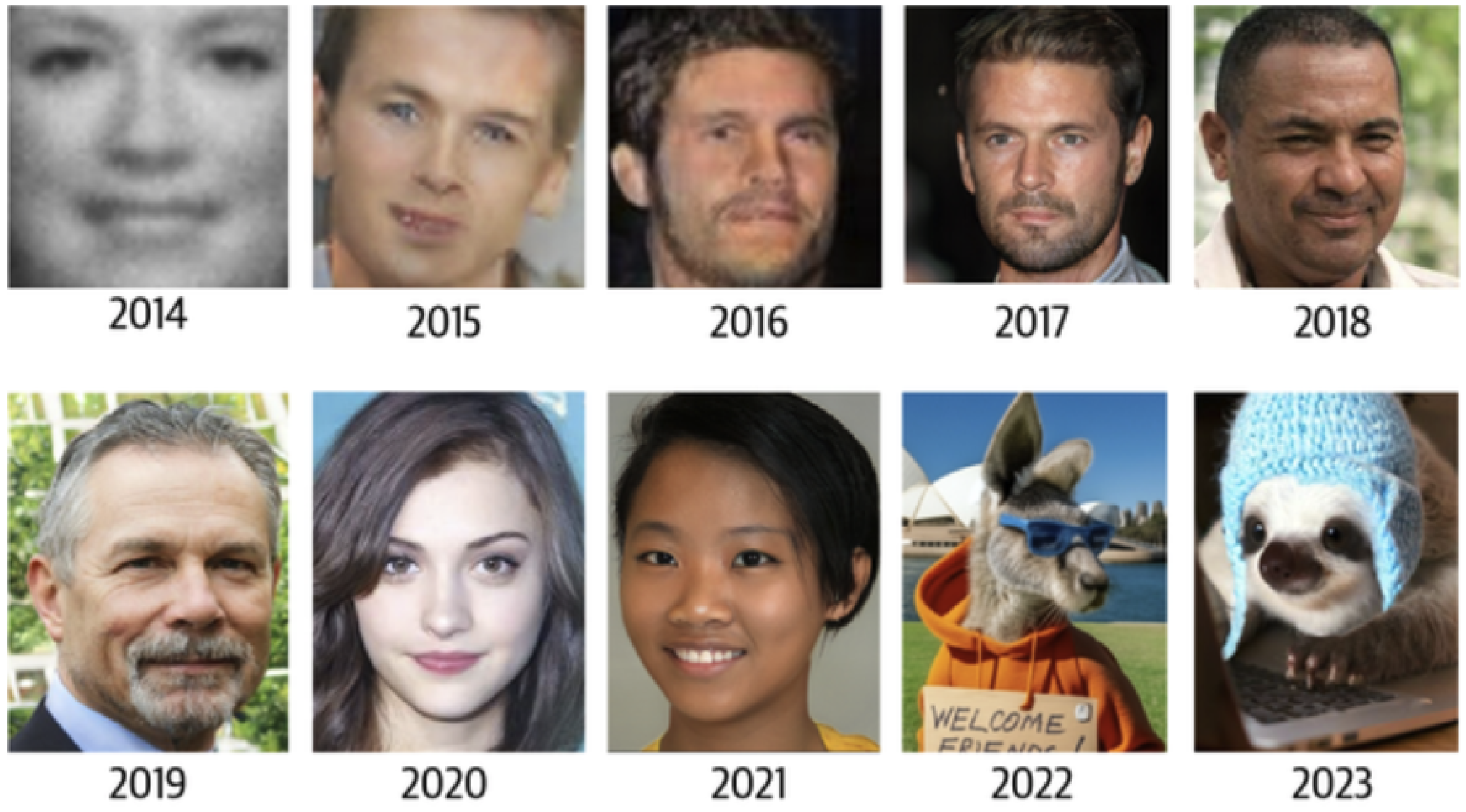
Vision Models

Philipp Koehn

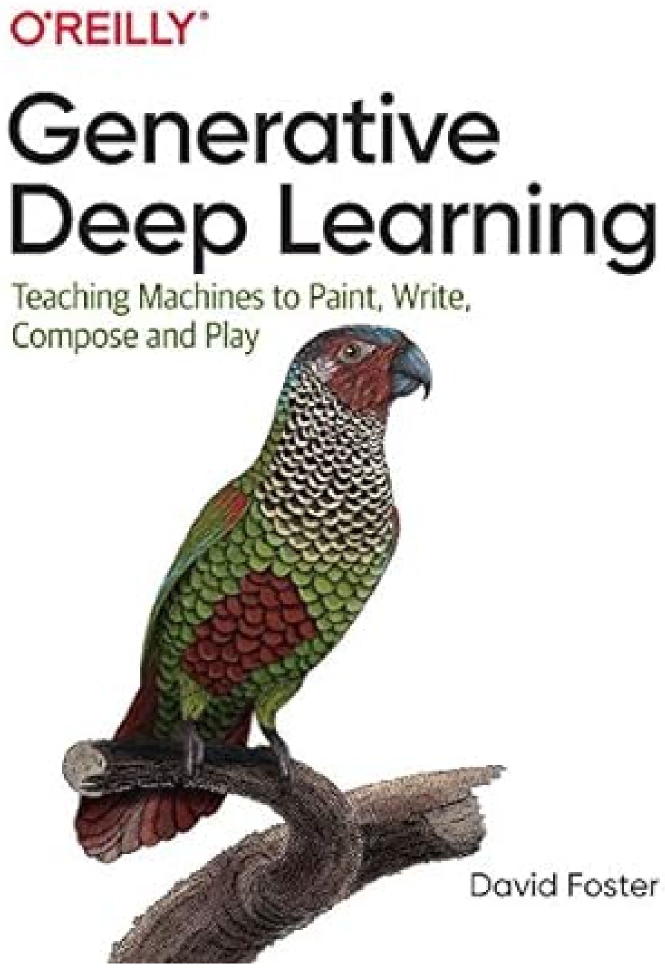
25 April 2024

Image Generation



- How does this work?

Textbook



O'REILLY

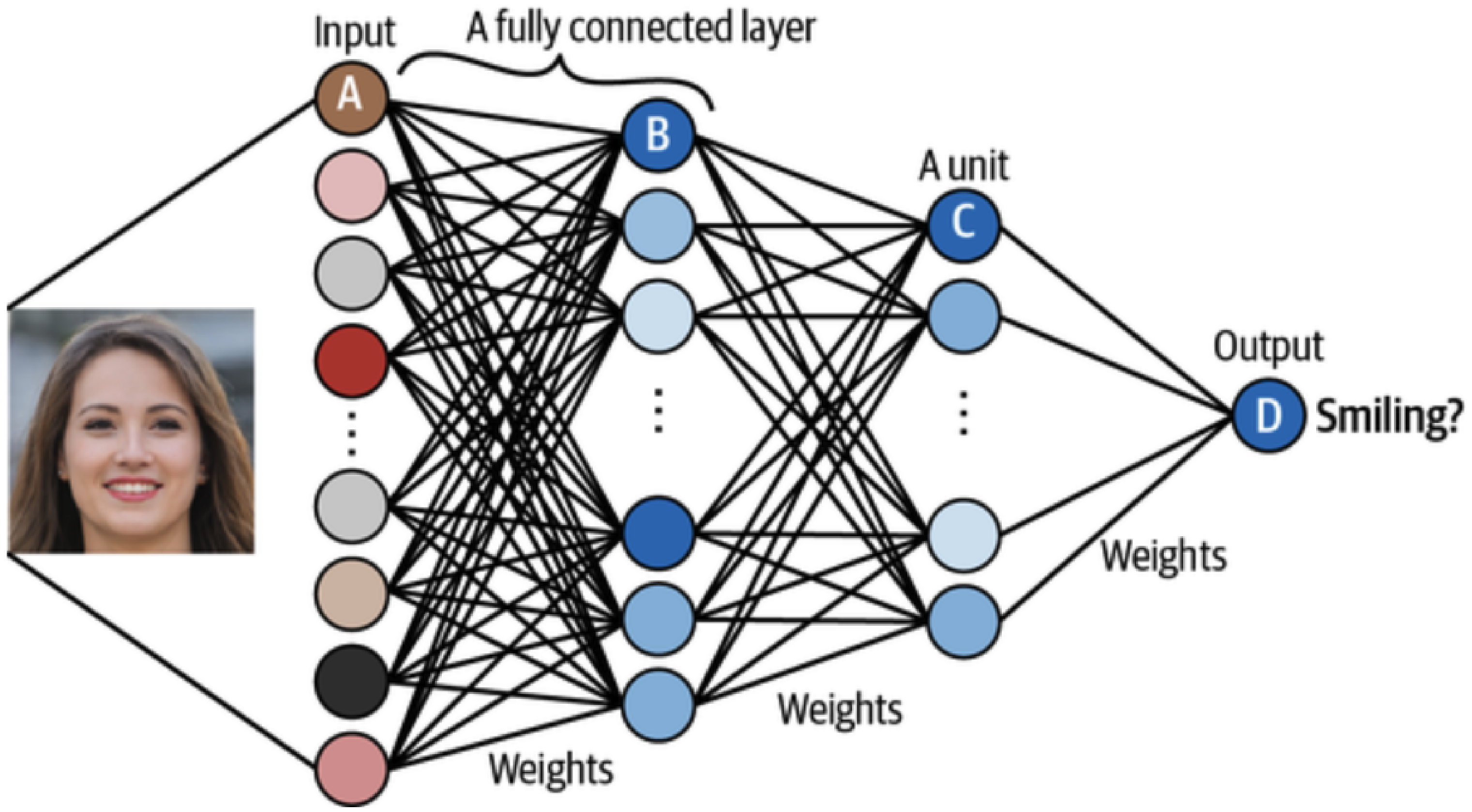
Generative Deep Learning

Teaching Machines to Paint, Write, Compose and Play

David Foster

image classification

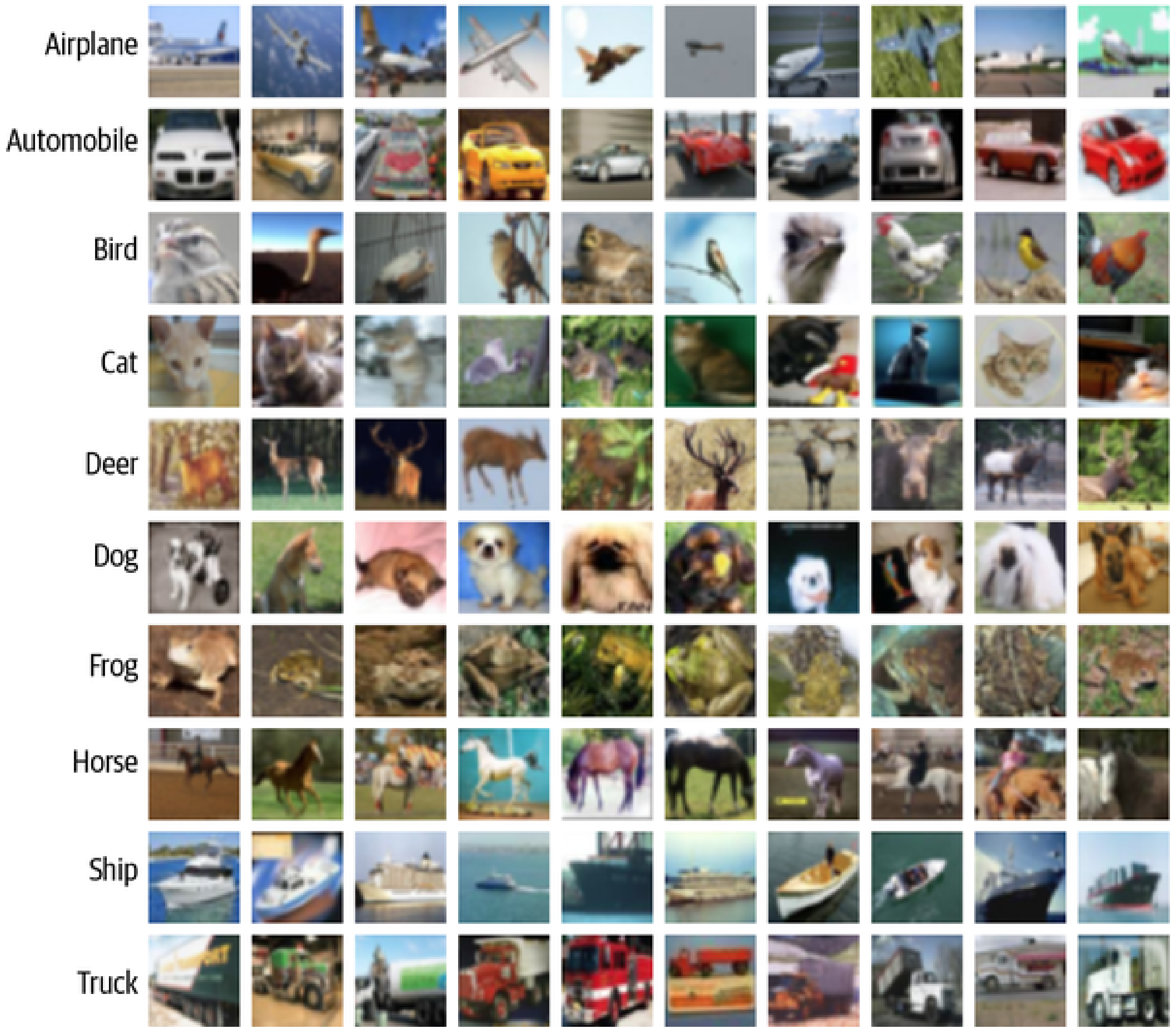
Image Classification



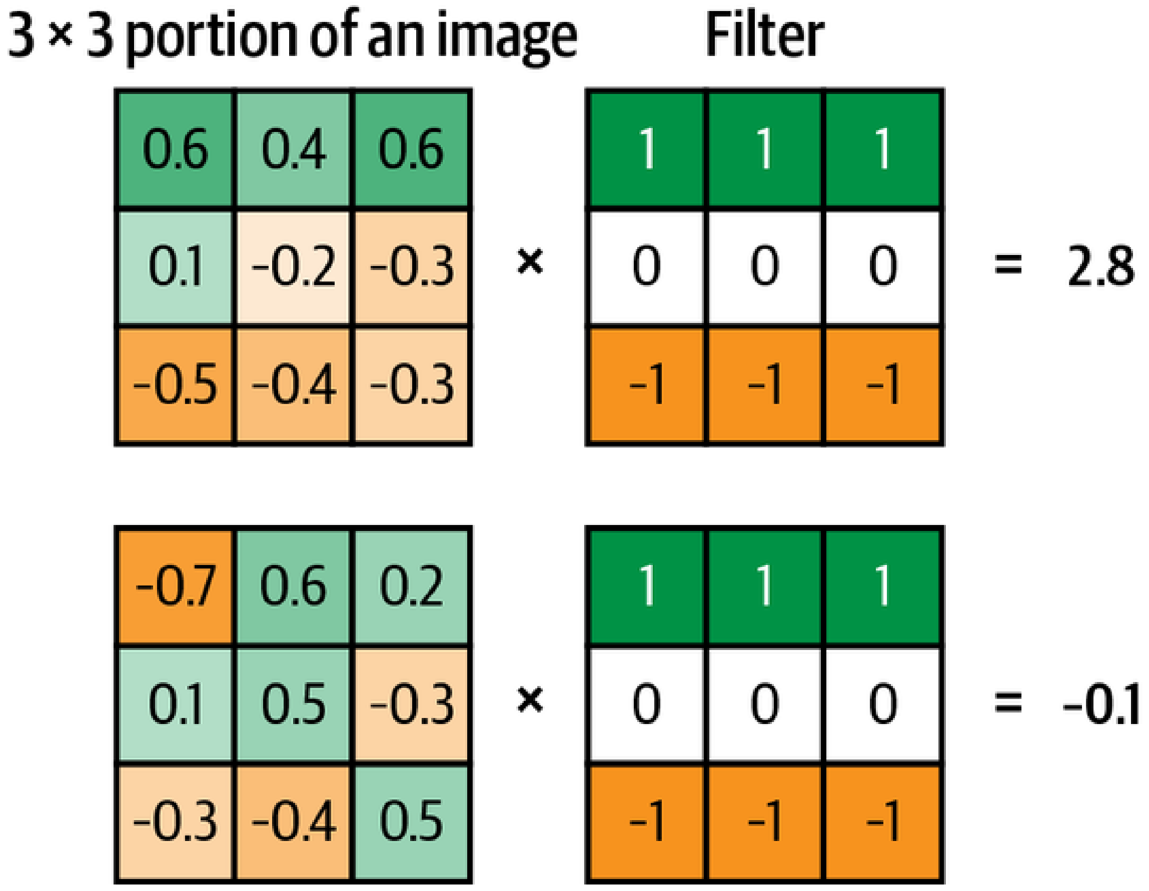
- Input: Image, Output: Class

CIFAR-10

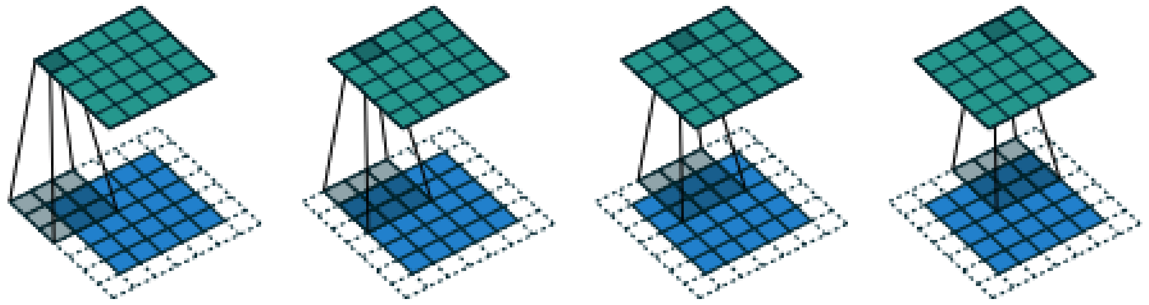
- 60,000 images
- 32x32 resolution
- 10 classes



Simple Convolution

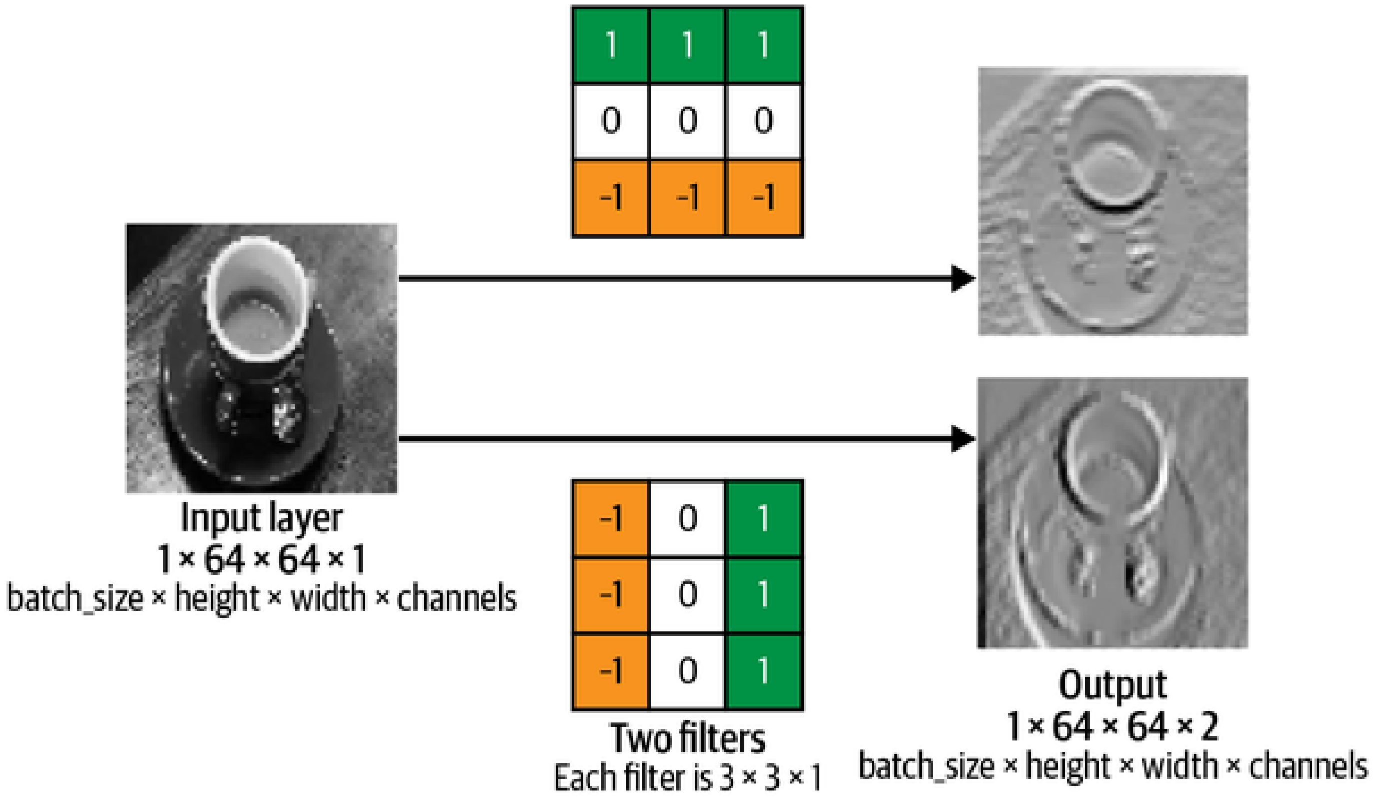


Processing Image with Convolution



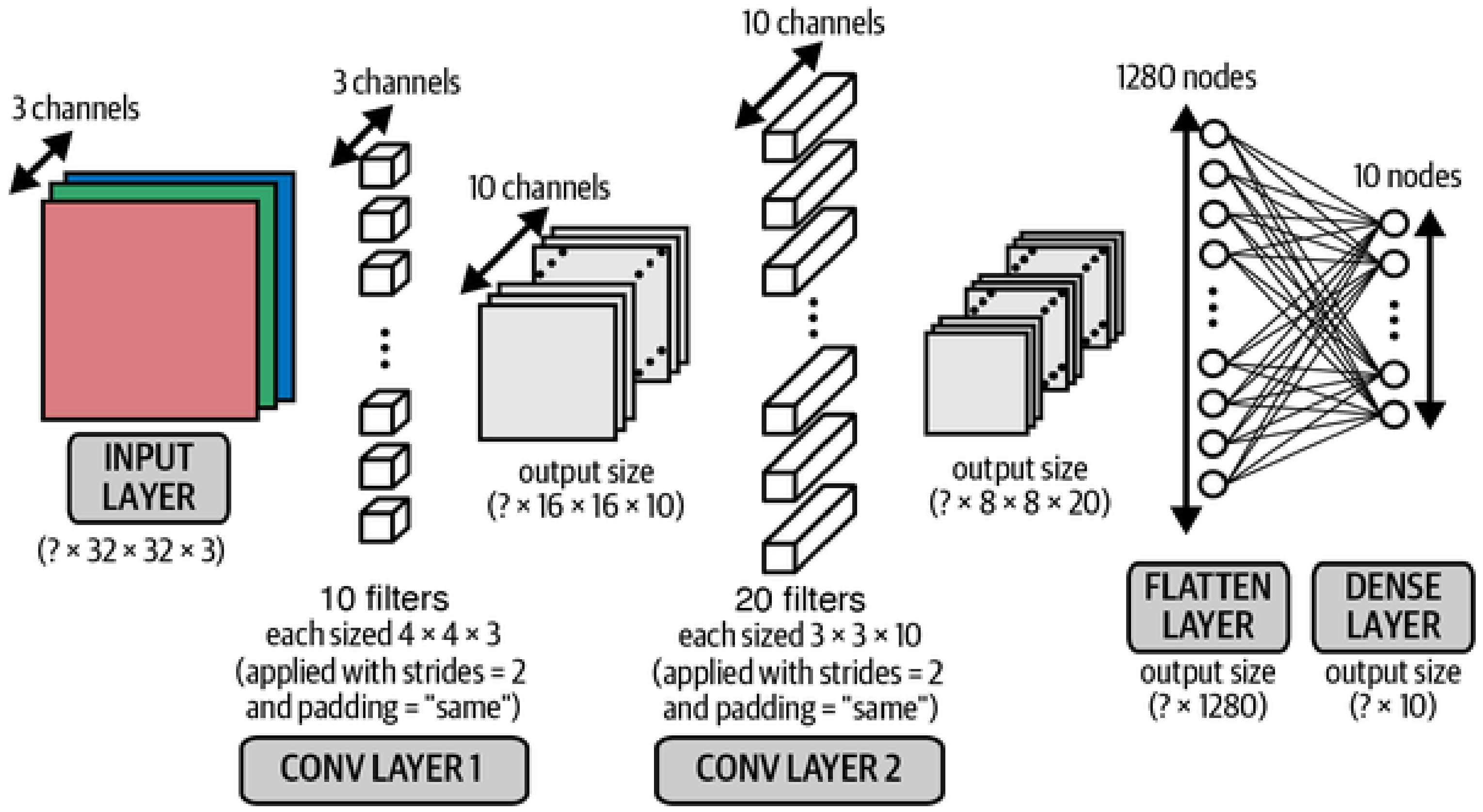
- Passing a 3x3 kernel over a 5x5 image
- Using padding to preserve size of representation

