DDIM-based Synthetic Medical Image Augmentation for increased CNN Performance in COVID-19 Classification

Simone Rossetti
DIAG, Sapienza University of Rome
rossetti@diag.uniroma1.it
simone@deepplants.com

Abstract

Large-scale annotated datasets have proven to empower deep neural networks. However, obtaining large annotated datasets is challenging in many domains. The medical domain is one such area where labelling requires expertise, therefore annotating a large number of data is highly expensive and time-consuming but most of all inefficient in a pandemic scenario. We address the stated critical issue with the help of generative models. We demonstrate that DMs (Diffusion Models), can be used to generate synthetic augmented medical images even when trained with a small number of available real images, as well as GANs (Generative Adversarial Networks). From the experiments, we observe an increase in classification accuracy, sensitivity and specificity.

1 Introduction

In the context of the ongoing COVID-19 pandemic, following [19], we want to contribute to improving patient healthcare prognostic predictions. Data is the first step to developing any diagnostic/prognostic tool. While there exist large public datasets of more typical chest X-rays, there are not many collections of COVID-19 chest X-rays designed to be used for computational analysis.

Deal with the small datasets and limited amount of annotated samples is a recurrent problem in the medical imaging domain, especially when employing supervised machine learning algorithms that require labelled data and larger training examples. Although public medical datasets are available online, most datasets are still limited in size and only applicable to specific medical problems, the reason is that collecting medical data is a complex and expensive procedure and researchers use to attempt to overcome this challenge by using data augmentation. Data augmentation let the model acquire invariance (or equivariance) with respect to the task by perturbing the input signal in order to extract meaningful information. For this reason, augmentation techniques depend on the input domain and the desired task. We aim to develop a method capable of generating random perturbations of the input signal in order to increase the robustness of downstream tasks.

In computer vision, the standard data augmentation includes simple affine transformations of dataset images such as translation, rotation, flip, scale and colour jittering. Synthetic data augmentation of high-quality examples is a new, sophisticated data augmentation technique. Synthetic data learned using a generative model enable more variability and enrich the dataset to further improve the system training process. This approach permits sampling from a larger sample space which mixes the randomness of data augmentation and the amount of information. Generative models are a class of machine learning models that are used to generate synthetic data that resembles real-world data. These models learn the underlying distribution of a dataset and can then generate new samples from that distribution.

Preprint. Under review.

One approach inspired by game theory for training a model that syntheses images is known as Generative Adversarial Networks [7], as in [6, 19]. The model consists of two networks trained in an adversarial process. The first network generates synthetic data that resembles real-world data, and the second network discriminates between them repeatedly. However, GANs have several limitations, and challenges [1, 12, 15, 21]. GANs can sometimes generate a limited number of outputs, known as *mode collapse*. This occurs when the generator cannot capture the diversity of the data distribution and generates samples that are all similar [1]. GANs can be difficult to train because they involve two competing neural networks optimized against each other. This can lead to *instability* and oscillation during training, which can make it difficult to achieve good performance [15]. Moreover, GANs are *sensitive to the hyperparameters* (e.g., learning rate, batch size, etc.) used during training, and finding the optimal set is challenging [21, 1]. Sometimes, it can be *difficult to evaluate* the quality of generated samples from a GAN, especially when the data distribution is complex. This can make it hard to determine whether the GAN is producing good results [21, 12]. It also can be difficult to control the output of a GAN, as the generator is trained to generate samples that are difficult to distinguish from real data. This can make it hard to generate specific types of samples or enforce certain constraints on the generated data.

A much more promising approach comes from Diffusion Models [22], which are a class of Markov chain deep learning models based on variational Bayesian inference. DMs are based on denoising diffusion processes, which can be used to learn and evaluate complex distributions over high-dimensional data. Beyond the general advantages of DMs over GANs in the training procedure (DMs are generally easier to train, less sensitive to hyperparameters, more stable and do not suffer from mode collapse), DMs are a type of probabilistic model that approximates a continuous true posterior distribution using a variational distribution, which permits likelihood estimation and sampling. Moreover, DMs allow for interpolation in the embedding space and straightforward computation of conditionals (e.g. to compute a posterior) for denoising and inpainting natural images [22, 9]. One problem of DMs is that the longer the diffusion process, the better the quality of the reconstructed image; DMs typically require very long training (typically thousand of steps), making it challenging to train in high dimensional input spaces. Recent works [9] proposed Denoising Diffusion Probabilistic Models (DDPMs), an equivalent formulation to DMs from the connection with denoising Score Matching with Langevin Dynamics (SMLD) [25] to enhance the quality of generated images. [23] demonstrated the link of DMs to Score Based Models (SBMs) [27], introducing non-markovian DMs called Denoise Diffusion Implicit Models (DDIMs), which allow from $10\times$ to $50\times$ speed increase in the training concerning standard DMs while preserving the same sampling quality and being deterministic.

In this work, we use CNNs to classify pneumonia infection origin. We compare the performances of our classifier when trained with standard augmentation techniques and with synthetically augmented images using GANs, DDPMs and DDIMs. We conducted our experiments on an earlier version of the COVID-19 chest X-ray dataset [5] made of a small number of samples and four categories, see Figure 1.

2 Related work

DDIMs [23] are based on a large family of existing methods of learning generative models as transition operators of Markov chains [22, 2, 20, 24]. Among them, DDPMs [9] and SMLD [26, 25] have recently achieved high sample quality comparable to GANs [4, 11]. DDPMs optimize a variational lower bound to the log-likelihood, whereas SMLD optimize the score-matching objective [10]. Despite their different motivations, DDPMs and SMLD are closely related. Both use a denoising autoencoder objective for many noise levels, and both use a procedure similar to Langevin dynamics to produce samples. Since Langevin dynamics is a discretization of a gradient flow, DDPMs and SMLD require many steps to achieve good sample quality. DDIMs, on the other hand, is an implicit generative model [14] where samples are uniquely determined from the latent variables. Hence, DDIMs have specific properties that resemble GANs [7], such as the ability to produce semantically meaningful interpolations. DDIMs derive from a purely variational perspective, where the restrictions of Langevin dynamics are irrelevant; this could partially explain why is possible to observe superior sample quality compared to DDPM under fewer iterations.

