A Gentle Introduction to Diffusion Models

EE367/CS448I: Computational Imaging

stanford.edu/class/ee367

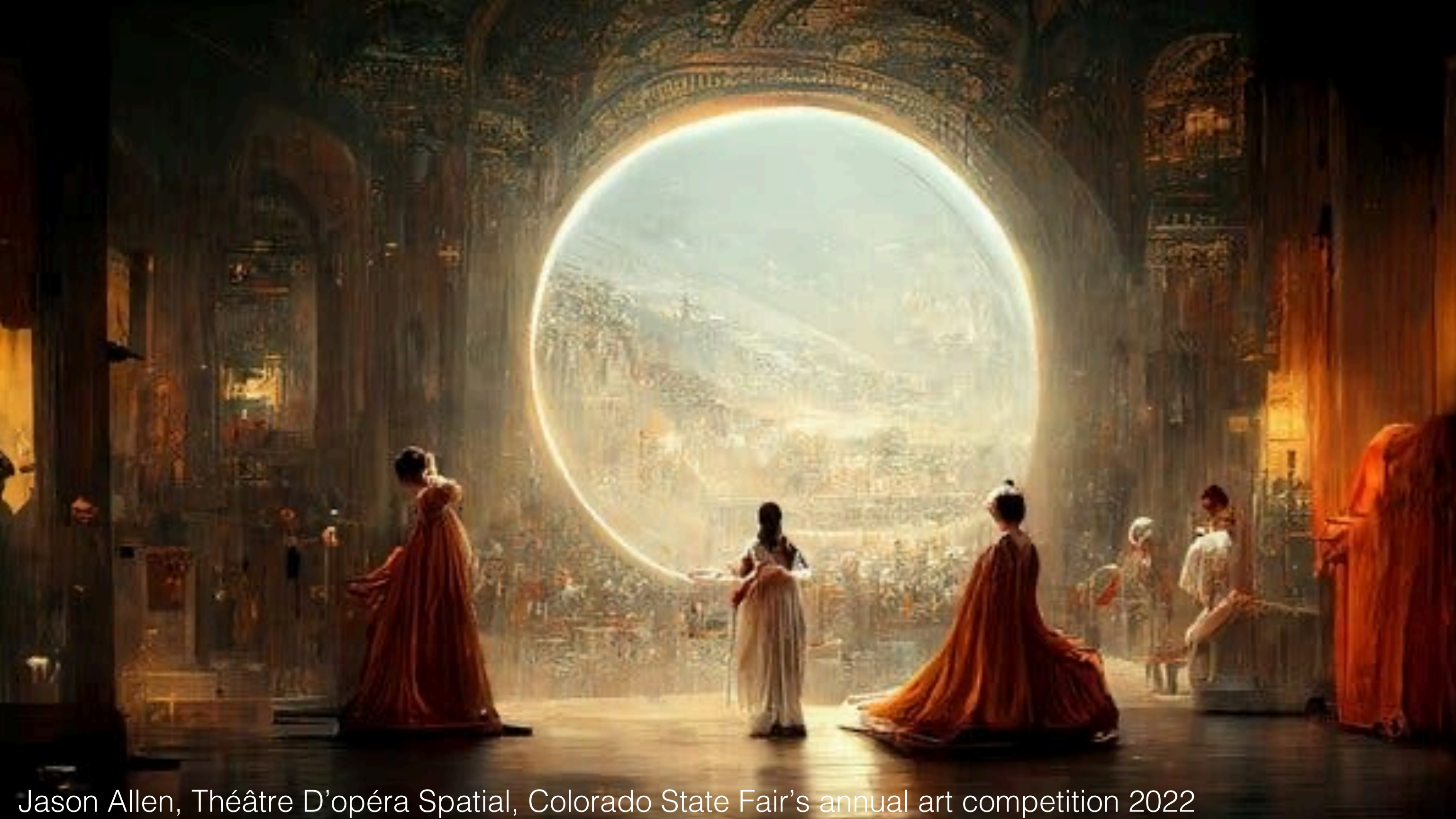
Lecture 12

Gordon Wetzstein

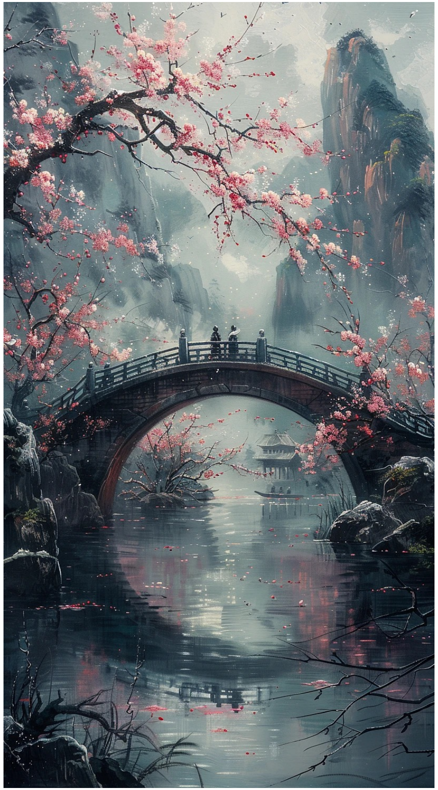
Stanford University

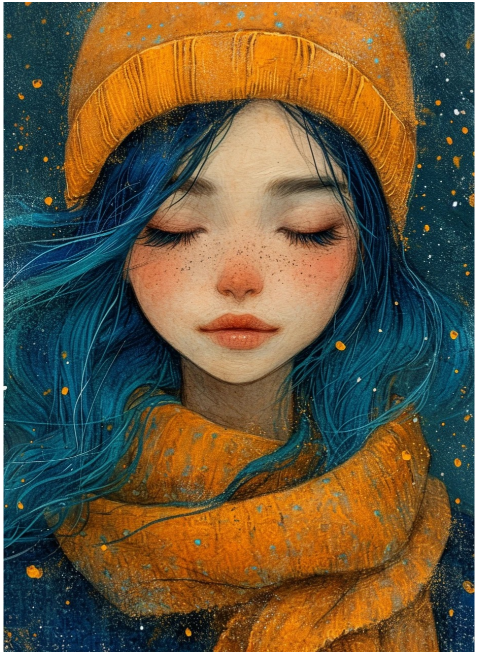
“Creating noise from data is easy; creating data from noise is generative modeling.”

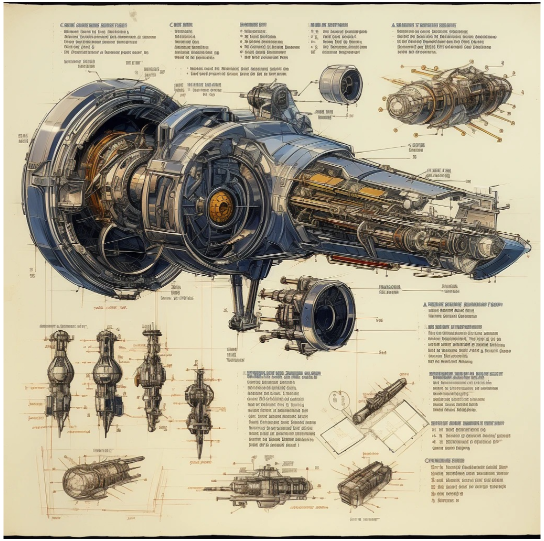
–Song et al., ICLR 2021

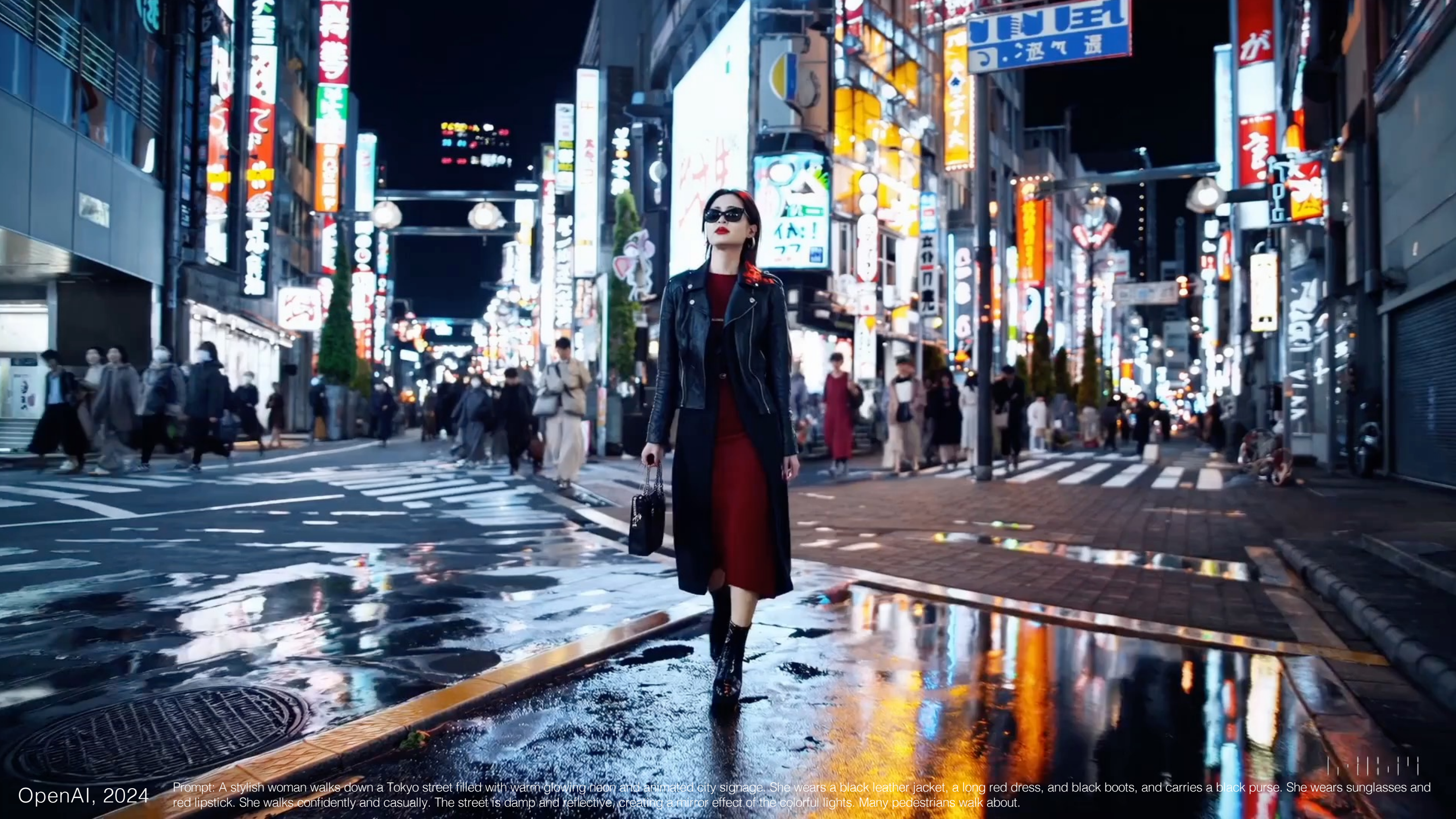

Jason Allen, Théâtre D’opéra Spatial, Colorado State Fair’s annual art competition 2022

Images generated with Midjourney


two people walk the bridge under a large cherry blossom, in the style of atmospheric paintings, gray and aquamarine, airbrush art, dark white and dark gray, lovely --ar 35:64 --stylize 750 --v 6


a girl with blue hair with eyes closed, in the style of the stars art group (xing xing) , yellow and orange, water drops, romantic illustrations, intense lines, gongbi, loose paint application --ar 71:98 --stylize 750 --v 6


sketches blueprint of futuristic sci fi, warp engine, warp bubble configurations, hyperdrive engines, equations for calibrating warp engine, star drive engines, placement inside ship, formulas and annotations, schematic by parts, golden ratio, fake detail, trending pixiv fanbox, acrylic palette knife, style of style of makoto shinkai studio ghibli genshin impact james gilleard greg rutkowski chiho aoshi


OpenAI, 2024

Prompt: A stylish woman walks down a Tokyo street filled with warm glowing neon and animated city signage. She wears a black leather jacket, a long red dress, and black boots, and carries a black purse. She wears sunglasses and red lipstick. She walks confidently and casually. The street is damp and reflective, creating a mirror effect of the colorful lights. Many pedestrians walk about.

Google DeepMind Genie 3

Disclaimer

- Extremely active area – cannot possibly cover everything in 1 introductory lecture!
- Focus on fundamentals of diffusion models to get you started (today) and simple & robust methods to solve inverse problems with diffusion model-based priors (next lecture)

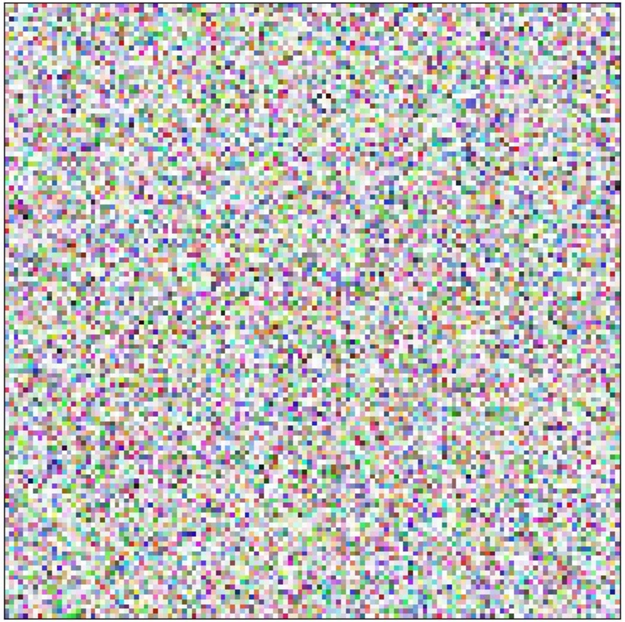
Overview

- High-level introduction
- Score-based generative modeling / diffusion models
  - The score function
  - Estimating the score function (i.e., training the diffusion model)
  - Sampling (i.e., generating new samples)
- Diffusion model architectures & attention mechanisms
- Outlook

High-level Introduction

Slides used with permission from Steve Seitz
(Graphics in 5 Minutes)

