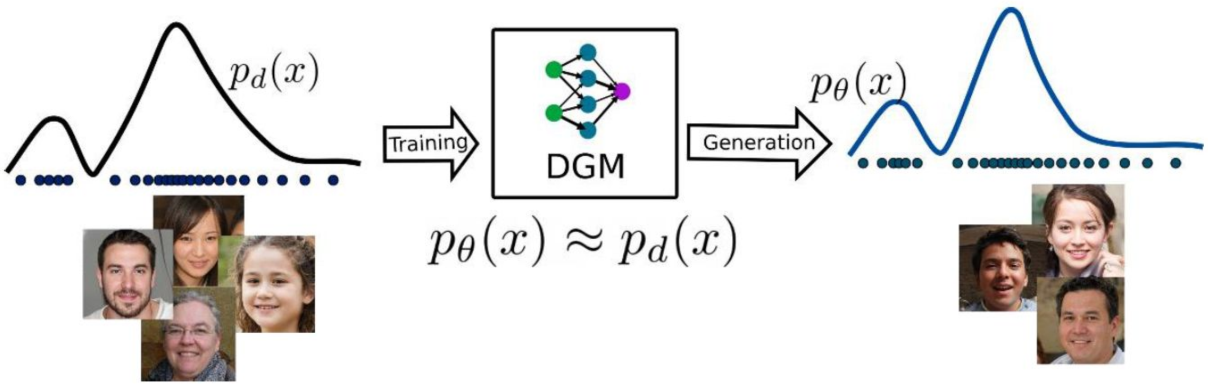
Diffusion Models

Geometry of Data, Fall 2023

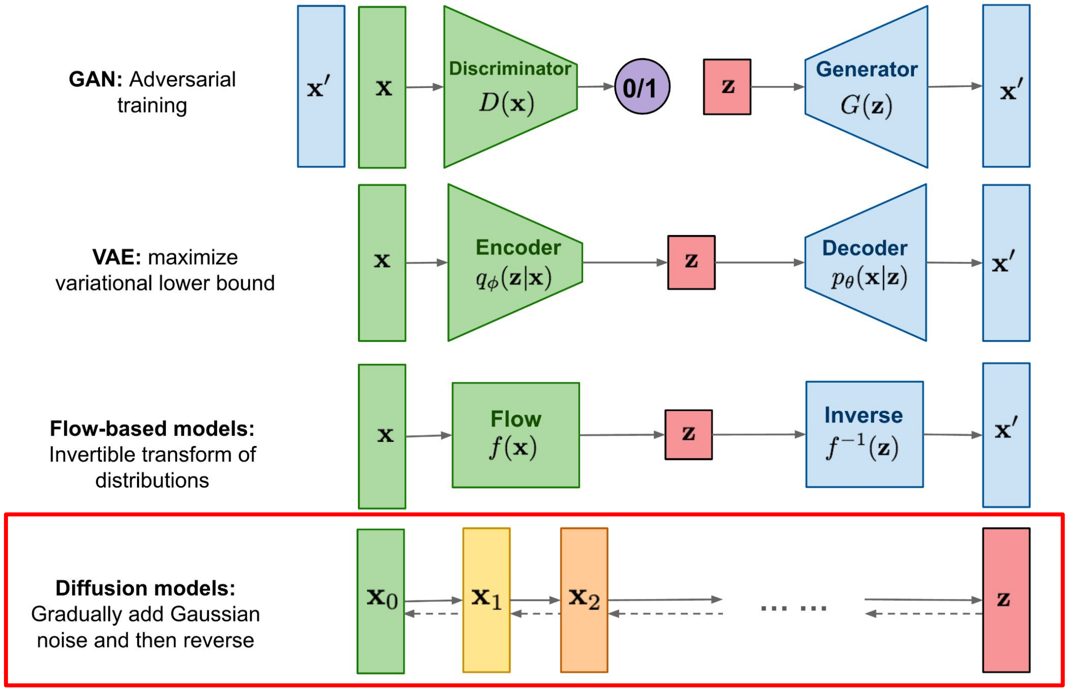
Deep generative modelling

1. Learn a neural network to approximate **p(x)**
2. Sample from learnt **p’(x)** to generate novel data



https://data-science-blog.com/blog/2022/02/19/deep-generative-modelling/

Types of Generative models



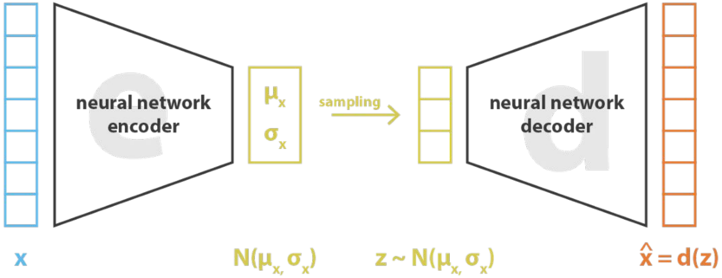
https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Background | Variational Autoencoders (VAE)

The VAE generative process is:

- first, a latent representation **z** is sampled from the prior distribution **p(z)**

- second, the data **x** is sampled from the conditional likelihood distribution **p(x|z)**



$$\text{loss} = \|x - \hat{x}\|^2 + \mathrm{KL}[\,N(\mu_x, \sigma_x), N(0, I)\,] = \|x - d(z)\|^2 + \mathrm{KL}[\,N(\mu_x, \sigma_x), N(0, I)\,]$$

*Note that, the KL-divergence between two gaussians **p & q**, is defined as follows:*

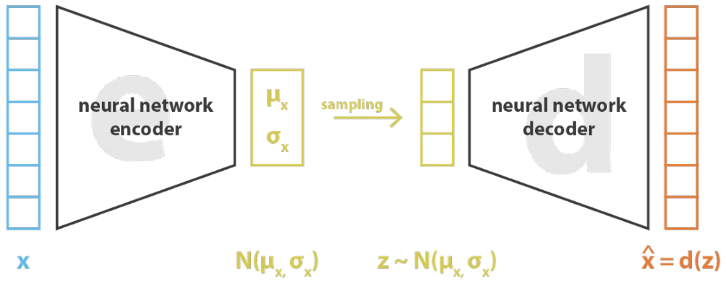
$$D_{KL}(p||q) = \frac{1}{2}\left[\log\frac{|\Sigma_q|}{|\Sigma_p|} - k + (\boldsymbol{\mu_p} - \boldsymbol{\mu_q})^T\Sigma_q^{-1}(\boldsymbol{\mu_p} - \boldsymbol{\mu_q}) + tr\left\{\Sigma_q^{-1}\Sigma_p\right\}\right]$$

