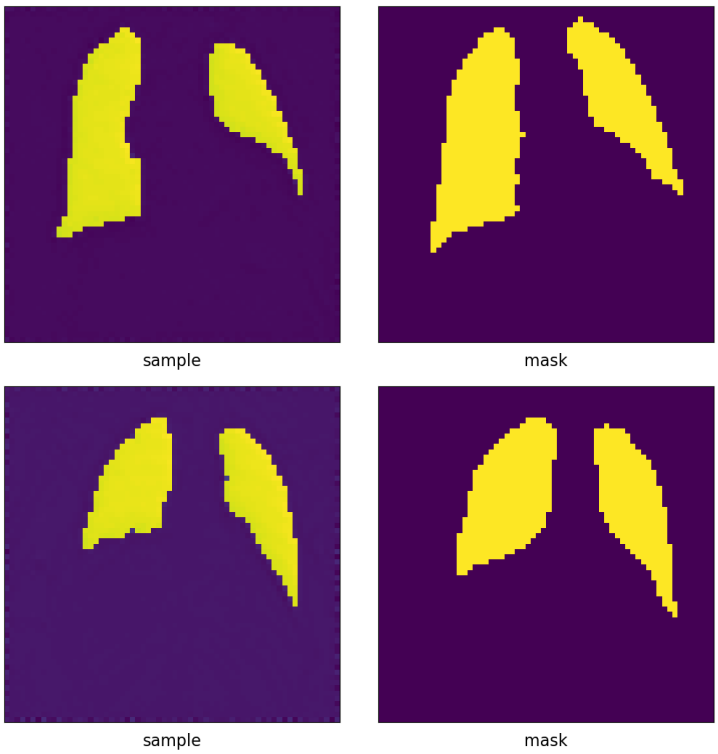


Figure 4: Examples of generated lung segmentation mask(left) and the original mask(right), the conditioning was supplemented via addition

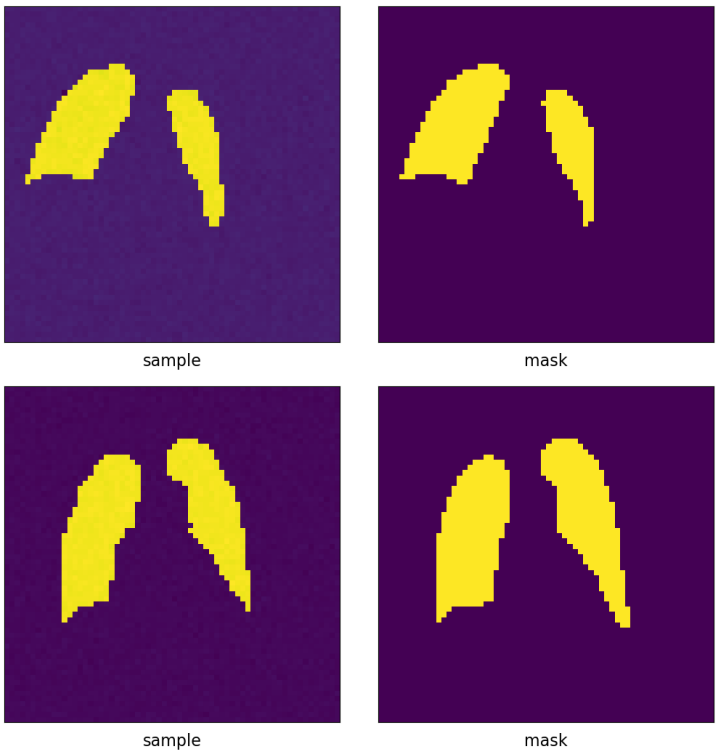


Figure 5: Examples of generated lung segmentation mask(left) and the original mask(right), the conditioning was supplemented via concatenation

Our diffusion models were trained for 2400 epochs and 8000 gradient steps, with a learning rate of 0.003 and a batch size of 512. Here the produced results are like expected, supplying the conditioning with addition has reached much better results than supplying it with concatena-

tion, see Table 2. The infection segmentation problem is a much more difficult task, therefore differences in model performance are more noticeable. Also, the variance of the metrics was high between batches. Although measuring the metrics is a time-consuming task, in the future I should let some experiments run for a longer time so we can get a clearer picture of the model's performance and reach a level where it can rival the U-Nets. The reason for the models' subpar performance compared to some U-Nets could be that the original purpose of diffusion models is image synthesis, while we can get some acceptable results in image segmentation further research is required to reach state-of-the-art results on multiple datasets.

