Text-to-image generation with diffusion models

Leonardo de Lellis Rossi*, FEEC†

July 2022

Abstract

This work is a Final Project for the class IA025. Diffusion models are a class of generative models that convert Gaussian noise into samples from a learned data distribution via an iterative denoising process [2, 4, 7, 8, 11, 13, 14, 15, 18, 19]. This work proposes the implementation of text-to-image diffusion models, able to demonstrate the use of a diffusion model in the task of generating images from text on machines with limited memory and without access to multiple GPUs.

1 Introduction

In recent years, different types of generative models have been proposed: GANs [1, 3, 5, 19], VAEs [4, 19], flow-based models [4, 19] and diffusion models [4, 7, 8, 11, 13, 14, 15, 18, 19]. Each type has demonstrated great success in generating high-quality samples, however, each has some limitations of its own. GAN models are known for their potentially unstable training and lower generation diversity due to their contradictory training nature. The VAE depends on a surrogate loss. Flow models need to use specialized architectures to build the reversible transform.

Thus, this work[1] proposes implementations of text-to-image diffusion models (adaptation of Imagen from Saharia et al. (2022) [14]) with fewer parameters, able to demonstrate the use of diffusion models in the task of generating images from text on machines with limited memory and without access to multiple GPUs. In addition, the work investigates the use of Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID) in the search for better results and also analyzes the impact of the text conditioning scale applied in the Classifier-Free guidance.

2 Diffusion models

Diffusion models [4, 7, 8, 11, 13, 14, 15, 18, 19] are inspired by non-equilibrium thermodynamics [11, 18, 19]. They define a Markov chain of scattering steps to slowly add random noise to the data, and then learn to reverse the scattering process to build desired data samples from the noise. The model learns the reverse of the predefined diffusion process [2, 4, 7]. Transitions of this chain are learned to reverse a diffusion process, which is a Markov chain that gradually adds noise to the data in the opposite direction of sampling until the signal is destroyed [8, 15].

Diffusion consists of small amounts of Gaussian noise, so it is sufficient to set the sampling chain transitions to conditional Gaussians, allowing for a particularly simple neural network parameterization [7, 11, 18].

Diffusion models are similar to VAE, the encoder is defined as a fixed diffusion process rather than a learnable neural network, and the decoder is defined as a learnable denoising process that generates data [15, 18]. But unlike VAE or flow models, diffusion models are learned with a fixed procedure and the latent variable has high dimensionality (same as the original data).

2.1 Forward diffusion process

Given a data point sampled from a real data distribution ($x_0 \sim q(x)$), a forward diffusion process adds a small amount of Gaussian noise to the sample in T steps, producing a sequence of noisy samples $(x_1, ..., x_T)$ [7, 8, 11, 18]. The step sizes are controlled by a variance schedule $\{\beta_t \in (0,1)\}_{t=1}^{T}$.

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

*1261900@dac.unicamp.br

†Faculty of Electrical and Computer Engineering (FEEC) - Unicamp - Campinas / Brazil

[1]Code available at https://github.com/leolellisr/deep_learning_projects/tree/main/11_txt2img

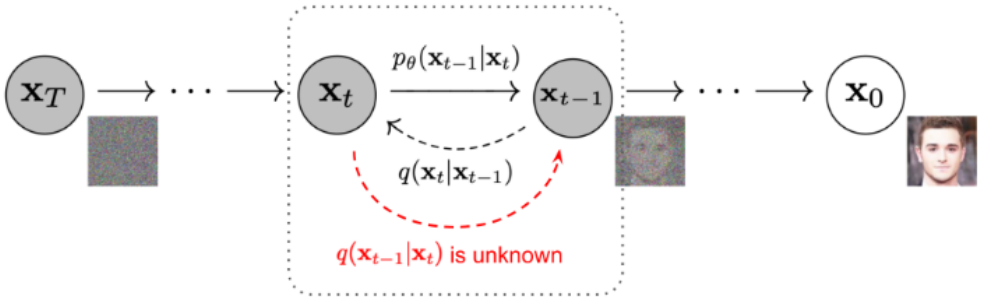


Figure 1: Diffusion Process. Up: Forward diffusion process; Down: Backward diffusion process [7].

The data sample x_0 gradually loses its distinguishable features as the step becomes larger. Eventually when $T \to \infty, x_T$ is equivalent to an isotropic Gaussian distribution [8, 19].

The input x_t can be sampled at any arbitrary time step t in a closed form using reparameterization, like $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$ [8, 11, 19].

$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\mathbf{z}_{t-1}$ where $\mathbf{z}_{t-1}, \mathbf{z}_{t-2}, \ldots \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
$= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\mathbf{z}}_{t-2}$ where $\bar{\mathbf{z}}_{t-2}$ merges two Gaussians (*).
$= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z}$

The forward process variances β_T can be learned by reparameterization or held constant as hyperparameters, and expressiveness of the reverse process is ensured in part by the choice of Gaussian conditionals in $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, because both processes have the same functional form when β_T are small [7, 15]. A notable property of the forward process is that it admits sampling x_t at an arbitrary timestep t in closed form: using the notation $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$ [7, 8, 11]

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

When two Gaussians are merged with different variance, $\mathcal{N}(\mathbf{0}, \sigma_1^2\mathbf{I})$ and $\mathcal{N}(\mathbf{0}, \sigma_2^2\mathbf{I})$, the new distribution is $\mathcal{N}(\mathbf{0}, (\sigma_1^2+\sigma_2^2)\mathbf{I})$. Here the merged standard deviation is $\sqrt{(1-\alpha_t) + \alpha_t(1-\alpha_{t-1})} = \sqrt{1-\alpha_t\alpha_{t-1}}$ [7, 8, 11].

A larger update step can be afforded when the sample gets noisier, so $\beta_1 < \beta_2 < \ldots < \beta_T$ and $\bar{\alpha}_1 > \ldots > \bar{\alpha}_T$ [7, 15, 19].

Figure 2: Latent samples from linear (top) and cosine (bottom) schedules respectively at linearly spaced values of t from 0 to T. The latents in the last quarter of the linear schedule are almost purely noise, whereas the cosine schedule adds noise m [19]

2.2 Backward diffusion process

The diffusion process can be reversed and sample from $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$, so it can be able to recreate the true sample from a Gaussian noise input, $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ [7, 8, 11]. If β_t is small enough, $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ will also be Gaussian. Unfortunately, $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ cannot easily be estimated because it needs to use the entire dataset. Therefore, a model needs to learn to approximate these conditional probabilities to run the backward diffusion process.

