Image Generation

Generative Artificial Intelligence

M. Sc. Angewandte Informatik

Prof. Dr. Marc Aubreville

Hochschule
Flensburg
University of
Applied Sciences

Introduction / Motivation

- Image generation opens up fascinating opportunities:
  - Data augmentation for machine learning models
  - Personalized content creation, including enhanced artistic expression
  - Generation of realistic images from simulations (—> knowledge of ground truth)
  - Escaping the copyright-problem that web-images often comprise

Hochschule Flensburg
University of Applied Sciences

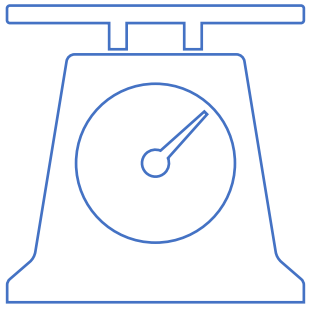
Hochschule
Flensburg
University of
Applied Sciences

Metrics
for Image Generation

Metrics for image generation assessment

- In order to assess image generation not only visually, we have to briefly introduce metrics.

- However, as we will see, this is a non-trivial task, and and active research topic.



Metric 1: Inception Score (Salimans et al., 2016)

- Generative adversarial networks lack an objective function, which makes it difficult to compare performance of different models (Salimans et al., 2016).

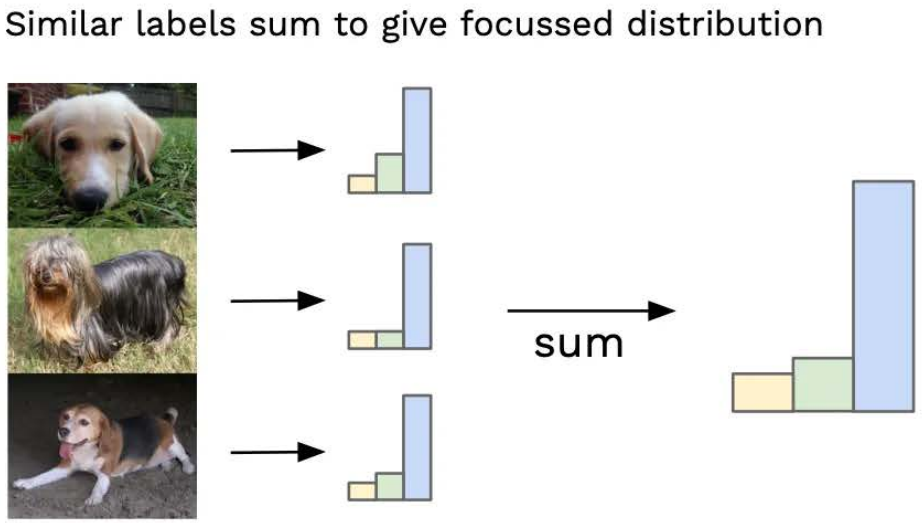
- The first work in this area hypothesized, that when feeding the generated images through an Inception network trained on ImageNet, the network should be able to decide for one of the classes.

