

COMP547

DEEP UNSUPERVISED LEARNING

Lecture #10 – Diffusion Models

**KOÇ
UNIVERSITY**

Aykut Erdem // Koç University // Spring 2026



Previously on COMP547

- Energy-based models
- Score-based Models

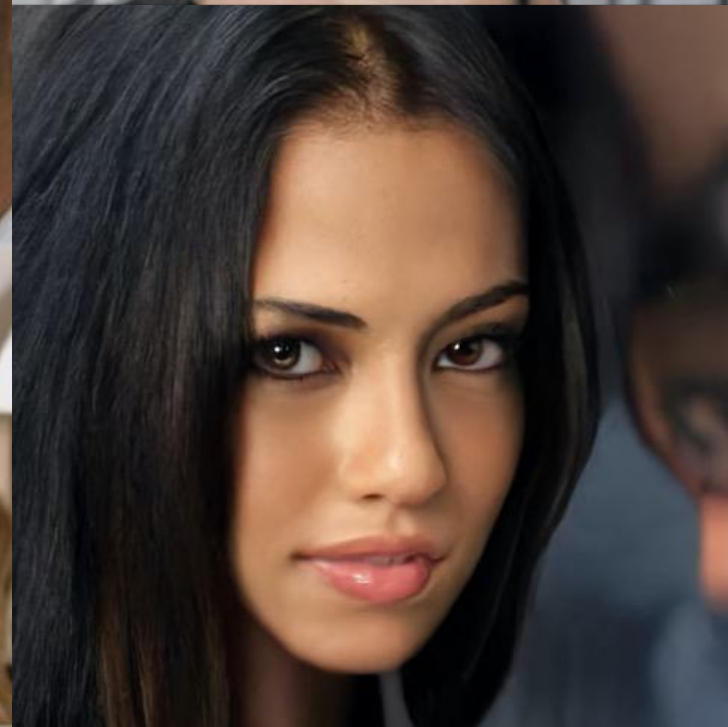


Image: Synthetic face images (Song et al., 2021)

Lecture overview

- Motivation
- Denoising Diffusion Models
- Deeper Dive into Diffusion Models

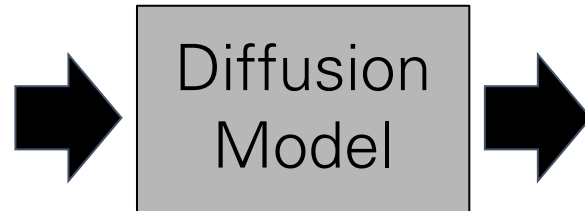
Disclaimer: Much of the material and slides for this lecture were borrowed from
—Jascha Sohl-Dickstein’s talk "Deep Unsupervised Learning using Nonequilibrium Thermodynamics"
—Sangwoo Mo’s talk titled “Introduction to Diffusion Models”
—Pieter Abbeel, Wilson Yan, Kevin Frans, Philipp Wu’s Berkeley CS294-158 class

Lecture overview

- Motivation
- Energy-based models
- Score-based Models
- Denoising Diffusion Models
- Deeper Dive into Diffusion Models

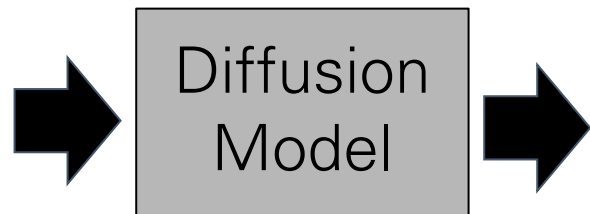
Motivation: Diffusion Models

a masterful oil painting
a persian exotic cat
discovering their
astounding crypto
losses while checking
their phone



Motivation: Diffusion Models

Beautiful, snowy Tokyo city is bustling. The camera moves through the bustling city street, following several people enjoying the beautiful snowy weather and shopping at nearby stalls. Gorgeous sakura petals are flying through the wind along with snowflakes.



Motivation: Diffusion Models

Reflections in the window of a train traveling through the Tokyo suburbs.

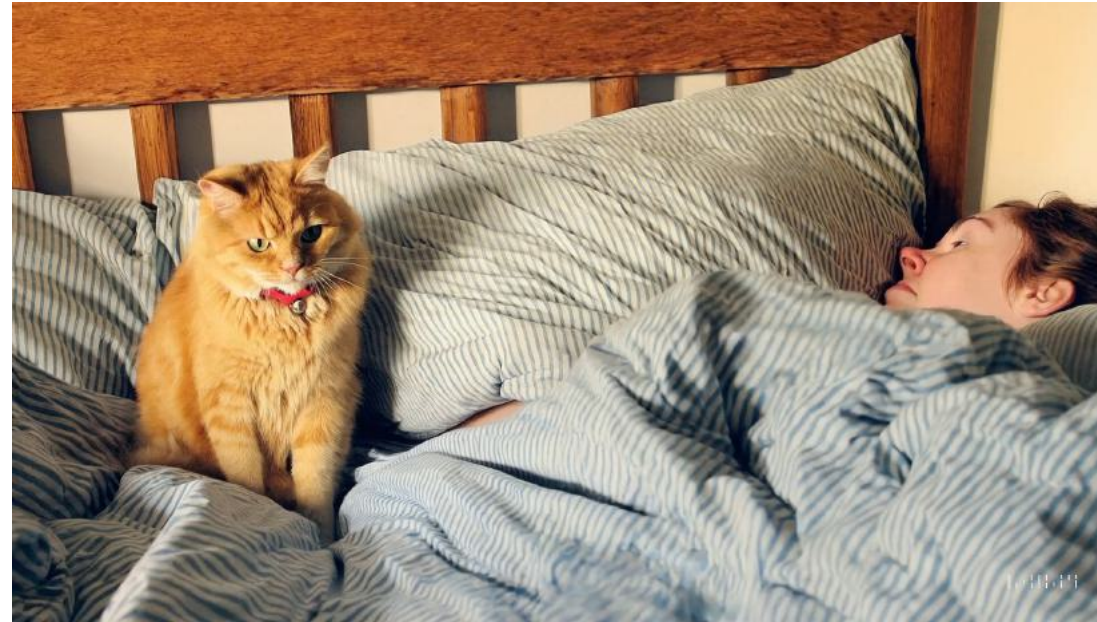
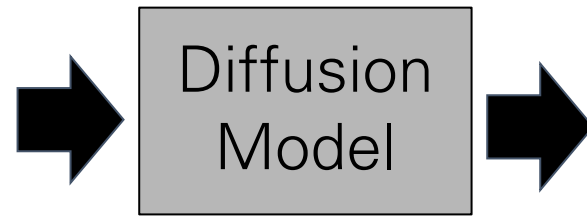


Diffusion Model



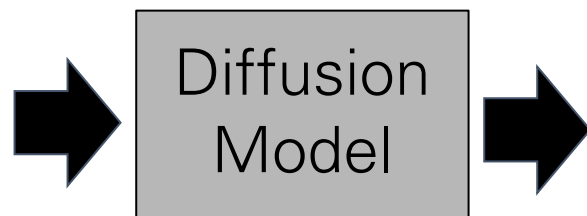
Motivation: Diffusion Models

A cat waking up its sleeping owner demanding breakfast. The owner tries to ignore the cat, but the cat tries new tactics and finally the owner pulls out a secret stash of treats from under the pillow to hold the cat off a little longer.



Motivation: Diffusion Models

Five gray wolf pups frolicking and chasing each other around a remote gravel road, surrounded by grass. The pups run and leap, chasing each other, and nipping at each other, playing. More Prompt: Five gray wolf pups frolicking and chasing each other around a remote gravel road, surrounded by grass. The pups run and leap, chasing each other, and nipping at each other, playing.



Lecture overview

- Motivation
- **Denoising Diffusion Models**
 - Overview
 - Denoising Diffusion Probabilistic Models
 - Diffusion Models Beat GANs on Image Synthesis; Classifier Guidance
 - Classifier-Free Diffusion Guidance
 - GLIDE: Text-Guided Diffusion Models
 - Dall-E: 1, 2, 3
 - Imagen
- Deeper Dive into Diffusion Models

Observation 1: Diffusion Destroys Structure



- Dye density represents probability density
- **Goal:** Learn structure probability density
- **Observation:** Diffusion destroys structure

Data distribution



Uniform distribution

Idea: Recover Structure by Reversing Time



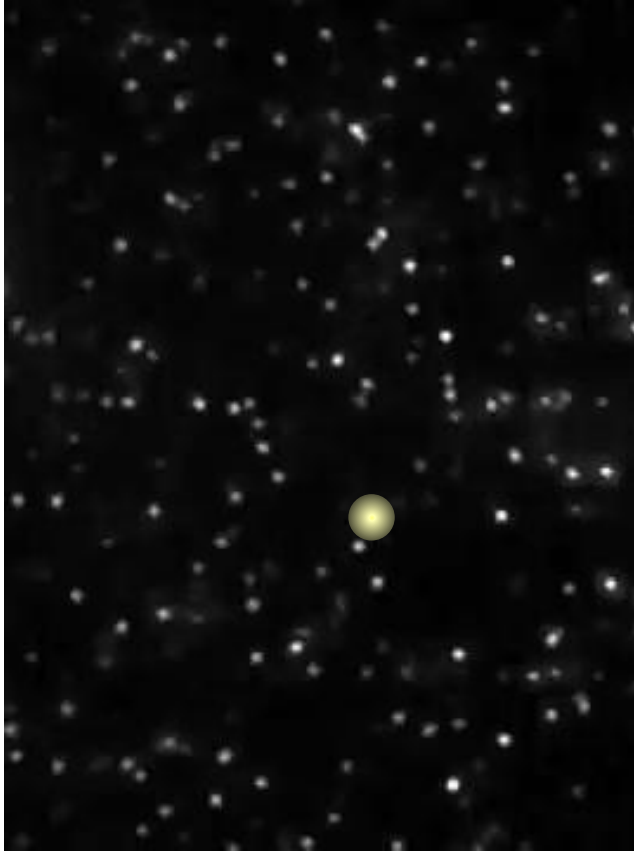
- What if we could reverse time?
- Recover data distribution by starting from uniform distribution and running dynamics backwards

Data distribution



Uniform distribution

Observation 2: Microscopic Diffusion is Time Reversible



- Microscopic view
- Brownian motion
- Position updates are small Gaussians
 - Both forwards and backwards in time

Nanoparticles in water

Overview of Diffusion Probabilistic Models



Clean

+ Gaussian noise

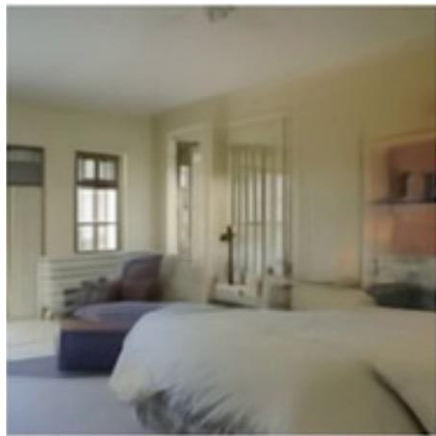


Denoising model

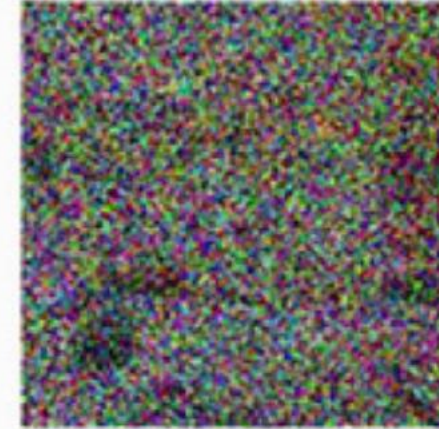
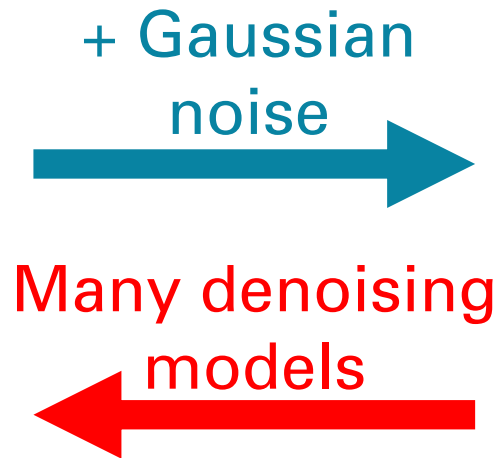


Noisy

Diffusion Probabilistic Models



Clean



Different noise levels

Overview of Diffusion Probabilistic Models

- Destroy all structure in data distribution using diffusion process
- Learn reversal of diffusion process
 - Estimate function for mean and covariance of each step in the reverse diffusion process (binomial rate for binary data)

Lecture overview

- Motivation
- **Denoising Diffusion Models**
 - Overview
 - **Denoising Diffusion Probabilistic Models**
 - Diffusion Models Beat GANs on Image Synthesis; Classifier Guidance
 - Classifier-Free Diffusion Guidance
 - GLIDE: Text-Guided Diffusion Models
 - Dall-E: 1, 2, 3
 - Imagen
- Deeper Dive into Diffusion Models

State of Affairs of this class + field anno 2021

- Autoregressive models
 - MADE, PixelRNN/CNN, Gated PixelCNN, PixelSNAIL
- Flow models
 - Autoregressive Flows, NICE, RealNVP, Glow, Flow++
- Latent Variable Models
 - VAE, IWAE, VQ-VAE
- GANs

GANs gave the most realistic images, but have lots of bells and whistles + struggle with truly covering what's in data distribution

→ Diffusion Models

Denoising Diffusion Probabilistic Models

Denoising Diffusion Probabilistic Models

Jonathan Ho
UC Berkeley

Ajay Jain
UC Berkeley

Pieter Abbeel
UC Berkeley

Denoising Diffusion Probabilistic Models

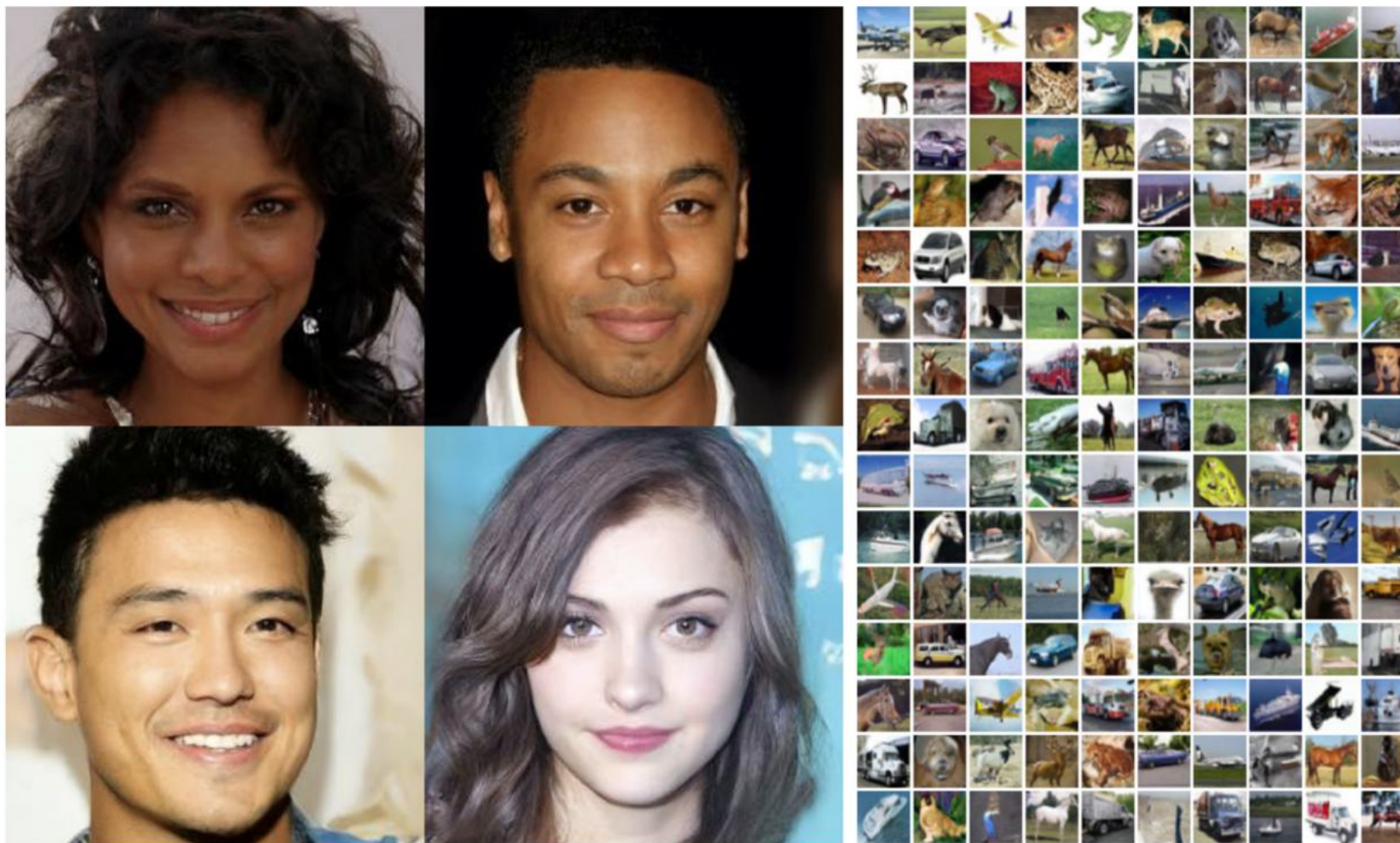
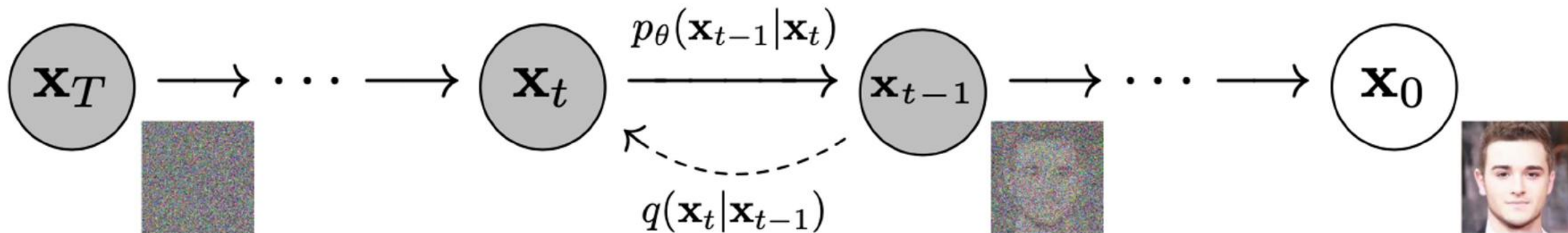


Figure 1: Generated samples on CelebA-HQ 256×256 (left) and unconditional CIFAR10 (right)

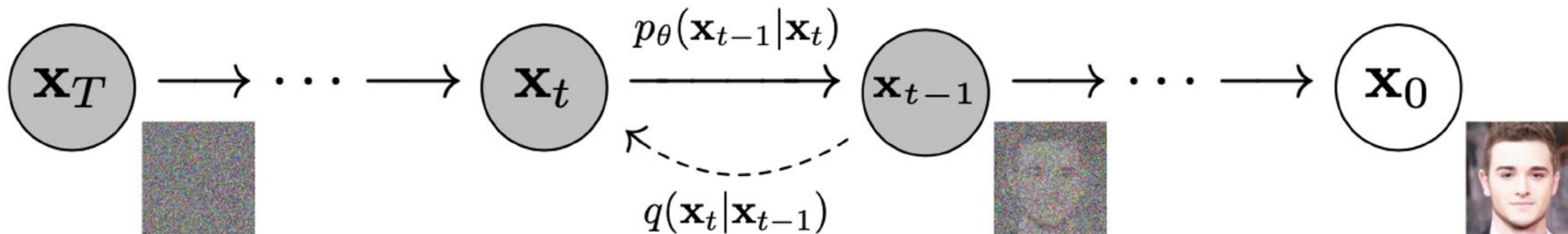
Denoising Diffusion Probabilistic Models



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

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

Denoising Diffusion Probabilistic Models



Algorithm 1 Training

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

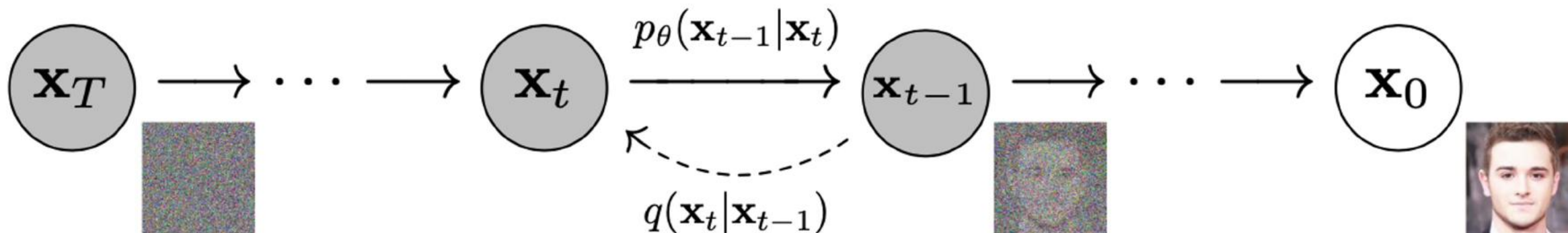
Algorithm 2 Sampling

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

$$\epsilon_\theta \approx \sigma \nabla_x \log p_\theta(x)$$

(conceptually helpful; but we never explicitly represent p_θ)

Denoising Diffusion Probabilistic Models



Training is performed by optimizing the usual variational bound on negative log likelihood:

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

Variance reduction version:

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

Architecture

- Backbone of PixelCNN++ , which is a U-Net based on a Wide ResNet.
- Replaced weight normalization with group normalization to make implementation simpler.
- The 32×32 models use four feature map resolutions (32×32 to 4×4), and the 256×256 models use six.
- All models have two convolutional residual blocks per resolution level and self-attention blocks at the 16×16 resolution between the convolutional blocks.
- Diffusion time t is specified by adding the Transformer sinusoidal position embedding into each residual block.
- The CIFAR10 model has 35.7 million parameters
- The LSUN and CelebA-HQ models have 114 million parameters. Also trained a larger variant of the LSUN Bedroom model with approximately 256 million parameters by increasing filter count.

Denoising Diffusion Probabilistic Models

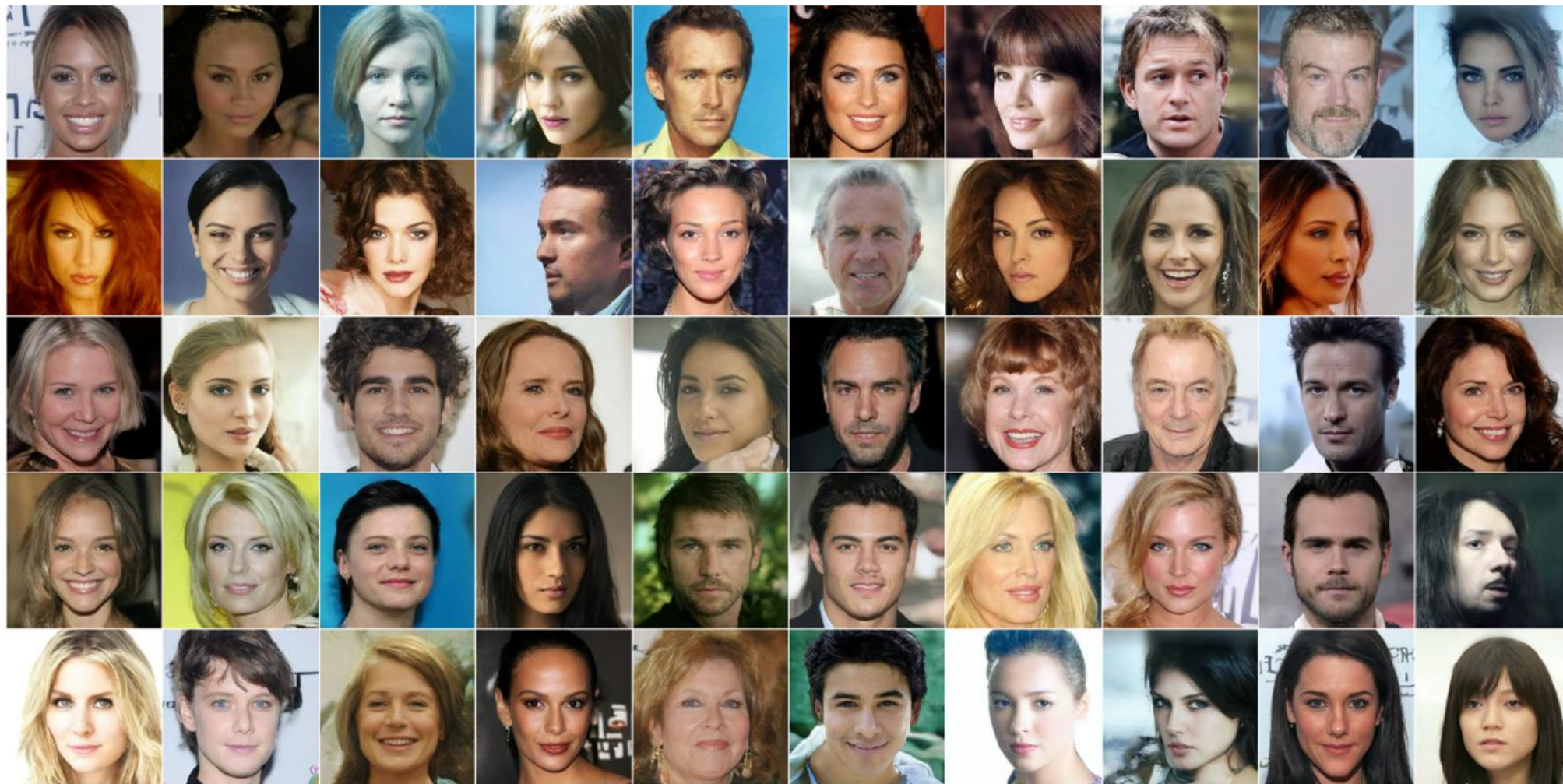


Figure 11: CelebA-HQ 256×256 generated samples

Denoising Diffusion Probabilistic Models



Figure 3: LSUN Church samples. FID=7.89

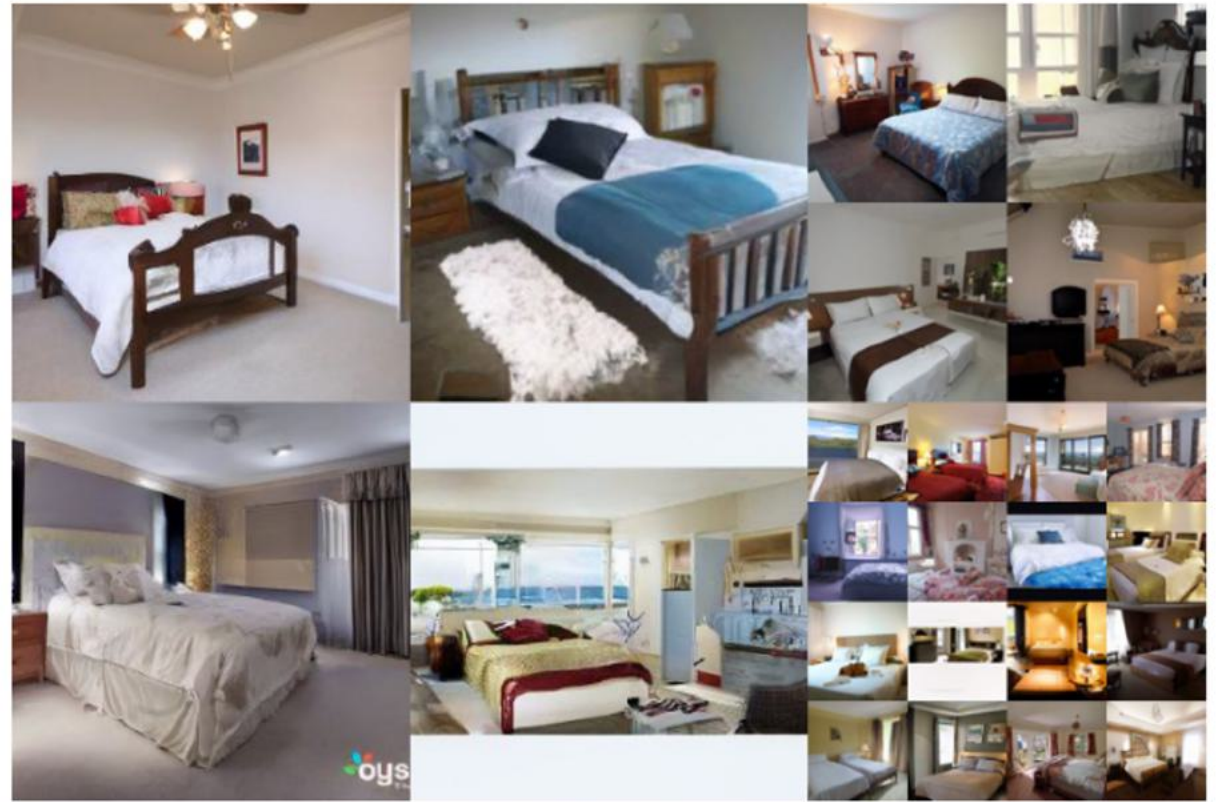


Figure 4: LSUN Bedroom samples. FID=4.90

Denoising Diffusion Probabilistic Models

Table 1: CIFAR10 results. NLL measured in bits/dim.

Model	IS	FID	NLL Test (Train)
Conditional			
EBM [11]	8.30	37.9	
JEM [17]	8.76	38.4	
BigGAN [3]	9.22	14.73	
StyleGAN2 + ADA (v1) [29]	10.06	2.67	
Unconditional			
Diffusion (original) [53]			≤ 5.40
Gated PixelCNN [59]	4.60	65.93	3.03 (2.90)
Sparse Transformer [7]			2.80
PixelIQN [43]	5.29	49.46	
EBM [11]	6.78	38.2	
NCSNv2 [56]		31.75	
NCSN [55]	8.87 ± 0.12	25.32	
SNGAN [39]	8.22 ± 0.05	21.7	
SNGAN-DDLS [4]	9.09 ± 0.10	15.42	
StyleGAN2 + ADA (v1) [29]	9.74 ± 0.05	3.26	
Ours (L , fixed isotropic Σ)	7.67 ± 0.13	13.51	≤ 3.70 (3.69)
Ours (L_{simple})	9.46 ± 0.11	3.17	≤ 3.75 (3.72)

Denoising Diffusion Probabilistic Models



Figure 8: Interpolations of CelebA-HQ 256x256 images with 500 timesteps of diffusion.