$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \quad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$

The reverse conditional probability is tractable when conditioned on x_0 [7, 8, 11]:

$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t\mathbf{I})$

Using Bayes' rule:

$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)}$

$\propto \exp\Big(-\frac{1}{2}\big(\frac{(\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_{t-1})^2}{\beta_t} + \frac{(\mathbf{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big)$

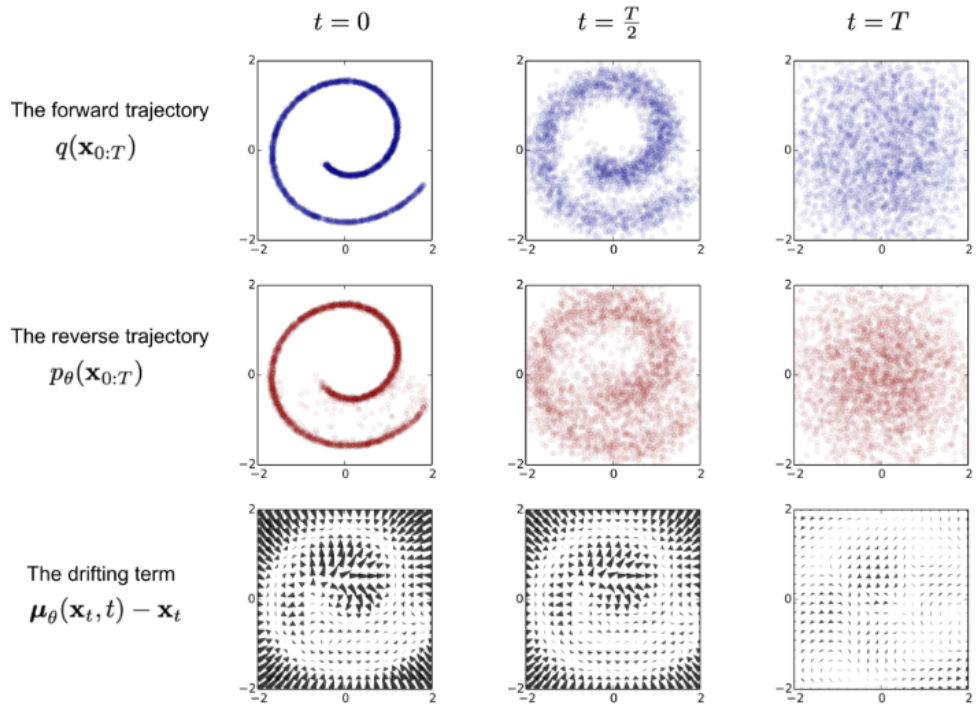


Figure 3: An example of training a diffusion model for modeling a 2D swiss roll data [17]. The top row shows time slices from the forward trajectory $q(x^{(0\ldots T)})$. The data distribution (left) undergoes Gaussian diffusion, which gradually transforms it into an identity-covariance Gaussian (right). The middle row shows the corresponding time slices from the trained reverse trajectory $p(x^{(0\ldots T)})$. An identity-covariance Gaussian (right) undergoes a Gaussian diffusion process with learned mean and covariance functions and is gradually transformed back into the data distribution (left). The bottom row shows the drift term, $f_\mu(x^t, t) - x^t$, for the same reverse diffusion process.

$= \exp\Big(-\frac{1}{2}\big(\frac{\mathbf{x}_t^2 - 2\sqrt{\alpha_t}\mathbf{x}_t\mathbf{x}_{t-1} + \alpha_t\mathbf{x}_{t-1}^2}{\beta_t} + \frac{\mathbf{x}_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\mathbf{x}_{t-1} + \bar{\alpha}_{t-1}\mathbf{x}_0^2}{1-\bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big)$
$= \exp\Big(-\frac{1}{2}\big((\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})\mathbf{x}_{t-1}^2 - (\frac{2\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)\mathbf{x}_{t-1} + C(\mathbf{x}_t, \mathbf{x}_0)\big)\Big)$

Following the standard Gaussian density function, the mean and variance can be parameterized as follows [7, 15, 19]:

$\tilde{\beta}_t = 1/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}) = 1/(\frac{\alpha_t - \bar{\alpha}_t + \beta_t}{\beta_t(1-\bar{\alpha}_{t-1})}) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t$

$\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) = (\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})$
$= (\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t$

$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0$

Making $\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\mathbf{z}_t)$ and pluging it into the above equation, it is obtained:

$\tilde{\mu}_t = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\mathbf{z}_t)$
$= \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_t\Big)$

Such a setup is very similar to VAE and thus a variational lower bound (VLB) can be used to optimize the negative log-likelihood [2, 8, 11].

$-\log p_\theta(\mathbf{x}_0) \leq -\log p_\theta(\mathbf{x}_0) + D_{KL}(q(\mathbf{x}_{1:T}|\mathbf{x}_0)\|p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0))$
$= -\log p_\theta(\mathbf{x}_0) + E_{\mathbf{x}_{1:T}\sim q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})/p_\theta(\mathbf{x}_0)}\Big]$
$= -\log p_\theta(\mathbf{x}_0) + E_q\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})} + \log p_\theta(\mathbf{x}_0)\Big]$
$= E_q\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\Big]$

Let $L_{VLB} = E_{q(\mathbf{x}_{0:T})}\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\Big] \geq -E_{q(\mathbf{x}_0)}\log p_\theta(\mathbf{x}_0)$

To convert each term in the equation to be analytically computable, the objective can be further rewritten to be a combination of several KL-divergence and entropy terms [8, 11, 15]:

$$
\begin{aligned}
& L_{VLB} = E_{q(\mathbf{x}_{0:T})}\Big[\log \tfrac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\Big] \\
&= E_q\Big[\log \tfrac{\prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= E_q\Big[-\log p_\theta(\mathbf{x}_T) + \textstyle\sum_{t=1}^{T}\log \tfrac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= E_q\Big[-\log p_\theta(\mathbf{x}_T) + \textstyle\sum_{t=2}^{T}\log \tfrac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \tfrac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= E_q\Big[-\log p_\theta(\mathbf{x}_T) + \textstyle\sum_{t=2}^{T}\log \Big(\tfrac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\cdot\tfrac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}\Big) + \log \tfrac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= E_q\Big[-\log p_\theta(\mathbf{x}_T) + \textstyle\sum_{t=2}^{T}\log \tfrac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \sum_{t=2}^{T}\log \tfrac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)} + \log \tfrac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= E_q\Big[-\log p_\theta(\mathbf{x}_T) + \textstyle\sum_{t=2}^{T}\log \tfrac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \tfrac{q(\mathbf{x}_T|\mathbf{x}_0)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \tfrac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= E_q\Big[\log \tfrac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \textstyle\sum_{t=2}^{T}\log \tfrac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\Big] \\
&= E_q[\underbrace{D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)\parallel p_\theta(\mathbf{x}_T))}_{L_T} + \textstyle\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}]
\end{aligned}
$$