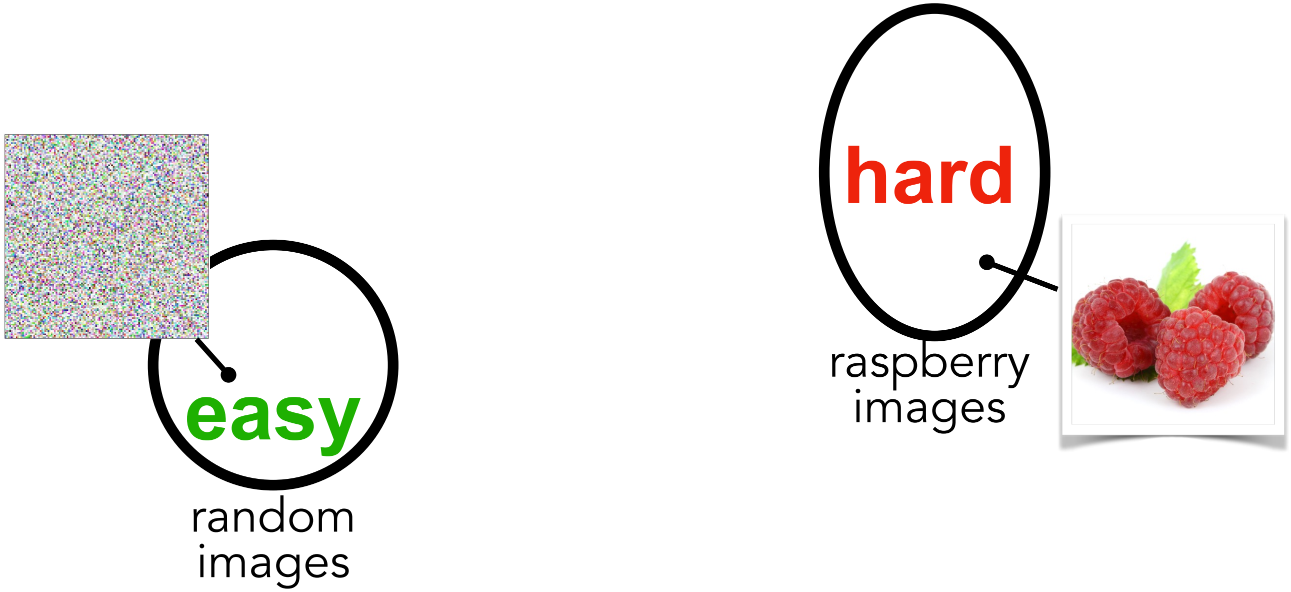
Generate 100 images

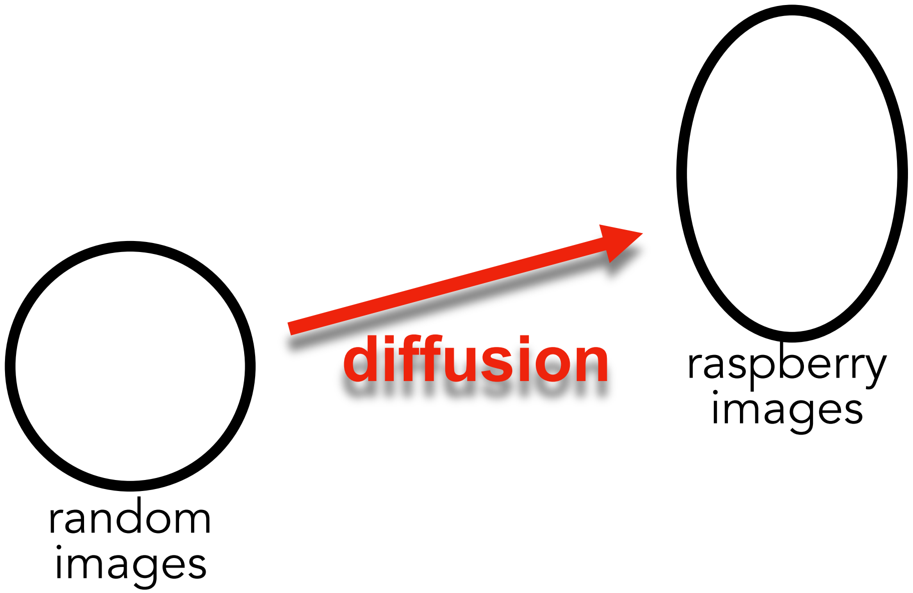


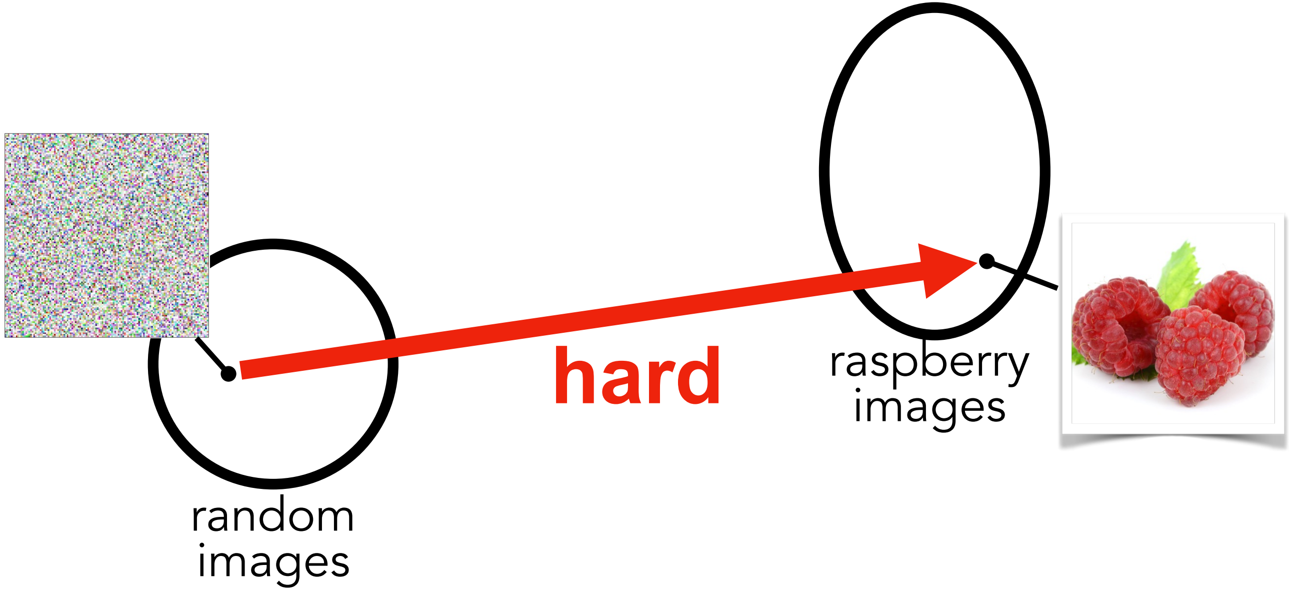
Generate 100 images

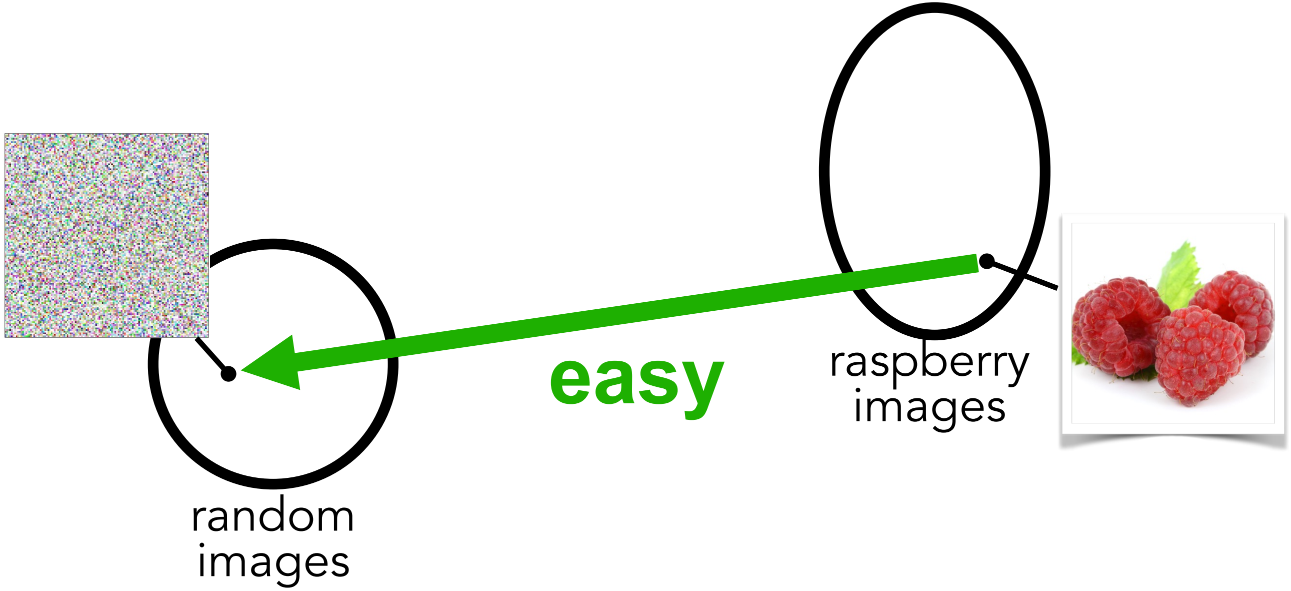
Steve Seitz, Graphics in 5 Minutes, https://www.youtube.com/@g5min

