UCF Center for Research in Computer Vision

CAP6412
Advanced Computer Vision

Mubarak Shah

shah@crcv.ucf.edu

HEC-245

Lecture-10: Diffusion Models-Introduction Part-I

Generative AI

- **Text**
  - Large Language Models (LLM)
    - Generate/Predict the next word
    - Auto-regressive
  - Visual Language Models (VLM)/Large Multimodal Models (LMM)

- **Images**
  - DALL-E
    - Auto-regressive model for image Generation
  - Diffusion Models
    - Generate Images, Videos, audio, …

Generative AI

- **Text**
  - Large Language Models (LLM)
    - Generate/Predict the next word
    - Auto-regressive
  - Visual Language Models (VLM)/Large Multimodal Models (LMM)

- **Images**
  - DALL-E
    - Auto-regressive model for image Generation
  - **Diffusion Models**
    - **Generate Images**, Videos, audio, …

Diffusion models in vision: A survey

IEEE Transactions on Pattern Analysis and Machine Intelligence 45 (9), 10850 , 2023



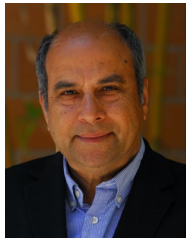
Alin Croitoru
University of Bucharest, Romania
alincroitoru97@gmail.com



Vlad Hondru
University of Bucharest, Romania
vladhondru25@gmail.com



Radu Tudor Ionescu
University of Bucharest, Romania
raducu.ionescu@gmail.com



Mubarak Shah
University of Central Florida, US
shah@crcv.ucf.edu

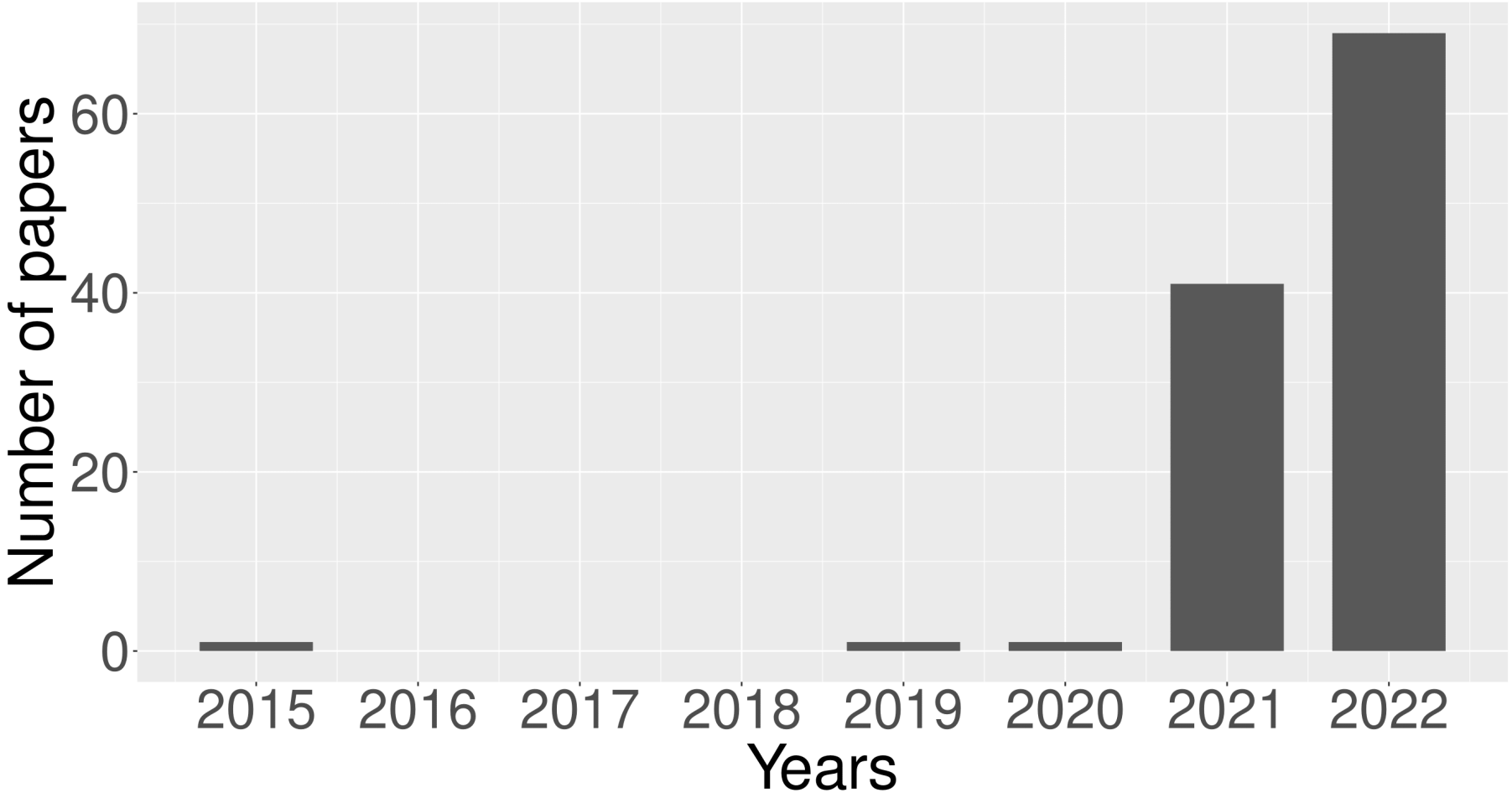
Agenda

1. Motivation
2. High-level overview
3. Denoising diffusion probabilistic models
4. Noise Conditioned Score Network
5. Conditional Generation
6. Stochastic Differential Equations
7. Applications
8. Research directions

Outline

1. Motivation
2. High-level overview
3. Denoising diffusion probabilistic models
4. Noise Conditioned Score Network
5. Conditional Generation
6. Stochastic Differential Equations
7. Applications
8. Research directions

Diffusion Models Papers over the years



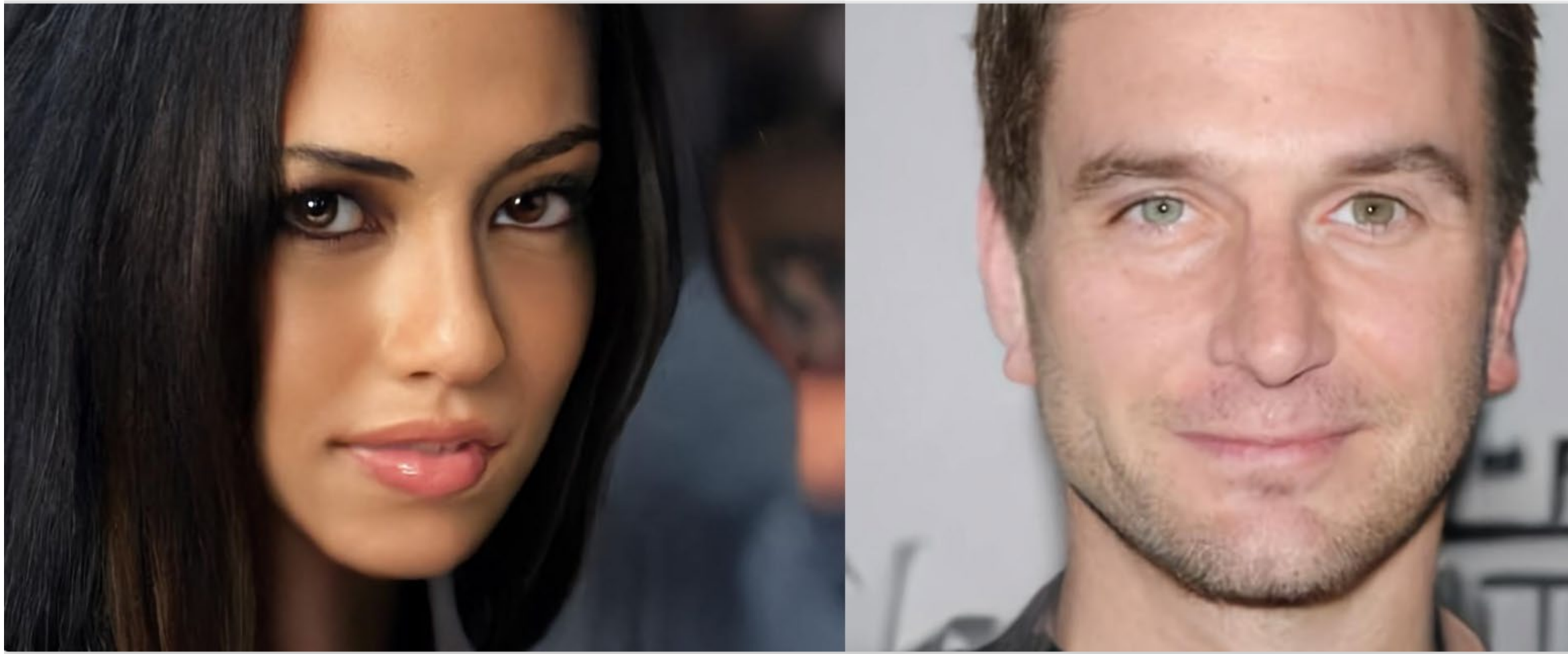
Diffusion Models

- They are fundamentally different from other generative methods.
- They aim to decompose the image generation process (sampling) in many small “denoising” steps.
- Model can correct itself over these small steps and gradually produce a good sample
- Therefore, initially they were slower than GANs.
- They take the input image and gradually add Gaussian noise to it through a series of steps. We will call this the forward process.
- Afterward, a neural network is trained to recover the original data by reversing the noising process.
- By being able to model the reverse process, we can generate new data. This is the so-called reverse process.

https://theaisummer.com/diffusion-models/

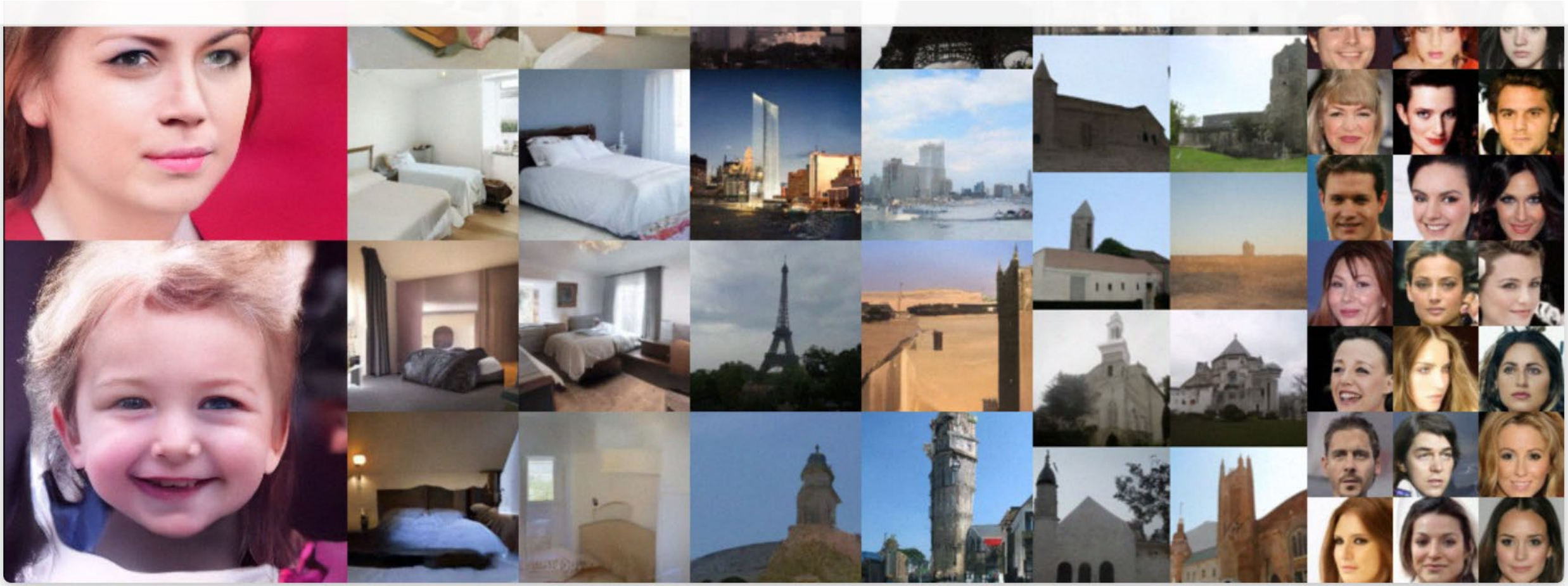
Diffusion Applications

- Image generation
- Text-to-image
- Super resolution
- Image editing
- Inpainting
- Image segmentation
- Video Generation
- …..