Background | Variational Autoencoders (VAE)

The "**probabilistic decoder**" is defined by **p(x|z)**, that describes the distribution of the decoded variable given the encoded one

The "**probabilistic encoder**" is defined by **p(z|x)**, that describes the distribution of the encoded variable given the decoded one

We use Bayes' theorem to get:



$$\text{loss} = \|x - \hat{x}\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)] = \|x - d(z)\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)]$$

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} = \frac{p(x|z)p(z)}{\int p(x|u)p(u)du}$$

Background | Variational Autoencoders (VAE)

In theory, we know **p(z)** and **p(x|z)**, we can use the Bayes theorem to compute **p(z|x)**

However, this kind of computation is often **intractable** due to the **integral in the denominator**

Here we are going to approximate **p(z|x)** by a Gaussian distribution **q_x(z)** whose mean and covariance are defined by two functions, **g** and **h**, of the parameter **x**.

$$q_x(z) \equiv \mathcal{N}(g(x), h(x)) \qquad g \in G \qquad h \in H$$

$$\begin{aligned}
(g^*, h^*) &= \underset{(g,h)\in G\times H}{\arg\min}\; KL(q_x(z), p(z|x)) \\
&= \underset{(g,h)\in G\times H}{\arg\min} \left(\mathbb{E}_{z\sim q_x}(\log q_x(z)) - \mathbb{E}_{z\sim q_x}\left(\log \frac{p(x|z)p(z)}{p(x)}\right)\right) \\
&= \underset{(g,h)\in G\times H}{\arg\min} \left(\mathbb{E}_{z\sim q_x}(\log q_x(z)) - \mathbb{E}_{z\sim q_x}(\log p(z)) - \mathbb{E}_{z\sim q_x}(\log p(x|z)) + \mathbb{E}_{z\sim q_x}(\log p(x))\right) \\
&= \underset{(g,h)\in G\times H}{\arg\max} \left(\mathbb{E}_{z\sim q_x}(\log p(x|z)) - KL(q_x(z), p(z))\right) \\
&= \underset{(g,h)\in G\times H}{\arg\max} \left(\mathbb{E}_{z\sim q_x}\left(-\frac{||x - f(z)||^2}{2c}\right) - KL(q_x(z), p(z))\right)
\end{aligned}$$

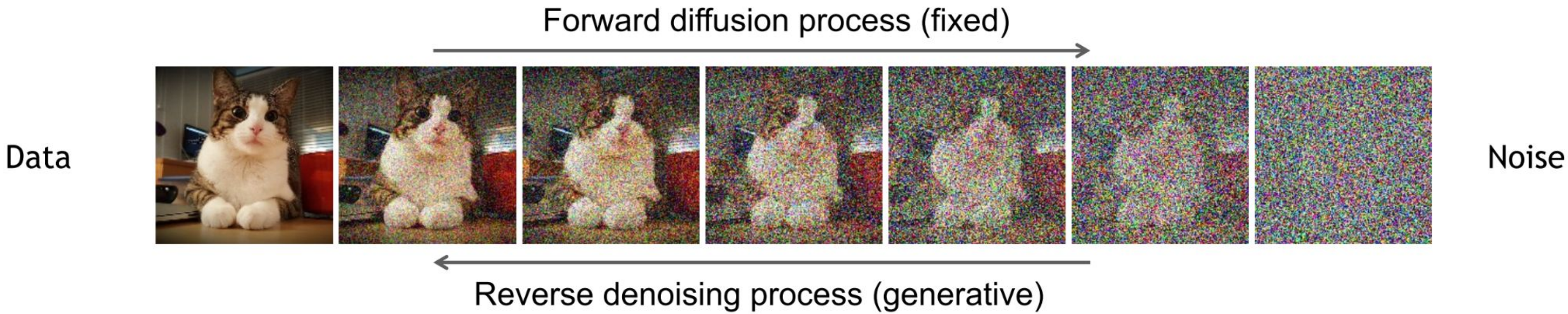
Overall,

$$(f^*, g^*, h^*) = \underset{(f,g,h)\in F\times G\times H}{\arg\max} \left(\mathbb{E}_{z\sim q_x}\left(-\frac{||x - f(z)||^2}{2c}\right) - KL(q_x(z), p(z))\right)$$

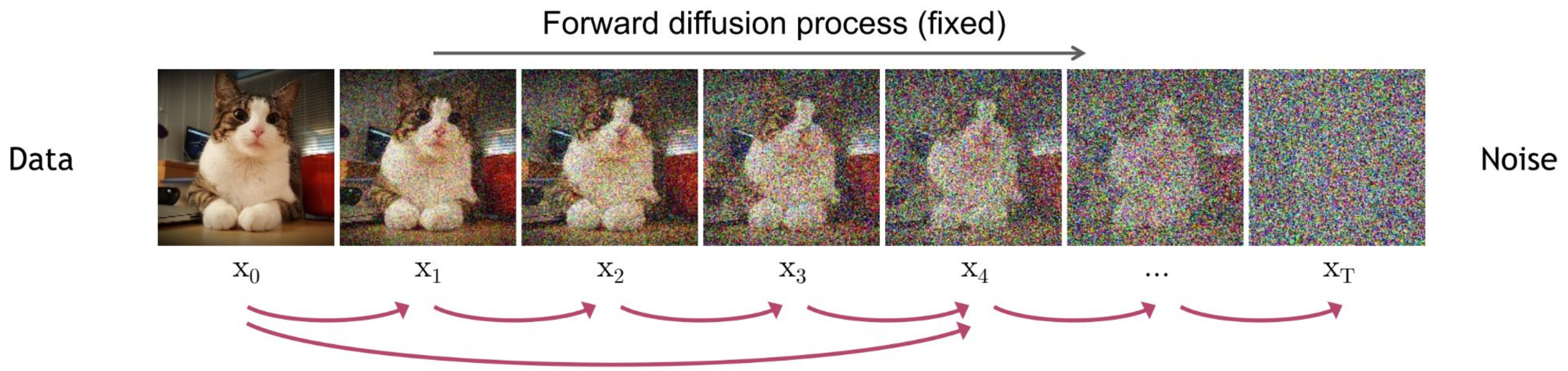
Denoising Diffusion Probabilistic Models (DDPM)

Forward diffusion: Markov chain of diffusion steps to slowly add gaussian noise to data

Reverse diffusion: A model is trained to generate data from noise by iterative denoising