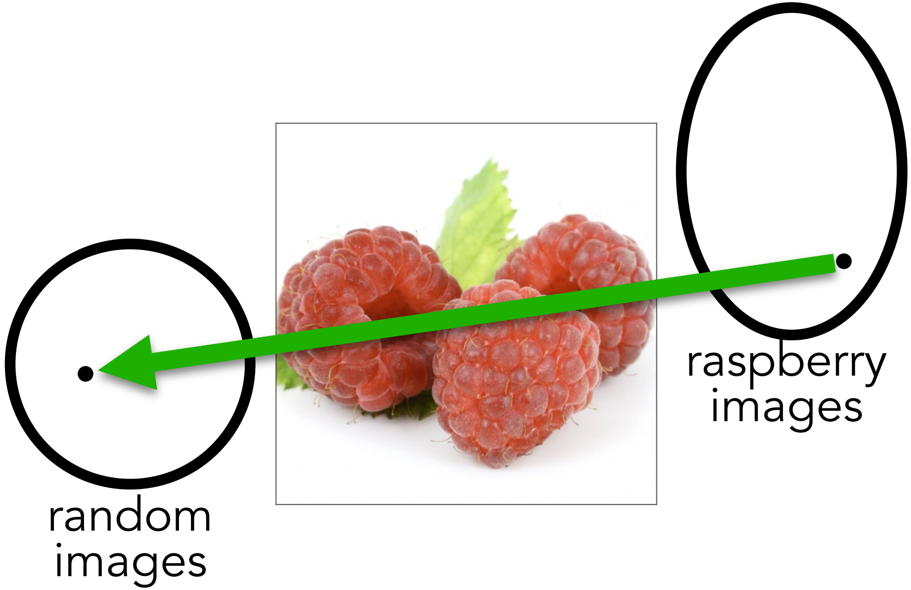
Generate 100 images of **raspberries**

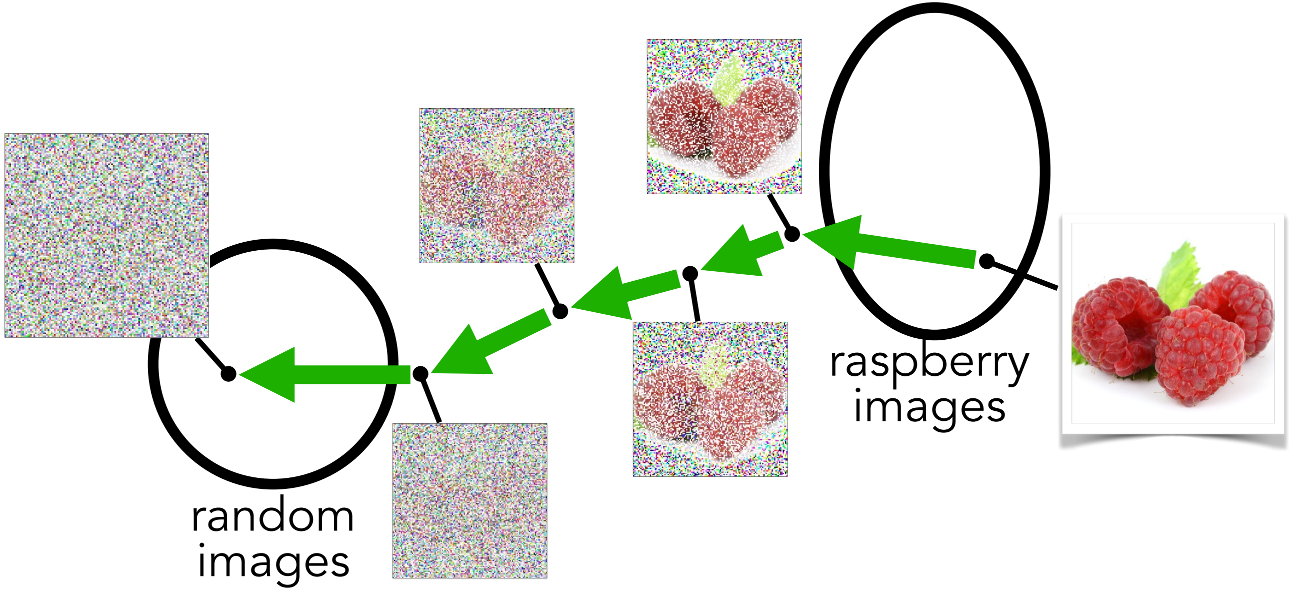

easy
random images
hard
raspberry images

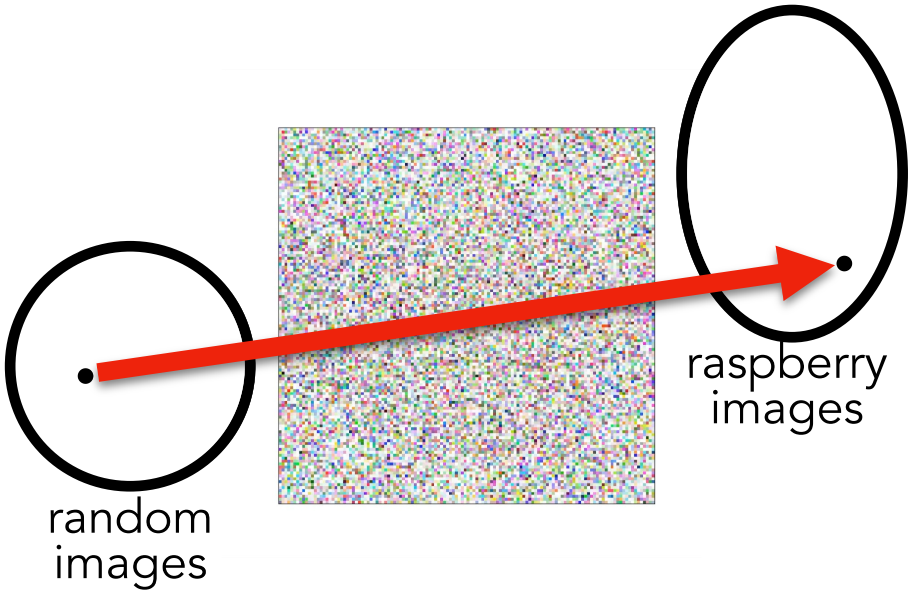

diffusion
random images
raspberry images

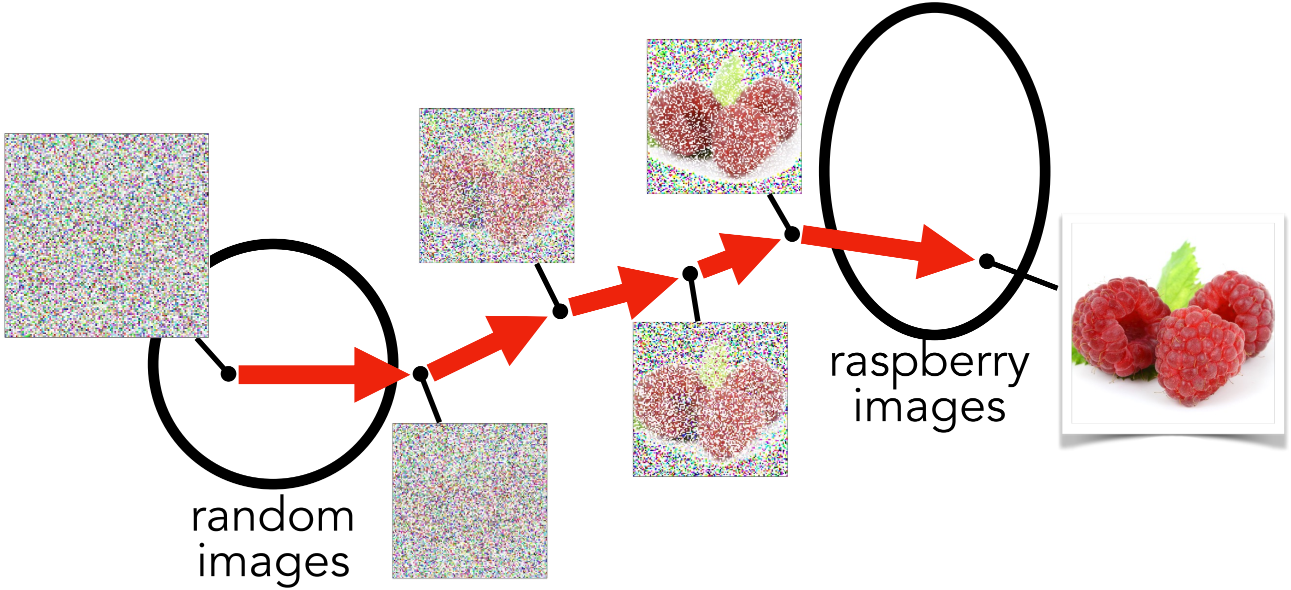

hard
raspberry images
random images

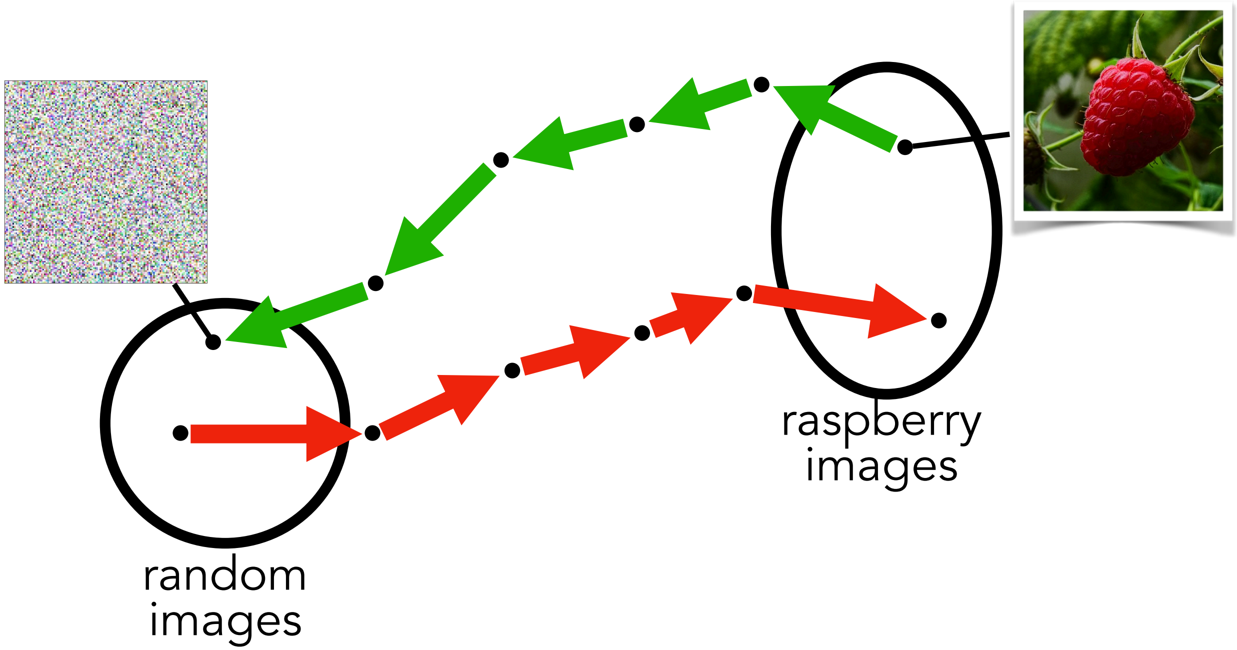

easy
raspberry images
random images

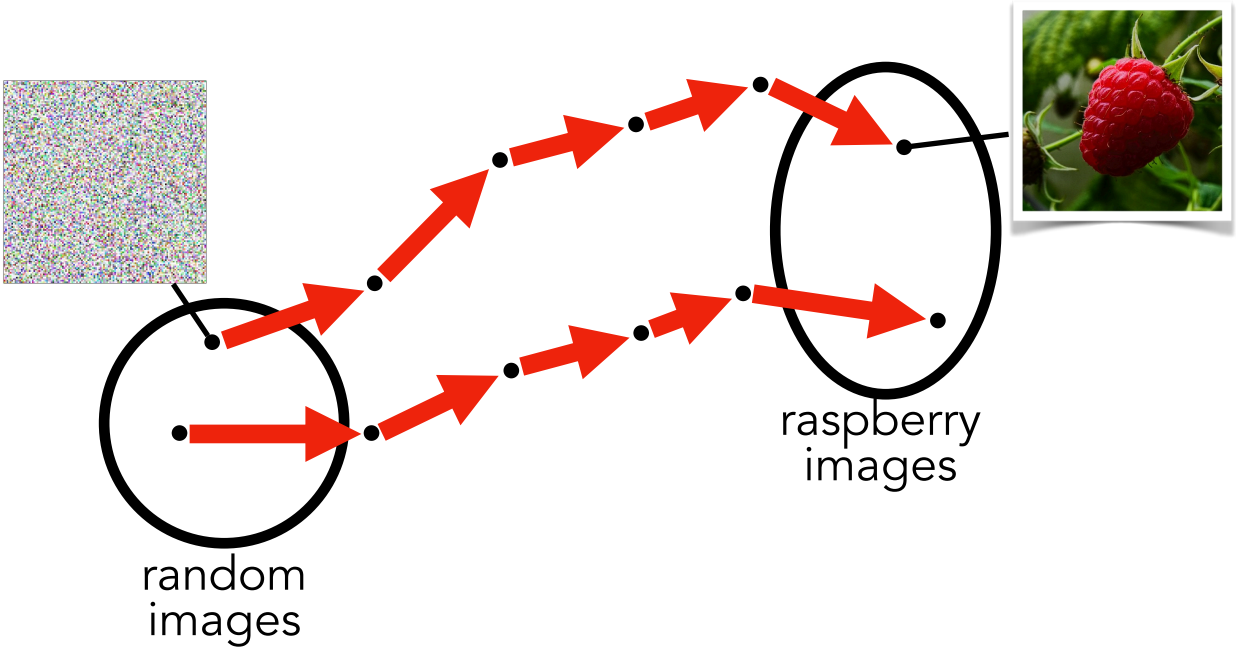

raspberry images
random images

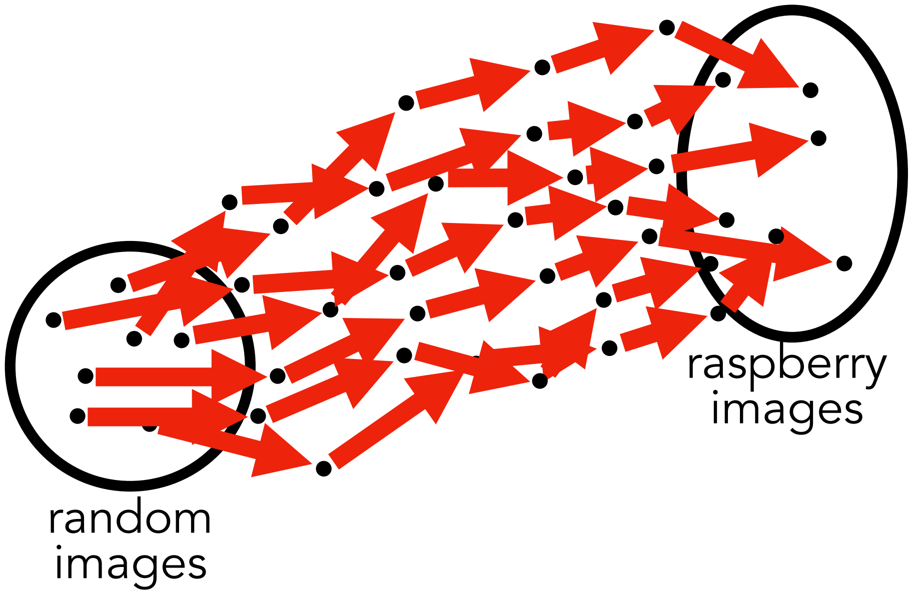

raspberry images
random images

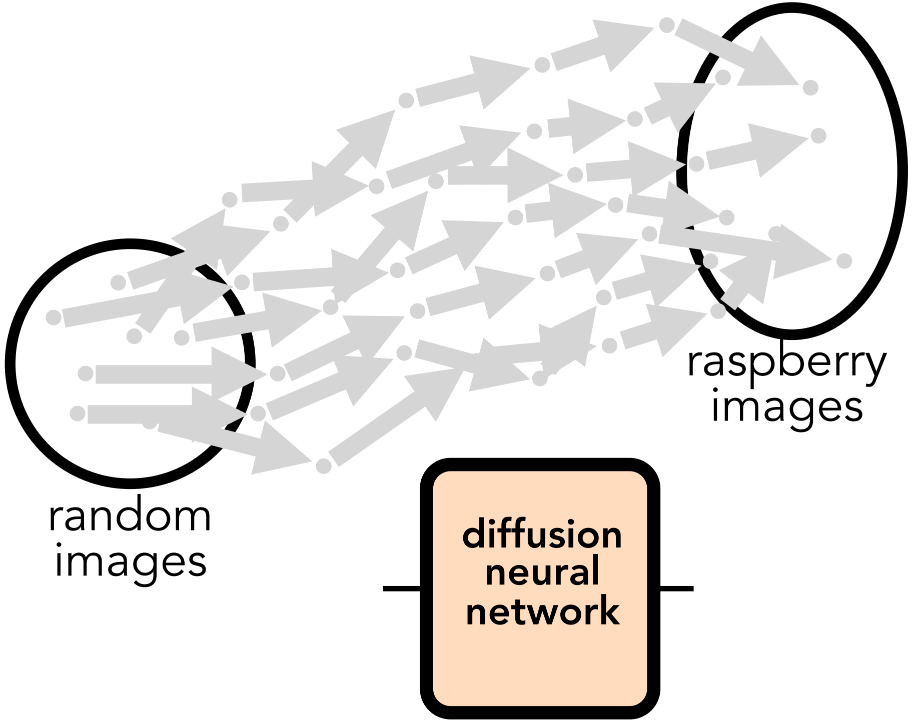

random images
raspberry images

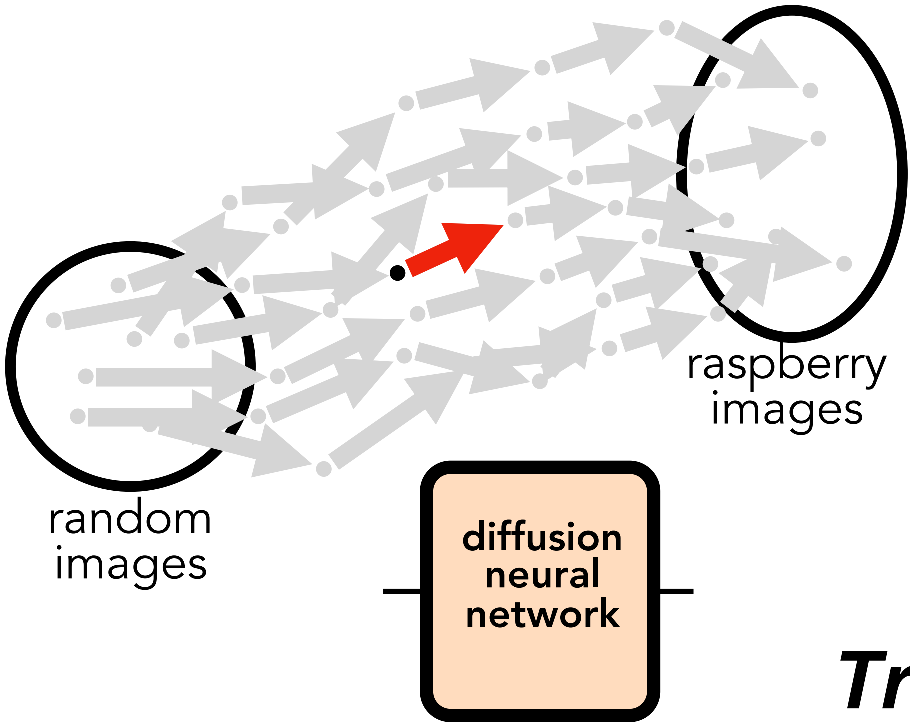

random images
raspberry images


random images
raspberry images

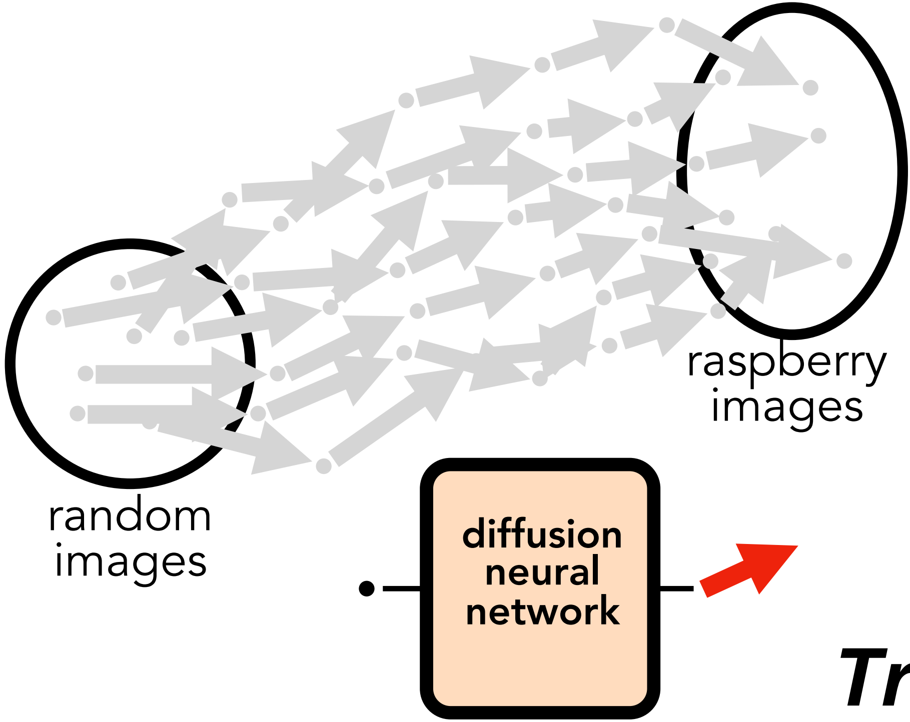

random images
raspberry images


random images
raspberry images

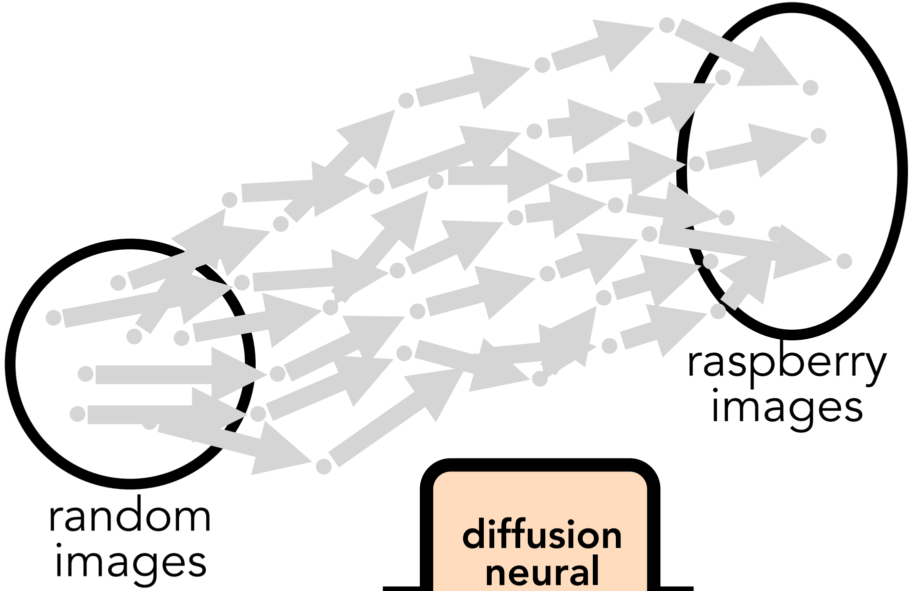

random images
raspberry images
diffusion neural network



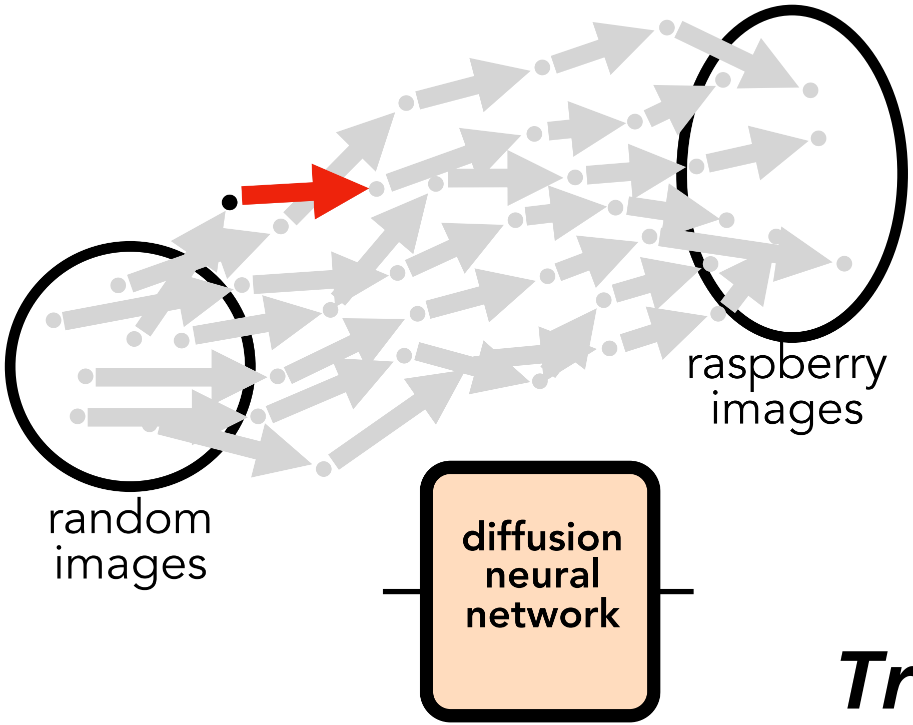
Training



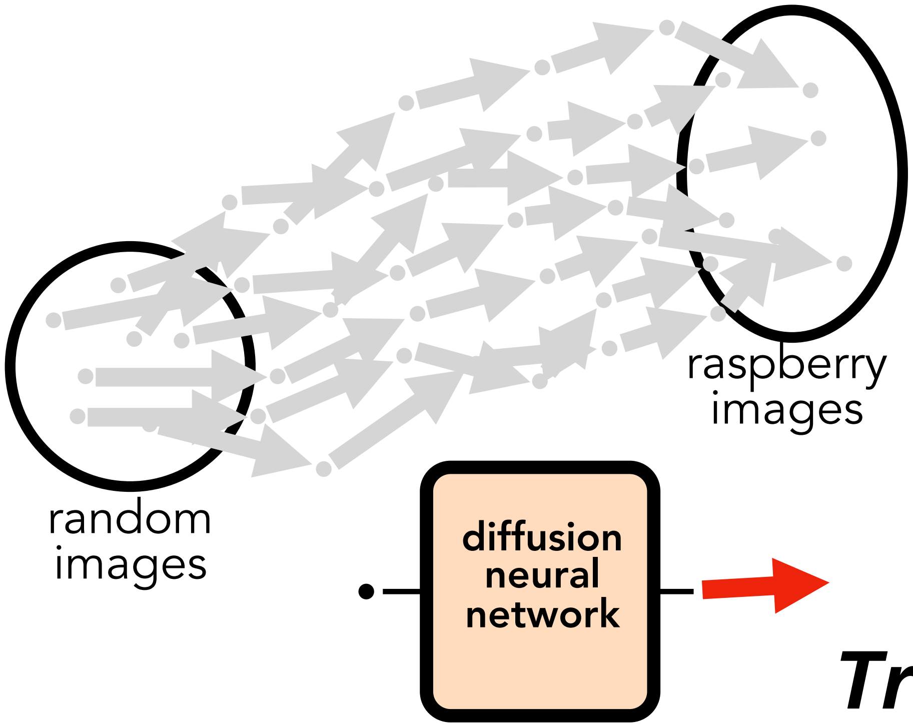
Training



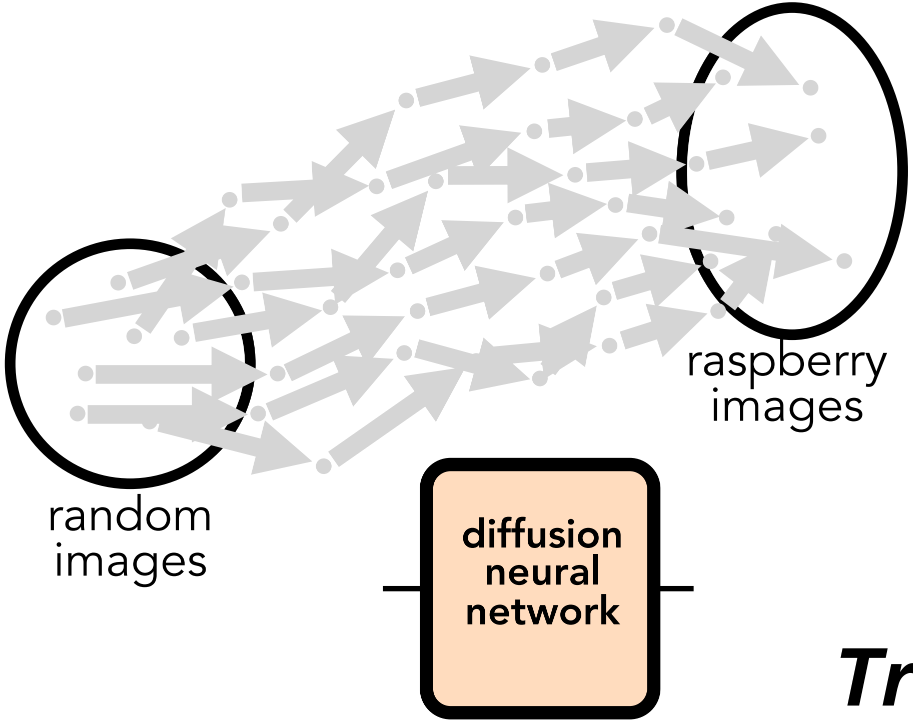
Training



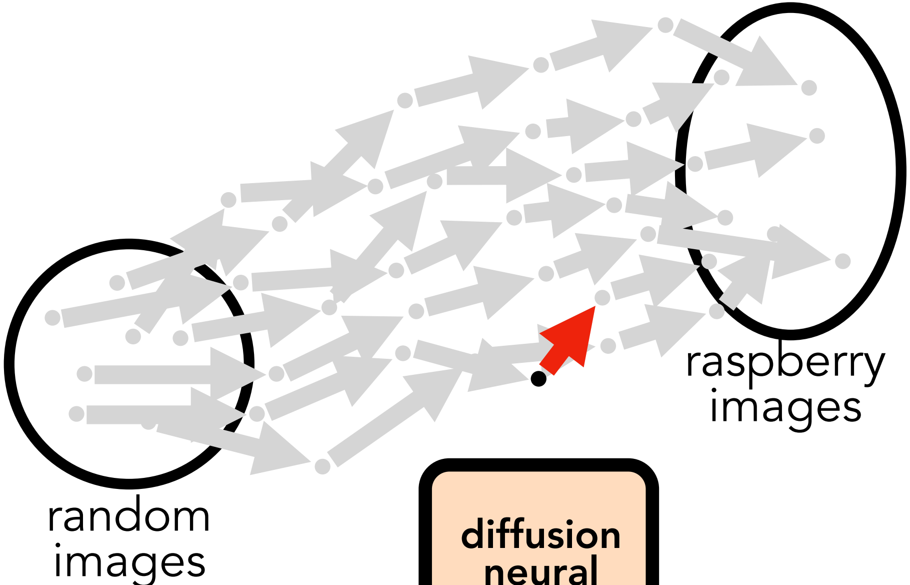
Training

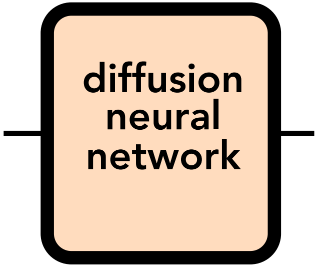


Training

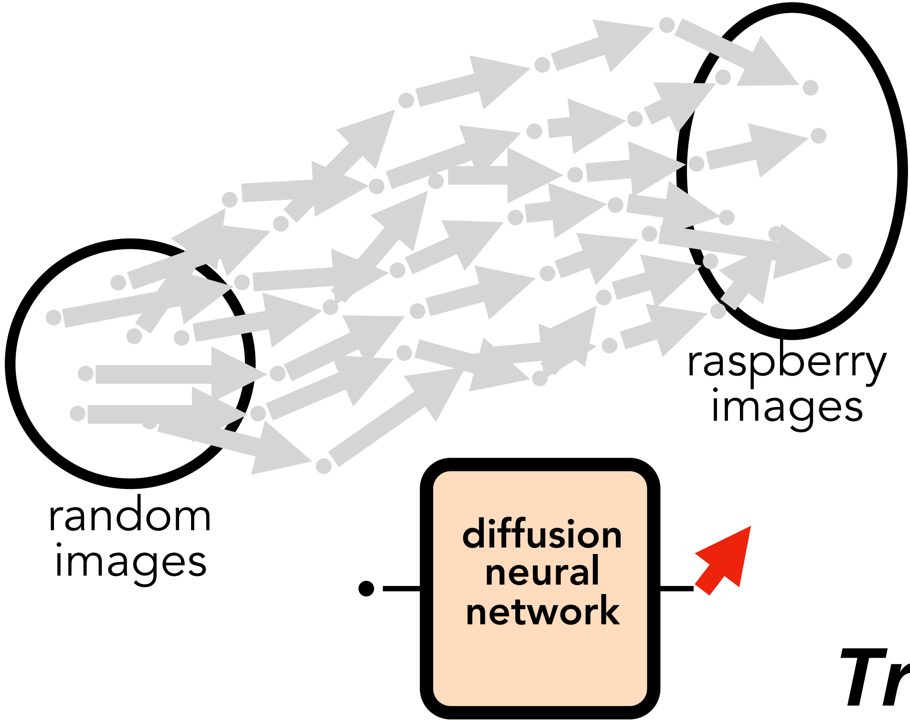


Training

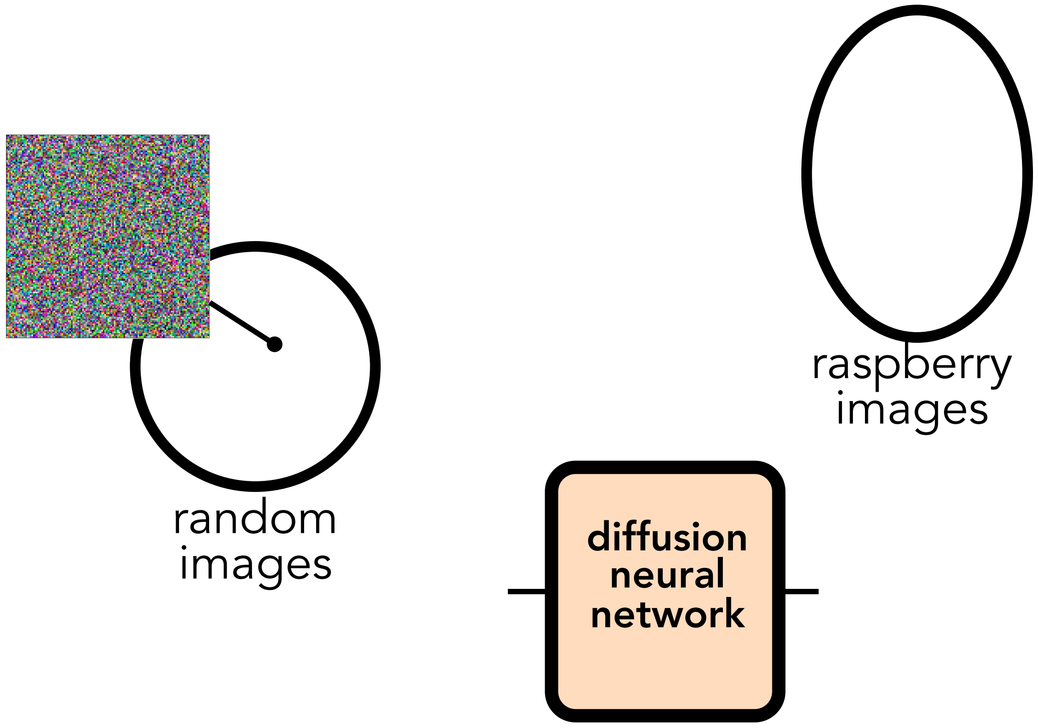


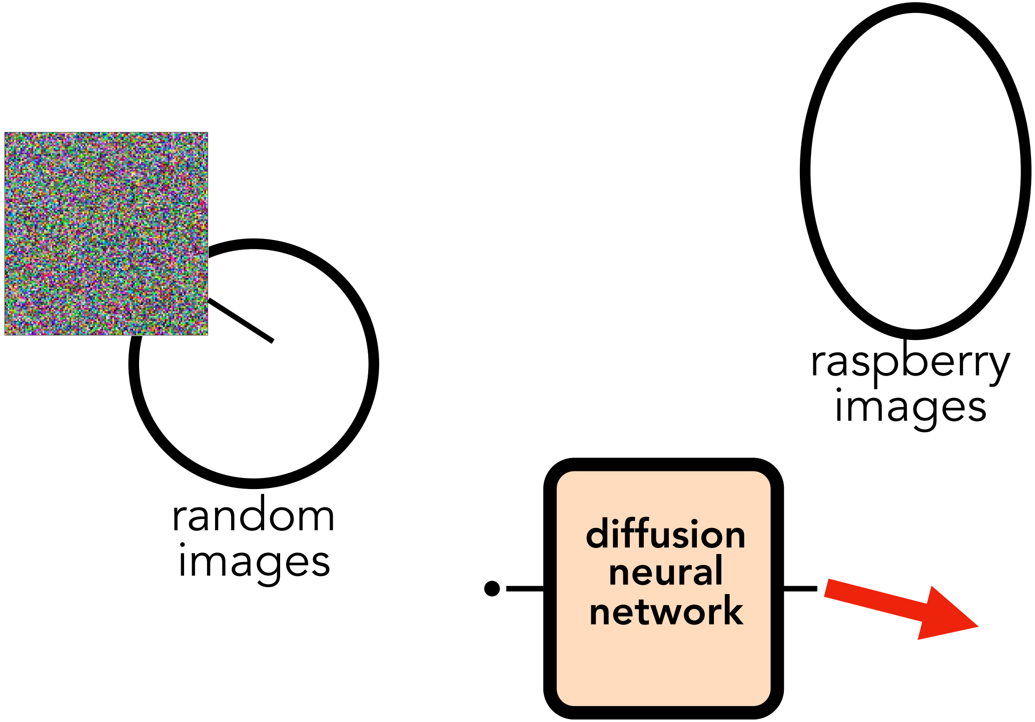


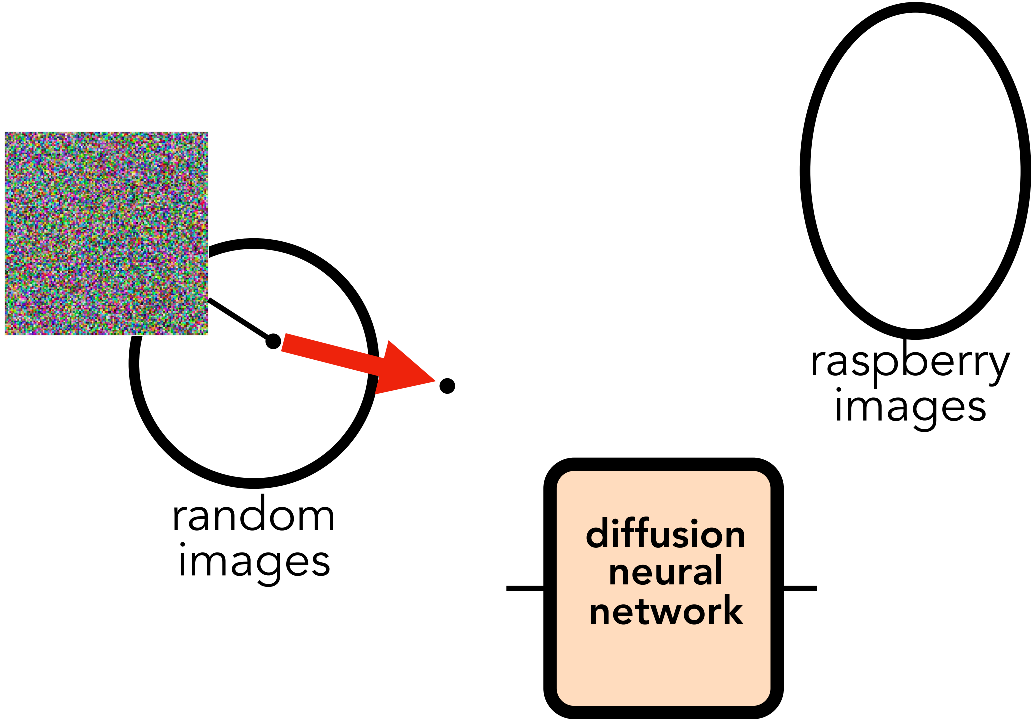
Training

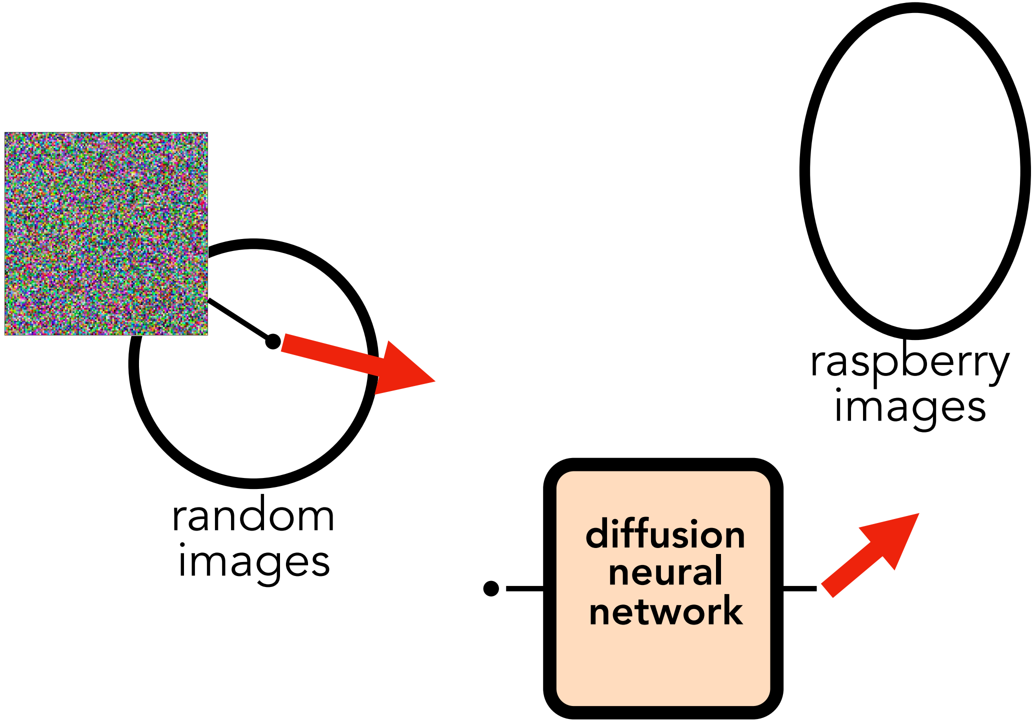


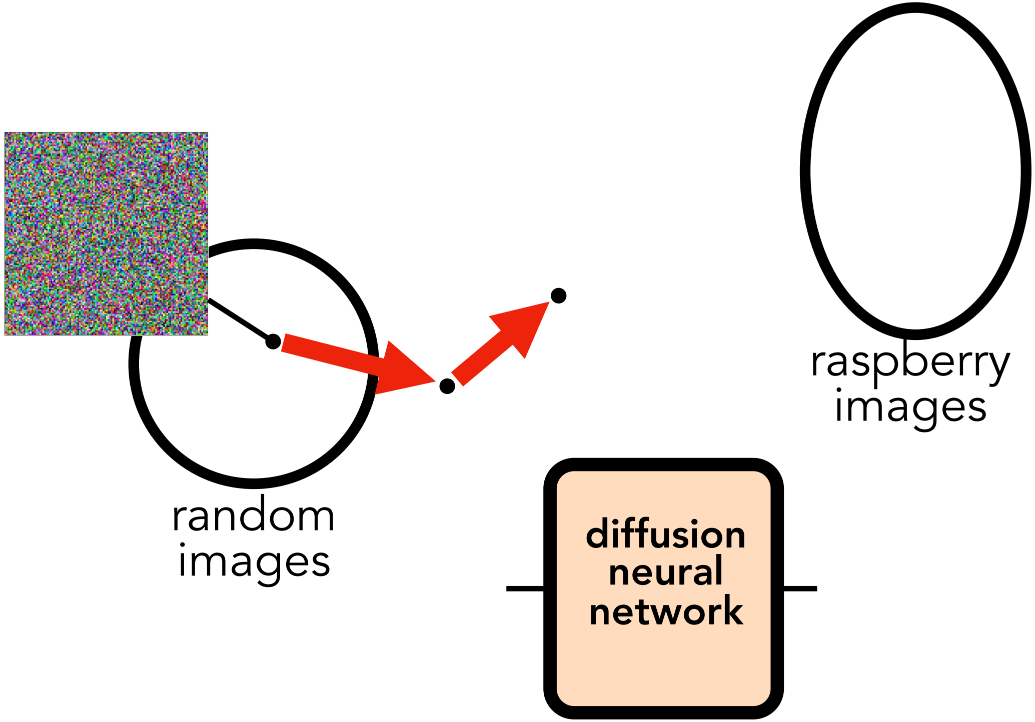
Training

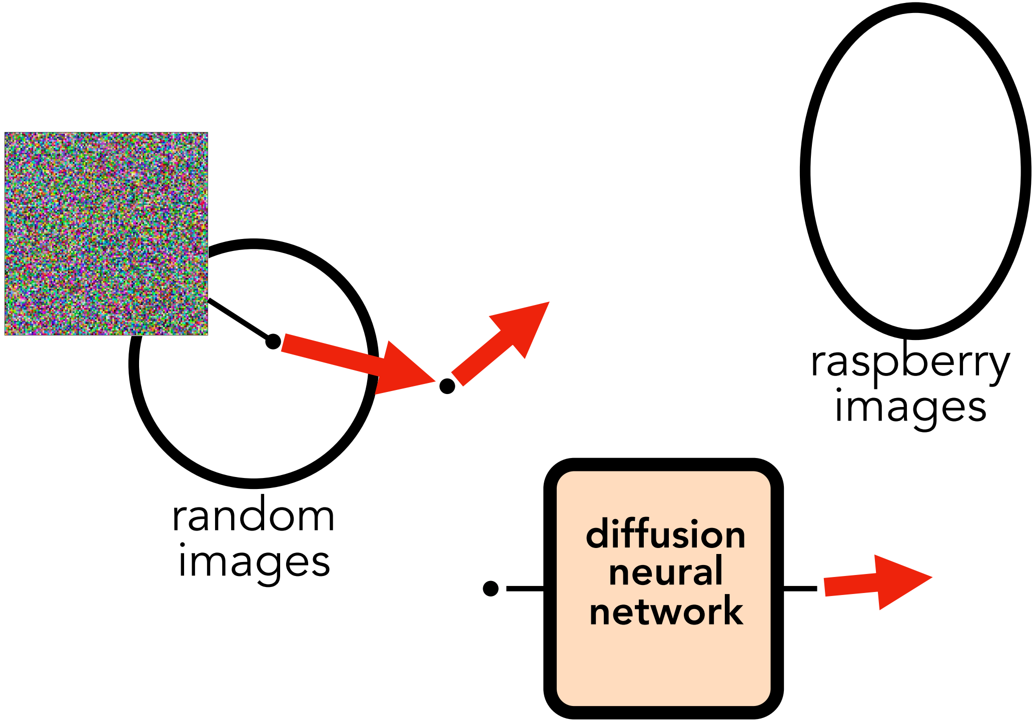

random images
raspberry images
diffusion neural network

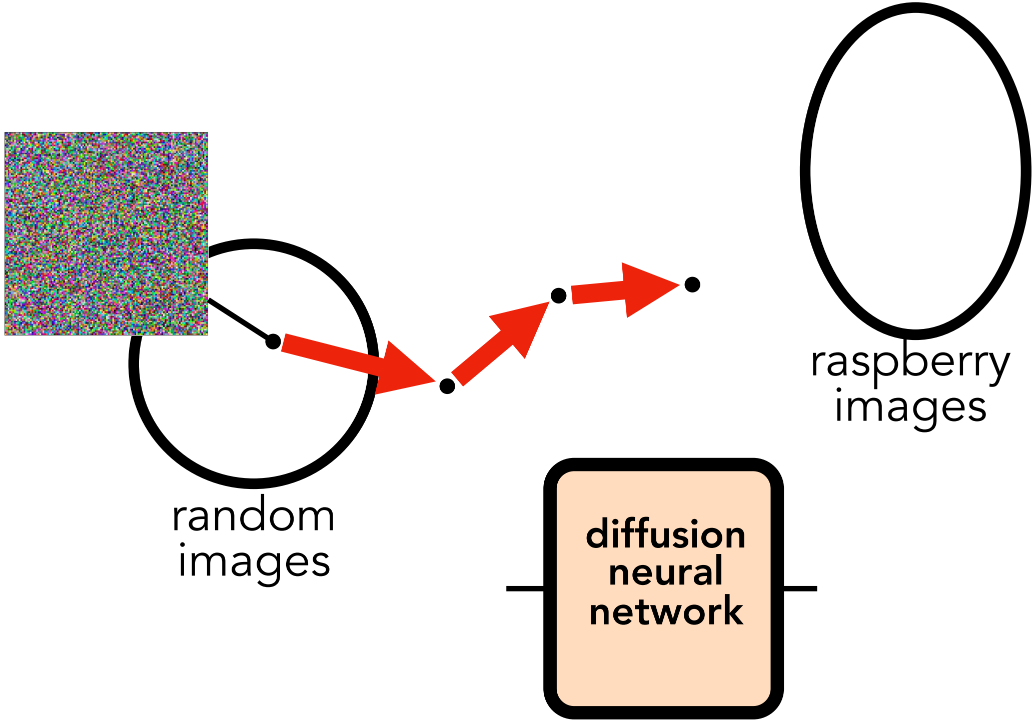

raspberry images
random images
diffusion neural network

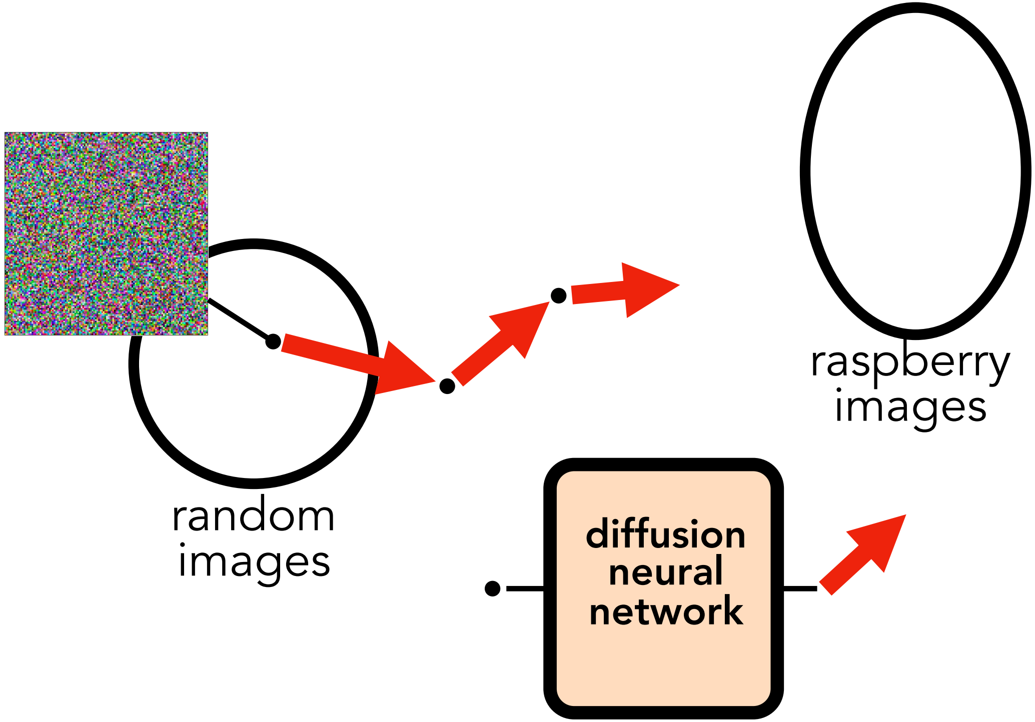

random images
raspberry images
diffusion neural network

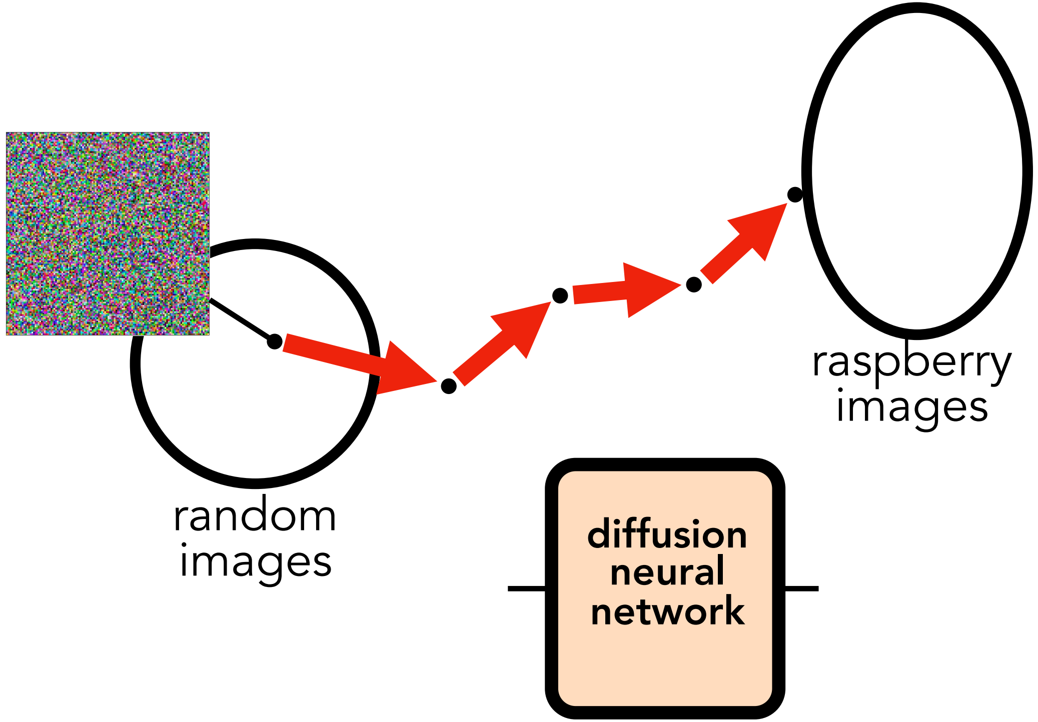

random images
raspberry images
diffusion neural network

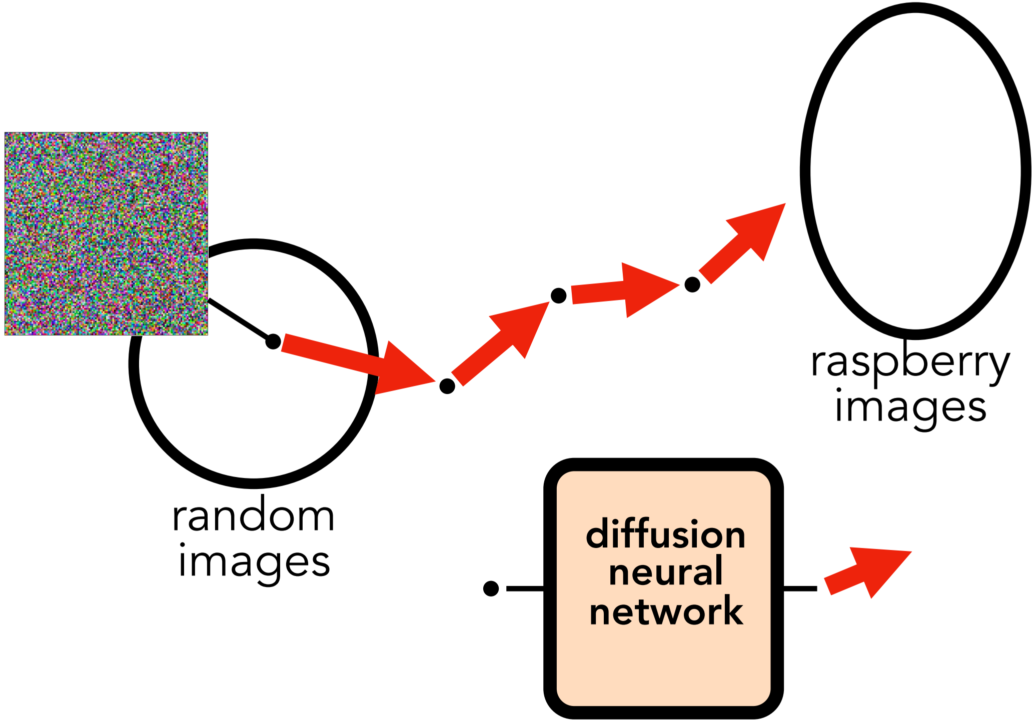

random images
raspberry images
diffusion neural network

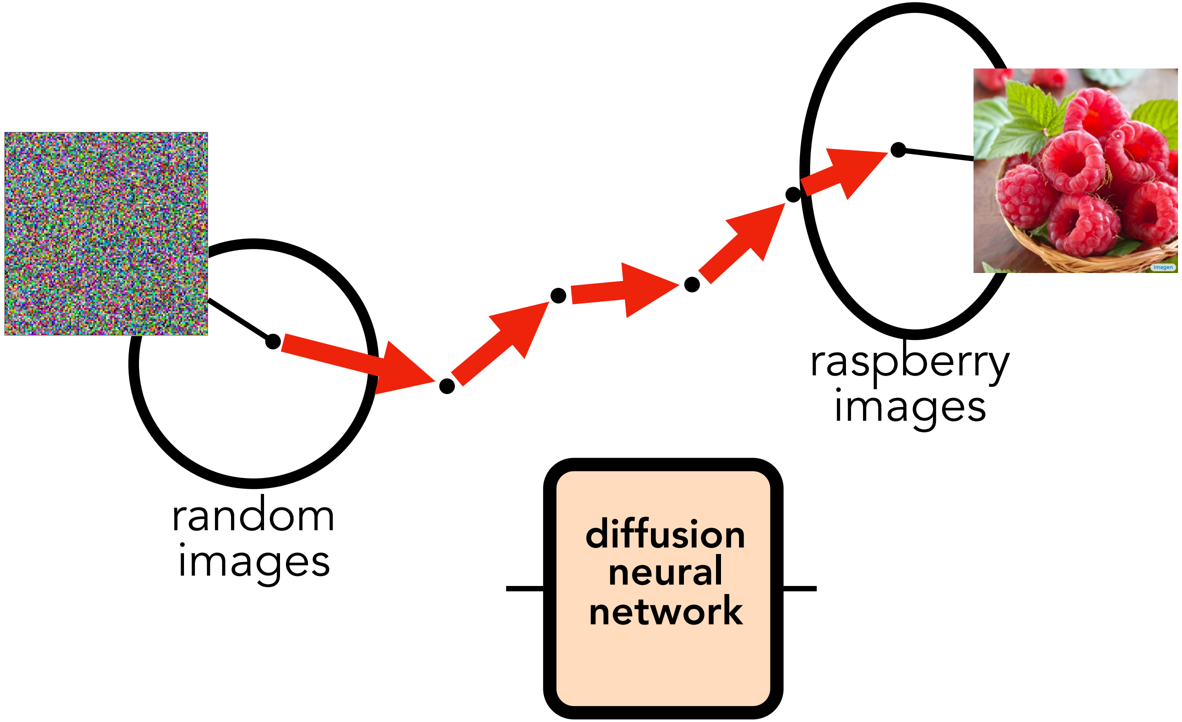