1024 x 1024 samples generated from score-based models [20]

© Copyright 2022 Yang Song. Powered by Jekyll with al-folio theme.



Samples from the NCSNv2 [18] model. From left to right: FFHQ 256x256, LSUN bedroom 128x128, LSUN tower 128x128, LSUN church_outdoor 96x96, and CelebA 64x64.

© Copyright 2022 Yang Song. Powered by Jekyll with al-folio theme.



1024 x 1024 samples from a score-based model trained on the FFHQ dataset.

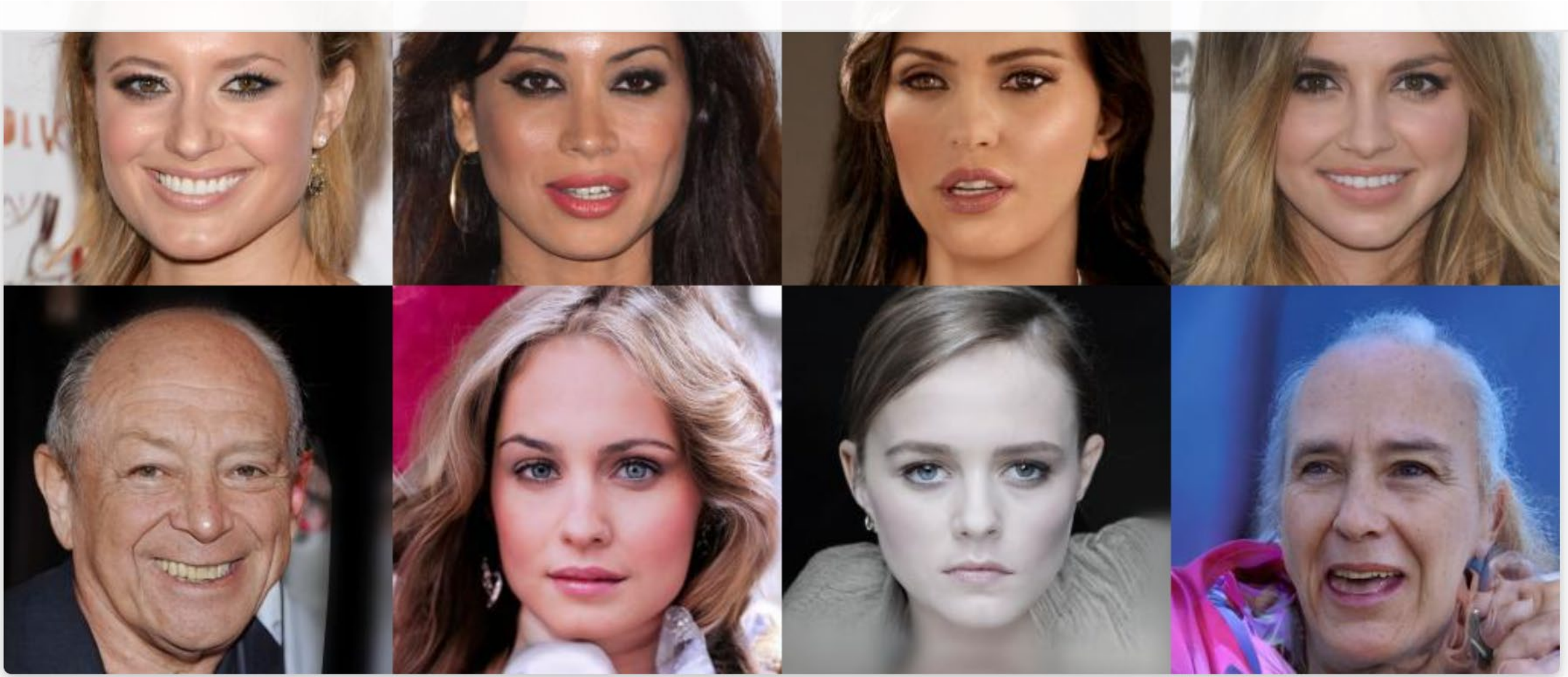
Some additional (uncurated) samples for other datasets (taken from this GitHub repo):

© Copyright 2022 Yang Song. Powered by Jekyll with al-folio theme.



256 x 256 samples on LSUN bedroom.

© Copyright 2022 Yang Song. Powered by Jekyll with al-folio theme.



256 x 256 samples on CelebA-HQ.

Inpainting



Image Colorization



Image Colorizations



We can even colorize gray-scale portrays of famous people in history (Abraham Lincoln) with a time-dependent score-based model trained on FFHQ. The image resolution is 1024 x 1024.

Stable Diffusion

High-Resolution Image Synthesis with Latent Diffusion Models

Robin Rombach[1] * Andreas Blattmann[1] * Dominik Lorenz[1] Patrick Esser[R] Björn Ommer[1]

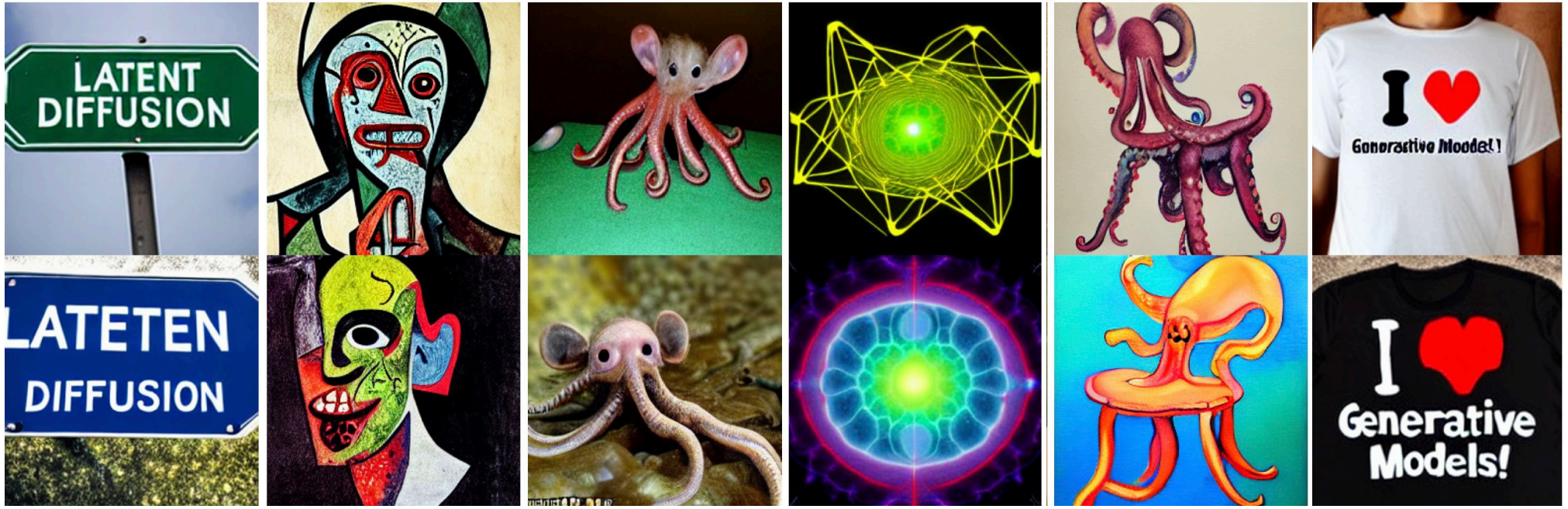
[1]Ludwig Maximilian University of Munich & IWR, Heidelberg University, Germany [R]Runway ML

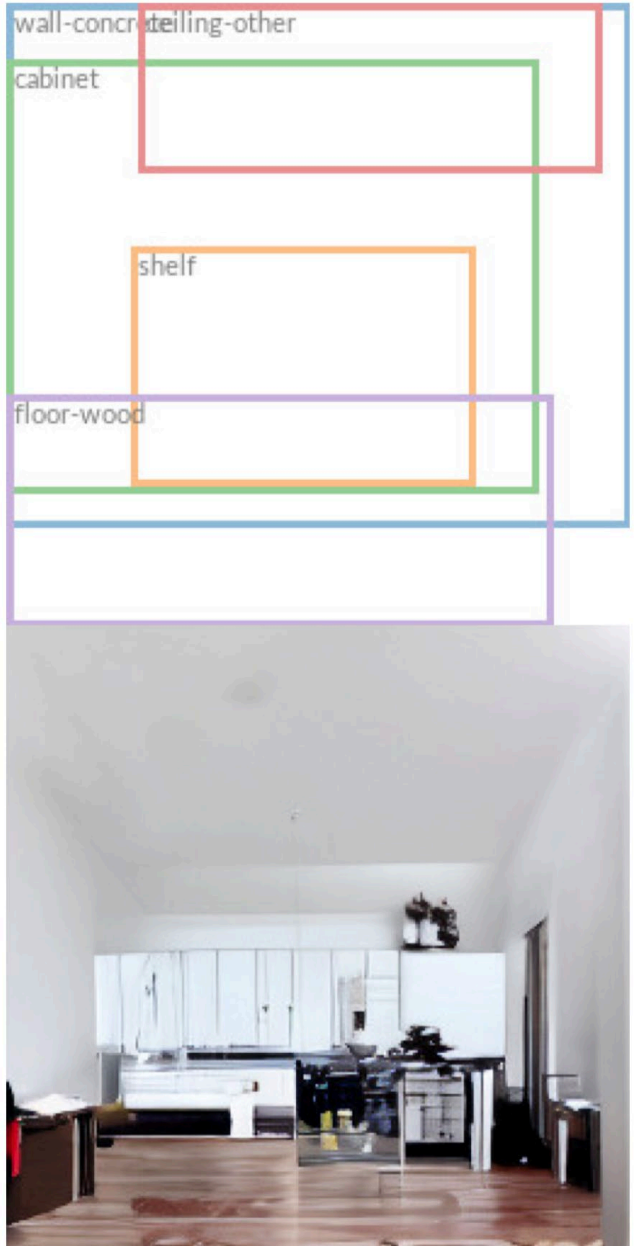
https://github.com/CompVis/latent-diffusion