Denoising Diffusion Probabilistic Models

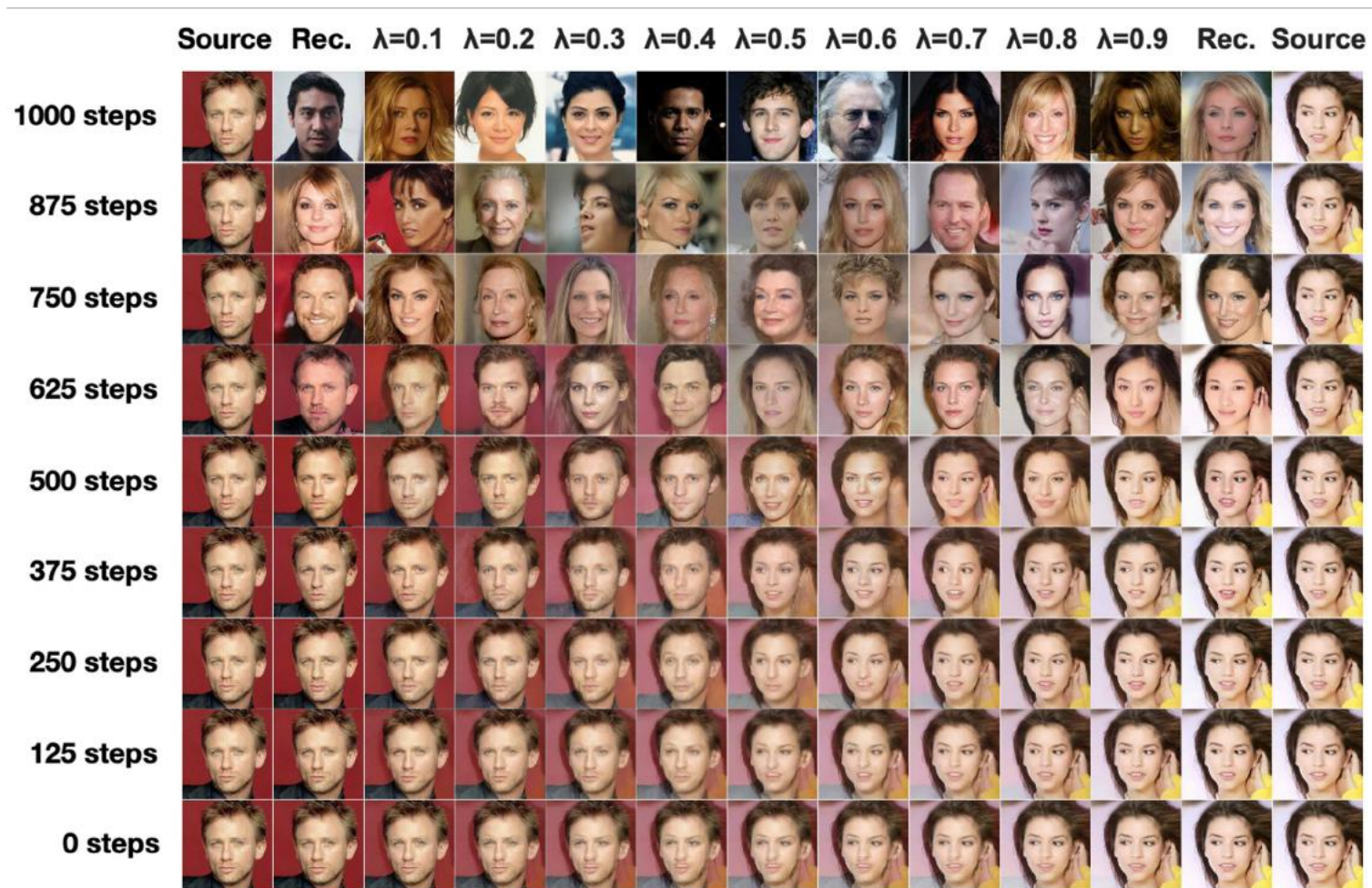


Figure 9: Coarse-to-fine interpolations that vary the number of diffusion steps prior to latent mixing.

Denoising Diffusion Probabilistic Models

Connection to autoregressive decoding Note that the variational bound (5) can be rewritten as:

$$L = D_{\text{KL}}(q(\mathbf{x}_T) \parallel p(\mathbf{x}_T)) + \mathbb{E}_q \left[\sum_{t \geq 1} D_{\text{KL}}(q(\mathbf{x}_{t-1} | \mathbf{x}_t) \parallel p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)) \right] + H(\mathbf{x}_0) \quad (16)$$

(See Appendix A for a derivation.) Now consider setting the diffusion process length T to the dimensionality of the data, defining the forward process so that $q(\mathbf{x}_t | \mathbf{x}_0)$ places all probability mass on \mathbf{x}_0 with the first t coordinates masked out (i.e. $q(\mathbf{x}_t | \mathbf{x}_{t-1})$ masks out the t^{th} coordinate), setting $p(\mathbf{x}_T)$ to place all mass on a blank image, and, for the sake of argument, taking $p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$ to be a fully expressive conditional distribution. With these choices, $D_{\text{KL}}(q(\mathbf{x}_T) \parallel p(\mathbf{x}_T)) = 0$, and minimizing $D_{\text{KL}}(q(\mathbf{x}_{t-1} | \mathbf{x}_t) \parallel p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t))$ trains p_θ to copy coordinates $t + 1, \dots, T$ unchanged and to predict the t^{th} coordinate given $t + 1, \dots, T$. Thus, training p_θ with this particular diffusion is training an autoregressive model.

We can therefore interpret the Gaussian diffusion model (2) as a kind of autoregressive model with a generalized bit ordering that cannot be expressed by reordering data coordinates. Prior work has shown that such reorderings introduce inductive biases that have an impact on sample quality [38], so we speculate that the Gaussian diffusion serves a similar purpose, perhaps to greater effect since Gaussian noise might be more natural to add to images compared to masking noise. Moreover, the Gaussian diffusion length is not restricted to equal the data dimension; for instance, we use $T = 1000$, which is less than the dimension of the $32 \times 32 \times 3$ or $256 \times 256 \times 3$ images in our experiments. Gaussian diffusions can be made shorter for fast sampling or longer for model expressiveness.

Key Inspiration Works:

[Sohl-Dickstein et al, ICML 2015]

Deep Unsupervised Learning using Nonequilibrium Thermodynamics

Jascha Sohl-Dickstein
Stanford University

JASCHA@STANFORD.EDU

Eric A. Weiss
University of California, Berkeley

EAWISS@BERKELEY.EDU

Niru Maheswaranathan
Stanford University

NIRUM@STANFORD.EDU

Surya Ganguli
Stanford University

SGANGULI@STANFORD.EDU

- Introduced Diffusion Models

[Song & Ermon, NeurIPS 2019]

Generative Modeling by Estimating Gradients of the Data Distribution

Yang Song
Stanford University
yangsong@cs.stanford.edu

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

- Annealed Langevin Dynamics for sampling against energy model with score function ($\text{grad} \log p(x)$)
- Very promising sample generation

Lecture overview

- Motivation
- **Denoising Diffusion Models**
 - Overview
 - Denoising Diffusion Probabilistic Models
 - **Diffusion Models Beat GANs on Image Synthesis; Classifier Guidance**
 - Classifier-Free Diffusion Guidance
 - GLIDE: Text-Guided Diffusion Models
 - Dall-E: 1, 2, 3
 - Imagen
- Deeper Dive into Diffusion Models

Diffusion Models Beat GANs on Image Synthesis

Diffusion Models Beat GANs on Image Synthesis

Prafulla Dhariwal*
OpenAI
prafulla@openai.com

Alex Nichol*
OpenAI
alex@openai.com

Abstract

We show that diffusion models can achieve image sample quality superior to the current state-of-the-art generative models. We achieve this on unconditional image synthesis by finding a better architecture through a series of ablations. For conditional image synthesis, we further improve sample quality with classifier guidance: a simple, compute-efficient method for trading off diversity for fidelity using gradients from a classifier. We achieve an FID of 2.97 on ImageNet 128×128 , 4.59 on ImageNet 256×256 , and 7.72 on ImageNet 512×512 , and we match BigGAN-deep even with as few as 25 forward passes per sample, all while maintaining better coverage of the distribution. Finally, we find that classifier guidance combines well with upsampling diffusion models, further improving FID to 3.94 on ImageNet 256×256 and 3.85 on ImageNet 512×512 . We release our code at <https://github.com/openai/guided-diffusion>.

GANs [19] currently hold the state-of-the-art on most image generation tasks [5, 68, 28] as measured by sample quality metrics such as FID [23], Inception Score [54] and Precision [32]. However, some of these metrics do not fully capture diversity, and it has been shown that GANs capture less diversity than state-of-the-art likelihood-based models [51, 43, 42]. Furthermore, GANs are often difficult to train, collapsing without carefully selected hyperparameters and regularizers [5, 41, 4].

Diffusion Models Beat GANs on Image Synthesis

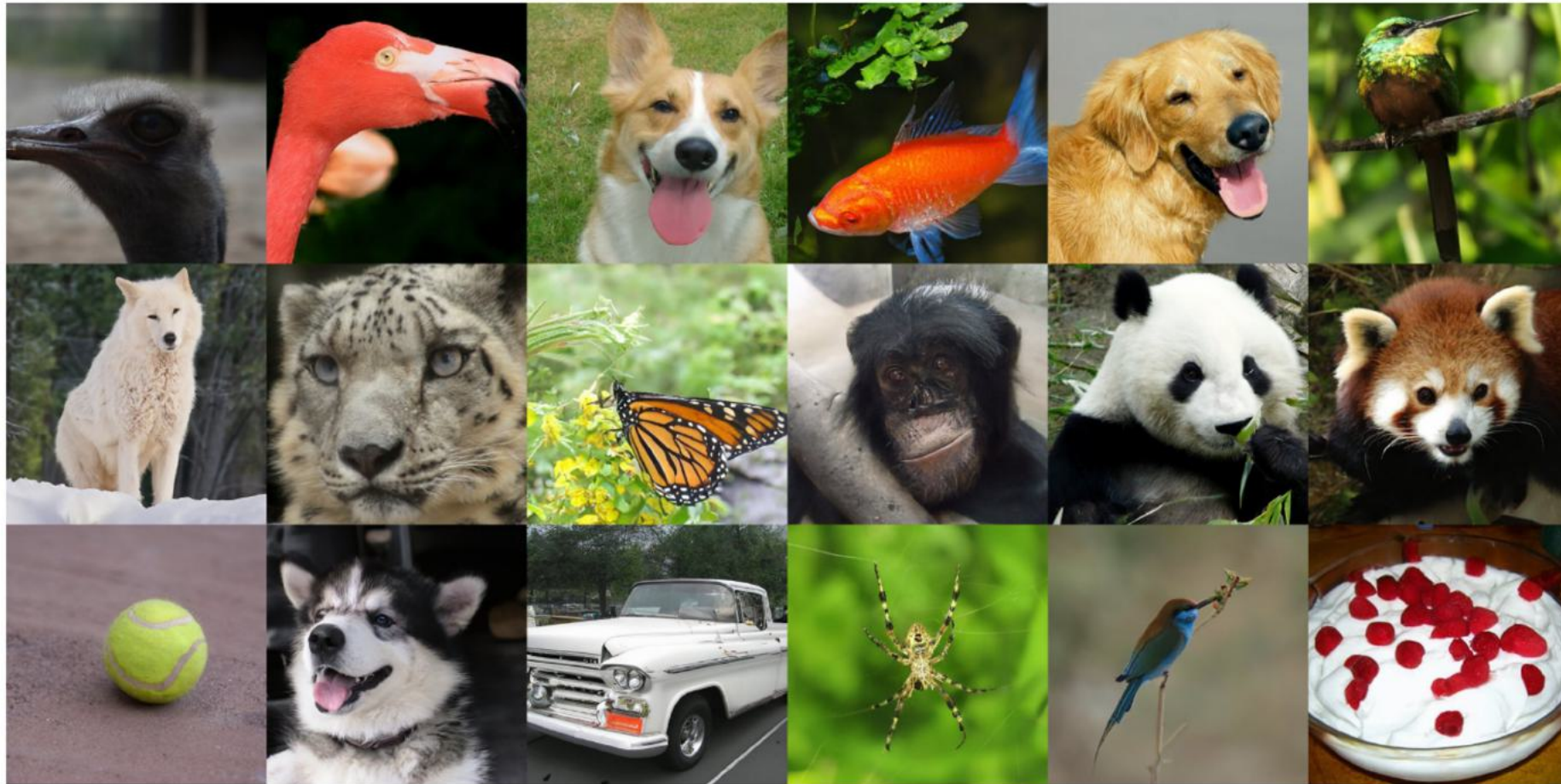


Figure 1: Selected samples from our best ImageNet 512×512 model (FID 3.85)

Algorithmic Improvements

Learned variance

Nichol and Dhariwal [43] find that fixing the variance $\Sigma_\theta(x_t, t)$ to a constant as done in Ho et al. [25] is sub-optimal for sampling with fewer diffusion steps, and propose to parameterize $\Sigma_\theta(x_t, t)$ as a neural network whose output v is interpolated as:

$$\Sigma_\theta(x_t, t) = \exp(v \log \beta_t + (1 - v) \log \tilde{\beta}_t) \quad (1)$$

Deterministic Sampler

Song et al. [57] propose DDIM, which formulates an alternative non-Markovian noising process that has the same forward marginals as DDPM, but allows producing different reverse samplers by changing the variance of the reverse noise. By setting this noise to 0, they provide a way to turn any model $\epsilon_\theta(x_t, t)$ into a deterministic mapping from latents to images, and find that this provides an alternative way to sample with fewer steps. We adopt this sampling approach when using fewer than 50 sampling steps, since Nichol and Dhariwal [43] found it to be beneficial in this regime.

Architecture Improvements

- Increasing depth versus width, holding model size relatively constant.
- Increasing the number of attention heads.
- Using attention at 32×32 , 16×16 , and 8×8 resolutions rather than only at 16×16 .
- Using the BigGAN [5] residual block for upsampling and downsampling the activations, following [60].
- Rescaling residual connections with $\frac{1}{\sqrt{2}}$, following [60, 27, 28].

Classifier Guidance

Key idea: add an additional term in the sampling process to direct towards the desired class

Train a classifier $p_\phi(y|x_t, t)$ on noisy images x_t

Use gradients $\nabla_{x_t} \log p_\phi(y|x_t, t)$ to guide towards a class label

$$x_{t-1} = w_0 x_t + w_1 \epsilon_\theta(x_t, t) + w_3 \nabla_{x_t} \log p_\phi(y|x_t, t) \quad \epsilon_\theta \approx \sigma \nabla_{x_t} \log p_\theta(x_t)$$

Similar Image Quality + Better Diversity than BigGAN



Figure 6: Samples from BigGAN-deep with truncation 1.0 (FID 6.95, left) vs samples from our diffusion model with guidance (FID 4.59, middle) and samples from the training set (right).

Lecture overview

- Motivation
- **Denoising Diffusion Models**
 - Overview
 - Denoising Diffusion Probabilistic Models
 - Diffusion Models Beat GANs on Image Synthesis; Classifier Guidance
 - **Classifier-Free Diffusion Guidance**
 - GLIDE: Text-Guided Diffusion Models
 - Dall-E: 1, 2, 3
 - Imagen
- Deeper Dive into Diffusion Models

CLASSIFIER-FREE DIFFUSION GUIDANCE

Jonathan Ho & Tim Salimans

Google Research, Brain team

{jonathanho, salimans}@google.com

ABSTRACT

Classifier guidance is a recently introduced method to trade off mode coverage and sample fidelity in conditional diffusion models post training, in the same spirit as low temperature sampling or truncation in other types of generative models. Classifier guidance combines the score estimate of a diffusion model with the gradient of an image classifier and thereby requires training an image classifier separate from the diffusion model. It also raises the question of whether guidance can be performed without a classifier. We show that guidance can be indeed performed by a pure generative model without such a classifier: in what we call classifier-free guidance, we jointly train a conditional and an unconditional diffusion model, and we combine the resulting conditional and unconditional score estimates to attain a trade-off between sample quality and diversity similar to that obtained using classifier guidance.

Classifier-Free Guidance

- Motivation
 - Classifier-guidance enabled “low temperature” / “sharper” samples
 - BUT: Can we achieve this without an additional classifier to train, purely in the generative model itself?
- Key Idea, consider the extra term in classifier free guidance sampling + apply Bayes rule:

$$\begin{array}{l}
 = \\
 = \\
 = \\
 = \\
 =
 \end{array}
 \left| \begin{array}{l}
 +w\nabla_x \log p(y|x) \\
 +w\nabla_x \log \frac{p(x|y)p(y)}{p(x)} \\
 +w\nabla_x \log p(x|y) + w\nabla_x \log p(y) - w\nabla_x \log p(x) \\
 +w\nabla_x \log p(x|y) + 0 - w\nabla_x \log p(x) \\
 w\epsilon_\theta(x, y) - w\epsilon_\theta(x)
 \end{array} \right.
 \begin{array}{l}
 \leq \text{classifier-guidance} \\
 \\
 \leq \text{classifier-**free** guidance} \\
 \leq \text{in DDPM notation}
 \end{array}$$

No need for classifier, just need to train both conditional and unconditional generator

Sampling with Classifier-Free Guidance

Algorithm 2 Conditional sampling with classifier-free guidance

Require: w : guidance strength

Require: \mathbf{c} : conditioning information for conditional sampling

Require: $\lambda_1, \dots, \lambda_T$: increasing log SNR sequence with $\lambda_1 = \lambda_{\min}$, $\lambda_T = \lambda_{\max}$

1: $\mathbf{z}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

2: **for** $t = 1, \dots, T$ **do**

 ▷ Form the classifier-free guided score at log SNR λ_t

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

 ▷ Sampling step (could be replaced by another sampler, e.g. DDIM)

4: $\tilde{\mathbf{x}}_t = (\mathbf{z}_t - \sigma_{\lambda_t} \tilde{\epsilon}_t) / \alpha_{\lambda_t}$

5: $\mathbf{z}_{t+1} \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}_{\lambda_{t+1}|\lambda_t}(\mathbf{z}_t, \tilde{\mathbf{x}}_t), (\tilde{\sigma}_{\lambda_{t+1}|\lambda_t}^2)^{1-v} (\sigma_{\lambda_t|\lambda_{t+1}}^2)^v)$ if $t < T$ else $\mathbf{z}_{t+1} = \tilde{\mathbf{x}}_t$

6: **end for**

7: **return** \mathbf{z}_{T+1}

Training with Classifier-Free Guidance

Algorithm 1 Joint training a diffusion model with classifier-free guidance

Require: p_{uncond} : probability of unconditional training

1: **repeat**

2: $(\mathbf{x}, \mathbf{c}) \sim p(\mathbf{x}, \mathbf{c})$ ▷ Sample data with conditioning from the dataset

3: $\mathbf{c} \leftarrow \emptyset$ with probability p_{uncond} ▷ Randomly discard conditioning to train unconditionally

4: $\lambda \sim p(\lambda)$ ▷ Sample log SNR value

5: $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

6: $\mathbf{z}_\lambda = \alpha_\lambda \mathbf{x} + \sigma_\lambda \epsilon$ ▷ Corrupt data to the sampled log SNR value

7: Take gradient step on $\nabla_\theta \|\epsilon_\theta(\mathbf{z}_\lambda, \mathbf{c}) - \epsilon\|^2$ ▷ Optimization of denoising model

8: **until** converged

Effect of Classifier-Free Guidance



Figure 2: The effect of guidance on a mixture of three Gaussians, each mixture component representing data conditioned on a class. The leftmost plot is the non-guided marginal density. Left to right are densities of mixtures of normalized guided conditionals with increasing guidance strength.

Effect of Classifier-Free Guidance



Figure 1: Classifier-free guidance on the malamute class for a 64x64 ImageNet diffusion model. Left to right: increasing amounts of classifier-free guidance, starting from non-guided samples on the left.

Classifier-Free Guidance



Figure 3: Classifier-free guidance on 128x128 ImageNet. Left: non-guided samples, right: classifier-free guided samples with $w = 3.0$. Interestingly, strongly guided samples such as these display saturated colors. See Fig. 8 for more.

FID / IS over training run

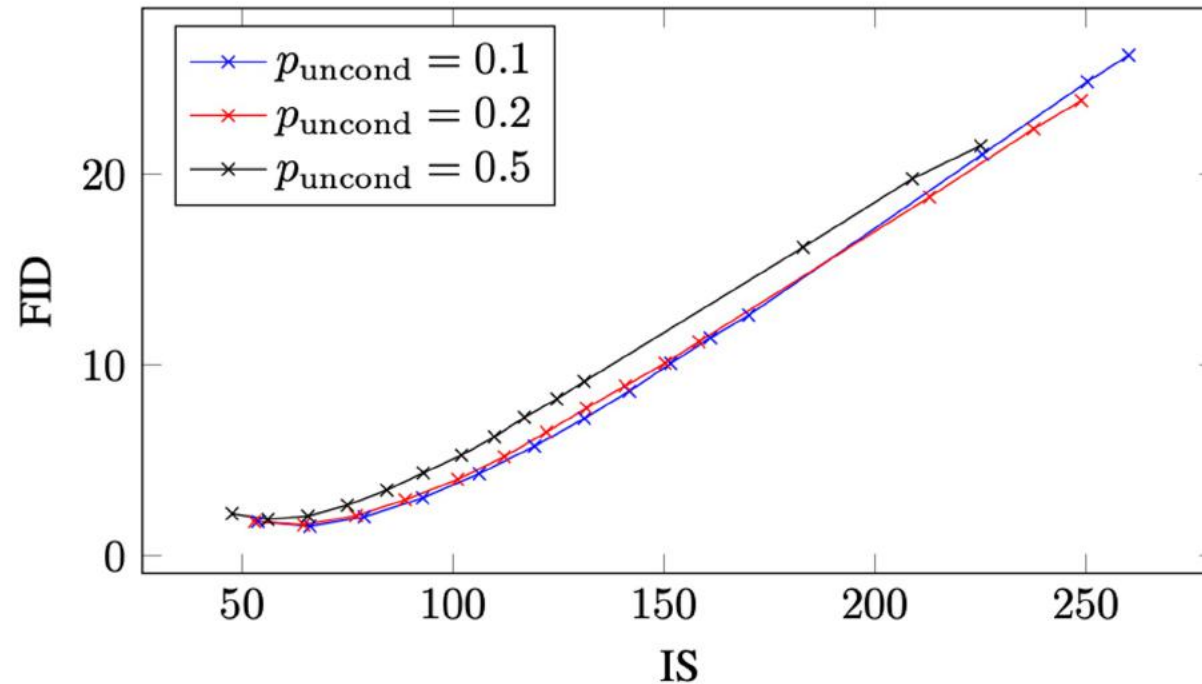


Figure 4: IS/FID curves over guidance strengths for ImageNet 64x64 models. Each curve represents a model with unconditional training probability p_{uncond} . Accompanies Table 1.

IS keeps getting better – i.e. more and more “realistic” member of a class
FID briefly improves, then keeps getting worse – i.e. lose diversity

Classifier-Free Guidance

- Achieves guidance in a very simple way;
 - One-line change of code during training (randomly drop out conditioning)
 - During sampling just mix conditional and unconditional score estimates

Lecture overview

- Motivation
- **Denoising Diffusion Models**
 - Overview
 - Denoising Diffusion Probabilistic Models
 - Diffusion Models Beat GANs on Image Synthesis; Classifier Guidance
 - Classifier-Free Diffusion Guidance
 - **GLIDE: Text-Guided Diffusion Models**
 - Dall-E: 1, 2, 3
 - Imagen
- Deeper Dive into Diffusion Models

GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models

**Alex Nichol* Prafulla Dhariwal* Aditya Ramesh* Pranav Shyam Pamela Mishkin Bob McGrew
Ilya Sutskever Mark Chen**

[Nichol*, Dhariwal*, Ramesh* et al, 2022]

GLIDE

Main Idea:

- 3.5 Billion parameter text-conditional diffusion model for 64x64
- 1.5 Billion parameter text-conditional diffusion model for super-resolution to 256x256
- Classifier-Free Diffusion Guidance with caption conditioning (rather than class conditioning)
- Also tried CLIP-Guided Diffusion but that didn't work as well

GLIDE



“a hedgehog using a calculator”



“a corgi wearing a red bowtie and a purple party hat”



“robots meditating in a vipassana retreat”



“a fall landscape with a small cottage next to a lake”



“a surrealist dream-like oil painting by salvador dali of a cat playing checkers”



“a professional photo of a sunset behind the grand canyon”



“a high-quality oil painting of a psychedelic hamster dragon”



“an illustration of albert einstein wearing a superhero costume”

GLIDE



“a man with red hair”



“a vase of flowers”



“an old car in a snowy forest”



“a man wearing a white hat”

Figure 2. Text-conditional image inpainting examples from GLIDE. The green region is erased, and the model fills it in conditioned on the given prompt. Our model is able to match the style and lighting of the surrounding context to produce a realistic completion.

GLIDE



Figure 3. Iteratively creating a complex scene using GLIDE. First, we generate an image for the prompt “a cozy living room”, then use the shown inpainting masks and follow-up text prompts to add a painting to the wall, a coffee table, and a vase of flowers on the coffee table, and finally to move the wall up to the couch.

GLIDE

- Classifier-Free Diffusion Guidance with caption conditioning (rather than class conditioning)

$$\hat{\epsilon}_{\theta}(x_t|c) = \epsilon_{\theta}(x_t|\emptyset) + s \cdot (\epsilon_{\theta}(x_t|c) - \epsilon_{\theta}(x_t|\emptyset))$$

- Also tried CLIP-Guided Diffusion but that didn't work as well

$$\hat{\mu}_{\theta}(x_t|c) = \mu_{\theta}(x_t|c) + s \cdot \Sigma_{\theta}(x_t|c) \nabla_{x_t} (f(x_t) \cdot g(c))$$

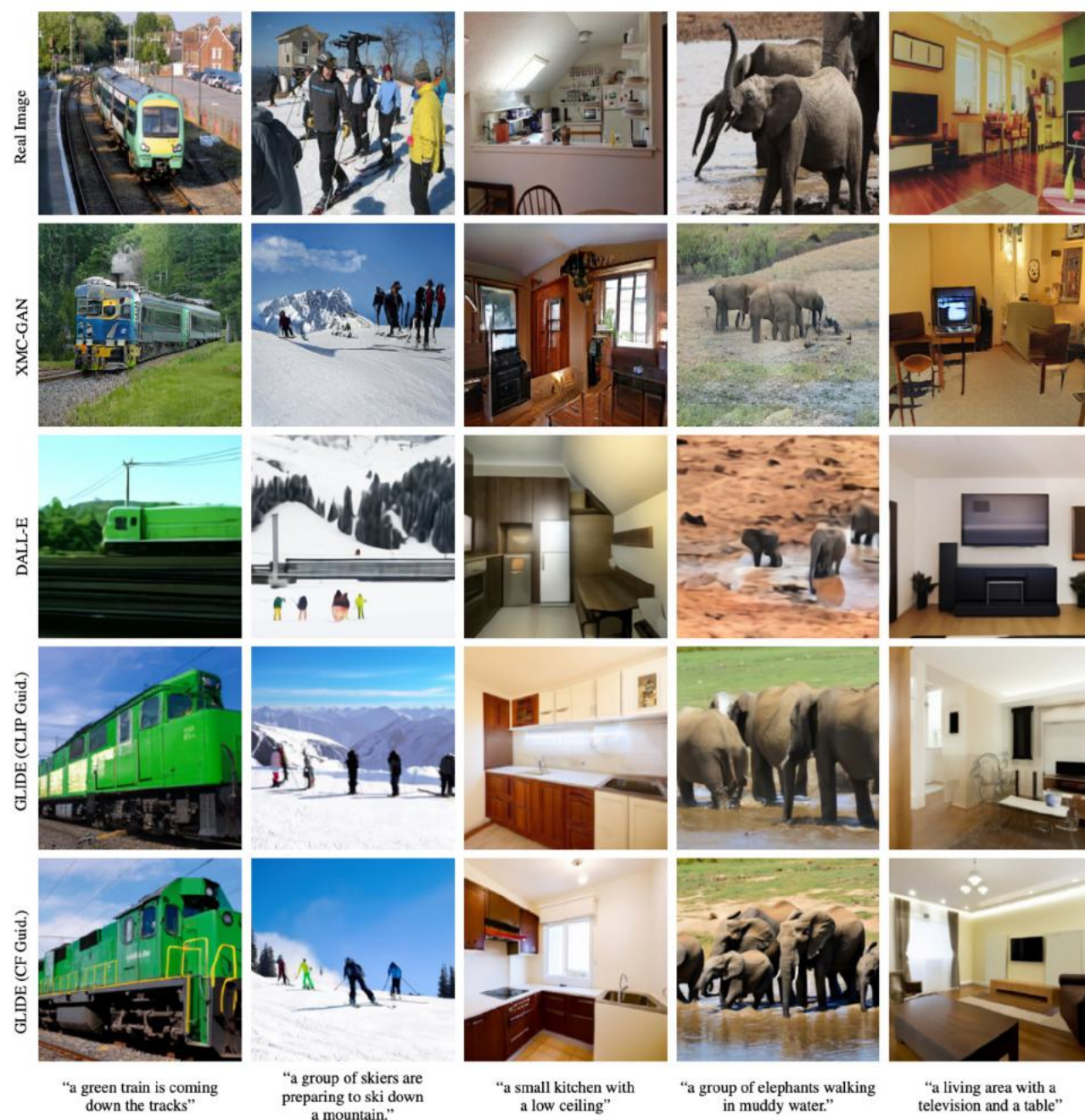


Figure 5. Random image samples on MS-COCO prompts. For XMC-GAN, we take samples from Zhang et al. (2021). For DALL-E, we generate samples at temperature 0.85 and select the best of 256 using CLIP reranking. For GLIDE, we use CLIP guidance with scale 2.0 and classifier-free guidance with scale 3.0. We do not perform any CLIP reranking or cherry-picking for GLIDE.

Lecture overview

- Motivation
- **Denoising Diffusion Models**
 - Overview
 - Denoising Diffusion Probabilistic Models
 - Diffusion Models Beat GANs on Image Synthesis; Classifier Guidance
 - Classifier-Free Diffusion Guidance
 - GLIDE: Text-Guided Diffusion Models
 - **Dall-E: 1, 2, 3**
 - Imagen
- Deeper Dive into Diffusion Models

Dall-E1

Zero-Shot Text-to-Image Generation

**Aditya Ramesh¹ Mikhail Pavlov¹ Gabriel Goh¹ Scott Gray¹
Chelsea Voss¹ Alec Radford¹ Mark Chen¹ Ilya Sutskever¹**

NOTE: Dall-E1 does not involve a diffusion model, but later Dall-E models do

Dall-E1

Key Idea:

- Transformer with text tokens and image tokens
- Image tokens from discrete variational autoencoder, $k=8192$
 - \sim VQ-VAE with Gumbel-Softmax hence trainable as VAE, some small architecture changes
- 250 million text image pairs
- Sample generation with post-processing by CLIP re-ranking of samples

Dall-E1

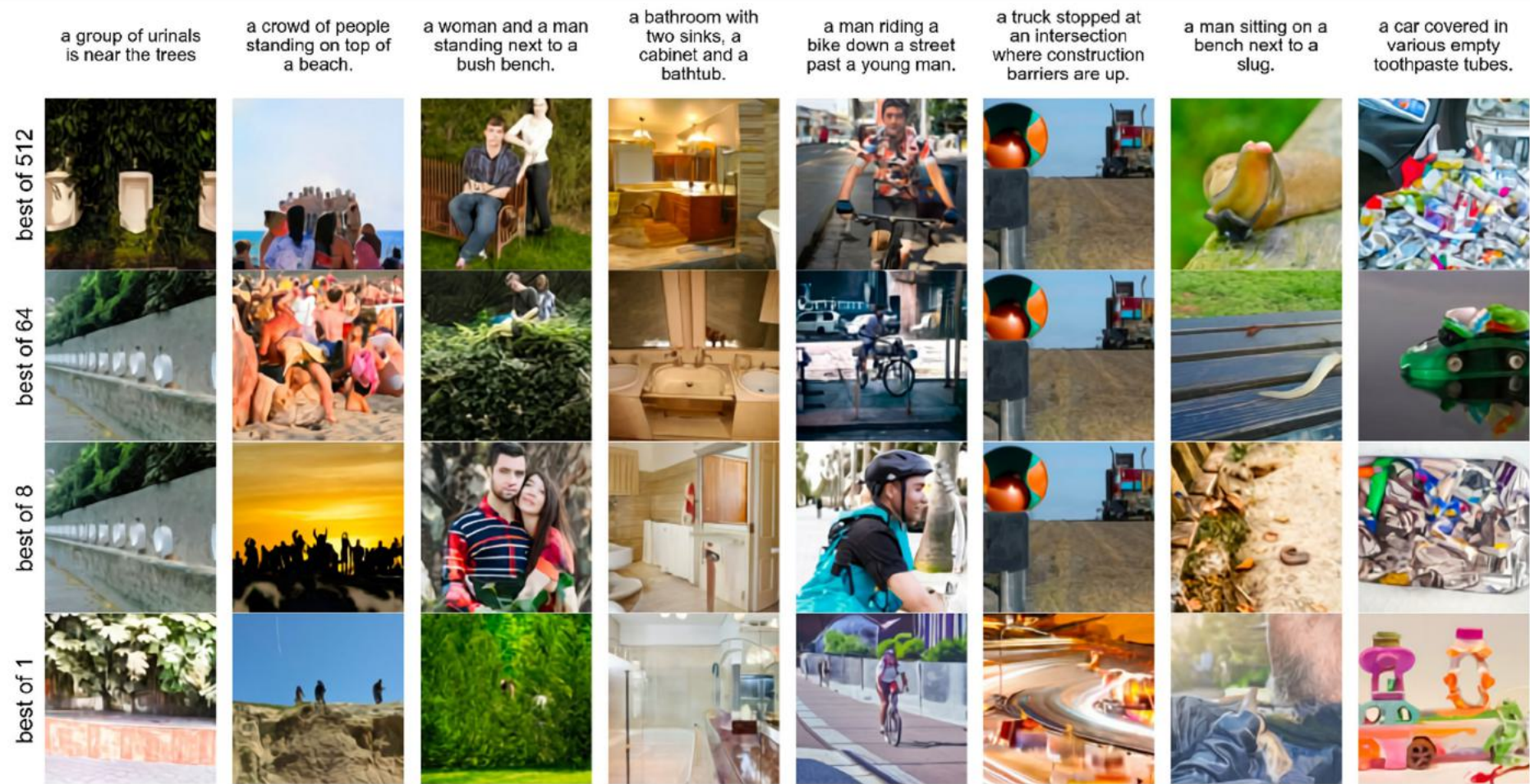


Figure 6. Effect of increasing the number of images for the contrastive reranking procedure on MS-COCO captions.