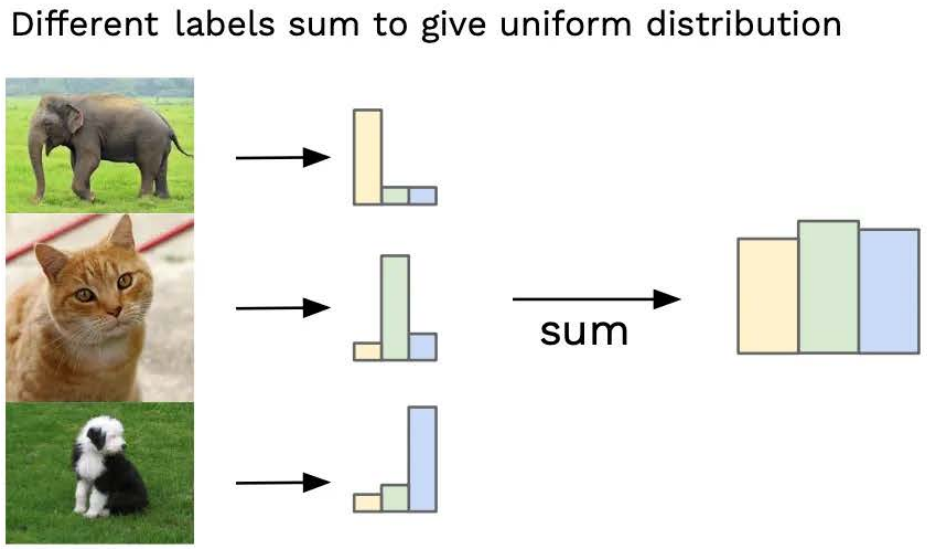
- In other words: we assume the distribution $p(y \mid \mathbf{x})$ for individual images $\mathbf{x}$of the InceptionV3 network output to have a low entropy.

- Over all images, however, we expect the marginal distribution $p(y) = \int p(y \mid x = G(z))dz$ to have high entropy (e.g., be close to uniform distribution).

https://arxiv.org/pdf/1606.03498.pdf

Metric 1: Inception Score (Salimans et al., 2016)





low entropy for individual images

high entropy for the marginal distribution

- The Inception Score combines both in that it calculates the KL-divergence between both distributions:

$$IS = \exp\left(\mathbb{E}_x KL(p(y\,|\,x)\|p(y))\right)$$

- It uses the exponential of the KL divergence to ensure better interpretability.

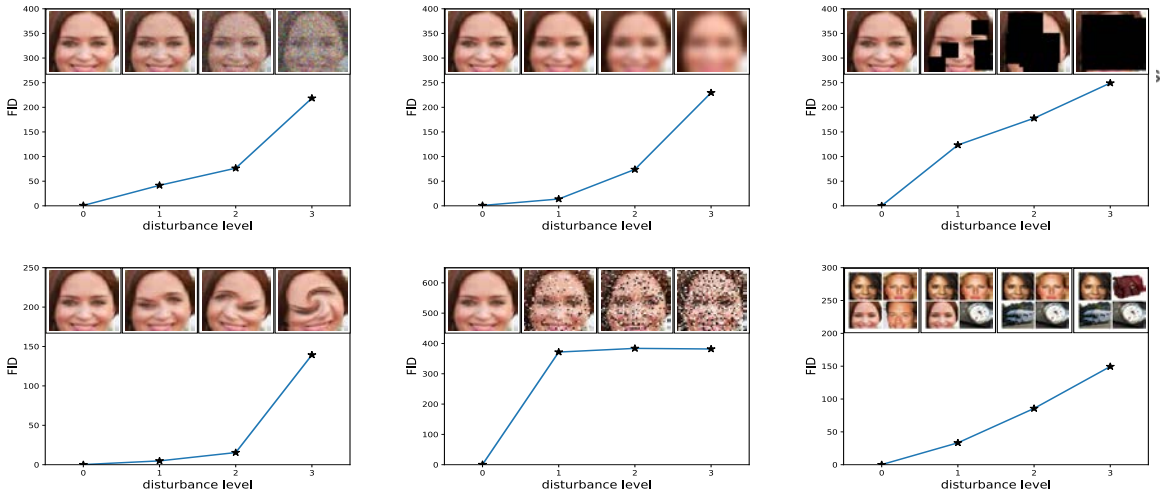
https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

https://arxiv.org/pdf/1606.03498.pdf

Metric 2: Fréchet Inception Distance (FID, Heusel et al., NeurIPS, 2017)

- The Inception Score has known limitations, however, since the real world is not only the ImageNet classes.

- It is desirable to derive a metric that reflects all kinds of image pertubations, such as adding noise or blurring the image.