The variational lower bound (VLB) loss equation can be written as: $L_{VLB} = L_T + L_{T-1} + \ldots + L_0$

where:

$L_T = D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))$

$L_t = D_{KL}(q(\mathbf{x}_t|\mathbf{x}_{t+1},\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}))$ for $1 \le t \le T-1$

$\mathrm{L}_0 = -\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)$

Every KL term in L_{VLB} (except for L_0) compares two Gaussian distributions and therefore they can be computed in closed form. L_T is constant and can be ignored during training because q has no learnable parameters and x_T is a Gaussian noise [8, 19].

2.3 Training process

A diffusion model is trained by optimizing the sum of denoising score matching losses for various noise levels which aims to learn the recovery of clean images from corrupted images [2]. Instead of using a simple sum of losses, Ho et al. (2020) [7] observed that their empirically obtained weighted sum of losses was more beneficial to sample quality. Their weighted objective is the current standard objective for training diffusion models. However, it remains unknown why this performs well or whether it is optimal for sample quality [2, 7].

In forward to a neural network learns to approximate the conditioned probability distributions in the reverse diffusion process, $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t,t), \Sigma_\theta(\mathbf{x}_t,t))$, μ_θ would have to be trained to predict $\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_t\Big)$. Because x_t is available as input at training time, the Gaussian noise term can be reparameterized instead to predict z_t from the input x_t at time step t:

$\mu_\theta(\mathbf{x}_t,t) = \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_\theta(\mathbf{x}_t,t)\Big)$

Thus $\mathbf{x}_{t-1} = \mathcal{N}(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_\theta(\mathbf{x}_t,t)\Big), \Sigma_\theta(\mathbf{x}_t,t))$

The loss term L_t is parameterized to minimize the difference from $\tilde{\mu}$: $L_t = E_{\mathbf{x}_0,\mathbf{z}}\Big[\frac{1}{2\|\Sigma_\theta(\mathbf{x}_t,t)\|_2^2}\|\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)\|^2\Big]$

$= \mathrm{E}_{\mathbf{x}_0,\mathbf{z}}\Big[\frac{1}{2\|\Sigma_\theta\|_2^2}\|\frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_t\Big) - \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_\theta(\mathbf{x}_t,t)\Big)\|^2\Big]$

$= \mathrm{E}_{\mathbf{x}_0,\mathbf{z}}\Big[\frac{\beta_t^2}{2\alpha_t(1-\bar{\alpha}_t)\|\Sigma_\theta\|_2^2}\|\mathbf{z}_t - \mathbf{z}_\theta(\mathbf{x}_t,t)\|^2\Big]$

$= E_{\mathbf{x}_0,\mathbf{z}}\Big[\frac{\beta_t^2}{2\alpha_t(1-\bar{\alpha}_t)\|\Sigma_\theta\|_2^2}\|\mathbf{z}_t - \mathbf{z}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z}_t, t)\|^2\Big]$

Empirically, Ho et al. (2020) [7] found that training the diffusion model works better with a simplified objective that ignores the weighting term: $L_t^{simple} = E_{\mathbf{x}_0,\mathbf{z}_t}\Big[\|\mathbf{z}_t - \mathbf{z}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z}_t, t)\|^2\Big]$

The final simple objective is: $L_{simple} = L_t^{simple} + C$, where C is a constant not depending on θ.

3 Imagen

Imagen is based on a transformers language model in text comprehension and hinges on the strength of diffusion models in generating images [14]. The authors report that generic large language models (e.g. T5), pre-trained on text-only corpora, are effective in encoding text for image synthesis: increasing

the size of the language model in Imagen increases sample fidelity and image text alignment much more than increasing the image diffusion model size.

Imagen achieves a Fréchet Inception Distance (FID) score of 7.27 in the COCO dataset, without ever training on COCO [14]. Imagen consists of a text encoder that maps text to a sequence of embeddings and a cascade of conditional diffusion models that map these embeddings to images of increasing resolutions.

Architecturally, it consists of a cascading DDPM conditioned on text embeddings from a large pre-trained T5 model (attention network) [14]. It also contains dynamic clipping for improved classifier-free guidance, noise level conditioning, and a memory efficient uNet design. Imagen utilizes a pipeline of a base 64×64 model, and two text-conditional super-resolution diffusion models to upsample a 64 × 64 generated image into a 256 × 256 image, and then to 1024 × 1024 image [14].

For the base model, they adapt the U-Net architecture from Nichol and Dhariwal (2021) [11] for the base 64×64 text-to-image diffusion model. The network is conditioned on text embeddings via a pooled embedding vector, added to the diffusion timestep embedding. Imagen also conditions the entire sequence of text embeddings by adding cross attention over the text embeddings at multiple resolutions. The authors also state that Layer Normalization for text embeddings in the attention and pooling layers helps considerably improve performance [14].

For the super-resolution models (64 × 64 → 256 × 256), they also use the U-Net model adapted from Nichol and Dhariwal (2021) [11]. The 256×256 → 1024×1024 super-resolution model trains on 64×64 → 256×256 crops of the 1024 × 1024 image. During inference, the model receives the full 256 × 256 low-resolution images as inputs and returns upsampled 1024 × 1024 images as outputs.

4 Proposal

This work proposes implementations of the base 64×64 text-to-image diffusion model of Imagen [14] (adaptation of U-Net architecture from Nichol and Dhariwal (2021) [11, 12]) with fewer parameters, to demonstrate the use of diffusion models in the task of generating images from text on machines with limited memory and without access to multiple GPUs. In addition to the models, this work presents an analysis of the impact of the text conditioning scale and the use of Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID) metrics to improve the models' performance.

5 Methodology

It is proposed to implement text-to-image models, based on the Imagen architecture, using only the base 64x64 UNet of Nichol and Dhariwal (2021) [11, 12]) with fewer parameters, to demonstrate the use of diffusion models in the task of generating images from text on a machine with only one GPU and memory limitations.