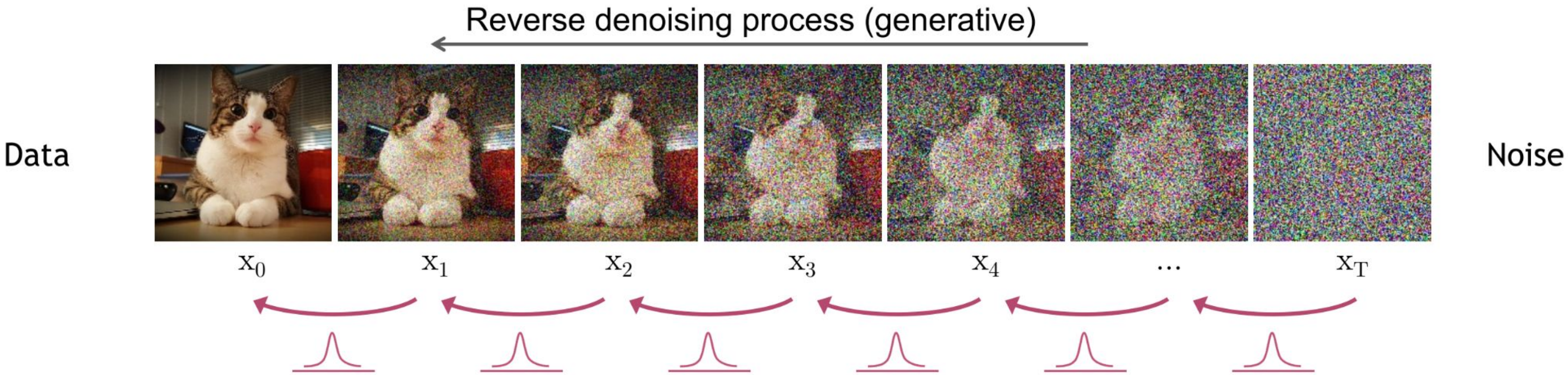
DDPM | Forward diffusion



We add a small amount of gaussian noise to a sample $\mathbf{x}_0$ in **T** timesteps to produces noised samples, $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$. The steps are controlled by the noise schedule as follows:

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

DDPM | Reverse Diffusion



We learn a neural network model **(p_θ)** to approximate these conditional probabilities **$q(x_{(t-1)} \mid x_t)$** in order to run the reverse diffusion process as follows:

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

Training the denoising model

For training, we can form variational upper bound that is commonly used for training variational autoencoders,

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] =: L$$

which simplifies to,

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

and where **$q(x_{(t-1)} | x_t, x_0)$** is a tractable posterior:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1};\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0),\tilde{\beta}_t\mathbf{I}), \quad \text{where } \tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{1-\beta_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \text{ and } \tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

Parameterization of the diffusion model

The model is primarily trained on the term $\mathbf{L_{(t-1)}}$ above, which is a KL-divergence of two normal distributions, $\mathbf{q(x_{(t-1)} \mid x_t, x_0)}$ and $\mathbf{p_\theta(x_{(t-1)} \mid x_t)}$ and has a simple form:

$$L_{t-1} = D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}||\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t)||^2\right] + C$$

In Ho et al. NeurIPS 2020, above is reparameterized to be a noise-prediction network instead of a mean-prediction network,

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0\sim q(\mathbf{x}_0),\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\underbrace{\frac{\beta_t^2}{2\sigma_t^2(1-\beta_t)(1-\bar{\alpha}_t)}}_{\lambda_t}||\epsilon - \epsilon_\theta(\underbrace{\sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon}_{\mathbf{x}_t}, t)||^2\right] + C$$

Parameterization of the diffusion model

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \left[\underbrace{\frac{\beta_t^2}{2\sigma_t^2 (1 - \beta_t)(1 - \bar{\alpha}_t)}}_{\lambda_t} ||\epsilon - \epsilon_\theta(\underbrace{\sqrt{\bar{\alpha}_t}\, \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon}_{\mathbf{x}_t}, t)||^2 \right] + C$$

Note that λ_t above is a just a time-dependent reweighting parameter.

It is observed that for training the model, it is **helpful** if we set **$\lambda_t = 1$.**

Making the objective even simpler,

$$L_{\text{simple}} = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t \sim \mathcal{U}(1, T)} \left[||\epsilon - \epsilon_\theta(\underbrace{\sqrt{\bar{\alpha}_t}\, \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon}_{\mathbf{x}_t}, t)||^2 \right]$$

Overall algorithm (like we see it!)

Algorithm 1 Training

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

Algorithm 2 Sampling

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

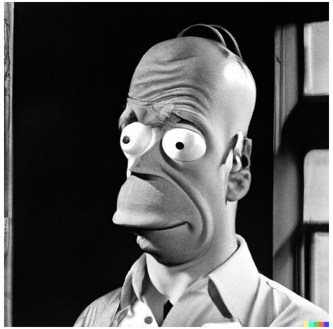
Conditional diffusion models

In conditional diffusion models, an additional input, **y** (eg. a class label or a text sequence) is available and we try to model the conditional distribution **p(x | y)** instead.

This allows us to generate data given the conditioning signal.

Some examples generated from Google's Imagen [1], and OpenAI's Dalle-2 [2] on the right.

A medieval painting of the wifi not working



A still of Homer Simpson in Psycho (1960)

An Alpaca is smiling and underwater in the pool

A tulip pushing a baby carriage

[1] Saharia, Chitwan, et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." *arXiv preprint arXiv:2205.11487* (2022).
[2] Ramesh, Aditya, et al. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125* (2022).

Conditional diffusion models

In practice, the denoising model **$p(x_{t-1} | x_t, y)$** is also conditioned on '**y**' in addition to the image from the previous timestep, **x_t**

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

Variational upper bound: $L_\theta(\mathbf{x}_0|\mathbf{c}) = \mathbb{E}_q\left[L_T(\mathbf{x}_0) + \sum_{t>1} D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{c})) - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1, \mathbf{c})\right].$

Practical considerations

- **Scalar conditioning:** encode scalar as a vector embedding, simple spatial addition or adaptive group normalization layers.
- **Image conditioning:** channel-wise concatenation of the conditional image.
- **Text conditioning:** single vector embedding – spatial addition or adaptive group norm / a seq of vector embeddings – cross-attention.

Score-model based guidance

Using the gradient of an independently pre-trained score model as guidance
Given a conditional model **p(x$_t$ | y)**, we use gradients from an extra score model **p(y | x$_t$)** during sampling.

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s.