FID is evaluated for upper left: Gaussian noise, upper middle: Gaussian blur, upper right: implanted black rectangles, lower left: swirled images, lower middle: salt and pepper noise, and lower right: CelebA dataset contaminated by ImageNet images. The disturbance level rises from zero and increases to the highest level. The FID captures the disturbance level very well by monotonically increasing.

https://papers.nips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf

Metric 2: Fréchet Inception Distance (FID, Heusel et al., NeurIPS, 2017)

Hochschule Flensburg
University of Applied Sciences

- The FID only uses the encoder of the InceptionV3 architecture to yield more generalizing features.

- The FID then calculates a metric comparing the mean m and covariances C of the features in the generated and original data distribution (w):

$$d^2(m, C) = \|m - m_w\|_2^2 + \mathrm{Tr}\left(C + C_w - 2\left(CC_w\right)^{\frac{1}{2}}\right)$$

- The FID has shown to be correlated to human judgement for a range of pertubations.

- Effectively, however, it only measures similarity of distributions, FID=0 indicates equality of distributions.

https://papers.nips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf

FID of popular approaches

- Comparing the FID metric across popular approaches gives us an intuition of which values to expect:

CelebA-HQ 256 × 256		FFHQ 256 × 256	
Method	FID ↓	Method	FID ↓
GLOW [37]	69.0	VDVAE ($t = 0.7$) [11]	38.8
NVAE [69]	40.3	VDVAE ($t = 1.0$)	33.5
PIONEER (B.) [23]	39.2 (25.3)	VDVAE ($t = 0.8$)	29.8
NCPVAE [1]	24.8	VDVAE ($t = 0.9$)	28.5
VAEBM [77]	20.4	*VQGAN*+P.SNAIL	21.9
Style ALAE [56]	19.2	BigGAN	12.4
DC-VAE [54]	15.8	VQGAN	9.6
VQGAN	10.2	U-Net GAN (+aug) [66]	10.9 (7.6)
PGGAN [31]	8.0	StyleGAN2 (+aug) [34]	3.8 (3.6)

https://arxiv.org/abs/2012.09841

Weaknesses of the FID

Hochschule Flensburg
University of Applied Sciences

- FID uses the InceptionV3 latents to compare the distribution to an original distribution.

- This may be, however, susceptible to imperceivable artifacts (that the model does see, the latents are not invariant to)

Manipulation	Configuration	FID @ COCO 30k	FID @ CelebA-HQ
JPEG compression	quality=95%	0.268	0.560
	quality=90%	1.713	2.381
	quality=80%	6.658	6.291
	quality=70%	10.469	9.156
	quality=60%	13.274	11.617
	quality=50%	15.129	13.519
Resampling	NN interpolation	5.239	3.705
	bilinear interpolation	0.330	0.569
Color change	8-bit color palette	27.989	31.289
	brightness +10%	0.085	0.112
	brightness -10%	0.054	0.101
	contrast +10%	0.051	0.077
	contrast -10%	0.072	0.098