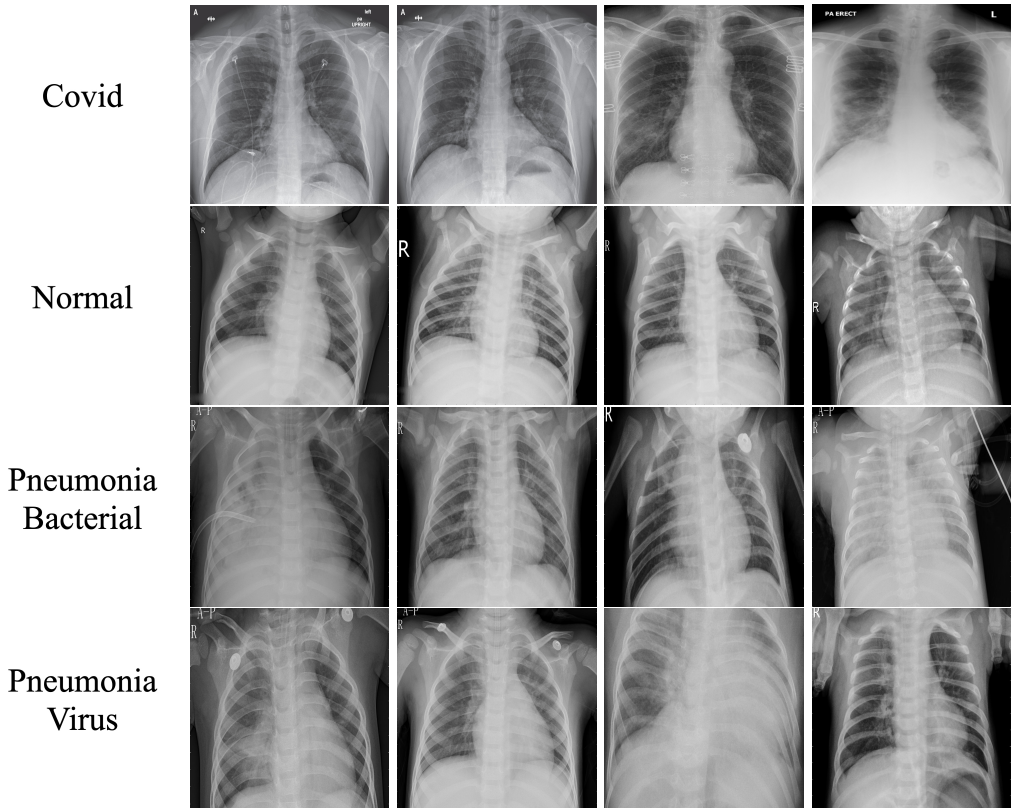


Figure 1: Samples taken from 'COVID-19 chest X-ray dataset' [5] *test* set.

3 Background

We now follow the derivation of DDIMs starting with the more generic definition of DMs.

3.1 DMs [22]

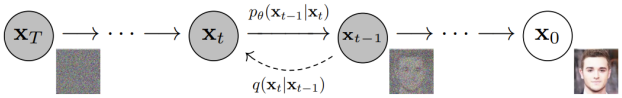


Figure 2: From [9]; the directed graphical model considered in [22] and [9].

DMs are a family of flexible and tractable Markov chain generative models derived from Langevin dynamics and Annealed Importance Sampling (AIS) [16]. DMs are latent variable models of the form $p_\theta(\mathbf{x}_0) := \int p(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T}$ where latents $\mathbf{x}_1, \ldots, \mathbf{x}_T$ have same dimensionality of the data $\mathbf{x}_0 \sim q(\mathbf{x}_0)$. The joint distribution $p(\mathbf{x}_{0:T})$ is called the *reverse process*. It is defined as a Markov chain with learned Gaussian transitions starting at $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$ (i.e. starts by pure noise and tries to iteratively denoise the signal):

$$p(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{i=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \tag{1}$$

While the *forward diffusion process* is a high-dimensional fixed Markov chain (rather than low-dimensional trainable as in variational autoencoders [17]) which degrades the signal at each time step according to a constant variance schedule $\beta_1, \ldots, \beta_T$, it has the form $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{i=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \qquad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_t, \beta_t \mathbf{I}) \tag{2}$$

Training is performed by optimizing the usual variational bound on negative log-likelihood L, which originates from annealed importance sampling and the Jarzynski equality – since $p_\theta(\mathbf{x}_0)$ is intractable we instead evaluate the relative probability of the forward and reverse trajectories averaged over forward trajectories:

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t=1}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L \quad (3)$$

3.2 DDPMs [9]

DDPMs design a weighted variational bound according to a novel connection between DMs and SMLD, with a progressive lossy decompression scheme that can be interpreted as a generalization of autoregressive decoding, see Fig. 3. Given the property of the forward process that admits sampling $\mathbf{x}_t$ at any arbitrary step t:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_0, (1-\alpha_t)\mathbf{I}), \text{ where } \alpha_t := \prod_{s=1}^{t} 1-\beta_s \quad (4)$$

they design efficient training optimizing random terms of L with stochastic gradient descent; L can be simplified to:

$$\mathbb{E}_q\left[\underbrace{D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} - \underbrace{\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right] \quad (5)$$

where L_T can be ignored since it is constant for fixed β_t. D_{KL} is used to compare $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ to the forward process posterior $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ which is tractable given $\mathbf{x}_0$:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2\mathbf{I}) \quad (6)$$

$$\text{whereb } \boldsymbol{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\alpha_{t-1}}\beta_t}{1-\alpha_t}\mathbf{x}_0 + \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1})}{1-\alpha_t}\mathbf{x}_t \text{ and } \sigma_t^2 := \frac{1-\alpha_{t-1}}{1-\alpha_t}\beta_t \quad (7)$$

thus all KL divergences are comparisons between Gaussians, so they can be calculated in a Rao-Blackwellized fashion with closed-form expressions instead of high variance Monte Carlo estimates. The loss term reduces to:

$$L = L_{t-1} - C = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2} \parallel \boldsymbol{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t) \parallel^2\right] \quad (8)$$

where C is constant and does not depend on θ. However, L can be further simplified by reparametrization of Eq. 4 considering $\mathbf{x}_t(\mathbf{x}_0, \boldsymbol{\epsilon}) = \sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and substituting the mean terms in Eq. 8 we end up parametrizing:

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \boldsymbol{\mu}_t\left(\mathbf{x}_t, \frac{1}{\sqrt{\alpha_t}}(\mathbf{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t))\right) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) \quad (9)$$

where $\boldsymbol{\epsilon}_\theta$ is a function approximator intended to predict $\boldsymbol{\epsilon}$ from $\mathbf{x}_t$. The loss in Eq. 8 simplifies to:

$$\mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2(1-\beta_t)(1-\alpha_t)} \parallel \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}, t) \parallel^2\right] \quad (10)$$

which resembles denoising score matching over multiple noise scales indexed by t [25]. Sampling $\mathbf{x}_t \sim p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ can be obtained computing $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t\mathbf{z}$ where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

3.3 DDIMs [23]

[23] observe that the loss in Eq. 10 only depends on the "marginal" $q(\mathbf{x}_t|\mathbf{x}_0)$ and not on the "joint" $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$. They decide to explore a new non-Markovian generative process which leads to the same

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$
6: **until** converged

Algorithm 2 Sampling

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

Figure 3: From [9]; the model is trained efficiently optimizing terms at random time steps with stochastic gradient descent (Algorithm 1). Sampling can be obtained by predicting the amount of noise at time step t (Algorithm 2).