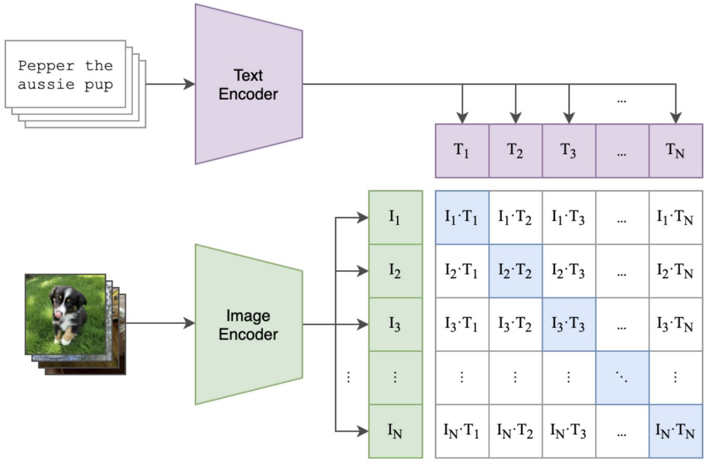
Input: class label y, gradient scale s model   Classifier gradient
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
    $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
    $x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma\, \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$
end for
return x_0

CLIP guidance

Given an image **x** and a prompt **y**, a CLIP model computes the alignment **cos_sim(x, y)** which indicates how similar the image and the prompt are.

To use this signal for guidance, we assume that the CLIP similarity score is a good estimation of the function **p(y|x)**

The gradient of this score wrt the noised image, $\mathbf{x}_t$ at timestep **t** is used as the guidance gradient



*Note that this requires the CLIP model to compute score for **noised-images** at intermediate timesteps, hence a noised-CLIP model is trained for guidance*

Classifier-free guidance

Given both a conditional and an unconditional diffusion model, we can design an “implicit” classifier as follows:

$$p(\mathbf{c}|\mathbf{x}_t) \propto p(\mathbf{x}_t|\mathbf{c})/p(\mathbf{x}_t)$$

Conditional diffusion model   Unconditional diffusion model

In practice, **p(x|c)** and **p(x)** are trained together by randomly dropping the conditioning signal with a certain probability during training.

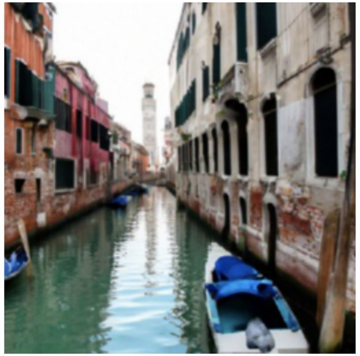
Using above, the score-gradient becomes:

$$\begin{aligned}\nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega \log p(\mathbf{c}|\mathbf{x}_t)] &= \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega(\log p(\mathbf{x}_t|\mathbf{c}) - \log p(\mathbf{x}_t))] \\ &= \nabla_{\mathbf{x}_t}[(1+\omega)\log p(\mathbf{x}_t|\mathbf{c}) - \omega \log p(\mathbf{x}_t)]\end{aligned}$$

GLIDE | OpenAI

- A 64x64 base diffusion model
- A 64 -> 256 conditional super-resolution model
- Evaluates both classifier-free and **CLIP guidance**

CLIP guidance: Use the CLIP alignment score **p(x, y)** as a estimation of **p(y | x)**



“a boat in the canals of venice”

“a painting of a fox in the style of starry night”

“a crayon drawing of a space elevator”

“a futuristic city in synthwave style”

Nichol, Alex, et al. "Glide: Towards photorealistic image generation and editing with text-guided diffusion models." *arXiv preprint arXiv:2112.10741* (2021).
Radford, Alec, et al. "Learning transferable visual models from natural language supervision." *International Conference on Machine Learning*. PMLR, 2021.

DALL.E 2 | OpenAI

- 1kx1k text-conditioned image generation
- Uses a **prior** to produce CLIP embeddings conditioned on the text-caption
- Uses a **decoder** to produce images conditioned on the CLIP embeddings

a shiba inu wearing a beret and black turtleneck

a close up of a handpalm with leaves growing from it

panda mad scientist mixing sparkling chemicals, artstation

a corgi’s head depicted as an explosion of a nebula

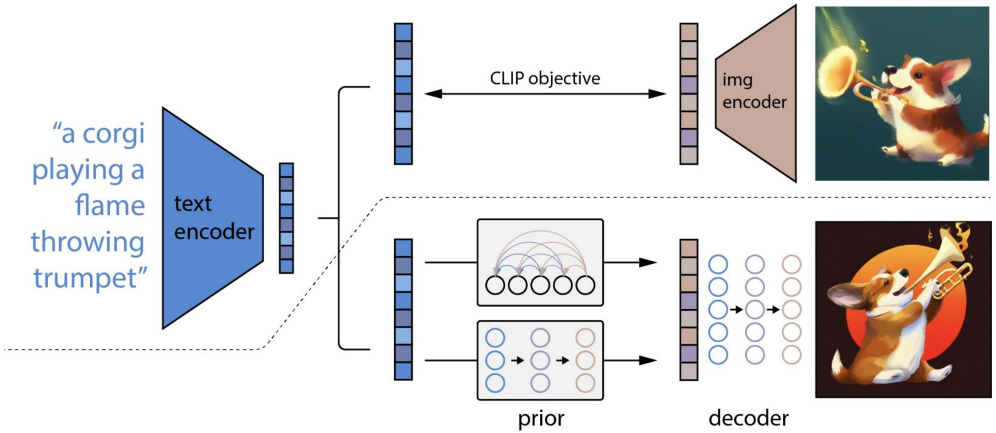
Ramesh, Aditya, et al. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125* (2022).

DALL.E 2 | Open AI

Conditioning on CLIP-embeddings

- Helps capture multimodal representations

- The bi-partite latent enables several text-controlled image manipulation tasks



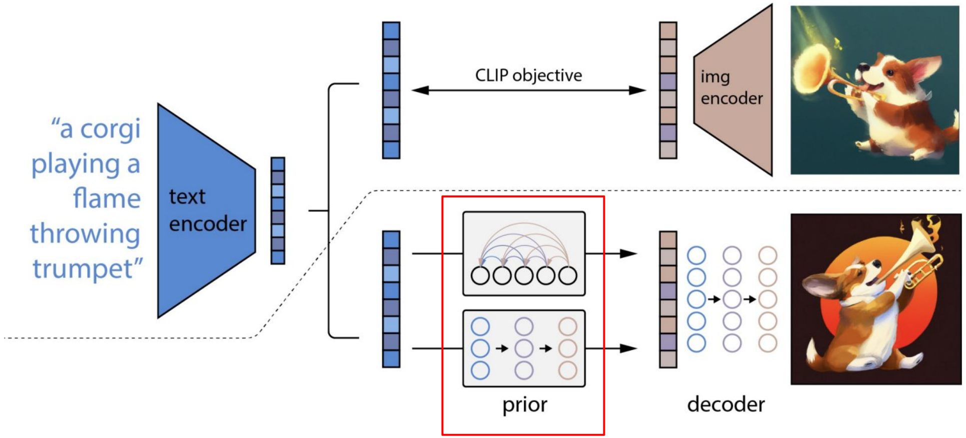
Ramesh, Aditya, et al. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125* (2022).