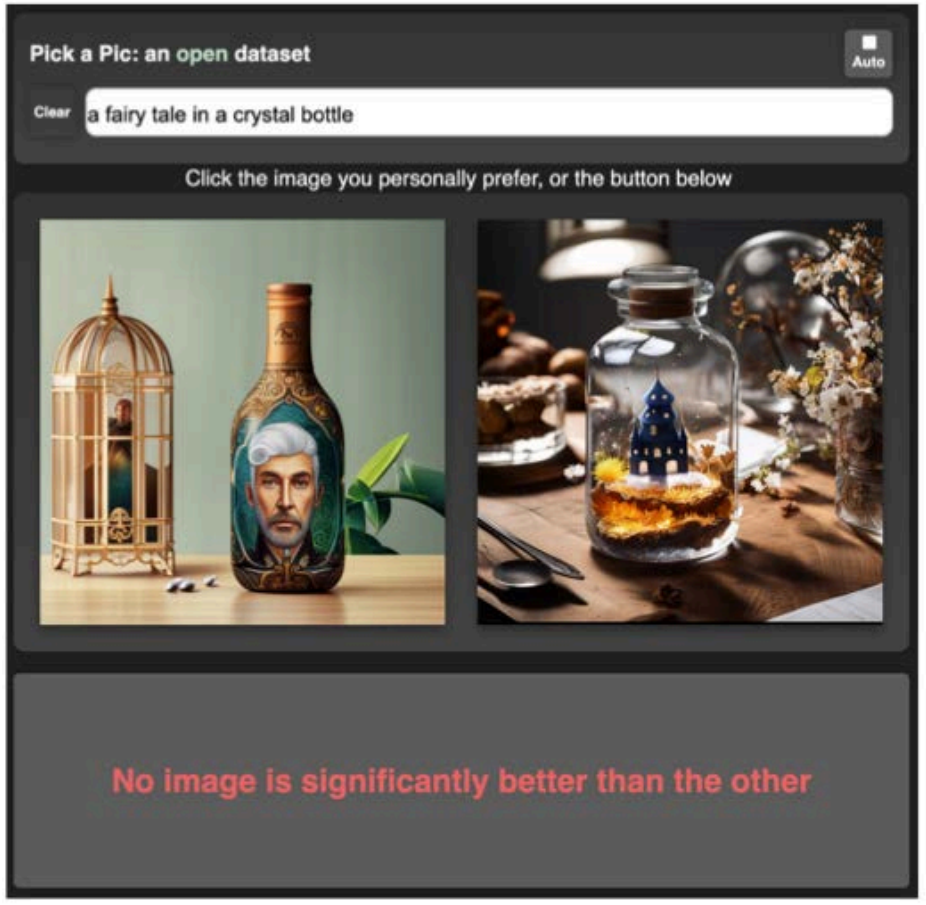
- The FID is additionally strongly dependent on the dataset / data distribution it is being compared against (typically we compare against MS COCO and don't use this dataset for training).

- If images of a certain style (e.g., artistic, very beautiful images) need to be generated, this implicitly means that the FID can't be high, while the image fidelity is (data distribution shift).

https://arxiv.org/pdf/2306.00637.pdf

PickScore (Kirstain et al., 2023)