Findings

- Generalization ability to compose many concepts at high level of abstraction (including unusual combinations)



(a) a tapir made of accordion. a tapir with the texture of an accordion.

(b) an illustration of a baby hedgehog in a christmas sweater walking a dog

(c) a neon sign that reads "backprop". a neon sign that reads "backprop". backprop neon sign

(d) the exact same cat on the top as a sketch on the bottom

Dall-E2 / unCLIP

Hierarchical Text-Conditional Image Generation with CLIP Latents

Aditya Ramesh*
OpenAI
aramesh@openai.com

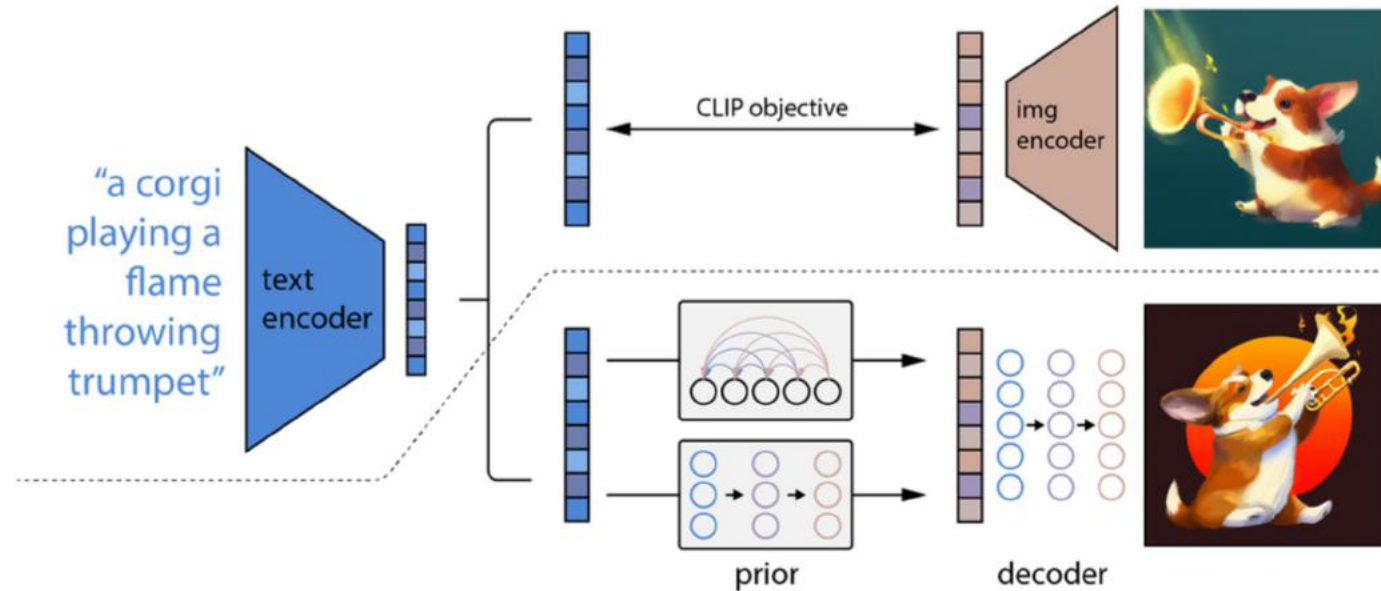
Prafulla Dhariwal*
OpenAI
prafulla@openai.com

Alex Nichol*
OpenAI
alex@openai.com

Casey Chu*
OpenAI
casey@openai.com

Mark Chen
OpenAI
mark@openai.com

Dall-E2 / unCLIP

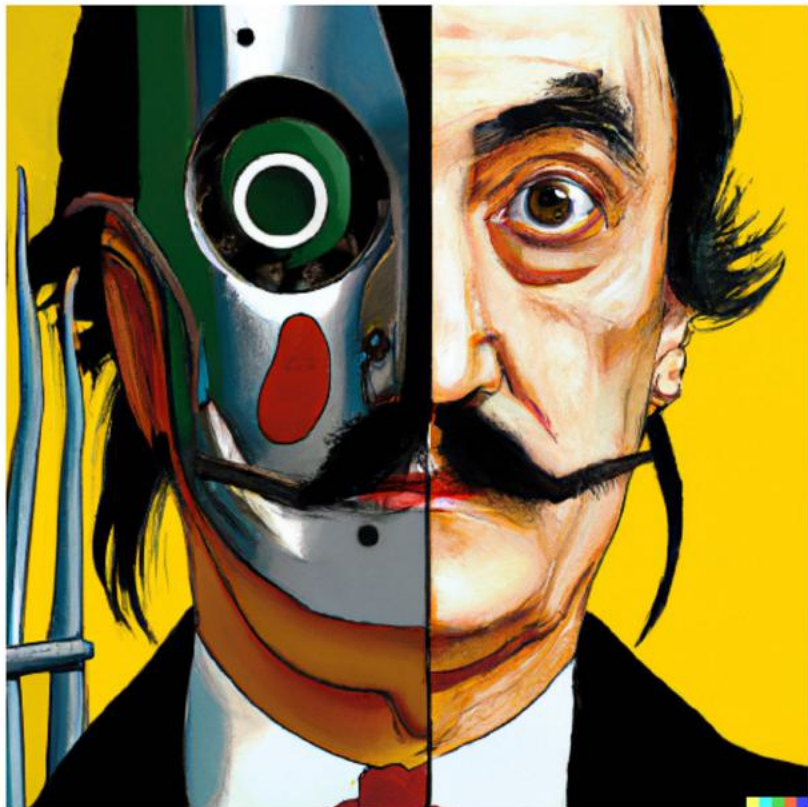


First 64x64
Then upsample 256x256
Then upsample 1024x1024
No conditioning in upsamplers

Figure 2: A high-level overview of unCLIP. Above the dotted line, we depict the CLIP training process, through which we learn a joint representation space for text and images. Below the dotted line, we depict our text-to-image generation process: a CLIP text embedding is first fed to an autoregressive or diffusion prior to produce an image embedding, and then this embedding is used to condition a diffusion decoder which produces a final image. Note that the CLIP model is frozen during training of the prior and decoder.

Decoder = ~ GLIDE but conditioned on CLIP image embedding space (+text)
Enable classifier-free guidance by setting CLIP embeddings = 0 (10%) and dropping text caption 50% of the time

Dall-E2 / unCLIP: Samples



vibrant portrait painting of Salvador Dalí with a robotic half face



a shiba inu wearing a beret and black turtleneck



a close up of a handpalm with leaves growing from it

Encode with CLIP then Decode/Sample

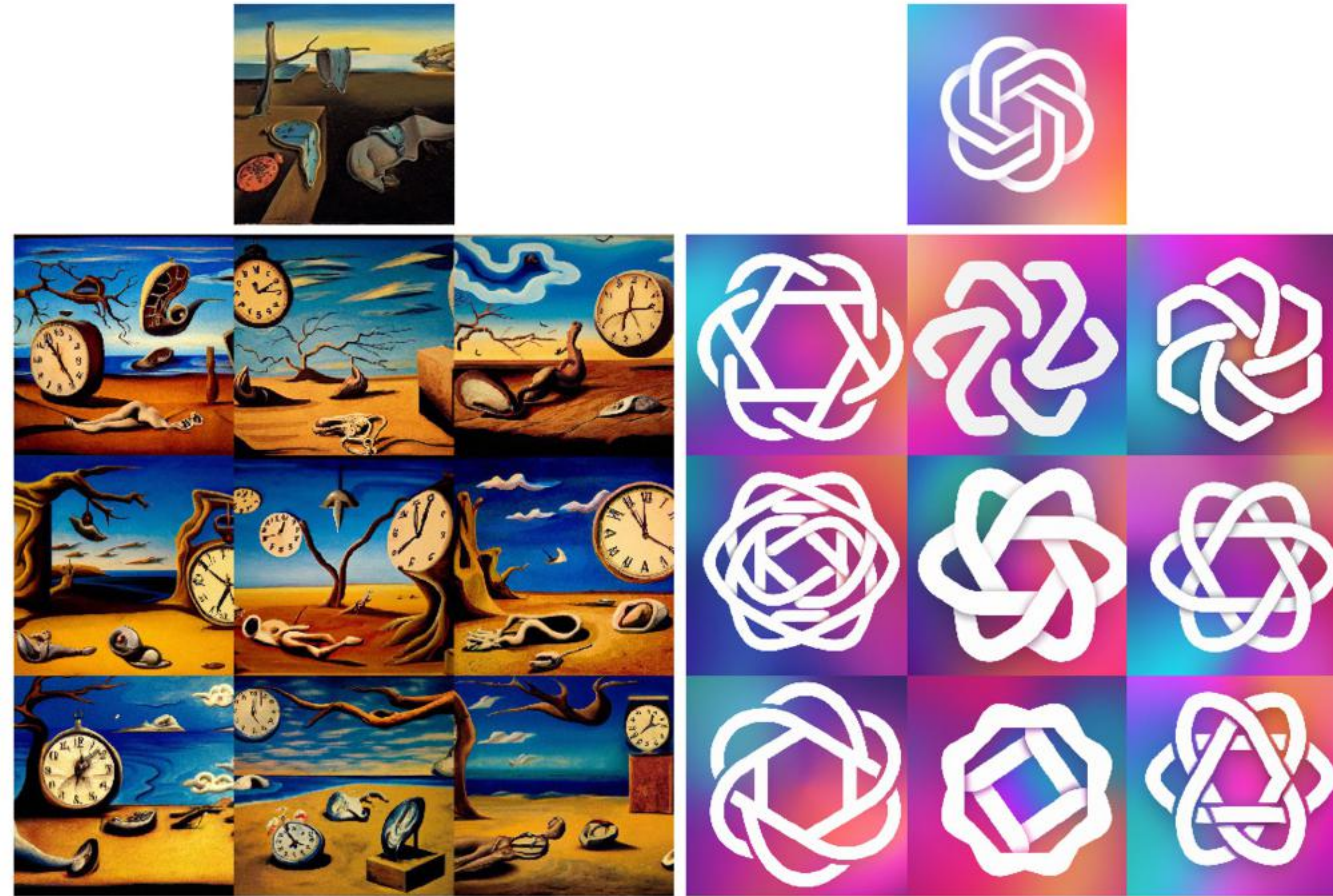


Figure 3: Variations of an input image by encoding with CLIP and then decoding with a diffusion model. The variations preserve both semantic information like presence of a clock in the painting and the overlapping strokes in the logo, as well as stylistic elements like the surrealism in the painting and the color gradients in the logo, while varying the non-essential details.

Interpolation in CLIP image embeddings



Dall-E3

Improving Image Generation with Better Captions

James Betker^{*†}
jbetker@openai.com

Gabriel Goh^{*†}
ggoh@openai.com

Li Jing^{*†}
lijing@openai.com

Tim Brooks[†]

Jianfeng Wang[‡] **Linjie Li**[‡] **Long Ouyang**[†] **Juntang Zhuang**[†] **Joyce Lee**[†] **Yufei Guo**[†]

Wesam Manassra[†]

Prafulla Dhariwal[†]

Casey Chu[†]

Yunxin Jiao[†]

Aditya Ramesh^{*†}
aramesh@openai.com

Dall-E3

Key Hypothesis:

- Existing text-to-image models struggle to follow detailed image descriptions and often ignore words or confuse the meaning of prompts. We hypothesize that this issue stems from noisy and inaccurate image captions in the training dataset.

Proposed Method:

- train a bespoke image captioner and use it to recaption the training dataset
- at sample time: use GPT-4 to “upsample” the prompt

Underlying Model:

- ~ stable diffusion VAE: 256x256 -> 32x32 latents + diffusion (not VAE!) decoder; might have super-resolution diffusion or might not (details undisclosed)

Results:

- Even better :)

Weaknesses:

- Spatial awareness
- Text rendering not as good as hoped for (possibly because of tokenizer)



In a fantastical setting, a highly detailed furry humanoid skunk with piercing eyes confidently poses in a medium shot, wearing an animal hide jacket. The artist has masterfully rendered the character in digital art, capturing the intricate details of fur and clothing texture.



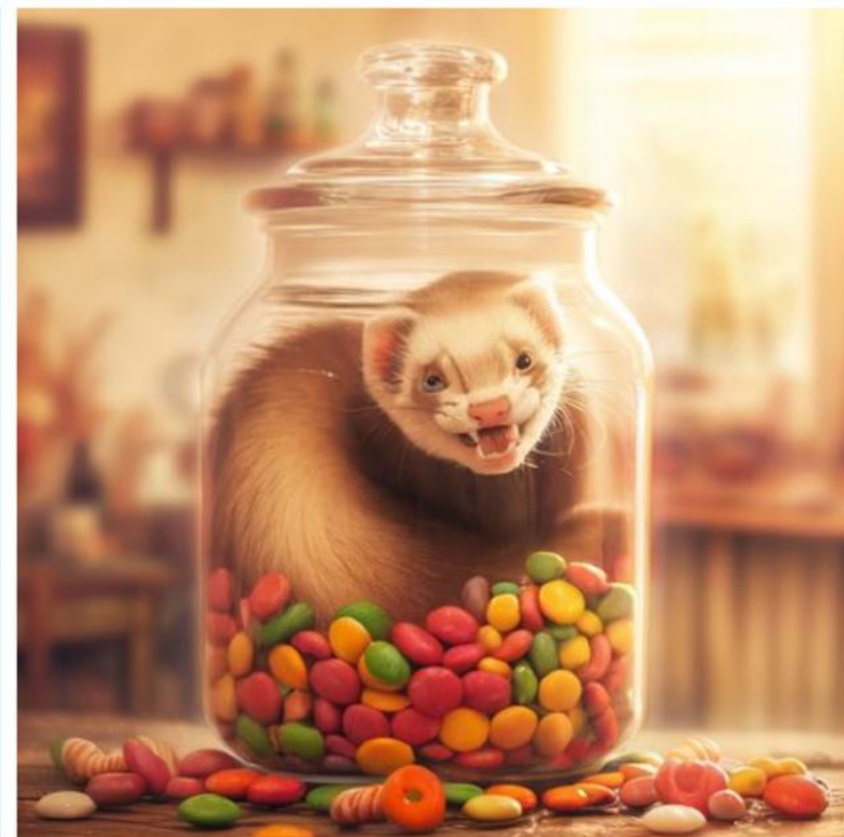
A illustration from a graphic novel. A bustling city street under the shine of a full moon. The sidewalks bustling with pedestrians enjoying the nightlife. At the corner stall, a young woman with fiery red hair, dressed in a signature velvet cloak, is haggling with the grumpy old vendor. The grumpy vendor, a tall, sophisticated man is wearing a sharp suit, sports a noteworthy moustache is animatedly conversing on his steampunk telephone.



Ancient pages filled with sketches and writings of fantasy beasts, monsters, and plants sprawl across an old, weathered journal. The faded dark green ink tells tales of magical adventures, while the high-resolution drawings detail each creature's intricate characteristics. Sunlight peeks through a nearby window, illuminating the pages and revealing their timeworn charm.



A vibrant 1960s-style poster depicting interplanetary migration, with a retro rocket ship blasting off from earth towards a distant, colorful planet. Bold typography announces "Join the galactic adventure!" with smaller text underneath reading "Explore new worlds, build a brighter future." The background features a swirling galaxy of stars and constellations.



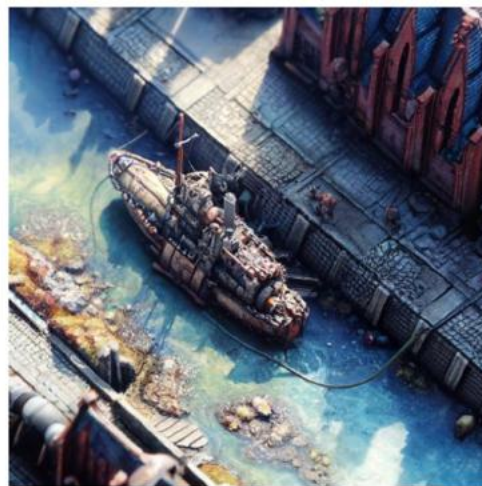
A mischievous ferret with a playful grin squeezes itself into a large glass jar, surrounded by colorful candy. The jar sits on a wooden table in a cozy kitchen, and warm sunlight filters through a nearby window.



A bird scaring a scarecrow.



Paying for a quarter-sized pizza with a pizza-sized quarter.



A smafml vesfef epropoeilled on watvewr by ors, sauls, or han engie.



A large, vibrant bird with an impressive wingspan swoops down from the sky, letting out a piercing call as it approaches a weathered scarecrow in a sunlit field. The scarecrow, dressed in tattered clothing and a straw hat, appears to tremble, almost as if it's coming to life in fear of the approaching bird.



A person is standing at a pizza counter, holding a gigantic quarter the size of a pizza. The cashier, wide-eyed with astonishment, hands over a tiny, quarter-sized pizza in return. The background features various pizza toppings and other customers, all of them equally amazed by the unusual transaction.



A small vessel, propelled on water by oars, sails, or an engine, floats gracefully on a serene lake. the sun casts a warm glow on the water, reflecting the vibrant colors of the sky as birds fly overhead.

Figure 6 – Effect of using "upsampled" drawbench captions to create samples with DALL-E 3. Original drawbench captions on top, upsampled captions on bottom. Images are best of 4 for each caption.

Lecture overview

- Motivation
- **Denoising Diffusion Models**
 - Overview
 - Denoising Diffusion Probabilistic Models
 - Diffusion Models Beat GANs on Image Synthesis; Classifier Guidance
 - Classifier-Free Diffusion Guidance
 - GLIDE: Text-Guided Diffusion Models
 - Dall-E: 1, 2, 3
 - Imagen
- Deeper Dive into Diffusion Models

Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding

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†, Mohammad Norouzi*

{sahariac,williamchan,mnorouzi}@google.com
{srbs,lala,jwhang,jonathanho,davidfleet}@google.com

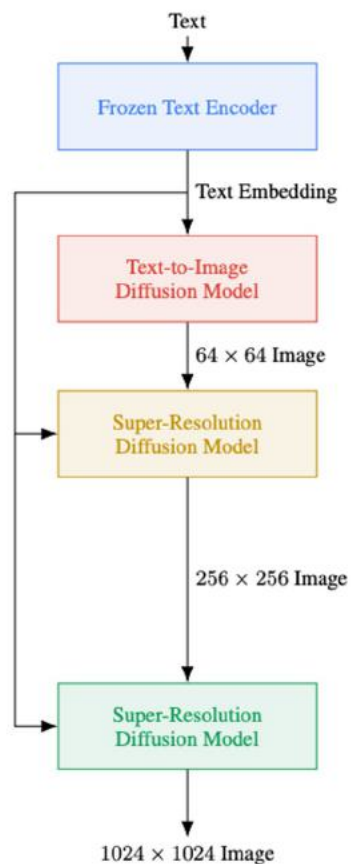
Google Research, Brain Team
Toronto, Ontario, Canada

Imagen achieves a new state-of-the-art FID score of 7.27 on the COCO dataset, without ever training on COCO, and human raters find Imagen samples to be on par with the COCO data itself in image-text alignment. To assess text-to-image models in greater depth, we introduce DrawBench, a comprehensive and challenging benchmark for text-to-image models. With DrawBench, we compare Imagen with recent methods including VQ-GAN+CLIP, Latent Diffusion Models, GLIDE and DALL-E 2, and find that human raters prefer Imagen over other models in side-by-side comparisons, both in terms of sample quality and image-text alignment. See imagen.research.google for an overview of the results.

Imagen Architecture

2.5M training steps for each model

2B
600M
400M



"A Golden Retriever dog wearing a blue checkered beret and red dotted turtleneck."

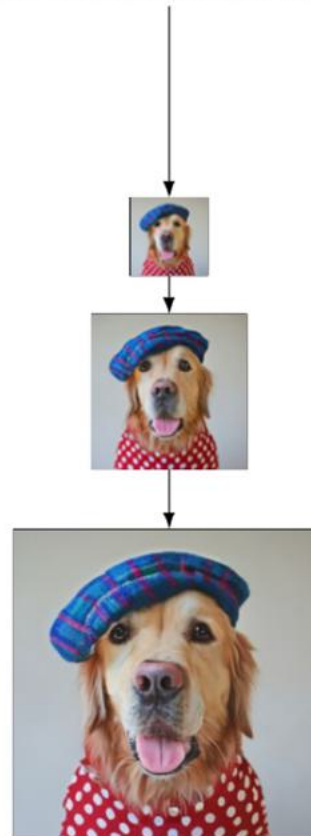


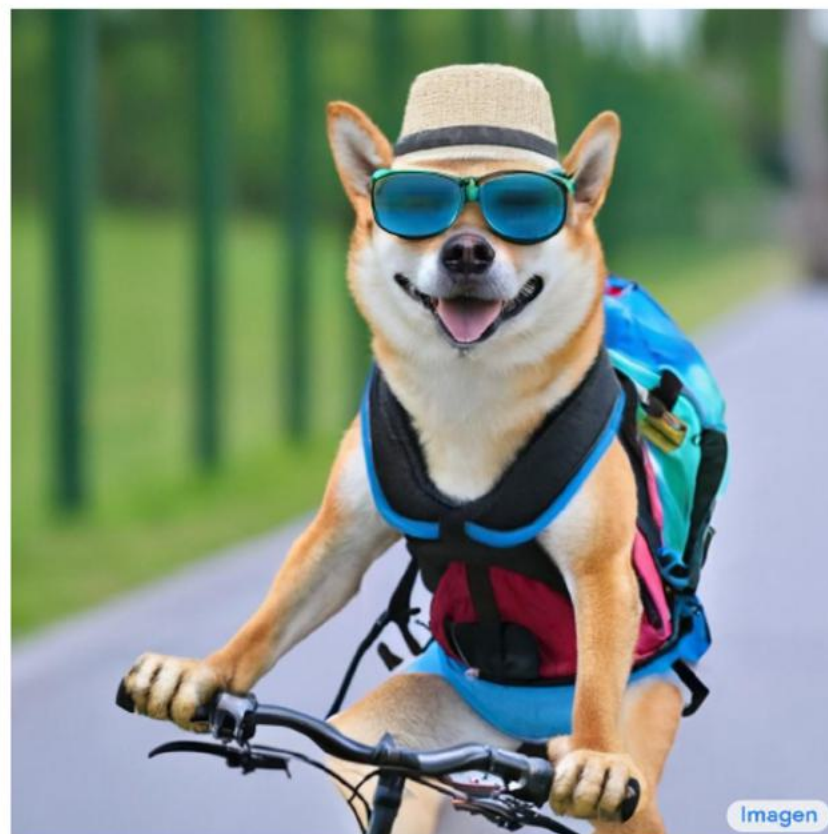
Figure A.4: Visualization of Imagen. Imagen uses a frozen text encoder to encode the input text into text embeddings. A conditional diffusion model maps the text embedding into a 64×64 image. Imagen further utilizes text-conditional super-resolution diffusion models to upsample the image, first $64 \times 64 \rightarrow 256 \times 256$, and then $256 \times 256 \rightarrow 1024 \times 1024$.

Imagen Key Observations / Ideas

- LLMs trained on text-only (e.g. T5) very effective at encoding text for image synthesis
- Increasing LLM size boost sample fidelity and image-text alignment more than increasing image diffusion model size
- Efficient U-Net: more model parameters at low res



Sprouts in the shape of text 'Imagen' coming out of a fairytale book.



A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat.



A high contrast portrait of a very happy fuzzy panda dressed as a chef in a high end kitchen making dough. There is a painting of flowers on the wall behind him.



Teddy bears swimming at the Olympics 400m Butterfly event.



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



A cute sloth holding a small treasure chest. A bright golden glow is coming from the chest.

Imagen (Ours)



DALL-E 2 [54]



New York Skyline with Hello World written with fireworks on the sky.

Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - Prediction Space / Loss: x , ϵ , v
 - Issues with Common Noise Schedules
 - Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers
 - Image Editing
 - Using scores in diffusion models: text-to-3D, text-to-SVG
 - Faster Sampling
 - Diffusion as Pre-Training
 - Other Fields: visuomotor control, materials discovery

DDIM Sampler

Consider the noising forward process $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$

Let $\alpha_t = \sqrt{\bar{\alpha}_t}, \sigma_t = \sqrt{1 - \bar{\alpha}_t}$

Then $x_t = \alpha_t x_0 + \sigma_t \epsilon$

DDIM Sampler

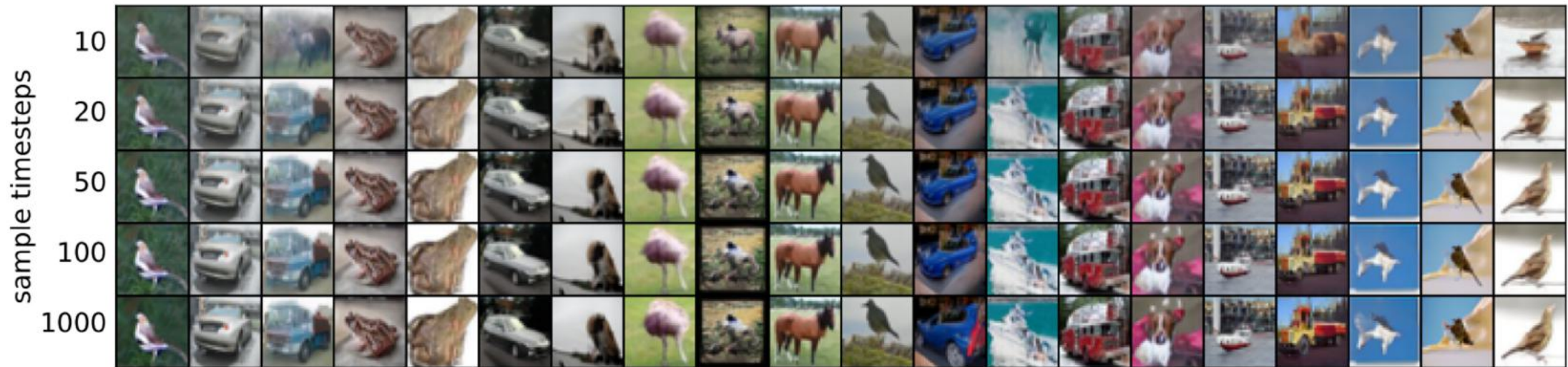
Deterministic DDIM update:

$$x_0 = \frac{x_t - \sigma_t \epsilon}{\alpha_t}$$

$$x_{t-1} = \alpha_{t-1} \underbrace{\left(\frac{x_t - \sigma_t \hat{\epsilon}(x_t)}{\alpha_t} \right)}_{\text{“predicted } x_0\text{”}} + \underbrace{\sigma_{t-1} \hat{\epsilon}(x_t)}_{\text{“direction pointing to } x_t\text{”}}$$

DDIM Sampler

Can choose a different step size during inference than training



Most models (e.g. SD) can generate good samples even with 50 steps

Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - **Prediction Space / Loss: x , ϵ , v**
 - Issues with Common Noise Schedules
 - Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers
 - Image Editing
 - Using scores in diffusion models: text-to-3D, text-to-SVG
 - Faster Sampling
 - Diffusion as Pre-Training
 - Other Fields: visuomotor control, materials discovery

Different Reparameterizations

Epsilon - space (DDPM)

$$x_t = \alpha_t x_0 + \sigma_t \epsilon$$

$$\mathcal{L}(x_0, t) = \|\epsilon - \hat{\epsilon}_\theta(\alpha_t x_0 + \sigma_t \epsilon)\|_2^2$$

Different Reparameterizations

x - space

$$x_t = \alpha_t x_0 + \sigma_t \epsilon$$

$$\mathcal{L}(x_0, t) = \|x_0 - \hat{x}_\theta(\alpha_t x_0 + \sigma_t \epsilon)\|_2^2$$

Different Reparameterizations

Relationship between x-space and epsilon-space

$$\begin{aligned}\mathcal{L}(x_0, t) &= \|x_0 - \hat{x}_\theta(x_t)\|_2^2 \\ &= \left\| \frac{x_t - \sigma_t \epsilon}{\alpha_t} - \frac{x_t - \sigma_t \hat{\epsilon}_\theta(x_t)}{\alpha_t} \right\|_2^2 \\ &= \frac{\sigma_t^2}{\alpha_t^2} \|\epsilon - \hat{\epsilon}_\theta(x_t)\|_2^2\end{aligned}$$

$$x_t = \alpha_t x_0 + \sigma_t \epsilon$$

$$x_0 = \frac{x_t - \sigma_t \epsilon}{\alpha_t}$$

$$\frac{\alpha_t^2}{\sigma_t^2} \|x_0 - \hat{x}_\theta(x_t)\|_2^2 = \|\epsilon - \hat{\epsilon}_\theta(x_t)\|_2^2$$

Different Reparameterizations

How about something else?

Note that DDPM defines: $\alpha_t = \sqrt{\bar{\alpha}_t}$, $\sigma_t = \sqrt{1 - \bar{\alpha}_t}$

(α_t, σ_t) defines vectors on the unit circle.

