Score and Diffusion Models

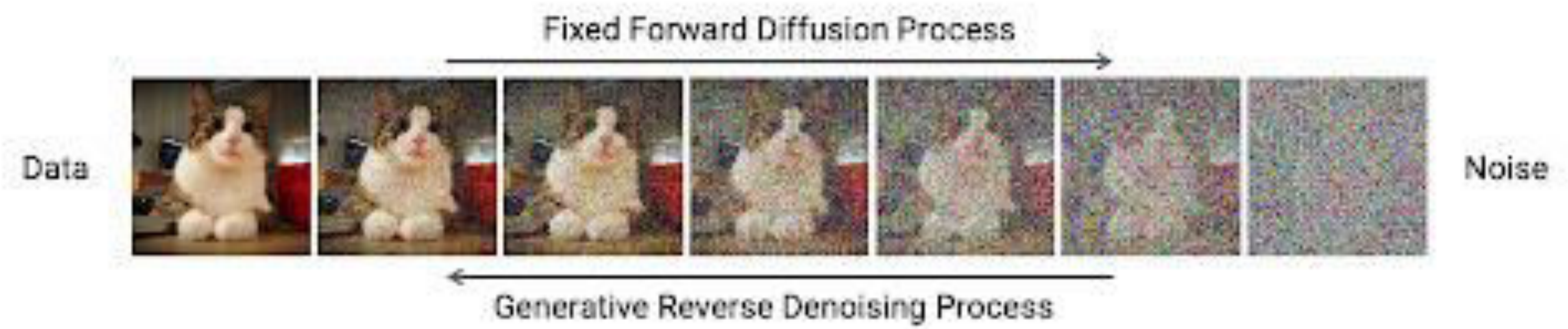
Week 5

Variational Models

- Last week: variational models
- Typically: use strong priors over the latent variable distribution:
  - Low rank
  - Simple parametric form
- Limitations:
  - Blury results - high-frequency (fine) details are often not low-rank
  - The parametric choice of p(z) might be too restrictive

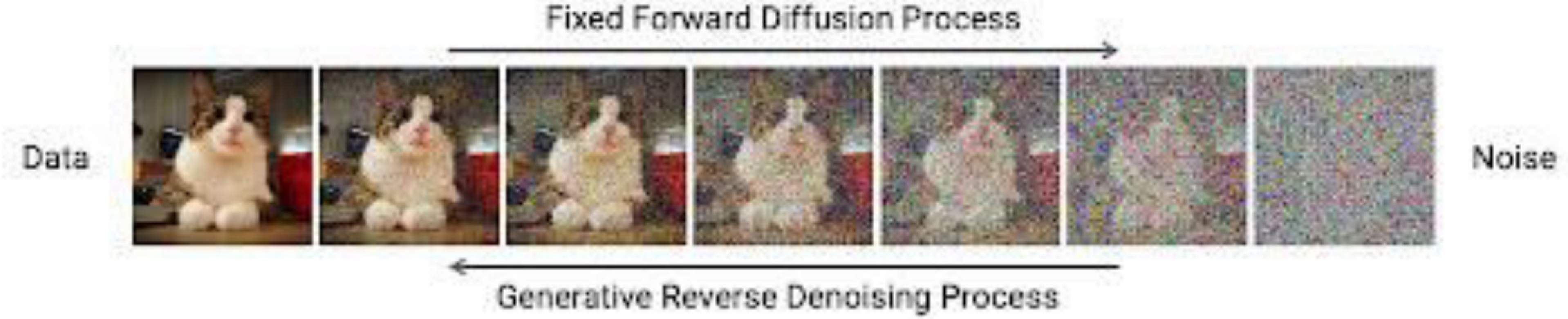
Diffusion Processes

- Variational models: link between p(x) and latent distribution p(z)
  - No known correspondences
- Diffusion models: add noise to each x until it becomes noise
  - Noise added through time
  - Small amount of noise at every time step