COCO Dataset is used for training, validation, and testing. The images are used as input for the diffusion model with the embeddings and attentional masks generated by a T5 encoder model from the image captions. The model is trained based on the perception prioritized (P2) loss, proposed by Choi et. al (2021) [2]. The model is also validated with the Fréchet Inception Distance (FID) [6]. The model training, validation, and testing flowchart is shown in Figure 4.

5.1 Machine specifications

The machine used to run all the experiments has the following specifications: (i) **PC name**: Intel(R) Xeon(R) CPU @ 2.00GHz; (ii) **GPU name**: GPU 0: Tesla P100-PCIE-16GB; (iii) **Memory available**: 26GB; (iv) **Hard disk available**: 127GB; (v) **Number of sockets (available slots for physical processors)** : 1; (vi) **Number of cores each processor has**: 2; (vii) **Number of threads each core has**: 2.

5.2 Pretrained text encoder

Imagen explores pretrained text encoders: BERT, T5 and CLIP [14]. For simplicity, they froze the weights of these text encoders. Freezing has several advantages such as offline computation of embeddings, resulting in negligible computation or memory footprint during training of the text-to-image model. In their work, they find that there is a clear conviction that scaling the text encoder size improves the quality of text-to-image generation [14]. In this work, for memory limitation, it is used the base T5 encoder model provided by Google [16].

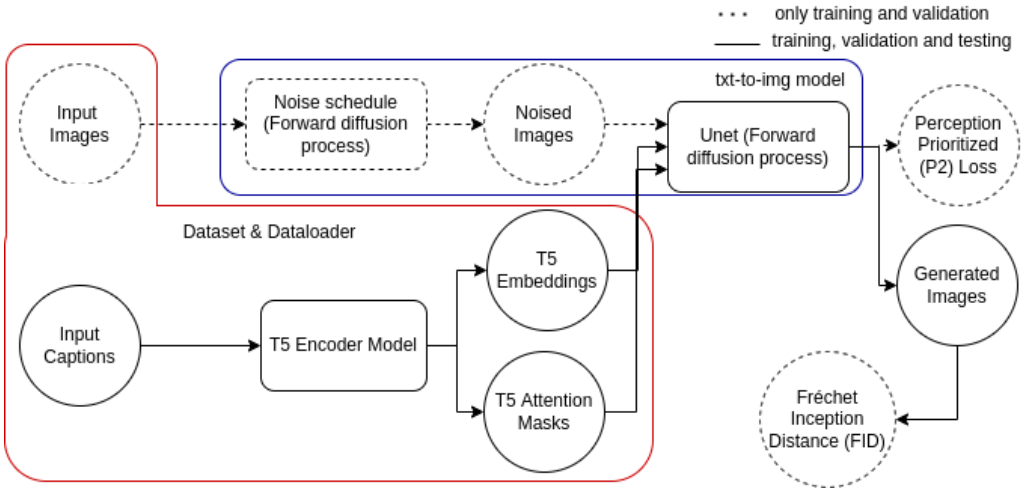


Figure 4: Model training, validation, and testing flowchart. Dashed components only are used during training and validation. Other components are used during training, validation, and testing.

5.3 Diffusion model

Diffusion models are a class of generative models that convert Gaussian noise into samples from a learned data distribution via an iterative denoising process [2, 14]. These models can be conditional, for example on class labels, text, or low-resolution images. In this work, as in Imagen [14], a diffusion model x is trained on a denoising objective of the form:

$$E_{\mathbf{x},\mathbf{c},\epsilon,t}\left[w_t \left\|\hat{\mathbf{x}}_\theta\left(\alpha_t \mathbf{x}+\sigma_t \epsilon, \mathbf{c}\right)-\mathbf{x}\right\|_2^2\right]$$

Where x and c are data-conditioning pairs; $t \sim \mathcal{U}([0,1]), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$; and α_t, θ_t, w_t are functions of t that influence sample quality. In this work, $\hat{\mathbf{x}}_\theta$ is trained to denoise $z_t := \alpha_t \mathbf{x} + \theta_t$ into x using a perception prioritized (P2) loss (different from Imagen [14], that uses a squared error loss, weighted to emphasize certain values of t). The sampling is made such as DDIM [2] starting from pure noise $z_1 \sim \mathcal{N}(0, \mathbf{I})$ and iteratively generating points that gradually decrease in noise content [2, 14].

5.4 Classifier-free guidance

Classifier guidance is a technique to improve sample quality while reducing diversity in conditional diffusion models using gradients from a pretrained model $p(c|z_t)$ during sampling [13, 14]. Classifier-free guidance is an alternative technique that avoids this pretrained model by instead jointly training a single diffusion model on conditional and unconditional objectives via randomly dropping c during training (e.g. with 10% probability). GLIDE [13] and Imagen [14] depend critically on classifier-free guidance for effective text conditioning. For this work, sampling is performed using the adjusted x-prediction ($z_t - \sigma\tilde{\epsilon}_\theta)/\alpha_t$, where

$\tilde{\epsilon}_\theta\left(\mathbf{z}_t, \mathbf{c}\right) = w\epsilon_\theta\left(\mathbf{z}_t, \mathbf{c}\right) + (1-w)\epsilon_\theta\left(\mathbf{z}_t\right)$

$\epsilon_\theta(\mathbf{z}_t, \mathbf{c})$ and $\epsilon_\theta(\mathbf{z}_t)$ are conditional and unconditional ϵ-predictions, given by $\epsilon_\theta := (\mathbf{z}_t - \alpha_t \hat{x})/t$, and w is the guidance weight. Setting $w = 1$ disables classifier-free guidance, while increasing $w > 1$ strengthens the effect of guidance.

5.5 Perception Prioritized (P2) Loss

The work of Kingma et al. (2021) [8] simplified the noise schedules of diffusion models in terms of signal-to-noise ratio (SNR). SNR of corrupted data x_t is a ratio of squares of mean and variance, which can be written as: $SNR(t) = \alpha_t/(1-\alpha_t)$

The variance of noisy data x_t can be written in terms of SNR: $\alpha_t = 11/(1 + SNR(t))$. SNR(t) is a monotonically decreasing function.