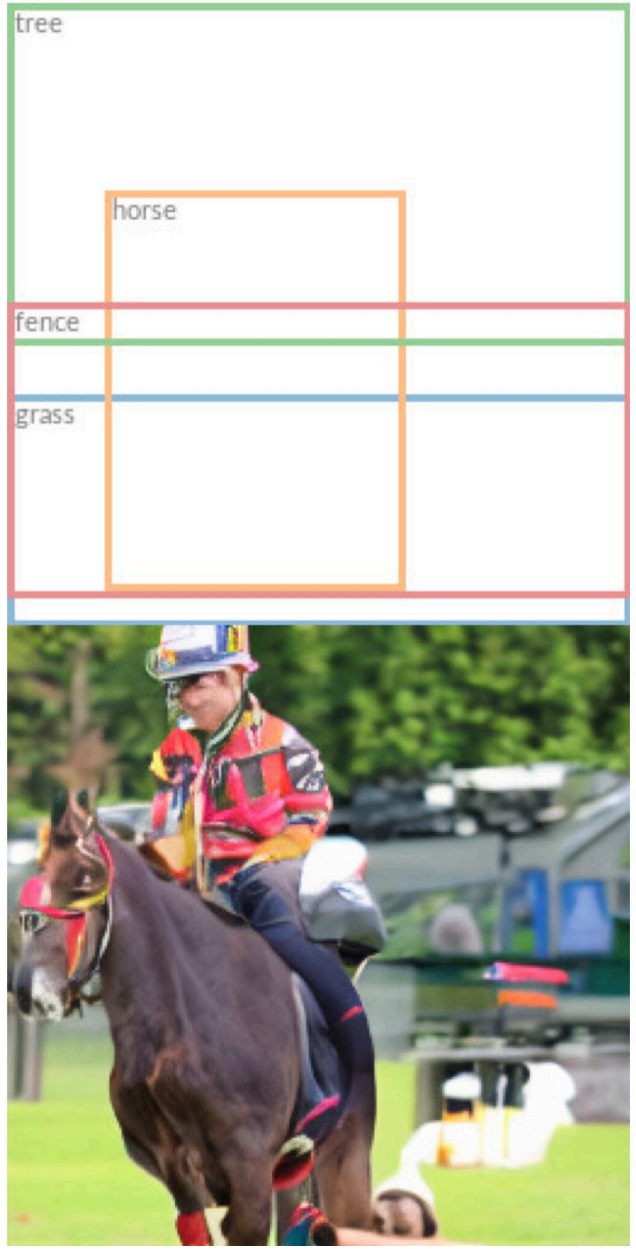
Text-to-Image

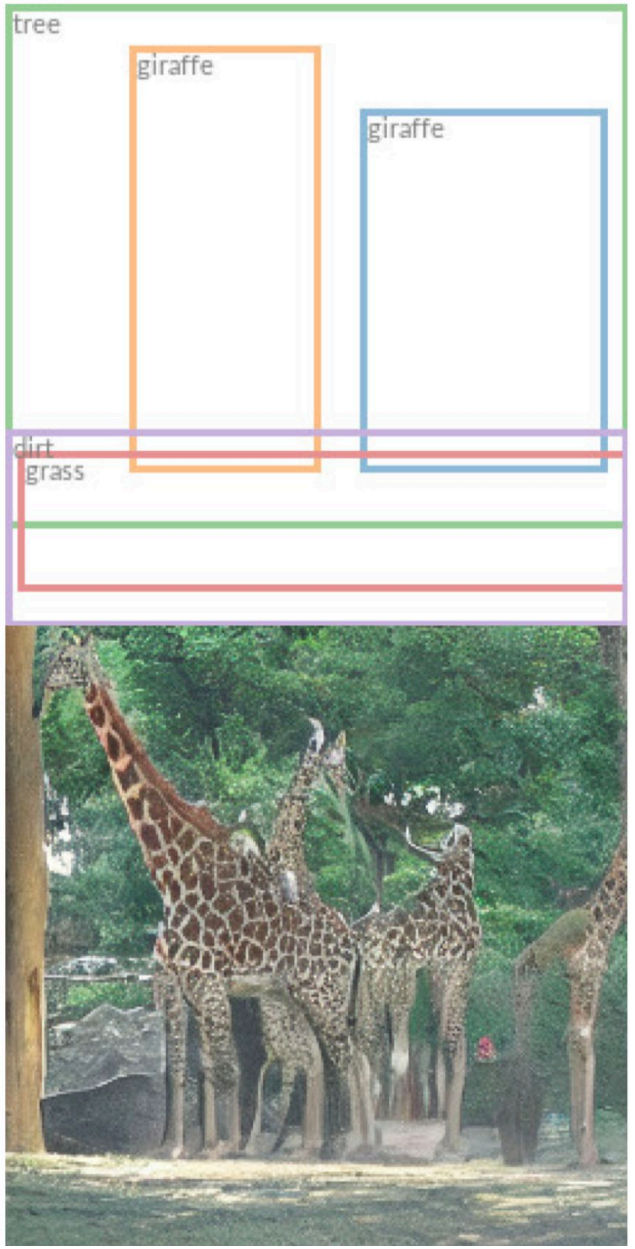
Text-to-Image Synthesis on LAION

'A street sign that reads "Latent Diffusion" '	*'A zombie in the style of Picasso'*	*'An image of an animal half mouse half octopus'*	*'An illustration of a slightly conscious neural network'*	*'A watercolor painting of a chair that looks like an octopus'*	*'A shirt with the inscription: "I love generative models!" '*




wall-concrete
ceiling-other
cabinet
shelf
floor-wood


tree
horse
fence
grass

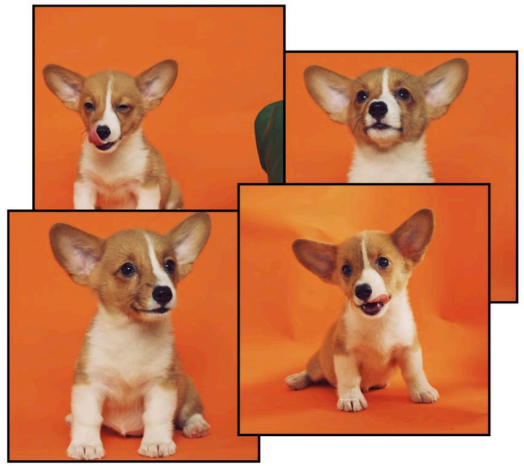

tree
giraffe
giraffe
dirt
grass

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation

Nataniel Ruiz[1,2], Yuanzhen Li[1], Varun Jampani[1], Yael Pritch[1], Michael Rubinstein[1], and Kfir Aberman[1]

[1]*Google Research* [2]*Boston University*

Motivation: Generate new context for single instance



Input images



in the Acropolis



swimming



sleeping



in a doghouse



in a bucket



getting a haircut



Input images



worn by a bear



in the jungle



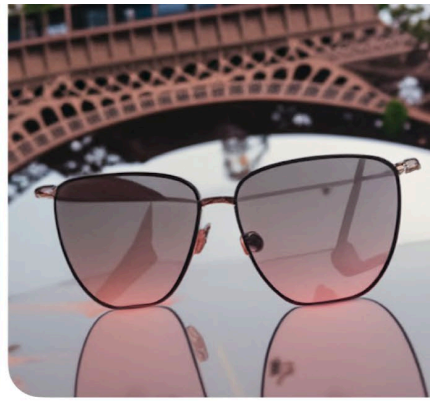
on red fabric



at Mt. Fuji



on top of snow



with Eiffel Tower

Input images







A [V] backpack in the Grand Canyon



A [V] backpack with the night sky



A [V] backpack in the city of Versailles



A wet [V] backpack in water



A [V] backpack in Boston

Input images





city of versailles

in water



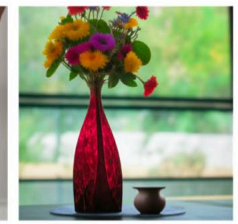
A [V] vase buried in the sands



Two [V] vases on a table



Milk poured into a [V] vase



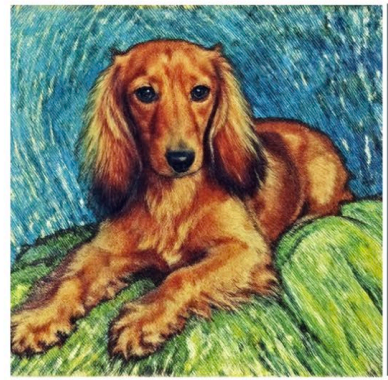
A [V] vase with a colorful flower bouquet



A [V] vase in the ocean

Input images





Vincent Van Gogh



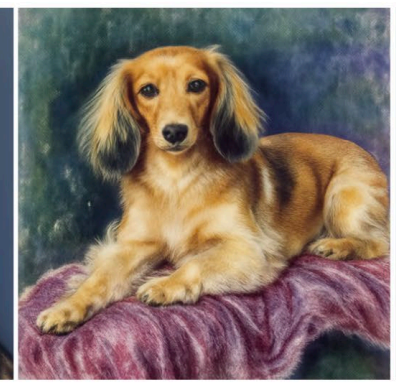
Michelangelo



Rembrandt



Johannes Vermeer

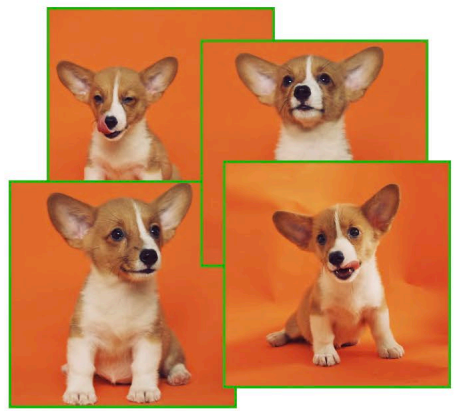


Pierre-Auguste Renoir

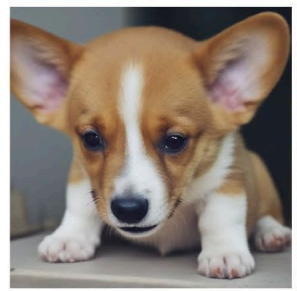


Leonardo da Vinci

Input images



Expression modification (¨*A [state] [V] dog*¨)



depressed



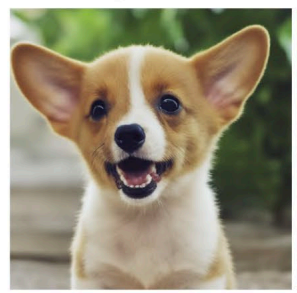
sleeping



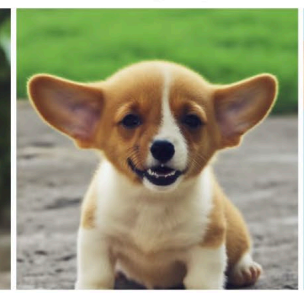
sad



joyous



barking



crying



frowning



screaming

Input images



Top view ↑



[V] cat seen from the top

Bottom view ↓



[V] cat seen from the bottom

Side view →



[V] cat seen from the side

Back view ↶



[V] cat seen from the back

Input images





Chef Outfit



Witch Outfit



Ironman Outfit



Nurse Outfit



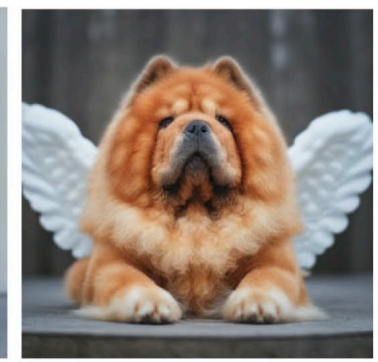
Purple Wizard Outfit



Superman Outfit



Police Outfit



Angel Wings

Color modification (*"A [color] [V] car"*)



Input



purple



red



yellow



blue



pink

Hybrids (*"A cross of a [V] dog and a [target species]"*)



Input



bear



panda



koala



lion



hippo

Video Generation: OpenAI Sora

Several giant wooly mammoths approach treading through a snowy meadow, their long wooly fur lightly blows in the wind as they walk, snow covered trees and dramatic snow capped mountains in the distance, mid afternoon light with wispy clouds and a sun high in the distance creates a warm glow, the low camera view is stunning capturing the large furry mammal with beautiful photography, depth of field.