Instead of DDIM by t , can we do by angle $\phi = \arctan(\sigma_t/\alpha_t)$

Different Reparameterizations

We can define the forward process as:

$$x_\phi = \cos(\phi)x_0 + \sin(\phi)\epsilon$$

Then, define the “velocity” as:

$$v_\phi = \frac{dx_\phi}{d\phi} = \cos(\phi)\epsilon - \sin(\phi)x_0$$

DDIM update rule is then:

$$x_{\phi_t - \delta} = \cos(\delta)z_{\phi_t} - \sin(\delta)\hat{v}_\theta(x_{\phi_t})$$

Different Reparameterizations

v - space

$$x_t = \alpha_t x_0 + \sigma_t \epsilon$$

$$\mathcal{L}(x_0, t) = \|v - \hat{v}(\alpha_t x_0 + \sigma_t \epsilon)\|_2^2$$

where

$$v = \alpha_t \epsilon - \sigma_t x_0$$

Different Reparameterizations

Relationship between x-space and v-space

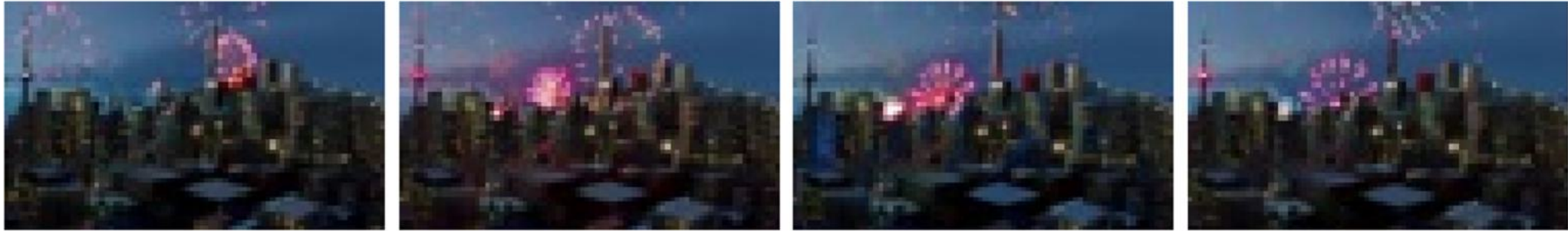
$$x = \alpha_t x_t - \sigma_t v$$

$$\begin{aligned}\mathcal{L}(x_0, t) &= \|x_0 - \hat{x}_\theta(x_t)\|_2^2 \\ &= \|(\alpha_t x_t - \sigma_t v) - (\alpha_t x_t - \sigma_t \hat{v}_\theta(x_t))\|_2^2 \\ &= \sigma_t^2 \|v - \hat{v}_\theta(x_t)\|_2^2\end{aligned}$$

$$\left(1 + \frac{\alpha_t^2}{\sigma_t^2}\right) \|x_0 - \hat{x}_\theta(x_t)\|_2^2 = \|v - \hat{v}_\theta(x_t)\|_2^2$$

Benefits of v-space

80×48 input video frames



SSR to 320×192 with ϵ -prediction



SSR to 320×192 with v-prediction

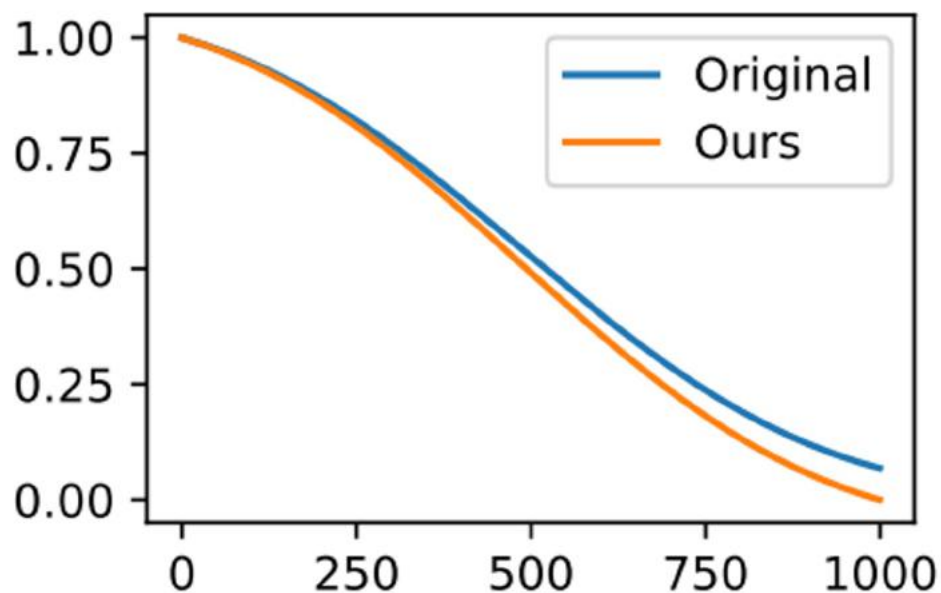


Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - Prediction Space / Loss: x , ϵ , v
 - **Issues with Common Noise Schedules**
 - Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers
 - Image Editing / Video Editing
 - Using scores in diffusion models: text-to-3D, text-to-SVG, Video Adapter
 - Faster Sampling
 - Diffusion as Pre-Training
 - Other Fields: visuomotor control, materials discovery

Issues with Common Noise Schedules

Most diffusion noise schedules do not end at 0 for $t = T$



(b) $\sqrt{\bar{\alpha}_t}$

$$\alpha_t = \sqrt{\bar{\alpha}_t}, \sigma_t = \sqrt{1 - \bar{\alpha}_t}$$

$$x_t = \alpha_t x_0 + \sigma_t \epsilon$$

Issues with Common Noise Schedules

DDPM uses epsilon-space for loss. What happens when $t = T$?

If $t = T$, then $\alpha_t = 0$ and $\sigma_t = 1$ (pure gaussian noise)

$$\mathcal{L}(x_0, t) = \|\epsilon - \hat{\epsilon}_\theta(0 * x_0 + 1 * \epsilon)\|_2^2$$

The objective becomes a trivial task (learn the identity function)

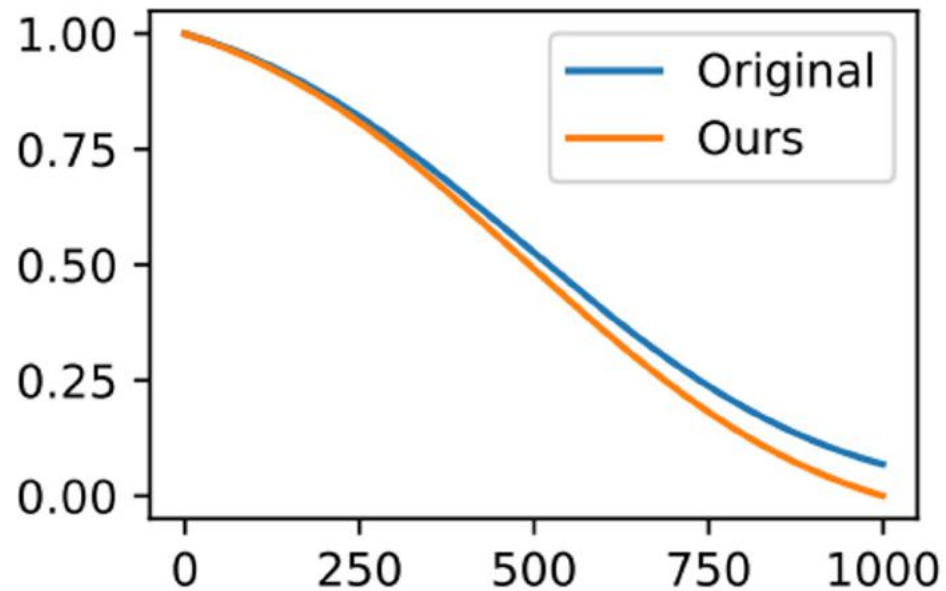
Issues with Common Noise Schedules

Why is this an issue?

- Leaks global information about x_0 during training since it is not fully noised
- During inference, we sample pure noise. The model will try to interpret signal where there is none.

Zero-Terminal SNR

We can enforce 0 terminal SNR by adjusting the noise schedule



(b) $\sqrt{\bar{\alpha}_t}$

$$\alpha_t = \sqrt{\bar{\alpha}_t}, \sigma_t = \sqrt{1 - \bar{\alpha}_t}$$

$$x_t = \alpha_t x_0 + \sigma_t \epsilon$$

Zero-Terminal SNR

But we can't use epsilon-space loss - v space to the rescue!

At $t = T$, v-space loss is

$$\mathcal{L}(x_0, t) = \|0 * \epsilon - 1 * x_0 - \hat{v}_\theta(0 * x_0 + 1 * \epsilon)\|_2^2 = \|x_0 - \hat{v}_\theta(\epsilon)\|_2^2$$

Process:

- Re-scale noise schedule to enforce zero-terminal SNR
- Finetune diffusion model with v-space loss on new noise schedule

Zero-Terminal SNR

SD



SD + Zero SNR



(e) A bald eagle against a white background

Zero-Terminal SNR

Factorized Gen.



w/o Zero SNR



Elucidating the Design Space of Diffusion-Based Generative Models

Tero Karras
NVIDIA

Miika Aittala
NVIDIA

Timo Aila
NVIDIA

Samuli Laine
NVIDIA

Abstract

We argue that the theory and practice of diffusion-based generative models are currently unnecessarily convoluted and seek to remedy the situation by presenting a design space that clearly separates the concrete design choices. This lets us identify several changes to both the sampling and training processes, as well as preconditioning of the score networks. Together, our improvements yield new state-of-the-art FID of 1.79 for CIFAR-10 in a class-conditional setting and 1.97 in an unconditional setting, with much faster sampling (35 network evaluations per image) than prior designs. To further demonstrate their modular nature, we show that our design changes dramatically improve both the efficiency and quality obtainable with pre-trained score networks from previous work, including improving the FID of a previously trained ImageNet-64 model from 2.07 to near-SOTA 1.55, and after re-training with our proposed improvements to a new SOTA of 1.36.

Table 1: Specific design choices employed by different model families. N is the number of ODE solver iterations that we wish to execute during sampling. The corresponding sequence of time steps is $\{t_0, t_1, \dots, t_N\}$, where $t_N = 0$. If the model was originally trained for specific choices of N and $\{t_i\}$, the originals are denoted by M and $\{u_j\}$, respectively. The denoiser is defined as $D_\theta(\mathbf{x}; \sigma) = c_{\text{skip}}(\sigma)\mathbf{x} + c_{\text{out}}(\sigma)F_\theta(c_{\text{in}}(\sigma)\mathbf{x}; c_{\text{noise}}(\sigma))$; F_θ represents the raw neural network layers.

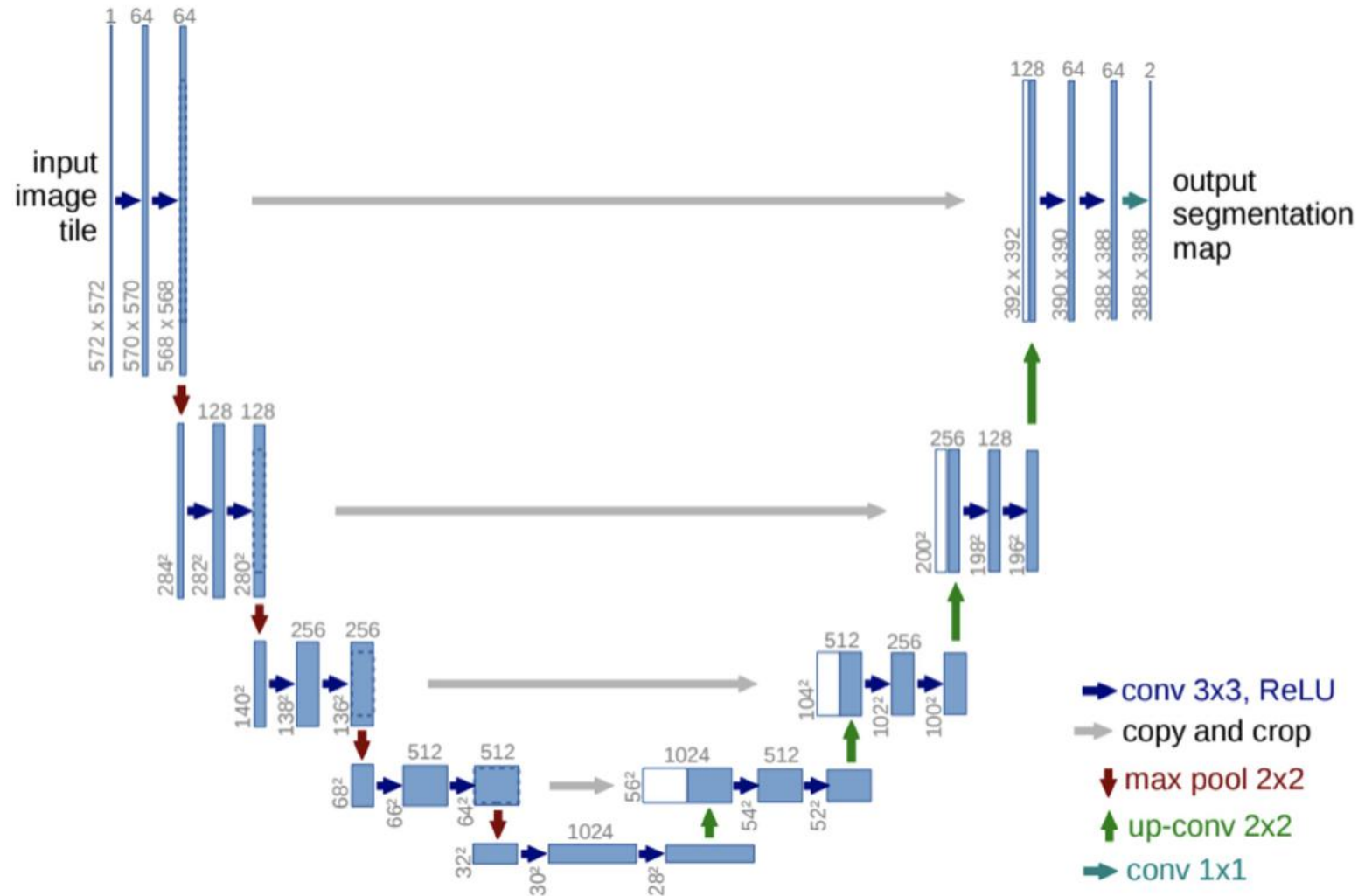
	VP [49]	VE [49]	iDDPM [37] + DDIM [47]	Ours (“EDM”)
Sampling (Section 3)				
ODE solver	Euler	Euler	Euler	2 nd order Heun
Time steps	$t_{i < N}$	$1 + \frac{i}{N-1}(\epsilon_s - 1)$	$\sigma_{\max}^2 (\sigma_{\min}^2 / \sigma_{\max}^2)^{\frac{i}{N-1}}$	$(\sigma_{\max}^{\frac{1}{\rho}} + \frac{i}{N-1}(\sigma_{\min}^{\frac{1}{\rho}} - \sigma_{\max}^{\frac{1}{\rho}}))^\rho$
Schedule	$\sigma(t)$	$\sqrt{e^{\frac{1}{2}\beta_d t^2 + \beta_{\min} t} - 1}$	\sqrt{t}	t
Scaling	$s(t)$	$1/\sqrt{e^{\frac{1}{2}\beta_d t^2 + \beta_{\min} t}}$	1	1
Network and preconditioning (Section 5)				
Architecture of F_θ	DDPM++	NCSN++	DDPM	(any)
Skip scaling $c_{\text{skip}}(\sigma)$	1	1	1	$\sigma_{\text{data}}^2 / (\sigma^2 + \sigma_{\text{data}}^2)$
Output scaling $c_{\text{out}}(\sigma)$	$-\sigma$	σ	$-\sigma$	$\sigma \cdot \sigma_{\text{data}} / \sqrt{\sigma_{\text{data}}^2 + \sigma^2}$
Input scaling $c_{\text{in}}(\sigma)$	$1/\sqrt{\sigma^2 + 1}$	1	$1/\sqrt{\sigma^2 + 1}$	$1/\sqrt{\sigma^2 + \sigma_{\text{data}}^2}$
Noise cond. $c_{\text{noise}}(\sigma)$	$(M-1)\sigma^{-1}(\sigma)$	$\ln(\frac{1}{2}\sigma)$	$M-1 - \arg \min_j u_j - \sigma $	$\frac{1}{4} \ln(\sigma)$
Training (Section 5)				
Noise distribution	$\sigma^{-1}(\sigma) \sim \mathcal{U}(\epsilon_t, 1)$	$\ln(\sigma) \sim \mathcal{U}(\ln(\sigma_{\min}), \ln(\sigma_{\max}))$	$\sigma = u_j, j \sim \mathcal{U}\{0, M-1\}$	$\ln(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$
Loss weighting $\lambda(\sigma)$	$1/\sigma^2$	$1/\sigma^2$	$1/\sigma^2$ (note: *)	$(\sigma^2 + \sigma_{\text{data}}^2) / (\sigma \cdot \sigma_{\text{data}})^2$
Parameters				
	$\beta_d = 19.9, \beta_{\min} = 0.1$	$\sigma_{\min} = 0.02$	$\bar{\alpha}_j = \sin^2(\frac{\pi}{2} \frac{j}{M(C_2+1)})$	$\sigma_{\min} = 0.002, \sigma_{\max} = 80$
	$\epsilon_s = 10^{-3}, \epsilon_t = 10^{-5}$	$\sigma_{\max} = 100$	$C_1 = 0.001, C_2 = 0.008$	$\sigma_{\text{data}} = 0.5, \rho = 7$
	$M = 1000$		$M = 1000, j_0 = 8^\dagger$	$P_{\text{mean}} = -1.2, P_{\text{std}} = 1.2$

* iDDPM also employs a second loss term L_{vib} † In our tests, $j_0 = 8$ yielded better FID than $j_0 = 0$ used by iDDPM

Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - Prediction Space / Loss: x , ϵ , v
 - Issues with Common Noise Schedules
 - **Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers**
 - Image Editing
 - Using scores in diffusion models: text-to-3D, text-to-SVG
 - Faster Sampling
 - Diffusion as Pre-Training
 - Other Fields: visuomotor control, materials discovery

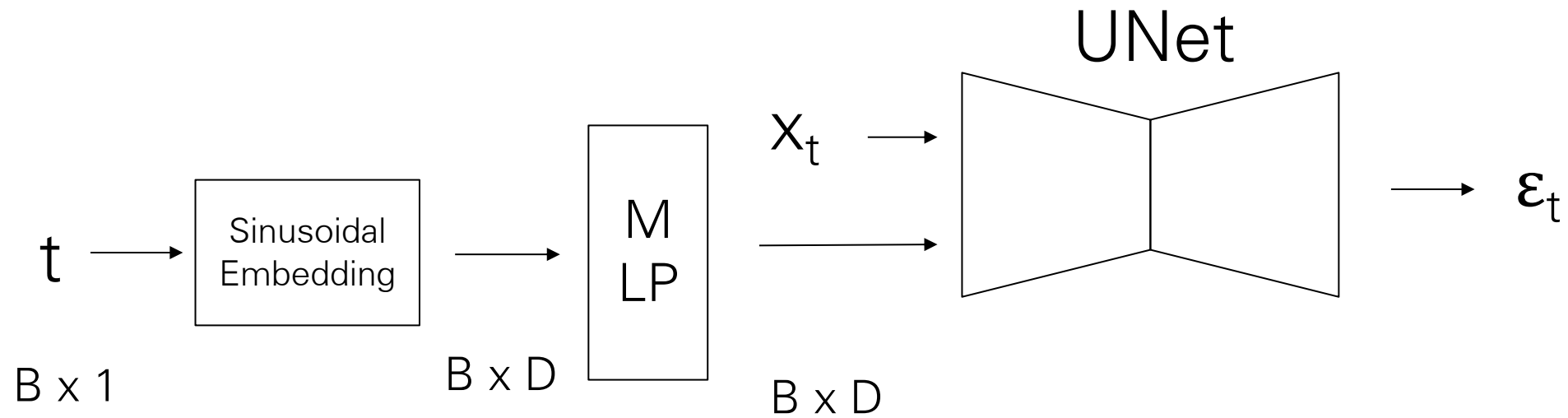
UNet - Base Model



Additionally
add some self-
attention
layers at lower
resolutions

UNet - Conditioning

Condition on diffusion timestep t



UNet - Conditioning

Shapes: h ($H \times W \times D$), t_{emb} ($1 \times 1 \times D$)

Method 1: Simple addition

$$h = h + t_{emb}$$

Method 2: Modulation parameters

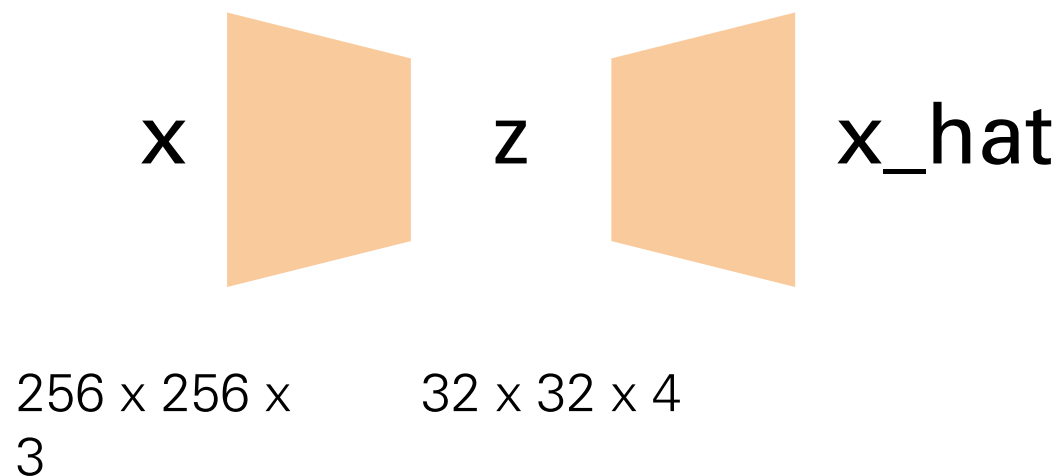
$$h = \text{GroupNorm}(h) * (1 + f_{scale}(t_{emb})) + f_{shift}(t_{emb})$$

Latent Diffusion Models

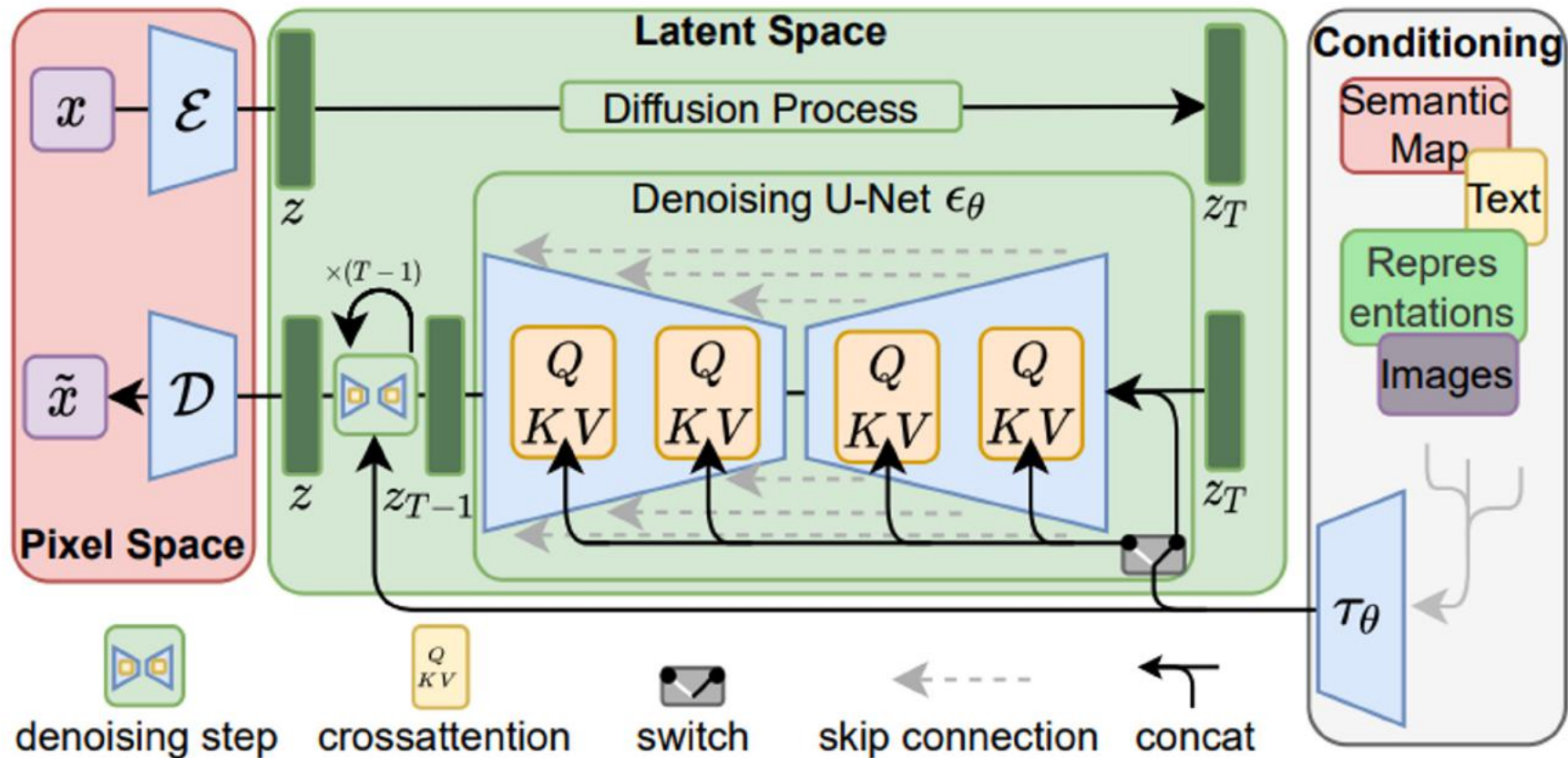
Pixels can still be expensive.

Can we learn a latent space like we did for AR models (VQGAN)?

Learn a VAE: very low weighting on KL loss ($1e-6$)

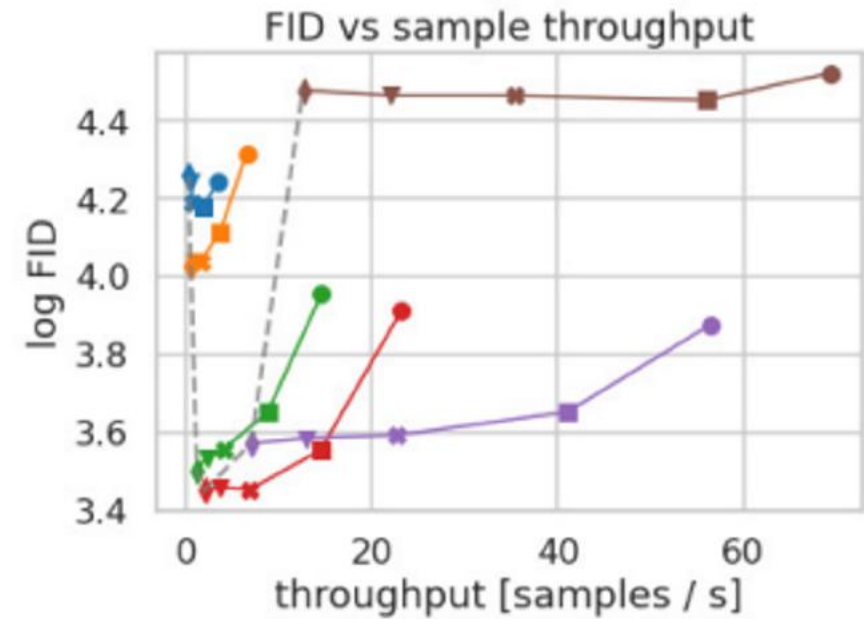
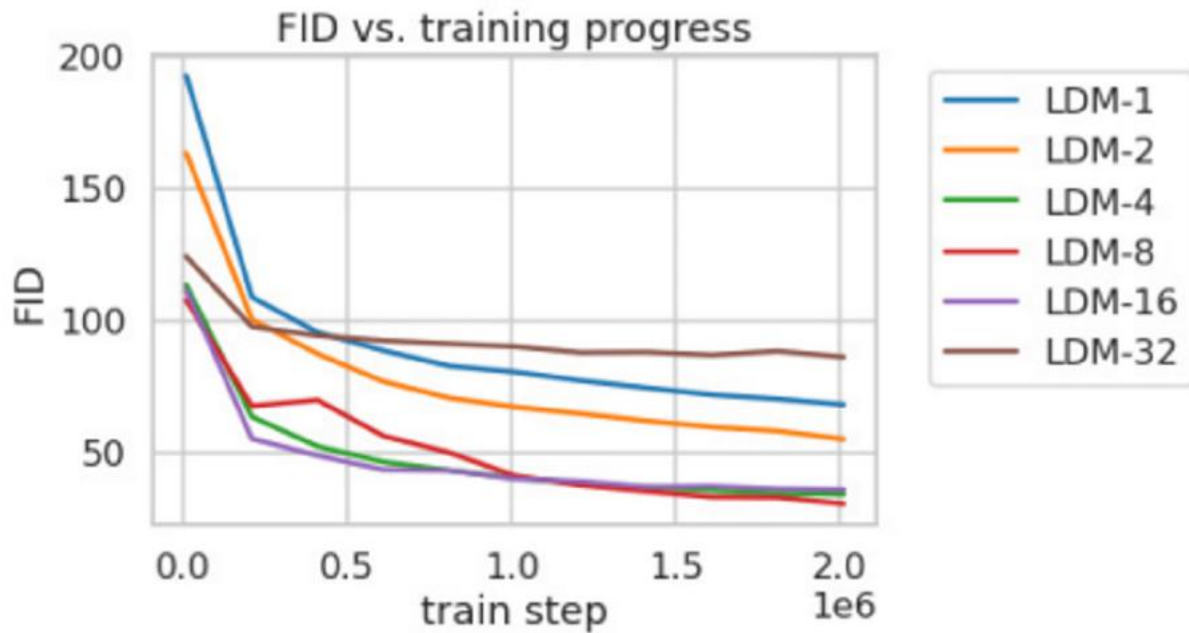


Latent Diffusion Model



Latent Diffusion Model

Speed - Quality Trade-off

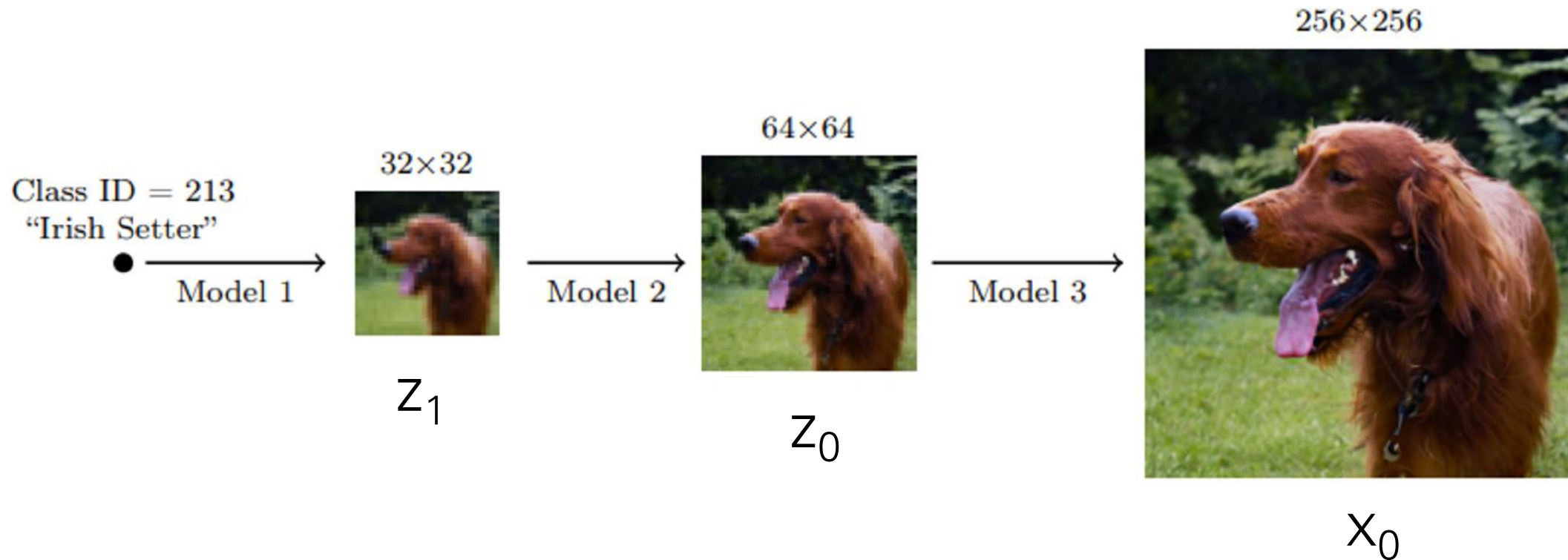


Stable Diffusion

- VAE that downsamples images by 8x spatially (256 -> 32, 512 > 64)
- UNet architecture with self-attention layers
- Text conditioning with cross attention on text features
 - Text features using CLIP text encoder