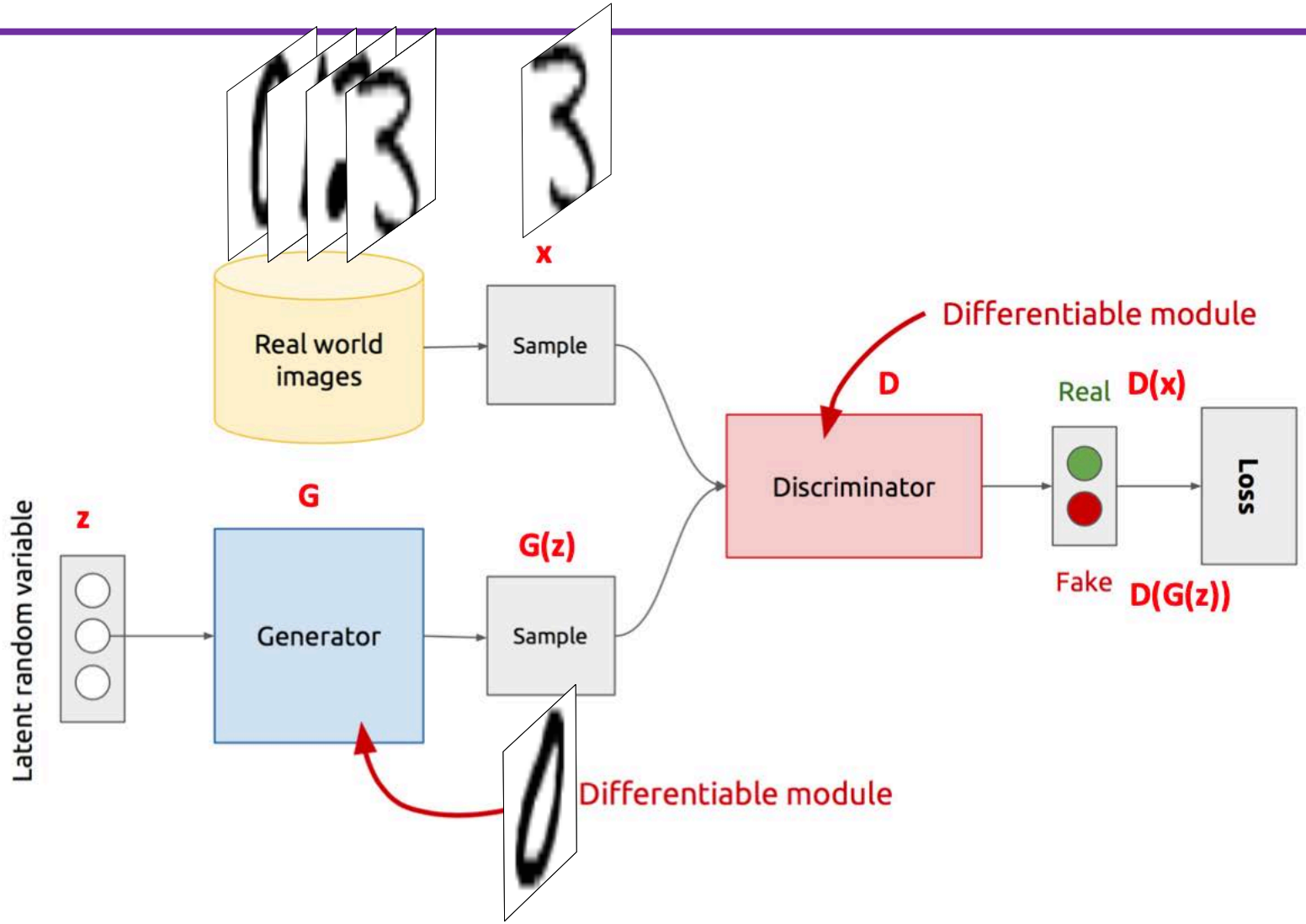
- Image generation quality is subject to many factors, most of whom can't sufficiently be captured using statistical assessment of image distributions.
- The best way to evaluate perceived quality is thus by human evaluation.
- However, this is tedious and can be costly.
- Kirstain et al. made a dataset available of 500k examples and 35k prompts, together with human preference ratings.
- On top of that, they trained a classifier to estimate human preference (PickScore).



Hochschule
Flensburg
University of
Applied Sciences

Adversarial Image Generation

Recap: Generative Adversarial Networks



Z is a random noise (Gaussian/uniform), it can be interpreted as the latent space representation of an image.

https://slazebni.cs.illinois.edu/spring17/lec11_gan.pdf

Recap: Generative Adversarial Networks

Hochschule Flensburg
University of Applied Sciences

- GANs are formulated as a minimax game, where the
  - Discriminator is trying to maximize the reward $V(D, G)$ (or minimize the loss)
  - Generator is trying to minimize the reward of the discriminator $V(D, G)$