Edge Detection

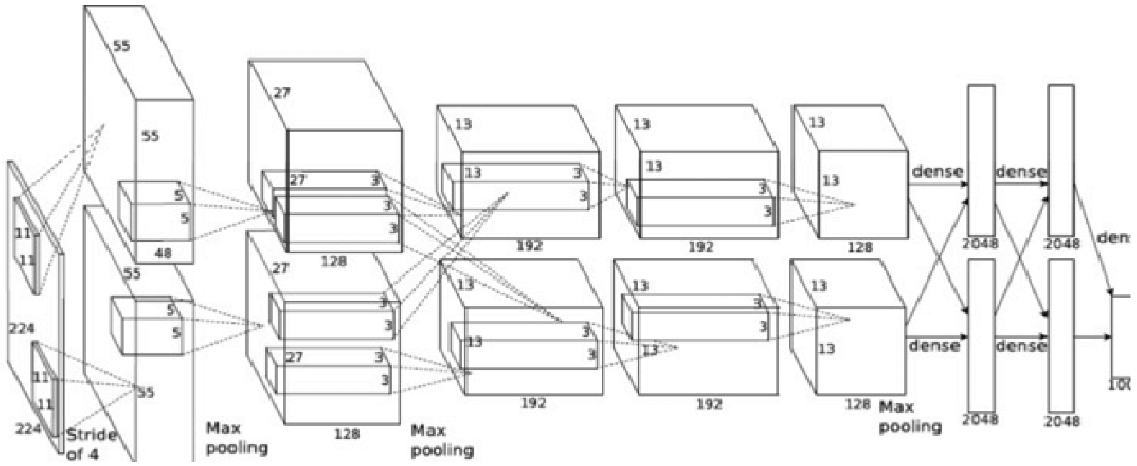


- Detecting large local color shifts: edges in image
- Two kernels: 2-dimensional vector for each point

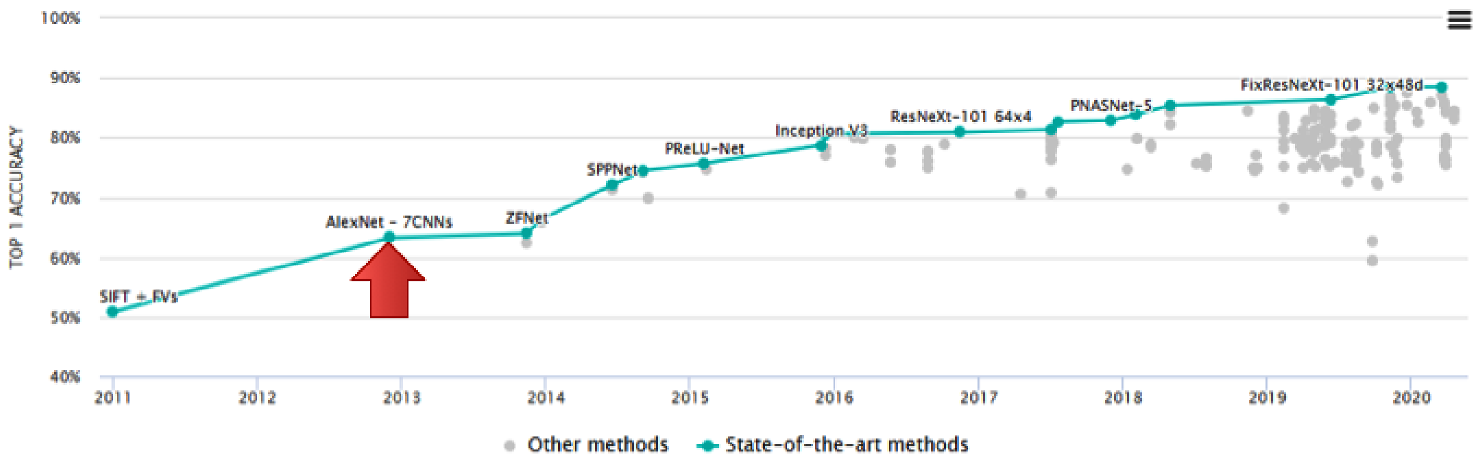
Convolutional Neural Network



ImageNet: 2012

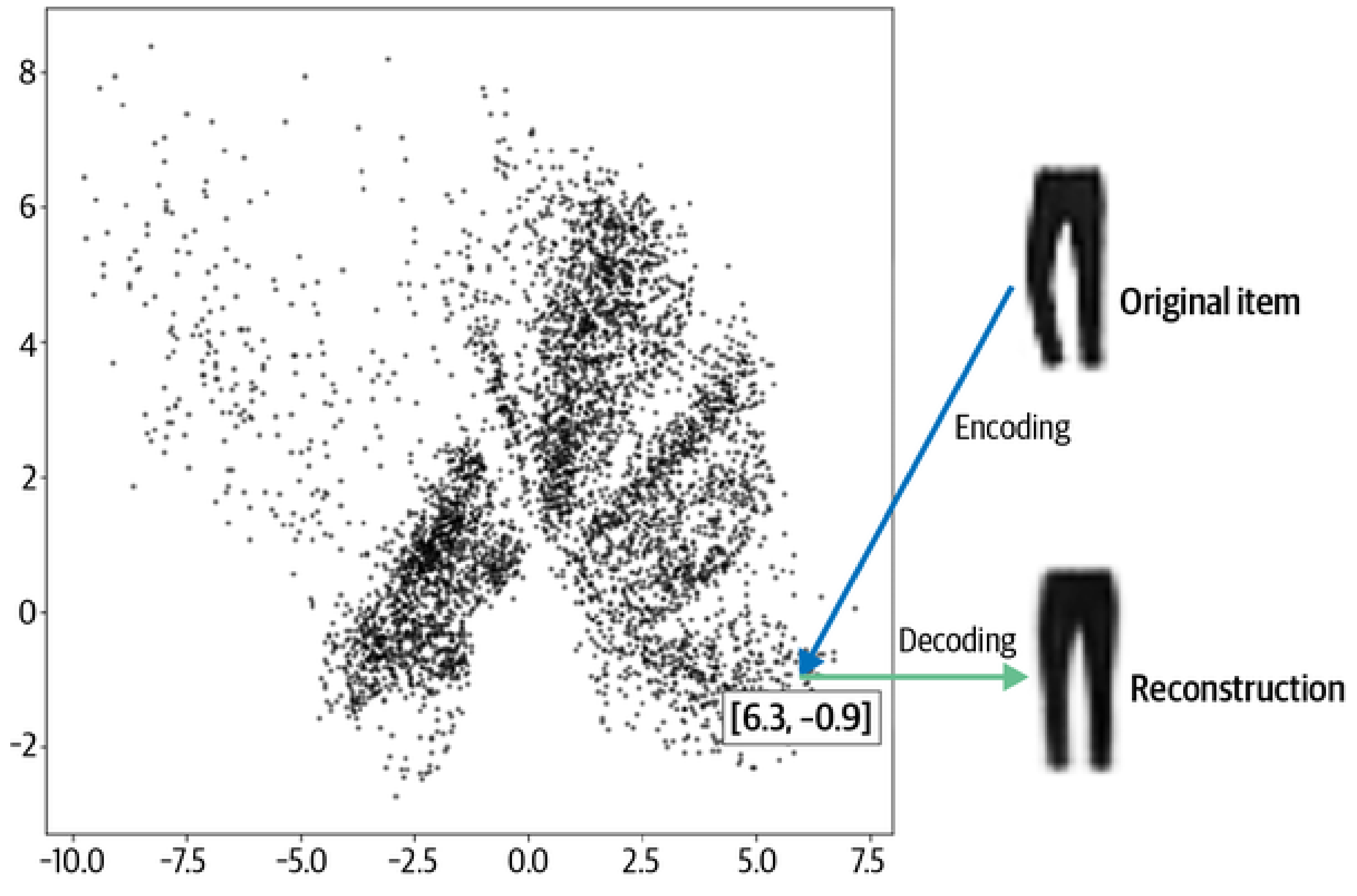


- ImageNet: Large scale competition for image classification (1.2 million labeled training examples, 1000 categories)
- Breakthrough for deep learning in 2012: large gains with CNN (AlexNet)