random images
raspberry images
diffusion neural network

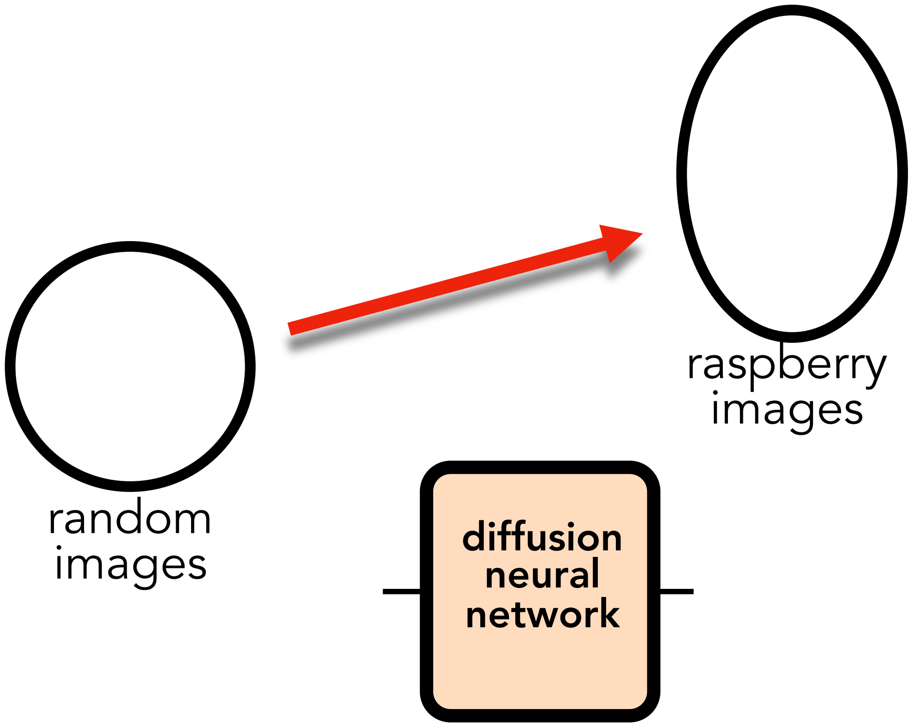

random images
raspberry images
diffusion neural network

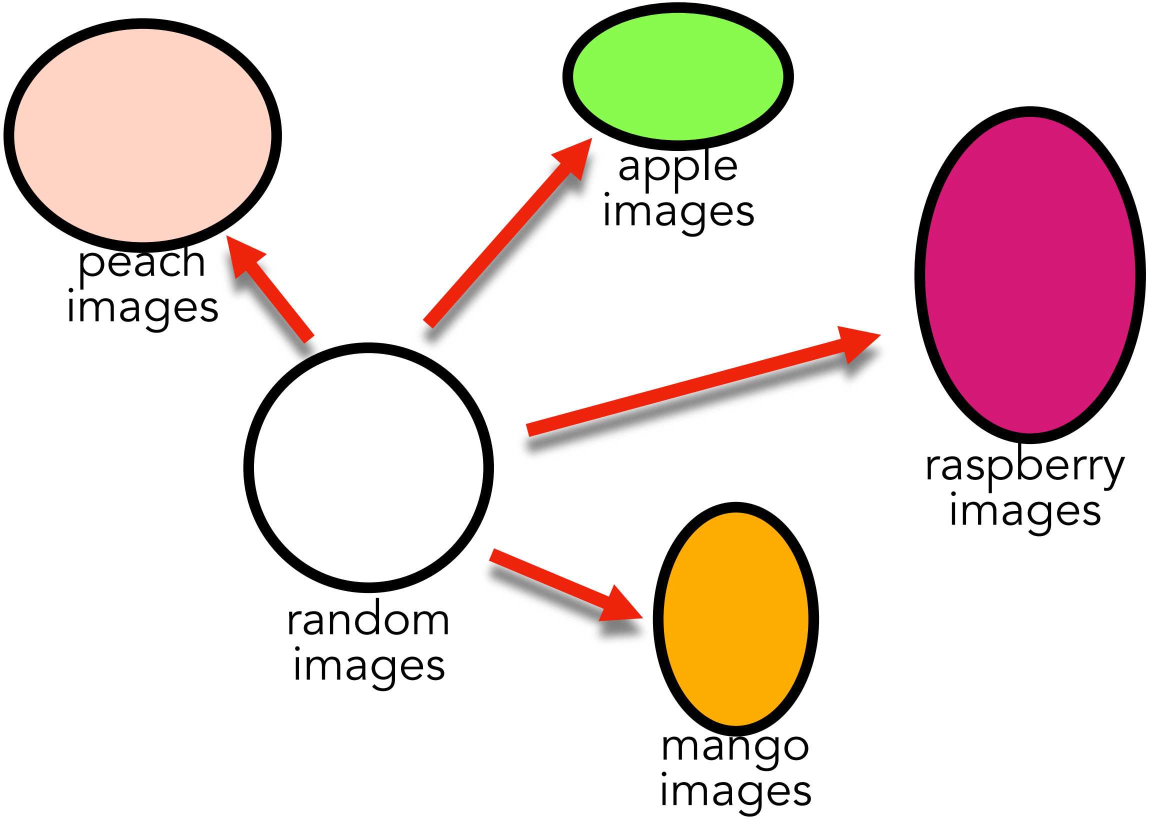

raspberry images
random images
diffusion neural network

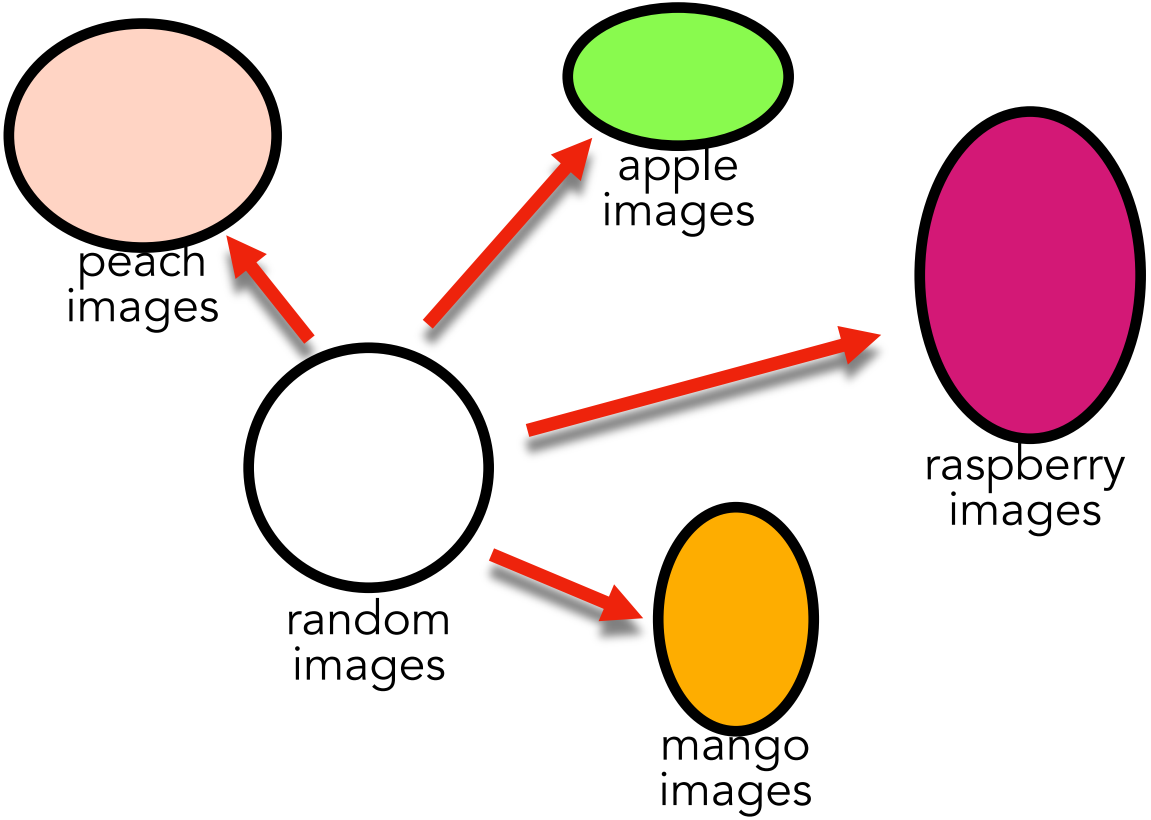

random
images
raspberry
images
diffusion
neural
network

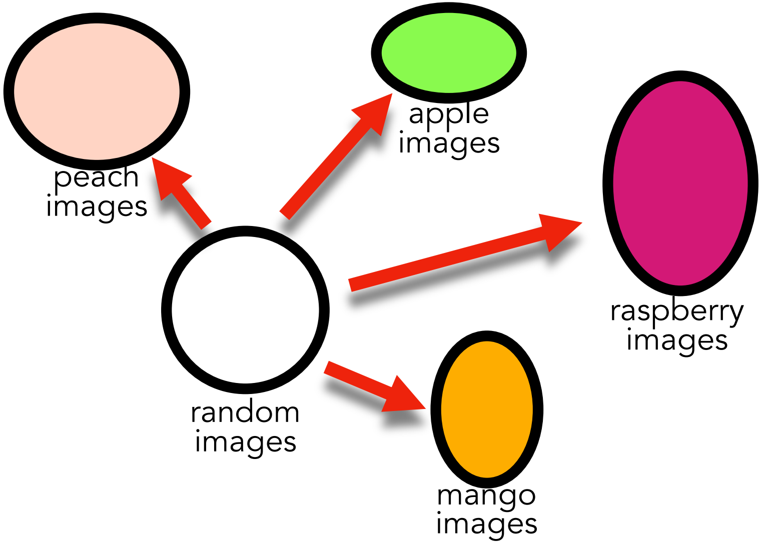

random
images
raspberry
images
diffusion
neural
network

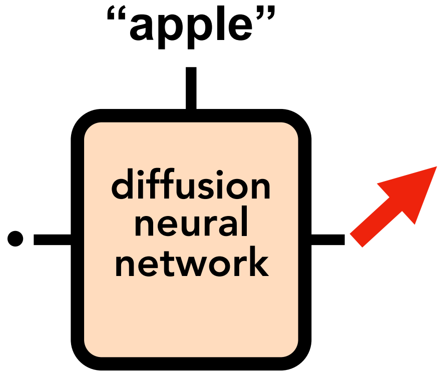

random images
raspberry images
diffusion neural network
Imagen

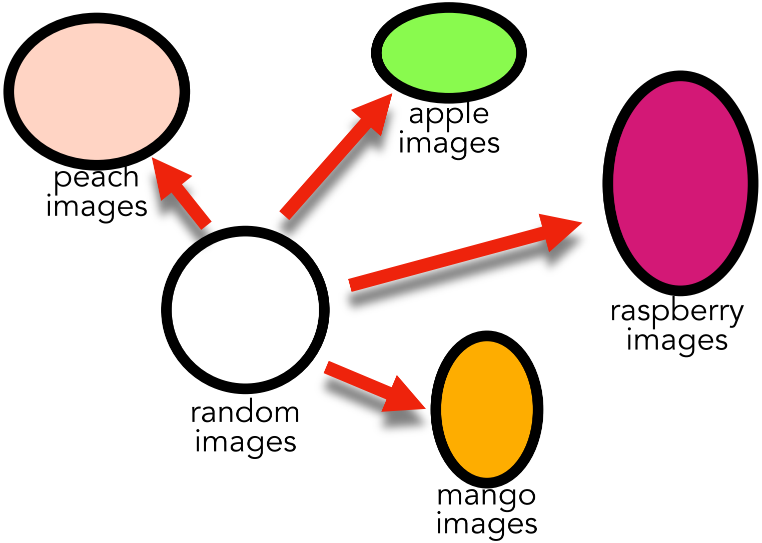

random images
raspberry images
diffusion neural network

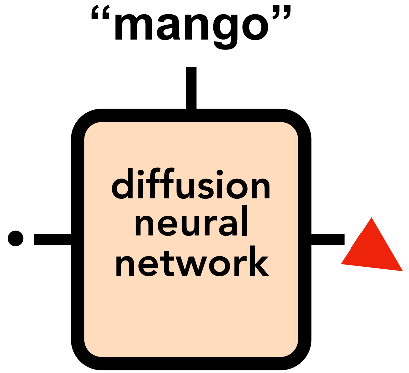

peach images
apple images
raspberry images
random images
mango images

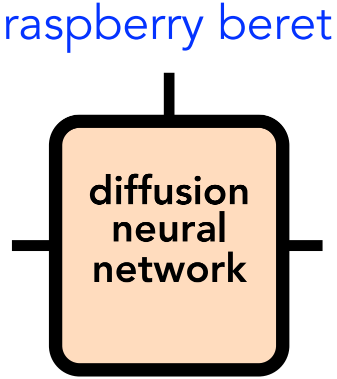

peach images
apple images
raspberry images
random images
mango images

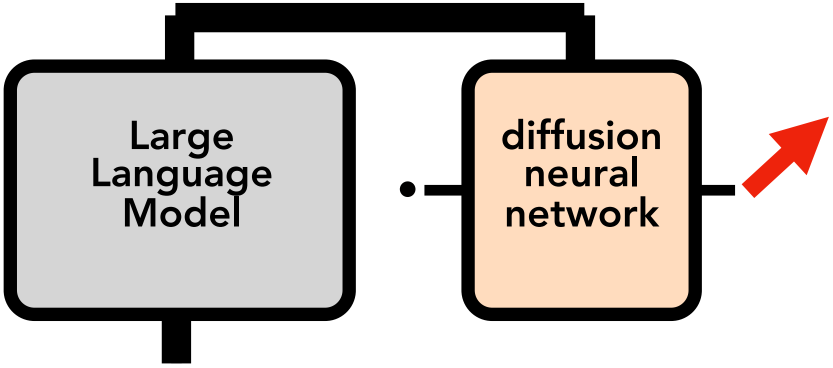

peach images
apple images
raspberry images
random images
mango images


"apple"
diffusion neural network

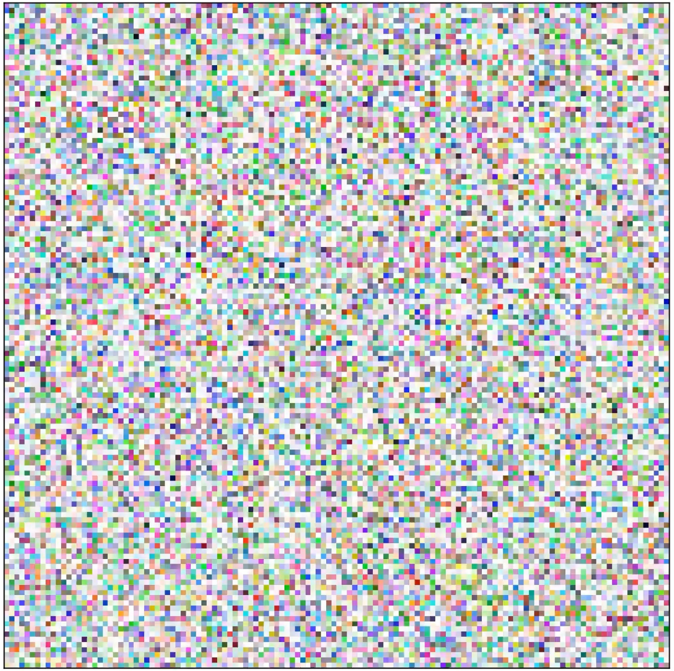

peach images
apple images
raspberry images
random images
mango images


"mango"
diffusion neural network


raspberry beret
diffusion neural network



raspberry beret



raspberry beret

raspberry beret

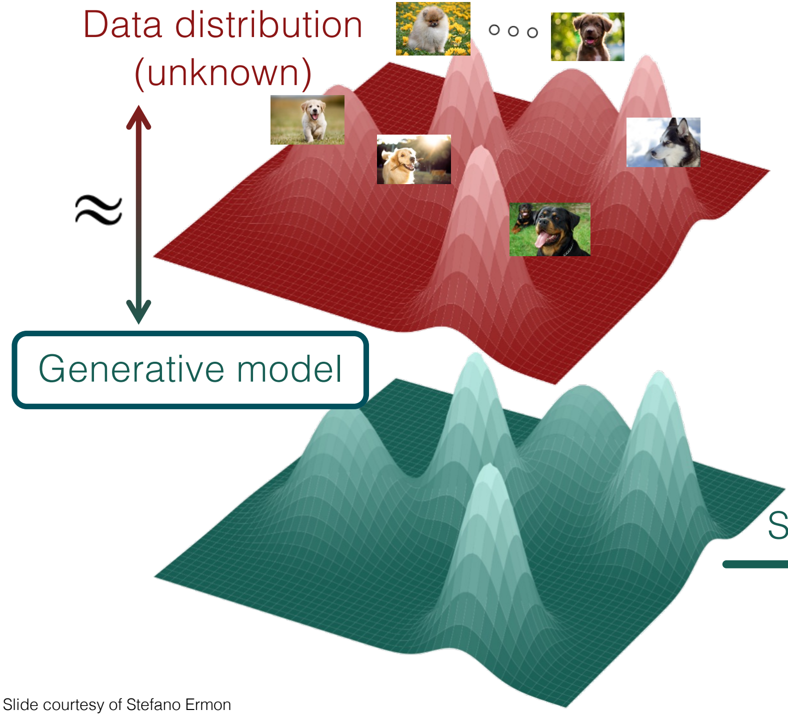
beret of raspberries

beret of raspberries

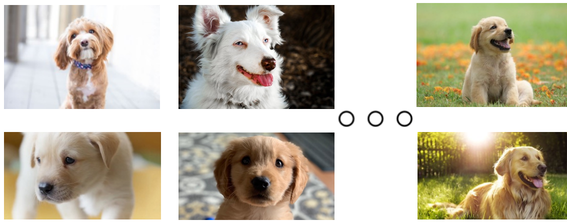
beret of raspberries

Score-based Generative Modeling

Estimating the probability distribution of data



Novel data points



Slide courtesy of Stefano Ermon

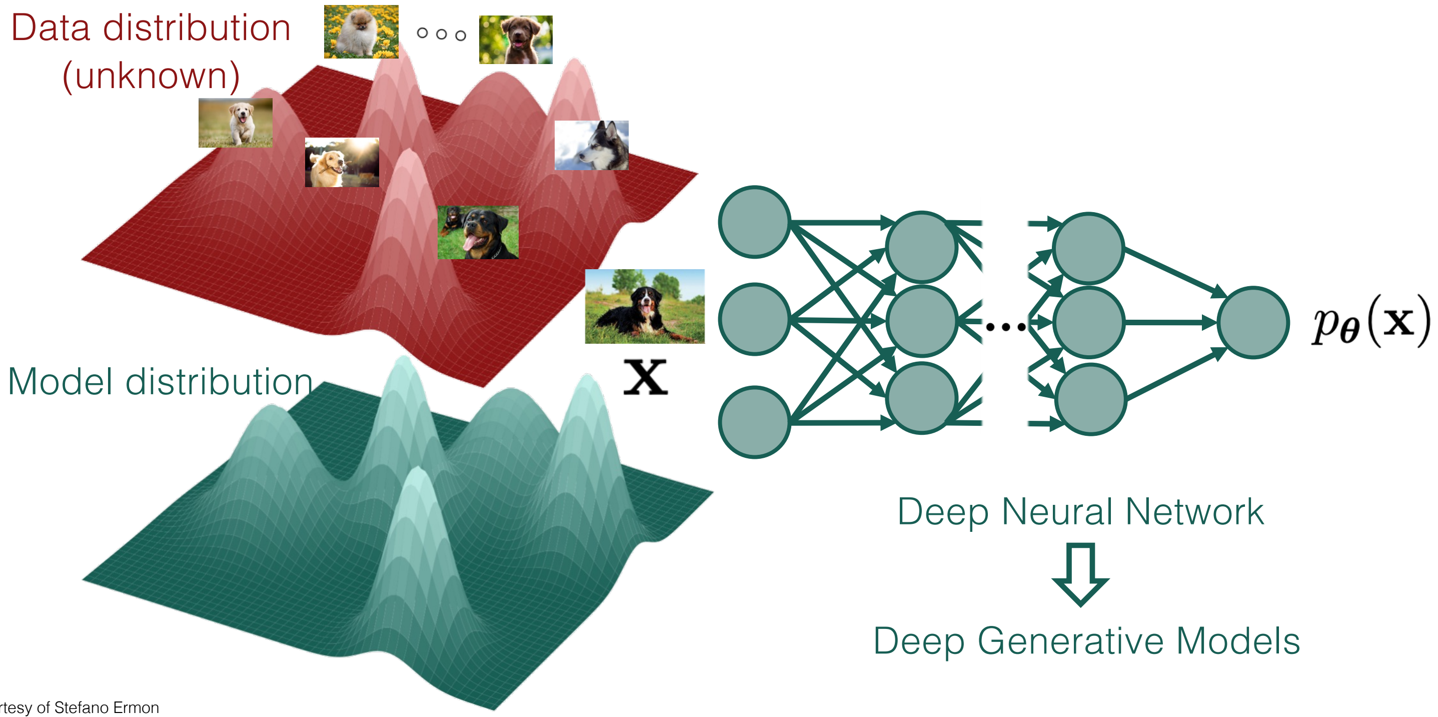
Estimating the probability distribution of data

Data distribution
(unknown)

Data distribution is extremely complex for high dimensional data.

How to build a complex model to fit the data distribution?

Estimating the probability distribution of data



Data distribution (unknown)
Model distribution
$\mathbf{x}$
$p_{\boldsymbol{\theta}}(\mathbf{x})$

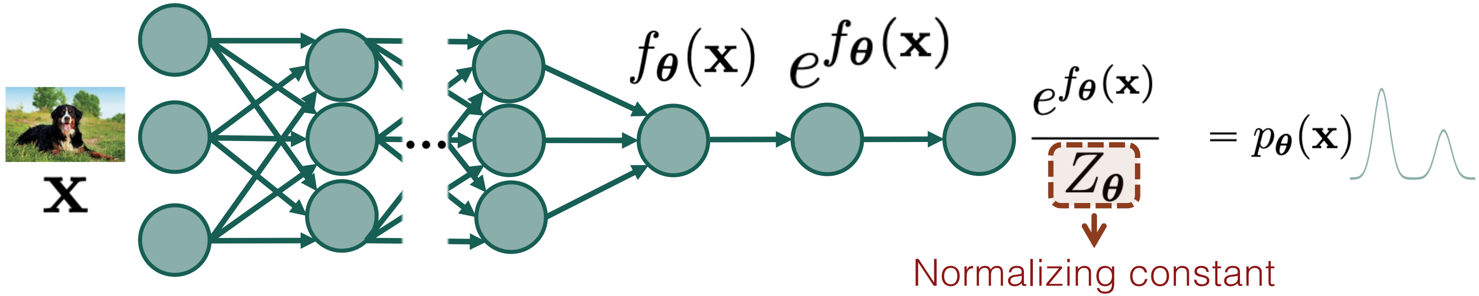
Deep Neural Network