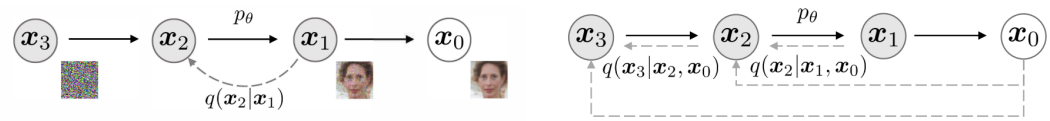


Figure 4: From [23]; Graphical models for diffusion (left) and non-Markovian (right) inference models.

surrogate objective function as DDPM and a faster generative process (fewer iterations). The forward process is a chain of a family of inference distributions indexed by real vector $\sigma \in \mathbb{R}^T_{\geq 0}$:

$$q_\sigma(\mathbf{x}_{1:T}|\mathbf{x}_0) = q_\sigma(\mathbf{x}_T|\mathbf{x}_0) \prod_{t=2}^{T} q_\sigma(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \tag{11}$$

where $q_\sigma(\mathbf{x}_T|\mathbf{x}_0) = \mathcal{N}(\sqrt{\alpha_T}\mathbf{x}_0, (1-\alpha_t)\mathbf{I})$ and the reverse conditional distribution:

$$q_\sigma(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}\mathbf{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\frac{\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_0}{\sqrt{1-\alpha_t}}, \sigma_t^2\mathbf{I}\right) \tag{12}$$

The mean function is chosen to ensure that $q_\sigma(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\sqrt{\alpha_t}\mathbf{x}_0, (1-\alpha_t)\mathbf{I})$. The forward process can be derived from the Bayes rule:

$$q_\sigma(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) = \frac{q_\sigma(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) q_\sigma(\mathbf{x}_t|\mathbf{x}_0)}{q_\sigma(\mathbf{x}_{t-1}|\mathbf{x}_0)} \tag{13}$$

The magnitude of σ control how much the process is stochastic. For $\sigma \to \mathbf{0}$ the process is deterministic and fixed. At this point the generative process $p_\theta(\mathbf{x}_{0:T})$ is straightforward: given a noisy observation $\mathbf{x}_t$, we first make a prediction of the corresponding $\mathbf{x}_0$, and the use if to obtain a sample $\mathbf{x}_{t-1}$ using the defined reverse conditional $q_\sigma(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$. During training for some $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ we obtain $\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}$. The model $\epsilon_\theta^{(t)}(\mathbf{x}_t)$ attempts to predict $\boldsymbol{\epsilon}_t$ from $\mathbf{x}_t$ without knowledge of $\mathbf{x}_0$. The *denoised observation* is:

$$f_\theta^{(t)}(\mathbf{x}_t) := (\mathbf{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\mathbf{x}_t))/\sqrt{\alpha_t} \tag{14}$$

The generative process with fixed prior $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ is then:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \begin{cases} \mathcal{N}(f_\theta^{(1)}(\mathbf{x}_1), \sigma_1^2\mathbf{I}) & \text{if } t = 1 \\ q_\sigma(\mathbf{x}_{t-1}|\mathbf{x}_t, f_\theta^{(t)}(\mathbf{x}_t)) & \text{otherwise} \end{cases} \tag{15}$$

Such parametrization can accelerate sampling and let us choose the time steps to perform without retraining a different model. From Eq. 15 we can generate $\mathbf{x}_{t-1}$ from $\mathbf{x}_t$ with:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{\mathbf{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\mathbf{x}_t)}{\sqrt{\alpha_t}}\right)}_{\text{predicted } \mathbf{x}_0} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2}\epsilon_\theta^{(t)}(\mathbf{x}_t)}_{\text{direction pointing to } \mathbf{x}_t} + \underbrace{\sigma_t\epsilon_t}_{\text{random noise}} \tag{16}$$

where $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ independent from x_t and $\alpha_0 := 1$.

When $\sigma_t = \sqrt{(1-\alpha_{t-1})/(1-\alpha_t)}\sqrt{1-\alpha_t/\alpha_{t-1}}$ for all t the process become Markovian. For $\sigma_t = 0$ for all t the forward process becomes deterministic except for $t = 1$: in the generative process, the terms before the random noise become zero. The resulting model becomes an implicit model (those models which directly represent the sampling process e.g. GANs)[14], where samples are generated from latent variables with a fixed procedure. They further simplify the expression of the loss in Eq. 10:

$$\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}, t) \|^2\right] \tag{17}$$

4 Method

The idea behind our approach is that with appropriate data augmentation a classifier becomes more robust to invariant properties of the classification task, see flowchart in Fig. 5. To our purpose, latent variable models have the advantage of allowing for instance interpolation in latent space. Moreover, diffusion and score-based generative models allow for image inpainting too. Following [19] we train our DDIM model, from [23], on randomly augmented samples of a dataset. Next, we generate new sets of synthetically augmented images (using standard reverse process, interpolation and inpaiting techniques) with different cardinalities and test a simple CNN performances over the improved datasets.

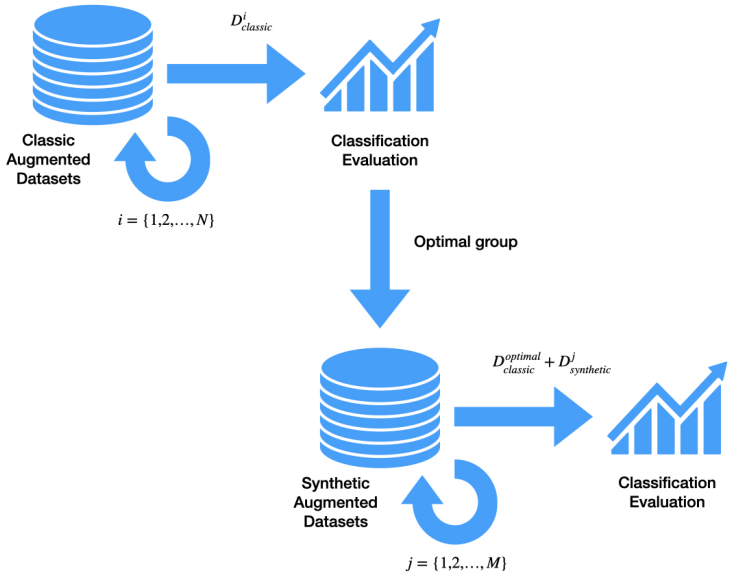


Figure 5: Experiment flowchart for evaluating synthetic data augmentation in the task of classifying COVID-19 and pneumonia infection.