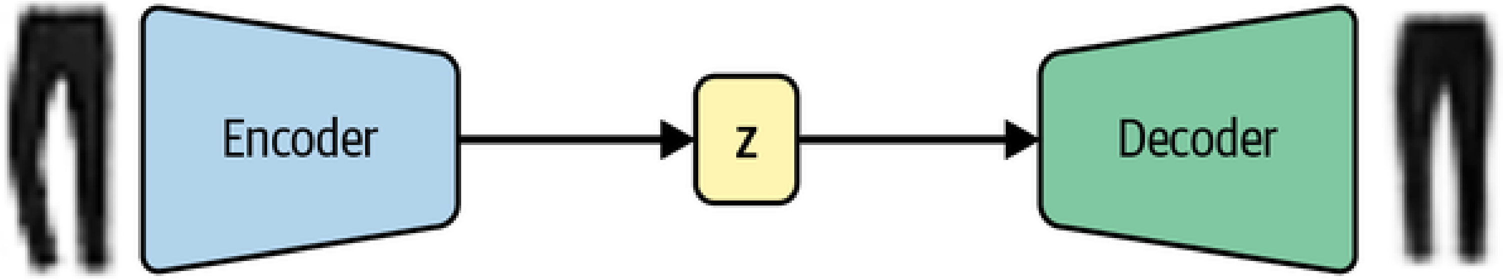


autoencoders

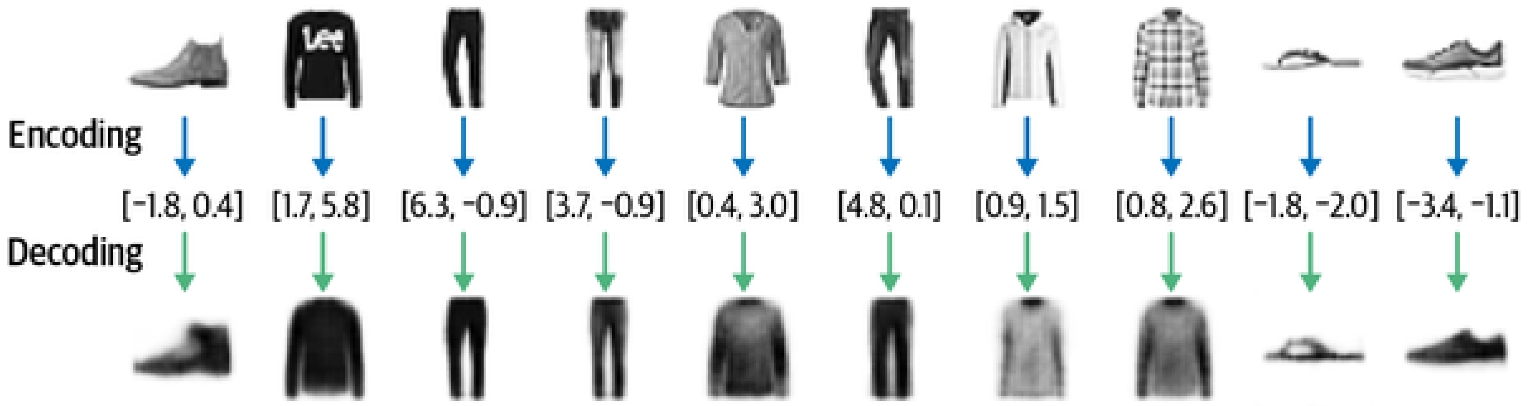
Autoencoders



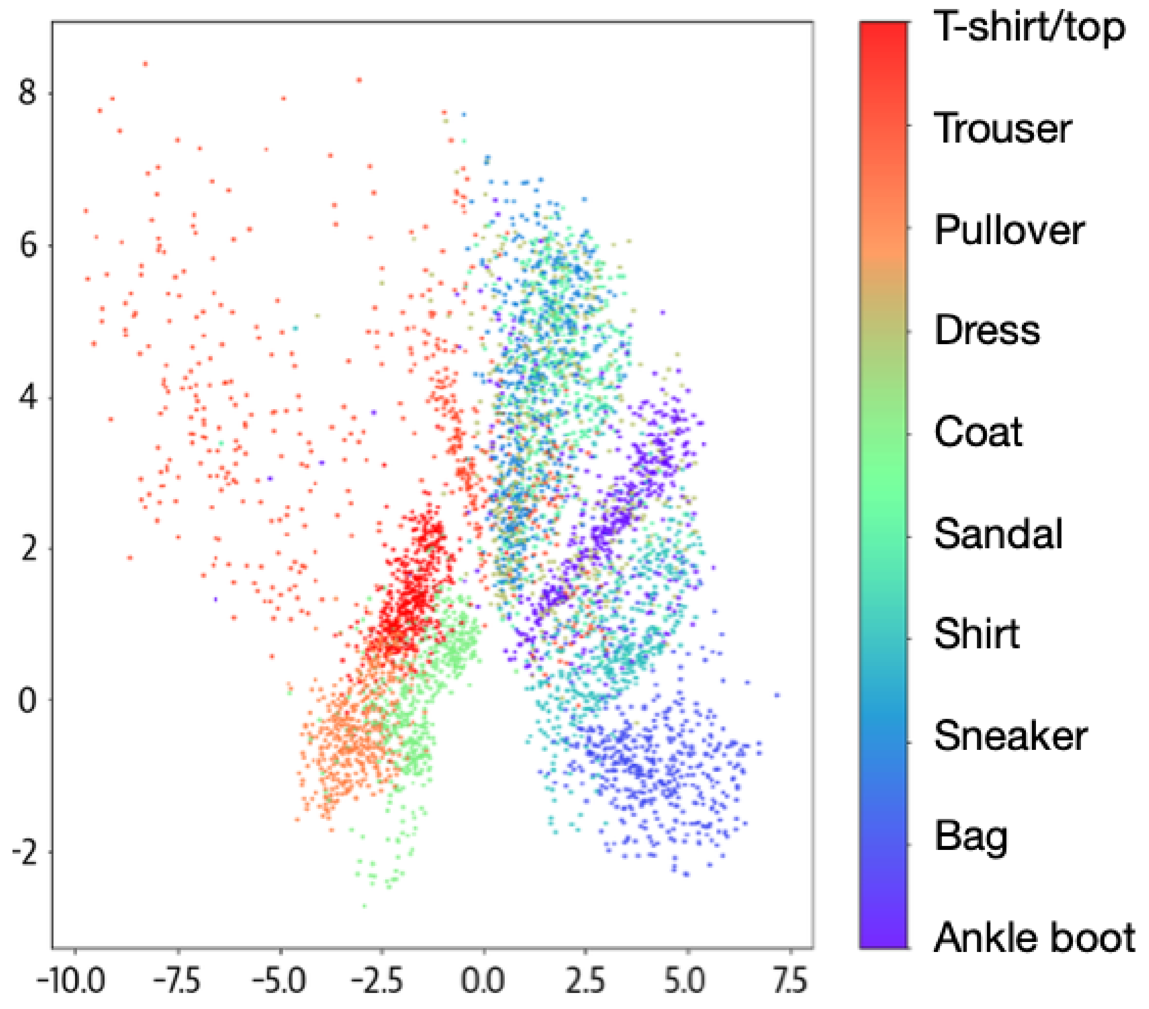
Encoder-Decoder Model



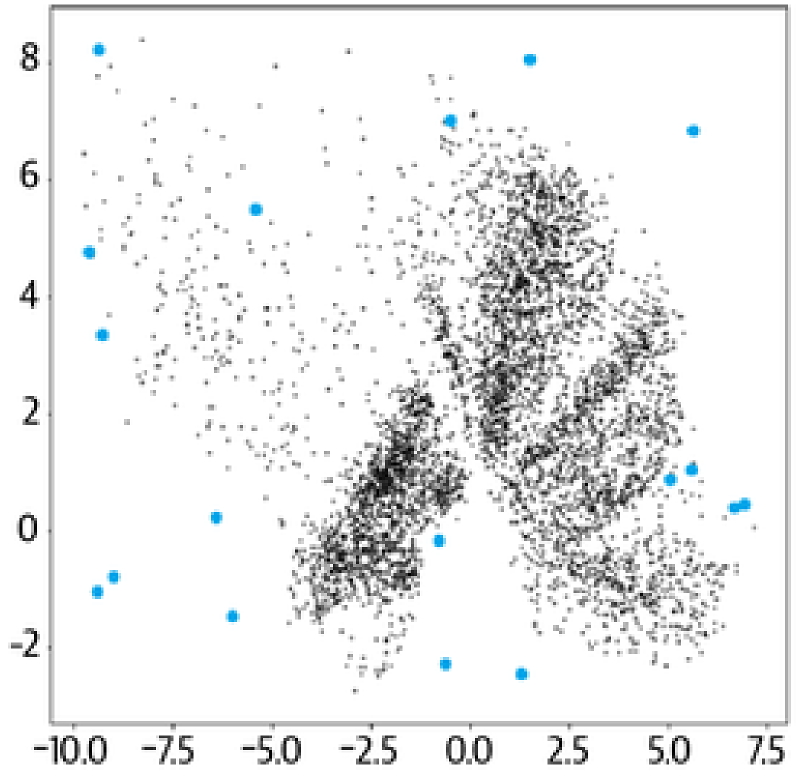
Example: Image to 2-dimensional vector

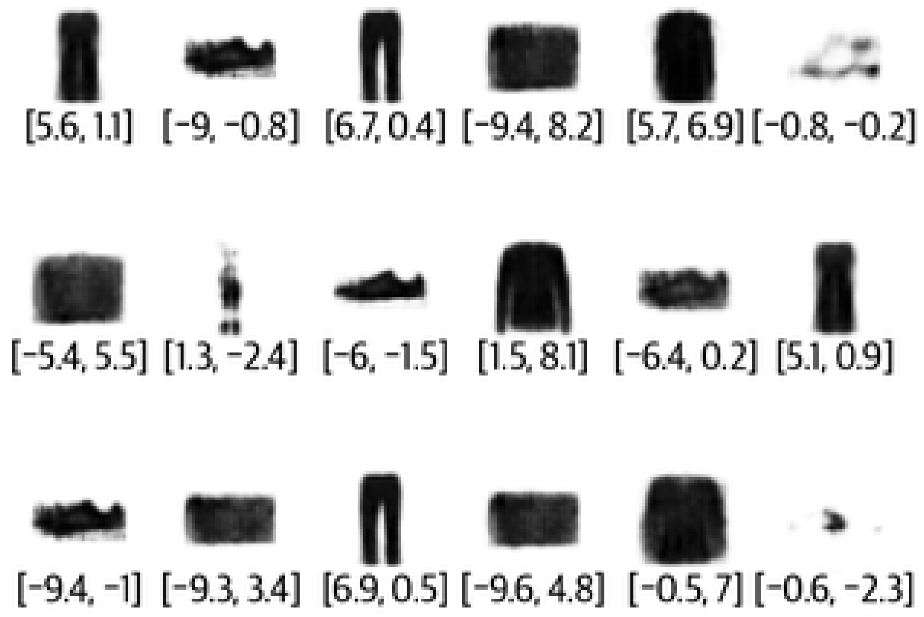


Visualization of Embeddings Space

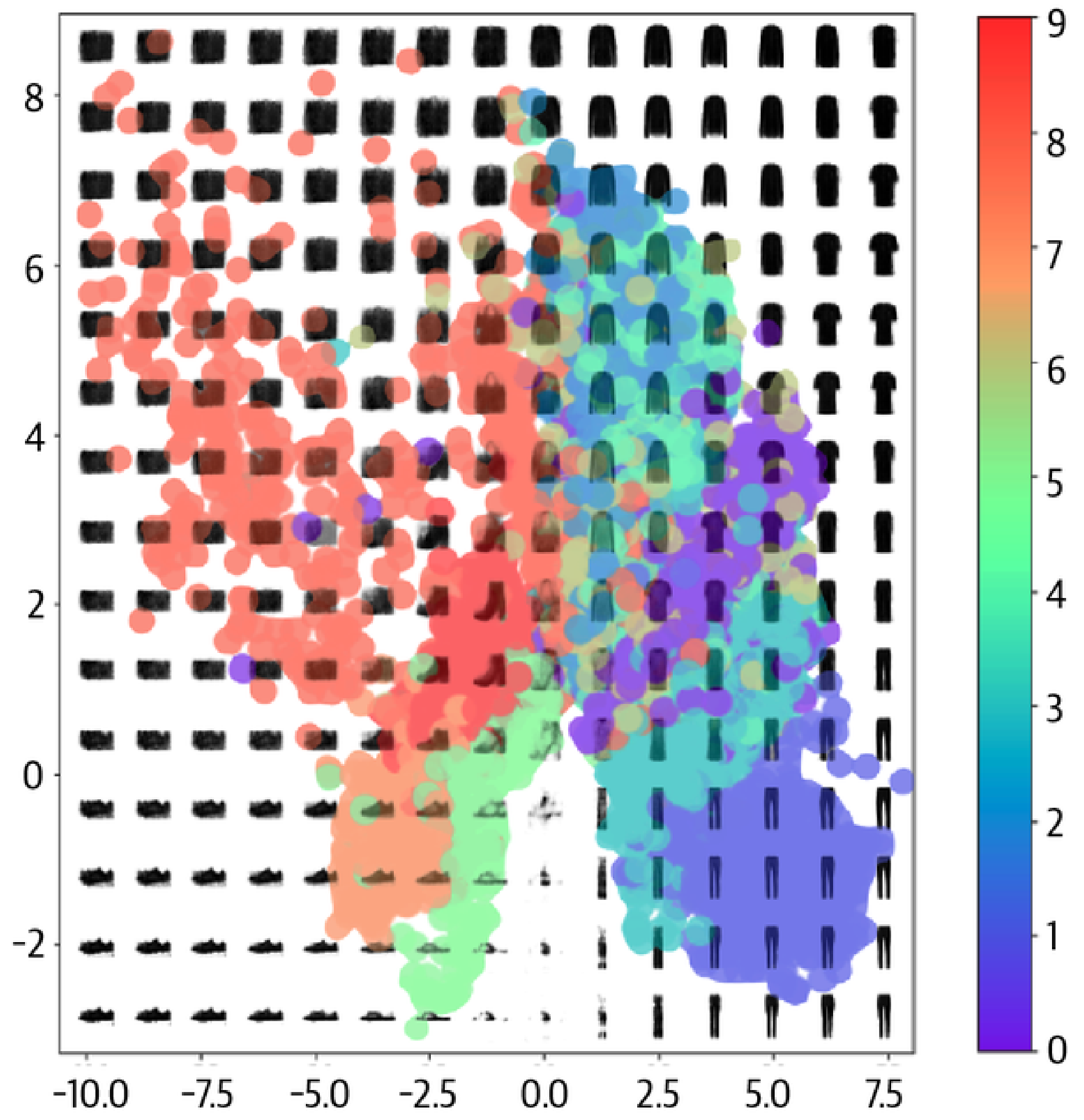


Generating Novel Items of Clothing

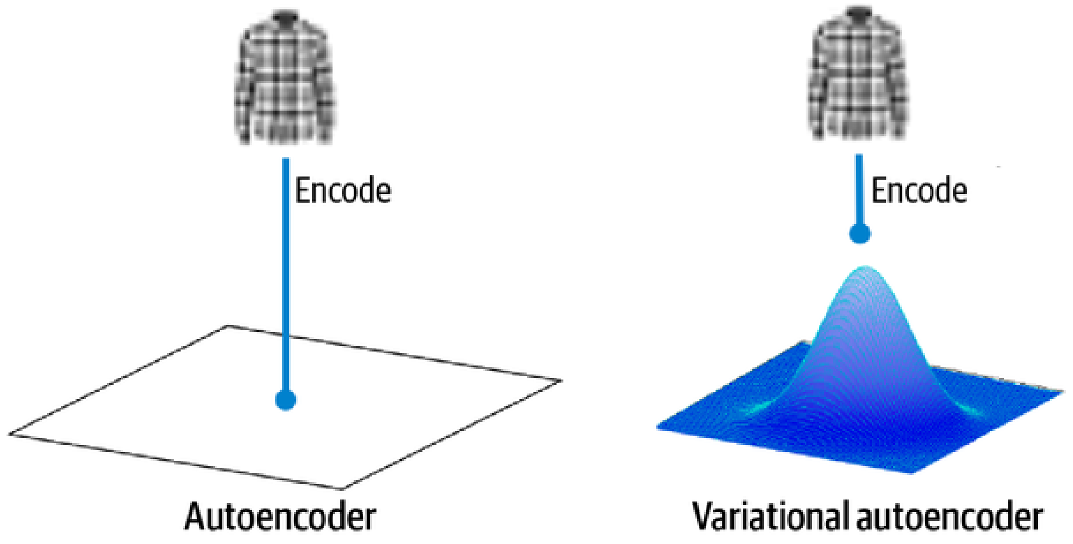




Generations from Full Embeddings Space



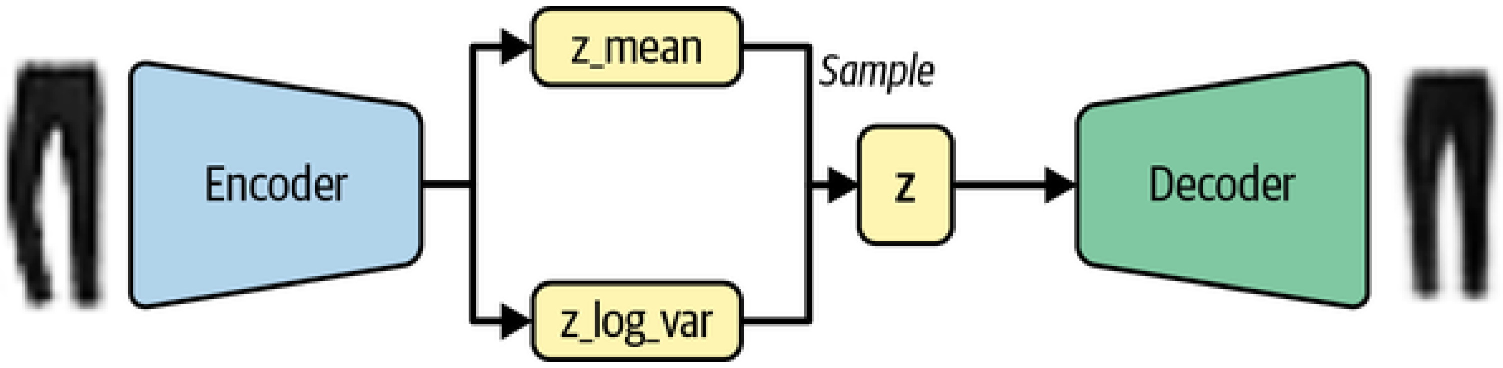
Variational Autoencoder



- Encoder predicts a normal distribution, specifically the **mean** μ and **variance** σ^2

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

Variational Autoencoder



- Encoder (as before)
- Predicts mean `z_mean` and variance `z_log_var` of the distribution (note: predict 2-dimensional point $\rightarrow$ 2 means and 2 variances)
- We randomly draw a point `z` from the distribution
- Decoder predicts from that randomly drawn point `z` (as before)

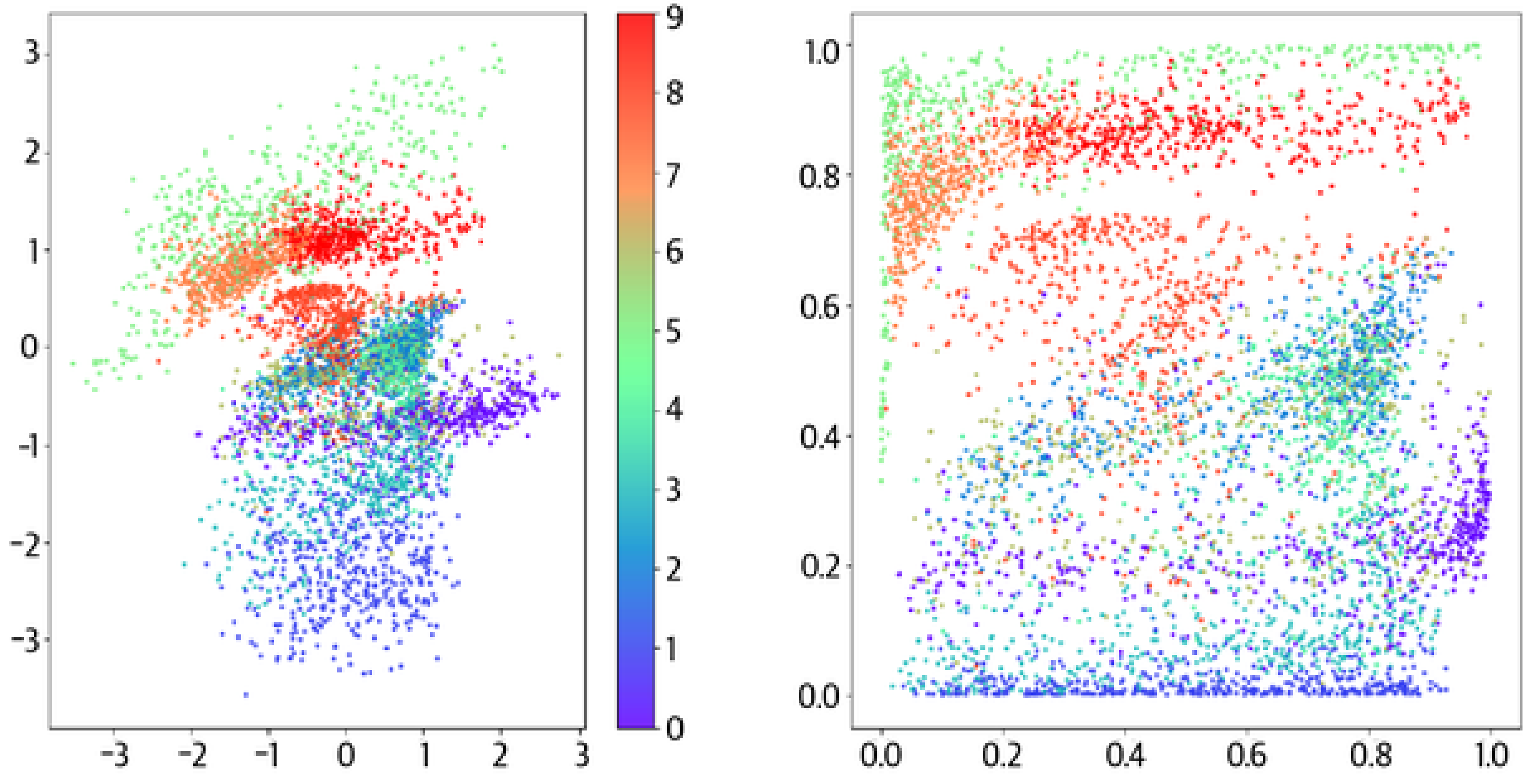
Additional Loss Term for Training

- Add preference that predicted mean and variance are normal distribution with mean 0 and variance 1: $N(0,1)$
- Computed as KL-divergence between predicted values μ and σ^2

$$\mathcal{D}_{KL}\left[N(\mu,\sigma) \| N(0,1)\right] = -\frac{1}{2}\sum\left(1 + \log\left(\sigma^2\right) - \mu^2 - \sigma^2\right)$$

What is it Good For?

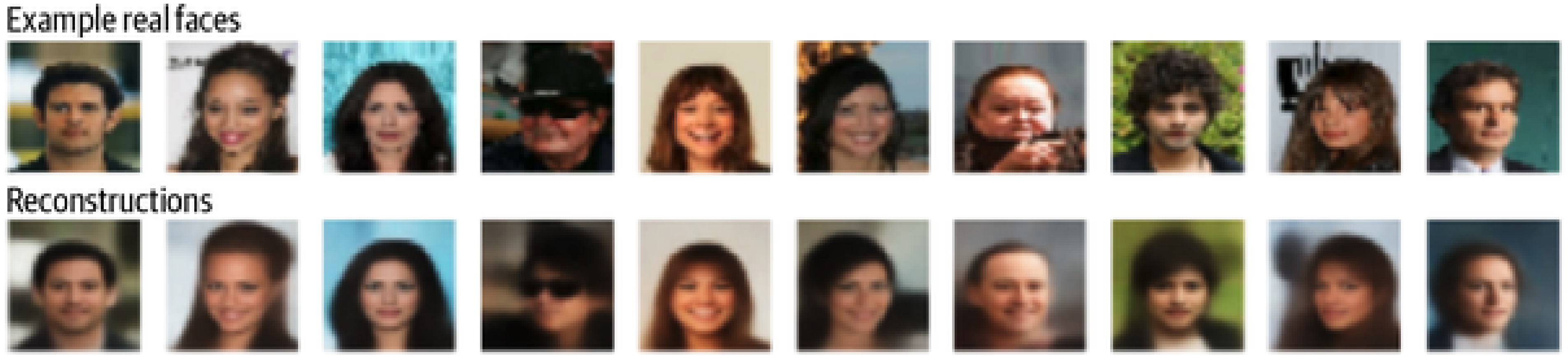
- General principle: adding noise is good (here: random sampling of point z)
- Better use of embedding space (bias towards center)



Left: z_mean, Right: p-values

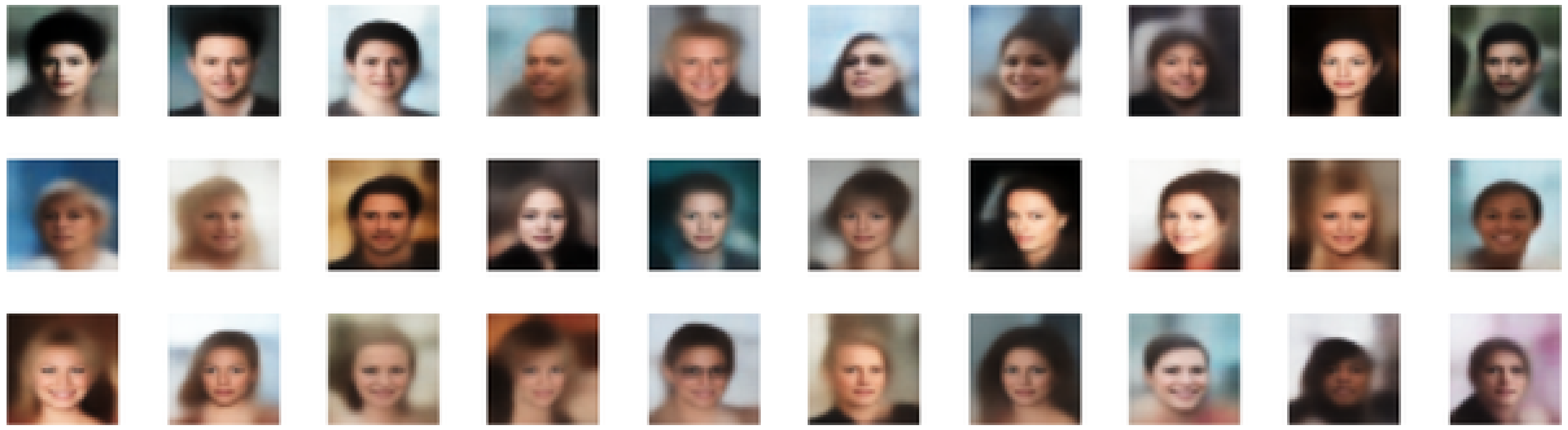
Variational Autoencoder for Images of Faces

- CelebA dataset of over 200,000 color images of celebrity faces
- We need a larger model (multiple convolutional layers, larger embedding sizes)

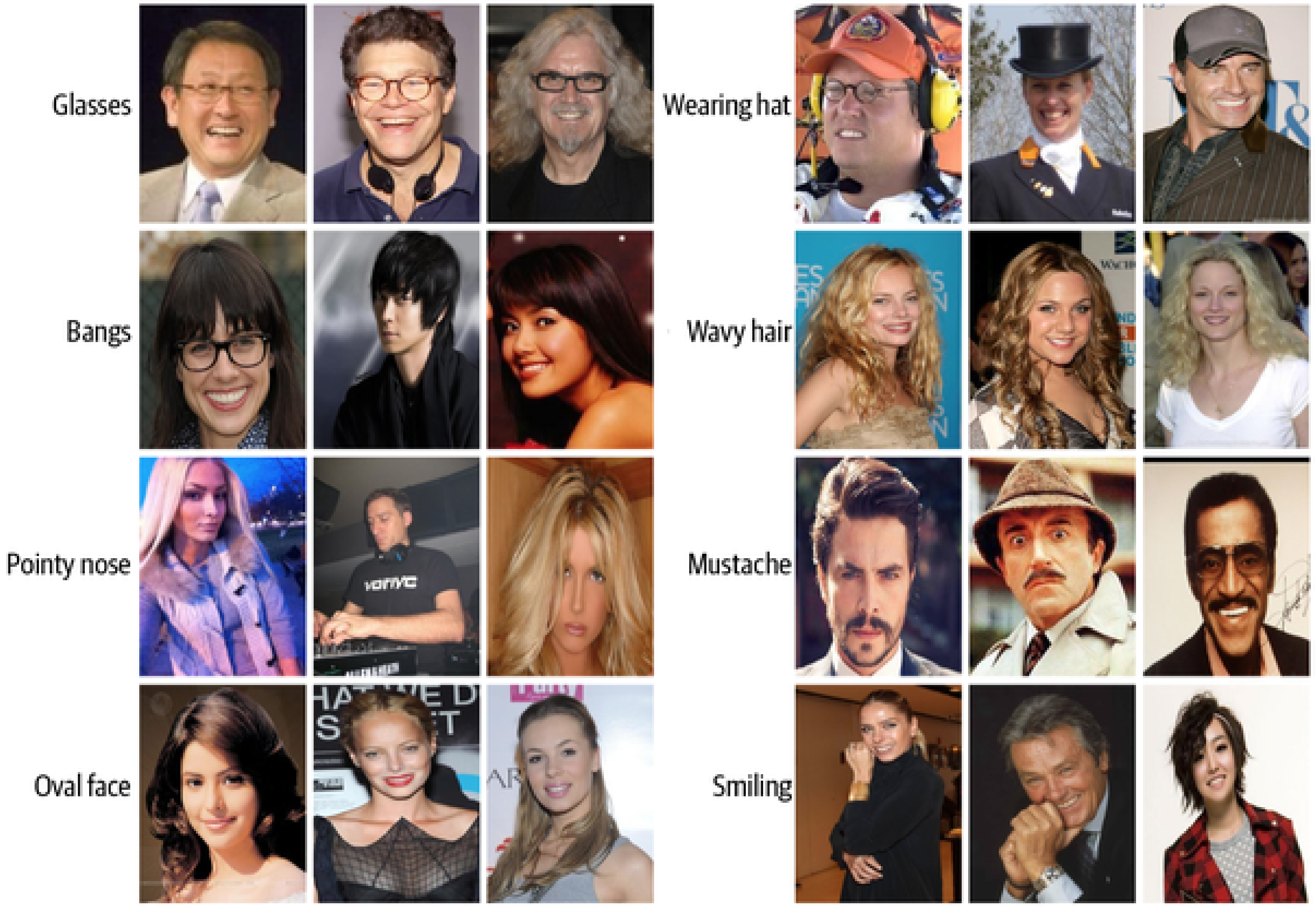


Generation of Faces

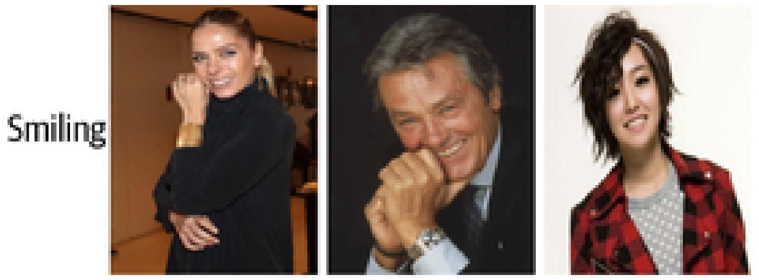
- Sample points (from a standard multivariate normal distribution)
- Decode the sampled points
- Plot the images