Cascaded Diffusion Models



$$p_{\theta}(x_0) = \int p_{\theta}(z_1)p_{\theta}(z_0 | z_1)p_{\theta}(x_0 | z_0)dz$$

Cascaded Diffusion Models

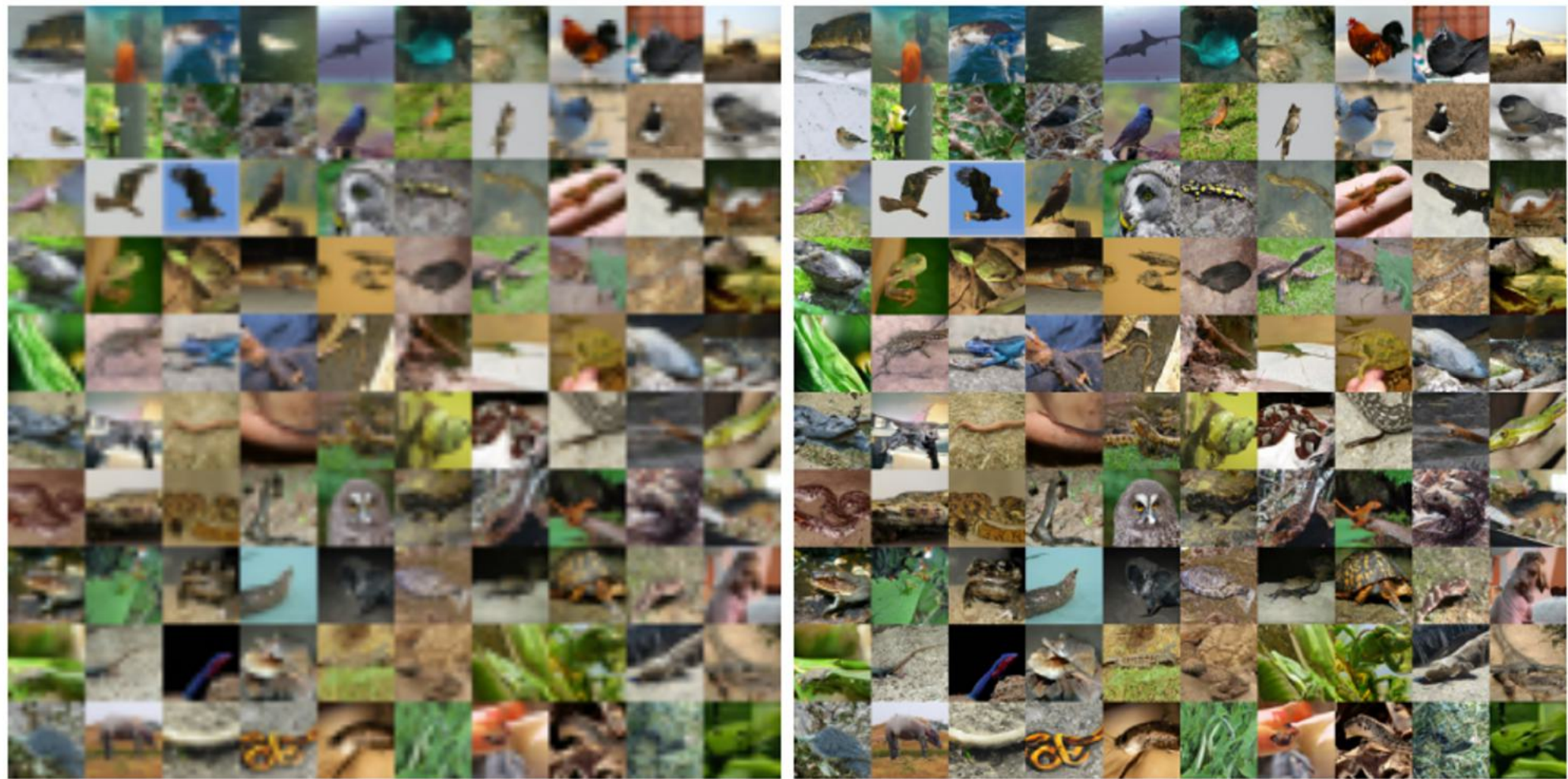
- Train 1 unconditional model, and the rest are low-resolution conditional super-resolution models
 - Condition on low-res image -> bilinear / bicubic upsample -> concatenate to noised input
- Models can be trained independently
- Hyperparameters can be tuned to be resolution specific

Cascaded Diffusion Models

Conditioning Augmentation: Augment the low-res input z

- Method 1: Blurring
 - Gaussian blurring filter
 - Good for models at 128, 256 resolution
- Method 2: Non-truncated Conditioning Augmentation
 - Train the super-resolution $p(x|z_0)$ model conditioned on samples from the prior $p(z_0)$ with added corruption from the forward process $q(z_s|z_0)$.
 - S (corruption / noise strength) is a hyperparameter
 - Good for < 128 resolution

Cascaded Diffusion Models



(a) 16×16 base

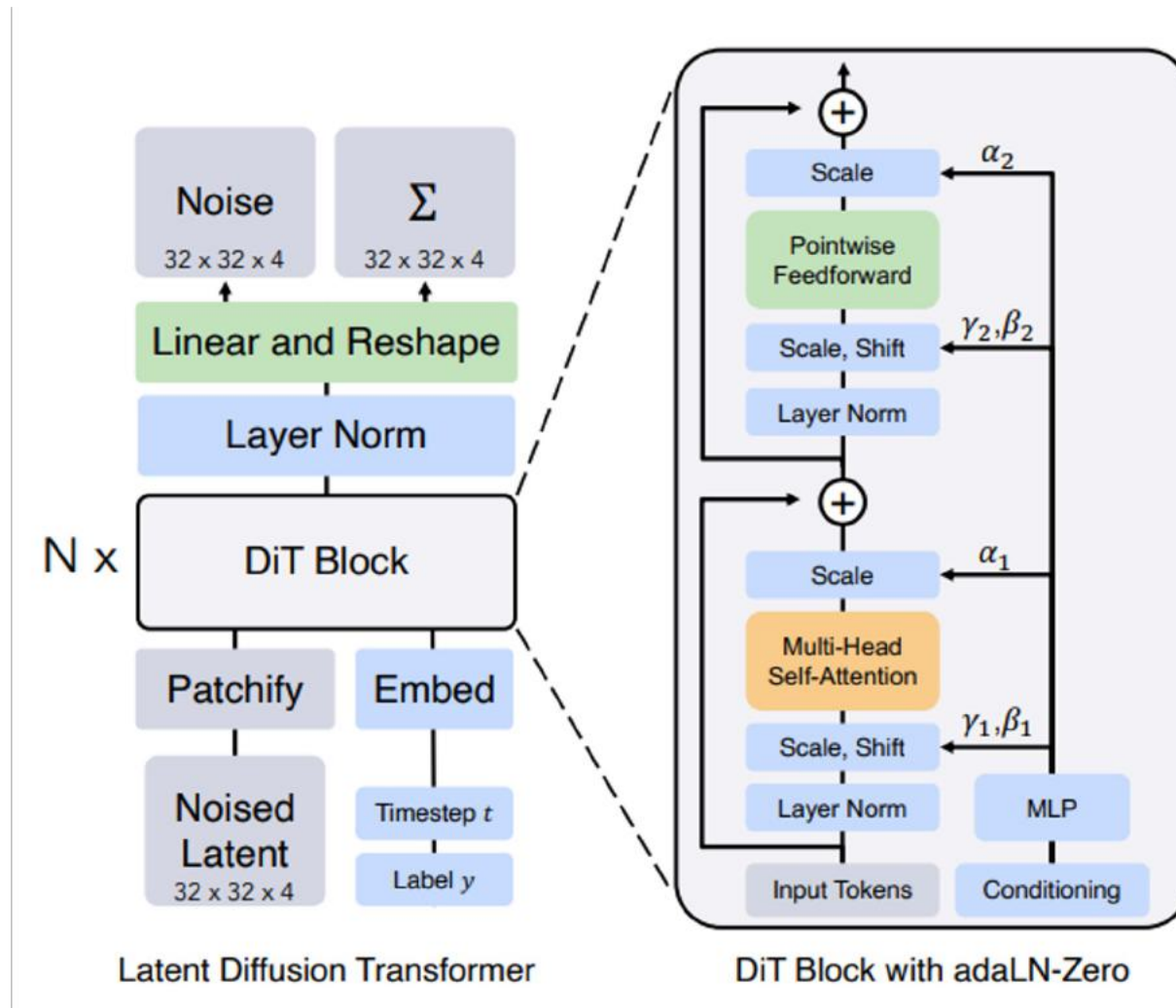
(b) $16 \times 16 \rightarrow 64 \times 64$ super-resolution, $s = 0$

Cascaded Diffusion Models

Large improvements in using conditioning augmentation

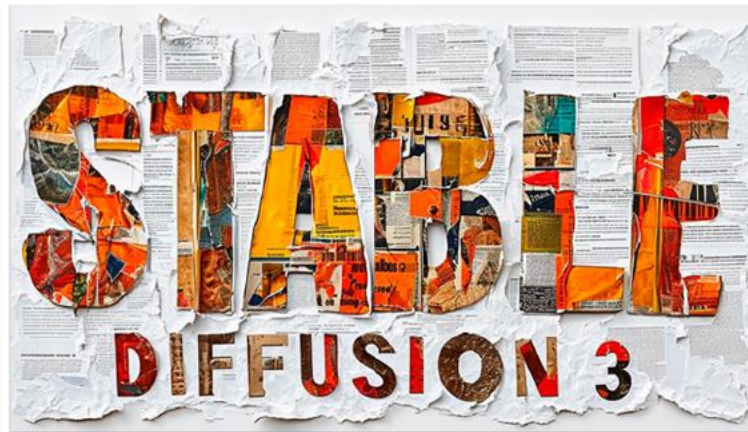
Conditioning	FID vs train	FID vs validation	IS
No cascading	2.35	2.91	52.72 \pm 1.15
<hr/> $16 \times 16 \rightarrow 64 \times 64$ cascading <hr/>			
$s = 0$	6.02	5.84	35.59 \pm 1.19
$s = 101$	3.41	3.67	44.72 \pm 1.12
$s = 1001$	2.13	2.79	54.47 \pm 1.05

Diffusion Transformers



Stable Diffusion 3

- 8B parameters - largest open-source text-image model to date
- Diffusion transformer architecture
- Uses Flow Matching [[paper](#)]

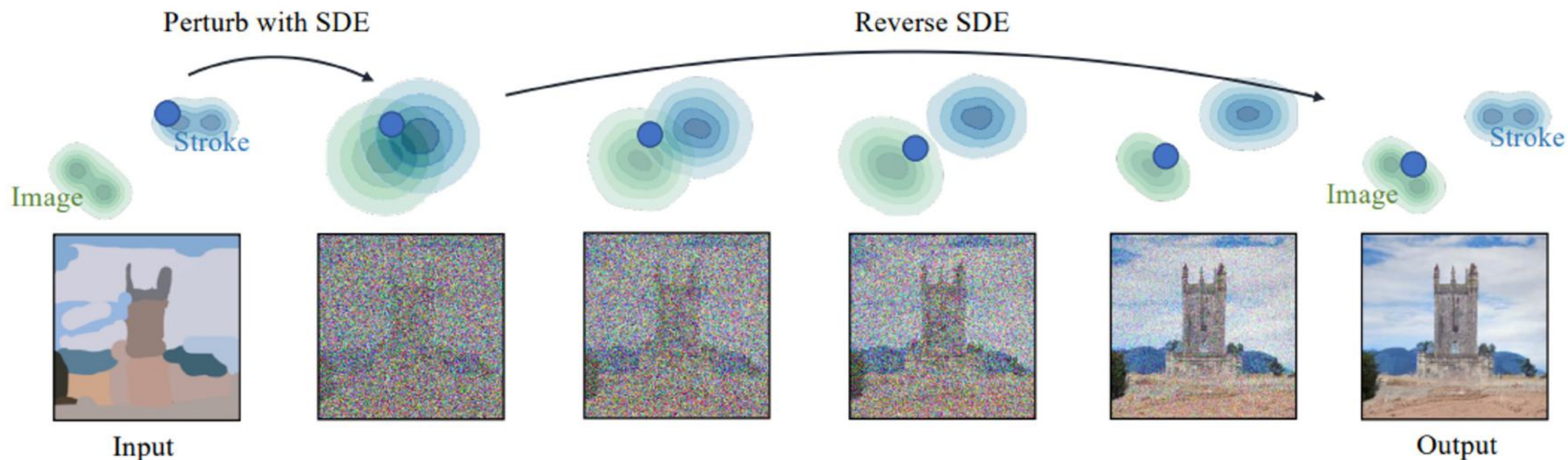


Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - Prediction Space / Loss: x , ϵ , v
 - Issues with Common Noise Schedules
 - Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers
 - **Image Editing**
 - Using scores in diffusion models: text-to-3D, text-to-SVG
 - Faster Sampling
 - Diffusion as Pre-Training
 - Other Fields: visuomotor control, materials discovery

SDEdit

Goal: High-level, editing of semantic image features while retaining global structure



SDEdit

Given some diffusion timestep t , image x_0 , edit caption c :

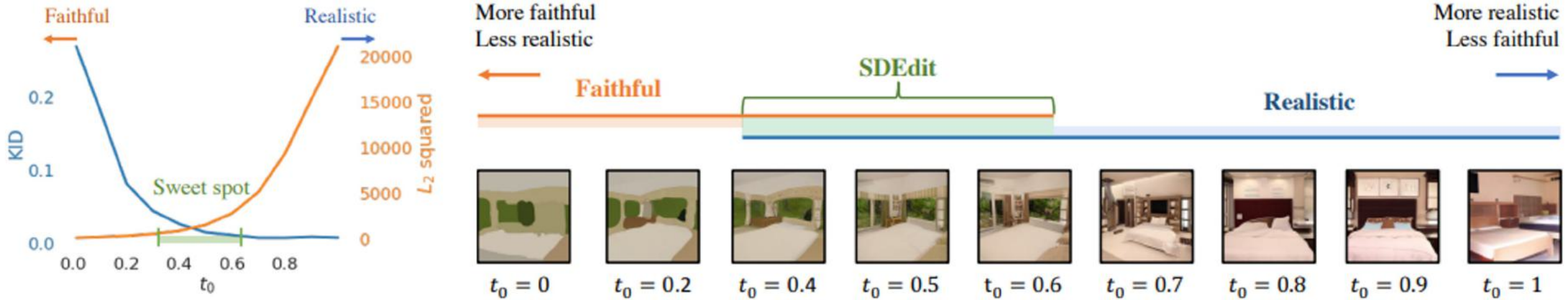
1) Apply the forward process to t : $q(x_t | x_0)$

2) Apply the reverse process to 0: $p(x_{t-1} | x_t, c)$ (t times)

SDEdit

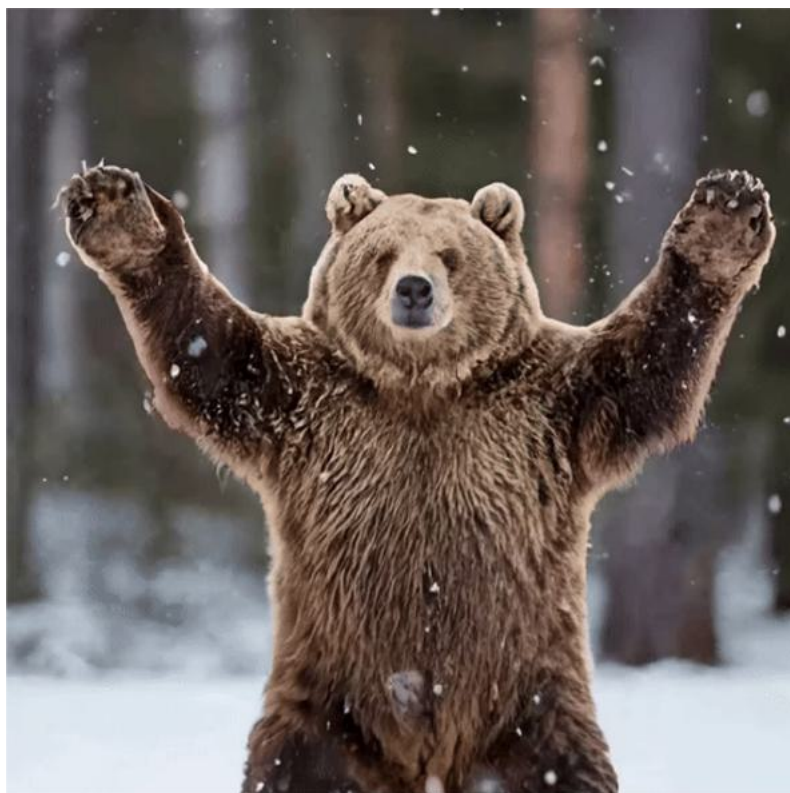
T is a hyperparameter

- Trade-off between faithfulness to original image, and alignment with target edit



Using SDEdit

Source Video



Made of wooden blocks



Using SDEdit

Source Video



Make of colorful toy bricks



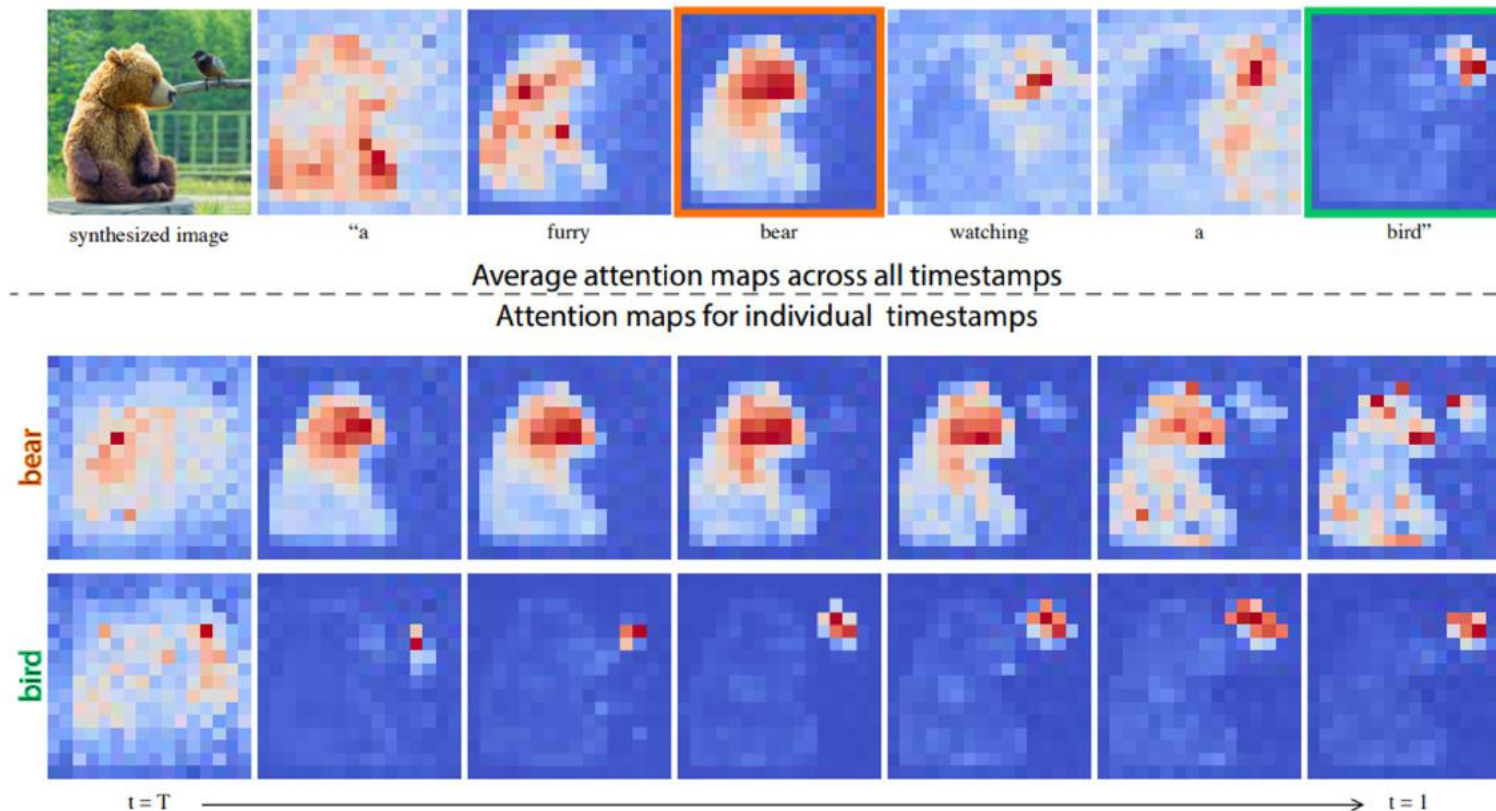
Prompt-to-Prompt (P2P)

Goal: Retain the fine-grained structure but change the content of an image

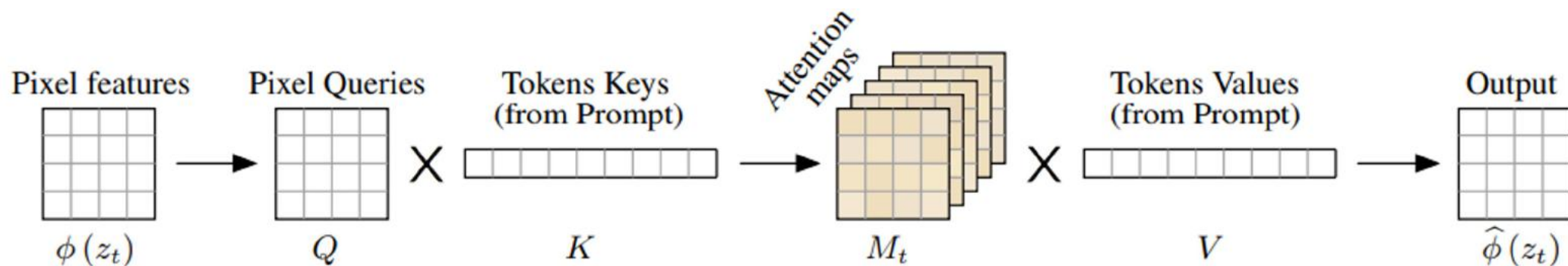


Prompt-to-Prompt (P2P)

Key Idea: Attention maps in UNets provide structure information

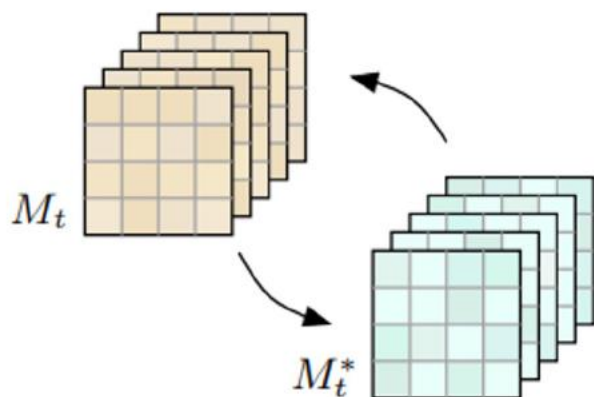


Prompt-to-Prompt (P2P)

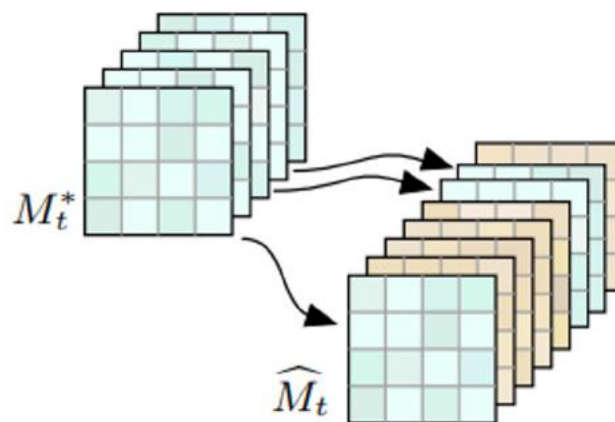


Text to Image Cross Attention

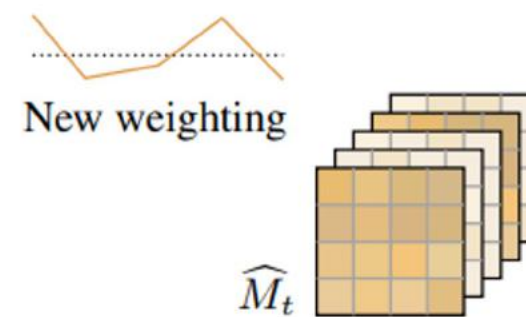
Cross Attention Control



Word Swap



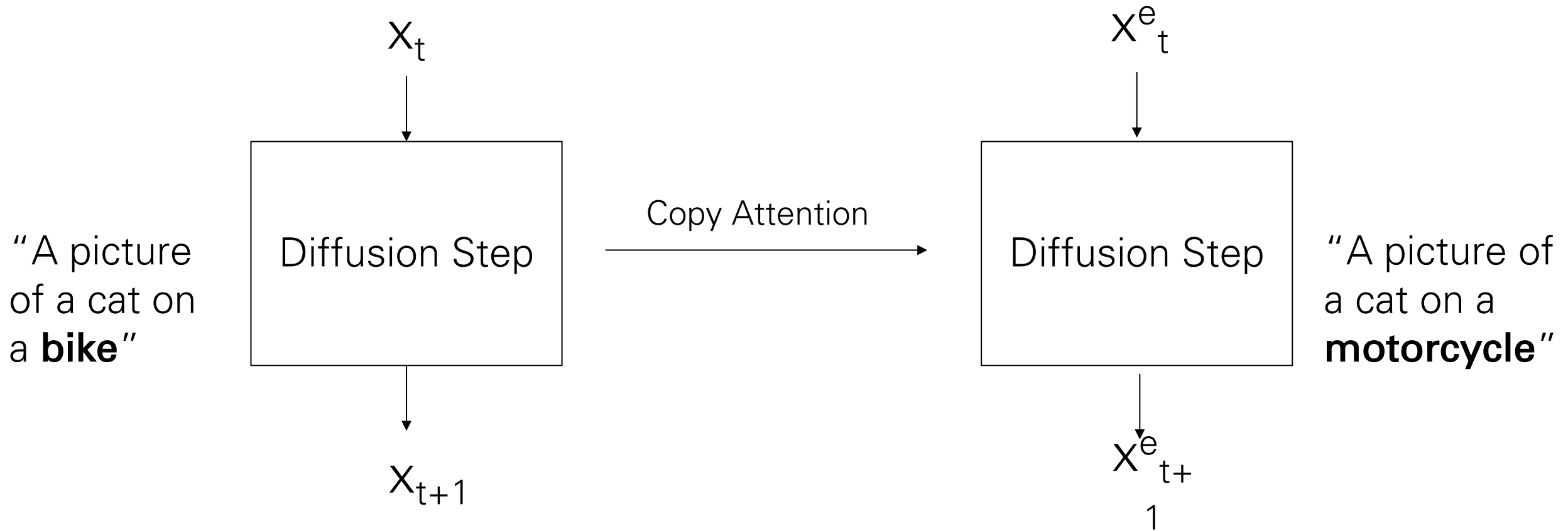
Adding a New Phrase



Attention Re-weighting

Sampling Process

Sample with original and edited prompt in parallel, and copy attention maps



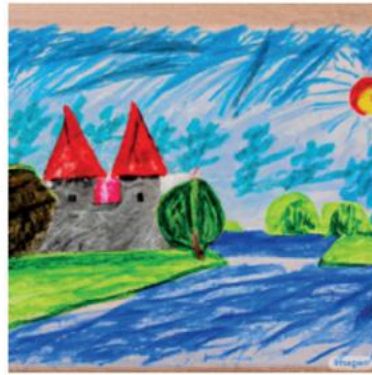
Prompt-to-Prompt (P2P)



"The boulevards are crowded today."



"Photo of a cat riding on a ~~bicycle.~~
car"



"Children drawing of a castle next to a river."



"a cake with decorations."

jelly beans

Real Image Editing

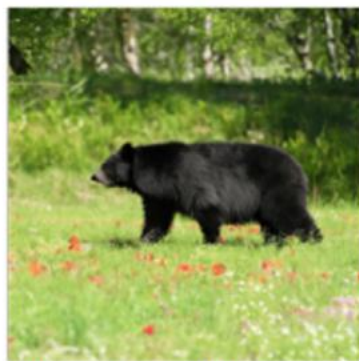
“A black bear is walking in the grass.”



real image



reconstructed



“...next to red flowers.”



“...when snow comes down.”



“while another black bear is watching.”



“Oil painting of...”

“Landscape image of trees in a valley...”



real image



reconstructed



“...at fall.”



“...at winter.”



“...at sunrise.”



“...at night.”