⇩

Deep Generative Models

Slide courtesy of Stefano Ermon

Key Challenge: Normalization Constant



$\mathbf{x}$
$f_{\boldsymbol{\theta}}(\mathbf{x})$
$e^{f_{\boldsymbol{\theta}}(\mathbf{x})}$
$\frac{e^{f_{\boldsymbol{\theta}}(\mathbf{x})}}{Z_{\boldsymbol{\theta}}} = p_{\boldsymbol{\theta}}(\mathbf{x})$
Normalizing constant

Can enforce positivity, but computing normalizing constant is intractable!

$$Z_{\boldsymbol{\theta}} = \int e^{f_{\boldsymbol{\theta}}(\mathbf{x})}\, \mathrm{d}\mathbf{x}$$

Slide courtesy of Stefano Ermon

Probability Density vs. Score Function

Probability density function

$$p(x) = \frac{e^{f_\theta}}{\cancel{Z_\theta}}$$

$$\nabla_x \log p(x) = \nabla_x f_\theta\ (x) - \underbrace{\boxed{\nabla_x \log Z_\theta}}_{0}$$

Score function

Probability Density vs. Score Function

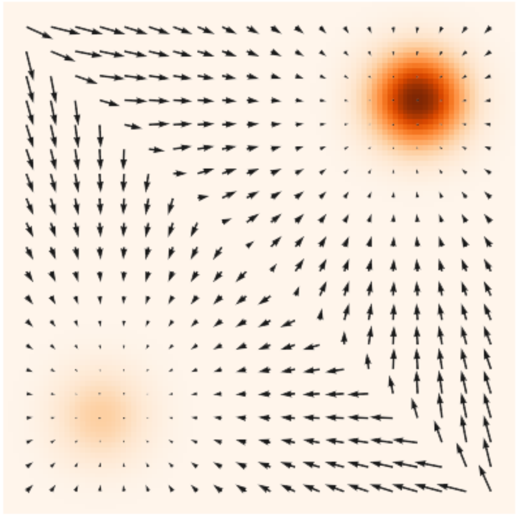
Probability density function

$$p(x) = \frac{e^{f_\theta}}{\cancel{Z_\theta}}$$

$$\Updownarrow$$

$$\nabla_x \log p(x) = \nabla_x f_\theta\ (x) - \underbrace{\boxed{\nabla_x \log Z_\theta}}_{0}$$

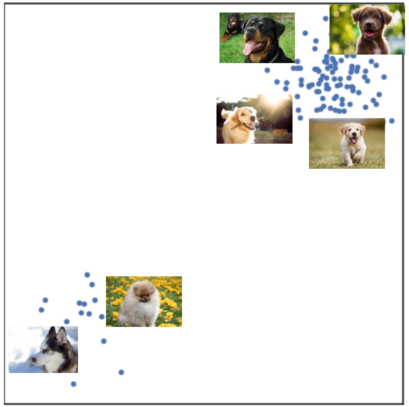
Score function


Score vs. density function

Slide courtesy of Stefano Ermon

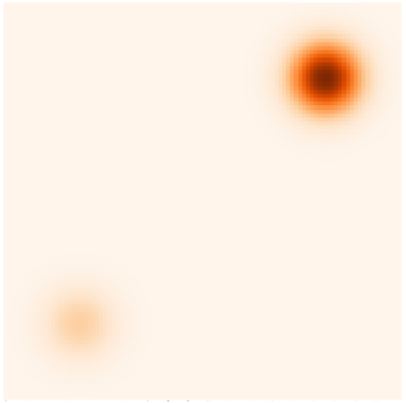
Learning the Score Function

Estimating Score Function from Data



Training data

$$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$$



Probability density function

Score function

$$p_\theta(\mathbf{x}) \approx p_{\text{data}}(\mathbf{x})$$

$$\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$$

Slide courtesy of Stefano Ermon

Tweedie's Formula

- Famous result in statistics!
- For an additive Gaussian corruption $\underbrace{x_t}_{\text{noisy image}} = \underbrace{x_0}_{\text{clean image}} + \underbrace{\sigma_t z}_{\text{noise } \epsilon_t},\ z \sim \mathcal{N}(0, I)$
- Tweedie: $$\mathbb{E}[x_0 | x_t] = x_t + \sigma_t^2 \, \nabla_{x_t} \log p_t(x_t)$$
- Thus, score function is $$\nabla_{x_t} \log p_t(x_t) = \frac{\mathbb{E}[x_0 | x_t] - x_t}{\sigma_t^2}$$