DALL.E 2 | Open AI

Proposes 2 types of priors:

1. **Autoregressive prior**
   Quantize image embeds into a sequence of discrete codes and predict them autoregressively

2. **Diffusion prior**
   Model continuous image embeddings by diffusion models conditioned on caption



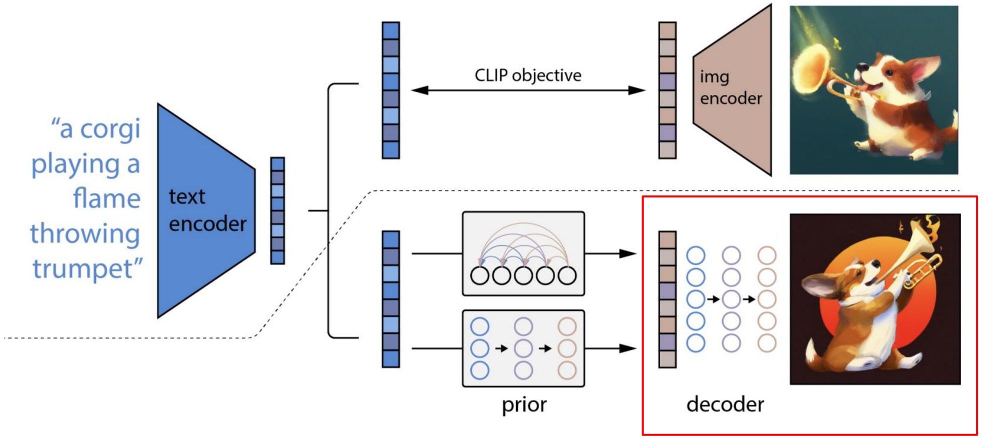
Ramesh, Aditya, et al. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125* (2022).

DALL.E 2 | Open AI

Decoder: produces images conditioned on CLIP image embeddings (and text caption)

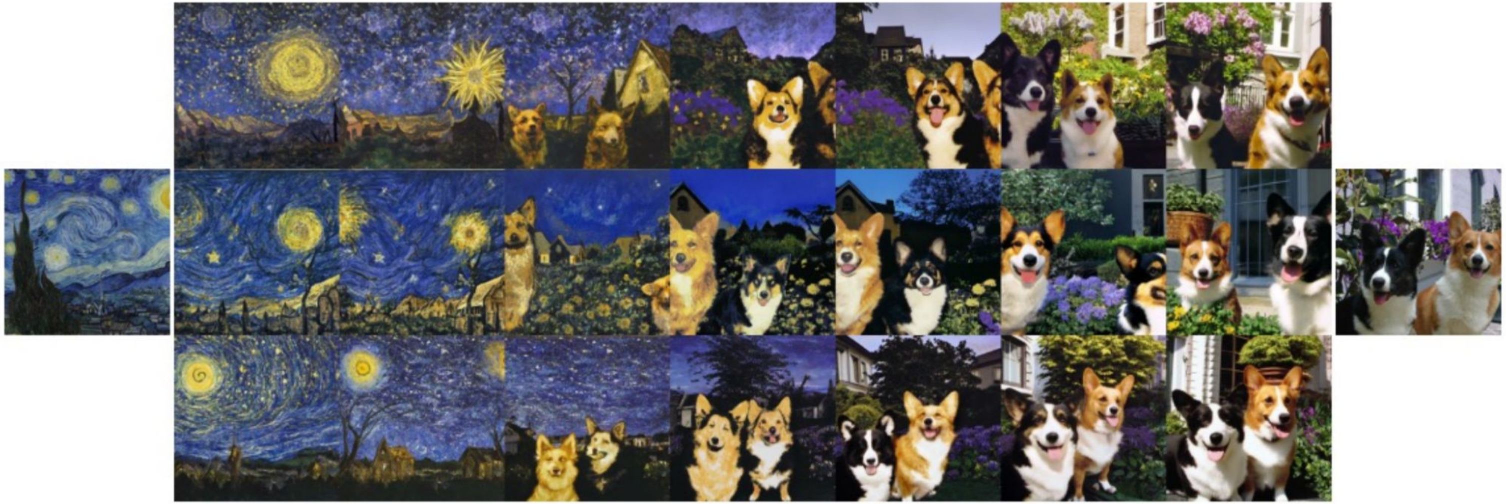
The model is trained as cascaded diffusion models 64->256->1024

It is observed that classifier-free guidance works better for sample quality here.



Ramesh, Aditya, et al. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125* (2022).

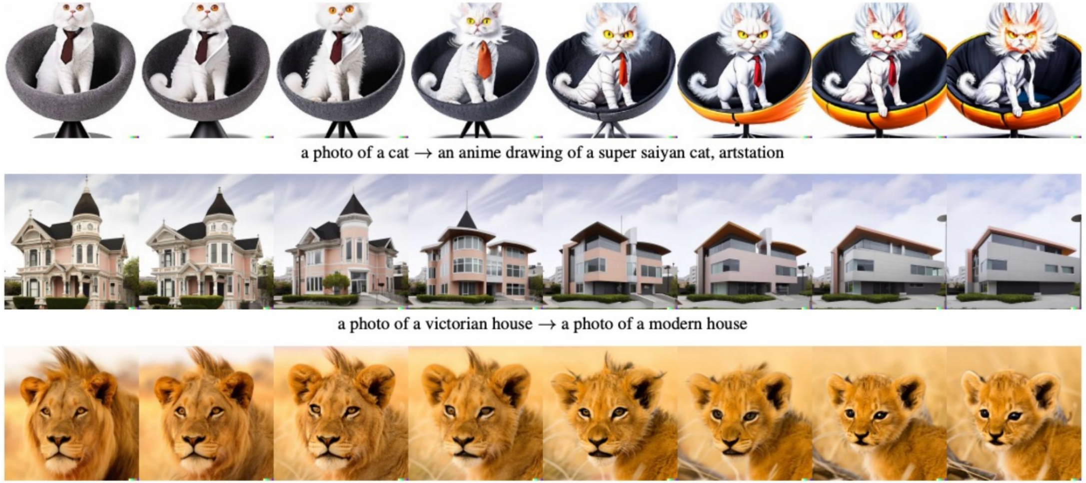
DALL.E 2 | Open AI



Interpolate CLIP embeddings to generate different interpolation trajectories

Ramesh, Aditya, et al. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125* (2022).