Failure Cases

Depends on DDIM reconstruction quality - especially bad with CFG

Input Image



DDIM Inversion



Input Image



DDIM Inversion



DreamBooth

Goal: Subject-driven image generation given 3-5 images of a specific subject



Input images



in the Acropolis



swimming



sleeping



in a doghouse



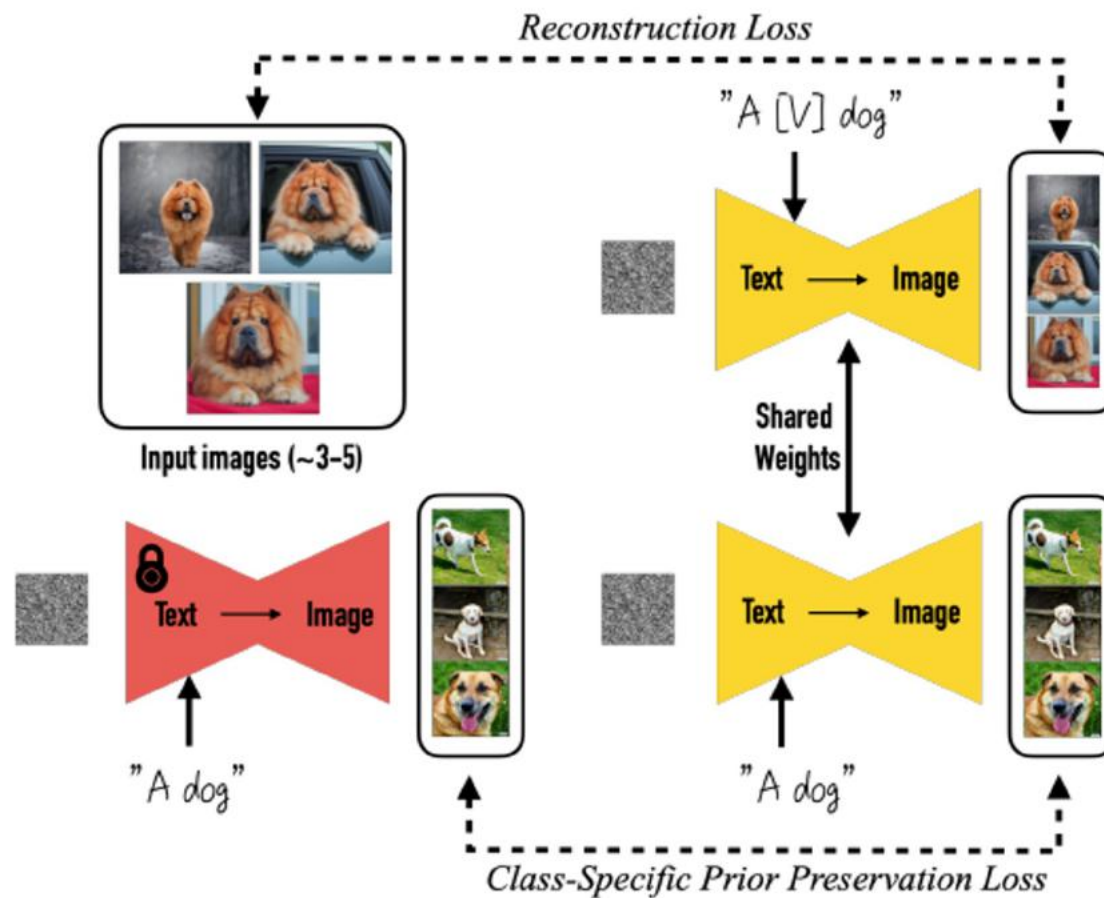
in a bucket



getting a haircut

DreamBooth

$$\mathbb{E}_{\mathbf{x}, \mathbf{c}, \epsilon, \epsilon', t} [w_t \|\hat{\mathbf{x}}_{\theta}(\alpha_t \mathbf{x} + \sigma_t \epsilon, \mathbf{c}) - \mathbf{x}\|_2^2 + \lambda w_{t'} \|\hat{\mathbf{x}}_{\theta}(\alpha_{t'} \mathbf{x}_{\text{pr}} + \sigma_{t'} \epsilon', \mathbf{c}_{\text{pr}}) - \mathbf{x}_{\text{pr}}\|_2^2],$$



DreamBooth



Input images



A [V] backpack in the Grand Canyon



A wet [V] backpack in water



A [V] backpack in Boston



A [V] backpack with the night sky



Input images



A [V] teapot floating in milk



A transparent [V] teapot with milk inside



A [V] teapot pouring tea



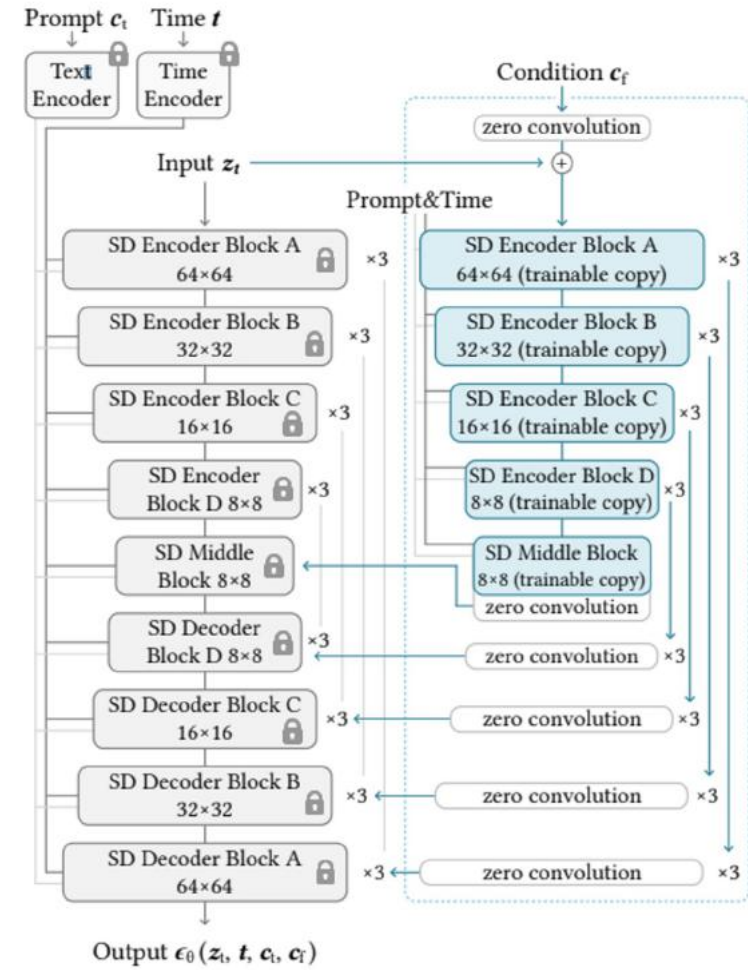
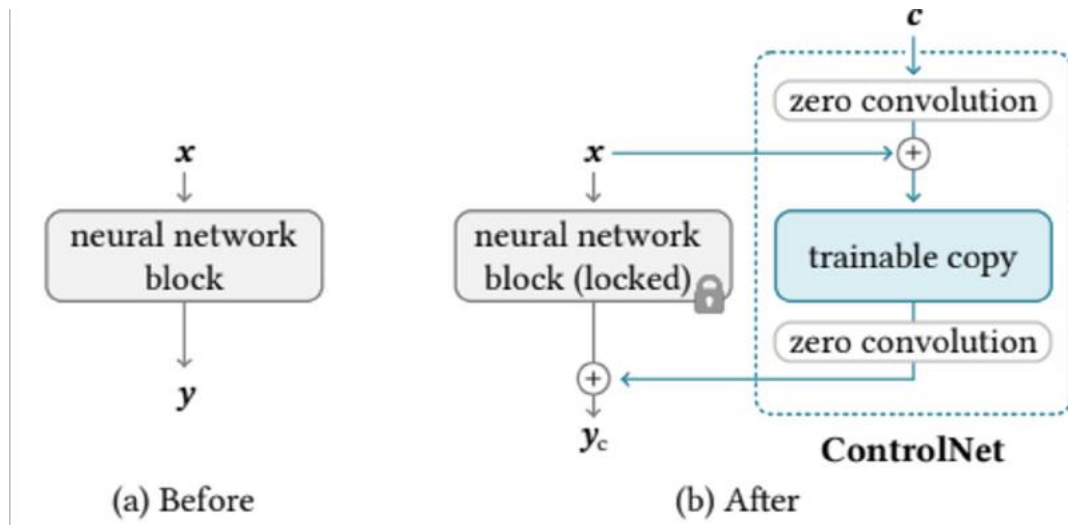
A [V] teapot floating in the sea

ControlNet

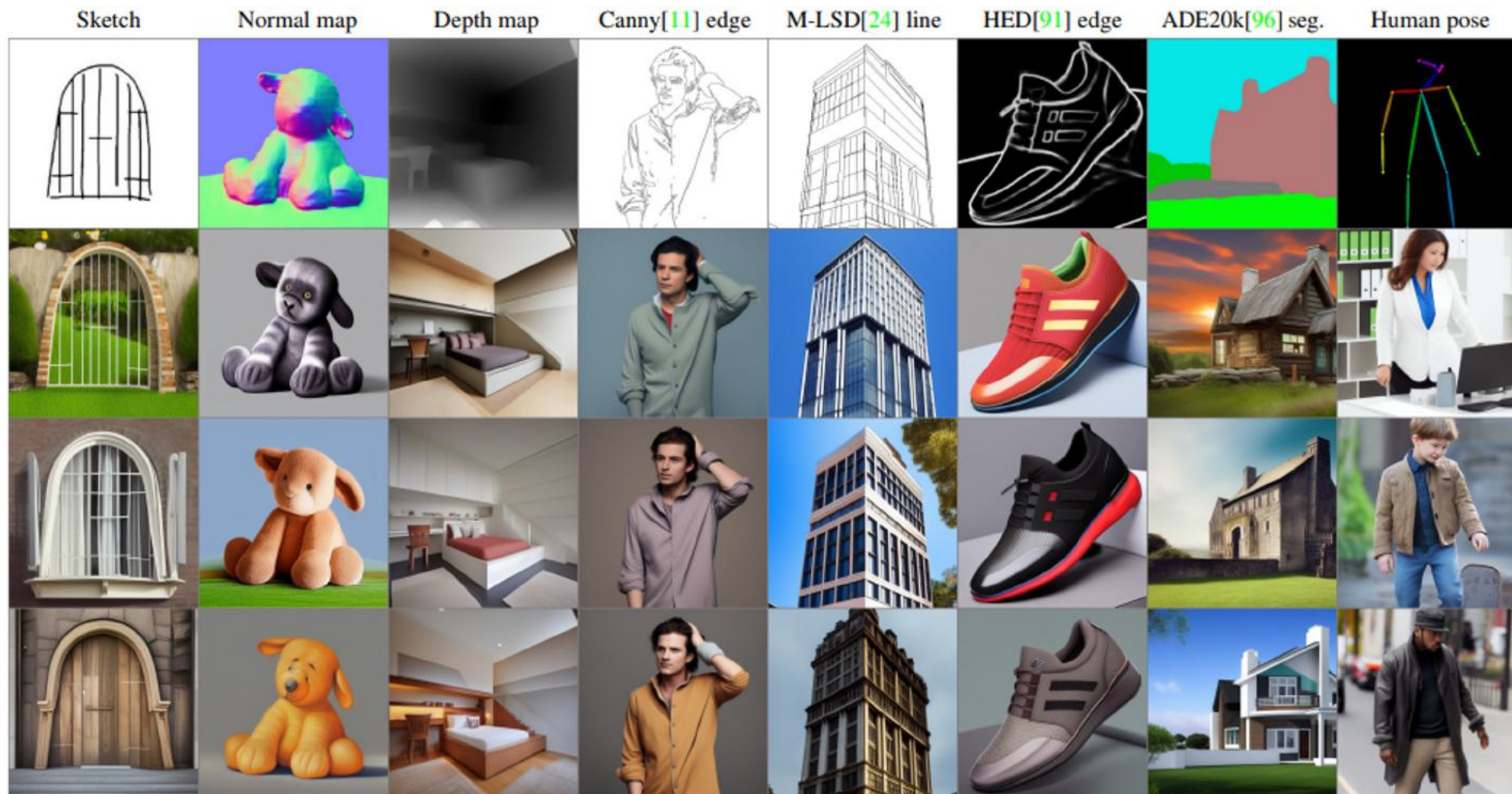
Goal: Finetune a diffusion model to include extra conditioning

- Canny edge map
- Depth
- Human Pose
- Lower Resolution Image
- Segmentation
- Sketch

ControlNet



ControlNet

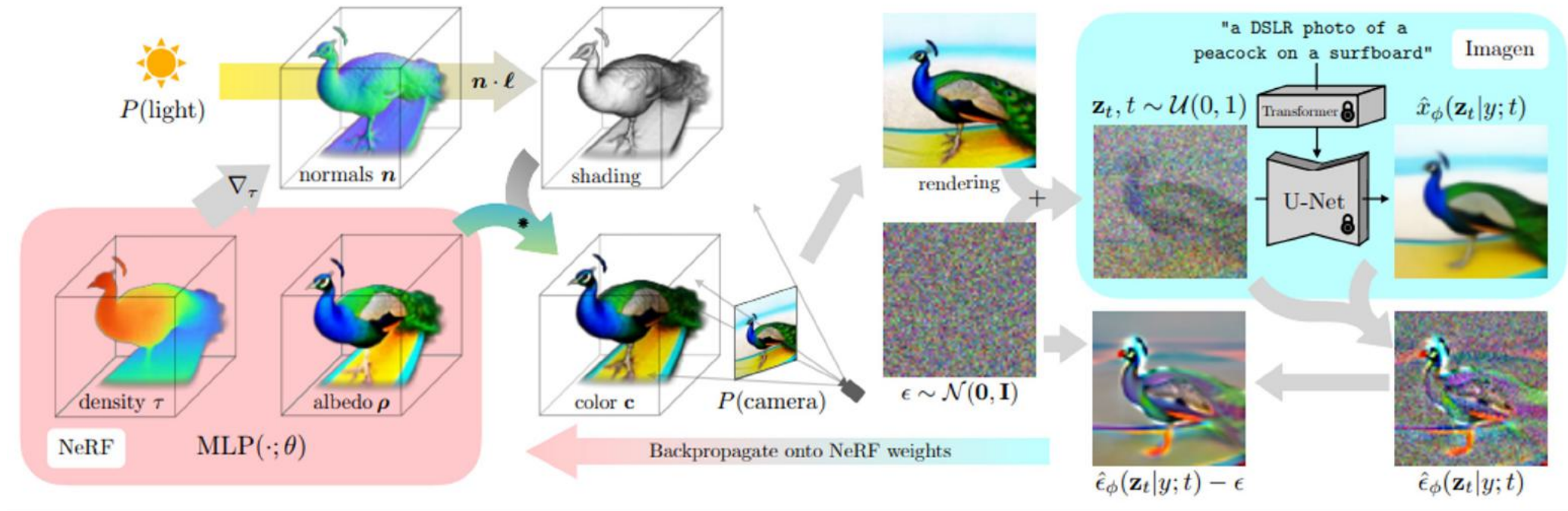


Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - Prediction Space / Loss: x , ϵ , v
 - Issues with Common Noise Schedules
 - Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers
 - Image Editing / Video Editing
 - **Using scores in diffusion models: text-to-3D, text-to-SVG**
 - Faster Sampling
 - Diffusion as Pre-Training
 - Other Fields: visuomotor control, materials discovery

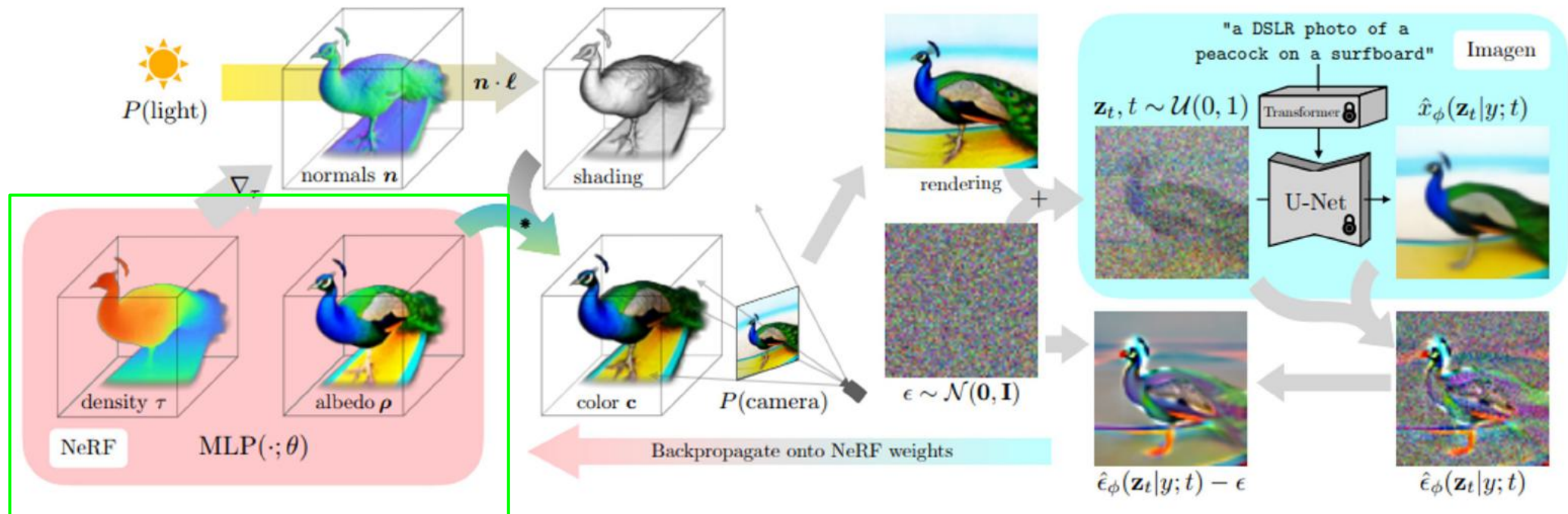
DreamFusion

How can you use a pretrained text-image model to generate 3D objects?



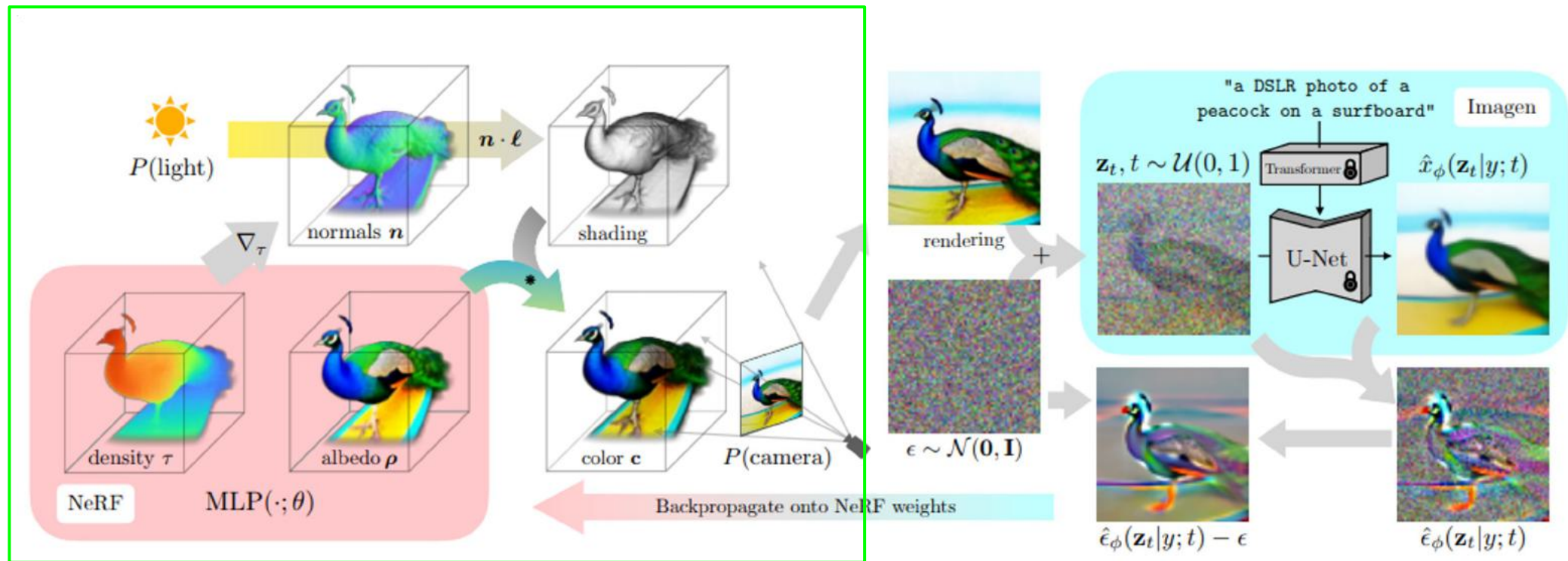
DreamFusion

Start with a randomly initialized parameterized 3D representation (NeRF)



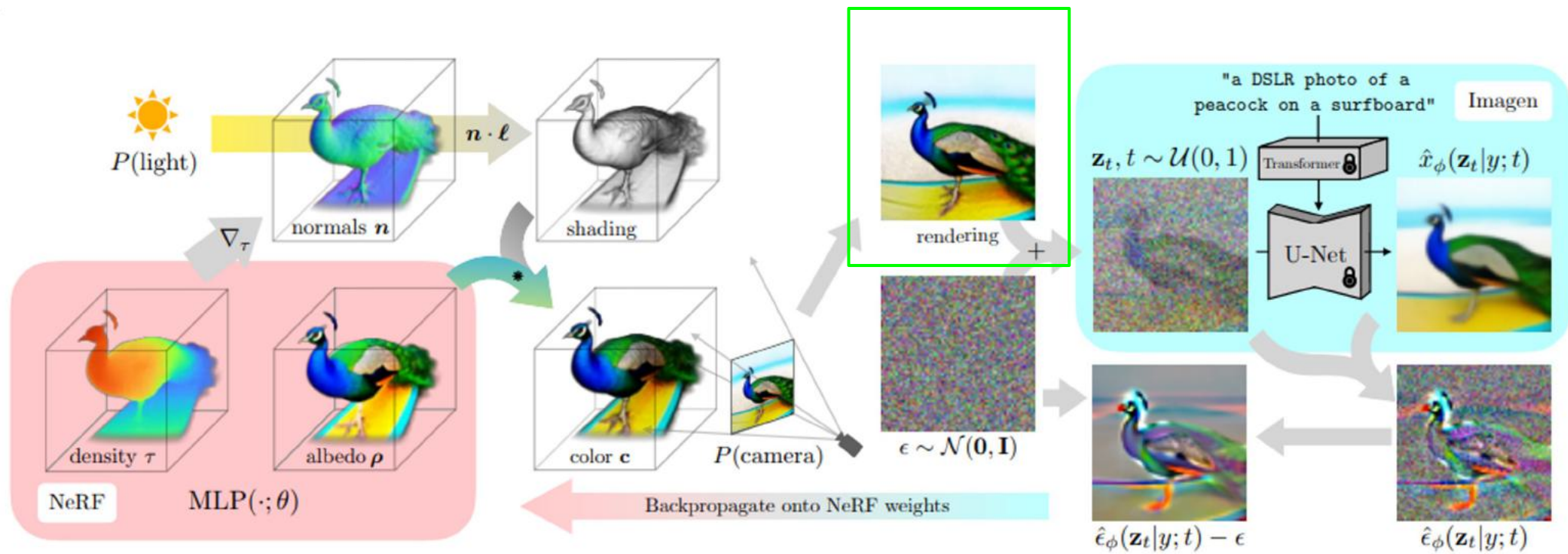
DreamFusion

Randomly sample camera angle and lighting



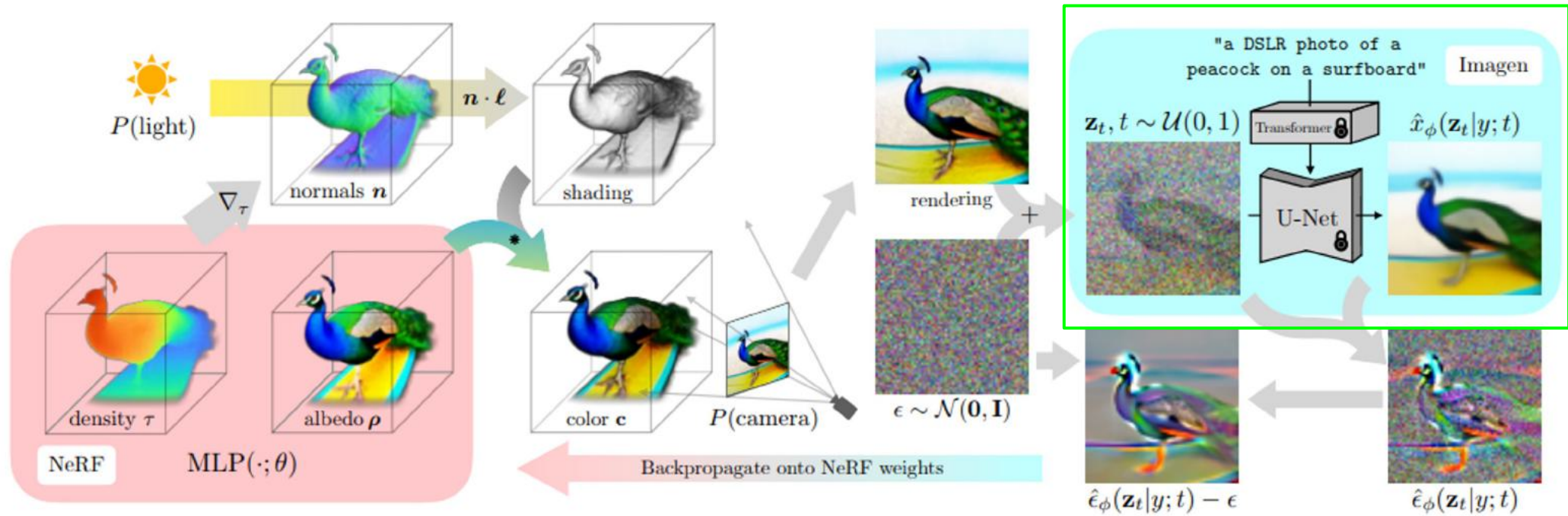
DreamFusion

Render the image (64 x 64)



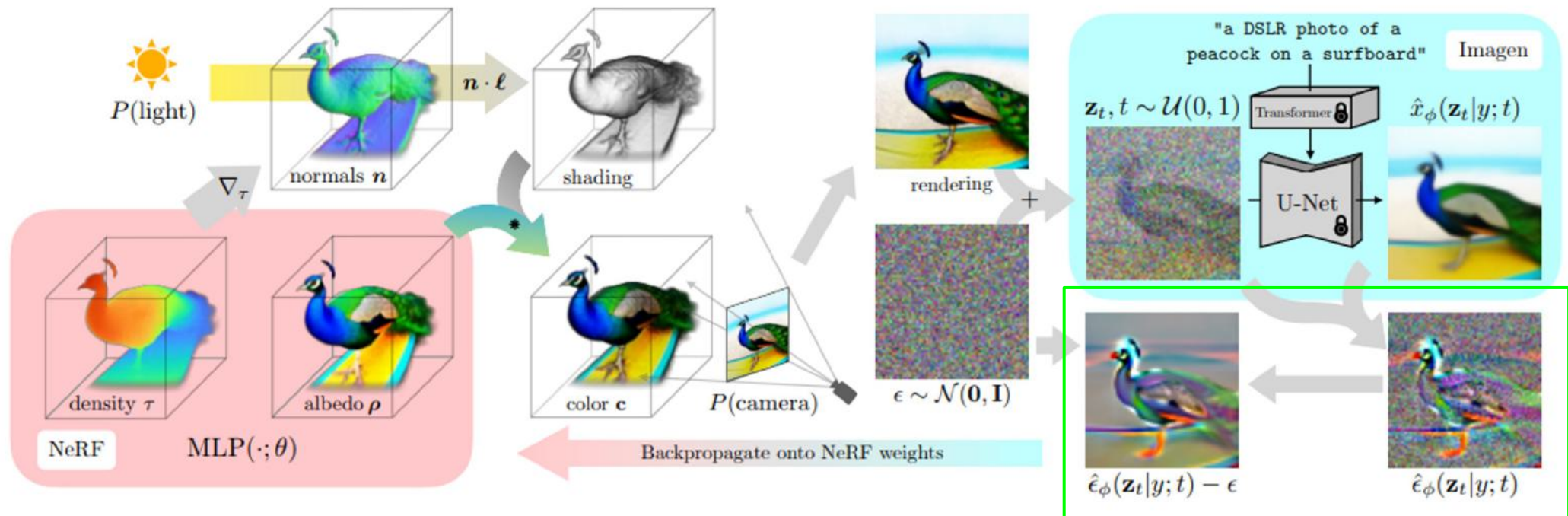
DreamFusion

Noise the image and feed it through the diffusion loss to predict the noise



DreamFusion

Compute the SDS loss and backpropagate



DreamFusion

Intuitively, we want the NeRF parameters to produce images that minimize the diffusion loss

$$\mathcal{L}_{\text{Diff}}(\phi, \mathbf{x}) = \mathbb{E}_{t \sim \mathcal{U}(0,1), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} [w(t) \|\epsilon_\phi(\alpha_t \mathbf{x} + \sigma_t \epsilon; t) - \epsilon\|_2^2]$$

$$\nabla_{\theta} \mathcal{L}_{\text{Diff}}(\phi, \mathbf{x} = g(\theta)) = \mathbb{E}_{t, \epsilon} \left[w(t) \underbrace{(\hat{\epsilon}_\phi(\mathbf{z}_t; y, t) - \epsilon)}_{\text{Noise Residual}} \underbrace{\frac{\partial \hat{\epsilon}_\phi(\mathbf{z}_t; y, t)}{\partial \mathbf{z}_t}}_{\text{U-Net Jacobian}} \underbrace{\frac{\partial \mathbf{x}}{\partial \theta}}_{\text{Generator Jacobian}} \right]$$

But the objective is generally brittle / expensive (backprop through the UNet)

DreamFusion

It works better to just remove the U-Net jacobian

$$\nabla_{\theta} \mathcal{L}_{\text{Diff}}(\phi, \mathbf{x} = g(\theta)) = \mathbb{E}_{t, \epsilon} \left[w(t) \underbrace{(\hat{\epsilon}_{\phi}(\mathbf{z}_t; y, t) - \epsilon)}_{\text{Noise Residual}} \underbrace{\frac{\partial \hat{\epsilon}_{\phi}(\mathbf{z}_t; y, t)}{\mathbf{z}_t}}_{\text{U-Net Jacobian}} \underbrace{\frac{\partial \mathbf{x}}{\partial \theta}}_{\text{Generator Jacobian}} \right]$$

$$\nabla_{\theta} \mathcal{L}_{\text{SDS}}(\phi, \mathbf{x} = g(\theta)) \triangleq \mathbb{E}_{t, \epsilon} \left[w(t) (\hat{\epsilon}_{\phi}(\mathbf{z}_t; y, t) - \epsilon) \frac{\partial \mathbf{x}}{\partial \theta} \right]$$

There is a theoretical motivation (see Appendix in [paper](#))

DreamFusion



an orangutan making a clay bowl on a throwing wheel*



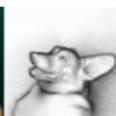
a raccoon astronaut holding his helmet†



a blue jay standing on a large basket of rainbow macarons*



a corgi taking a selfie*



a table with dim sum on it†



a lion reading the newspaper*



Michelangelo style statue of dog reading news on a cellphone



a tiger dressed as a doctor*



a steam engine train, high resolution*



a frog wearing a sweater*



a humanoid robot playing the cello*



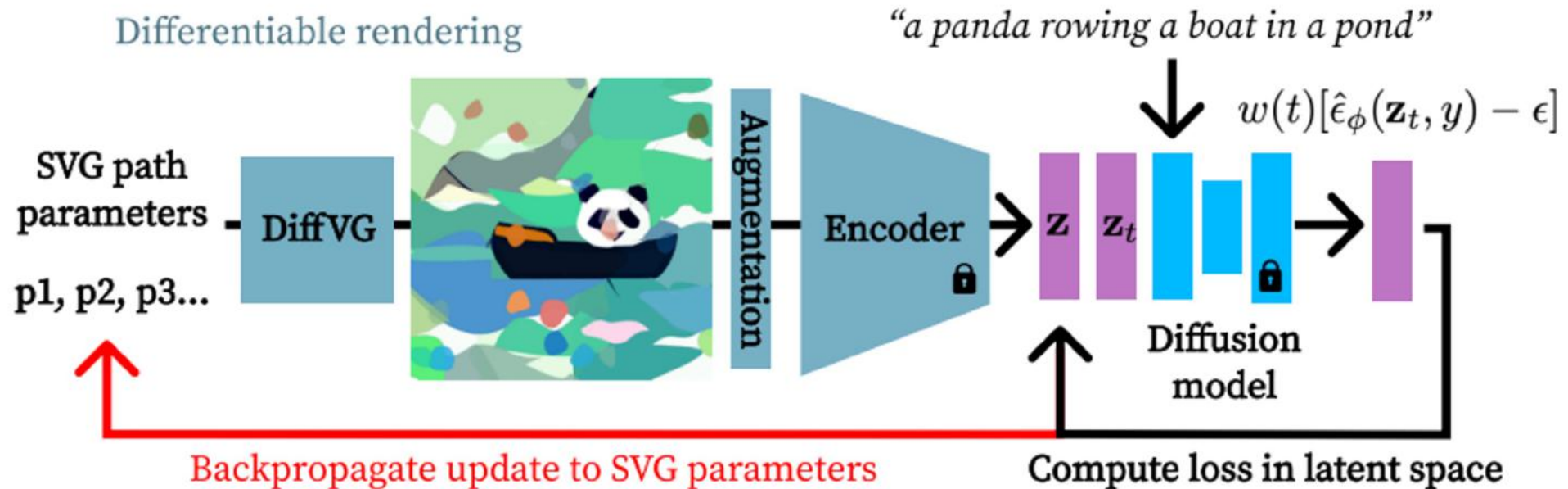
Sydney opera house, aerial view†



Text-to-SVG

Using SDS loss is not limited to text-to-3D

- SDS provides a general way to backprop / learn through any differential “rendering” pipeline



Text-to-SVG



a train*



an owl standing on a wire*



Underwater Submarine*



a boat*



A photo of a Ming Dynasty vase on a leather topped table.*



A smiling sloth wearing a leather jacket, a cowboy hat and a kilt.*



a tuba with red flowers protruding from its bell*



a blue poison dart frog sitting on a water lily*

Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - Prediction Space / Loss: x , ϵ , v
 - Issues with Common Noise Schedules
 - Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers
 - Image Editing
 - Using scores in diffusion models: text-to-3D, text-to-SVG
 - **Faster Sampling**
 - Diffusion as Pre-Training
 - Other Fields: visuomotor control, materials discovery

Progressive Distillation

PROGRESSIVE DISTILLATION FOR FAST SAMPLING OF DIFFUSION MODELS

Tim Salimans & Jonathan Ho
Google Research, Brain team
{salimans, jonathanho}@google.com

Progressive Distillation

- **Problem:** Diffusion models take many steps to sample.
 - DDPM, Diffusion Beats Gans: 1000 timesteps.
 - Large N good for training, but slow to evaluate.
- **Insight:** Distill an N -step diffusion model into $N/2$ steps.
 - Then repeat this progressively – $N/4$ steps, $N/8$, etc.