B. Efron, "Tweedie's formula and selection bias", Journal of the American Statistical Association, 2011

Denoising Score Matching

$$\nabla_{x_t} \log p_t(x_t) = \frac{\mathbb{E}[x_0 | x_t] - x_t}{\sigma_t^2} = s_\varphi(x_t; \sigma_t) \approx \frac{D_\theta(x_t; \sigma_t) - x_t}{\sigma_t^2} = -\frac{\epsilon_\phi(x_t; \sigma_t)}{\sigma_t^2}$$

score estimation network

denoising network

noise estimation network

- Minimize objective $\mathbb{E}_{x,t}[\|D_\theta(x_t; \sigma_t) - x\|_2^2]$

  or alternatively $\mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),\sigma_t}\left[\left\|\epsilon_\phi\left(x_{\sigma_t}; \sigma_t\right) - \epsilon\right\|_2^2\right]$

- This is the training objective of diffusion model!

  Note: exact loss formulation can be a bit more complicated, see e.g. Song et al., Appendix E.1

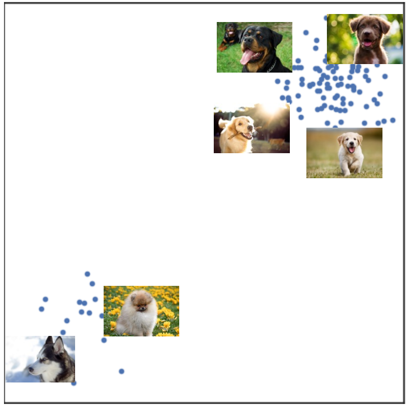
P. Vincent, "A connection between score matching and denoising autoencoders", Neural computation 2011; Song et al., ICLR 2021

The Fine Print

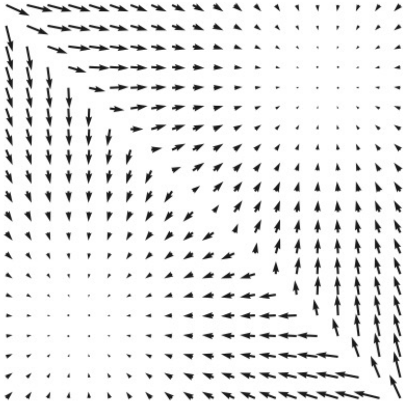
- I just showed you Tweedie’s formula and denoising score matching for the variance exploding (VE) formulation because it’s a bit simpler
- There is an alternative, variance-preserving (VP) formulation, which looks slightly different
- Attaching “Cheat Sheet” at the end of the slide deck with all relevant formulations

Diffusion Model Sampling

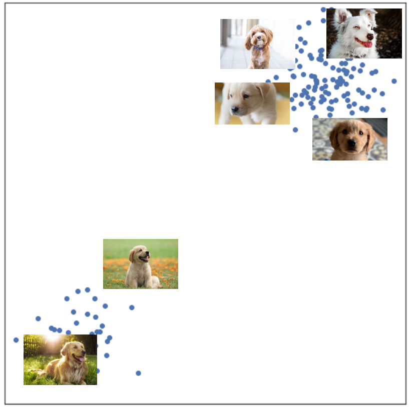
How to Generate New Data?



Data examples



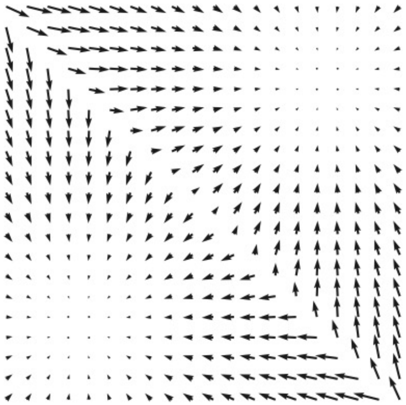
Score function



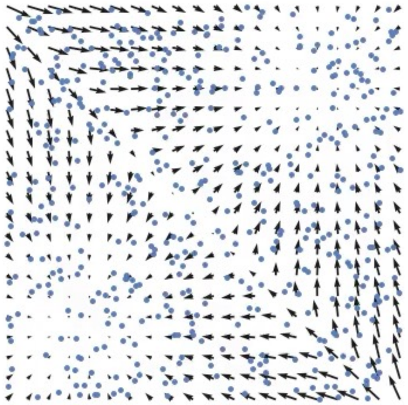
New data

Slide courtesy of Stefano Ermon

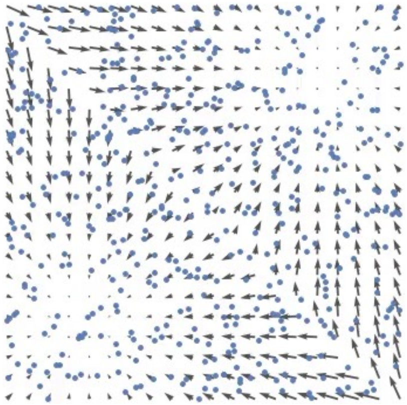
Langevin Dynamics



Score function



Follow the scores



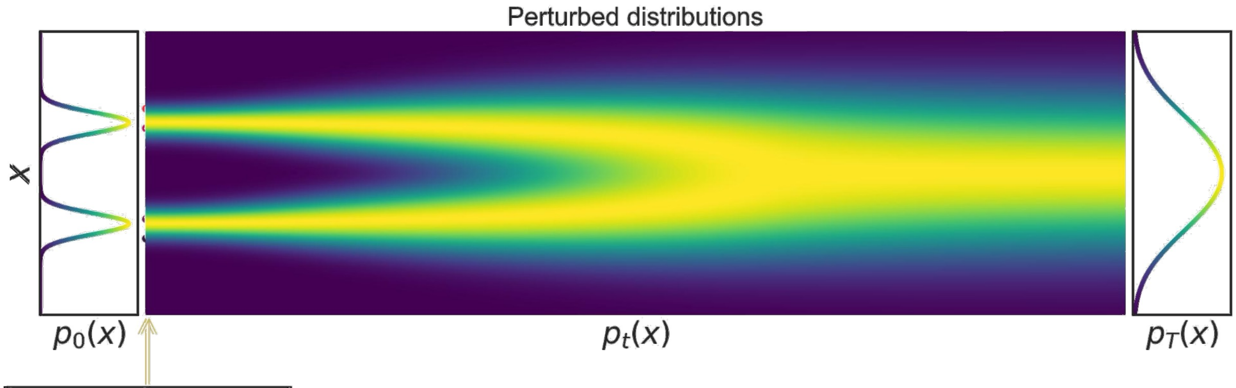
Follow the noisy scores

Langevin dynamics [Parisi 1981]
[Grenander and Miller, 1994]

✓ Correct samples guaranteed

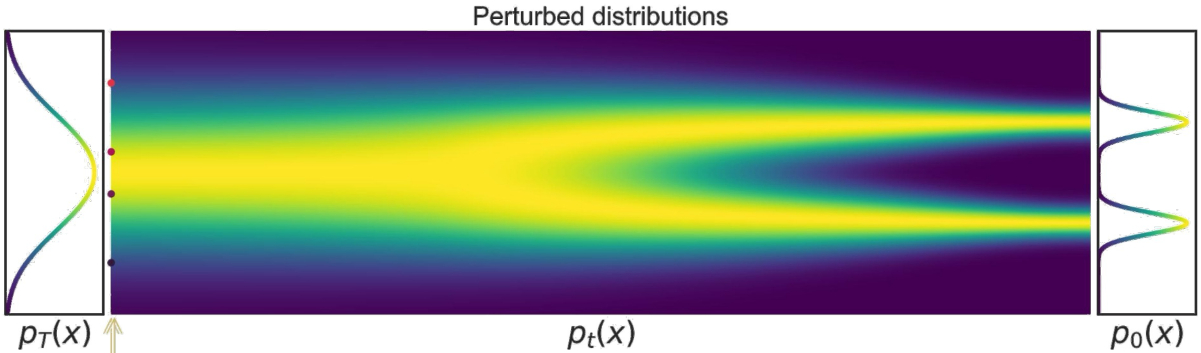
Slide courtesy of Stefano Ermon

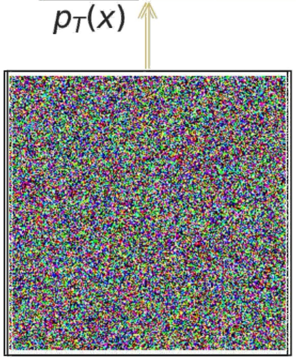
Langevin Dynamics: Forward Diffusion (t: 0→T)