4.1 Interpolation in latent space

To generate interpolations for DDIMs, following [23], we randomly sample two initial $\mathbf{x}_T$ values from the standard Gaussian, interpolate them with spherical linear interpolation, and then use the DDIM to obtain $\mathbf{x}_0$ samples, see Fig. 6. We generate interpolations in the latent space directly using DDIMs while in DDPMs, due to the stochasticity of the trajectory, this is only possible near the

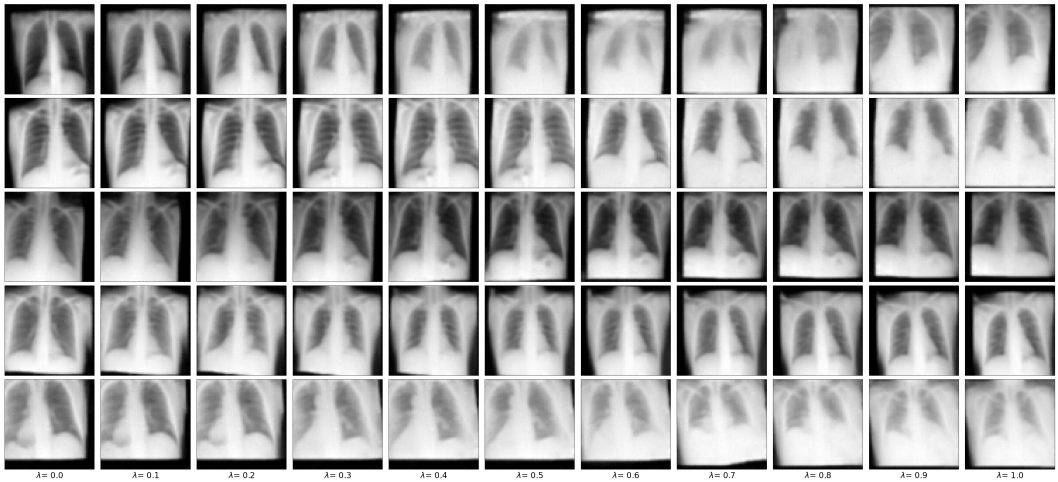


Figure 6: Interpolations in DDIM latent space of random samples about the *covid* disease. From left to right column interpolation goes from $\lambda = 0$ to $\lambda = 1$ with 0.1 interleave.

image level. We choose interpolation coefficient $\lambda \in [0, 1]$ and apply the following interpolation:

$$\mathbf{x}_T^\lambda = \frac{\sin((1-\lambda)\theta)}{\sin(\theta)}\mathbf{x}_T^0 + \frac{\sin(\lambda\theta)}{\sin(\theta)}\mathbf{x}_T^1, \quad \text{with} \quad \theta = \arccos\left(\frac{(\mathbf{x}_T^0)^\top \mathbf{x}_T^1}{\| \mathbf{x}_T^0 \| \| \mathbf{x}_T^1 \|}\right) \tag{18}$$

4.2 Inpainting

We use DDPMs and DDIMs to generate inpainting augmented images; see Fig. 10. Contrary to [13], we do not adopt any particular training strategy or fine-tuning. At inference time we mask the region of the image $\mathbf{x}_0$ which is unknown with the binary mask $\mathbf{m}$; $\mathbf{x}_0^{keep} = \mathbf{m} * \mathbf{x}_0$ is the known region while $\mathbf{x}_0^{replace} = (1 - \mathbf{m}) * \mathbf{x}_0$ is the unknown one. We sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and perform the following trajectory:

$$\begin{aligned}
\mathbf{x}_{t-1}^{keep} &= \sqrt{\alpha_{t-1}}\mathbf{x}_0 + \sqrt{1 - \alpha_{t-1} - \sigma_t^2}\epsilon_\theta^{(t)}(\mathbf{x}_t) + \sigma_t \epsilon_t \\
\mathbf{x}_{t-1}^{replace} &= \sqrt{\alpha_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\mathbf{x}_t)}{\sqrt{\alpha_t}}\right) + \sqrt{1 - \alpha_{t-1} - \sigma_t^2}\epsilon_\theta^{(t)}(\mathbf{x}_t) + \sigma_t \epsilon_t \\
\mathbf{x}_{t-1} &= \mathbf{m} * \mathbf{x}_{t-1}^{keep} + (\mathbf{1} - \mathbf{m}) * \mathbf{x}_{t-1}^{replace}
\end{aligned} \tag{19}$$

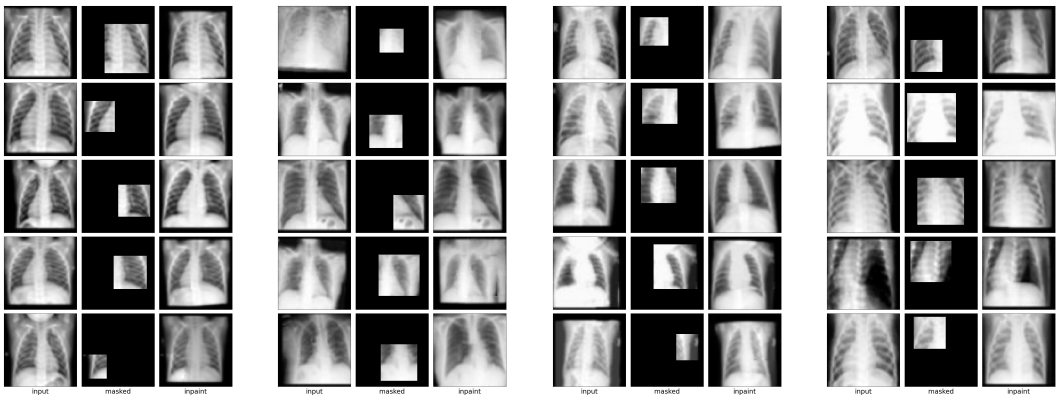
Again for $\sigma_t = 0$, the process becomes deterministic, and the trajectory is fixed. We show qualitative results in the next section.

5 Experiments

We generate images of chest X-ray affected by pneumonia disease. Following [19] (see flowchart in Fig. 5) we train our DDIM model on 2000 randomly augmented samples (which is the optimal set for [19]) of the COVID-19 chest X-ray dataset [5]. Namely, we train four models, one for each category: *covid*, *normal*, *pneumonia virus* and *pneumonia bacterial*; see Figure 1. The dataset is well structured and has a total of 270 training images (60,70,70,70) and 36 test images (9,9,9,9). We used the default train/test split proposed by the creators of the dataset.

5.1 Evaluation

To evaluate the quality of the generated samples, we used Kernel Inception Distance (KID) [3], an image quality metric proposed as a replacement for the popular Frechet Inception Distance (FID) [8].