Progressive Distillation

- **Standard Diffusion**

- Sample x , t , eps . Add eps to x .
- Update model to predict original x .

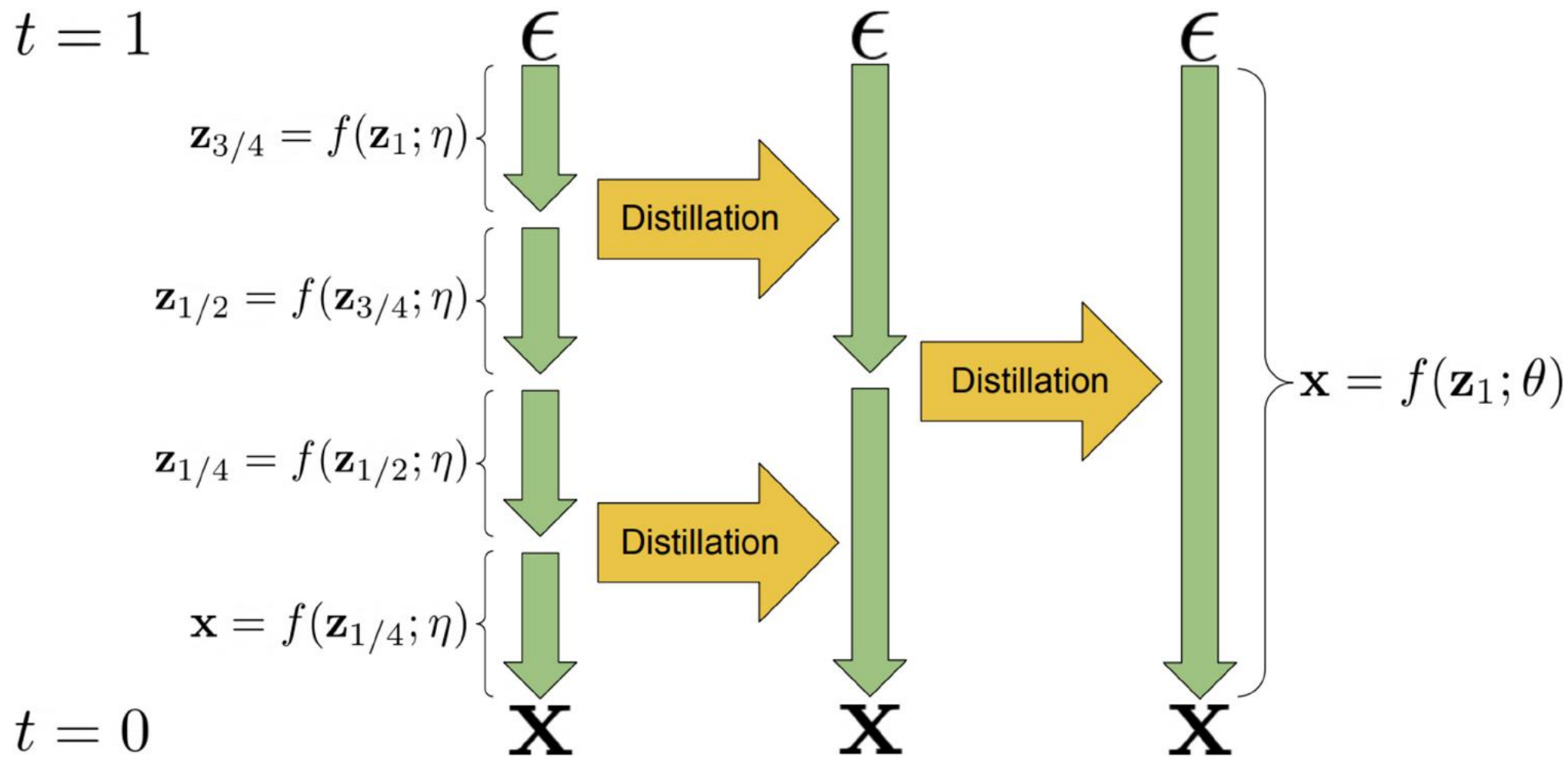
- **Progressive Distillation**

- Sample x , t , eps . Add eps to x .
- **Run teacher model twice to get x' .**
- Update model to predict x' .

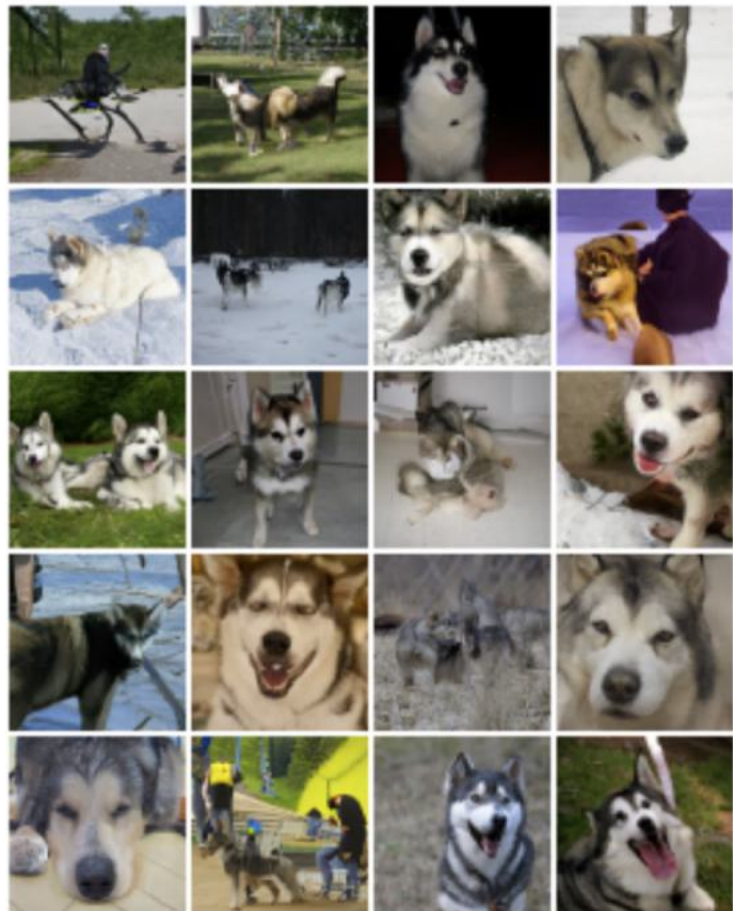
- **Why?**

- $p(x_{\text{original}} | x_{\text{noise}})$ is a noisier distribution with many answers, whereas $p(x' | x_{\text{noise}})$ is more determined.

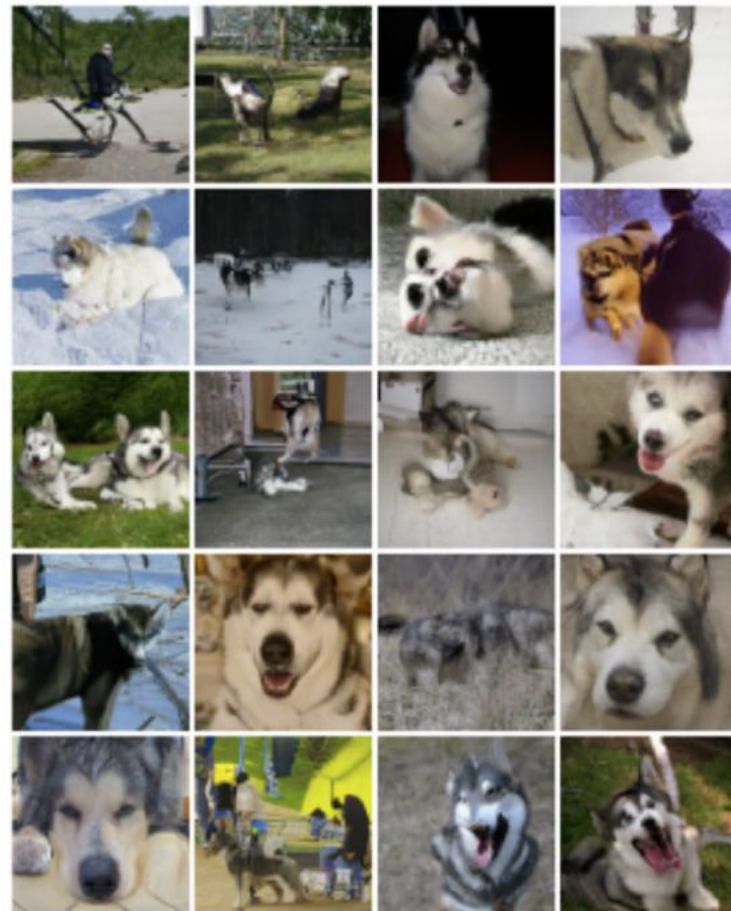
Progressive Distillation



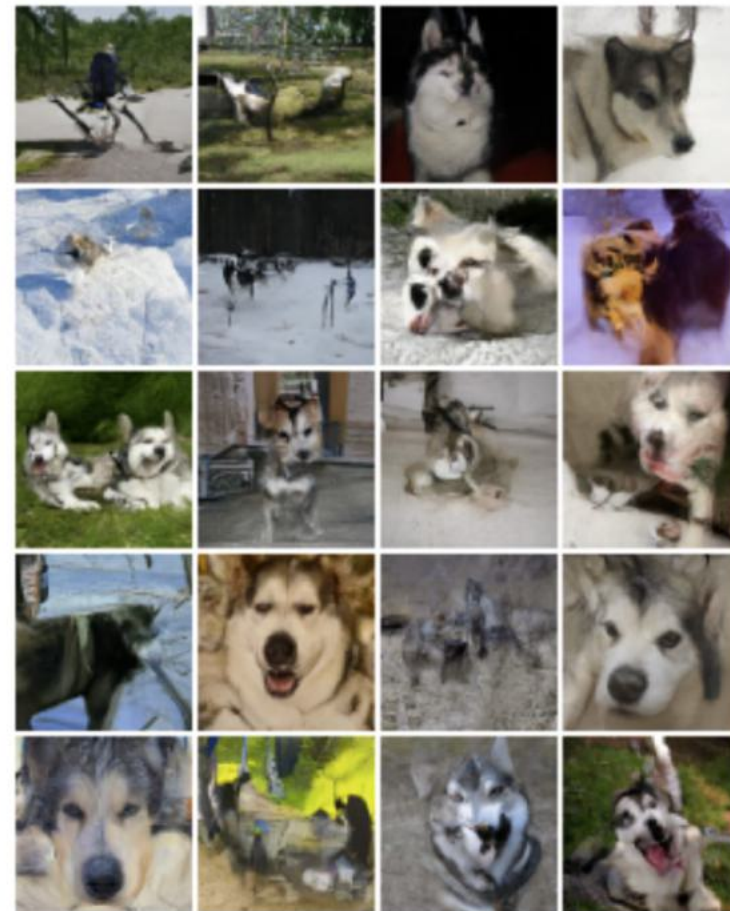
Progressive Distillation



(a) 256 sampling steps



(b) 4 sampling steps



(c) 1 sampling step

Denoising Diffusion GAN

- We need many diffusion steps because of Gaussian assumption.
 - The forward process is a Gaussian process.
 - True backward process is a Gaussian, **only for small step sizes.**
 - For large steps, not Gaussian.
 - $p(x_0)$ is multimodal, a result of many unimodal (gaussian) steps.

Denoising Diffusion GAN

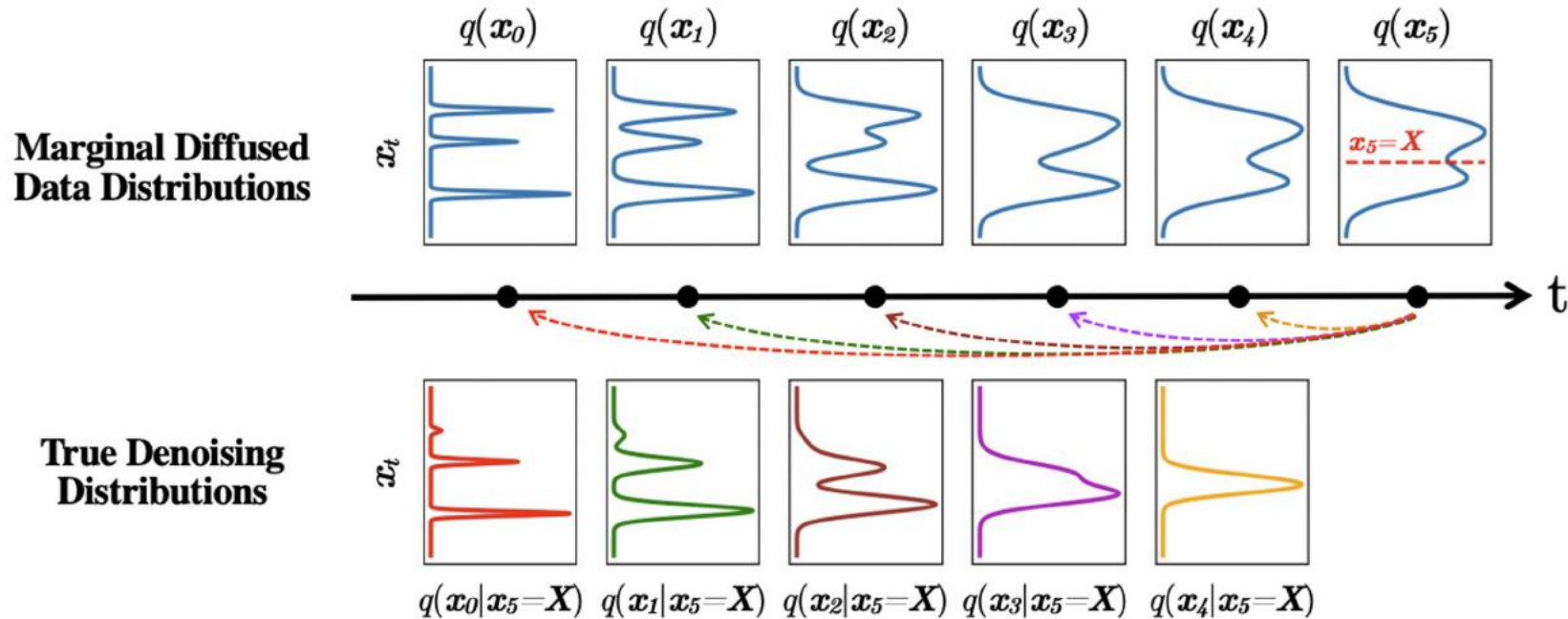


Figure 2: **Top:** The evolution of 1D data distribution $q(\mathbf{x}_0)$ through the diffusion process. **Bottom:**, The visualization of the true denoising distribution for varying step sizes conditioned on a fixed \mathbf{x}_5 . The true denoising distribution for a small step size (i.e., $q(\mathbf{x}_4|\mathbf{x}_5 = X)$) is close to a Gaussian distribution. However, it becomes more complex and multimodal as the step size increases.

Denoising Diffusion GAN

- Instead, train the reverse distribution with a GAN.

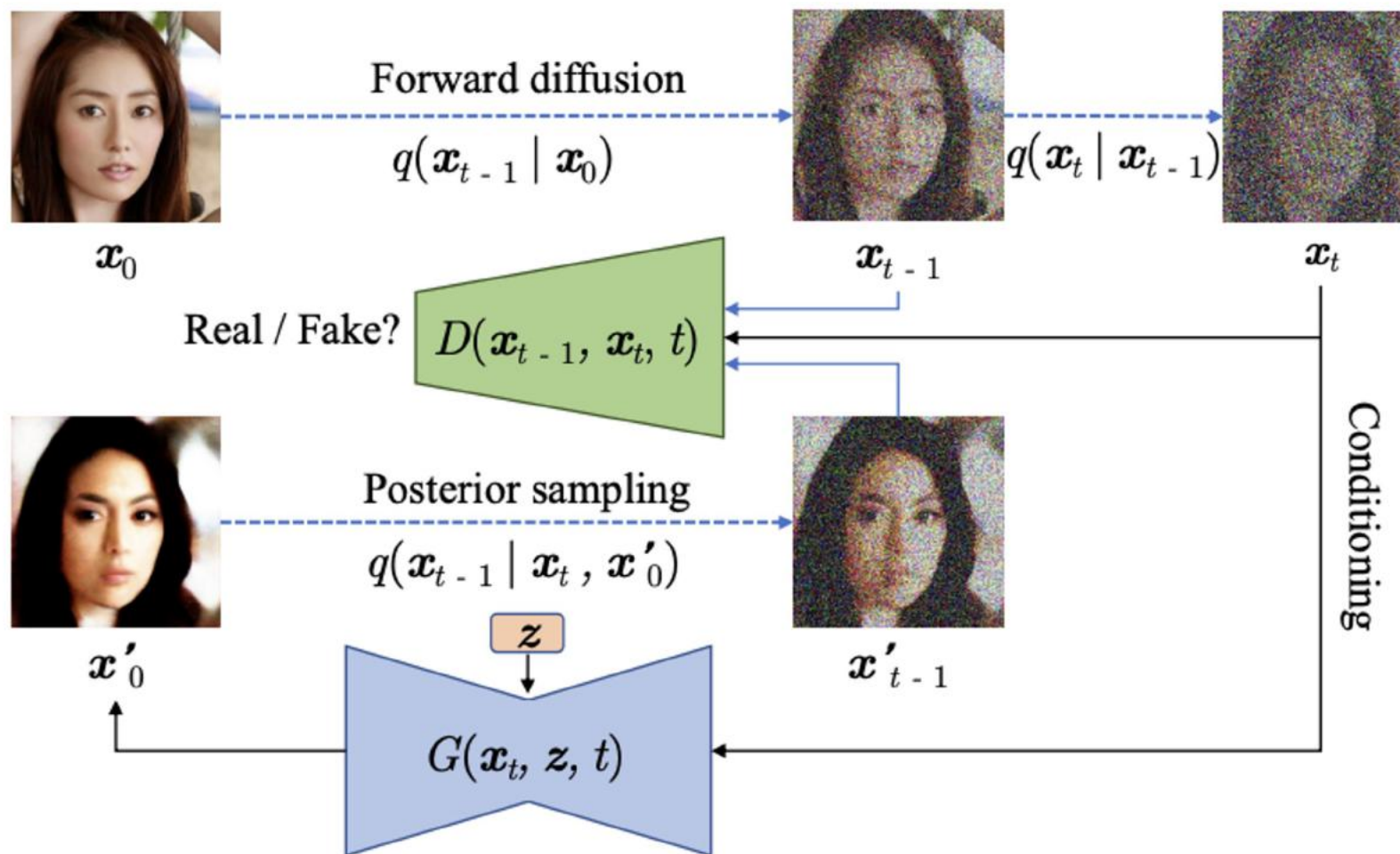
$$\min_{\theta} \sum_{t \geq 1} \mathbb{E}_{q(\mathbf{x}_t)} [D_{\text{adv}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t) || p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t))],$$

- We can sample from the true reverse distribution by running the forward process on data. Train discriminator:

$$\mathbb{E}_{q(\mathbf{x}_t)q(\mathbf{x}_{t-1}|\mathbf{x}_t)}[-\log(D_{\phi}(\mathbf{x}_{t-1}, \mathbf{x}_t, t))] = \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{t-1}|\mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_{t-1})}[-\log(D_{\phi}(\mathbf{x}_{t-1}, \mathbf{x}_t, t))].$$

- Then, train generator to maximize discriminator objective.

Denoising Diffusion GAN



Denoising Diffusion GAN



(4 Diffusion Steps)

MobileDiffusion

MobileDiffusion: Subsecond Text-to-Image Generation on Mobile Devices

Yang Zhao, Yanwu Xu, Zhisheng Xiao, Tingbo Hou
Google

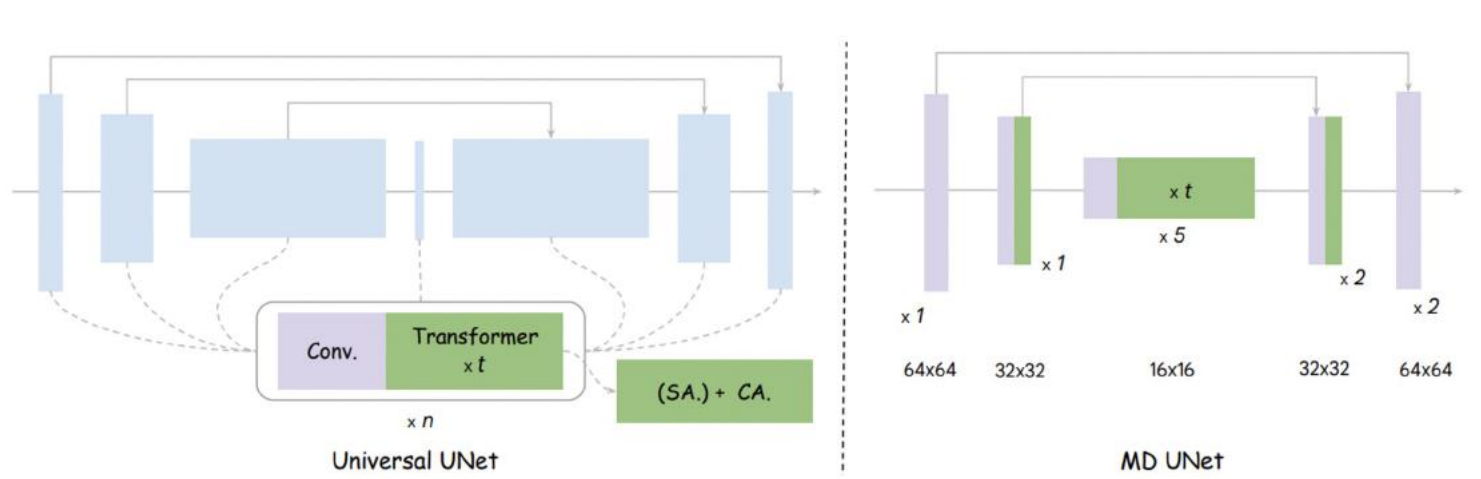
{yzhaoeric, yanwuxu, zsxiao, tingbo}@google.com



Figure 1. MobileDiffusion text-to-image generation in (a) 8 steps (*cfg-aware* distillation [23]), and (b) 1 step (UFOGen finetuning [57]).

MobileDiffusion

- Many architectural changes can reduce parameter counts.
 - More transformers inside middle of Unet. [26% efficiency gain]
 - Intuition: perform calculation in compressed latent space.
 - Decouple self-attention and cross-attention. [15% efficiency gain]
 - Cross-attention to text is important – include at all layers.
 - Self-attention is expensive at high-resolution – include only in middle.



MobileDiffusion

- Many architectural changes can reduce parameter counts.
 - Share key-value projections. [5% parameter reduction]
 - Set (key = value) for self-attention.
 - Finetune softmax into relu.
 - Attention softmax requires sum over exponents, pointwise relu faster.
 - Can train with softmax, then finetune to work with relu.
 - Reduce feedforward layer width. [10% parameter reduction]
 - Use separable convolutions [Howard et al, 2017] [10% param reduc.]
 - Reduce residual blocks. [15% parameter reduction]

MobileDiffusion

Models	#Channels	#ConvBlocks	#(SA+CA)	#Params(M)	#GFLOPs	Latency(ms)	Model Size (GB)
SD-XL [36]	(320, 640, 1280)	17	31+31	2,300	710	29.5	5.66
SD-1.4/1.5	(320, 640, 1280, 1280)	22	16+16	862	392	21.7	2.07
SnapFusion [23]	(320, 640, 1280, 1280)	18	14+14	848	285	15.0	1.97
MobileDiffusion	(320, 640, 1024)	11	15+18	386	182	9.9	1.04
MobileDiffusion-Lite	(320, 640, 896)	11	12+15	278	153	8.8	0.82

Table 1. Comparison with other recognized latent diffusion models. Latency and GFLOPs, computed with jit per forward step, are measured for an input latent size of 64×64 on TPU v3. Model size (fp16) includes all, *i.e.*, UNet, text encoder and VAE decoder.

SDXL-Lightning

[Submitted on 21 Feb 2024]

SDXL-Lightning: Progressive Adversarial Diffusion Distillation

[Shanchuan Lin](#), [Anran Wang](#), [Xiao Yang](#)

We propose a diffusion distillation method that achieves new state-of-the-art in one-step/few-step 1024px text-to-image generation based on SDXL. Our method combines progressive and adversarial distillation to achieve a balance between quality and mode coverage. In this paper, we discuss the theoretical analysis, discriminator design, model formulation, and training techniques. We open-source our distilled SDXL-Lightning models both as LoRA and full UNet weights.

SDXL-Lightning

- Key Idea: Denoising Diffusion GAN, for **distillation**.
- Distillation of diffusion models fails at small step sizes.
 - It's the same MSE ambiguity problem we covered earlier.
 - Small steps need to model a complicated flow – Gaussian not enough.
 - Instead – distill with a GAN.

SDXL-Lightning

- SDXL-Lightning GAN
 - Discriminator conditioned on pairs of $(x, x_{\text{nstep_denoised}})$.
 - Samples from either teacher network or student.

$$D(x_t, x_{t-ns}, t, t - ns, c)$$

- Re-use the weights from the teacher encoder for the discriminator.

SDXL-Lightning

- Janus Artifacts
 - Student model still has less capacity than the teacher = mismatch.
 - Relaxing mode-coverage reduces complexity.
 - Finetune on unconditional flows.

$$D(x_t, x_{t-ns}, t, t - ns, c)$$

$$D'(x_{t-ns}, t - ns, c)$$



Figure 2. “Janus” artifacts appear when the student network does not have the capacity to match the teacher’s sudden changes. This problem can be mitigated by relaxing the mode coverage requirement.

SDXL-Lightning

- Training Details
 - Distill 128 -> 32 with MSE loss.
 - Distill 32,8,4,3,1 with GAN loss.
 - First with conditional, then unconditional, in each stage.
 - Train with LoRA, then further finetune full model.
 - Train at multiple timesteps, even if not necessary for generation.

SDXL-Lightning



(a) A tanned woman, dressed in sportswear and sunglasses, climbing a peak with a group during the summer.



(b) A dolphin leaps through the waves, set against a backdrop of bright blues and teal hues.



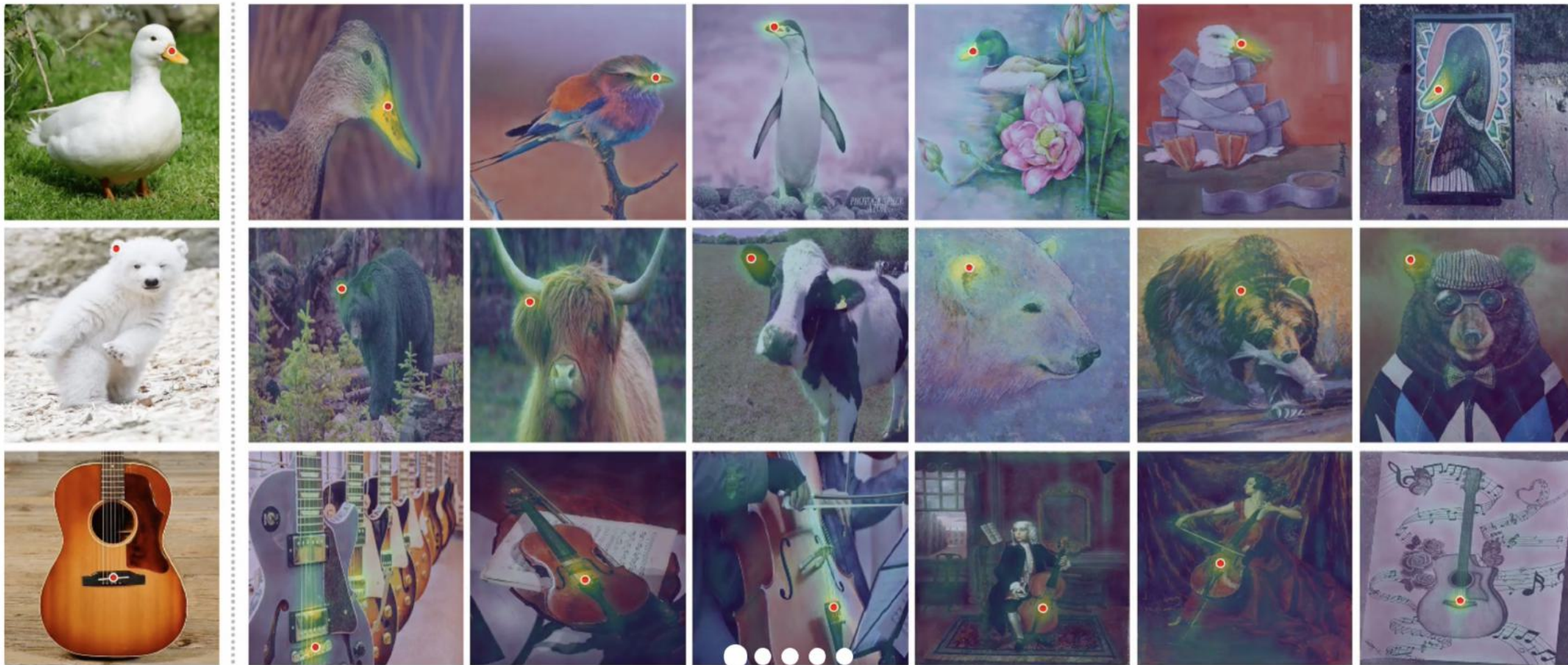
Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - Prediction Space / Loss: x , ϵ , v
 - Issues with Common Noise Schedules
 - Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers
 - Image Editing
 - Using scores in diffusion models: text-to-3D, text-to-SVG
 - Faster Sampling
 - **Diffusion as Pre-Training**
 - Other Fields: visuomotor control, materials discovery

Diffusion Features

- **Question:** Unsupervised learning is also wanted for representation learning. Can diffusion models be used for this?
- **Insight:** Diffusion model is trained with noisy images as input.
 - To extract features, **add noise** to image, then process.
 - Average over a batch of noise for better accuracy.
 - Correspondences can be located via cosine distance.
 - As T changes, tradeoff between semantics and details.