DALL.E 2 | Open AI



a photo of a cat → an anime drawing of a super saiyan cat, artstation

a photo of a victorian house → a photo of a modern house

a photo of an adult lion → a photo of lion cub

Change the image CLIP embedding towards the difference of the text CLIP embeddings of two prompts. Note that decoder latent is kept as a constant.

Ramesh, Aditya, et al. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125* (2022).

Imagen | Google Research

- Generates 1kx1k images
- Exceptional photo-realism
- Extremely simple parameterization
- SOTA on quantitative and qualitative benchmarks
- Proposes a new qualitative benchmark (drawbench)

Teddy bears swimming at the Olympics 400m Butter-fly event.

A cute corgi lives in a house made out of sushi.

A brain riding a rocketship heading towards the moon.

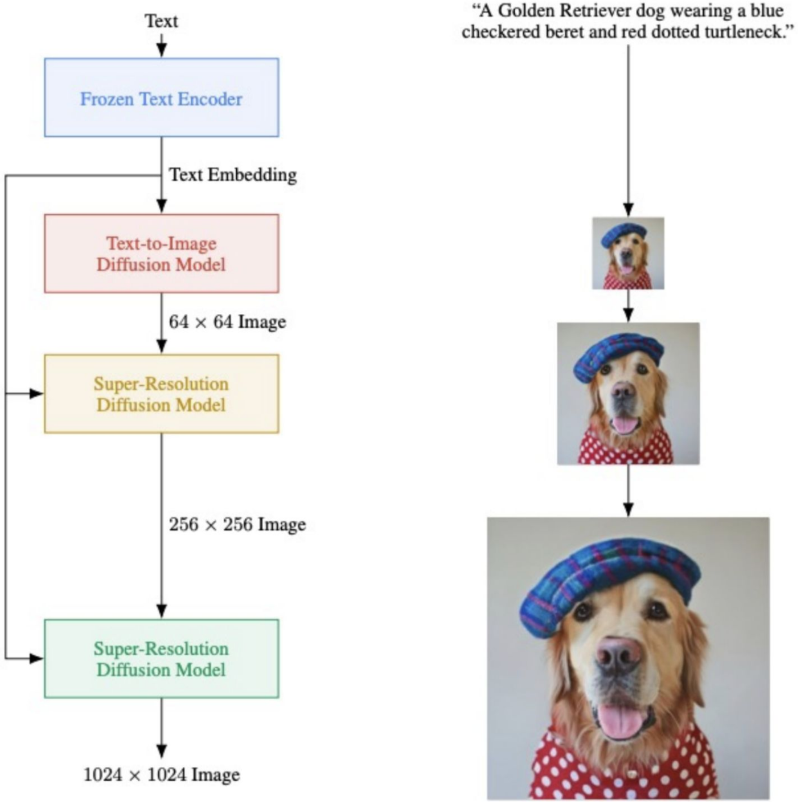
A dragon fruit wearing karate belt in the snow.

Saharia, Chitwan, et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." *arXiv preprint arXiv:2205.11487* (2022).

Imagen | Google Research

Model details

- Cascaded diffusion models
  64 -> 256 -> 1024
- Classifier-free guidance and dynamic thresholding
- Frozen large pretrained language models as text encoders (T5-XXL)

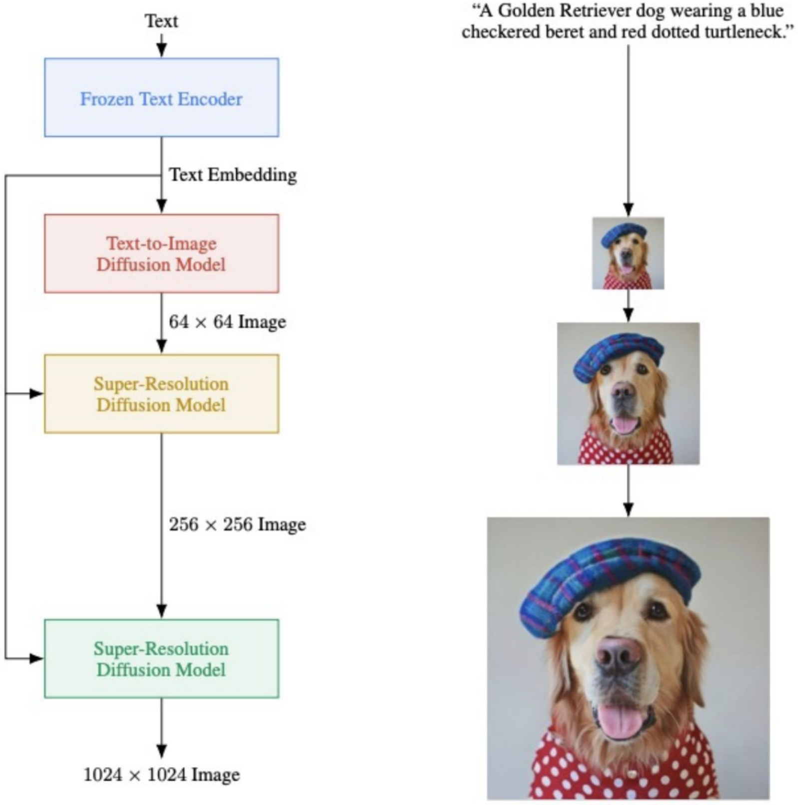


Saharia, Chitwan, et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." *arXiv preprint arXiv:2205.11487* (2022).

Imagen | Google Research

Main discoveries

- Better text-conditioning signal is important, i.e. large frozen text-encoders are used, eg. T5-XXL
- Stronger classifier-free guidance leads to better text-alignment but worse image quality



Saharia, Chitwan, et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." *arXiv preprint arXiv:2205.11487* (2022).

Imagen | Google Research

The paper also proposes a new benchmark called the “drawbench”

- Collection of 200 prompts that test semantic understanding and image diversity.

Saharia, Chitwan, et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." *arXiv preprint arXiv:2205.11487* (2022).