CelebA Dataset Labeled with Features



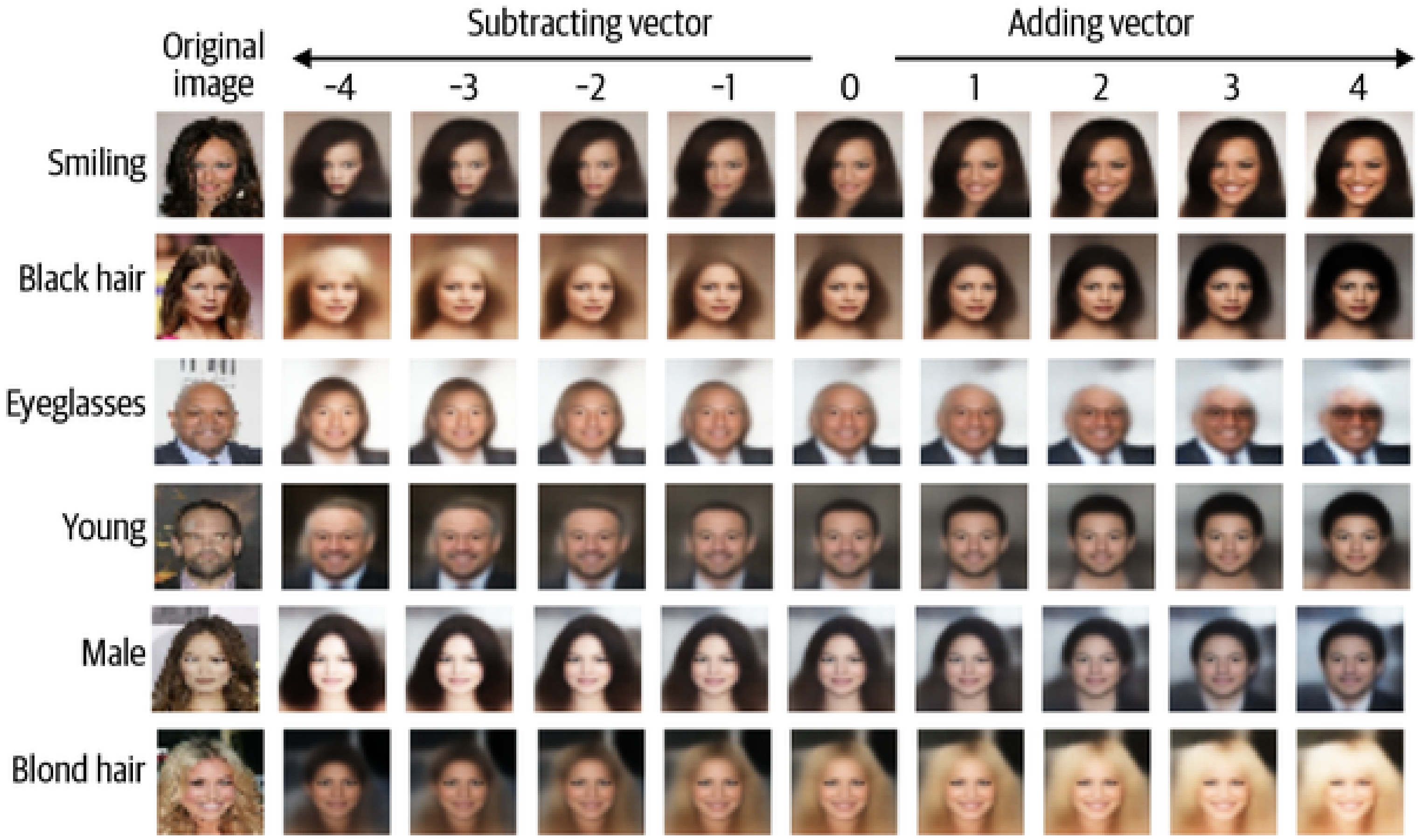
Vectors for Features



- We want to obtain a vector for one feature, e.g. `Smiling`
- Recall: encoder predicts vector representation for each image
- Vector that points in the direction of `Smiling`
  - take average of all vectors for images labelled `Smiling`: a_{smiling}
  - take average of all vectors for images not labelled `Smiling`: $a_{\overline{\text{smiling}}}$
  - subtract the two vectors: $v_{\text{smiling}} = a_{\text{smiling}} - a_{\overline{\text{smiling}}}$

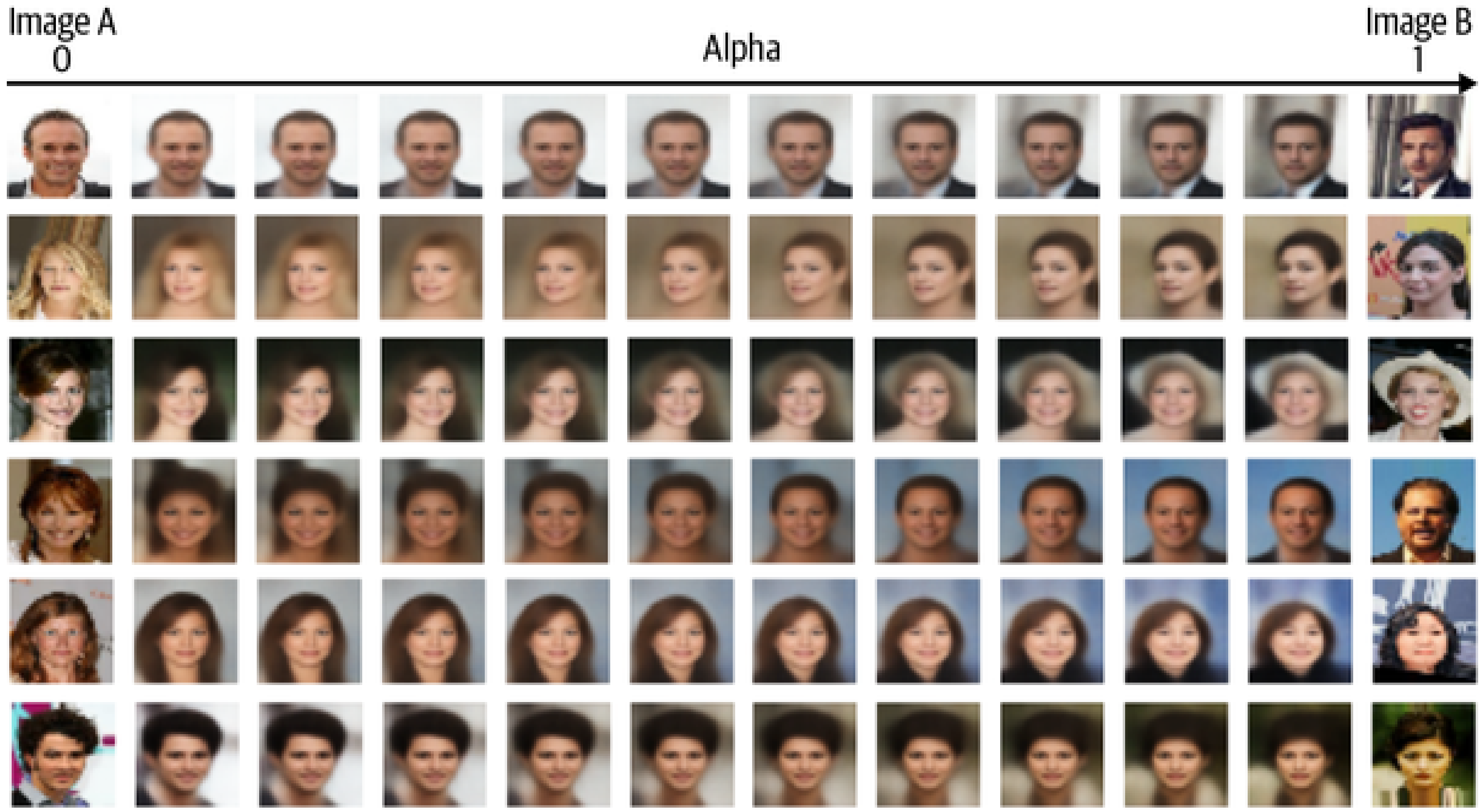
Generation with Feature Vector

- Randomly sample point z (or use representation for existing image)
- Subtract and feature vector v_{smiling} with varying factor, generate



Combine Images

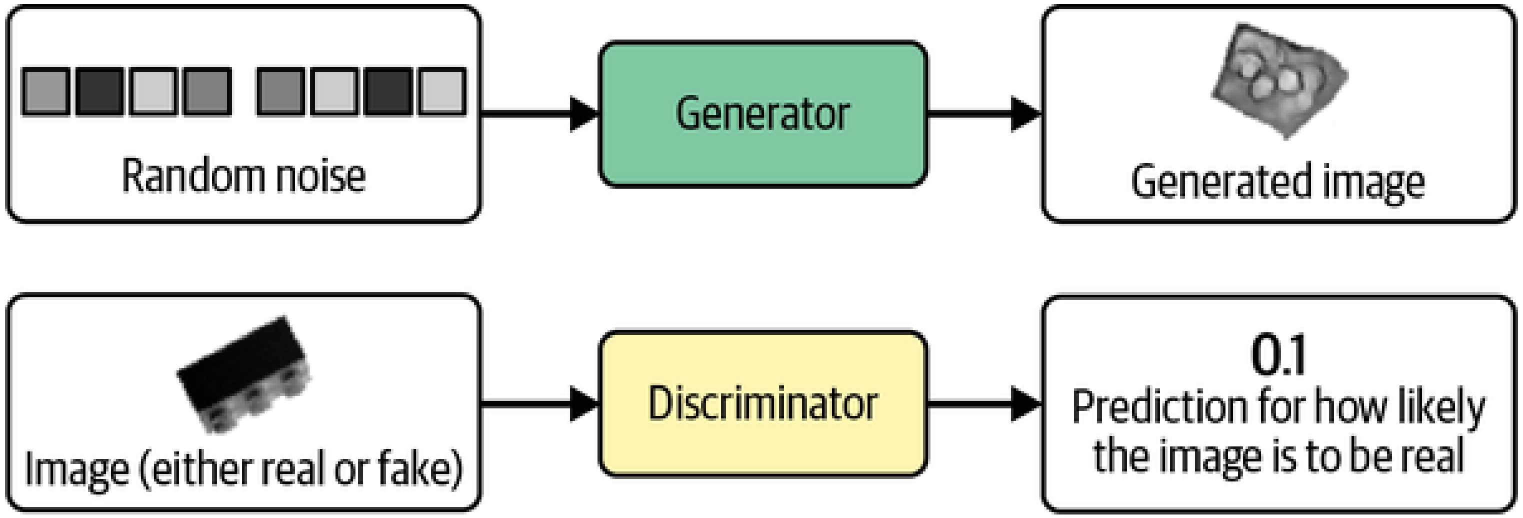
- Take representations of two images: z_A and z_B
- Combine them with weight $\alpha \in [0, 1]$ into new vector $z = \alpha z_A + (1 - \alpha) z_B$



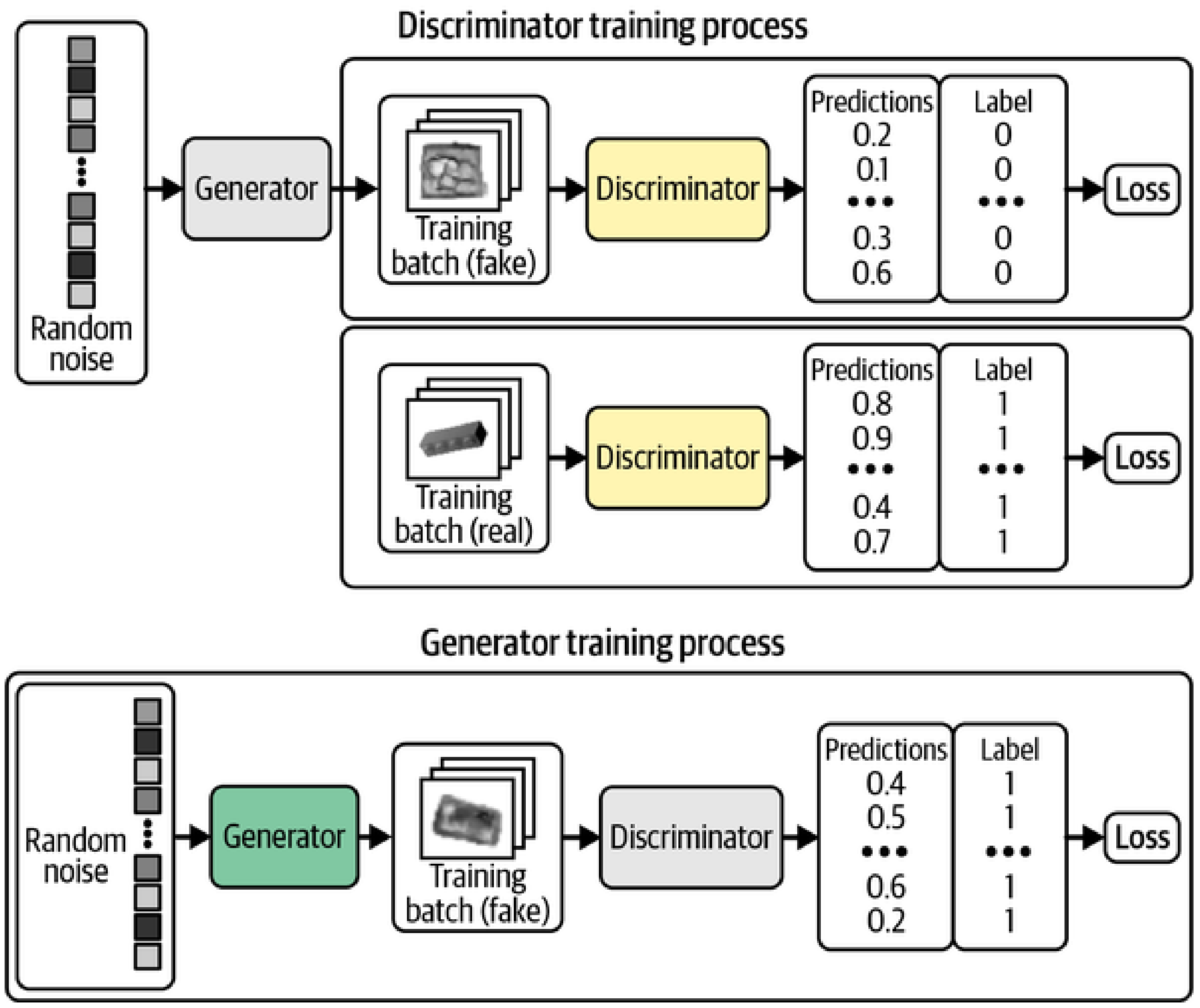
generative adversarial training

Discriminator

- The story so far: we can generate novel images
- New task: detect if image is generated or real


Random noise
Generator
Generated image
Image (either real or fake)
Discriminator
0.1
Prediction for how likely the image is to be real

Training


Discriminator training process
Random noise
Generator
Training batch (fake)
Discriminator
Predictions 0.2 0.1 ••• 0.3 0.6
Label 0 0 ••• 0 0
Loss
Training batch (real)
Discriminator
Predictions 0.8 0.9 ••• 0.4 0.7
Label 1 1 ••• 1 1
Loss
Generator training process
Random noise
Generator
Training batch (fake)
Discriminator
Predictions 0.4 0.5 ••• 0.6 0.2
Label 1 1 ••• 1 1
Loss

Training Stages

- Train initial generator
- Iterate
  - train discriminator
  - train generator with additional discriminator loss
- Note: when training generator, leave discriminator parameters fixed
  (and vice versa)

Common Problems

- Discriminator overpowers generator
- Generator overpowers discriminator
  (for instance mode collapse: generates unique image that fools discriminator)
- Training loss of generator is not informative
  (mainly battles discriminator and not fitting the training data)
- Many hyperparameters

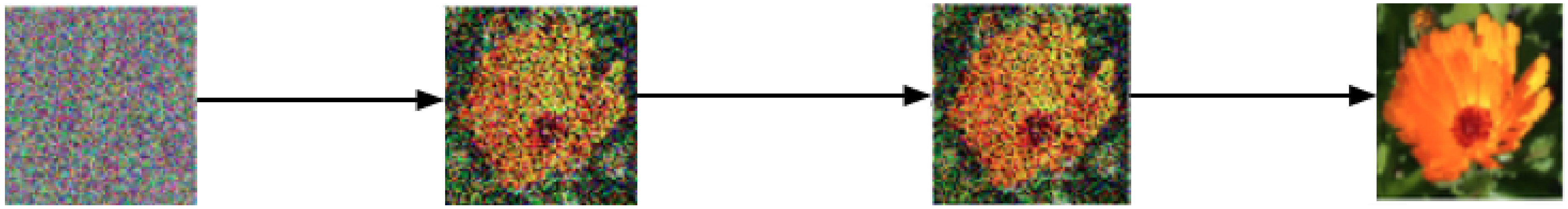
Extensions

- Wasserstein GAN
- Lipschitz Constraint
- Gradient Penalty Loss
- Conditional GAN

diffusion models

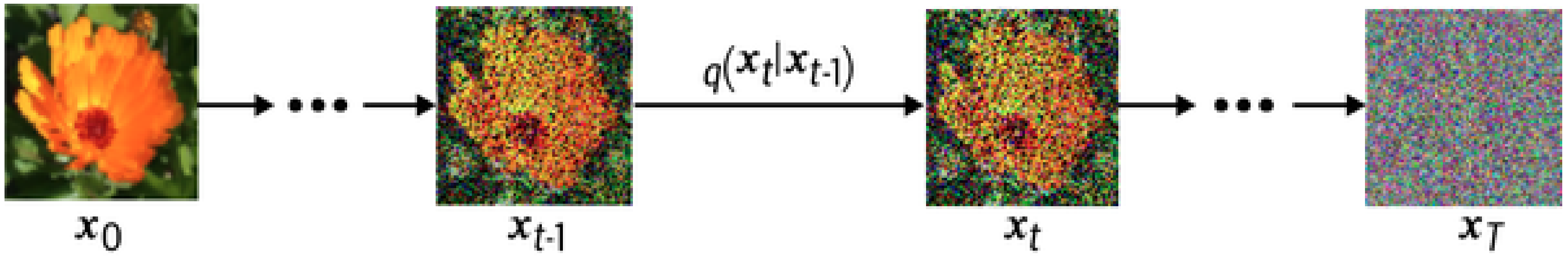
Diffusion Models

- Key idea
  - create noise image
  - generate image from noise
  - repeat this for, say, 20 steps



Create Training Data

- Training data for this process can be creating by adding noise



- This is done in stages, each time adding Gaussian noise ϵ_t
  (mean 0 and unit variance, but then scaling to effective variance β_t below)
$$\boldsymbol{x}_t = \sqrt{1-\beta_t}\,\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\,\epsilon_{t-1}$$

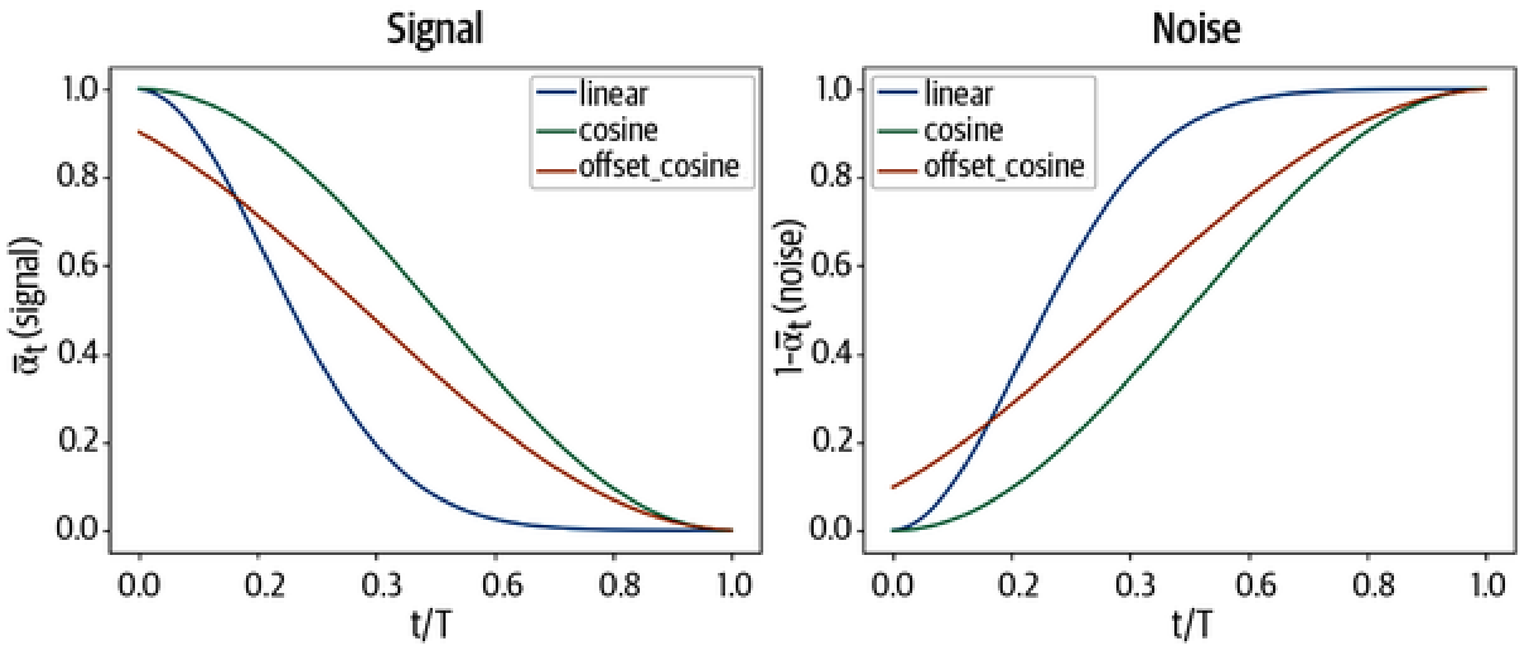
- Or, written as a probability distribution from one image to the next
$$q\left(\boldsymbol{x}_t \,|\, \boldsymbol{x}_{t-1}\right) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{1-\beta_t}\,\boldsymbol{x}_{t-1}, \beta_t\,\mathbf{I}\right)$$

- After a large number of steps, this becomes indistinguishable from a noise image

Diffusion Schemes

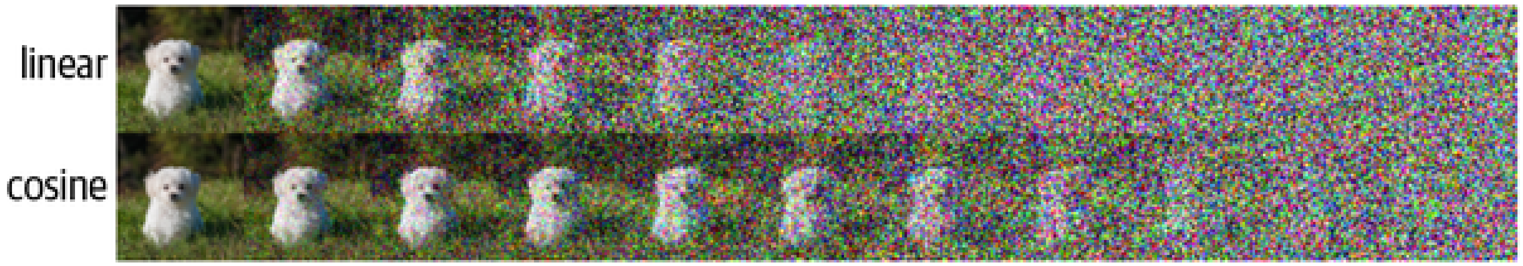
- The variance β_t is changed throughout the process
  - small changes to initial, original image
  - larger changes towards the end to ensure randomness

- Common options: linear, cosine, offset cosine



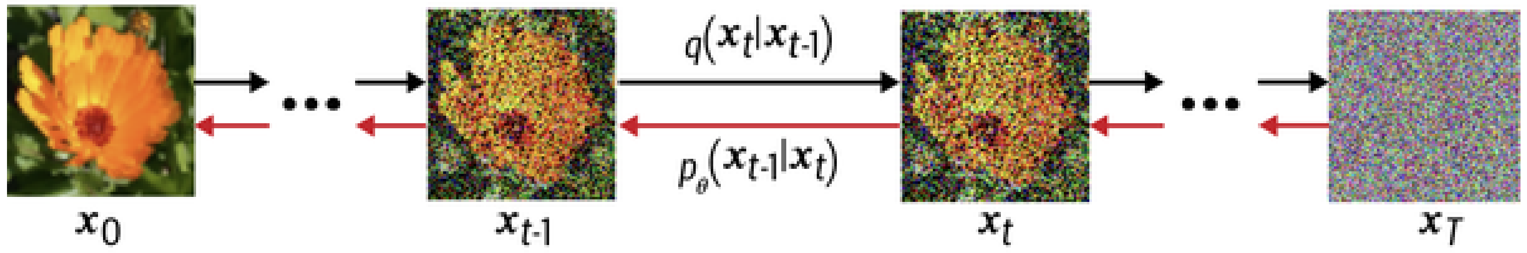
Effect of Diffusion Schemes

- Cosine diffusion scheme makes less changes initially



Goal: Learn How to Reverse this Process

- Learn a model p_θ that maps a noisy image x_t back to a less noisy image x_{t-1}