(a) *normal* (b) *covid* (c) *pneumonia bac* (d) *pneumonia vir*

Figure 7: Inpainting using DDIM latent space. Left is the source image, centre is the masked input and right is the reconstructed image.

Table 1: Number of images for each class in the COVID-19 dataset.

Set/Class	**Covid**	**Normal**	**Pneumonia_bac**	**Pneumonia_vir**	**Total**
Train	60	70	70	70	270
Test	9	9	9	9	36
Total	69	79	79	79	306

4-fold_750_ep_0.827_acc	Accuracy	Sensitivity	Specificity	F1Score
covid	1	1	1	1
normal	0.936	0.748	0.999	0.852
pneumonia_bac	0.827	0.925	0.795	0.854
pneumonia_vir	0.891	0.636	0.976	0.768

(a) Classic augmented trainset, 750 epochs, 0.827 total accuracy.

4-fold_720_ep_0.907_acc	Accuracy	Sensitivity	Specificity	F1Score
covid	1	1	1	1
normal	0.972	0.896	0.997	0.943
pneumonia_bac	0.908	0.907	0.909	0.907
pneumonia_vir	0.933	0.824	0.97	0.888

(b) GAN [19] synthetic augmented trainset, 720 epochs, 0.907 total accuracy.

4-fold_350_ep_0.923_acc	Accuracy	Sensitivity	Specificity	F1Score
covid	1	1	1	1
normal	0.971	0.884	1	0.939
pneumonia_bac	0.923	0.993	0.9	0.944
pneumonia_vir	0.951	0.813	0.997	0.894

(c) Ours synthetic augmented trainset, 350 epochs, 0.923 total accuracy.

Figure 8: CNN predictions improvement given by synthetic data augmentation.

Both metrics measure the difference in the generated and training distributions in the representation space of an InceptionV3 [28] network pre-trained on ImageNet. According to the paper [3], KID was