Diffusion Features



Diffusion Features

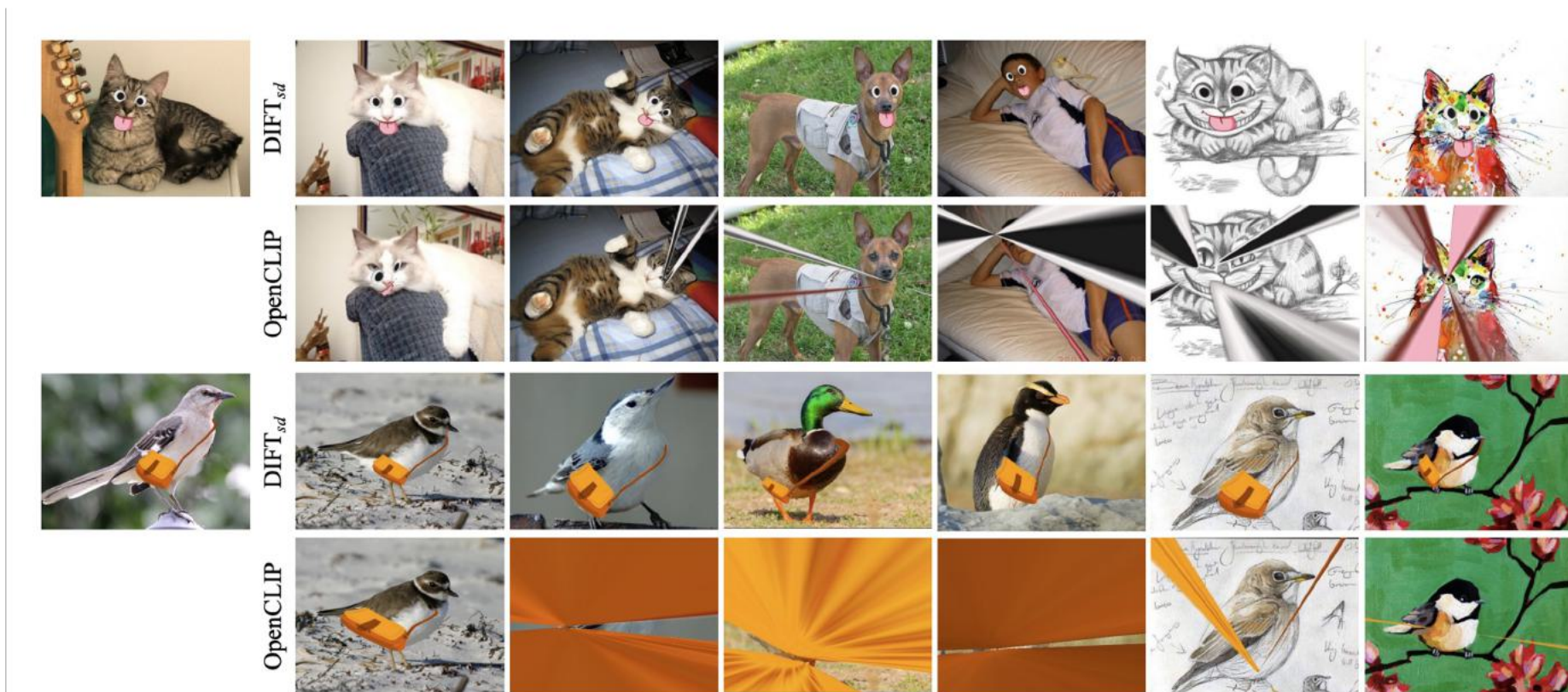


Figure 6: Edit propagation. The first column shows the source image with edits (i.e., the overlaid stickers), and the rest columns are the propagated results on new images from different instances, categories, and domains, respectively. Compared to OpenCLIP, DIFT_{sd} propagates edits much more accurately. More results are in Fig. 20 of Appendix E.

Diffusion Classifier

Your Diffusion Model is Secretly a Zero-Shot Classifier

Alexander C. Li Mihir Prabhudesai Shivam Duggal Ellis Brown Deepak Pathak

Carnegie Mellon University

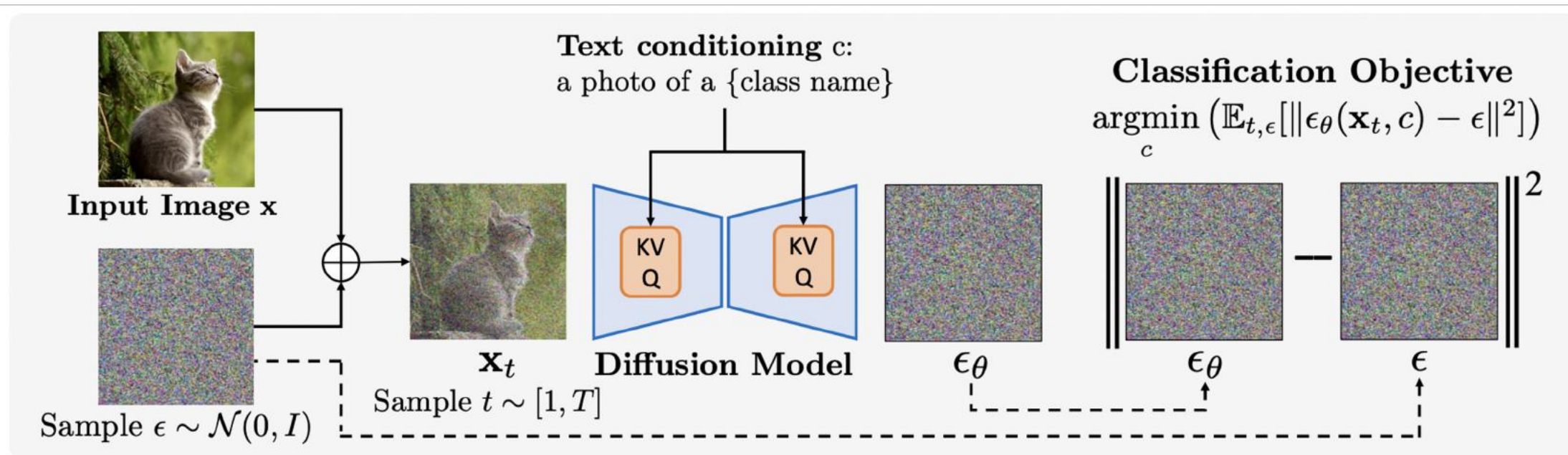
Diffusion Classifier

- **Problem:** We want to classify images, or otherwise compute a distance between [image, text] pairs.
- **Insight:** We have an approximate model of $p(\mathbf{x}|\mathbf{c})$. Use Bayes' rule:

$$p_{\theta}(\mathbf{c}_i | \mathbf{x}) = \frac{p(\mathbf{c}_i) p_{\theta}(\mathbf{x} | \mathbf{c}_i)}{\sum_j p(\mathbf{c}_j) p_{\theta}(\mathbf{x} | \mathbf{c}_j)}$$

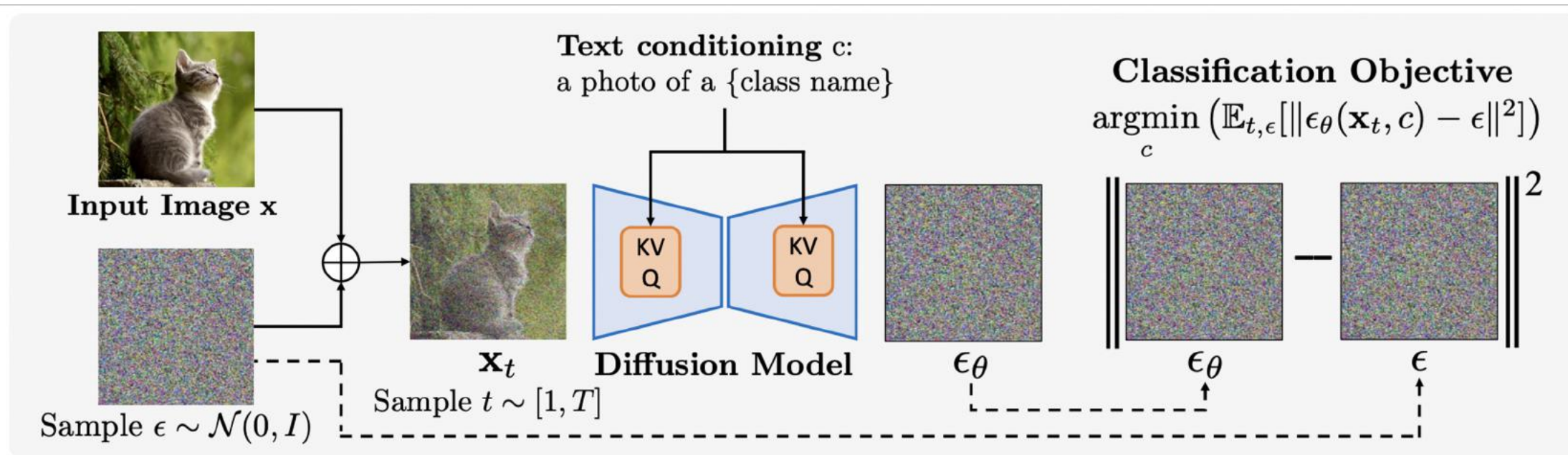
Diffusion Classifier

- [Image, text] distance can be calculated by adding noise, denoising with text-conditioning, and taking L2 distance.



Diffusion Classifier

- [Image, text] distance can be calculated by adding noise, denoising with text-conditioning, and taking L2 distance.



EmerDiff

Question: How do we pixel-level segmentation maps from diffusion models?

- Low-resolution solution: extract feature maps, then use K-means.
 - In a text-conditioned model, use the query vectors as features.
- To upscale: Figure out correspondence from features to pixels.
 - Add a (+,-) constant offset to the feature maps.
 - Then, take the difference in the generated image.

EmerDiff

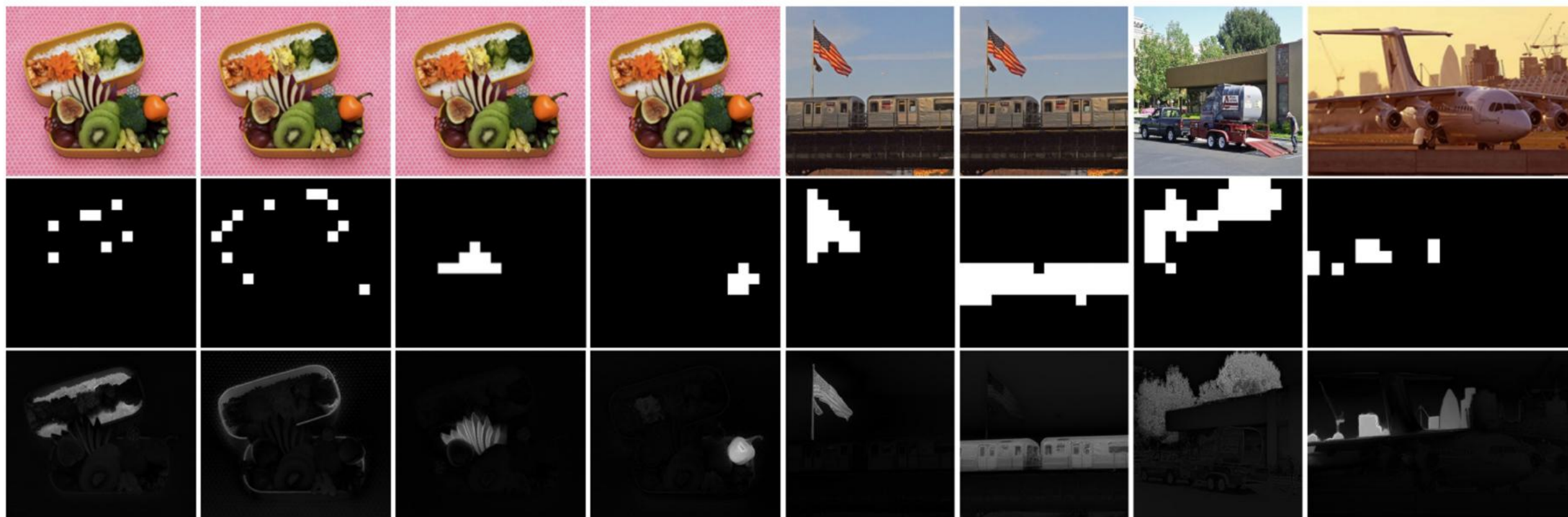
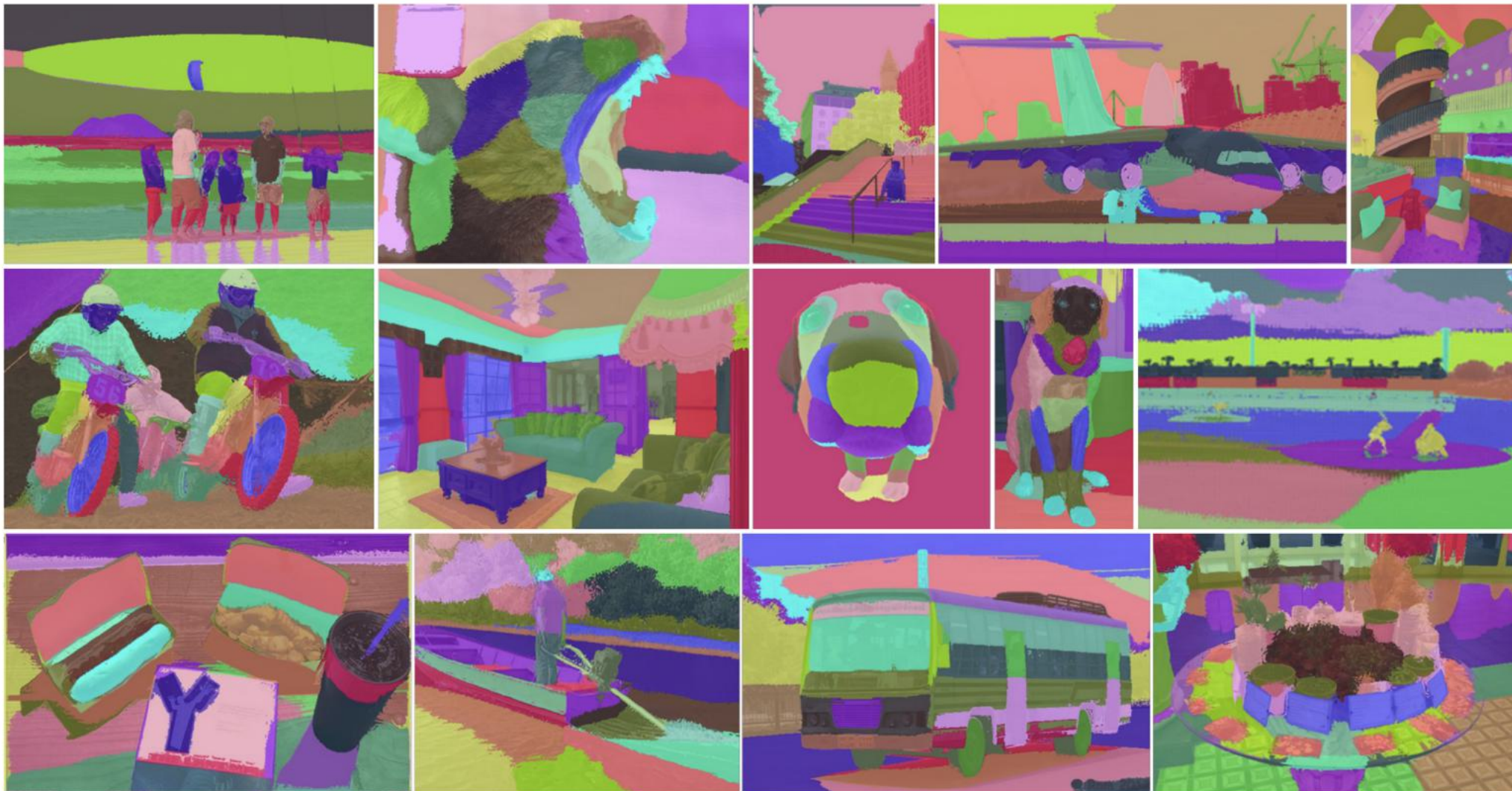


Figure 3: **Visualization of modulated denoising process.** First row: original image. Second row: low-resolution modulation mask $M \in \{0, 1\}^{h \times w}$. Third row: obtained difference map $d \in \mathbb{R}^{H \times W}$, where $H/h = W/w = 32$

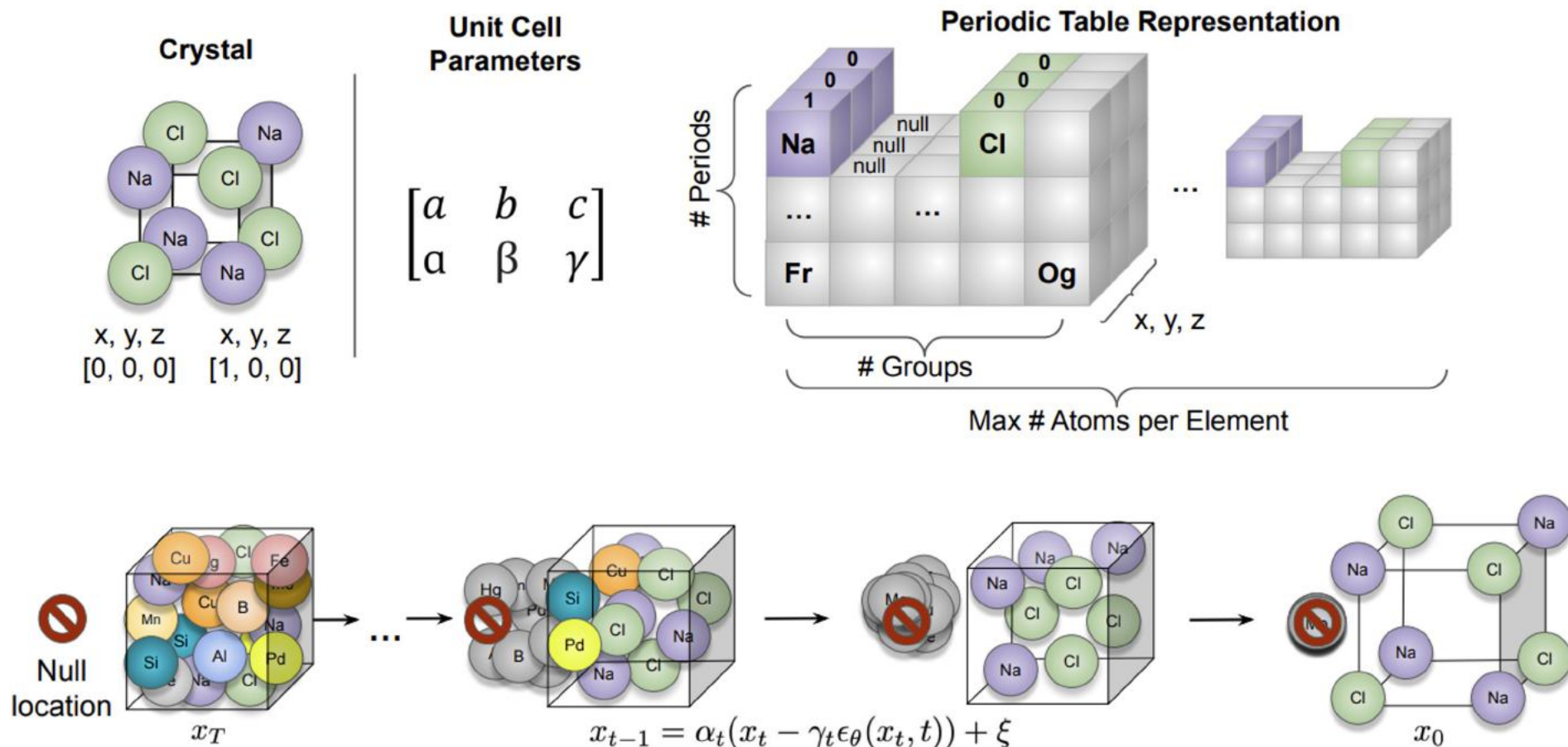
EmerDiff



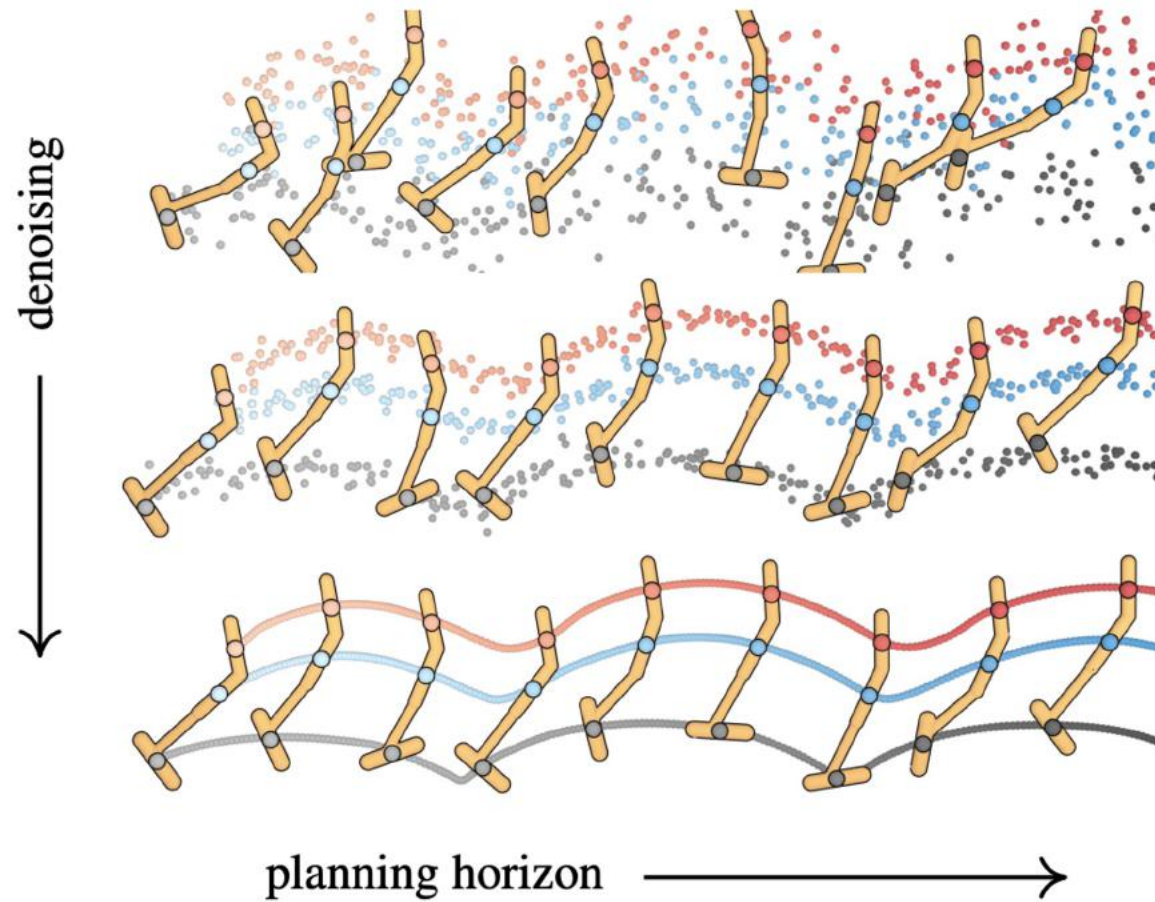
Lecture overview

- Motivation
- Denoising Diffusion Models
- **Deeper Dive into Diffusion Models**
 - DDIM Sampling
 - Prediction Space / Loss: x , ϵ , v
 - Issues with Common Noise Schedules
 - Architectures: UNet, latent space / stable diffusion, hierarchical generation, transformers
 - Image Editing
 - Using scores in diffusion models: text-to-3D, text-to-SVG
 - Faster Sampling
 - Diffusion as Pre-Training
 - **Other Fields: visuomotor control, materials discovery**

UniMat: Scalable Diffusion for Materials Generation

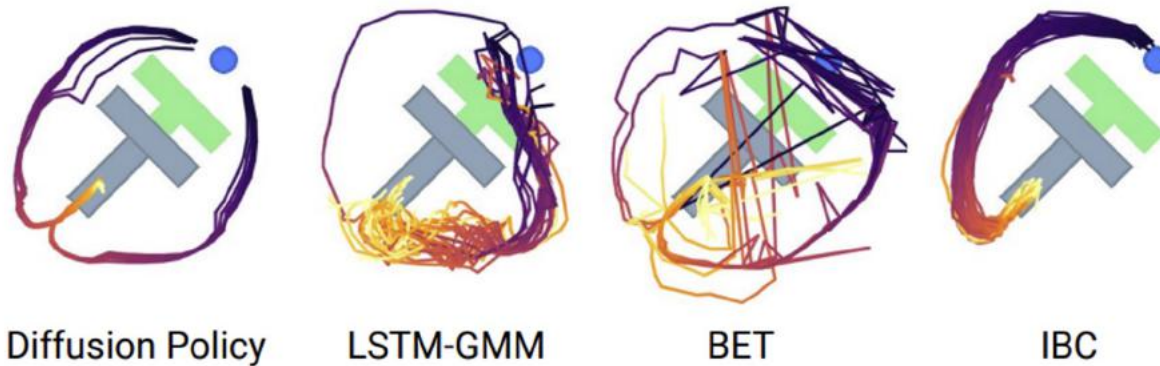


Planning with Diffusion for Flexible Behavior Synthesis



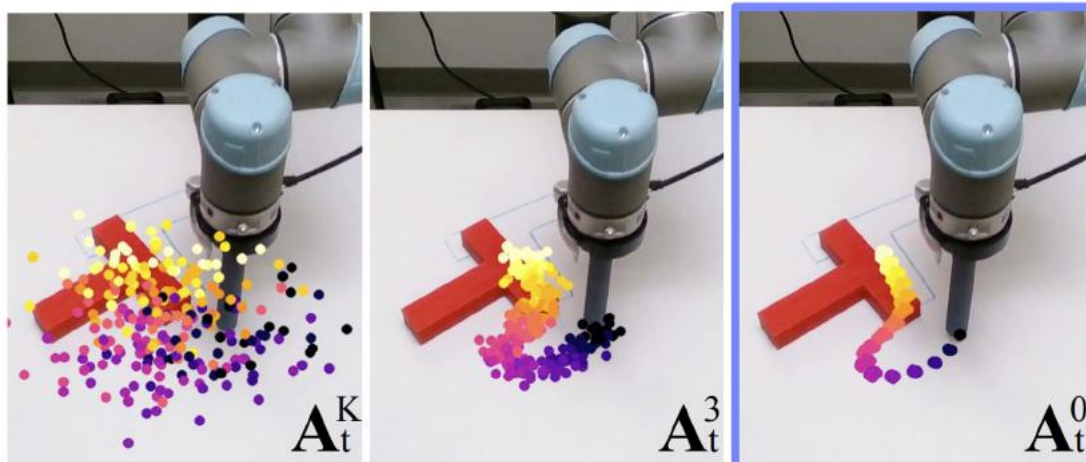
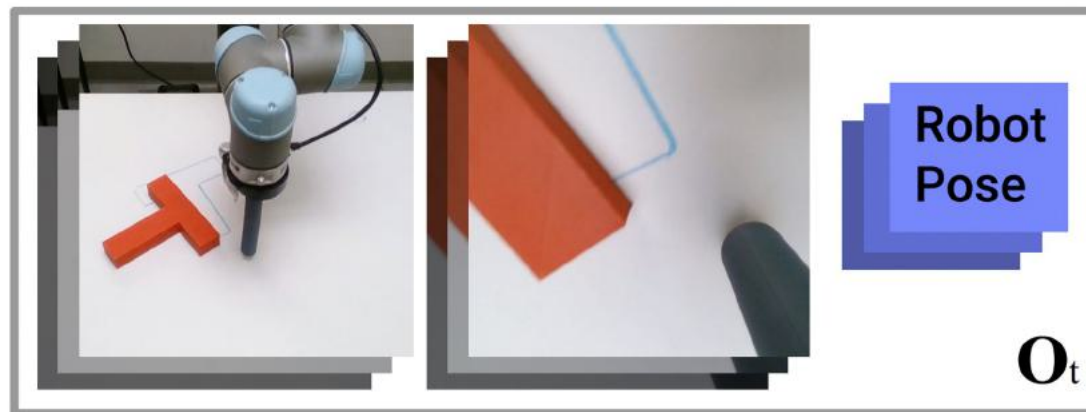
Diffusion Policy

- **Problem:** In robotics, expert trajectories are multimodal – Gaussian policies fail to capture behavior.
- **Solution:** Use diffusion model over actions, and diffuse over future action trajectories.

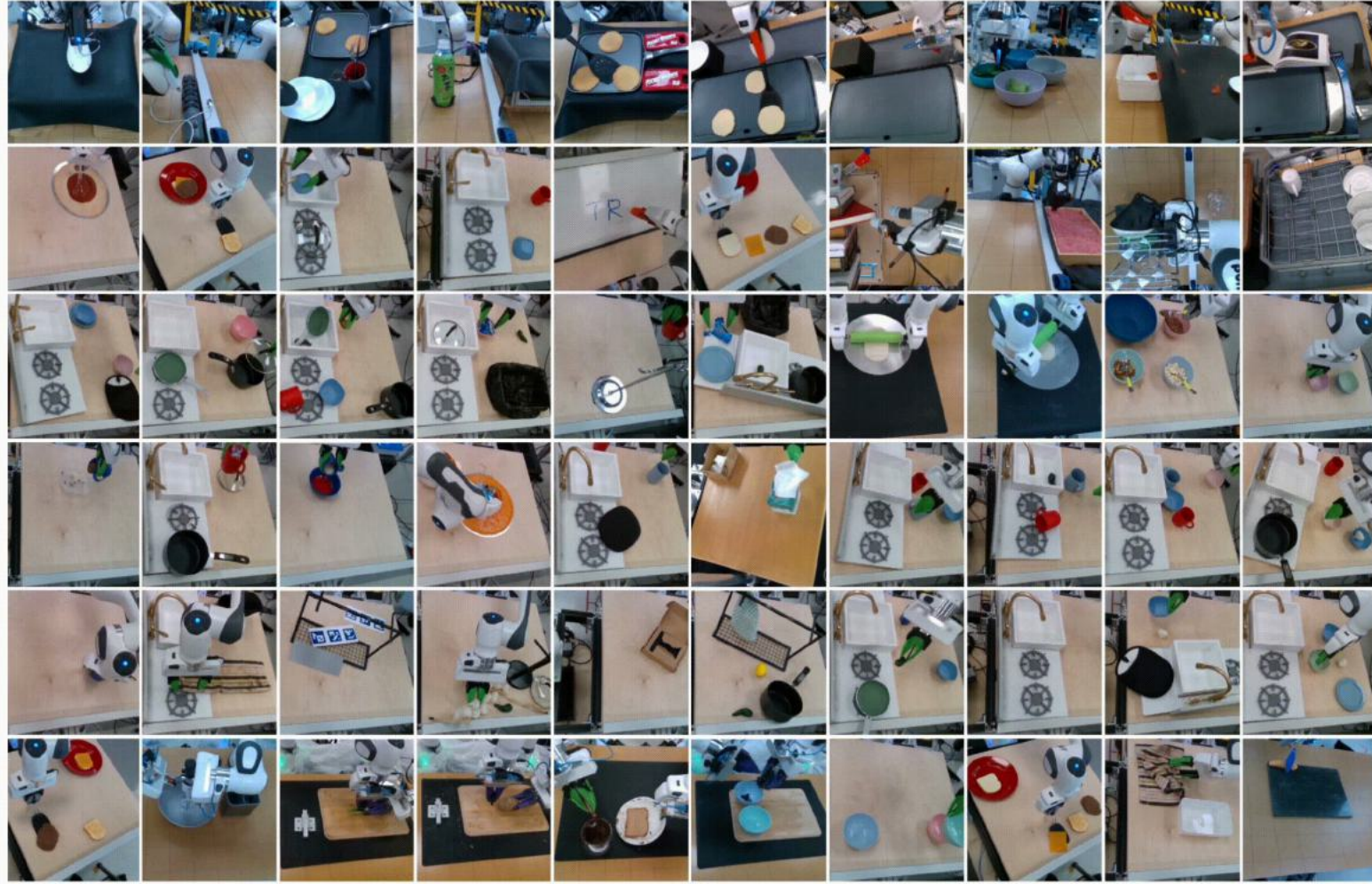


Diffusion Policy

Input: Image Observation Sequence



Teaching Robots New Behaviors



Next lecture: Flow Models