Imagen | Google Research

Model	FID-30K	Zero-shot FID-30K
AttnGAN [76]	35.49	
DM-GAN [83]	32.64	
DF-GAN [69]	21.42	
DM-GAN + CL [78]	20.79	
XMC-GAN [81]	9.33	
LAFITE [82]	8.12	
Make-A-Scene [22]	7.55	
DALL-E [53]		17.89
LAFITE [82]		26.94
GLIDE [41]		12.24
DALL-E 2 [54]		10.39
Imagen (Our Work)		**7.27**

Imagen achieves SOTA using auto-evaluation scores on COCO dataset

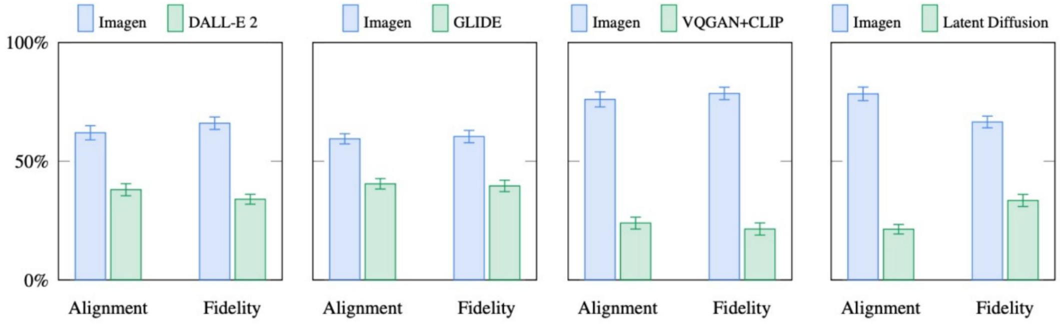


Imagen is preferred over recent work by human raters in sample quality & image-text alignment on DrawBench

Saharia, Chitwan, et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." *arXiv preprint arXiv:2205.11487* (2022).

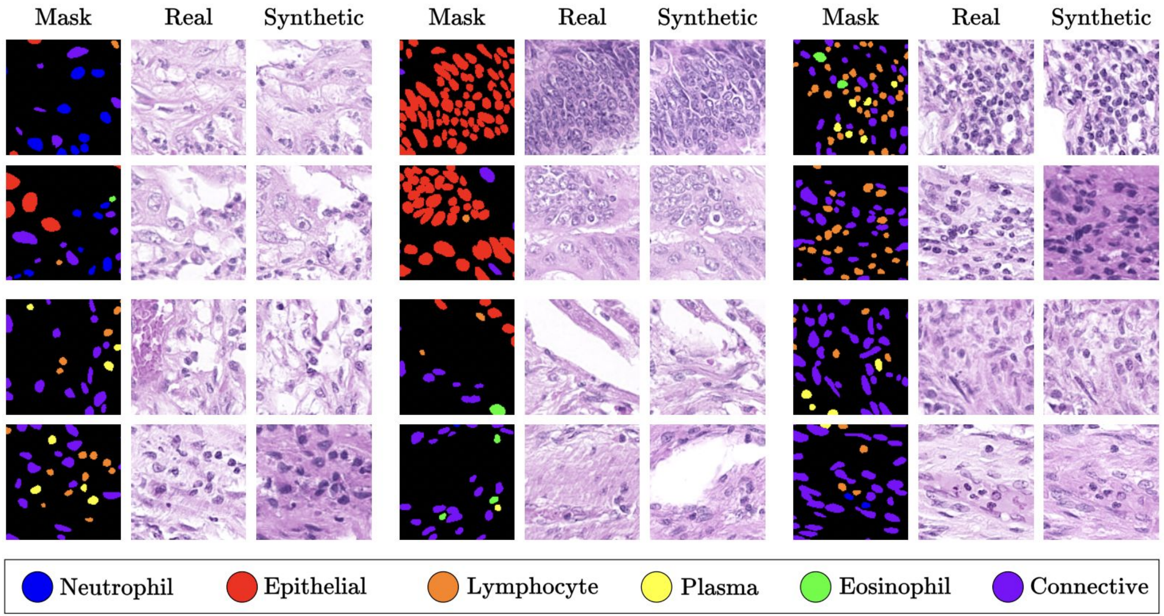
NASDM: Nuclei-Aware Semantic Histopathology Image Generation Using Diffusion Models

Accepted at MICCAI 2023

Shrivastava, Aman, and P. Thomas Fletcher. "NASDM: Nuclei-Aware Semantic Histopathology Image Generation Using Diffusion Models." *arXiv preprint arXiv:2303.11477* (2023).

Overview

The diffusion model can **generate realistic histopathological patches conditioned on the semantic locations of six different types of nuclei.**



Method

We condition the diffusion model on the **7-channel semantic mask** comprising of 6 individual nuclei semantics and an additional edge mask highlighting the nuclei instances overall

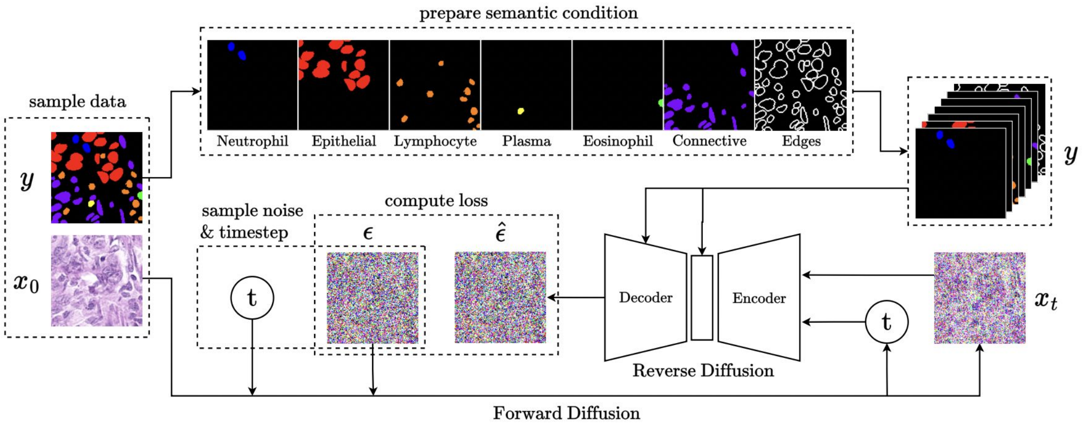


Fig. 1. NASDM training framework: Given a real image x_0 and semantic mask y, we construct the conditioning signal by expanding the mask and adding an instance edge map. We sample timestep t and noise ϵ to perform forward diffusion and generate the noised input x_t. The corrupted image x_t, timestep t, and semantic condition y are then fed into the denoising model which predicts $\hat{\epsilon}$ as the amount of noise added to the model. Original noise ϵ and prediction $\hat{\epsilon}$ are used to compute the loss in (4).