Slide courtesy of Stefano Ermon

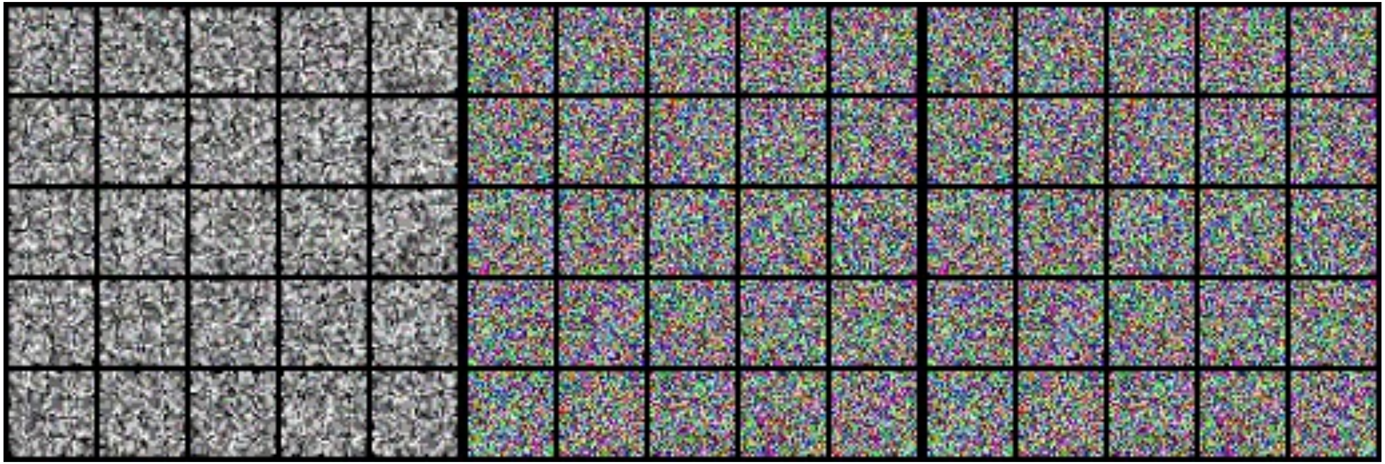
Langevin Dynamics: Reverse Diffusion (t: T→0)





Slide courtesy of Stefano Ermon

Langevin Dynamics: Reverse Diffusion (t: T→0)



Song and Ermon, NeurIPS 2022

(Continuous) Diffusion Math via SDE

- Forward diffusion can be described by Ito stochastic differential equation (SDE): $d\mathbf{x} = \underbrace{\mathbf{f}(\mathbf{x},t)}_{\text{drift coefficient}}\mathrm{d}t + \underbrace{g(t)}_{\text{diffusion coefficient}}\underbrace{d\mathbf{w}}_{\text{noise with variance } \mathrm{d}t}$

- Reverse diffusion can be described by Ito SDE:

$$d\mathbf{x} = [\mathbf{f}(\mathbf{x},t) - g^2(t)\nabla_x \log \underbrace{p_t}_{\text{distribution of noisy(!) images at step } t}(\mathbf{x})]\mathrm{d}t + g(t)\underbrace{d\tilde{\mathbf{w}}}_{\text{standard Wiener process, i.e., noise with variance } \mathrm{d}t}$$

Song et al., ICLR 2021

B. Anderson, "Reverse-time diffusion equation models", Stochastic Processes and their Applications, 1982

(Continuous) Diffusion Math via SDE

- Forward diffusion can be described by Ito stochastic differential equation (SDE):

$$\begin{tikzcd} \mathbf{x}_0 \sim p_0 = p_{data} \quad \text{(i.e., image from dataset)} \arrow[d, "{\text{solve } 0 \longrightarrow \mathrm{T}} \\ {d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)d\mathbf{w}}"] \\ \mathbf{x}_T \sim p_T = \mathcal{N} \end{tikzcd}$$

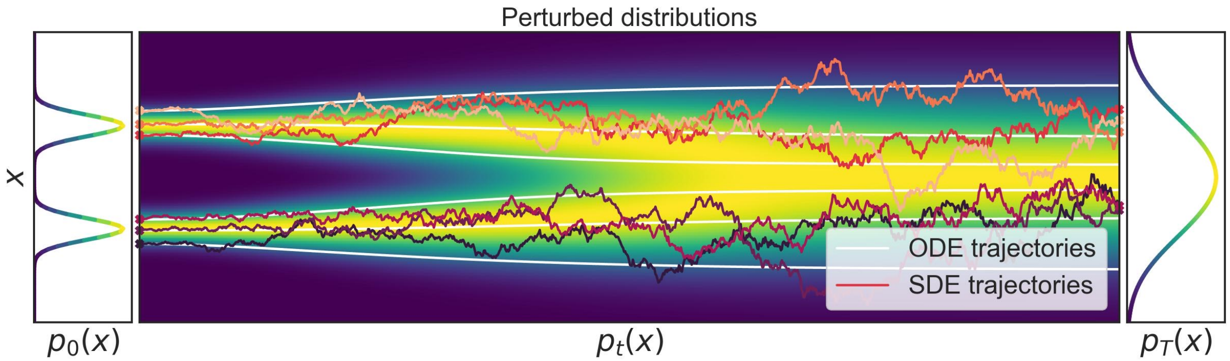
- Reverse diffusion can be described by Ito SDE:

$$\begin{tikzcd} \mathbf{x}_T \sim p_T = \mathcal{N} \arrow[d, "{\text{solve } \mathrm{T} \longrightarrow 0} \\ {d\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_x \log p_t(\mathbf{x})]\mathrm{d}t + g(t)d\tilde{\mathbf{w}}}"] \\ \mathbf{x}_0 \sim p_0 = p_{data} \end{tikzcd}$$

Song et al., ICLR 2021

B. Anderson, "Reverse-time diffusion equation models", Stochastic Processes and their Applications, 1982

SDE vs. ODE



SDE
(stochastic differential equation)

$\Longleftrightarrow$

Probability Flow ODE
(ordinary differential equation)

$$d\mathbf{x} = -\sigma^2(t)\nabla_x \log p_t(x)\,\mathrm{d}t + \sigma(t)d\tilde{\mathbf{w}}$$

$$d\mathbf{x} = -\sigma^2(t)\nabla_x \log p_t(x)\,\mathrm{d}t$$

Song et al., ICLR 2021

Diffusion Model Architectures

Text-to-image Generation Models

OpenAI's DALL-E 1,2,3

Google's Imagen

A chromeplated cat sculpture placed on a Persian rug.

Android Mascot made from bamboo.

Intricate origami of a fox and a unicorn in a snowy forest.

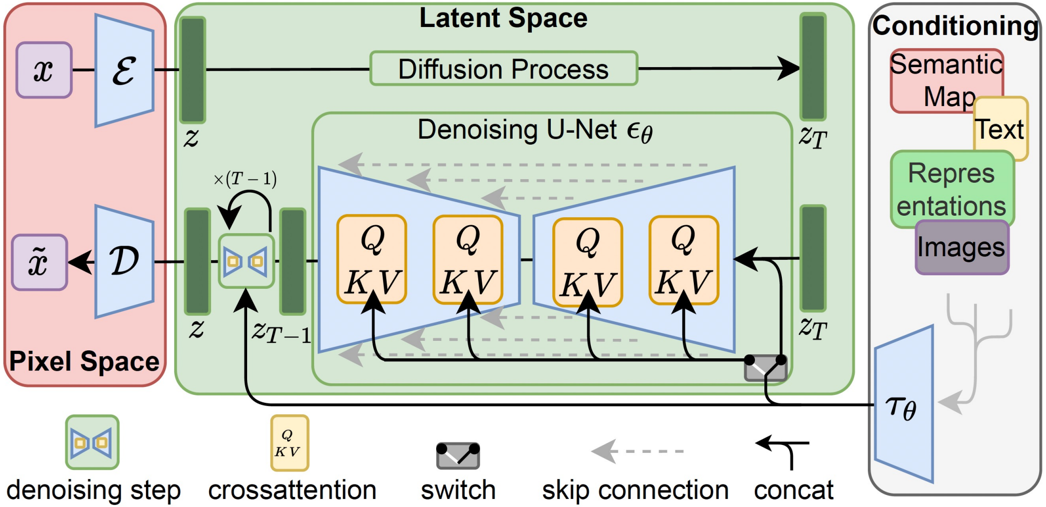
Midjourney

Adobe Firefly

… many more, but many of the models are proprietary …

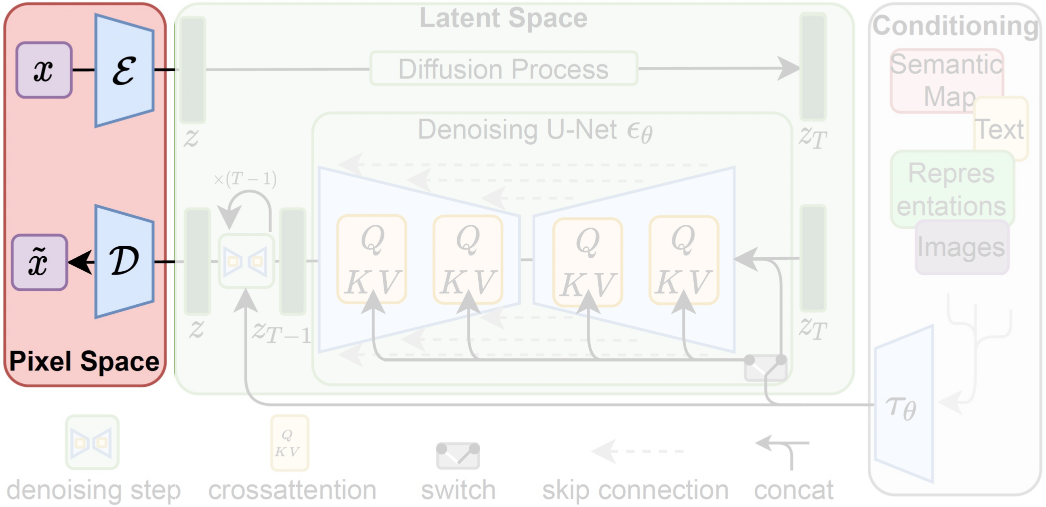
Latent Diffusion Models

1. Variational Autoencoder (VAE) based encoder $\mathcal{E}$ & decoder $\mathcal{D}$
2. Denoising U-Net ϵ_θ with self attention
3. Conditioning via cross attention



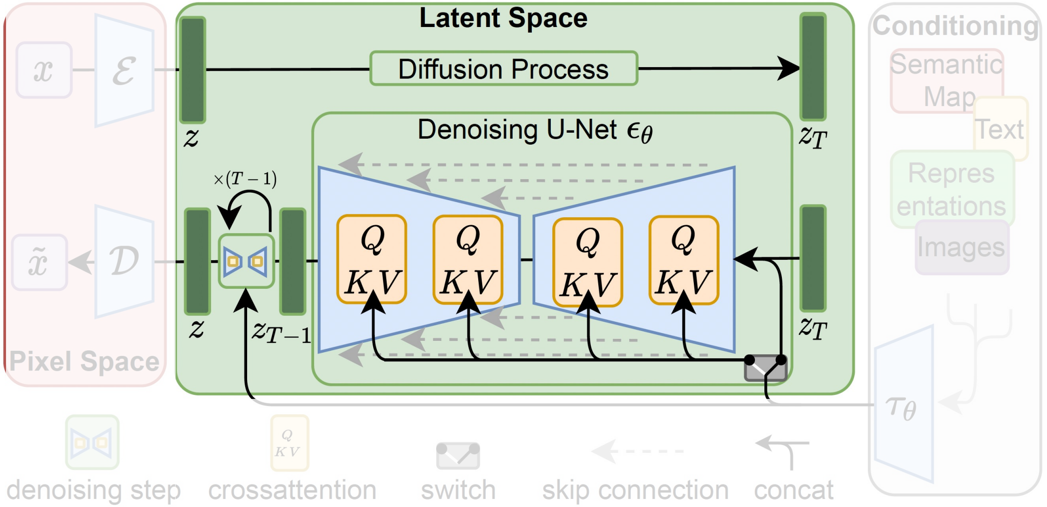
Latent Diffusion Models

1. Variational Autoencoder (VAE) based encoder $\mathcal{E}$ & decoder $\mathcal{D}$
2. Denoising U-Net ϵ_θ with self attention
3. Conditioning via cross attention



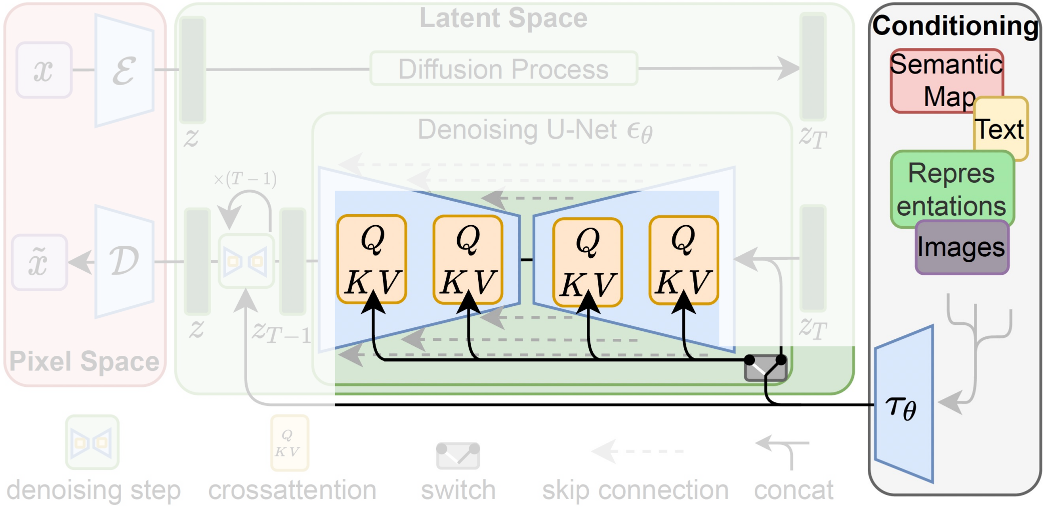
Latent Diffusion Models

1. Variational Autoencoder (VAE) based encoder $\mathcal{E}$ & decoder $\mathcal{D}$
2. Denoising U-Net ϵ_θ with self attention
3. Conditioning via cross attention



Latent Diffusion Models

1. Variational Autoencoder (VAE) based encoder $\mathcal{E}$ & decoder $\mathcal{D}$
2. Denoising U-Net ϵ_θ with self attention
3. Conditioning via cross attention



Stable Diffusion Versions

- SD 1.5: very common model, lightweight (860M parameters), lots of community support in the form of LoRAs, etc.
- SD XL: same idea, just larger U-Net, cross-attention context, etc.
- SD 3.0: no U-Net but Transformer-based backbone

- Training data: subset of LAION-5B text-image dataset

Stable Video Diffusion & AnimateDiff

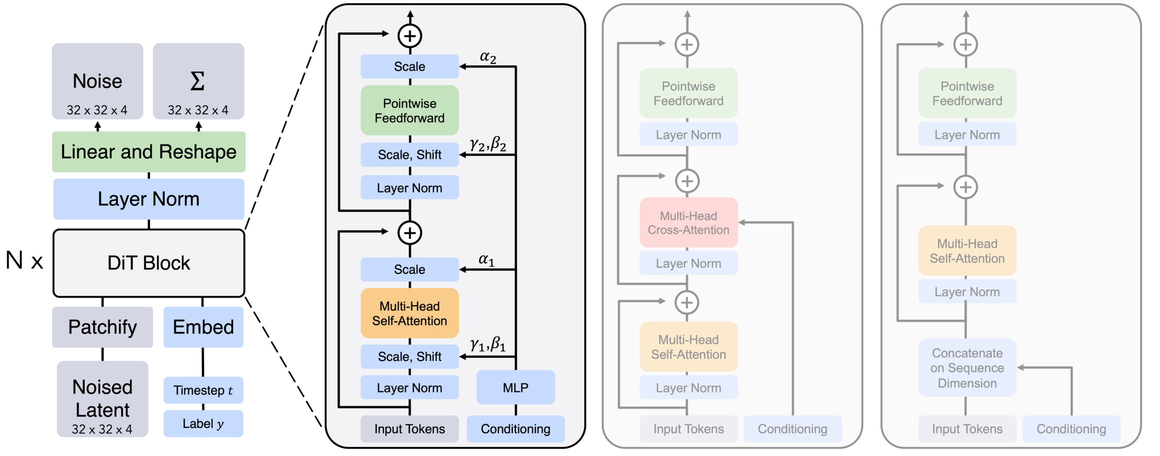
- **SVD**: image-to-video generation model by Stability AI
- **AnimateDiff**: adapter network for text-to-video generation

Both (and other video models) use (learned) temporal attention; only a few-second clips, about 30 frames total

Blattmann et al., "Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets", arxiv 2023
Guo et al., "AnimateDiff: Animate Your Personalized Text-to-Image Diffusion Models without Specific Tuning", ICLR 2024

Diffusion Transformers

- Replace U-Net with vision transformer (need to add conditioning)
- Turns out to be more scalable than U-Net!



Latent Diffusion Transformer — DiT Block with adaLN-Zero — DiT Block with Cross-Attention — DiT Block with In-Context Conditioning

Peebles and Xie, “Scalable Diffusion Models with Transformers”, ICCV 2023

OpenAI, 2024

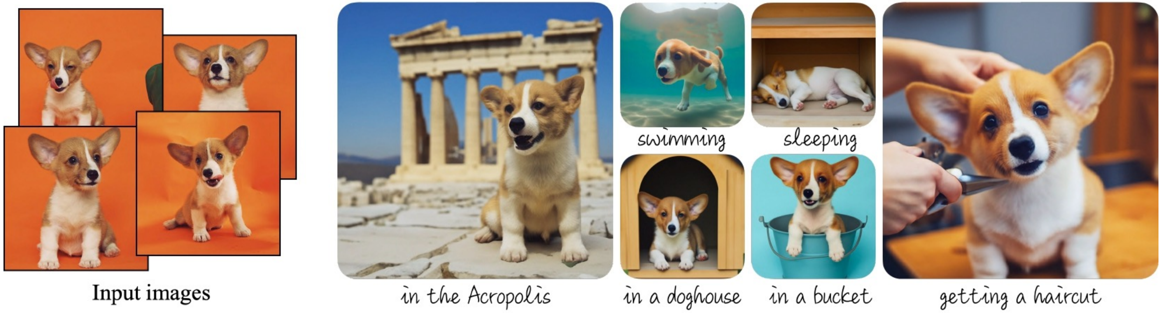
Prompt: A stylish woman walks down a Tokyo street filled with warm glowing neon and animated city signage. She wears a black leather jacket, a long red dress, and black boots, and carries a black purse. She wears sunglasses and red lipstick. She walks confidently and casually. The street is damp and reflective, creating a mirror effect of the colorful lights. Many pedestrians walk about.