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Sequential: 1-1                        [-1, 32, 32, 32]          --
│    └─Conv2d: 2-1                       [-1, 32, 64, 64]          320
│    └─ReLU: 2-2                         [-1, 32, 64, 64]          --
│    └─BatchNorm2d: 2-3                  [-1, 32, 64, 64]          64
│    └─MaxPool2d: 2-4                    [-1, 32, 32, 32]          --
├─Sequential: 1-2                        [-1, 64, 16, 16]          --
│    └─Conv2d: 2-5                       [-1, 64, 32, 32]          18,496
│    └─ReLU: 2-6                         [-1, 64, 32, 32]          --
│    └─BatchNorm2d: 2-7                  [-1, 64, 32, 32]          128
│    └─MaxPool2d: 2-8                    [-1, 64, 16, 16]          --
├─Sequential: 1-3                        [-1, 128, 8, 8]           --
│    └─Conv2d: 2-9                       [-1, 128, 16, 16]         73,856
│    └─ReLU: 2-10                        [-1, 128, 16, 16]         --
│    └─BatchNorm2d: 2-11                 [-1, 128, 16, 16]         256
│    └─MaxPool2d: 2-12                   [-1, 128, 8, 8]           --
├─Sequential: 1-4                        [-1, 4]                   --
│    └─Flatten: 2-13                     [-1, 8192]                --
│    └─Linear: 2-14                      [-1, 256]                 2,097,408
│    └─ReLU: 2-15                        [-1, 256]                 --
│    └─Dropout: 2-16                     [-1, 256]                 --
│    └─Linear: 2-17                      [-1, 4]                   1,028
│    └─Softmax: 2-18                     [-1, 4]                   --
==========================================================================================
Total params: 2,191,556
Trainable params: 2,191,556
Non-trainable params: 0
Total mult-adds (M): 43.22
==========================================================================================
Input size (MB): 0.02
Forward/backward pass size (MB): 3.50
Params size (MB): 8.36
Estimated Total Size (MB): 11.88
==========================================================================================
```

Figure 9: CNN COVID-19 X-ray classifier architecture.

proposed because FID has no unbiased estimator; its expected value is higher when it is measured on fewer images. KID is more suitable for small datasets because its expected value does not depend on the number of samples it is measured on. It is also computationally lighter, numerically more stable, and simpler to implement because it can be estimated per batch.

We evaluate images at the minimal possible resolution of the Inception network (75×75 instead of 299×299), and the metric is only measured on the validation set for computational efficiency. Since the dataset is relatively small, we go over the train and validation splits multiple times per epoch (i.e. 20 times), because the KID estimation is noisy and compute-intensive, we want to evaluate only after many iterations, but for many iterations.

As in [19] we evaluated the classification performance using a classification total accuracy measure. Additionally, we calculated confusion matrices, sensitivity, specificity, accuracy and f1score measures for each lesion category.

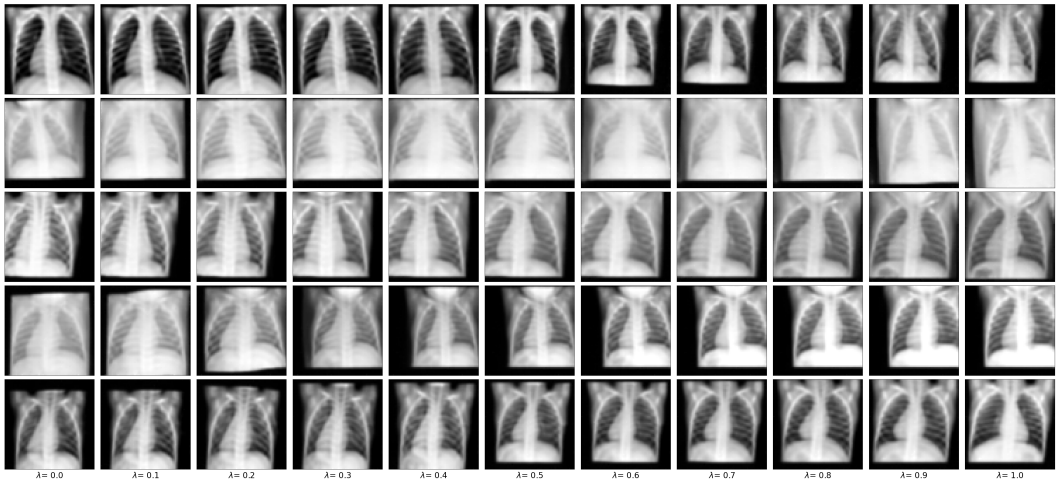
5.2 Architecture

We build a U-Net [18] with identical input and output dimensions for image denoising, according to Eq. 17.

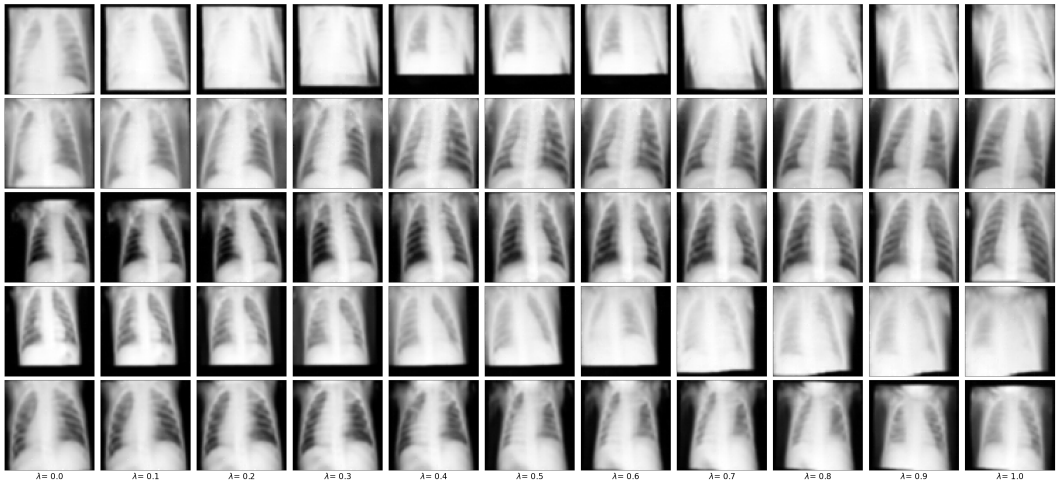
U-Net is a famous semantic segmentation architecture, whose central idea is that it progressively downsamples and then upsamples its input image, and adds skip connections between layers having the exact resolution. These help with gradient flow and avoid introducing a representation bottleneck, unlike usual autoencoders. Based on this, one can view diffusion models as denoising autoencoders without a bottleneck.

The network takes two inputs, the noisy images and the variances of their noise components. The latter is required since denoising a signal requires different operations at different levels of noise. We transform the noise variances using sinusoidal embeddings, similar to positional encodings used in transformers [29]. This helps the network to be susceptible to noise levels, which is crucial for good performance. We build the network using the *Tensorflow Keras* API on *Google Colab*. We embedded a function of the noise level as in score-based models to tackle time in a continuous manner so we can change the sampling schedule at inference time, without retraining the network. We do not input the embedding to each convolution block separately for simplicity as is commonly done in DMs.