Similar to VAE, diffusion models can trained by optimizing a variational lower bound (VLB), which is a sum of denoising score matching losses [2, 15]:

$$L_{VLB} = \sum_t L_t$$

Where weights for each loss term are uniform. For each step t, denoising score matching loss L_t is a distance between two Gaussian distributions, which can be rewritten in terms of a noise predictor ϵ_θ as:

$$L_t = D_{KL}\left(q\left(x_{t-1} \mid x_t, x_0\right) \| p_\theta\left(x_{t-1} \mid x_t\right)\right)$$

$$=\mathrm{E}_{x_0,\epsilon}\left[\frac{\beta_t}{(1-\beta_t)(1-\alpha_t)}\left\|\epsilon - \epsilon_\theta\left(x_t, t\right)\right\|^2\right]$$

A neural network ϵ_θ can be trained to predict the noise ϵ added in a noisy image x_t for given time step t. As shown in section 2, Ho et al. (2020) [7] empirically observed that the following simplified objective is more beneficial to sample quality:

$$L_{simple} = \sum_t E_{x_0,\epsilon}\left[\left\|\epsilon - \epsilon_\theta\left(x_t, t\right)\right\|^2\right]$$

In terms of variational lower bound (VLB), the objective can be represented as $L_{simple} = \sum_t$, with weighting scheme $\lambda_t = (1-\beta_t)(1-\alpha_t)/\beta_t$. In a continuous-time setting, this weighting scheme can be expressed in terms of SNR:

$\lambda_t^{'} = -1/log - SNR'(t) = -SNR(t)/SNR'(t)$

Where $SNR'(t) = d(SNR(t))/dt$.

A perception prioritized (P2) loss was proposed by Choi et. al (2021) [2], a weighting scheme for the diffusion objective training, which aims to prioritize learning from more important noise levels. The authors opt to assign minimal weights to the unnecessary clean-up stage thereby assigning relatively higher weights to the rest. In particular, they aim to emphasize training on the content stage to encourage the model to learn perceptually rich contexts. To this end, they construct the following weighting scheme:

$$\lambda_t' = \frac{\lambda_t}{(k + SNR(t)^\gamma)}$$

where λ_t is the previous standard weighting scheme and γ is a hyperparameter that controls the strength of down-weighting focus on learning imperceptible details. k is a hyperparameter that prevents exploding weights for extremely small SNRs and determines the sharpness of the weighting scheme.

Prior works [2, 11, 7] suggest that the baseline objective $\sum_t = \lambda_t L_t$ offers a better inductive bias for sample quality than the VLB objective $\sum_t = L_t$, which does not impose any inductive bias during training.

The work Kingma et al. [8] simplified the noise schedules of diffusion models in terms of signal-to-noise ratio (SNR). Choi et al. (2021) [2] indicate that linear and cosine noise schedules focus training on the content stage the most and the cleaning stage the least. The baseline weighting hypothesizes that models learn perceptually rich content by solving pretext tasks at the content stage.

5.6 Fréchet Inception Distance (FID)

The models are also validated with the Fréchet Inception Distance (FID), a metric used to assess the quality of images created by a generative model (initially proposed for GANs, but used in the Imagen paper) [6, 2, 14]. Unlike the Inception Score (IS), which only evaluates the distribution of the generated images, the Fréchet Inception Distance (FID) compares the distribution of the generated images with the distribution of the real images that were used to train the generator [6].

6 Dataset

MS COCO dataset contains images of complex everyday scenes containing people, animals, and objects in their natural context [9]. The scenes are labeled using per-instance segmentations to aid in precise object localization, with a total of 2.5 million labeled instances in 328k images. For caption annotation, COCO provides five written caption descriptions for each image.

The Python API of COCO [10] was used to download the captioning dataset (2014 version). For memory limitations, only 20k images were downloaded, 15k for training (75%), 3k for validation (15%), and 2k for testing (10%). Since COCO provides 5 written caption descriptions to each image, the total examples are 100k examples, 75k for training (75%), 15k for validation (15%), and 10k for testing (10%). The images and captions were saved on Google Drive for optimization.

Table 1: Parameters and best results for trained models. * Parameters percentages are based on Imagen's [14] total parameters.

model	small	base	large	full	Imagen [14]
batch size	32	64	64	32	1024
max batch size	4	8	8	4	-
max lenght	64	128	256	256	512
diffusion steps	500	500	1000	1000	1000
unet emb dim	32	32	64	512	512
unet hidden size	64	64	128	256	-
eval every epochs	100	200	1000	100	-
eval val steps	100	100	1000	100	-
cond scale	5	8	8	5	-
early stops	-	5000	20k	2000	-
total parameters*	4.87M (0.174%)	9.75M (0.34%)	19.5M (0.697%)	58.78M (2.1%)	2.8B
total epochs	28k	28k	21k	5k	2.5M
best p2 loss	-	0.0152	0.0156	0.05	-
best FID score	-	318.15	214.54	-	7.27

7 Experiments

This work experiments, following the methodology described in Chapter 5, have been done by feeding text-to-image diffusion models with images (which have Gaussian noise added through the forward diffusion Process) and with embeddings and attention masks collected from annotations with a T5 Encoder model, to minimize the prioritized perception (P2) loss and validate the images generated by these models also with Fréchet Inception Distance (FID).

Considering the limitations of the machine used, different versions of the text-to-image model were implemented, with different parameters, to optimize the training and improve the performance of the models. A comparison between the different parameters and est results of each version of the implemented model and the data available in the Imagen paper is presented in Table 1.

The following versions of this work's text-to-image model have been implemented: (i) **small**: version used to test the model architecture; (ii) **base**: version used to test overfit in one batch; (iii) **large**: version used for parameter optimization between base and full version; (iv) **full**: version with some Imagen parameters (with limitations).

7.1 Overfit with one batch

To test the functioning of the implemented architecture, the base version of the text-to-image diffusion model was trained with a single batch in 4k steps. The results obtained during training and validation are shown in Figure 5. Some of the images generated by the overfitted model are shown in Figure 6.