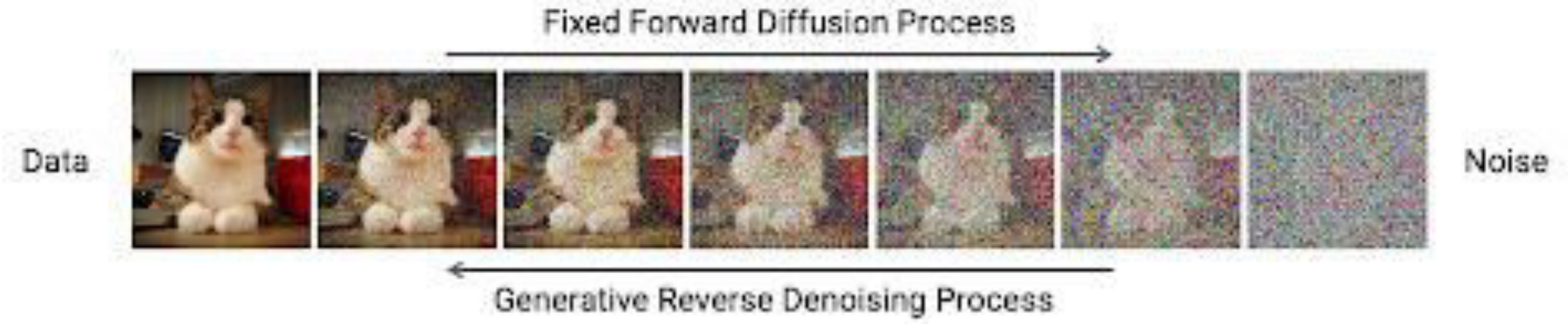
Reverse Diffusion Process

- Goal is to reverse the diffusion process
- Map the distribution of Gaussian noise p(z) to p(x)
- Diffrently from VI, can do it in many, very small steps
- At each stage, we only invert a tiny temporal step – not so hard
- Inversion = denoising a small amount



Life of a Sample

- Let us start from sample x at time 0
- At each time step t:
  - Shrink to origin by factor x = x - x * f(t) (f(t) > 0)
  - Add random Gaussian noise x = x + g(t) W (W = N(0, I))
- At time 1, sample will lose original information, contain pure noise



Types of Processes

- Variance preserving (VP):

$$f(t) = \sqrt{(}1 - g^2(t))$$

  - Variance is unchanged through time, more is allocated to noise

- Variance exploding (VE):

$$f(t) = 0$$

  - Variance of image is constant, while increasing amounts of noise added

Stochastic Differential Equations (SDEs)

- SDEs: mathematical way to describe continuous stochastic processes
- Big topic, we just introduce what we need for diffusion
- Let us consider our discrete time process:
  - $x(t+dt) = x(t) - x(t) * f(t)\, dt + N(0, g(t)I)$
- Taking step size to 0, we obtain the following SDE:
  - $dx = - f(t)\, x\, dt + g(t)\, dW_t$
  - dW_t is an infinitesimal amount of Gaussian noise added at time t
  - $E[dW_t] = 0$

Solution of the SDE

- SDE: changes to the sample at each time step
- Solution of SDE: the distribution of the sample at each t
- Intuitively: scaled version of the initial sample, added Gaussian noise
- Will not derive it:

Distribution of sample x0 at time t

$$\left[\mathcal{N}\left(\boldsymbol{x};\ s(t)\,\boldsymbol{x}_0,\ s(t)^2\,\sigma(t)^2\,\mathbf{I}\right)\right]$$

Scaling factor for mean x0 at time t

$$\exp\left(\int_0^t f(\xi)\,\mathrm{d}\xi\right) \quad = \quad s(t)$$

Variance of total noise at time t

$$\sqrt{\int_0^t \frac{g(\xi)^2}{s(\xi)^2}\,\mathrm{d}\xi} \quad = \quad \sigma(t)$$

Solution of SDE in VE Process

- Reminder: in VE process, scale is fixed S(t) =1 (f(t) = 0)
- Inspecting previous equations:

$$dx = g(t)dW$$

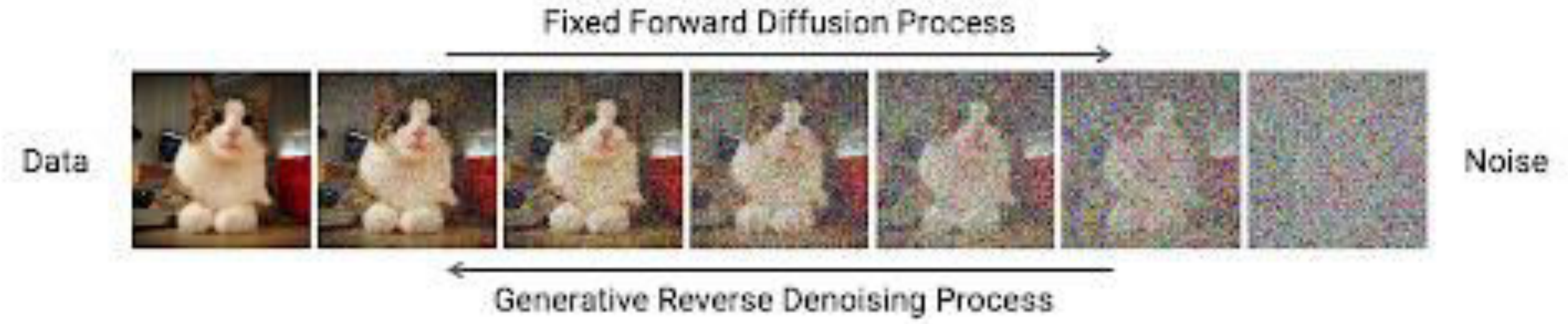
$$\mathcal{N}(x; x_0, \sigma^2(t)I))$$

$$\sigma^2(t) = \int_{\eta=0}^{t} g^2(\eta)d\eta$$

- The variance at time t, is the integral all the variances of the noises

Inverting the Process

- So far: a noising process transforming p(x) at t=0, to p(z) at t=1
- Objective: inverse process taking p(z) at t=1 to p(x) at t=0
- Why not just flip the sign of the SDE?



Inverting SDE in VE Process

- Invering the VE SDE yields:

$$dx = -g(t)dW$$

- Gaussian noise is actually symmetric to sign, equation is unchanged!
- Clearly not the way to invert the process…

The Fokker-Planck Equation

- The SDE described how each sample evolves in time
- How does the entire probability distribution p(x,t) evolve in time?
- Solution provided by the FP equation:

$$\frac{\partial p(x,t)}{\partial t} = -\nabla_x[f(x,t)p(x,t)] + \frac{1}{2}g^2(t)\Delta_x[p(x,t)]$$

- This is a **deterministic** partial differential equation
- To invert it, we simply flip the sign

Inverting Fokker-Planck Equation

- How do we achieve inversion of the sign of the FPE?

$$\frac{\partial p(x,t)}{\partial t} = -\nabla_x[f(x,t)p(x,t)] + \frac{1}{2}g^2(t)\Delta_x[p(x,t)]$$

- Idea: construct a deterministic process resulting in same FPE!
- Named: probability flow ODE

$$dx = [f(x,t) - \frac{1}{2}g^2(t)\nabla_x \log p(x,t)]dt$$

Inverting The Noise Process

- Given the probability flow ODE, reverse the process by sign inversion

$$dx = -[f(x,t) - \frac{1}{2}g^2(t)\nabla_x \log p(x,t)]dt$$

- Can also construct a stochastic process with the same marginals

$$dx = -[f(x,t) - \frac{1}{2}(1+\lambda^2)g^2(t)\nabla_x \log p(x,t)]dt + \lambda g(t)dW$$

- Deterministic process called DDIM
- Stochastic process classed DDPM

The Score Function

- Remaining challenge - how to compute?

$$\nabla_x \log p(x, t)$$

- This quantity is called the **score function**

Score Does not Require Normalized PDFs

- What happens is we do not know p(x), but up to normalization
- For example, assume we know E(x) but not Z

$$p(x) = \frac{e^{E(x)}}{Z}$$

- Score does not depend on Z!

Simple Example

- Assume at t=0, we have a single point $p(x) = \delta(x)$
- We add Gaussian noise through time s.t. $p(x,t) = \mathcal{N}(0, \sigma^2(t)I)$
- The Gaussian distribution has

$$p(x,t) \propto e^{E(x,t)} \qquad E(x,t) = -\frac{1}{2}\left(\frac{x}{\sigma(t)}\right)^2$$

- The score is therefore:

$$s(x,t) = -\frac{x}{\sigma^2(t)}$$

Denoising

- Task of denoising: removing noise from sample

$$D(x + n) \rightarrow x$$

- Where *D* is a denoising function and *n* is noise

Expression for Denoising

- Assume we have a noise sample f
- The data consists of x samples x1,x2…xM
- At time t, each x may take the values: N(xi, s(t)I)
- The probability that f is in fact generated by xi is

$$\frac{p(y|x_i)p(x_i)}{\sum_l p(y|x_l)p(x_l)} = \frac{\mathcal{N}(f;x_i,s(t)I)}{\sum_l \mathcal{N}(f;x_l,s(t)I)}$$