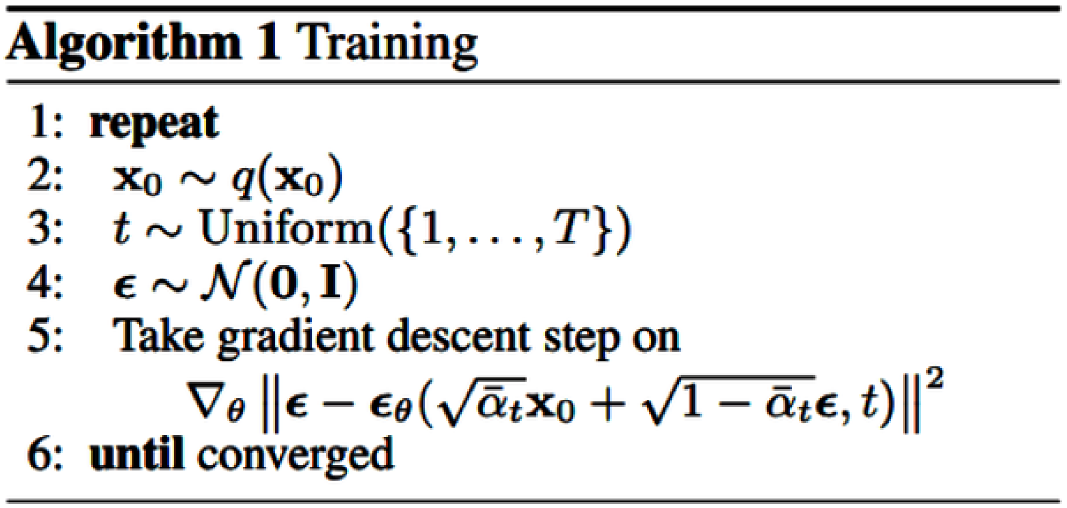
Refinements

- We actually learn a model that maps back x_t to the original image x_o

- The accumated noise can be computed

$$q\left(\boldsymbol{x}_t \mid \boldsymbol{x}_0\right) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{\overline{\alpha}_t}\,\boldsymbol{x}_0, (1 - \overline{\alpha}_t)\,\mathbf{I}\right)$$

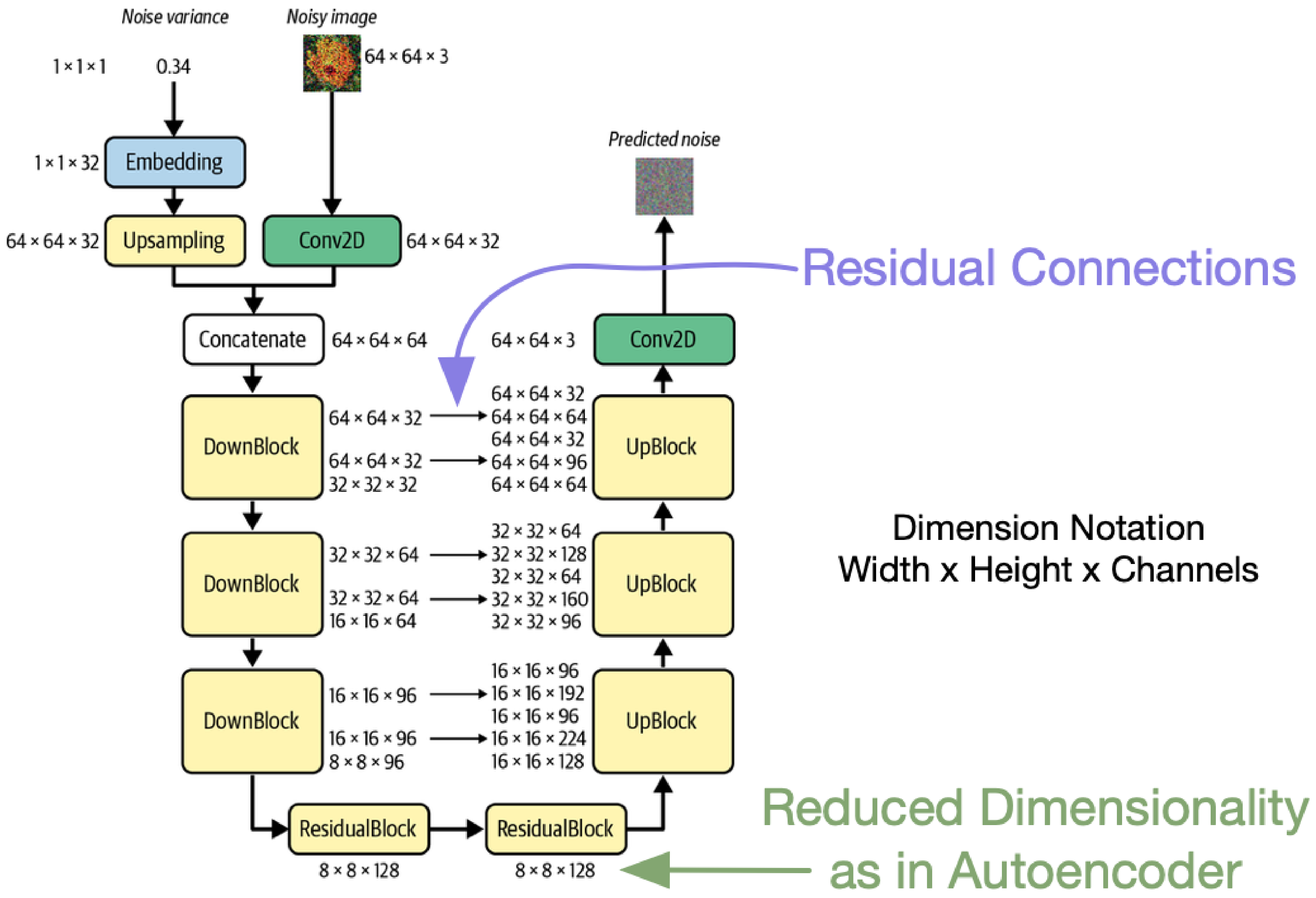
- The model predicts the noise $\epsilon_\theta(x_t)$