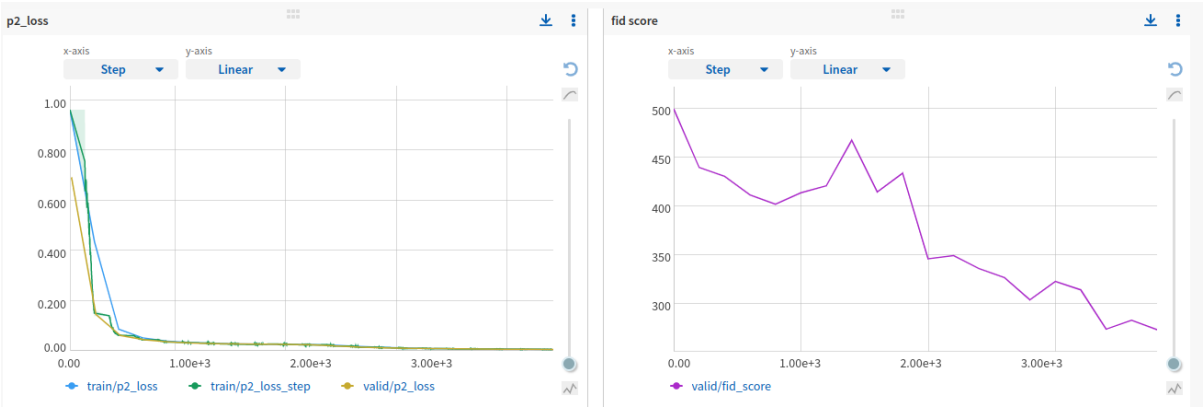


Figure 5: Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID) with the overfitted model. Complete results in `https://app.neptune.ai/leolellisr/dl-ia025`.

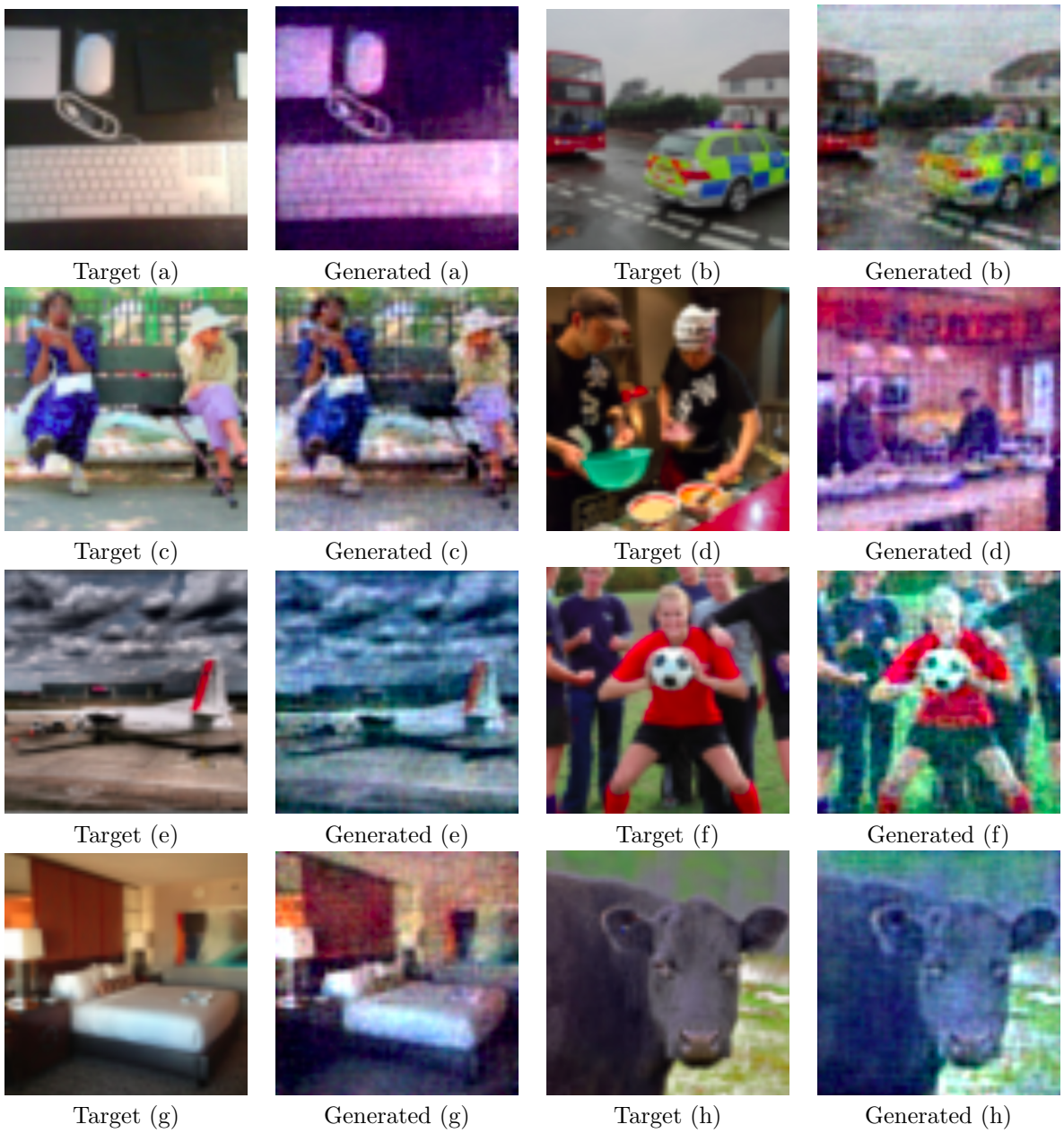


Figure 6: Images generated with the overfitted model and respective targets. **Captions:** (a) A mouse a keyboard and other computer accessories; (b) A Police car sitting in the way of a double deck tour bus; (c) A woman with a cane sitting next to another woman on a bench; (d) Two chefs in a kitchen preparing a meal; (e) an airplane sitting on a tarmac underneath a very cloudy sky; (f) An all girl crowd with one extra excited girl in the middle holding a football, looks like a goalie to me; (g) This is a big bed with a brown head board and a painting on the wall; (h) Black cow stands and look towards the camera. Complete results in `https://app.neptune.ai/leolellisr/dl-ia025`.

7.2 Analysis of text conditioning in Classifier-Free Guidance

To verify the influence of the text conditioning scale of the Free Guidance classification, two base models were trained with text conditioning scales 5 and 8. The results obtained during training and validation are shown in Figure 7. Some of the images generated by the two base models are shown in Figures 8 and 9.

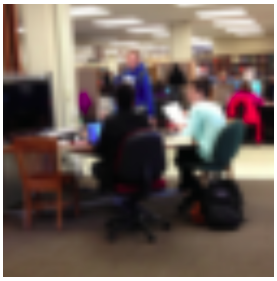
7.3 Training of large model

During the process of optimizing and training the large version of the text-to-image model (version with more parameters than the base one, but which is still capable of being retrained on the available machine), we chose to validate and save the model in two different methods: (i) only with the Perception

cond. scale = 5

cond. scale = 8

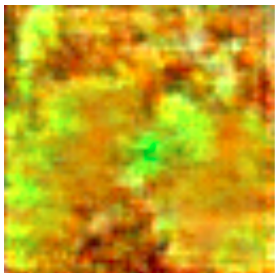
Figure 7: Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID) with the models trained with text conditioning scales 5 and 8. Complete results in `https://app.neptune.ai/leolellisr/dl-ia025`.