- The expected denoised value of f is threrefore

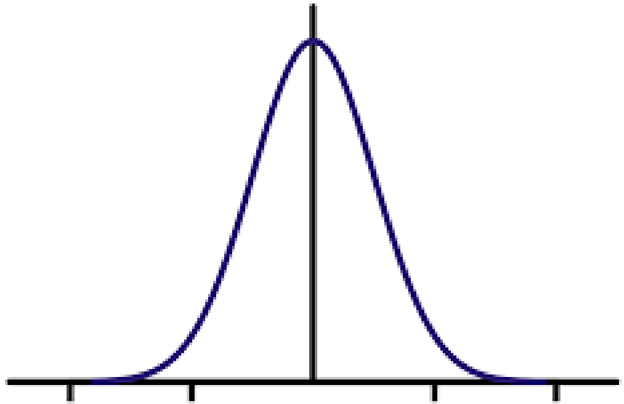
$$D(f) = \frac{\sum_i \mathcal{N}(f;x_i,s(t)I)x_i}{\sum_l \mathcal{N}(f;x_l,s(t)I)}$$

Estimating the Score

- There is a simple expression for the score:

$$score(x) = \frac{(D(x;\sigma) - x)}{\sigma^2}$$

- Intuition:



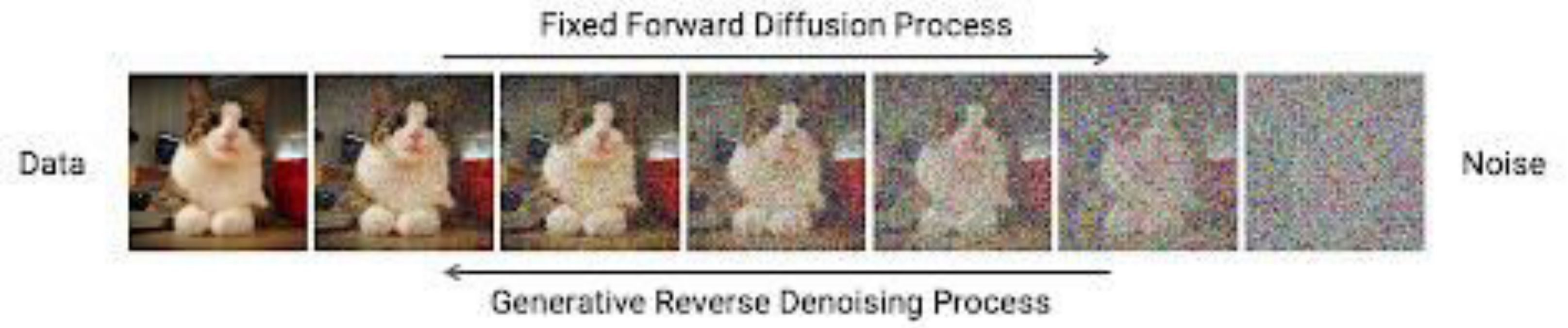
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-(x-\mu)^2/(2\sigma^2)},$$

Diffusion Models in Practice - Training

- First step: training denoising model $D(z_t, \sigma)$
  - Sample: x from training distribution, random time [0,1]
  - Shrink and add Gaussian noise according to noising schedule – result z_t
  - Train denoising model. $D(z_t, \sigma)$ taking as input noisy sample, noise

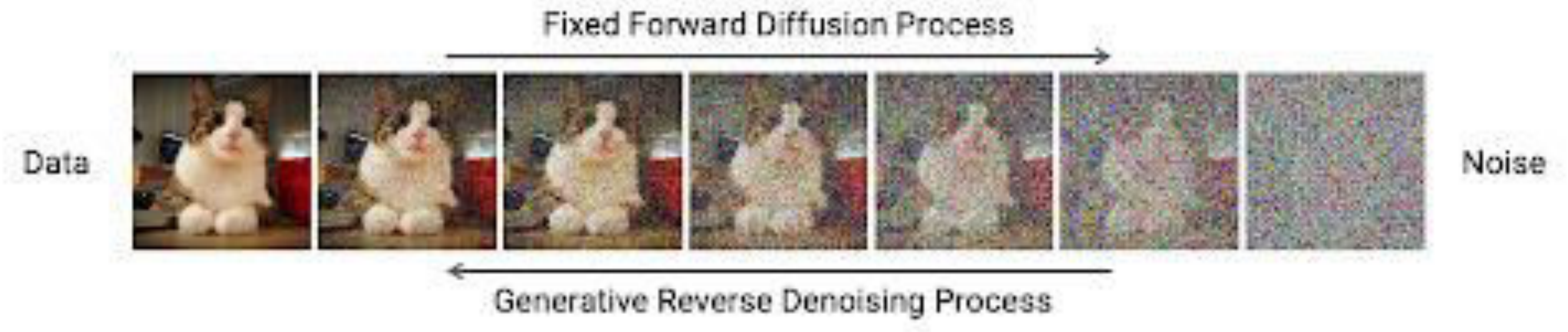
$$L(\theta) = \|D_\theta(z_t, \sigma) - x\|^2$$