Algorithm 1 Training

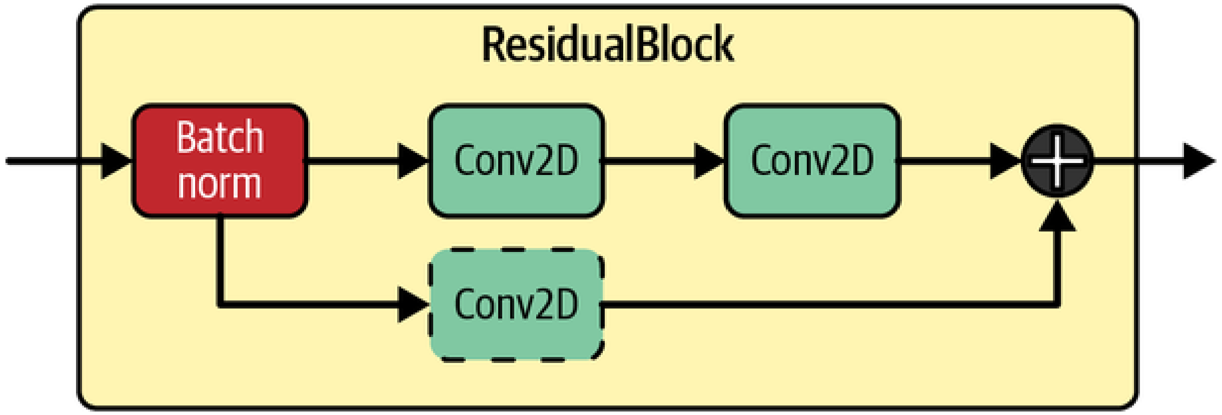
```
1: repeat
2:   $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3:   $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4:   $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5:   Take gradient descent step on
         $\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0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

U-Net Model Overview



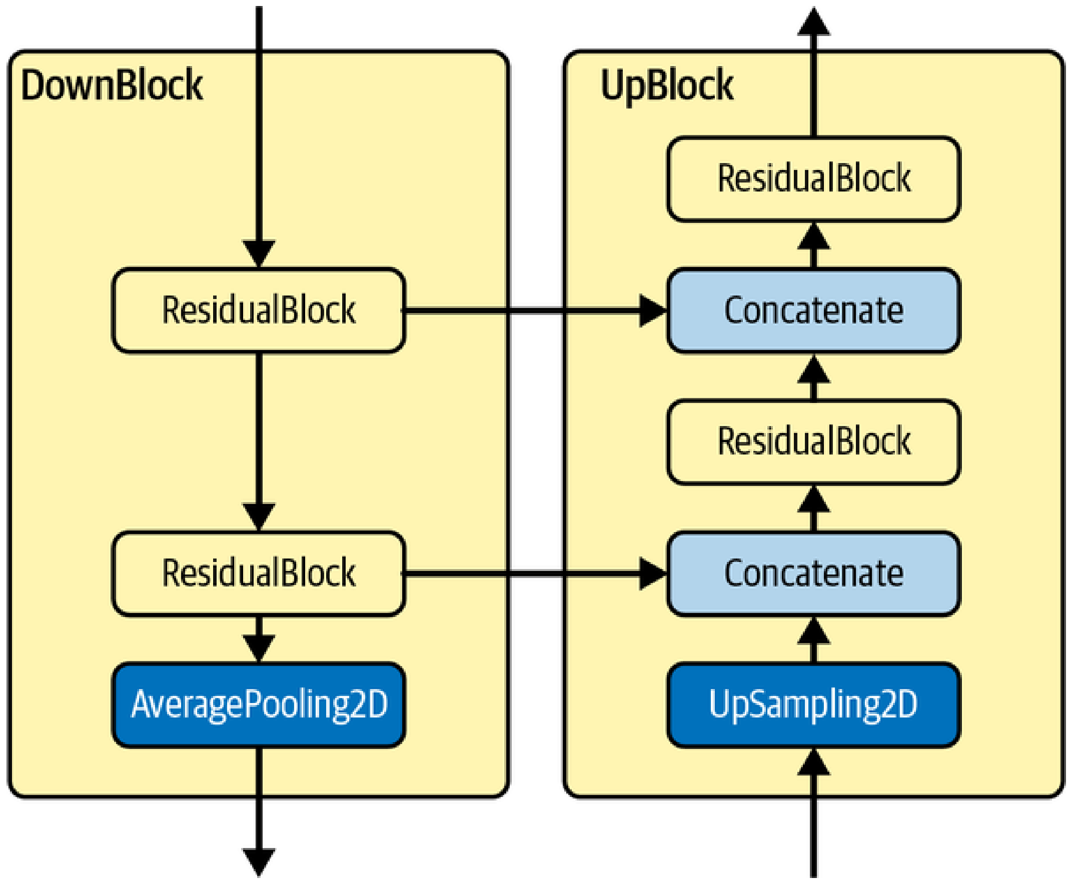
Dimension Notation
Width x Height x Channels

Residual Block

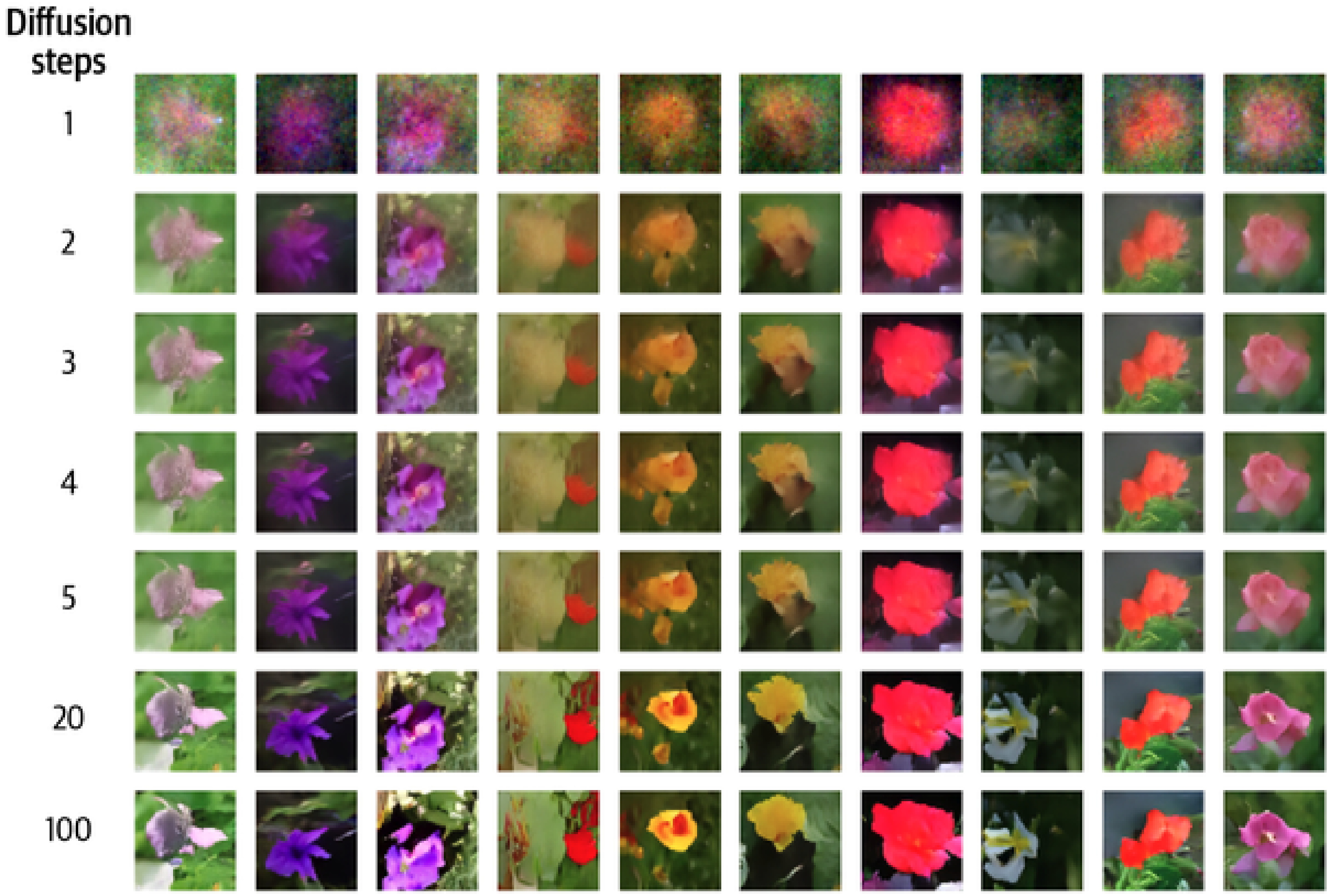


(may add convolution with kernel size 1 to residual connection
to generate tensor with the right number of channels)

DownBlock and UpBlock



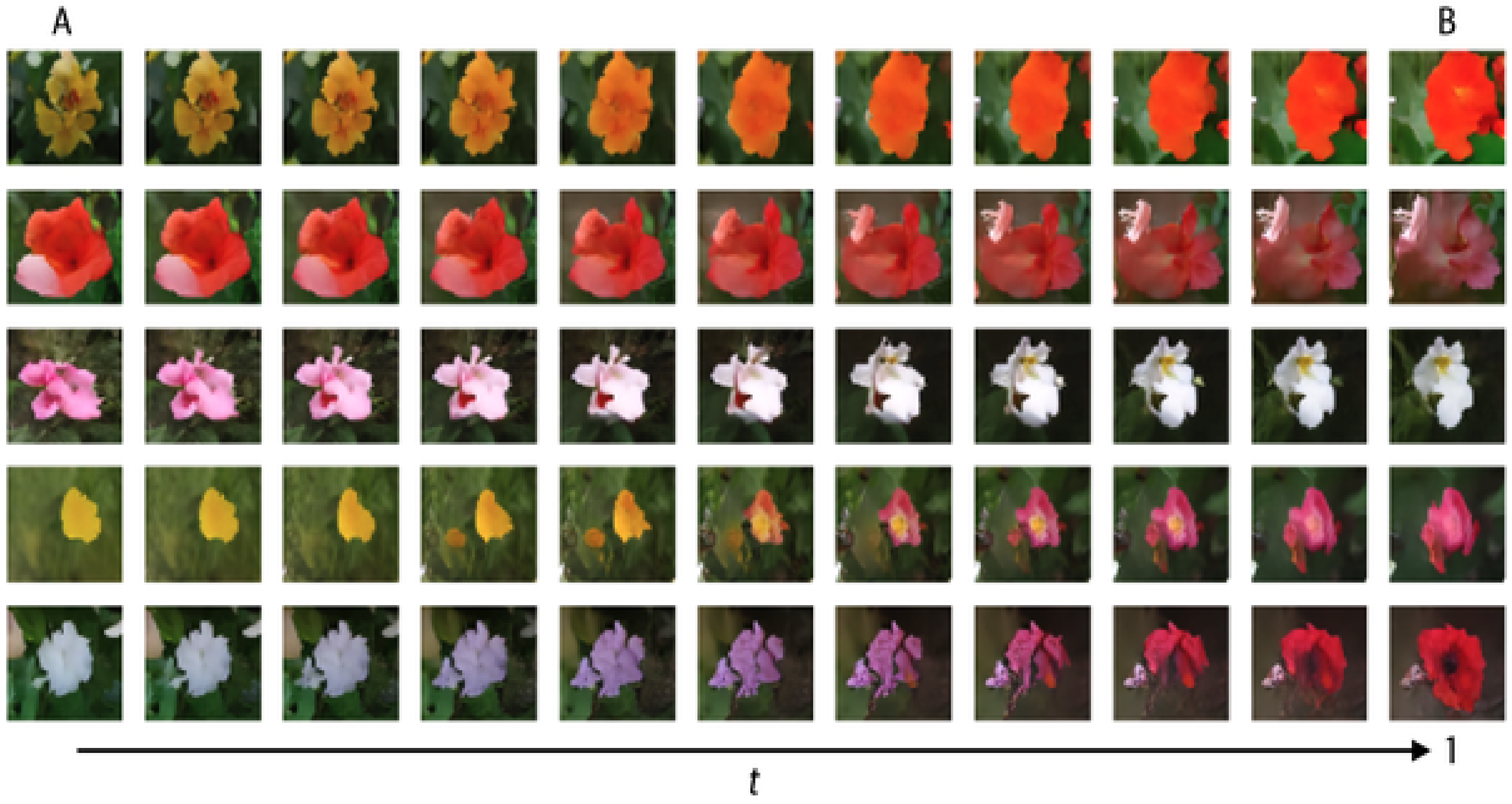
Generation in Multiple Steps



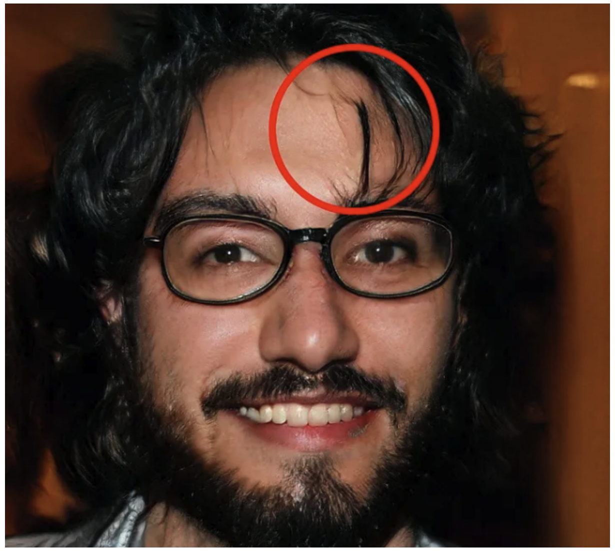
Interpolation

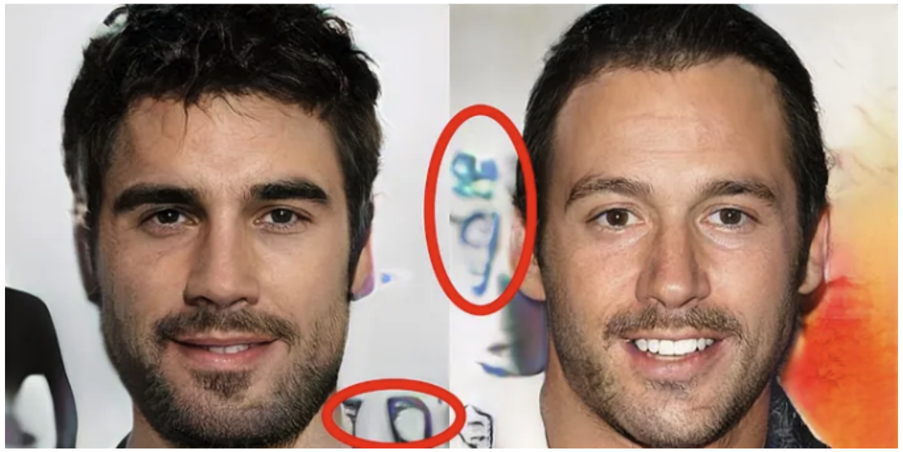
- Generation is deterministic given noise tensor a and b

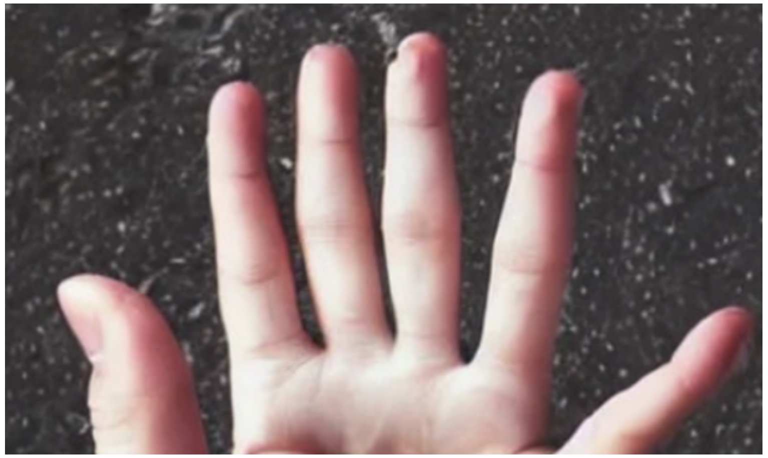
⇒ We can interpolate between different noise images

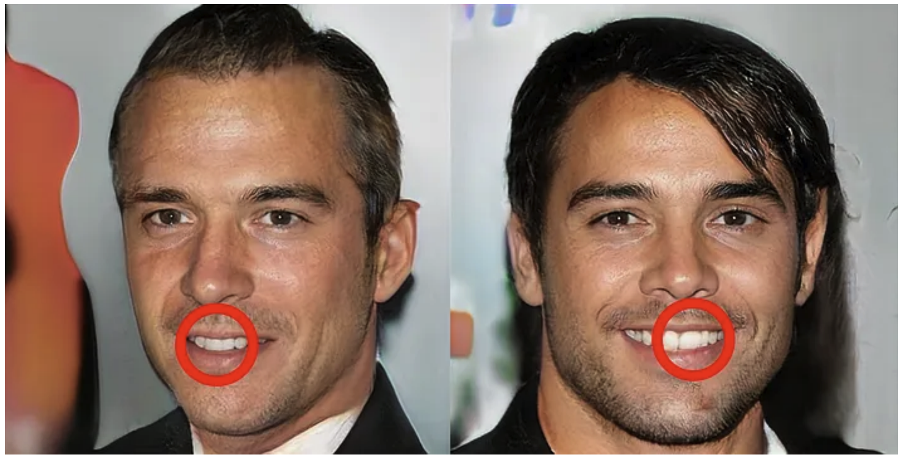


Common Failures



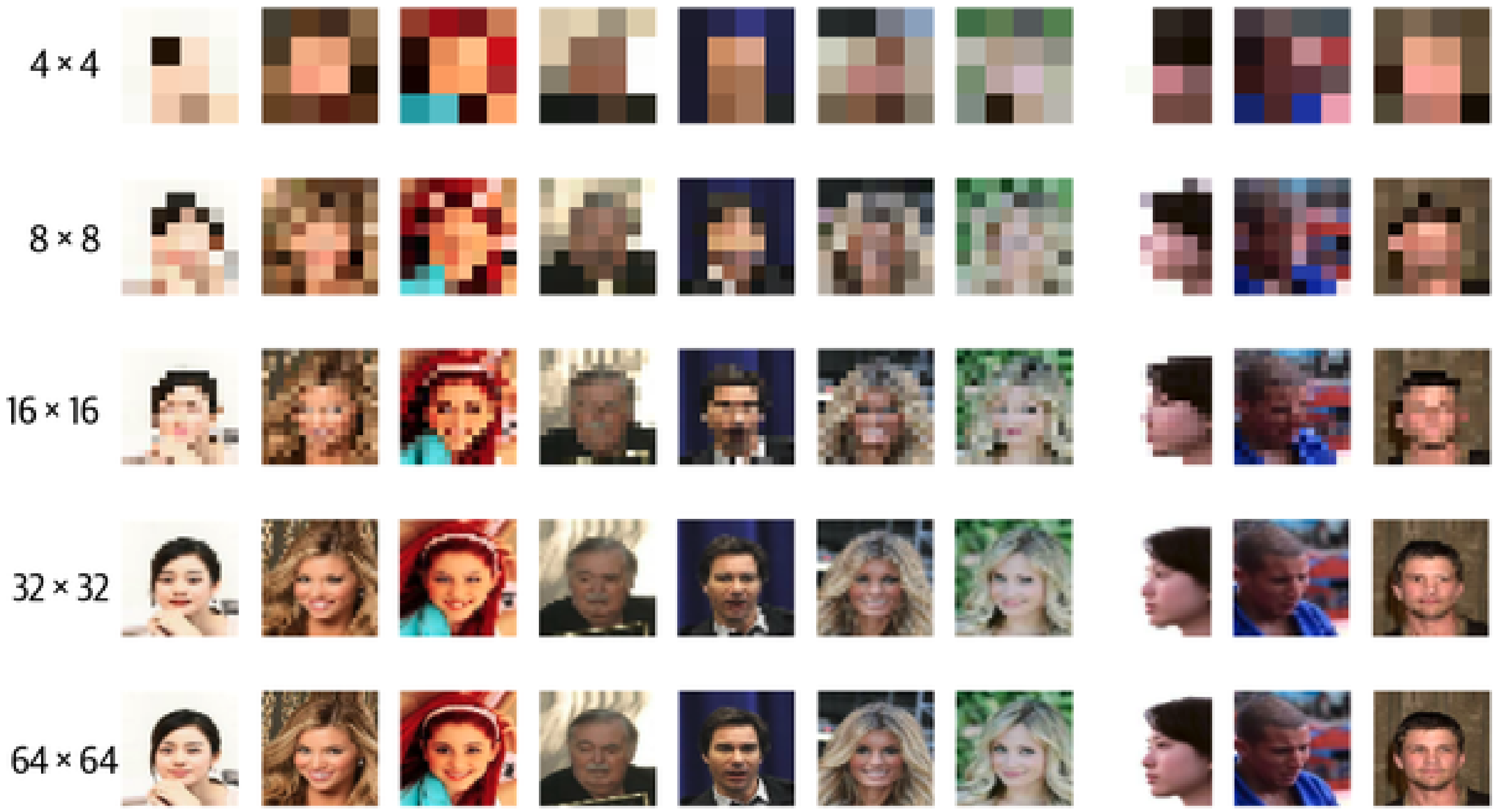






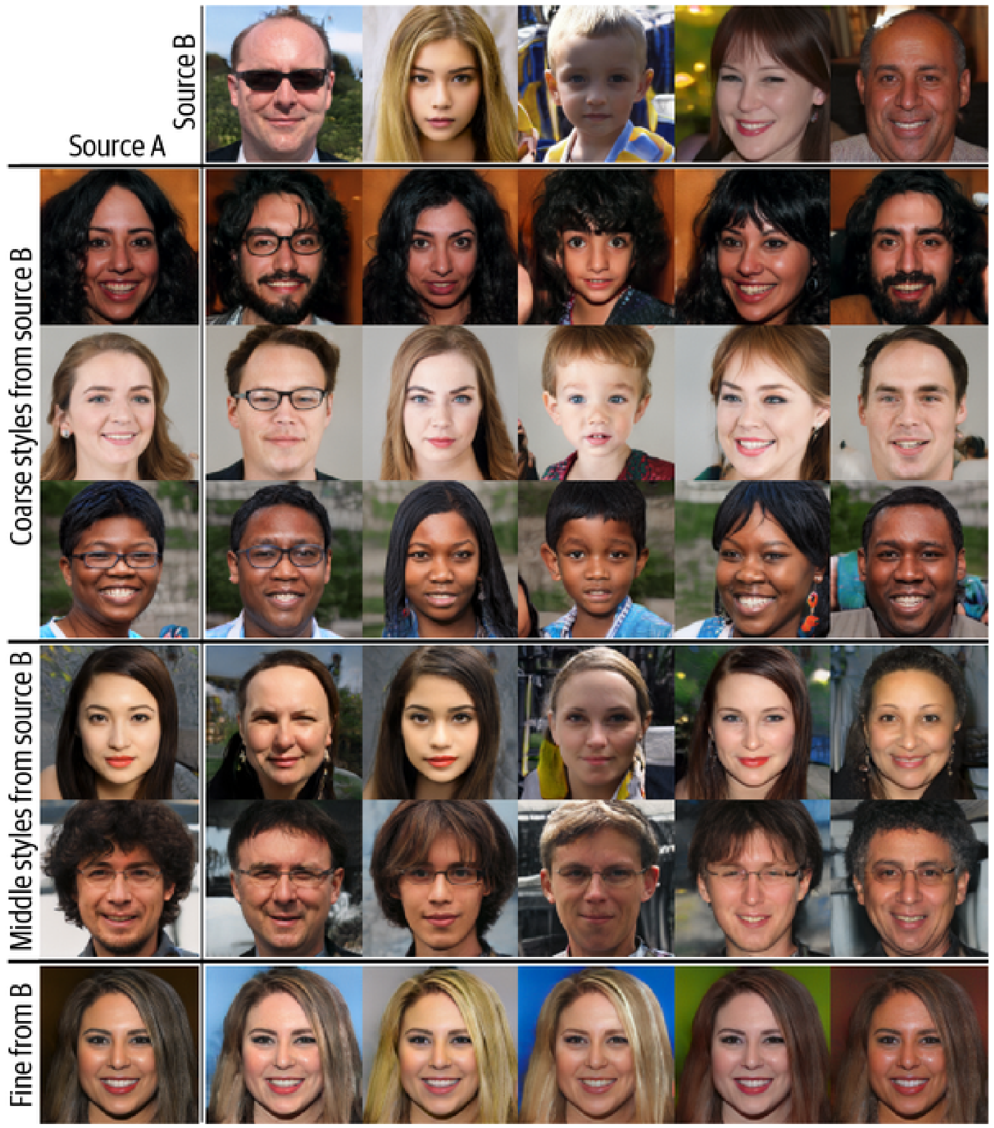
advanced gan

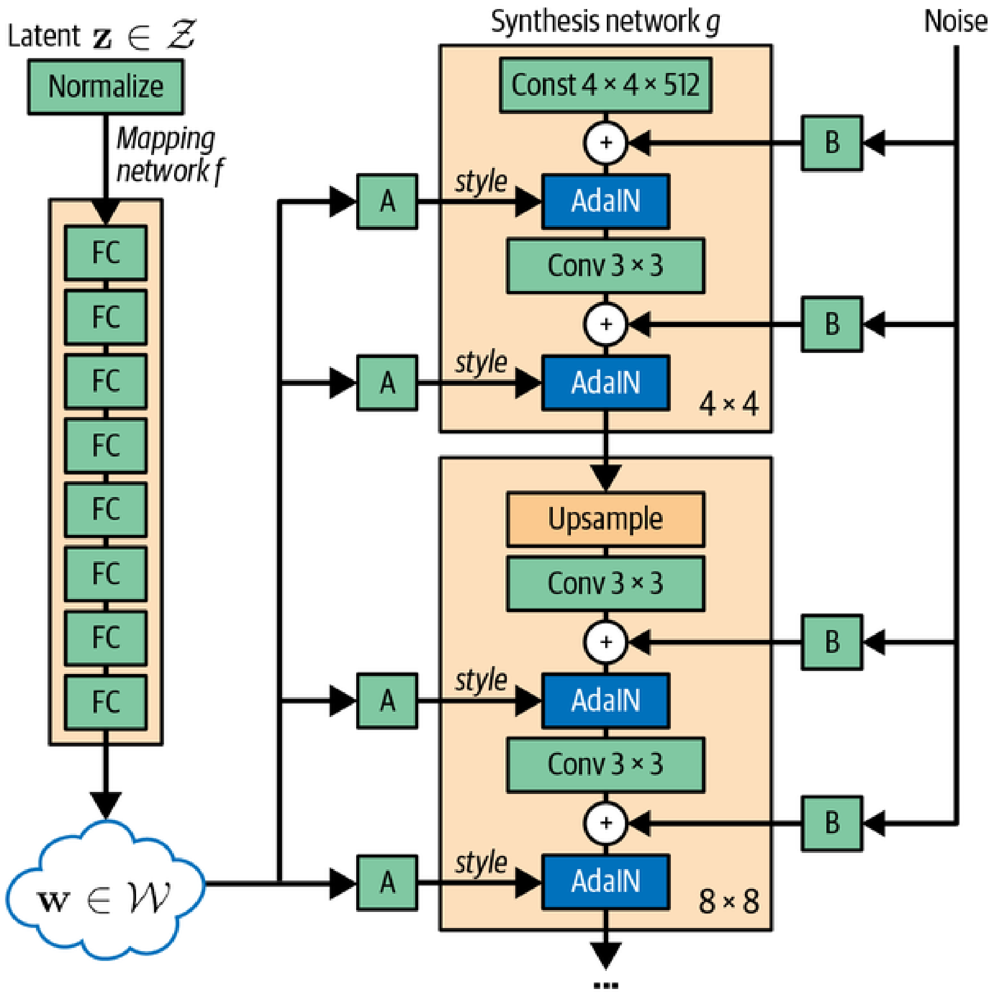
ProGAN: Progressively Upsizing the Image



- First train a model for lower resolution images
- Upscale the image in stages

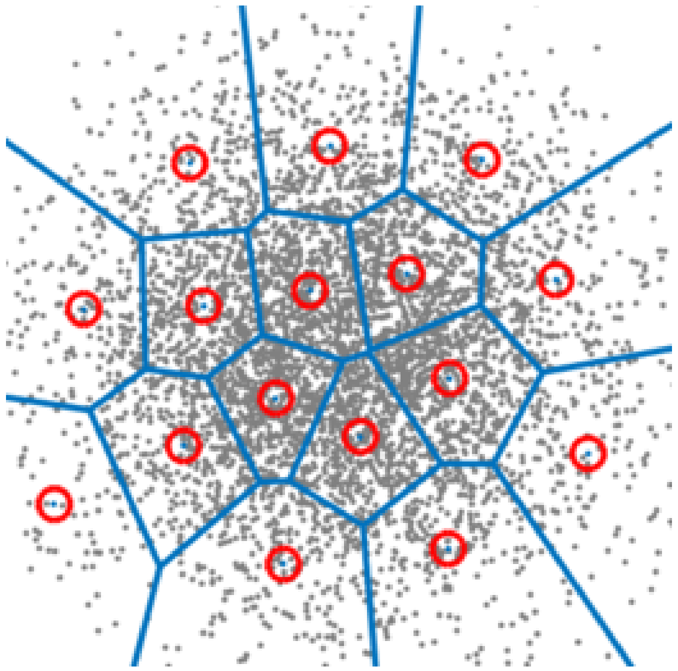
StyleGAN: Mixing Image Styles





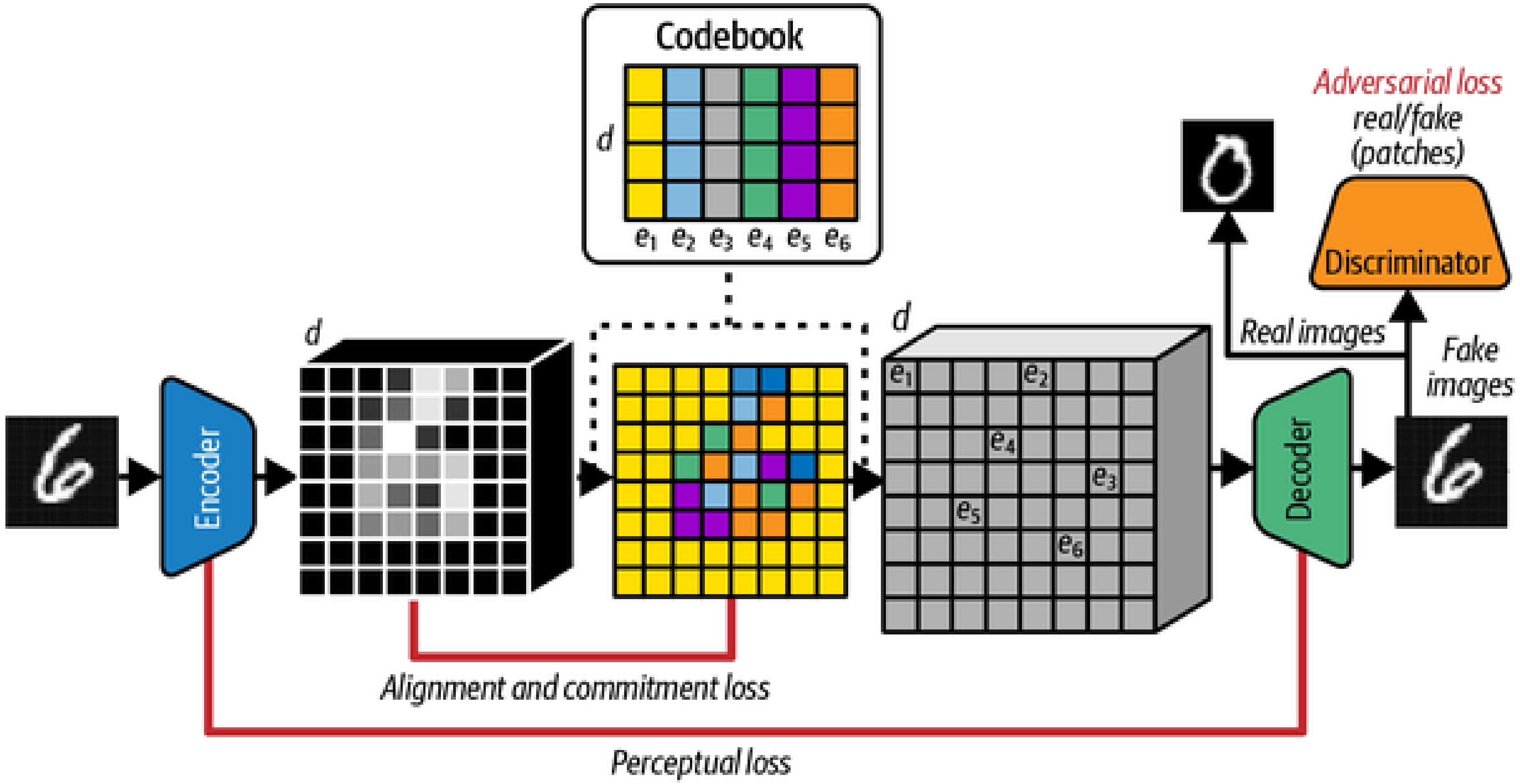
VQ-GAN: Vector Quantization

- Input: high dimensional vector
- K-Means Clustering



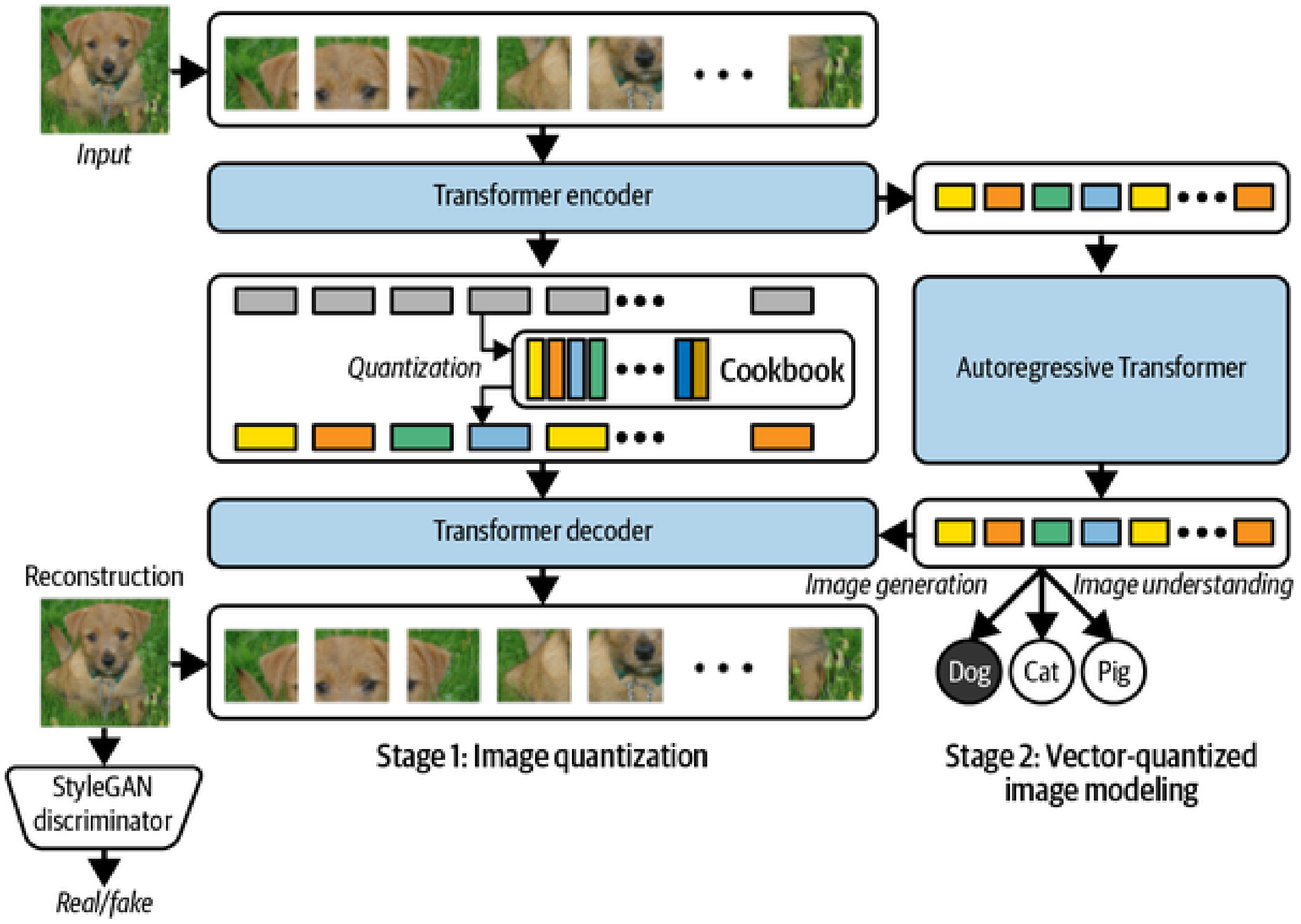
- Centroid vectors form a codebook, each vector is replaced with cluster ID

Discrete Latent Space



- Vectors are replaced by their nearest centroid

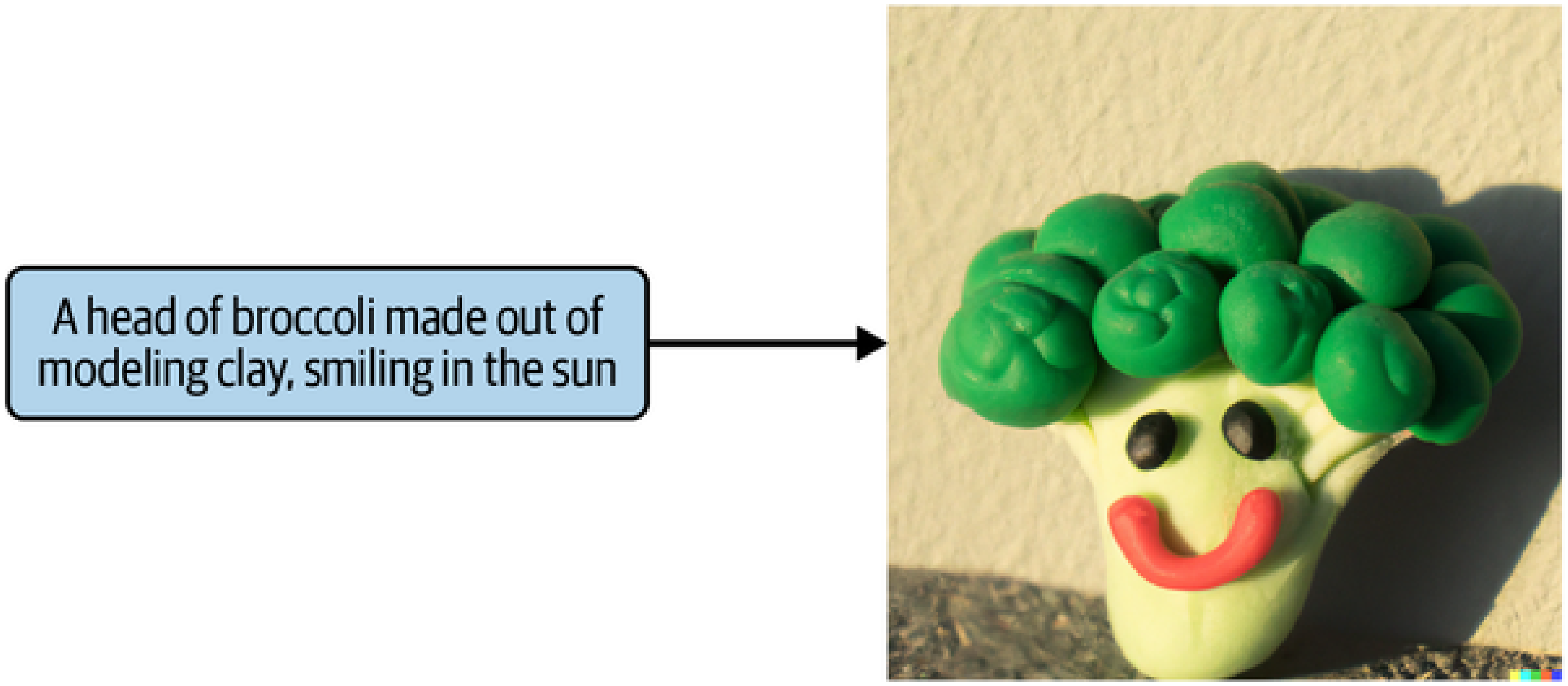
ViT VQ-GAN: Video Transformer



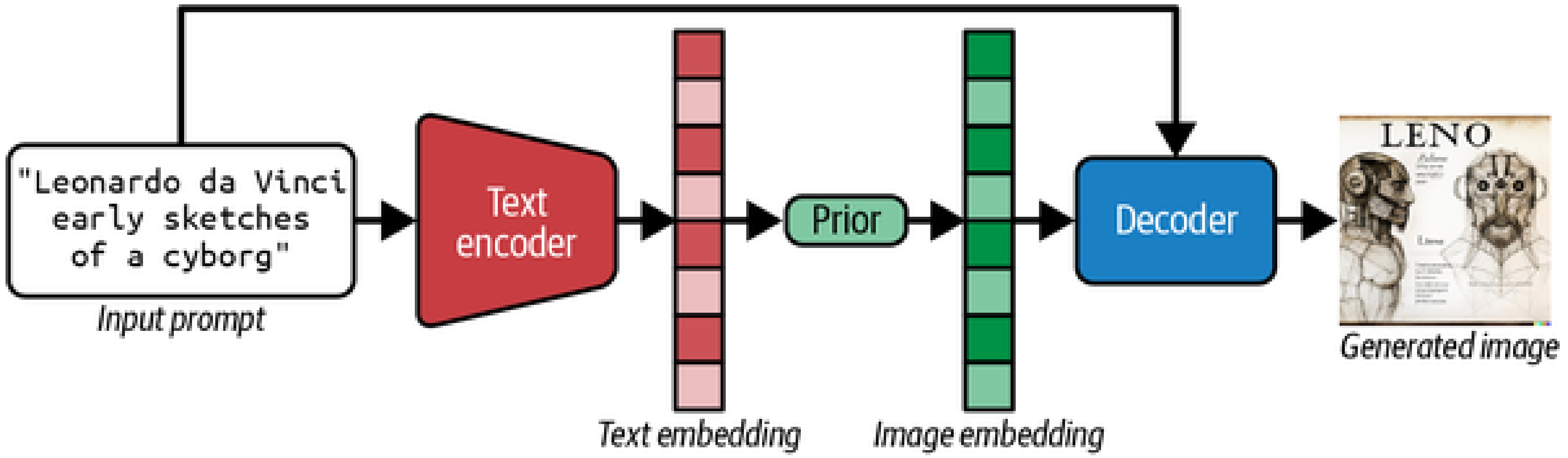
- Instead of using ConvNNs, Transformer model predicts sequence of patches
- This particular model also uses vector quantization

text to image generation

Text to Image

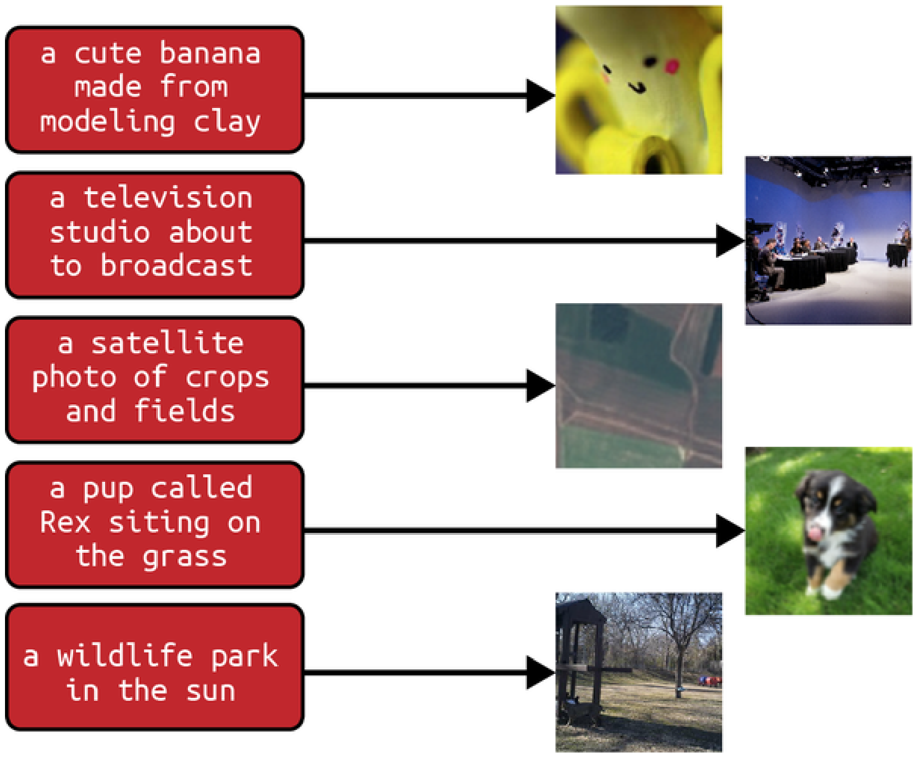

A head of broccoli made out of modeling clay, smiling in the sun

DALL.E 2



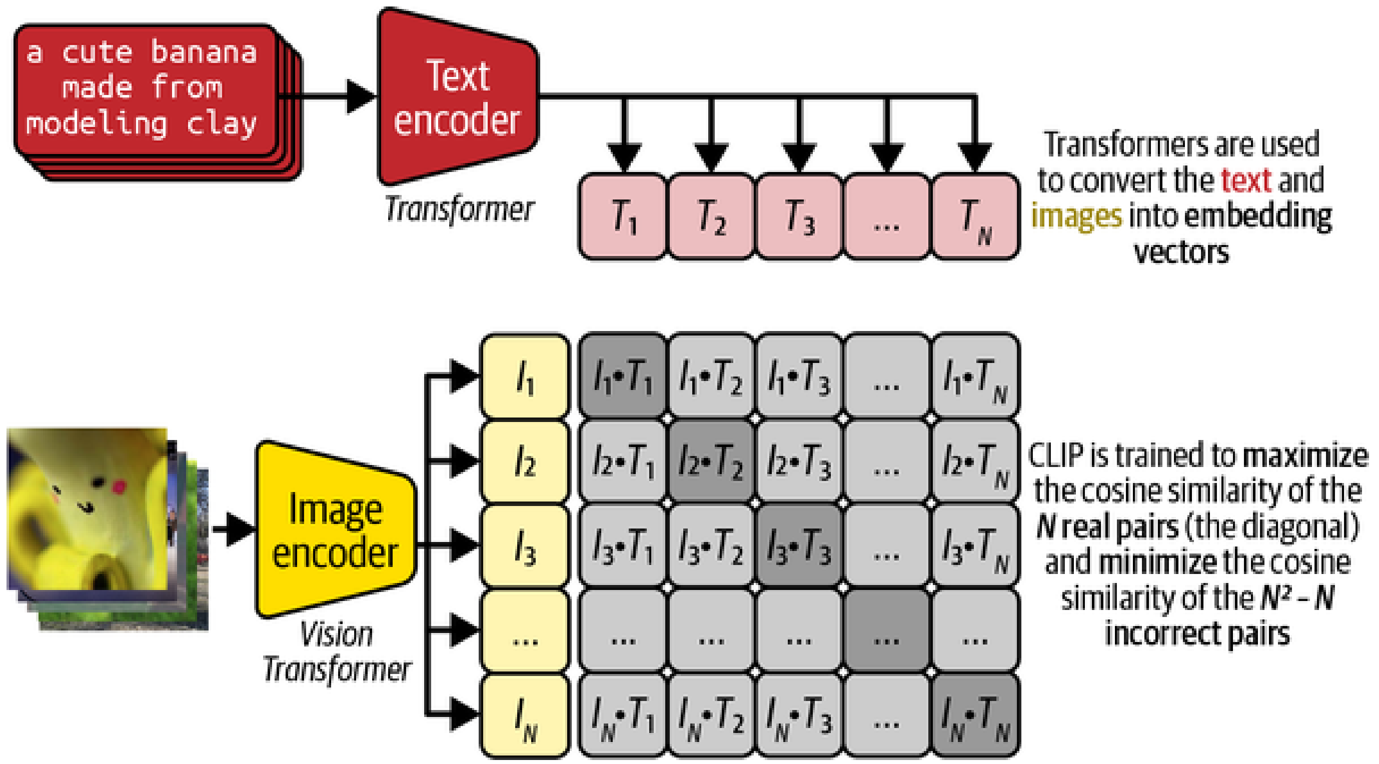
- Text is encoded as a prior to the image generation process
- Key training step: Contrastive Language-Image Pre-training (CLIP)