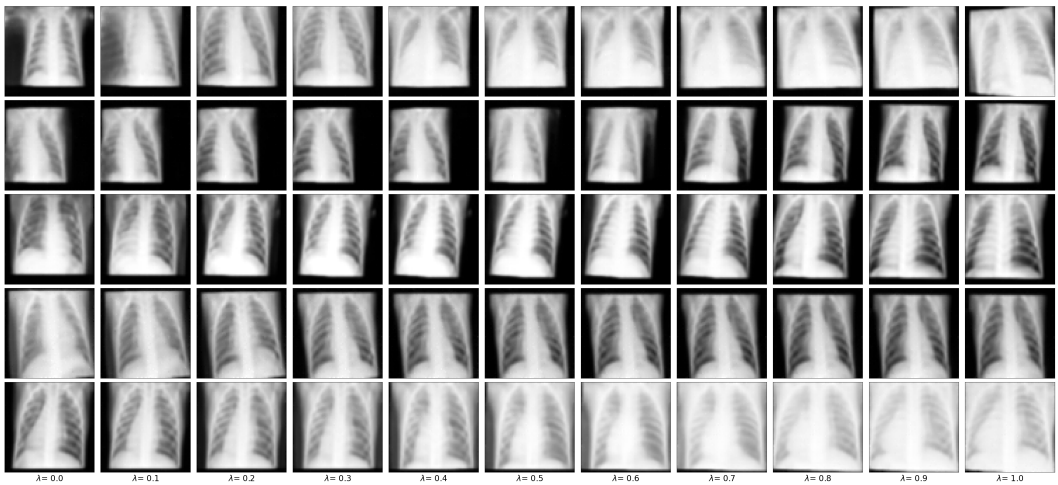
For the classification task, we consider a small convolutional network as in [19], and we report the architecture in Fig. 9. For both tasks, we adopt grayscale images with input size 64×64. We adopt the same training strategy, hyperparameters and augmentation techniques as in [19].



(a) *normal*

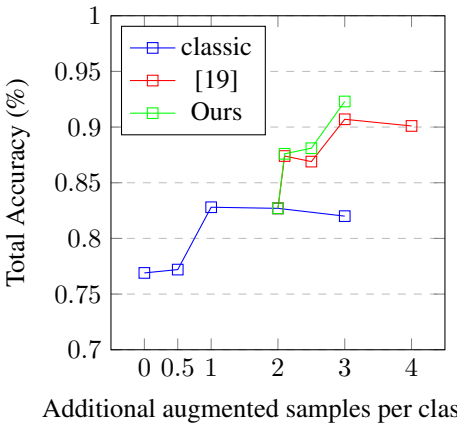


(b) *pneumonia bac*



(c) *pneumonia vir*

Figure 10: Interpolations in DDIM latent space of random samples about the *norma*, *pneumonia bac* and *pneumonia vir* disease. From left to right column interpolation goes from $\lambda = 0$ to $\lambda = 1$ with 0.1 interleave.



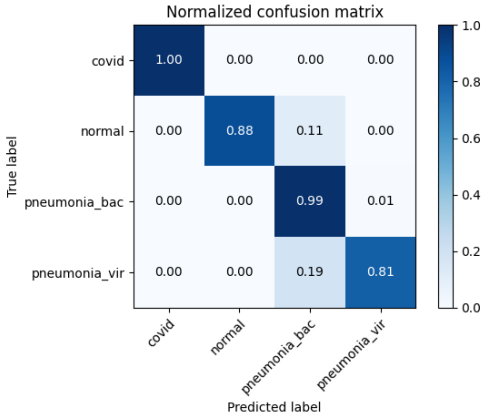
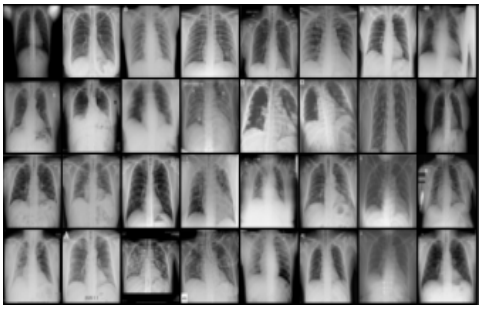


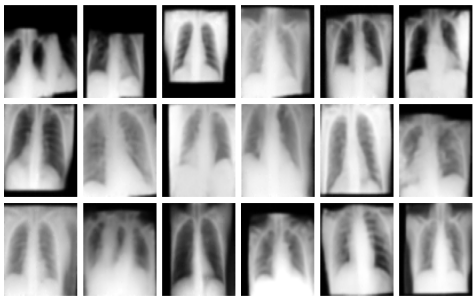
Figure 11: Total accuracy results for pneumonia infection classification of covid, normal, pneumonia bacteria and pneumonia virus with increasing training set size. The blue line shows the effect of adding classic data augmentation, and the red and green line shows the effect of adding synthetic data augmentation generated with GAN [19] and with DDIM (Ours).

6 Conclusion

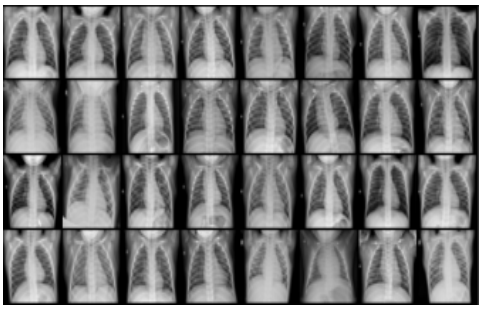
The lack of benchmark datasets in many medical fields was the primary motivation for this project. The main idea is to collect all the possible images for COVID-19 and use DDIMs to generate more images to help in the detection and classification of the virus from the available X-ray images. It was shown how the generated synthetic images could be used to augment the original dataset and eventually lead to a higher classification and detection accuracy of the chest X-ray images. We would like to investigate further generative adversarial architectures that generate multi-class samples together. Also, an extension of the presented work in domains different from medical tasks could be a priority to investigate in future.



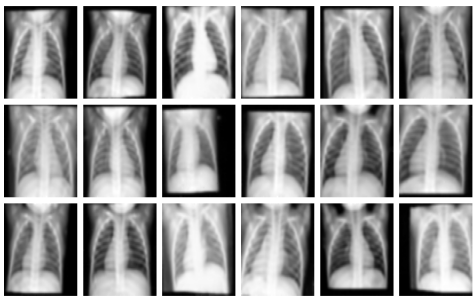
(a) covid real



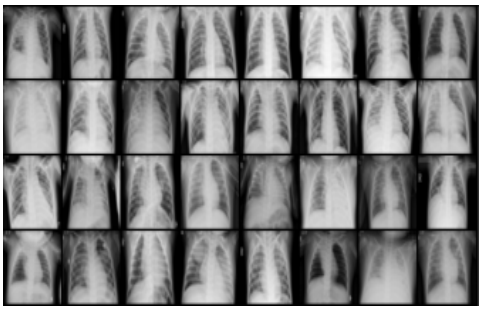
(b) covid synthetic



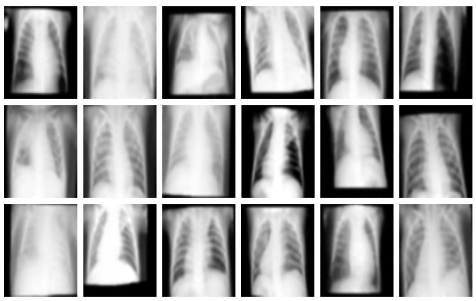
(c) normal real



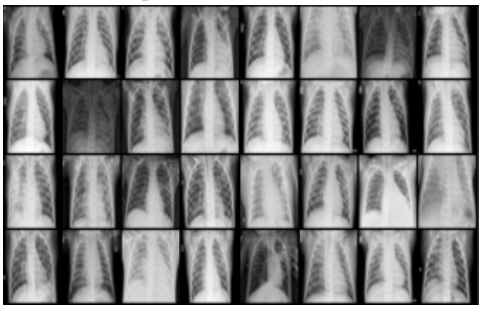
(d) normal synthetic



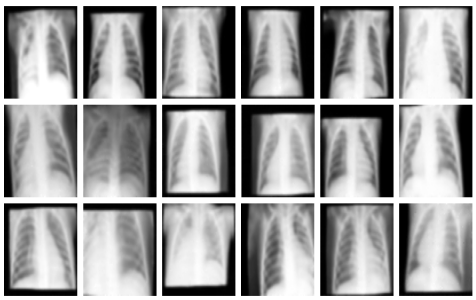
(e) pneumonia bacteria real



(f) pneumonia bacteria synthetic

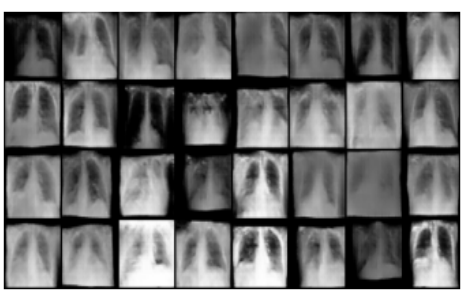


(g) pneumonia virus real

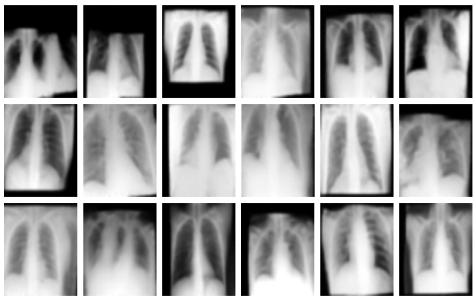


(h) pneumonia virus synthetic

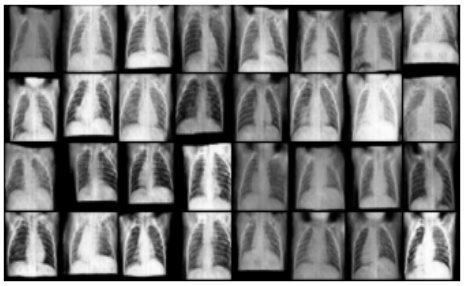
Figure 12: A pair of real (left) and fake X-ray images generated from DDIM (right). (a-b) for Covid ; (c-d) for Normal; (e-f) for Pneumonia bacteria; (g-h) for Pneumonia virus.



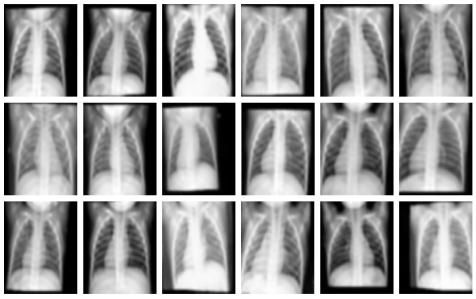
(a) covid synthetic GAN



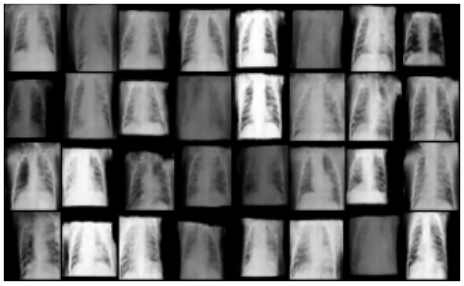
(b) covid synthetic DDIM



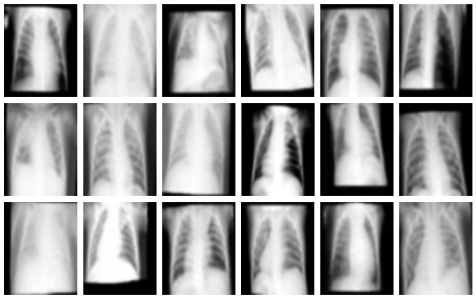
(c) normal synthetic GAN



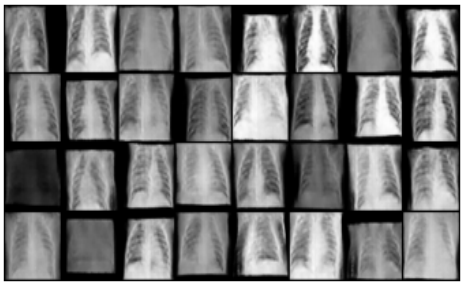
(d) normal synthetic DDIM



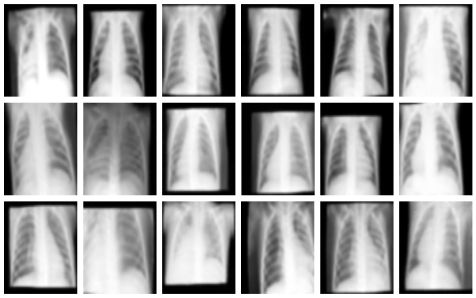
(e) pneumonia bacteria synthetic GAN



(f) pneumonia bacteria synthetic DDIM



(g) pneumonia virus synthetic GAN



(h) pneumonia virus synthetic DDIM

Figure 13: A pair of X-ray images generated from GAN [19] (left) and DDIM (right). (a-b) for Covid ; (c-d) for Normal; (e-f) for Pneumonia bacteria; (g-h) for Pneumonia virus.

References

[1] Martin Arjovsky and Leon Bottou. "Towards Principled Methods for Training Generative Adversarial Networks". In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=Hk4_qw5xe.

[2] Yoshua Bengio et al. "Deep Generative Stochastic Networks Trainable by Backprop". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 226–234. URL: https://proceedings.mlr.press/v32/bengio14.html.

[3] Mikołaj Bińkowski et al. *Demystifying MMD GANs*. 2018. DOI: 10.48550/ARXIV.1801.01401. URL: https://arxiv.org/abs/1801.01401.

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. *Large Scale GAN Training for High Fidelity Natural Image Synthesis*. 2018. DOI: 10.48550/ARXIV.1809.11096. URL: https://arxiv.org/abs/1809.11096.