Source: https://openai.com/index/sora/

Video Generation: Google Veo2

A low-angle shot captures a flock of pink flamingos gracefully wading in a lush, tranquil lagoon. The vibrant pink of their plumage contrasts beautifully with the verdant green of the surrounding vegetation and the crystal-clear turquoise water. Sunlight glints off the water's surface, creating shimmering reflections that dance on the flamingos' feathers. The birds' elegant, curved necks are submerged as they walk through the shallow water, their movements creating gentle ripples that spread across the lagoon. The composition emphasizes the serenity and natural beauty of the scene, highlighting the delicate balance of the ecosystem and the inherent grace of these magnificent birds. The soft, diffused light of early morning bathes the entire scene in a warm, ethereal glow.



Source: https://blog.google/technology/google-labs/video-image-generation-update-december-2024/

More Videos

https://x.com/minchoi

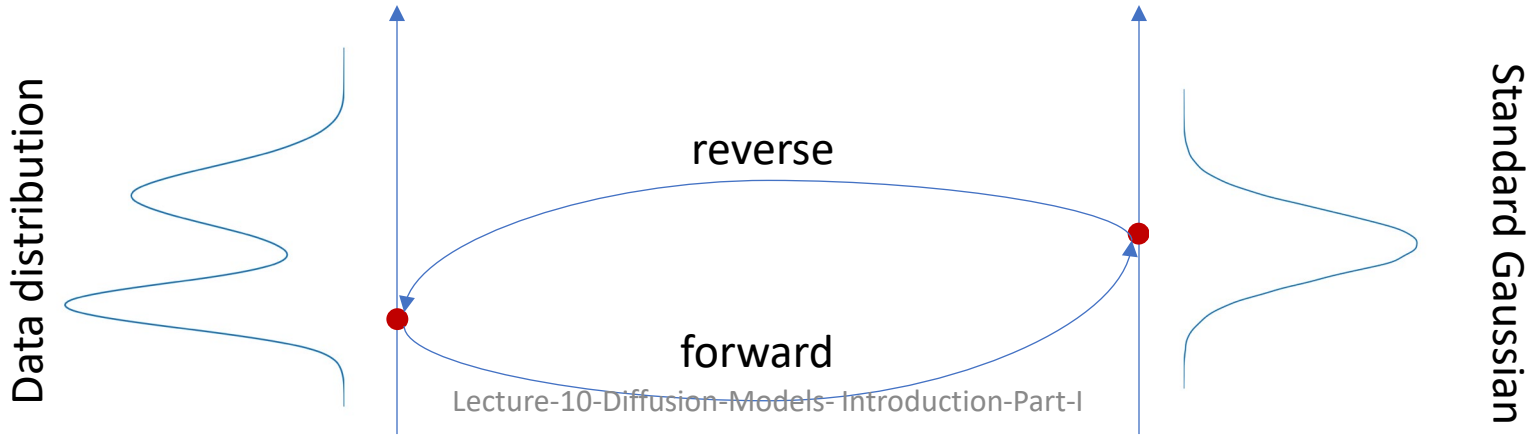
https://x.com/EHuanglu

Outline

1. Motivation
2. High-level overview
3. Denoising diffusion probabilistic models
4. Noise Conditioned Score Network
5. Conditional Generation
6. Stochastic Differential Equations
7. Research directions

High-level overview

- Diffusion models are probabilistic models used for image generation
- They involve reversing the process of gradually degrading the data
- Consist of two processes:
  - **The forward process**: data is progressively destroyed by adding noise across multiple time steps
  - **The reverse process**: using a neural network, noise is sequentially removed to obtain the original data



High-level overview

- Three categories:
  - Denoising Diffusion Probabilistic Models (DDPM)
  - Noise Conditioned Score Networks (NCSN)
  - Stochastic Differential Equations (SDE)

Outline

1. Motivation
2. High-level overview
3. Denoising diffusion probabilistic models
4. Noise Conditioned Score Network
5. Conditional Generation
6. Stochastic Differential Equations
7. Research directions

Notations

$q(x)$ - data distribution

$\mathcal{N}(x; \mu, \sigma \cdot I)$ - Gaussian distribution

Random Variable (image)

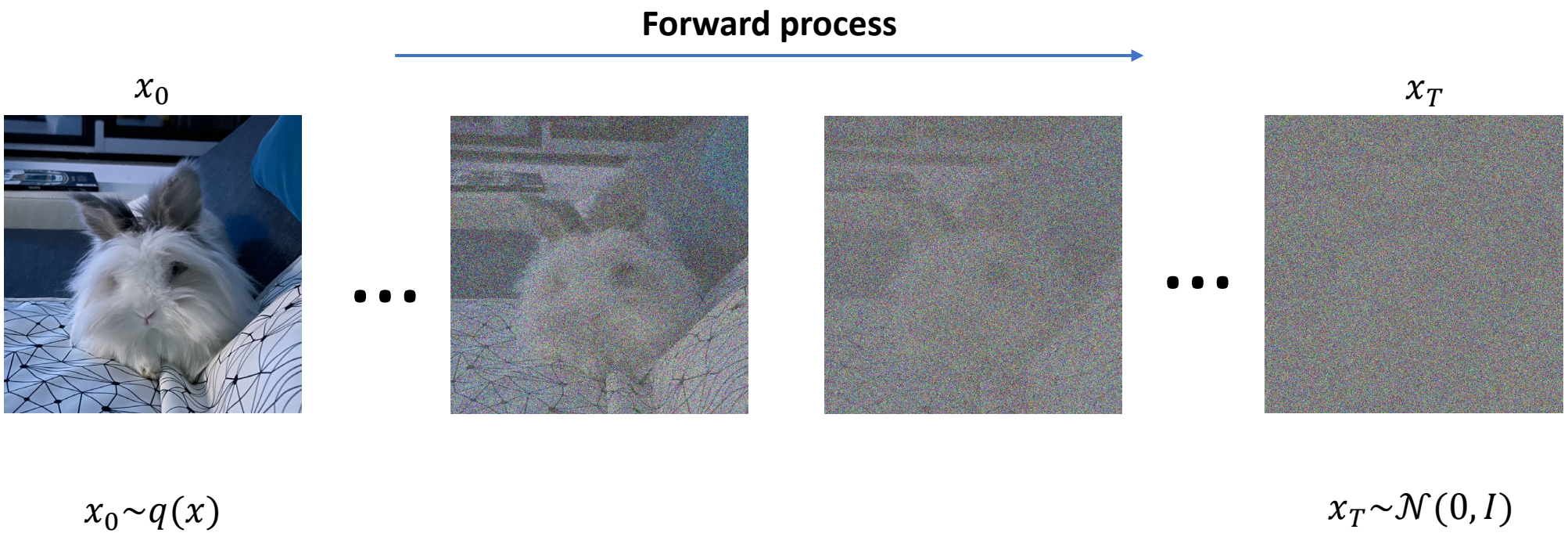
Mean Vector

Covariance matrix. I is the identity matrix

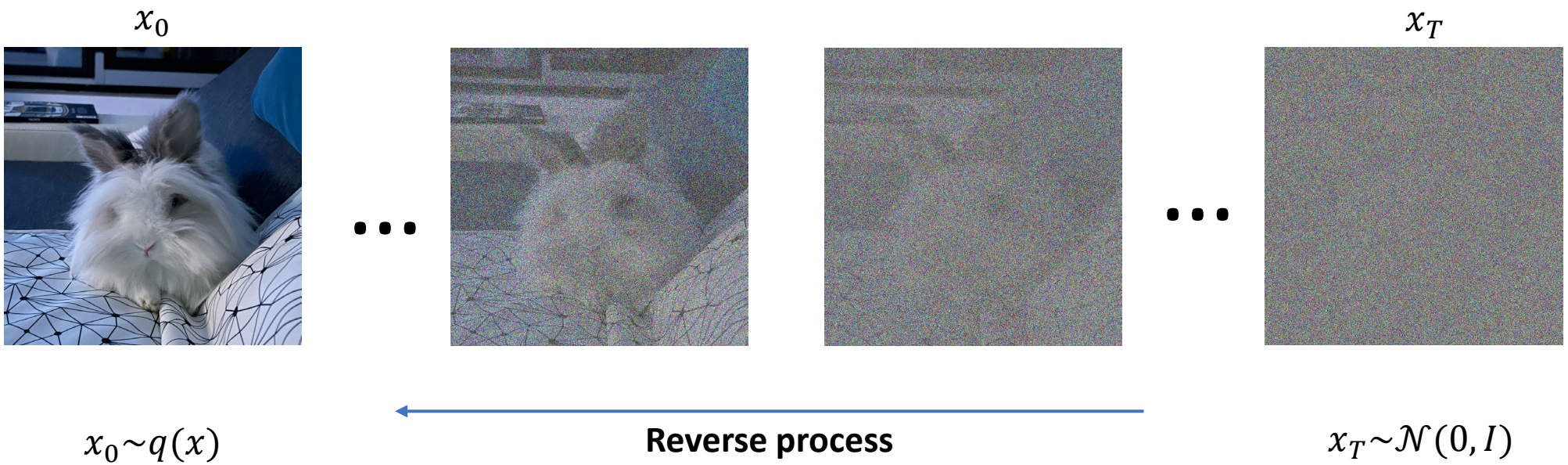
$$x = \mu + \sqrt{\sigma} \cdot z, z \sim \mathcal{N}(0, I)$$

Sample from this distribution

Denoising **D**iffusion **P**robabilistic **M**odels (DDPMs)



Denoising **D**iffusion **P**robabilistic **M**odels (DDPMs)