  - Repeat until convergence



Diffusion Models in Practice - Sampling

- Start with random noise $z \sim N(0, I)$ and t=1 – define this as xt
- Repeat until t=0
  - Denoise x_t using $D(x_t, \sigma(t))$
  - Estimate score using $score(x) = \frac{(D(x;\sigma)-x)}{\sigma^2}$
  - Reverse process using:
  - Advance time t = t - dt $dx = -\frac{1}{2}g^2(t)\nabla_x \log(p(x,t))dt$



Diffusion Models in Practice – Point Estimates

- The ELBO on the neg log-probability p(x) is approximately given by:
  - (Stated without proof)

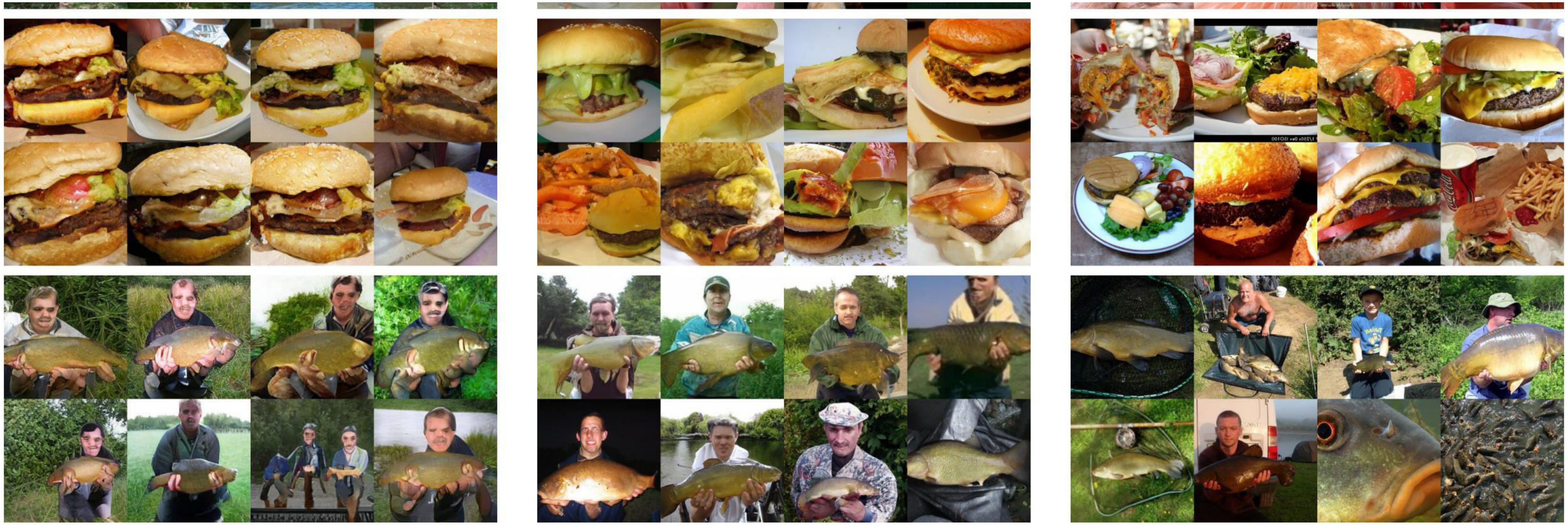
$$\mathcal{L}_T(\mathbf{x}) = \frac{T}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}), i\sim U\{1,T\}}\left[(\mathrm{SNR}(s) - \mathrm{SNR}(t))\,||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2\right],$$

$$\mathrm{SNR}(t) = \alpha_t^2/\sigma_t^2.$$

- Amazing result – probability estimation using denoising model
  - Time average of the denoising estimation error