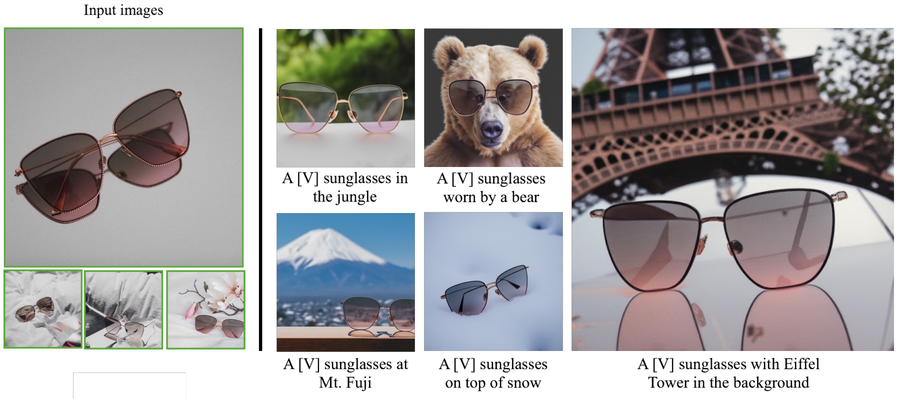
Probably state of the art: Google Nano Banana

Outlook

(i.e., things we didn’t have time to discuss)

Customization/Personalization




Input images
A [V] sunglasses in the jungle
A [V] sunglasses worn by a bear
A [V] sunglasses at Mt. Fuji
A [V] sunglasses on top of snow
A [V] sunglasses with Eiffel Tower in the background

- Add novel concepts to base model using few examples

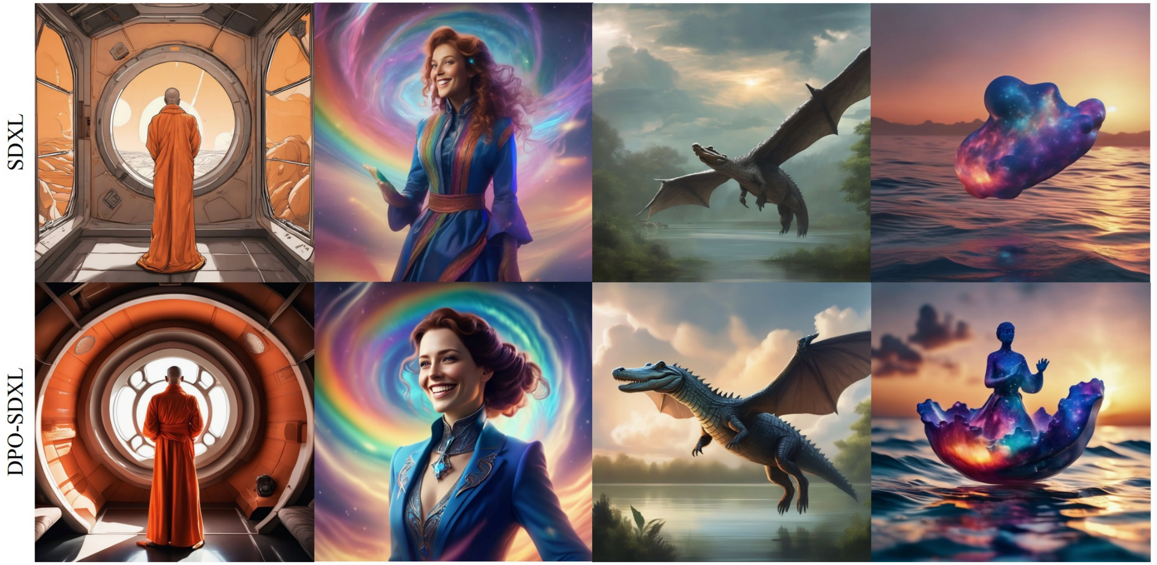
Ruiz et al., CVPR 2023

Customization/Personalization

Models fine-tuned for a single concept

Po et al., CVPR 2024

Reinforcement Learning from Human Feedback

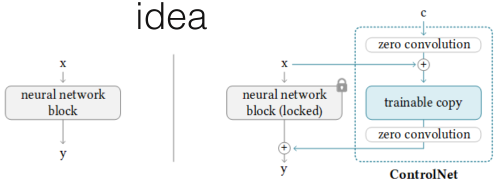


- Align generated images with preferences of humans via reinforcement learning from human feedback / direct preference optimization

Wallace et al., CVPR 2024

Control: ControlNet

idea



(a) Before (b) After

sketch 2 image

Cartoon line drawing “1girl, masterpiece, best quality, ultra-detailed, illustration”

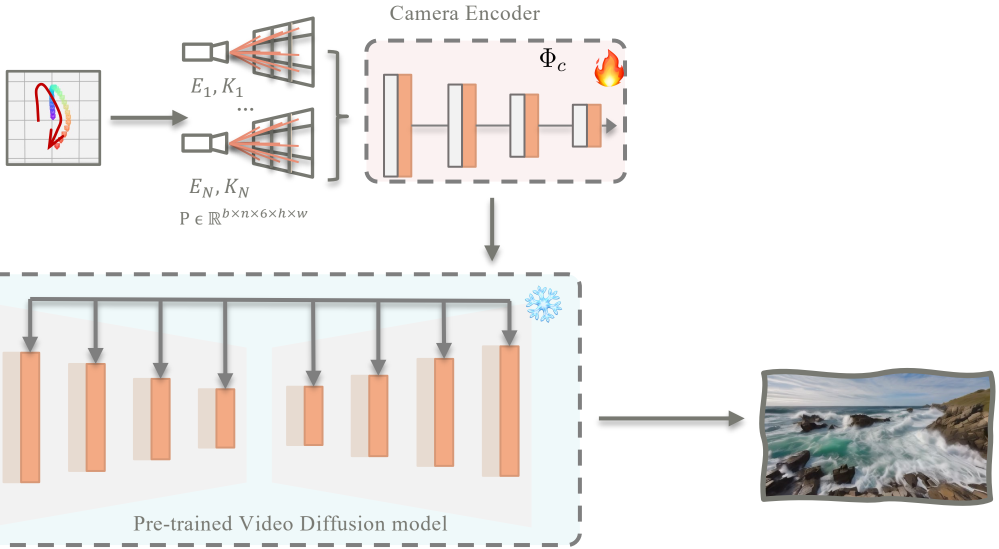
pose 2 image

semantic labels 2 image

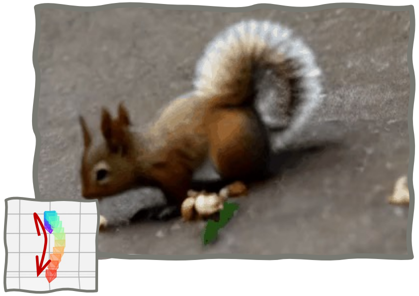
depth 2 image

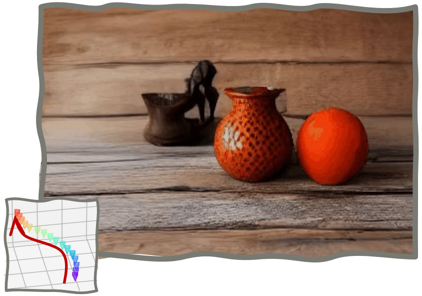
Zhang et al., ControlNet, ICCV 2023

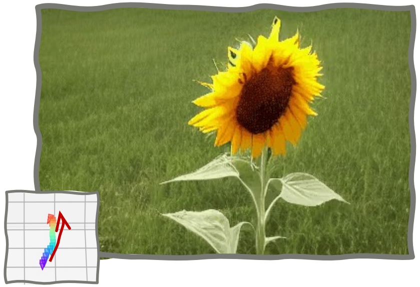
Control: CameraCtrl

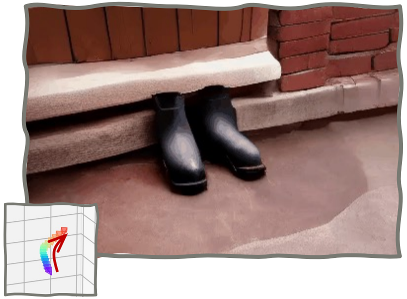

Camera Encoder
Φ_c
E_1, K_1
...
E_N, K_N
$P \in \mathbb{R}^{b \times n \times 6 \times h \times w}$
Pre-trained Video Diffusion model

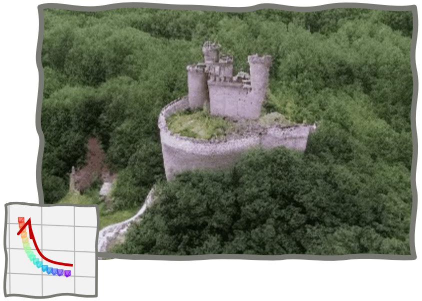
Control: CameraCtrl – text2video

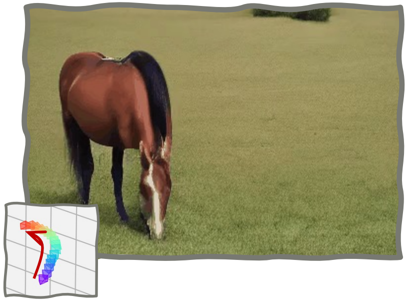

A squirrel is eating pine nuts.


A still life of vintage objects on a wooden table.

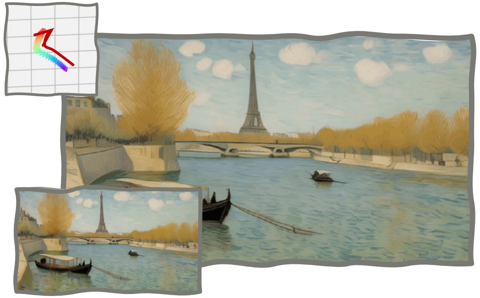

The sunflower in the sun.


A pair of worn leather boots on a porch.

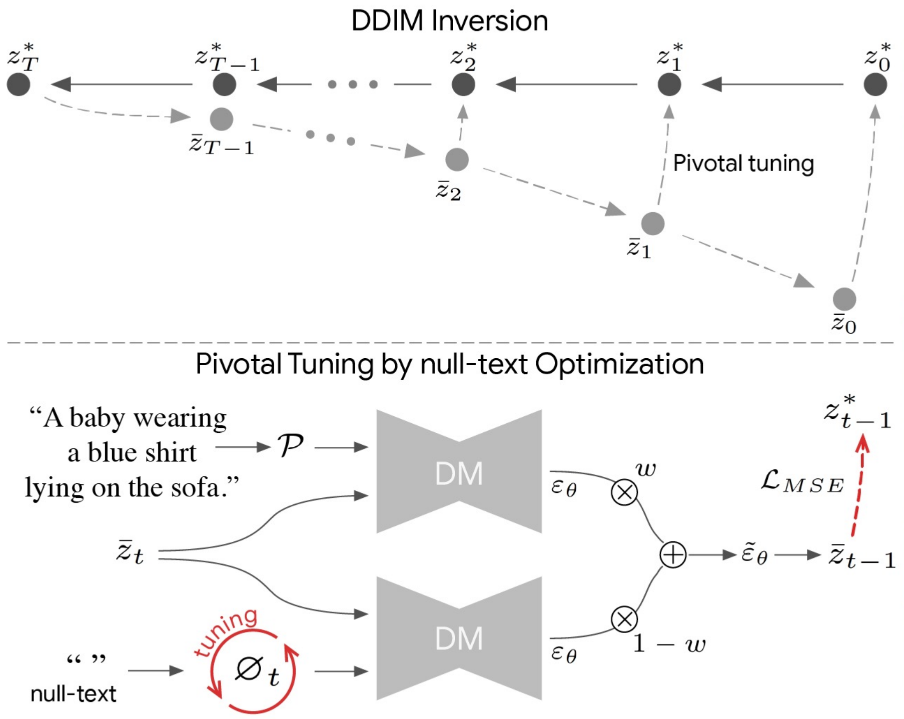

A castle in the forest.

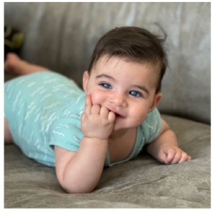

A horse is eating grass on the grassland.

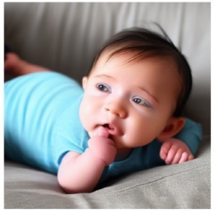
Control: CameraCtrl – image2video

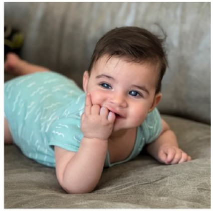


Inversion



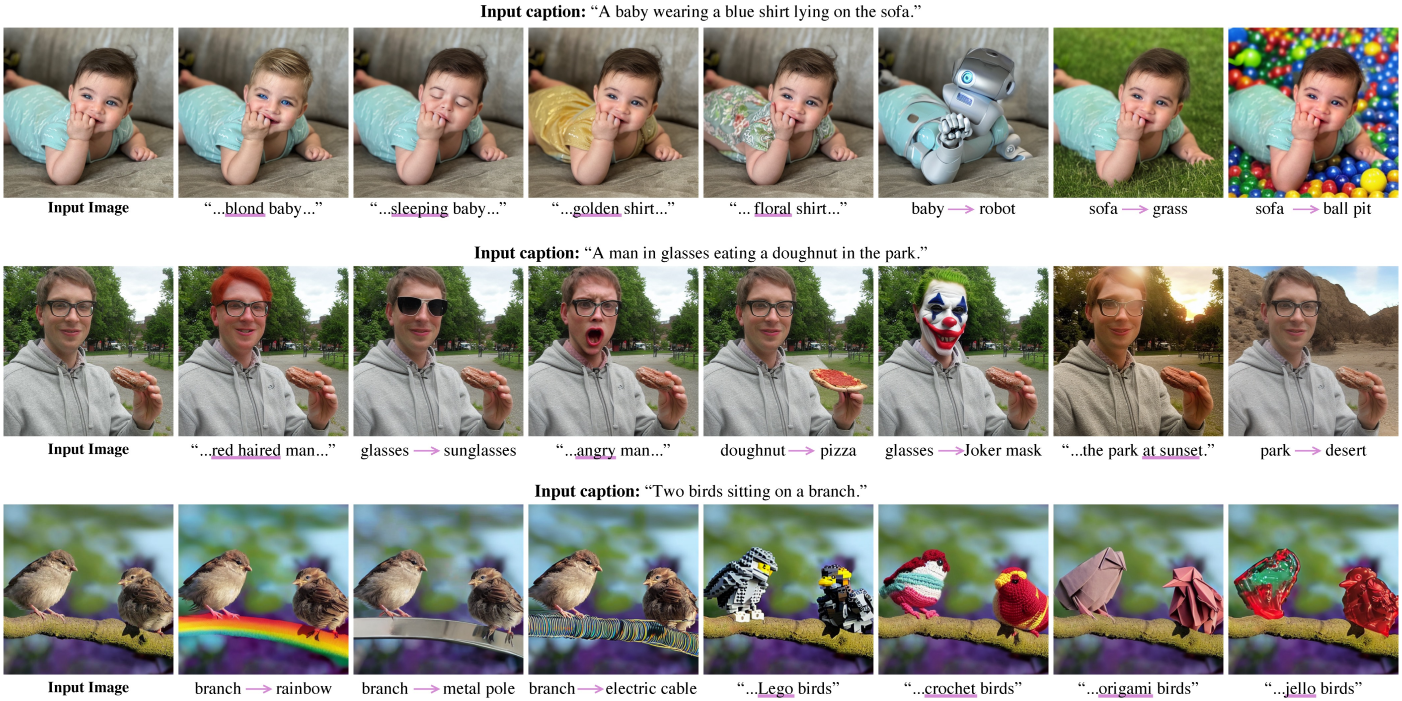

Input Image


Initial Inversion


Final Inversion

Mokady et al., Null-text Inversion, CVPR 2023

Inversion

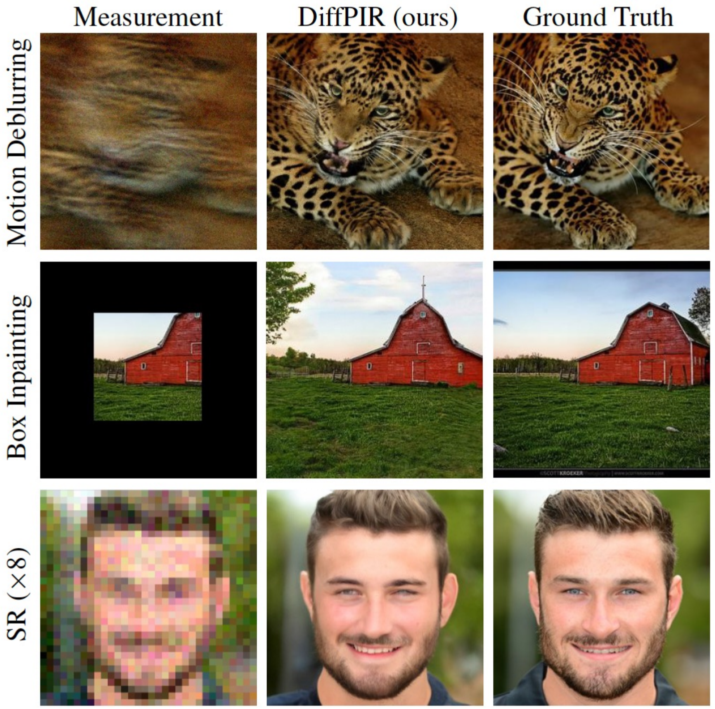


Mokady et al., Null-text Inversion, CVPR 2023

Key Takeaways from this Lecture

- Learning data distribution from examples is difficult
- SDE formulation gives us a way to go between data and Gaussian noise distribution; need score function to go from noise to data
- Tweedie's formula and denoising score matching gives us a robust way to estimate score function
- Latent diffusion models provide computational efficiency
- Diffusion transformer architectures scale well to internet-sized datasets, such as images or videos

Next: Solving Inverse Problems with Diffusion Model-based Priors



References and Further Reading

Diffusion Models – Theory & Applications:

- Y. Song, S. Ermon, "Generative modeling by estimating gradients of the data distribution", NeurIPS 2019
- J. Ho, A. Jain, P. Abbeel, "Denoising Diffusion Probabilistic Models", NeurIPS 2020 (DDPM)
- J. Song, C. Meng, S. Ermon, "Denoising diffusion implicit models", ICLR, 2021 (DDIM)
- Y. Song, J. Sohl-Dickstein, D. Kingma, A. Kumar, S. Ermon, B. Poole, "Score-based generative modeling through stochastic differential equations", ICLR 2021
- T. Karras, M. Aittala, T. Aila, S. Laine, "Elucidating the Design Space of Diffusion-Based Generative Models", NeurIPS 2022
- C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.Y. Zhu, S. Ermon, "SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations", ICLR 2022
- R. Po, W. Yifan, V. Golyanik, K. Aberman, J. Barron, A. Bermano, E. Chan, T. Dekel, A. Holynski, A. Kanazawa, K. Liu, L. Liu, B. Mildenhall, M. Nießner, B. Ommer, C. Theobalt, P. Wonka, G. Wetzstein, "State of the art on diffusion models for visual computing", Computer Graphics Forum (Eurographics State-of-the-art Report), 2024
- Sander Dieleman, "Guidance: a cheat code for diffusion models", https://sander.ai/2022/05/26/guidance.html

For an overview of all the methods shown in the "outlook" section and more, please take a look at the survey paper by Po et al. above.

Cheat Sheet

- Forward diffusion / noise model:
  - VP: $x_t^{(\mathrm{VP})} = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1-\bar{\alpha}_t} \cdot \epsilon,\ \epsilon \sim \mathcal{N}(0, \mathrm{I})$
  - VE: $x_t^{(\mathrm{VE})} = x_0 + \sigma_t \cdot \epsilon,\ \epsilon \sim \mathcal{N}(0, \mathrm{I})$
- Tweedie's formula:
  - VP: $\hat{x}_0^{(\mathrm{VP})} = \mathbb{E}[\mathrm{x}_0|\mathrm{x}_t] = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t + (1-\bar{\alpha}_t)\nabla_{x_t} \log p_t(x_t)\right)$
  - VE: $\hat{x}_0^{(\mathrm{VE})} = \mathbb{E}[x_0|x_t] = x_t + \sigma_t^2\, \nabla_{x_t} \log p_t(x_t)$
  - General: $\hat{x}_0 = \mathbb{E}[x_0|x_t] = \left(x_t + b_t^2 \nabla_{x_t} \log p_t(x_t)\right)/a_t$, for $x_t = a_t \cdot x_0 + b_t \cdot \epsilon,\ \epsilon \sim \mathcal{N}(0, \mathrm{I})$
- Score function:
  - VP: $\nabla_{x_t} \log p_t(x_t) = \frac{\sqrt{\bar{\alpha}_t}\mathbb{E}[x_0|x_t] - x_t}{1-\bar{\alpha}_t}$
  - VE: $\nabla_{x_t} \log p_t(x_t) = \frac{\mathbb{E}[x_0|x_t] - x_t}{\sigma_t^2}$

VR: variance-preserving formulation
VE: variance-exploding formulation

Cheat Sheet

- Relationship between score function, denoising network, and noise estimation network

  - VP:

$$\nabla_{x_t} \log p_t(x_t) = \frac{\sqrt{\bar{\alpha}_t}\mathbb{E}[x_0|x_t] - x_t}{1-\bar{\alpha}_t} = s_\varphi(x_t;\sigma_t) \approx \frac{\sqrt{\bar{\alpha}_t}D_\theta(x_t;\sigma_t) - x_t}{1-\bar{\alpha}_t} = -\frac{\epsilon_\phi(x_t;\sigma_t)}{\sqrt{1-\bar{\alpha}_t}}$$

  - VE:

$$\nabla_{x_t} \log p_t(x_t) = \frac{\mathbb{E}[x_0|x_t] - x_t}{\sigma_t^2} = \underset{\nearrow}{s_\varphi(x_t;\sigma_t)} \approx \frac{\underset{\uparrow}{D_\theta(x_t;\sigma_t)} - x_t}{\sigma_t^2} = -\frac{\underset{\nwarrow}{\epsilon_\phi(x_t;\sigma_t)}}{\sigma_t^2}$$

score
estimation network

denoising network

noise
estimation network