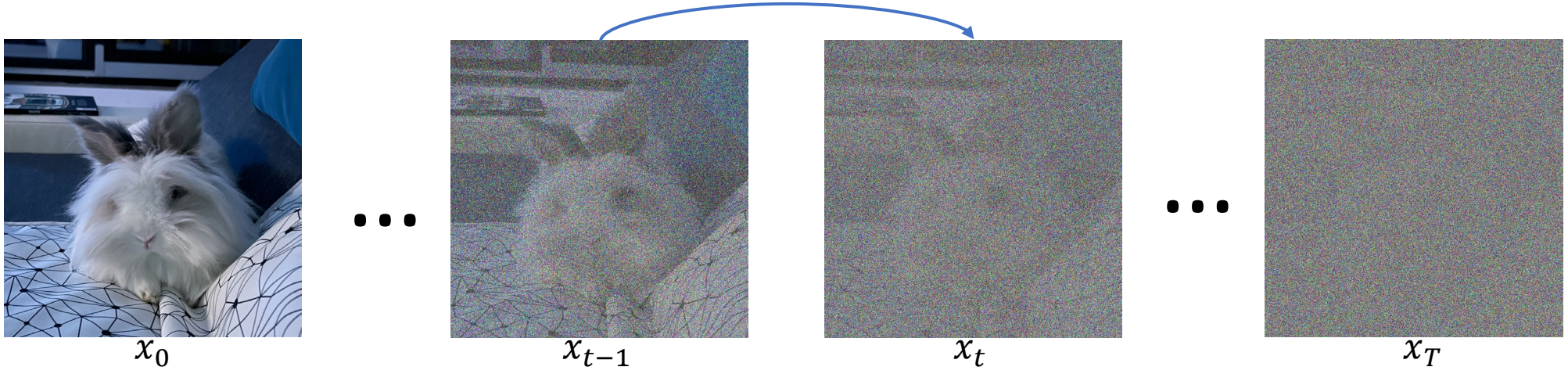
Denoising Diffusion Probabilistic Models (DDPMs)

Forward process (Iterative)

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \boldsymbol{\Sigma}_t = \beta_t\mathbf{I}) \quad \beta_t \ll 1\,, t = \overline{1,T}$$

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_{t-1} \quad \epsilon_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

The image is replaced with noise



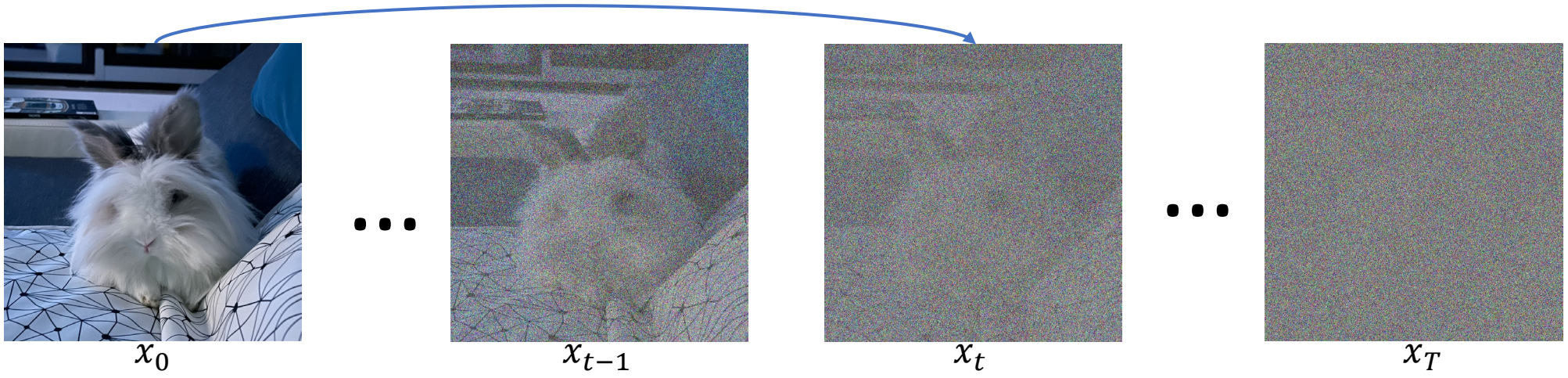
How do we go from x_0 to x_t?

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

Denoising Diffusion Probabilistic Models (DDPMs)

Forward process. Ancestral sampling (One Shot)

Can we go directly in one shot from x_0 to x_t?



Re-parameterization Trick

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_{t-1} \quad \epsilon_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

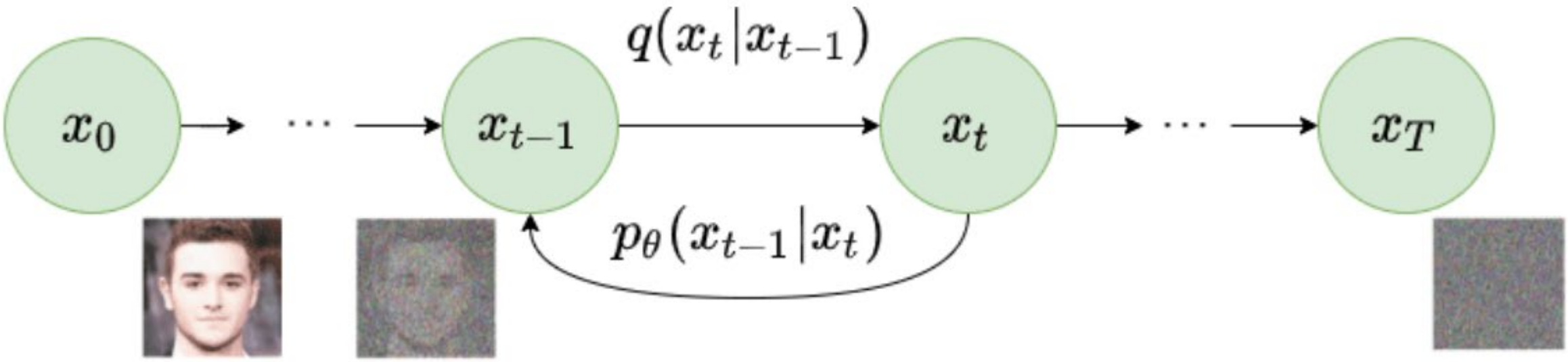
$$\begin{aligned}
\mathbf{x}_t &= \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-2} \\
&= \dots \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon_0}
\end{aligned}$$

$$\alpha_t = 1 - \beta_t, \quad \bar{\alpha}_t = \prod_{s=0}^{t} \alpha_s \quad \text{where} \quad \boldsymbol{\epsilon}_0, \dots, \epsilon_{t-2}, \epsilon_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

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

DDPMs. Training

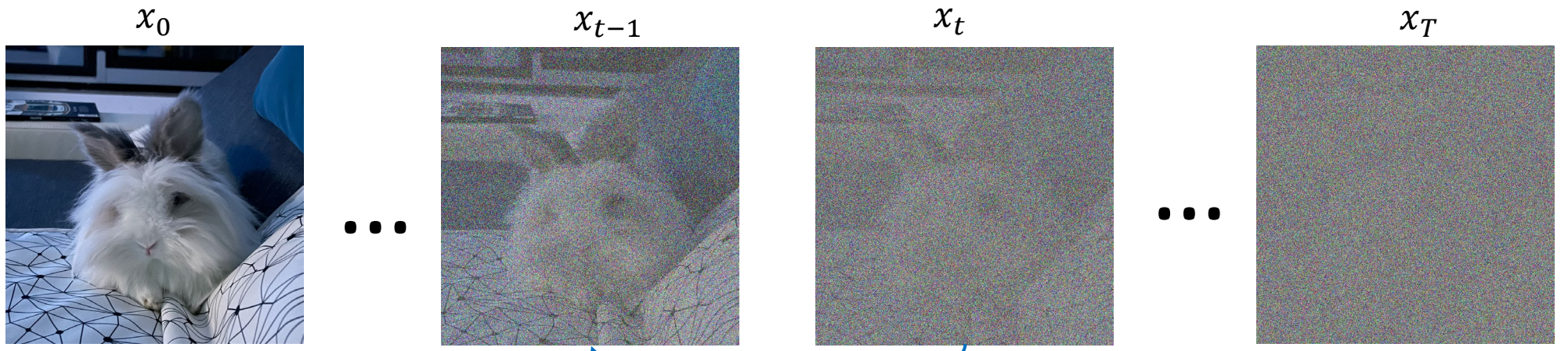
$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$



Reverse diffusion process. Image modified by Ho et al. 2020

DDPMs. Training objective

Remember that:

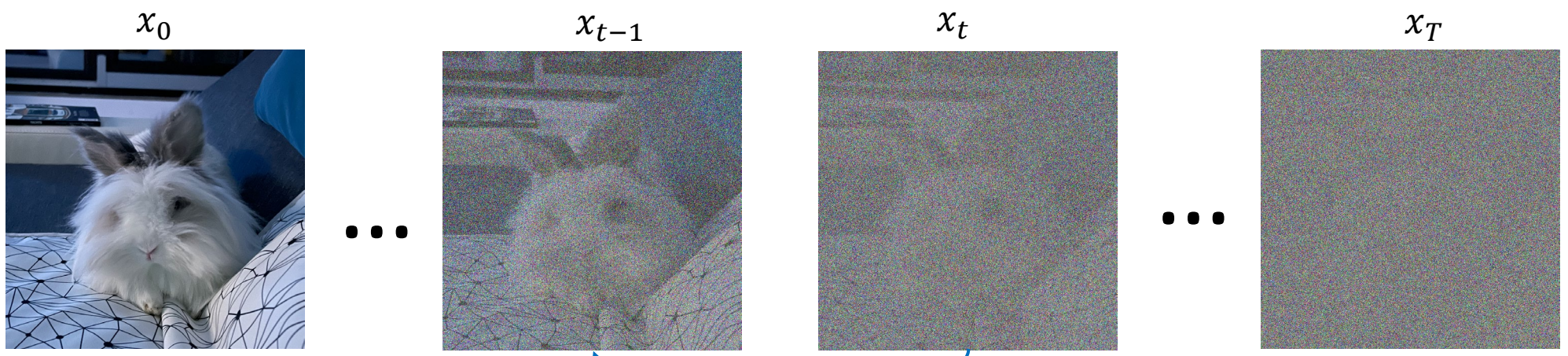


Neural network weights

Approximated by a neural network

DDPMs. Training objective

Simplification:



Neural network weights

Approximated by a neural network

Fix the variance instead of learning, and predict/learn the mean

DDPMs. Training objective

UNet-like neural network



x_t



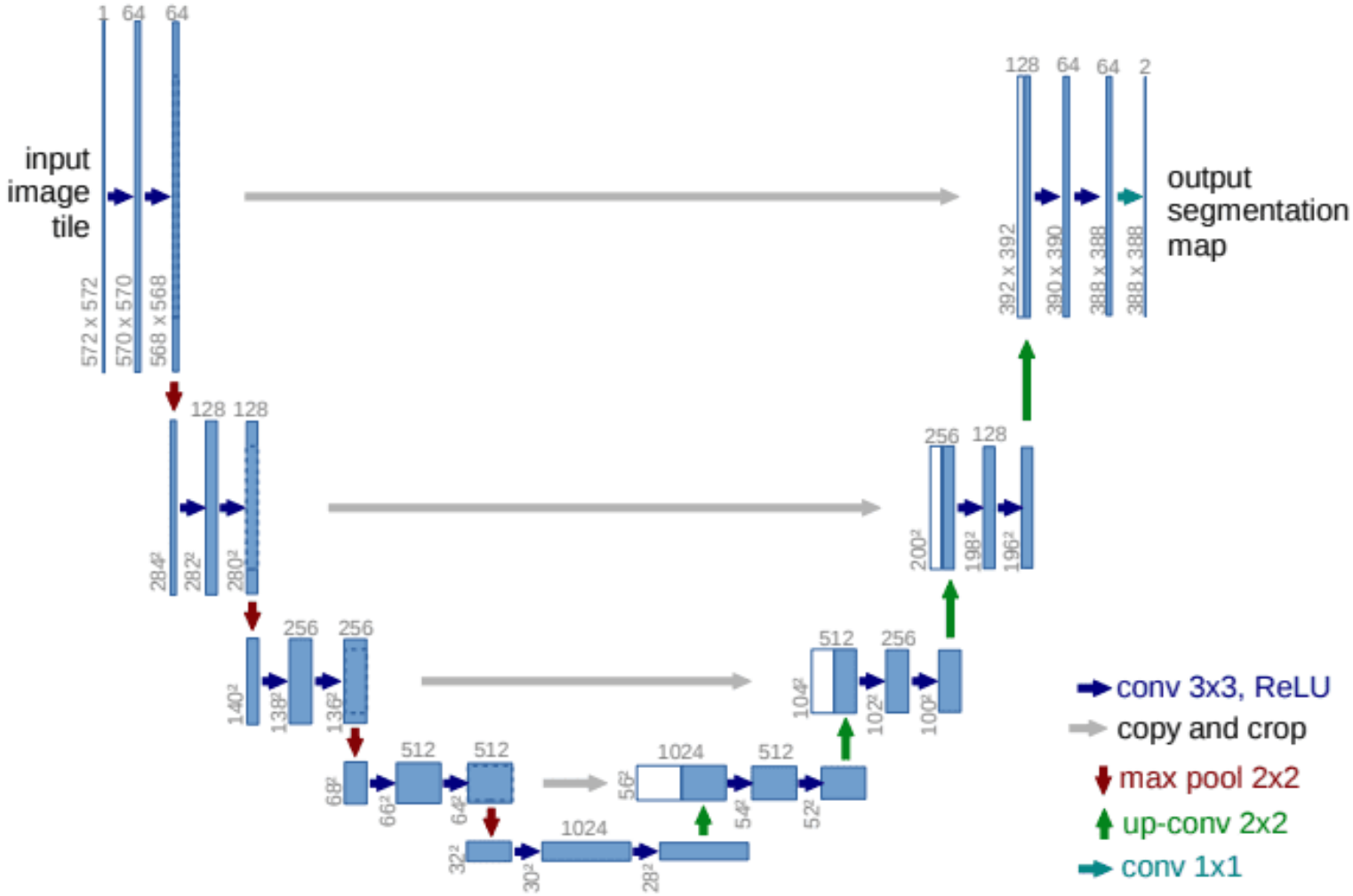
x_{t-1}

t

$\mu_\theta(x_t, t)$

$\sim \mathcal{N}(x_{t-1}, \mu_\theta(x_t, t), \sigma_t^2 \mathrm{I})$

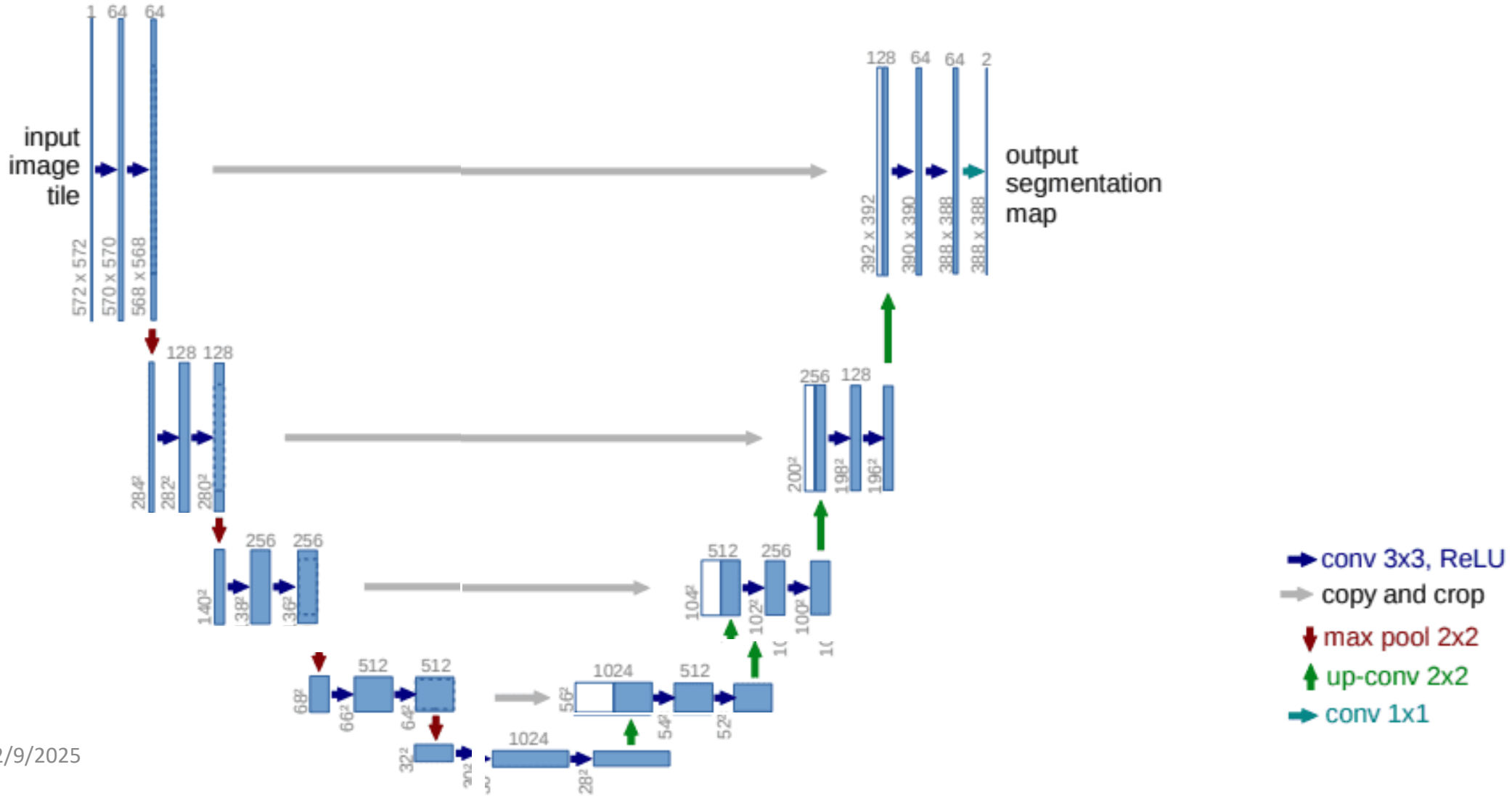
U-Net



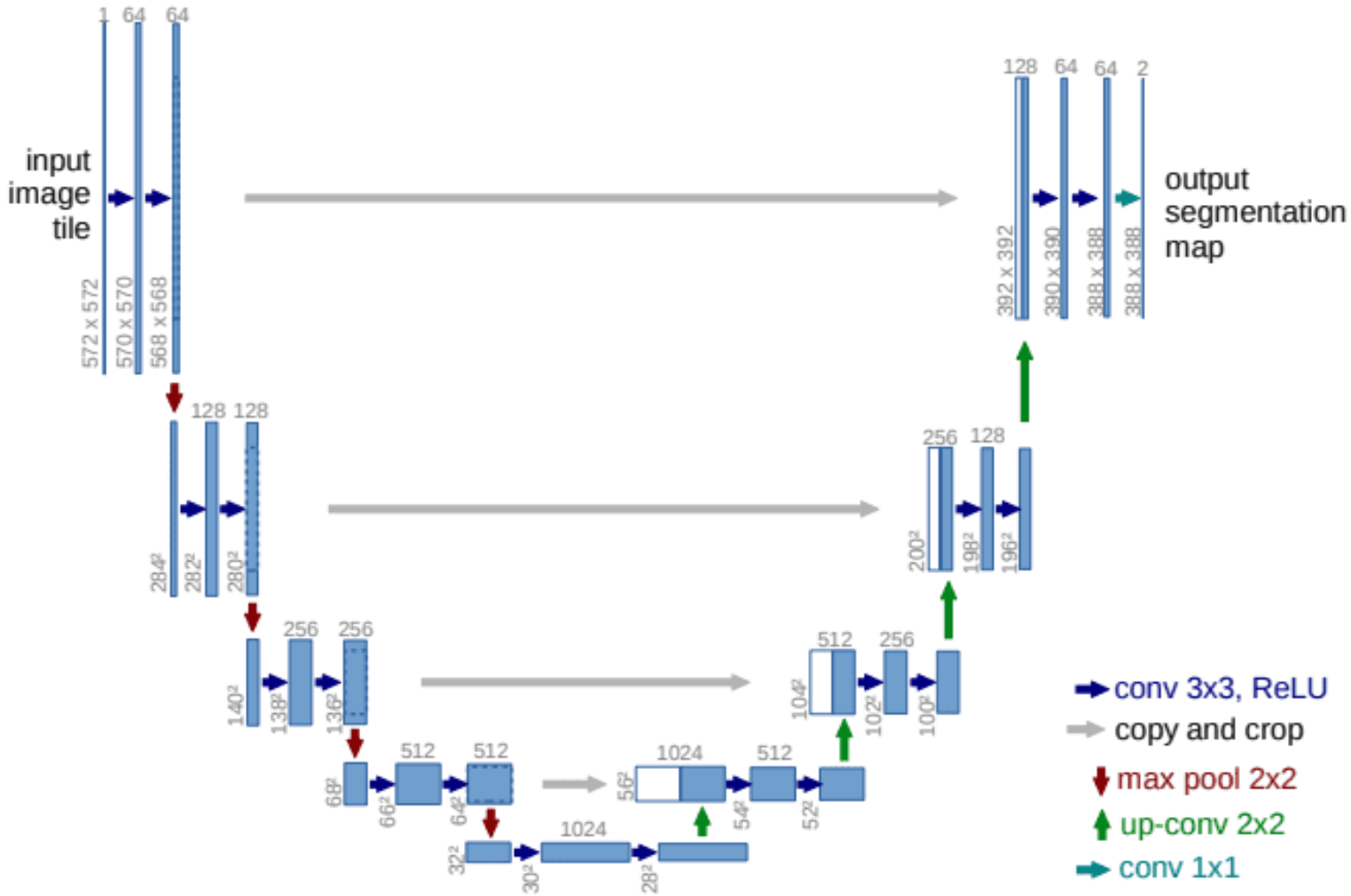
U-net: Convolutional networks for biomedical image segmentation

O Ronneberger, P Fischer, T **Brox** - Munich, Germany, October 5-9, MICCAI, 201 – Springe: Cited by 100914