CLIP: Mapping Between Text and Images


a cute banana made from modeling clay
a television studio about to broadcast
a satellite photo of crops and fields
a pup called Rex siting on the grass
a wildlife park in the sun

- Given pairs of images and text (scraped from the Internet)
- Learn representations of text (using Transformer models)
- Learn representations of images (using ViT-VQ GAN)
- Learn mapping between them

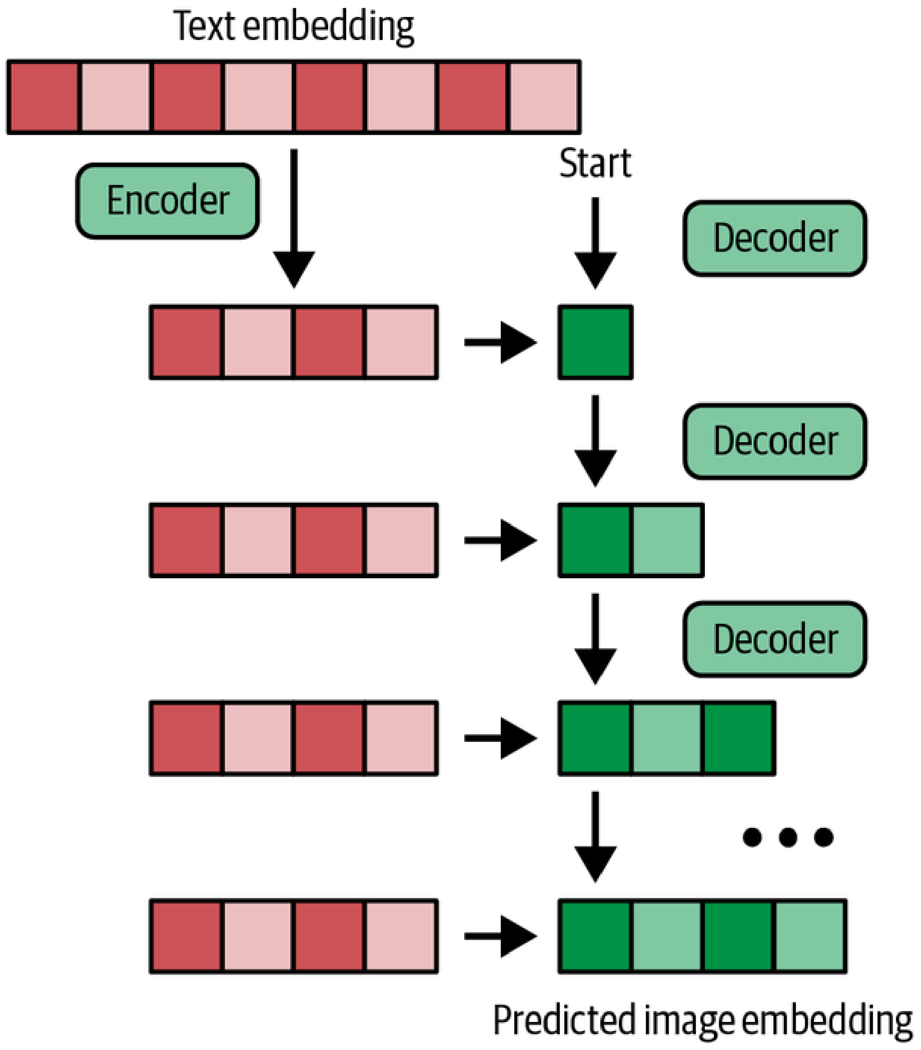
Contrastive Learning


a cute banana made from modeling clay
Text encoder
Transformer
T_1 T_2 T_3 ... T_N
Transformers are used to convert the text and images into embedding vectors
Image encoder
Vision Transformer
I_1 I_2 I_3 ... I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
... ...
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$
CLIP is trained to maximize the cosine similarity of the N real pairs (the diagonal) and minimize the cosine similarity of the $N^2 - N$ incorrect pairs

- Representation of text and image as a single vector, mapped to same size
- Training: minimize cosine similarity of real pairs
- Note: this is not a generative model

Generation

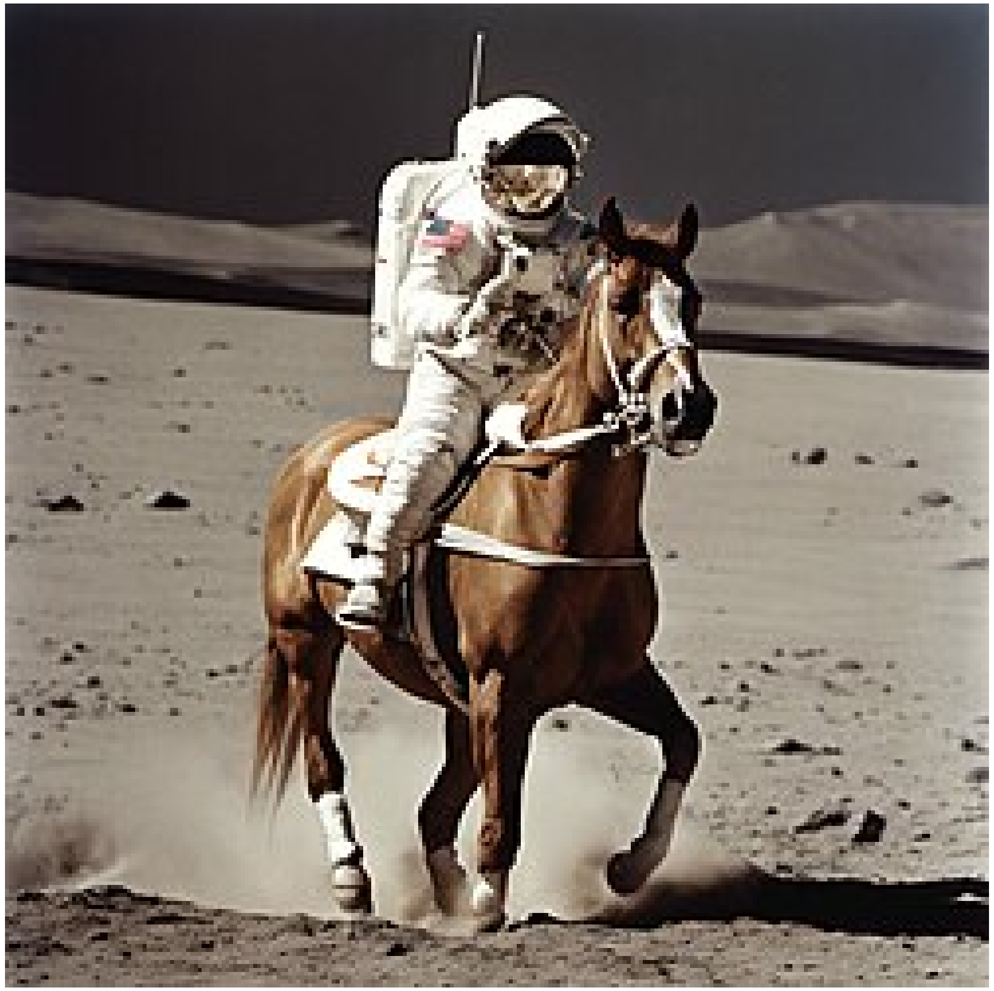
- Image decoder is a Transformer model
- Predicts patches of the image at each step
- Generation is also conditioned
  on the text representation (the prior)
- Alternatively: diffusion decoder



stable diffusion

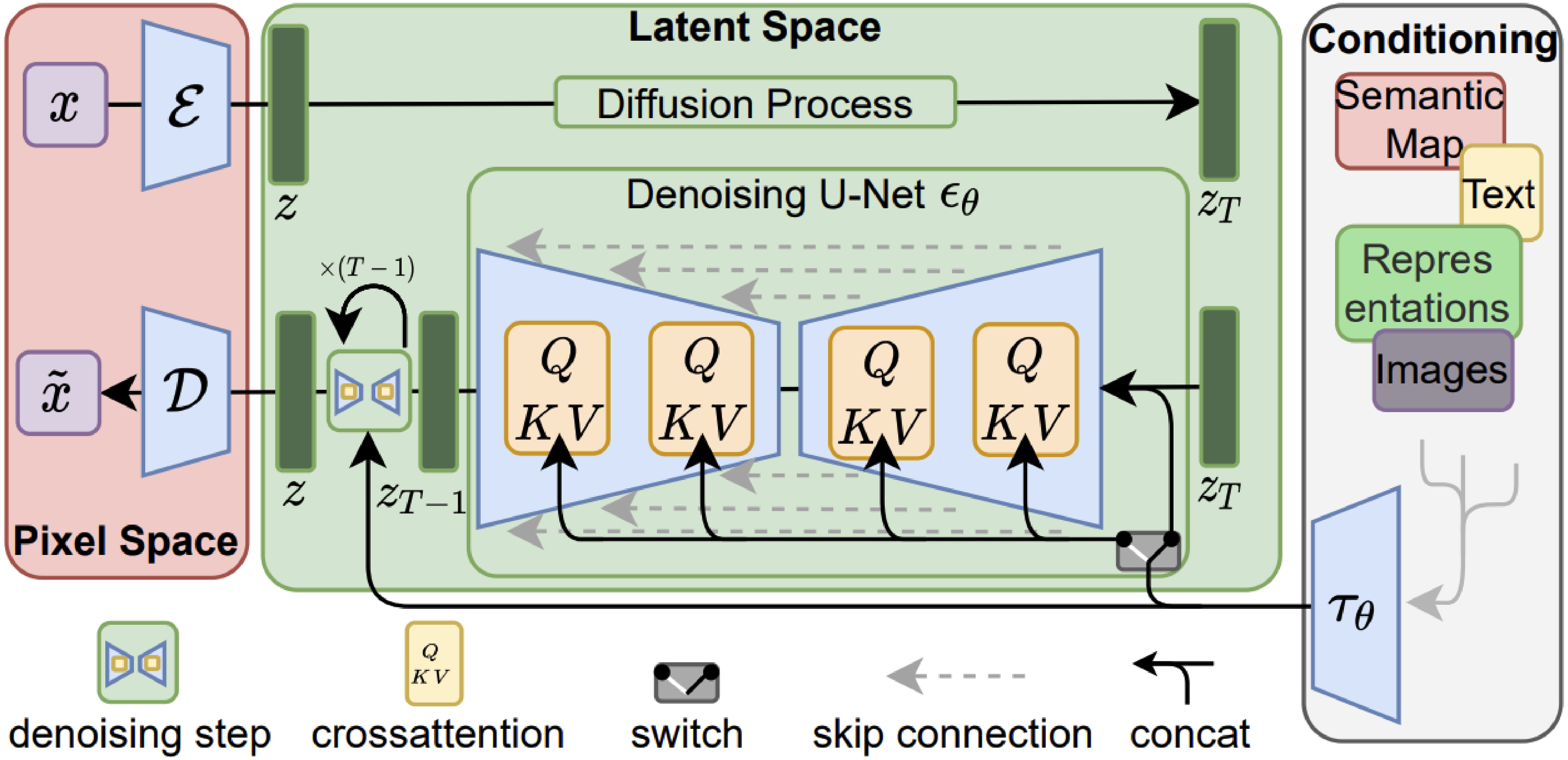
Robach et al. (2022):
High-Resolution Image Synthesis with Latent Diffusion Models

Stable Diffusion

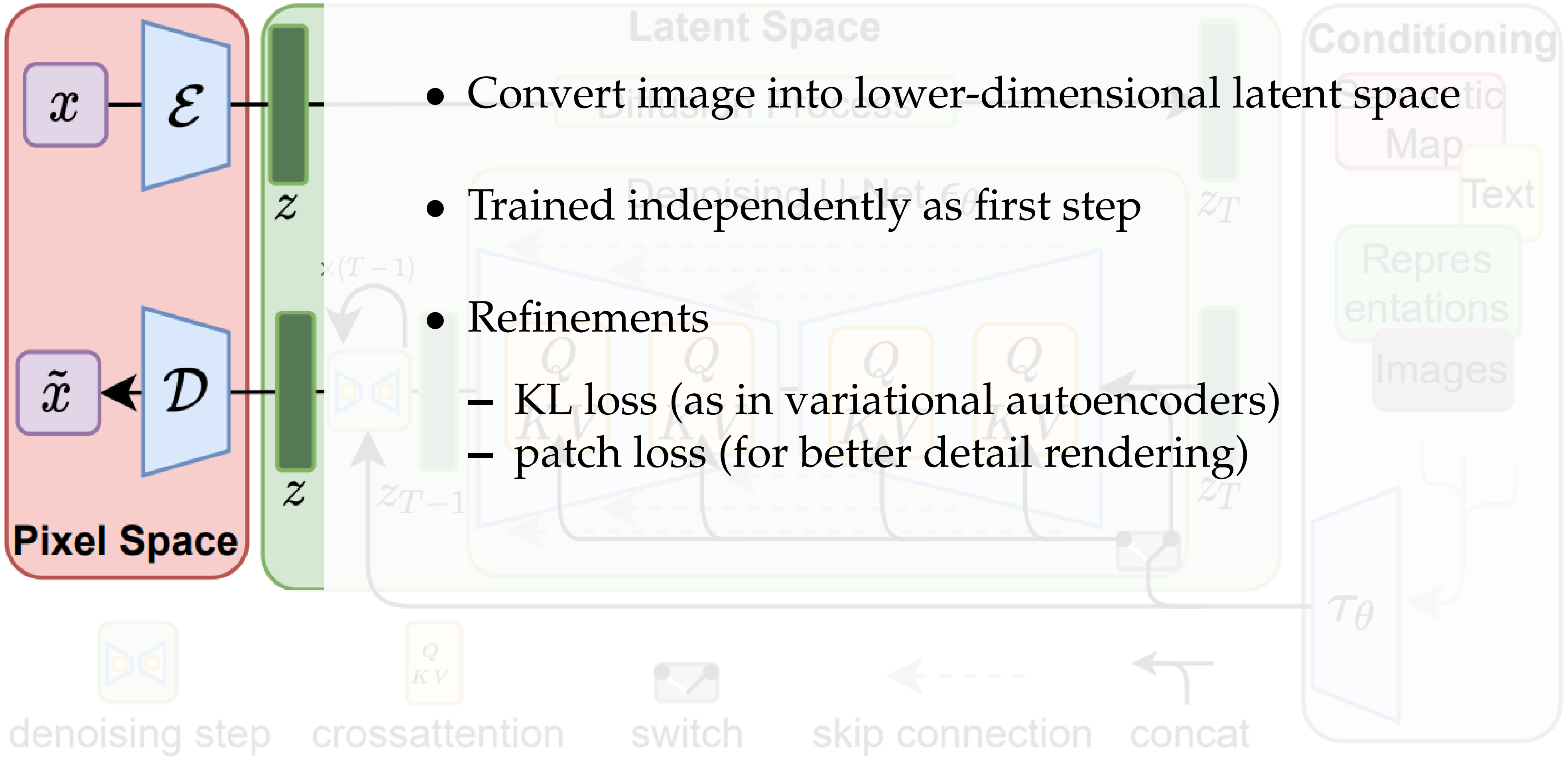


Generated from prompt “a photograph of an astronaut riding a horse”

Latent Diffusion Model

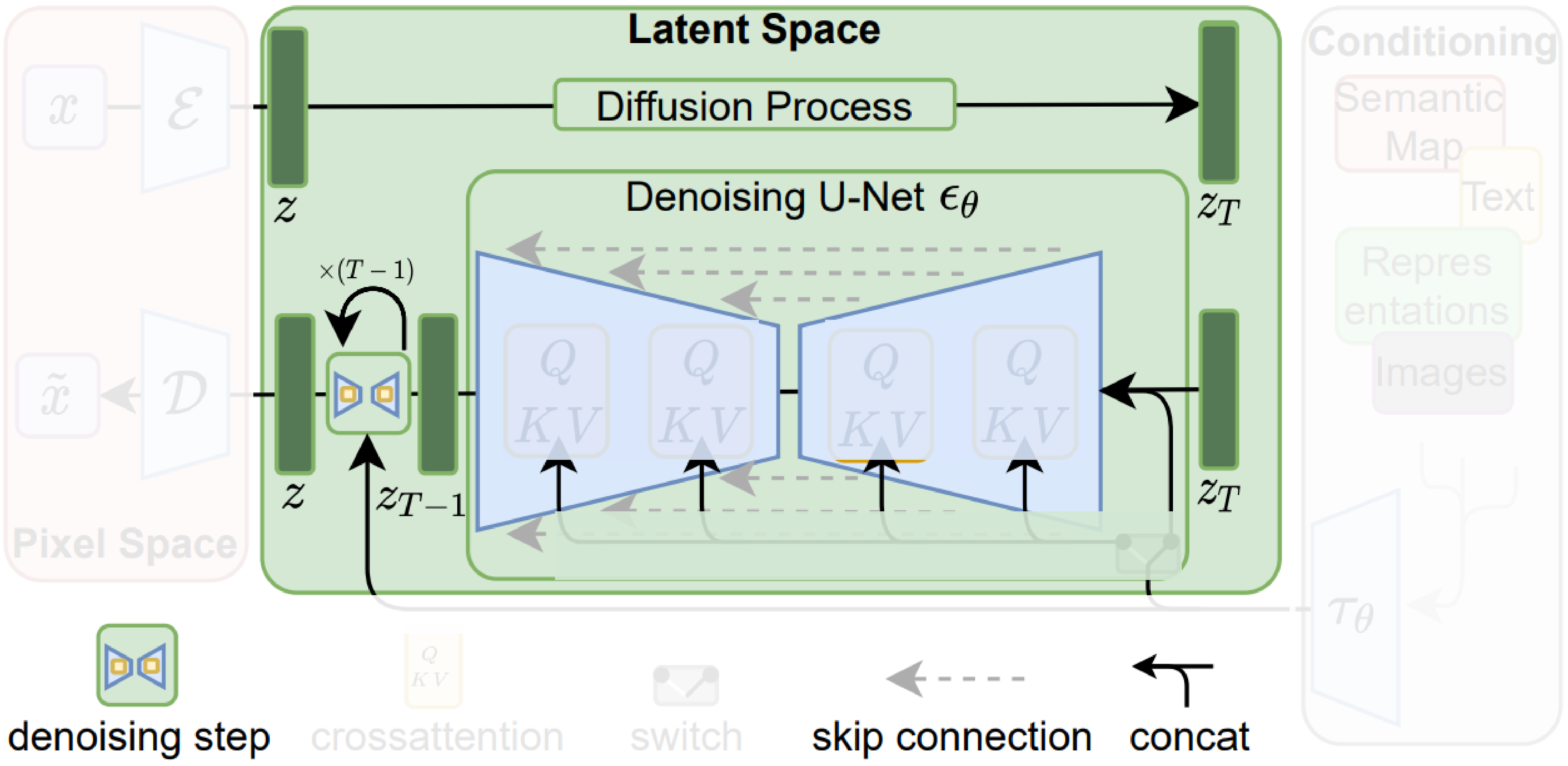


Autoencoder



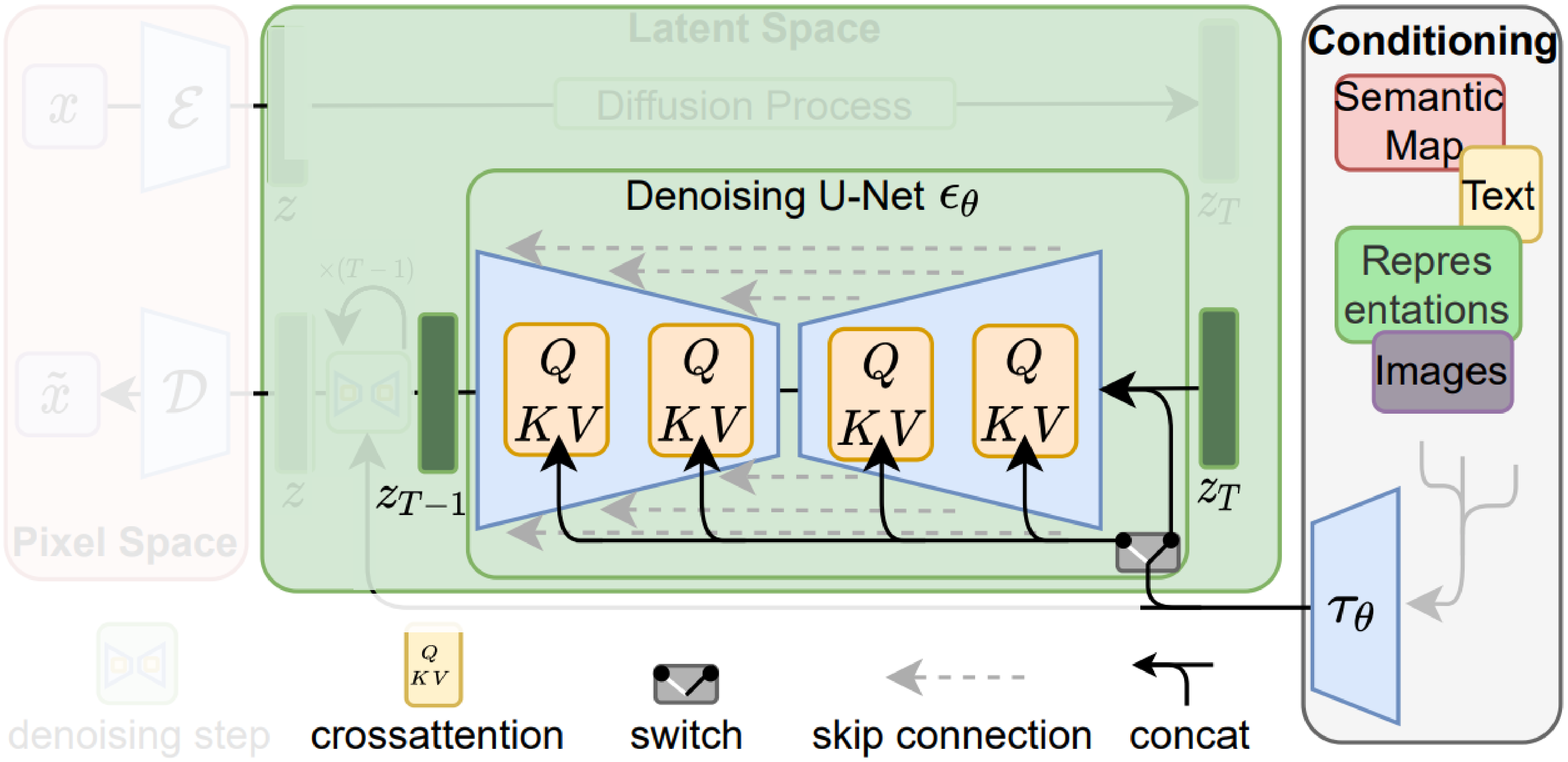
- Convert image into lower-dimensional latent space
- Trained independently as first step
- Refinements
  - KL loss (as in variational autoencoders)
  - patch loss (for better detail rendering)

Diffusion Model



Operates in latent space — otherwise the same U-Net from before

Conditioning on External Information



Attention (Query, Key, Value) to representations of semantic maps, text, images

Conditioning on External Information

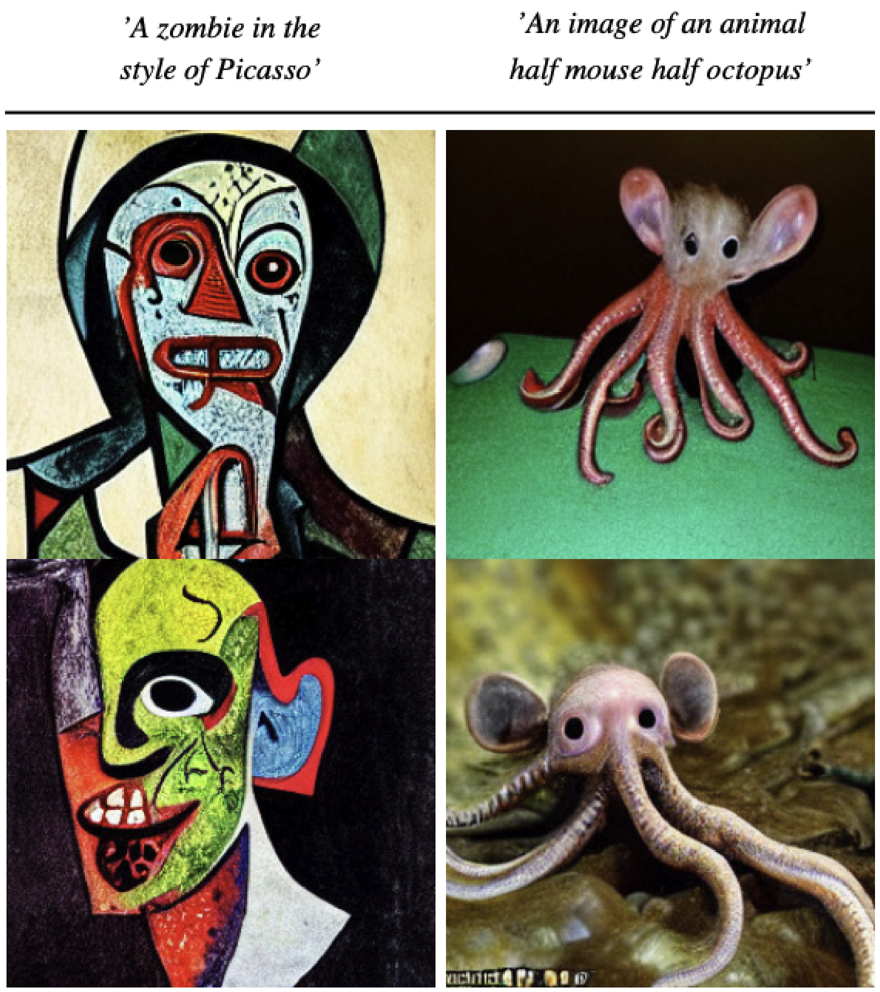