Diffusion Models: Top Image Generators

- Diffusion models (center) beat GANs (left). Right – original images



Diffusion Models: Top Likelihood Estimators

- Diffusion models beat VAE and AR models at point density estimation

Model (Bits per dim on test set)	Type	CIFAR10 no data aug.	CIFAR10 data aug.	ImageNet 32x32	ImageNet 64x64
Previous work					
ResNet VAE with IAF [Kingma et al., 2016]	VAE	3.11			
Very Deep VAE [Child, 2020]	VAE	2.87		3.80	3.52
NVAE [Vahdat and Kautz, 2020]	VAE	2.91		3.92	
Glow [Kingma and Dhariwal, 2018]	Flow		$3.35^{(B)}$	4.09	3.81
Flow++ [Ho et al., 2019a]	Flow	3.08		3.86	3.69
PixelCNN [Van Oord et al., 2016]	AR	3.03		3.83	3.57
PixelCNN++ [Salimans et al., 2017]	AR	2.92			
Image Transformer [Parmar et al., 2018]	AR	2.90		3.77	
SPN [Menick and Kalchbrenner, 2018]	AR				3.52
Sparse Transformer [Child et al., 2019]	AR	2.80			3.44
Routing Transformer [Roy et al., 2021]	AR				3.43
Sparse Transformer + DistAug [Jun et al., 2020]	AR		$2.53^{(A)}$		
DDPM [Ho et al., 2020]	Diff		$3.69^{(C)}$		
EBM-DRL [Gao et al., 2020]	Diff		$3.18^{(C)}$		
Score SDE [Song et al., 2021b]	Diff	2.99			
Improved DDPM [Nichol and Dhariwal, 2021]	Diff	2.94			3.54
Concurrent work					
CR-NVAE [Sinha and Dieng, 2021]	VAE		$2.51^{(A)}$		
LSGM [Vahdat et al., 2021]	Diff	2.87			
ScoreFlow [Song et al., 2021a] (variational bound)	Diff		$2.90^{(C)}$	3.86	
ScoreFlow [Song et al., 2021a] (cont. norm. flow)	Diff	2.83	$2.80^{(C)}$	3.76	
Our work					
VDM (variational bound)	Diff	**2.65**	$\mathbf{2.49}^{(A)}$	**3.72**	**3.40**

Conditional Diffusion Models

- Conditional DMs: basically the same as have multiple DMs
- Share the same denoiser, but condition it:
  - Noisy image
  - Time (or noise sigma)
  - Conditioning labels

$$E_{c \in C} E_{x \in X_c} E_{t \in [0,1]} \|D(z_t, t, c) - x\|^2$$

Classifier-Free Guidance

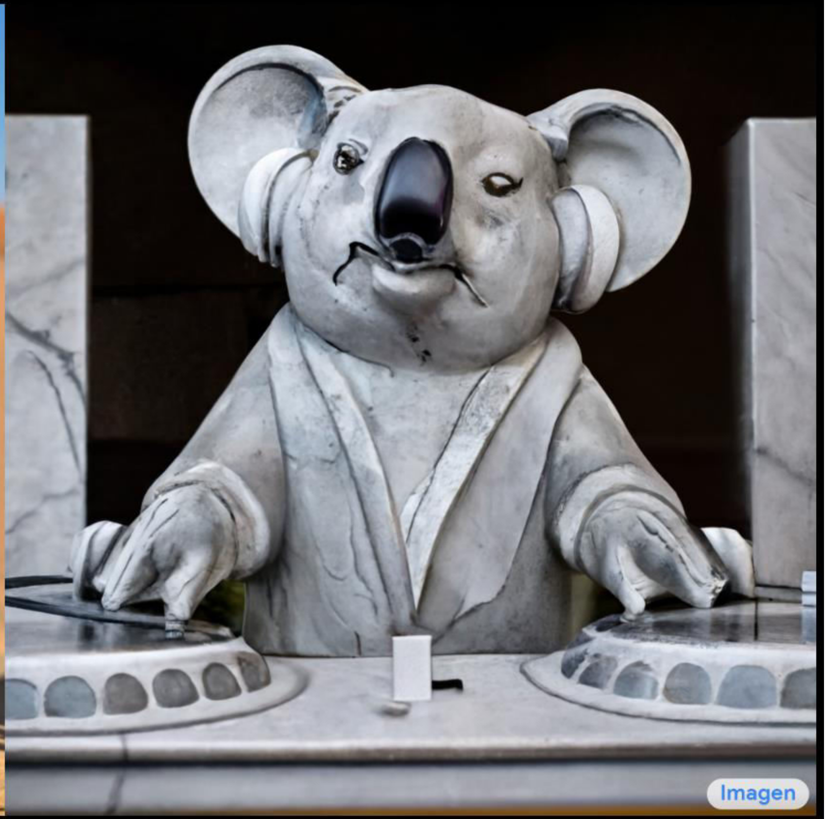
- Conditioning alone is often insufficient for text-guided generation
- Trick (not well grounded in theory):
  - Denoise according to conditional and unconditional models
  - Use the following combination as denoising value:

$$D_{comb}(x,t) = D_u(x,t) + w * [D_c(x,t,c) - D_u(x,t)]$$

Text-to-Image Models

- Dall-E 2, Imagen

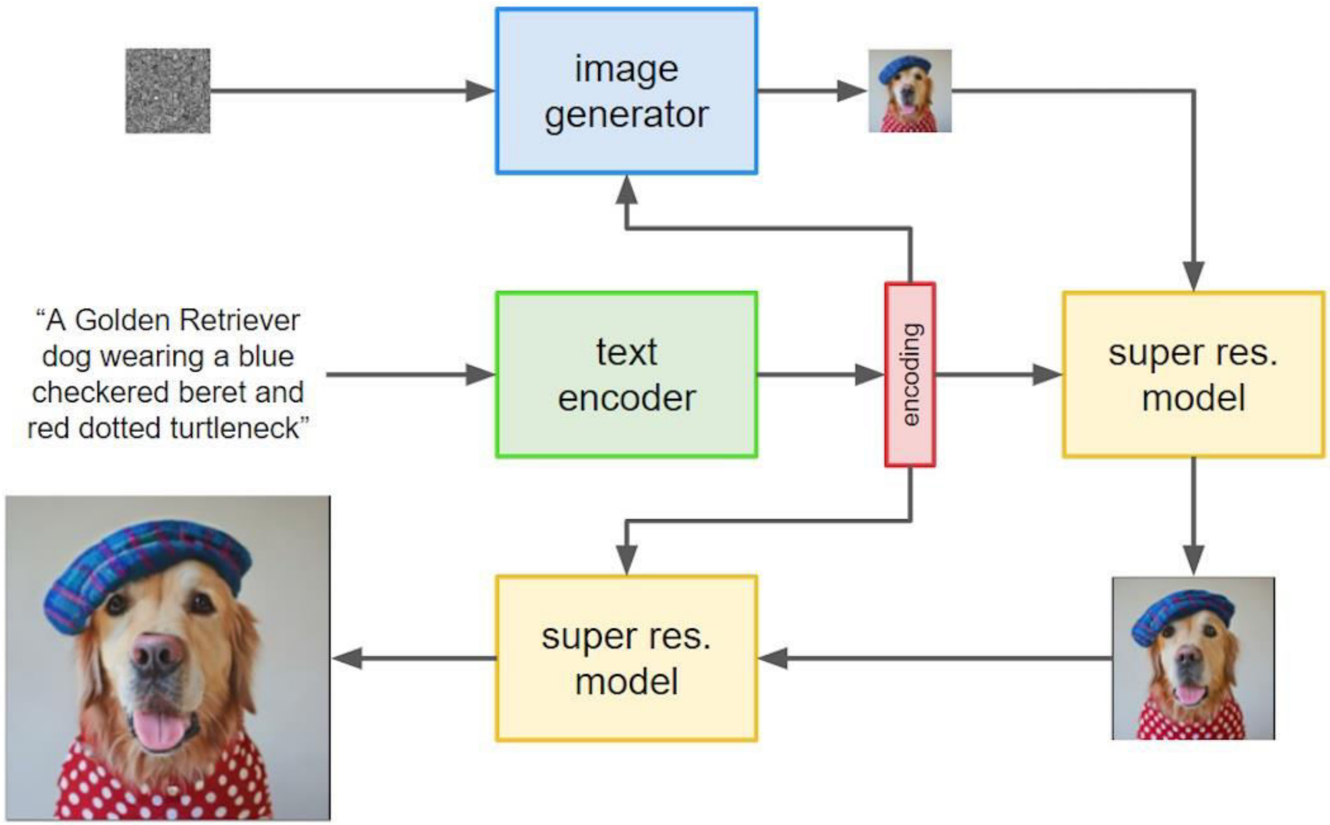
A small cactus wearing a straw hat and neon sunglasses in the Sahara desert.



A marble statue of a Koala DJ in front of a marble statue of a turntable. The Koala has wearing large marble headphones.