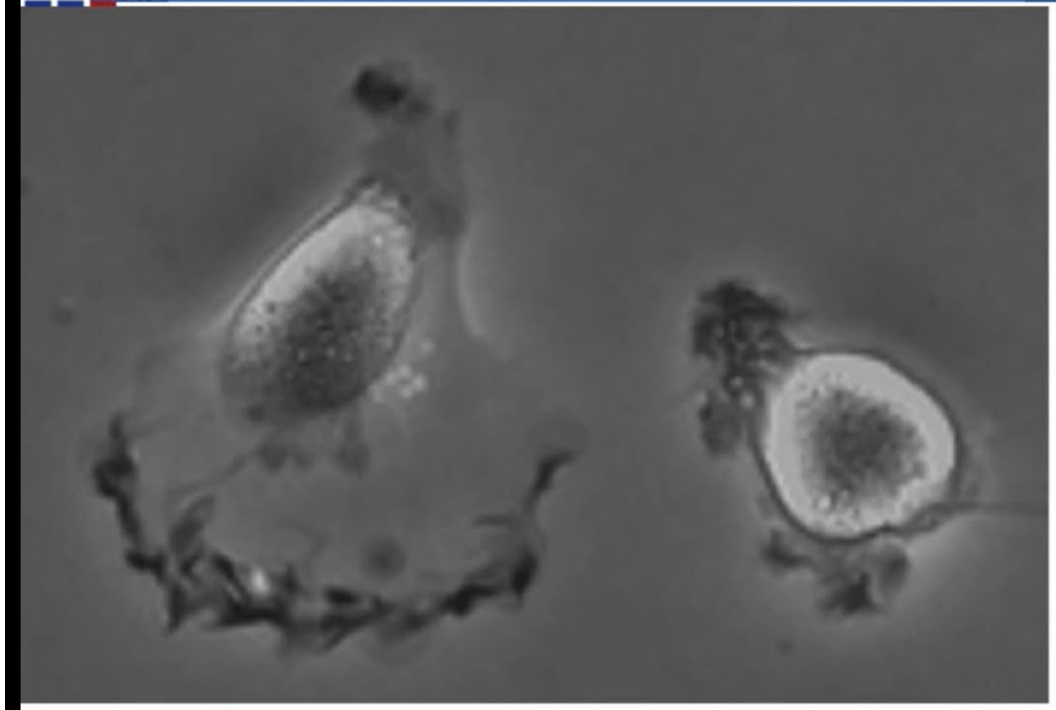
U-Net


input image tile
572 x 572
570 x 570
568 x 568
1
64
64
128
128
256
256
512
512
1024
1024
512
512
256
256
128
128
output segmentation map
392 x 392
390 x 390
388 x 388
388 x 388
128
64
64
2
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

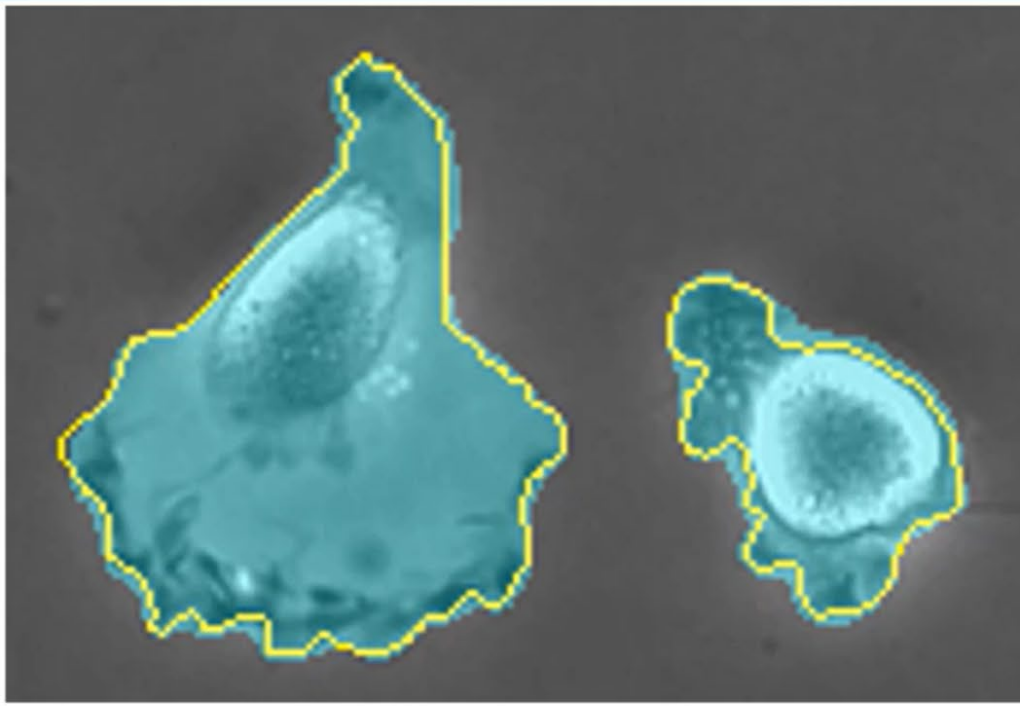
U-Net


input image tile
572 x 572
570 x 570
568 x 568
1 64 64
128 128
284²
282²
280²
256 256
140²
138²
136²
512 512
68²
66²
64²
1024
32²
30²
28²
1024
56²
512
54²
512
52²
512 256
104²
102²
100²
256 128
200²
198²
196²
128 64 64 2
392 x 392
390 x 390
388 x 388
388 x 388
output segmentation map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

ISBI cell tracking challenge 2015

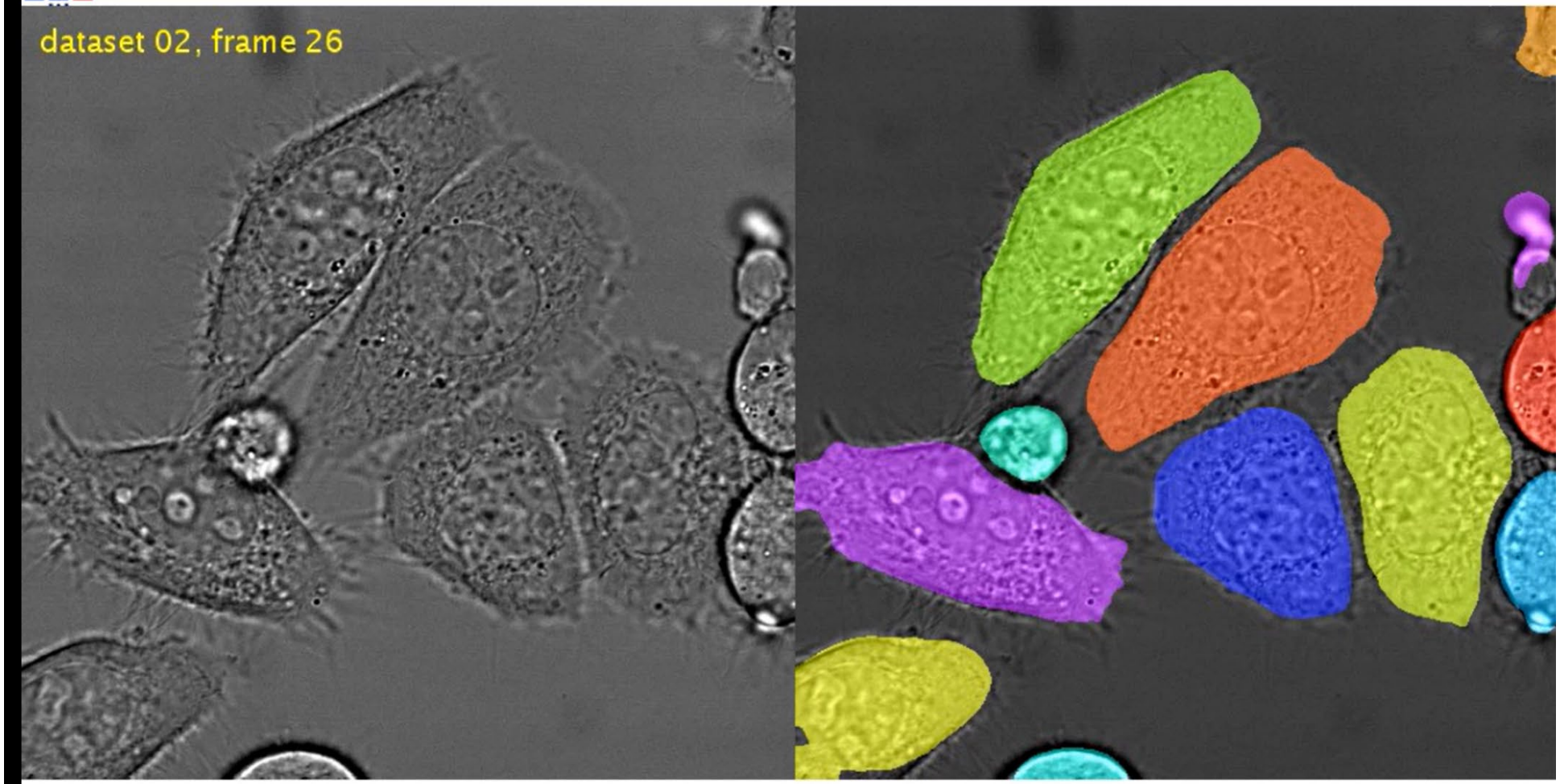


Glioblastoma-astrocytoma U373 cells on a polyacrylimide substrate
Phase contrast microscopy



Cyan: segmentation by u-net
Yellow borders: our manual ground truth

ISBI cell tracking challenge: **DIC-HeLa**



Our Results: 77.6%, 2nd: 46.0%

More details and implementation:
http://lmb.informatik.uni-freiburg.de/people/ronneber/u-net

Data provided by Dr. Ger
van Cappellen
Erasmus Medical Center

DDPMs. Training Objective

Cross Entropy and KL (Kullback-Leibler) divergence

- Entropy: $E(P) = - \Sigma_i P(i) \log P(i)$
- Cross Entropy: $C(P) = - \Sigma_i P(i) \log Q(i)$
- KL divergence: $D_{KL}(P \,||\, Q) = \Sigma_i P(i) \log[P(i)/Q(i)] = \Sigma_i P(i)[\log P(i) - \log Q(i)]$

DDPMs. Training Objective

$$\min_{\theta} \mathbb{E}_{x_0 \sim q(x_0)} \left[-\log p_\theta(x_0|x_1) + KL(q(x_T|x_0)||p_\theta(x_T)) + \sum_{t=2}^{T} KL(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) \right]$$

This term can be ignored because $p_\theta(x_T)$ is $\mathcal{N}(0, \mathrm{I})$ and does not depend on θ.

At each time step t, $p_\theta(x_{t-1}|x_t)$ is as close as possible to the true probability of the forward process when conditioned on the original image.

DDPMs. Training Objective. Simplifications

$$\min_{\theta} \mathbb{E}_{x_0 \sim q(x_0)} \left[-\log p_\theta(x_0|x_1) + \sum_{t=2}^{T} KL(q(x_{t-1}|x_t,\ x_0)||p_\theta(x_{t-1}|x_t)) \right]$$

- The first term measures the reconstruction error and can be addressed with an independent decoder.
- The KL divergences of 2 gaussians is L2 distance between their means
- DDPMs paper introduced two simplifications that led to a much simple objective that is based on the noise in the image.