Target (a)

Generated (a)

Target (b)



Generated (b)

Target (c)

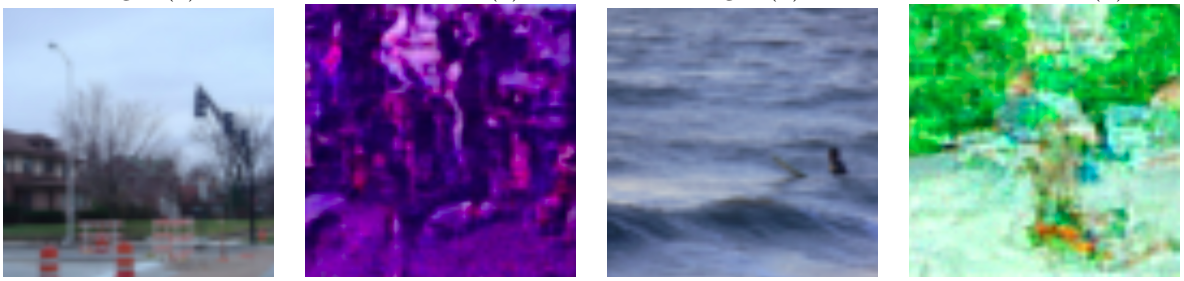
Generated (c)

Target (d)

Generated (d)

Figure 8: Images generated with the base model trained, text conditioning scale 5, and respective targets. **Captions:** (a) The people are in the library working and studying; (b) A girl sitting at a table full of bananas; (c) A group of people on the beach next to several surfboards; (d) A bride and groom are kissing on the beach. Complete results in `https://app.neptune.ai/leolellisr/dl-ia025`.

Target (a) Generated (a) Target (b) Generated (b)



Target (c) Generated (c) Target (d) Generated (d)

Figure 9: Images generated with the base model trained, text conditioning scale 8, and respective targets. **Captions:** (a) Roses in full bloom sit in a vase; (b) A man grinds a rail slide on a skateboard; (c) Some street lights hanging up above a wet street; (d) A man is in the wavy water with his surfboard. Complete results in `https://app.neptune.ai/leolellisr/dl-ia025`.

Prioritized (P2) Loss; (ii) with Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID). The results obtained during training and validation are shown in Figure 10. Some of the images generated by the two methods are shown in Figures 11 and 12.

8 Conclusion

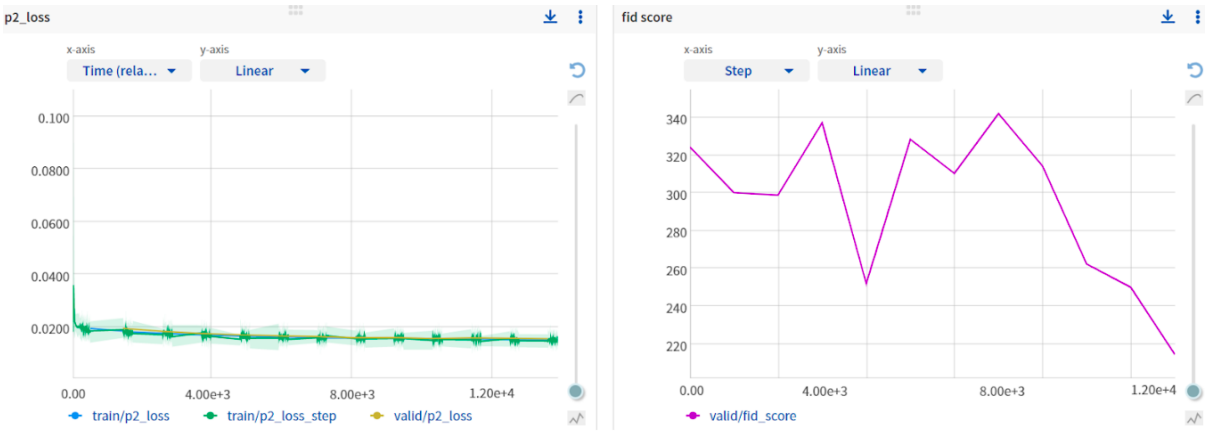
In this work, diffusion models were implemented to generate images from text, based on the architecture of Imagen's base model. Although the models still do not generalize well, generating still noisy images and presenting a Fréchet Inception Distance (FID) much larger than the based architecture, it was possible to demonstrate the use of diffusion models to generate images from text. The images generated by the base and large models match in shapes and colors with the available captions.

In the study of the text conditioning scale, it was noticed that larger scales (up to 10) collaborated for better Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID) results. While using a 5 text conditioning scale returned a Perception Prioritized (P2) Loss of (0.0168) and Fréchet Inception Distance (FID) of (344.7), using an 8 text conditioning scale returned a Perception Prioritized (P2) Loss of (0.0152) and Fréchet Inception Distance (FID) of (333.65). The images generated by the model that used a text conditioning scale of 8 also matched the input texts better.

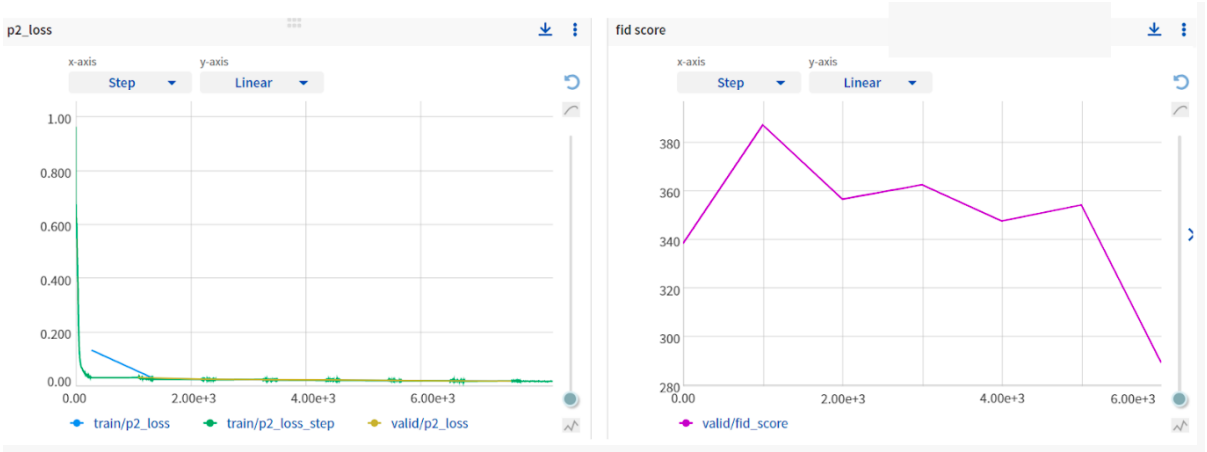
In the analysis of the validation metrics to save the model during the training and optimization of the large model, it was found that the use of only Perception Prioritized (P2) Loss produces better results for Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID) in less training time. However, the use of Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID) promoted better-generated images.