| | Addition | Concatenation |
|---|---|---|
| DICE index | 0.6755 | 0.4708 |
| Jaccard index | 0.5105 | 0.3087 |
| Accuracy | 0.9183 | 0.8967 |
| Sensitivity | 0.6687 | 0.3646 |
| Specificity | 0.9550 | 0.9736 |
| Balanced Accuracy | 0.8119 | 0.6691 |
| Precision | 0.6838 | 0.6674 |
| MCC | 0.6293 | 0.4431 |

Table 2: Measured metrics on the COVID-QU infection segmentation validation dataset from the last epoch.

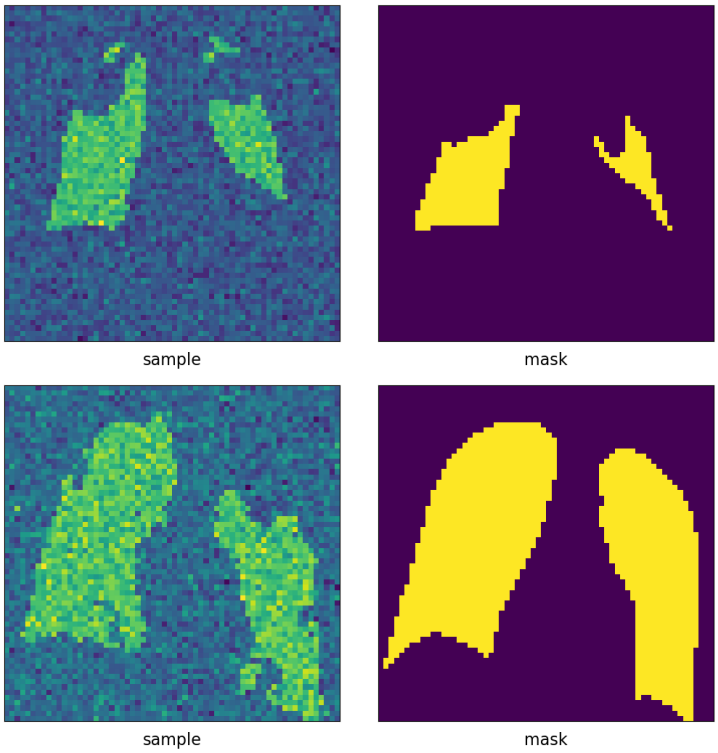


Figure 6: Examples of generated infection segmentation mask(left) and the original mask(right), the conditioning was supplemented via concatenation

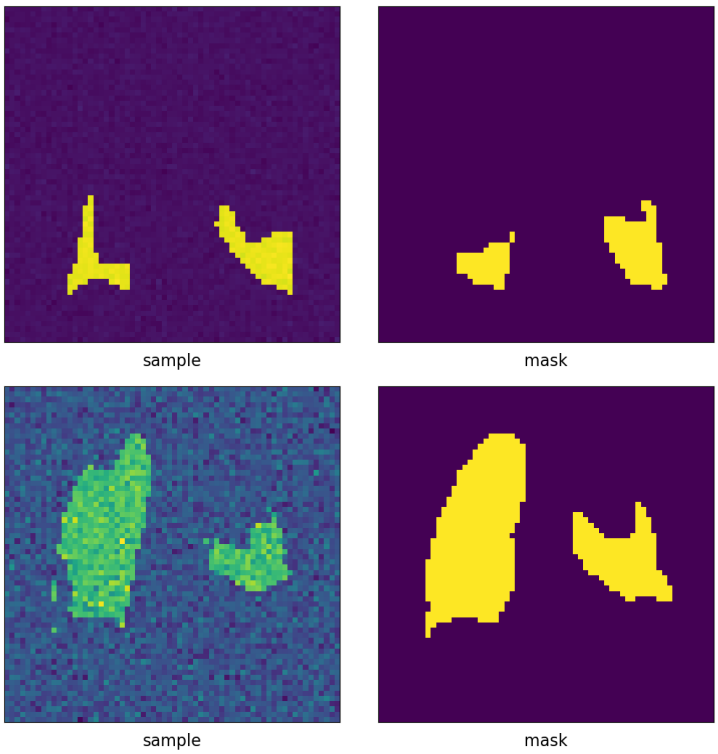


Figure 7: Examples of generated infection segmentation mask(left) and the original mask(right), the conditioning was supplemented via addition