DDPMs. Training Objective. Simplifications

$$\min_{\theta} \mathbb{E}_{x_0 \sim q(x_0)} \left[-\log p_\theta(x_0|x_1) + \sum_{t=2}^{T} KL(q(x_{t-1}|x_t,\ x_0)||p_\theta(x_{t-1}|x_t)) \right]$$

$$\mathrm{q}(x_{t-1}|x_t, x_0) = \mathcal{N}\left(x_{t-1};\ \tilde{\mu}(x_t, x_0), \tilde{\beta}_t I\right)$$

$$\tilde{\mu}(x_t, x_0) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \hat{\beta}_t}} z_t \right), z_t \sim \mathcal{N}(0, \mathrm{I})$$

$$\tilde{\beta}_t = \frac{1 - \hat{\beta}_{t-1}}{1 - \hat{\beta}_t} \cdot \beta_t$$

Notations:

$$\hat{\beta}_t = \prod_{i=1}^{t} \alpha_i$$

$$\alpha_t = 1 - \beta_t$$

DDPMs. Training Objective. Simplifications

$$\min_{\theta} \mathbb{E}_{x_0 \sim p(x_0)} \left[-\log p_\theta(x_0|x_1) + \sum_{t=2}^{T} KL(p(x_{t-1}|x_t,\ x_0)||p_\theta(x_{t-1}|x_t)) \right]$$

Tractable posterior:

$$\mathrm{q}(x_{t-1}|x_t, x_0) = \mathcal{N}\left(x_{t-1};\ \tilde{\mu}(x_t, x_0), \tilde{\beta}_t I\right)$$

$$\left.\begin{aligned} \tilde{\mu}(x_t, x_0) &= \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \hat{\beta}_t}}\, z_t \right), z_t \sim \mathcal{N}(0, \mathrm{I}) \\ \mu_\theta(x_t, x_0) &= \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \hat{\beta}_t}}\, z_\theta(x_t, t) \right) \end{aligned}\right\} \Rightarrow$$

Notations:

$$\hat{\beta}_t = \prod_{i=1}^{t} \alpha_i$$

$$\alpha_t = 1 - \beta_t$$

$$\tilde{\beta}_t = \frac{1 - \hat{\beta}_{t-1}}{1 - \hat{\beta}_t} \cdot \beta_t$$

$$\Rightarrow KL(q(x_{t-1}|x_t,\ x_0)\big|\big|p_\theta(x_{t-1}|x_t)\big) = \mathbb{E}_{z_t \sim \mathcal{N}(0,\mathrm{I})} \left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1 - \hat{\beta}_t)} \|z_t - z_\theta(x_t, t)\|_2^2 \right]$$

DDPMs. Training Objective. Simplifications

$$\min_{\theta} \mathbb{E}_{x_0 \sim p(x_0)} \left[-\log p_\theta(x_0|x_1) + \sum_{t=2}^{T} KL(p(x_{t-1}|x_t,\ x_0)||p_\theta(x_{t-1}|x_t)) \right]$$

Notations:

$$\hat{\beta}_t = \prod_{i=1}^{t} \alpha_i$$

$$\alpha_t = 1 - \beta_t$$

$$\tilde{\beta}_t = \frac{1-\hat{\beta}_{t-1}}{1-\hat{\beta}_t} \cdot \beta_t$$

$$p(x_{t-1}|x_t, x_0) = \mathcal{N}\left(x_{t-1};\ \tilde{\mu}(x_t, x_0), \tilde{\beta}_t I\right)$$

$$\left.\begin{aligned} \tilde{\mu}(x_t, x_0) &= \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\hat{\beta}_t}} z_t \right), z_t \sim \mathcal{N}(0, \mathrm{I}) \\ \mu_\theta(x_t, x_0) &= \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\hat{\beta}_t}} z_\theta(x_t, t) \right) \end{aligned}\right\} \Rightarrow$$

Ignored

$$\Rightarrow KL(q(x_{t-1}|x_t,\ x_0)||p_\theta(x_{t-1}|x_t)) = \mathbb{E}_{z_t \sim \mathcal{N}(0,\mathrm{I})} \left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\hat{\beta}_t)} \|z_t - z_\theta(x_t, t)\|_2^2 \right]$$

DDPMs. Training Algorithm

$$\min_{\theta} \underbrace{\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{x_0 \sim q(x_0), z_t \sim \mathcal{N}(0,\mathrm{I})} \|z_t - z_\theta(x_t, t)\|_2^2}_{\mathcal{L}_{simple}}$$

Training algorithm:

Repeat

$x_0 \sim q(x_0)$

$t \sim \mathcal{U}(\{1, \dots, T\})$

$z_t \sim \mathcal{N}(0, \mathrm{I})$

$x_t = \sqrt{\hat{\beta}_t} \cdot x_0 + \sqrt{(1 - \hat{\beta}_t)} z_t$

$\theta = \theta - lr \cdot \nabla_\theta \mathcal{L}_{simple}$

Until convergence

$$\hat{\beta}_t = \prod_{i=1}^{t} \alpha_i$$

DDPMs. Training Algorithm

$$\min_{\theta} \underbrace{\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{x_0 \sim q(x_0), z_t \sim \mathcal{N}(0,\mathrm{I})} \|z_t - z_\theta(x_t, t)\|_2^2}_{\mathcal{L}_{simple}}$$

$$\hat{\beta}_t = \prod_{i=1}^{t} \alpha_i$$

Training algorithm:

Repeat

$x_0 \sim q(x_0)$ %We sample an image from our data set

$t \sim \mathcal{U}(\{1, \dots, T\})$ %choose randomly a time step t of the forward process

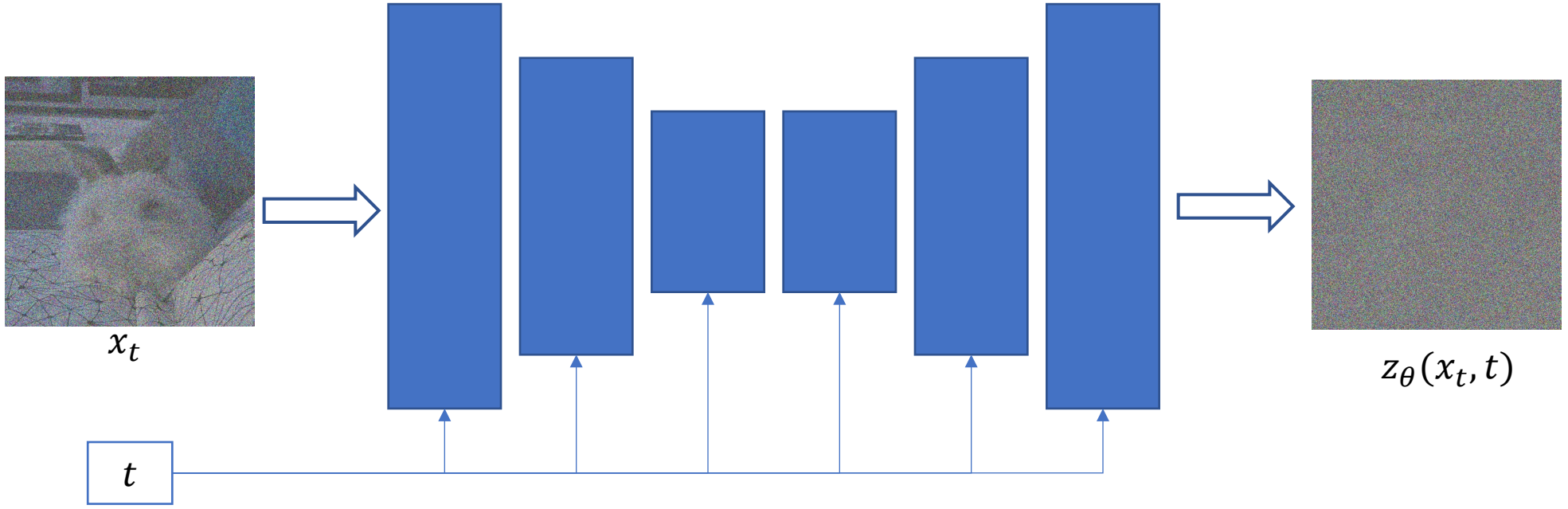
$z_t \sim \mathcal{N}(0, \mathrm{I})$ %sample the noise z_t

$x_t = \sqrt{\hat{\beta}_t} \cdot x_0 + \sqrt{(1 - \hat{\beta}_t)} z_t$ % Get noisy image

$\theta = \theta - lr \cdot \nabla_\theta \mathcal{L}_{simple}$ %Update neural network weights

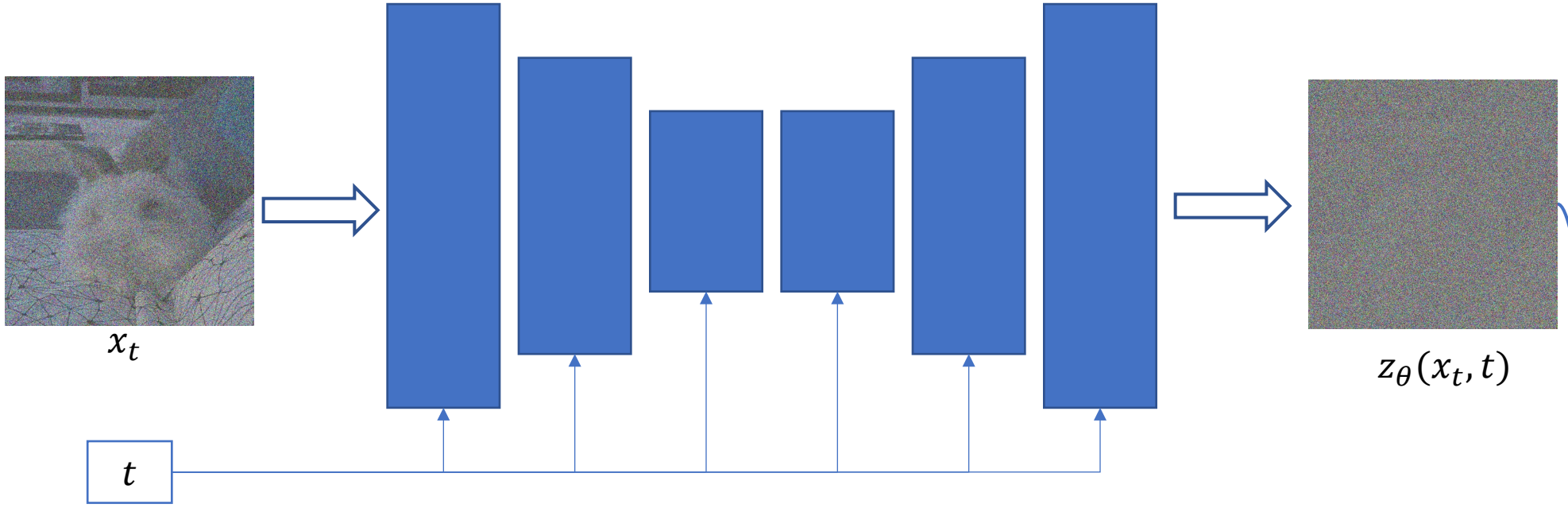
Until convergence

DDPMs. Sampling



- Pass the current noisy image along with *t* to the neural network
- With the resultant z_θ compute the mean of the gaussian distribution

DDPMs. Sampling



Sample the image x_{t-1} for the next iteration



x_{t-1}

$$\sim \mathcal{N}\left(x_{t-1}, \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\hat{\beta}_t}} z_\theta(x_t, t)\right), \sigma_t^2 \mathrm{I}\right)$$

$\mu_\theta(x_t, t)$

Algorithm 2 Sampling

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