It was also possible to demonstrate the feasibility of using diffusion models in the task of generating images from text, through the overfit in a batch. Finally, it was possible to generate a comparison between the validation with Perception Prioritized (P2) Loss and with Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID).

In conclusion, text-to-image diffusion models were successfully implemented by applying the knowledge acquired in class IA025 Introduction to Deep Learning, taught by professors Rodrigo Nogueira and Roberto Lotufo in the 1st semester of 2022.



only Perception Prioritized (P2) Loss



Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID)

Figure 10: Perception Prioritized (P2) Loss and Fréchet Inception Distance (FID) with the models trained with text conditioning scales 5 and 8. Complete results in `https://app.neptune.ai/leolellisr/dl-ia025`.

Target (a)

Generated (a)

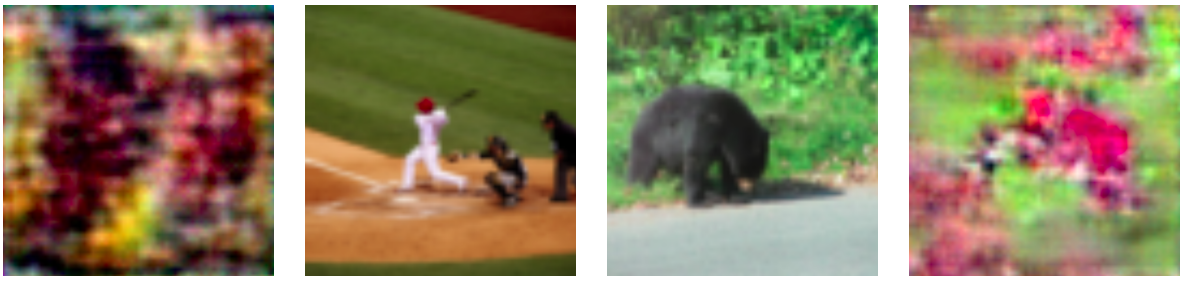
Target (b)

Generated (b)

Figure 11: Images generated with the large model - saving the model based on just Perception Prioritized (P2) Loss. **Captions:** (a) some birds are flying over a wave over the ocean; (b) Two cows stand in a field near a wire fence. Complete results in `https://app.neptune.ai/leolellisr/dl-ia025`.

9 Future Work

Among possible future works, it is believed that is possible: (i) Use of a machine with more memory and processing capacity; (ii) Use of a larger set of data for training, validation, and testing; (iii) Use of the same T5 encoder model as Imagen [14];(iv) Use of more than one uNet in the text-to-image model.



Target (a) Generated (a) Target (b) Generated (b)

Figure 12: Images generated with the large model - saving the model based on the Perception Prioritized (P2) Loss and in the Fréchet Inception Distance (FID). **Captions:** (a) a baseball player swinging the bat during the game; (b) A bear that is standing partially on a road and in the grass. Complete results in `https://app.neptune.ai/leolellisr/dl-ia025`.

References

[1] Jorge Agnese, Jonathan Herrera, Haicheng Tao, and Xingquan Zhu. A survey and taxonomy of adversarial neural networks for text-to-image synthesis. *WIREs Data Mining Knowl Discov. Wiley Periodicals*, 2020. Available at `https://doi.org/10.1002/widm.1345`.

[2] Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. *Conference on Computer Vision and Pattern Recognition*, 2022. Available at `https://arxiv.org/pdf/2204.00227.pdf`.

[3] Gustavo H. de Rosa and João P. Papa. A survey on text generation using generative adversarial networks. *Pattern Recognition*, 119:108098, 2021. Available at `https://www.sciencedirect.com/science/article/pii/S0031320321002855`.

[4] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *OpenAI*, 2021. Available at `https://arxiv.org/pdf/2105.05233.pdf`.

[5] Stanislav Frolov, Tobias Hinz, Federico Raue, Jörn Hees, and Andreas Dengel. Adversarial text-to-image synthesis: A review. *Neural Networks*, 144:187–209, 2021. Available at `https://www.sciencedirect.com/science/article/pii/S0893608021002823`.

[6] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *31st Conference on Neural Information Processing Systems (NIPS)*, 2018. Available at `https://arxiv.org/pdf/1706.08500.pdf`.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020.

[8] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *35th Conference on Neural Information Processing Systems (NeurIPS 2021)*, 2021. Available at `https://openreview.net/pdf?id=2LdBqxc1Yv`.

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollar. Microsoft coco: Common objects in context. *International Conference on Learning Representations (ICLR)*, 2015.

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollar. Pythonapi - cocoapi, 2020. Available at `https://github.com/cocodataset/cocoapi/tree/master/PythonAPI`.

[11] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *International Conference on Learning Representations (ICLR)*, 2021. Available at `https://arxiv.org/abs/2102.09672`.

[12] Alex Nichol and Prafulla Dhariwal. Improved diffusion. open-ai., 2021. Available at `https://github.com/openai/improved-diffusion`.

[13] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *International Conference on Machine Learning*, 2022. Available at `https://arxiv.org/pdf/2112.10741.pdf`.

[14] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *Google Research*, 2022. Available at `https://arxiv.org/pdf/2205.11487.pdf`.

[15] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *International Conference on Learning Representations (ICLR)*, 2022. Available at `https://openreview.net/forum?id=TIdIXIpzhoI`.

[16] Noam Shazeer. t5 $\mathrm{v1}_1 base - google$.

[17] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. *arXiv e-prints*, page arXiv:1503.03585, March 2015.

[18] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *International Conference on Learning Representations (ICLR)*, 2021. Available at `https://arxiv.org/pdf/2010.02502.pdf`.

[19] Lilian Weng. What are diffusion models?, 2021. Available at `https://lilianweng.github.io/posts/2021-07-11-diffusion-models/`.