- External input y is converted into an intermediate representation $\tau_\theta \in \mathbb{R}^{M \times d_r}$
- For use in the attention model, the intermediate representations of the U-Net are also flattened to $\varphi_i(z_t) \in \mathbb{R}^{N \times d_\epsilon^i}$
- Attention is $$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \cdot V$$

  with $$Q = W_Q^{(i)} \cdot \varphi_i(z_t) \qquad K = W_K^{(i)} \cdot \tau_\theta(y) \qquad V = W_V^{(i)} \cdot \tau_\theta(y)$$

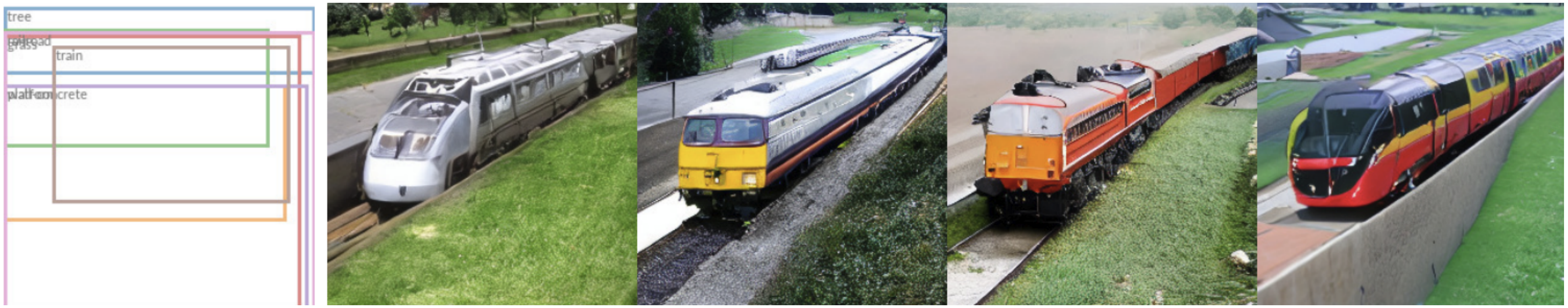
  that map to vectors of size d
- Parameters for τ_θ and ϵ_θ are jointly optimized using diffusion objective

Text to Image

- Text is represented as a sequence of words
- Transformer model generates $\tau_\theta \in \mathbb{R}^{M \times d_r}$
- Trained on language prompts
  - LAION-400M
  - 400 million text-image pairs
  - extracted from web pages with alt-text in HTML image tag
  - filtered in various ways

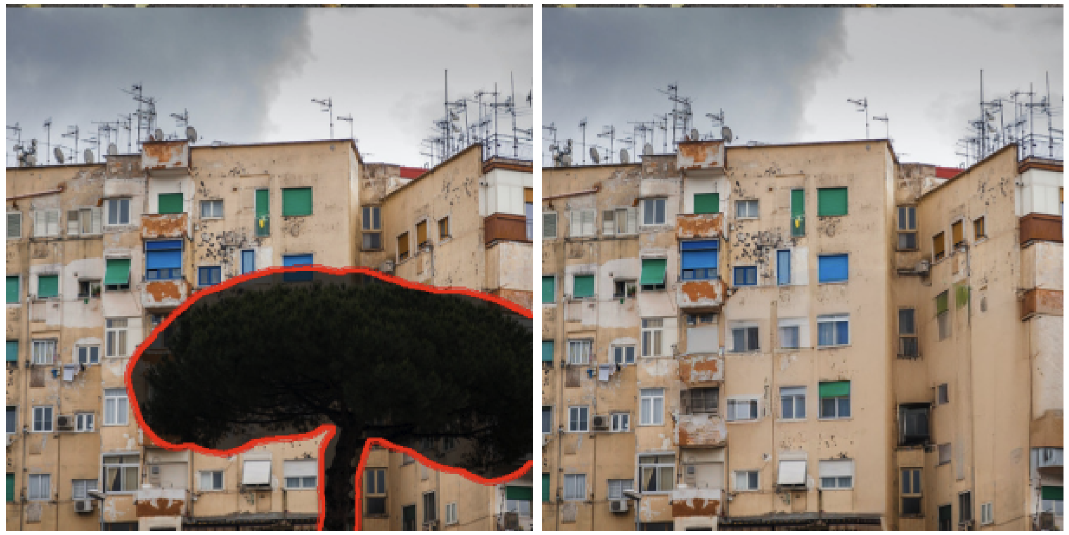

'A zombie in the style of Picasso'
'An image of an animal half mouse half octopus'

Semantic Maps



- Trained on Open Images dataset of images with labelled object detection
  - 9.2 million images collected from Flickr
  - bounding boxes with object labels
  - computer-assisted annotation: automatic labels vetted by humans

Inpainting



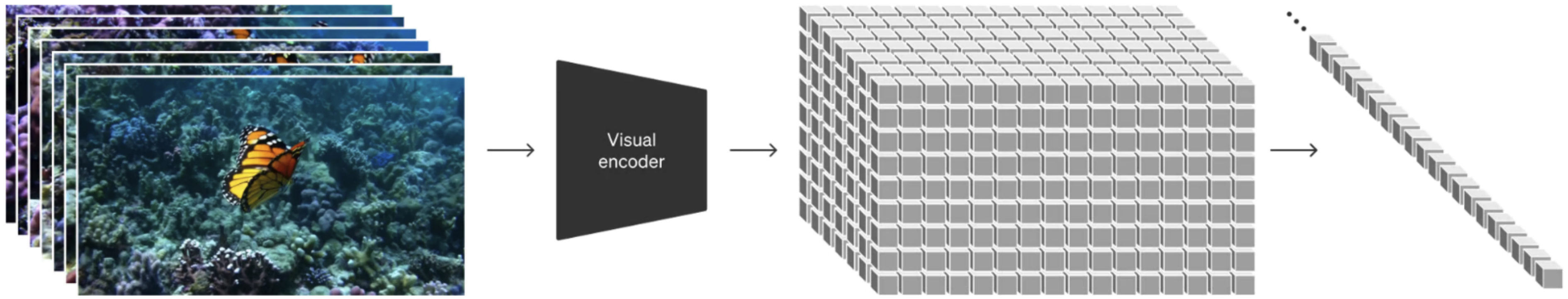
- Automatic generation of training data with synthetic masks
- Training aims to reconstruct the original image

video generation

OpenAI (2024): Video generation models as world simulators (Sora)

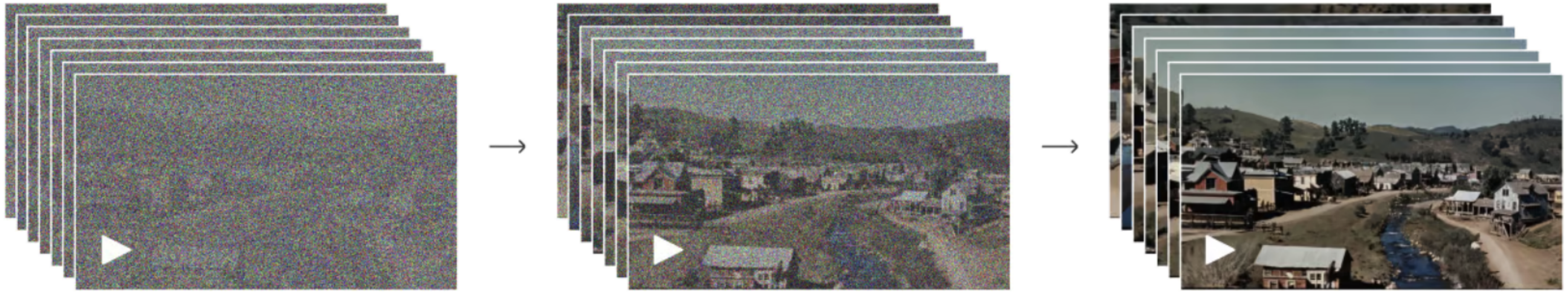
`https://openai.com/research/video-generation-models-as-world-simulators`

Video Representation



- Up to a full minute of high definition video
- Compress image into lower dimensional latent space
- Decompose representation into spacetime patches

Diffusion Model



- Given noisy patches and conditioning text prompts
- Predict original clean patches using Transformer model
- Training data generated by re-captioning existing videos
  - first, train captioning model
  - use it to produce highly descriptive text captions for video

Beyond Text Prompting

- Prompting with images (maybe images generated by DALL-E)
- Extending existing video (forward or backward)
- Video-to-video editing (video + text prompt $\rightarrow$ new video)
- Connecting videos

questions?