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

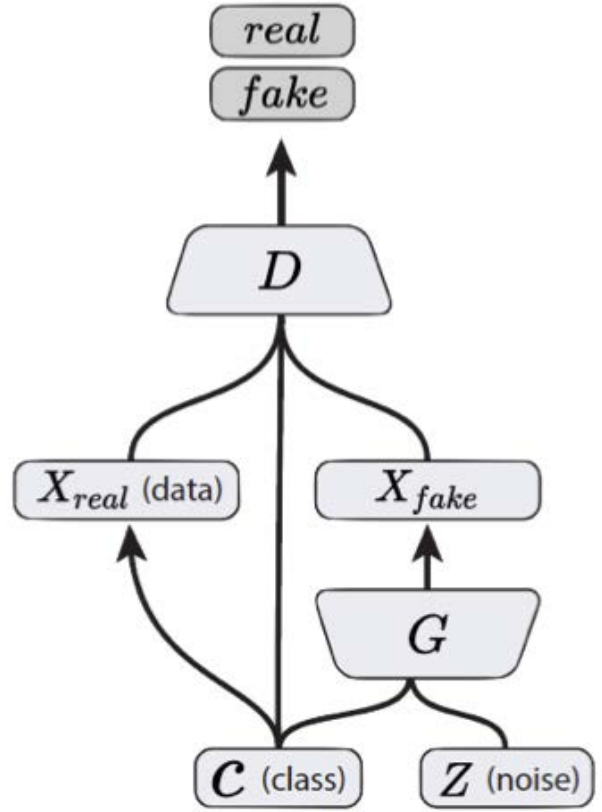
Expected value (log likelihood) of predicting the real images correctly

Expected value (log likelihood) of predicting the generated images correctly

  - The Nash equilibrium is achieved, if neither the generator nor the discriminator can improve to become better.

Recap: Conditional GANs

- Another architecture that is not prone to mode collapse are conditional GANs.
- They condition the network on additional information to improve multi-modal learning.
- It is, however, required to have a labelled dataset for this, as the condition controls the selection of samples.

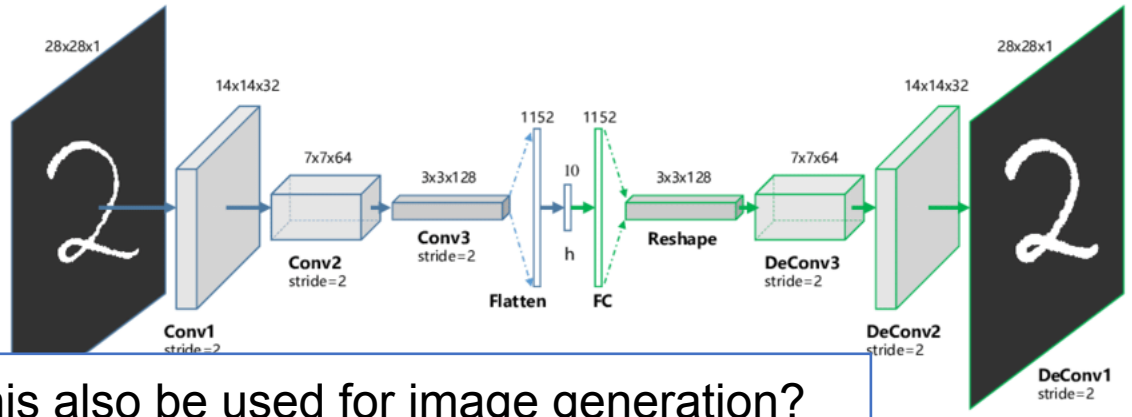


https://slazebni.cs.illinois.edu/spring17/lec11_gan.pdf

Recap: Autoencoders

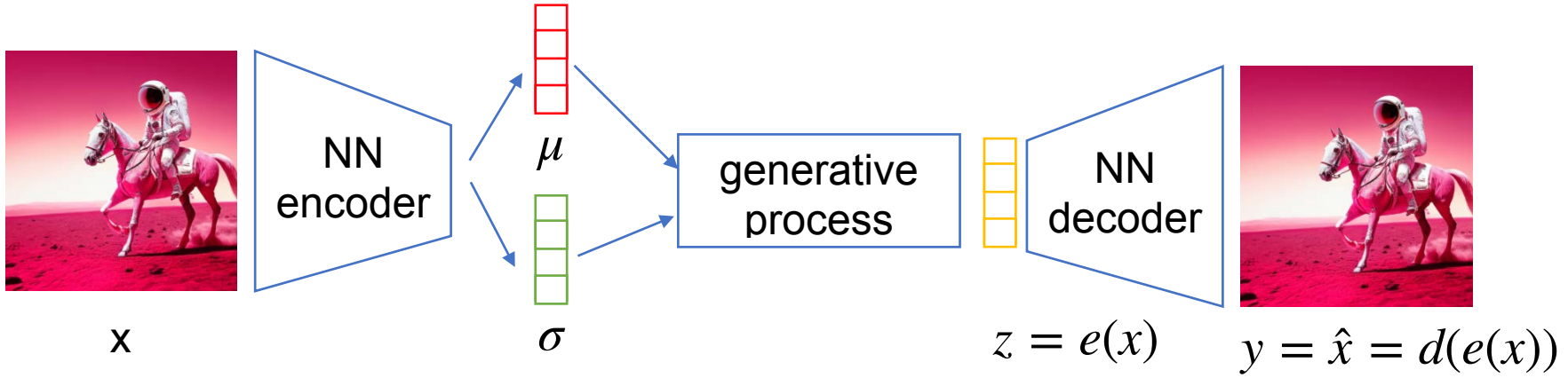
Hochschule Flensburg
University of Applied Sciences