Text-to-Image Architectures

- Sample Low-resolution image conditional on text
- Sample high-resolution image conditional on low-res and text



Latent Diffusion Models

- Tokenizer: Train to map image to a small token grid and back
- Diffusion: Perform diffusion of the low-res token grid
- Dekonizer: Map synthesized token grid to high-res image

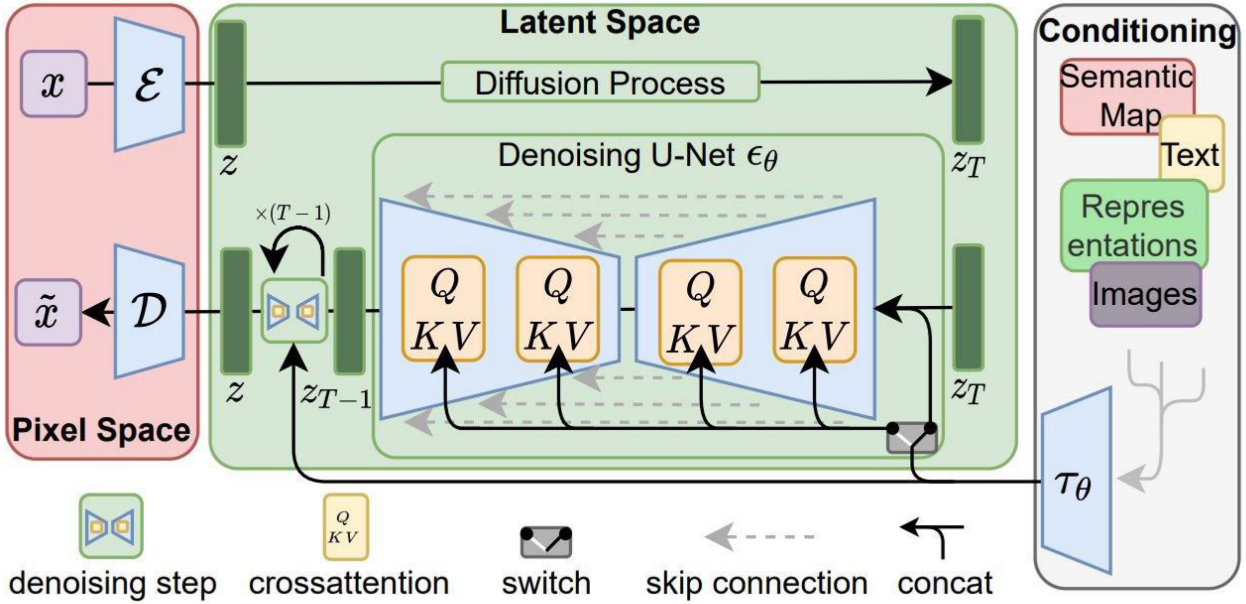


Imagen Video

- Same as above but for video
- https://imagen.research.google/video/

Dreamix

- Train imagen-video denoiser on the video of the user
- Then condition on some new text
- https://dreamix-video-editing.github.io/

Several Prediction Targets

- There are several loss functions used for the denoising models
- They are equivalent up to weighting different time steps
- In practice the choice makes a difference
  - Predict x directly (x prediction)
  - Predict noise direction (e prediction)
  - Prediction combination (v prediction) $\mathbf{v} \equiv \alpha_t \epsilon - \sigma_t \mathbf{x},$

$$L_\theta = \|\epsilon - \hat{\epsilon}_\theta(\mathbf{z}_t)\|_2^2 = \left\| \frac{1}{\sigma_t}(\mathbf{z}_t - \alpha_t \mathbf{x}) - \frac{1}{\sigma_t}(\mathbf{z}_t - \alpha_t \hat{\mathbf{x}}_\theta(\mathbf{z}_t)) \right\|_2^2 = \frac{\alpha_t^2}{\sigma_t^2} \|\mathbf{x} - \hat{\mathbf{x}}_\theta(\mathbf{z}_t)\|_2^2,$$

$$L_\theta = \|\mathbf{v}_t - \hat{\mathbf{v}}_t\|_2^2 = (1 + \frac{\alpha_t^2}{\sigma_t^2})\|\mathbf{x} - \hat{\mathbf{x}}_t\|_2^2$$

MagVIT/MUSE

- Instead of Gaussian noise, use binomial noise (remove pixels)
  - Repeat until all token complete
  - At each step, denoiser guesses all missing token
  - Most certain tokens are kept, the of predicted ones are deleted

A latte with "Muse" written in latte art