[5] Joseph Paul Cohen, Paul Morrison, and Lan Dao. "COVID-19 image data collection". In: *arXiv 2003.11597* (2020). URL: https://github.com/ieee8023/covid-chestxray-dataset.

[6] Maayan Frid-Adar et al. "GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification". In: *Neurocomputing* 321 (Dec. 2018), pp. 321–331. DOI: 10.1016/j.neucom.2018.09.013. URL: https://doi.org/10.1016%2Fj.neucom.2018.09.013.

[7] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani et al. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[8] Martin Heusel et al. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: (2017). DOI: 10.48550/ARXIV.1706.08500. URL: https://arxiv.org/abs/1706.08500.

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851. URL: https://proceedings.neurips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

[10] Aapo Hyvärinen. "Estimation of Non-Normalized Statistical Models by Score Matching". In: *Journal of Machine Learning Research* 6.24 (2005), pp. 695–709. URL: http://jmlr.org/papers/v6/hyvarinen05a.html.

[11] Tero Karras, Samuli Laine, and Timo Aila. "A Style-Based Generator Architecture for Generative Adversarial Networks". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 43.12 (Dec. 2018), pp. 4217–4228. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2020.2970919. URL: https://doi.org/10.1109/TPAMI.2020.2970919.

[12] Karol Kurach et al. "A Large-Scale Study on Regularization and Normalization in GANs". In: *International Conference on Machine Learning*. 2019. URL: https://arxiv.org/abs/1807.04720.

[13] Andreas Lugmayr et al. *RePaint: Inpainting using Denoising Diffusion Probabilistic Models*. 2022. DOI: 10.48550/ARXIV.2201.09865. URL: https://arxiv.org/abs/2201.09865.

[14] Shakir Mohamed and Balaji Lakshminarayanan. *Learning in Implicit Generative Models*. 2016. DOI: 10.48550/ARXIV.1610.03483. URL: https://arxiv.org/abs/1610.03483.

[15] Vaishnavh Nagarajan and J. Zico Kolter. "Gradient Descent GAN Optimization is Locally Stable". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 5591–5600. ISBN: 9781510860964.

[16] Radford M. Neal. *Annealed Importance Sampling*. 1998. DOI: 10.48550/ARXIV.PHYSICS/9803008. URL: https://arxiv.org/abs/physics/9803008.

[17] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1278–1286. URL: https://proceedings.mlr.press/v32/rezende14.html.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. DOI: 10.48550/ARXIV.1505.04597. URL: https://arxiv.org/abs/1505.04597.

[19] Simone Rossetti and Akash Garg. "GAN-based Synthetic Medical Image Augmentation for increased CNN Performance in COVID-19 Classification". 2020. URL: https://github.com/rossettisimone/AUGMENTATION_GAN.

[20] Tim Salimans, Diederik Kingma, and Max Welling. "Markov Chain Monte Carlo and Variational Inference: Bridging the Gap". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 1218–1226. URL: https://proceedings.mlr.press/v37/salimans15.html.

[21] Tim Salimans et al. "Improved Techniques for Training GANs". In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee et al. Vol. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/file/8a3363abe792db2d8761d6403605aeb7-Paper.pdf.

[22] Jascha Sohl-Dickstein et al. "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 2256–2265. URL: https://proceedings.mlr.press/v37/sohl-dickstein15.html.

[23] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising Diffusion Implicit Models". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=St1giarCHLP.

[24] Jiaming Song, Shengjia Zhao, and Stefano Ermon. "A-NICE-MC: Adversarial Training for MCMC". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/2417dc8af8570f274e6775d4d60496da-Paper.pdf.

[25] Yang Song and Stefano Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

[26] Yang Song and Stefano Ermon. "Improved Techniques for Training Score-Based Generative Models". In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS'20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546.

[27] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=PxTIG12RRHS.

[28] Christian Szegedy et al. *Rethinking the Inception Architecture for Computer Vision*. 2015. DOI: 10.48550/ARXIV.1512.00567. URL: https://arxiv.org/abs/1512.00567.

[29] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.