Results

We train the model on the **lizard dataset at 20× magnification split into 128 × 128 pixels patches**

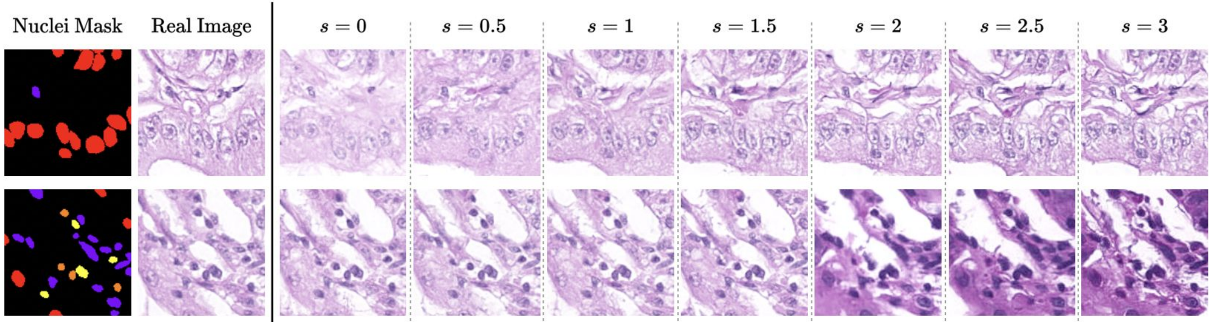
For training, we extract a total of 54,735 patches for training and 4,991 patches as a held-out set

Table 1. Quantitative Assesment: We report the performance of our method using standard generative metrics Fréchet Inception Distance (FID) metrics and Inception Score (IS) with the metrics reported in existing works. (-) denotes that the corresponding information was not reported in the original work.

Method	**Tissue type**	**Conditioning**	**FID(↓)**	**IS(↑)**
BigGAN [2]	bladder	none	158.4	-
AttributeGAN [32]	bladder	attributes	53.6	-
ProGAN [11]	glioma	morphology	53.8	1.7
Morph-Diffusion [18]	glioma	morphology	20.1	2.1
NASDM (Ours)	colon	semantic mask	**15.7**	**2.7**

Key insights

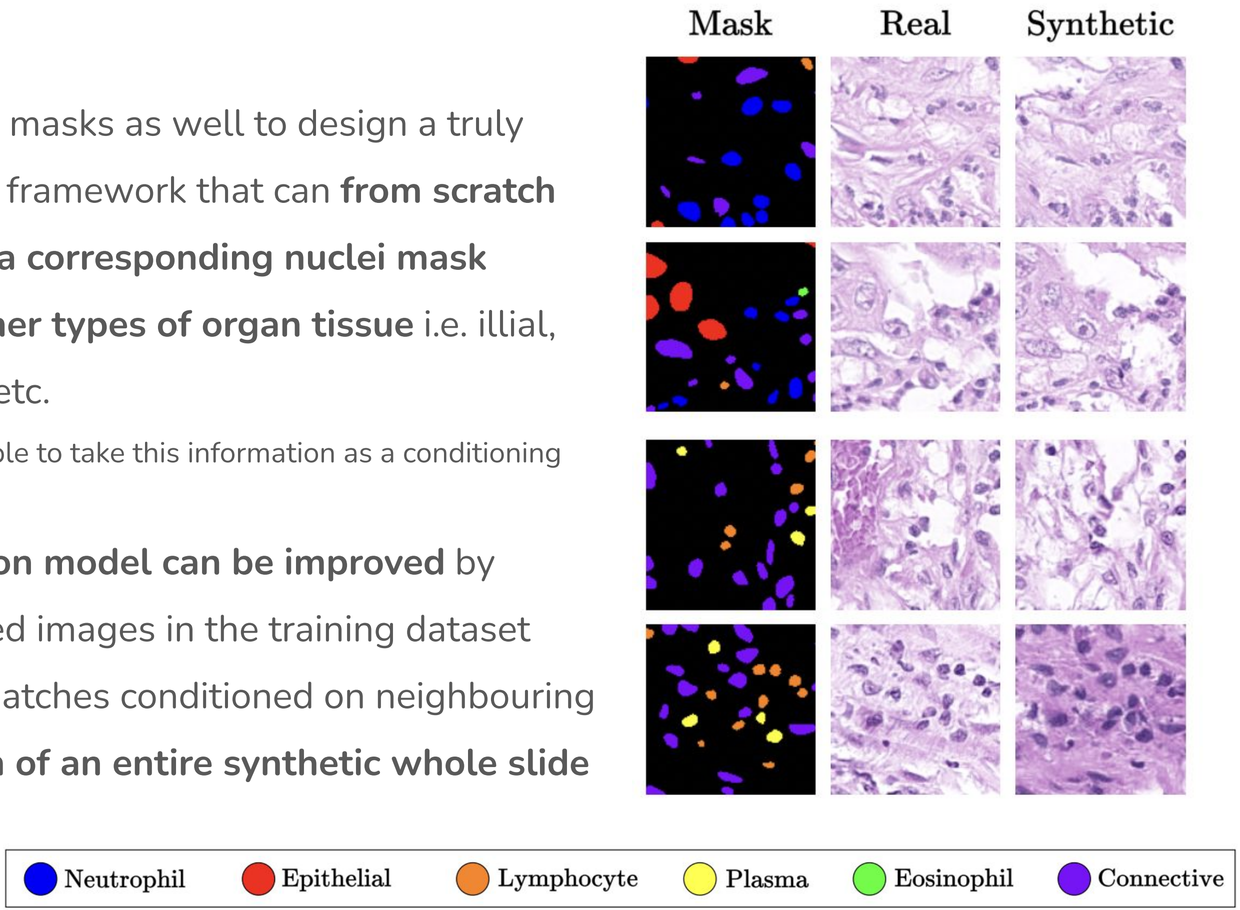
1. Diffusion models are extremely powerful and **can generate hyper-realistic images** which are hard to distinguish from real ones
2. The model already **achieves state-of-the-art performance quantitatively**
3. The model **can effectively processes the semantic conditioning information and generate images consistent with the mask**



	$s = 0$	$s = 0.5$	$s = 1$	$s = 1.5$	$s = 2$	$s = 2.5$	$s = 3$
FID	17.12	16.98	17.01	**15.70**	16.74	19.45	22.48
Inception Score	2.47	2.48	2.48	2.68	2.70	**2.73**	2.72

Future directions

1. Train a model to generate the masks as well to design a truly end-to-end tissue generation framework that can **from scratch generate a tissue patch and a corresponding nuclei mask**
2. **Extend the generation to other types of organ tissue** i.e. illial, glioma, breast, bladder, liver etc.
   a. Ideally the model should be able to take this information as a conditioning signal
3. **Study if a nuclei segmentation model can be improved** by addition of synthetic annotated images in the training dataset
4. Design a model to generate patches conditioned on neighbouring patches to **enable generation of an entire synthetic whole slide image**



Questions?