# 5 Summary

The goal of this summary was to show my progress over the last semester. I furthered my knowledge about diffusion models, and similar models such as Noise Conditioned Score Networks, and Stochastic Differential Equations. I have kept up to date with the literature. I established an environment conducive to the training and testing of diffusion models. This includes understanding the technical details of the implementations. And finally, I ran experiments to compare the effectiveness of different conditioning methods.

Now that I have a working environment for diffusion models, my focus can be redirected from technical details to experimentation. Hopefully, this will yield good results for my thesis next semester.

# References

[1] Tomer Amit et al. "Segdiff: Image segmentation with diffusion probabilistic models". In: *arXiv preprint arXiv:2112.00390* (2021).

[2] Anas M. Tahir et al. *COVID-QU-Ex Dataset*. 2022. DOI: 10.34740/KAGGLE/DSV/3122958. URL: https://www.kaggle.com/dsv/3122958.

[3] Dmitry Baranchuk et al. "Label-efficient semantic segmentation with diffusion models". In: *arXiv preprint arXiv:2112.03126* (2021).

[4] Florinel-Alin Croitoru et al. "Diffusion Models in Vision: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023), pp. 1–20. DOI: 10.1109/tpami.2023.3261988.

[5] Aysen Degerli et al. "Reliable Covid-19 Detection using Chest X-Ray Images". In: *2021 IEEE International Conference on Image Processing (ICIP)*. 2021, pp. 185–189. DOI: 10.1109/ICIP42928.2021.9506442.

[6] Prafulla Dhariwal and Alexander Nichol. "Diffusion models beat gans on image synthesis". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 8780–8794.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6840–6851.

[8] Jonathan Ho and Tim Salimans. "Classifier-free diffusion guidance". In: *arXiv preprint arXiv:2207.12598* (2022).

[9] Mohammed Ali mnmoustafa. *Tiny ImageNet*. 2017. URL: https://kaggle.com/competitions/tiny-imagenet.

[10] Alex Nichol and Prafulla Dhariwal. *Improved Denoising Diffusion Probabilistic Models*. 2021. arXiv: 2102.09672 [cs.LG].

[11] Alex Nichol et al. *GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models*. 2022. arXiv: 2112.10741 [cs.CV].

[12] Walter HL Pinaya et al. "Fast unsupervised brain anomaly detection and segmentation with diffusion models". In: *Medical Image Computing and Computer Assisted Intervention–MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part VIII*. Springer. 2022, pp. 705–714.

[13] Aditya Ramesh et al. *Hierarchical Text-Conditional Image Generation with CLIP Latents*. 2022. arXiv: 2204.06125 [cs.CV].

[14] Robin Rombach et al. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10684–10695.

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV].

[16] Chitwan Saharia et al. *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*. 2022. arXiv: 2205.11487 [cs.CV].

[17] Jascha Sohl-Dickstein et al. "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International Conference on Machine Learning*. PMLR. 2015, pp. 2256–2265.

[18] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising diffusion implicit models". In: *arXiv preprint arXiv:2010.02502* (2020).

[19] Yang Song and Stefano Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution". In: *CoRR* abs/1907.05600 (2019). arXiv: 1907.05600. URL: http://arxiv.org/abs/1907.05600.

[20] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *CoRR* abs/2011.13456 (2020). arXiv: 2011.13456. URL: https://arxiv.org/abs/2011.13456.

[21] Julia Wolleb et al. *Diffusion Models for Implicit Image Segmentation Ensembles*. 2021. arXiv: 2112.03145 [cs.CV].

[22] Junde Wu et al. *MedSegDiff-V2: Diffusion based Medical Image Segmentation with Transformer*. 2023. arXiv: 2301.11798 [eess.IV].

[23] Junde Wu et al. *MedSegDiff: Medical Image Segmentation with Diffusion Probabilistic Model*. 2023. arXiv: 2211.00611 [cs.CV].