- In computer vision, autoencoders are models that will encode and decode an image to retrieve the same image.
- Due to the bottleneck of the network, the feature vector there needs to be sufficiently discriminative to reconstruct the network.
- There are many applications for autoencoders, including:
  - Denoising (since noise is commonly uncorrelated to the (semantic) information of the image)
  - Projection / Clustering of Images
  - Image compression
  - Building unsupervised encoders to be used in model finetuning / transfer learning
  - Superresolution (increasing the resolution of the input image)



Could this also be used for image generation?

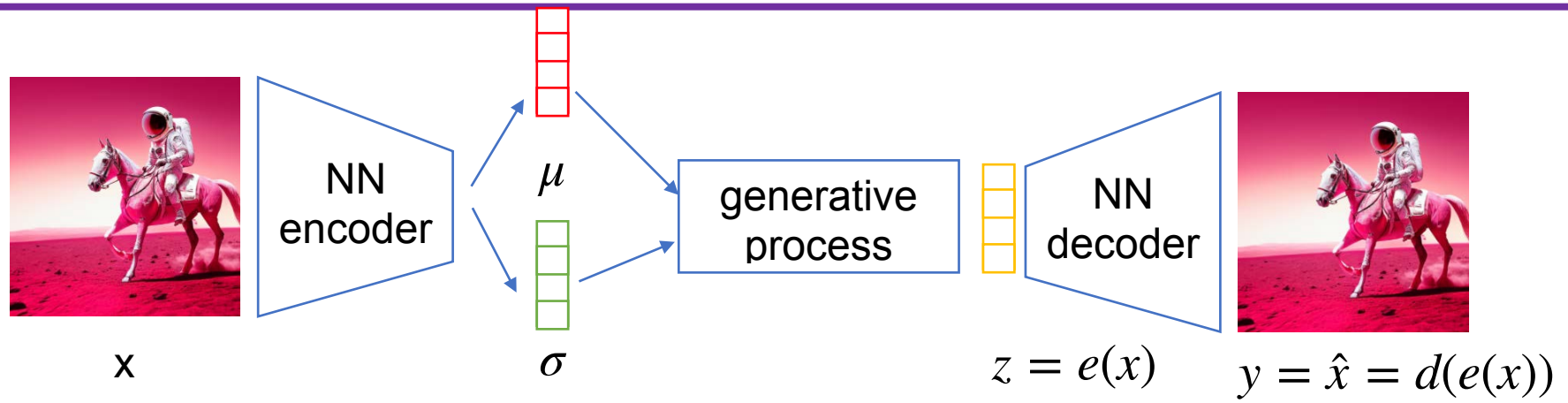
Adding variational constraints to autoencoders



- Instead of encoding an image directly, in variational autoencoders, we use the encoder to predict the parameters of a normally distributed variable (i.e., the mean and covariance matrix) of the respective input image in latent space.
- We then consider this distribution a random process and sample from it.
- The decoder has the task to decode from the sampled latent representation.
- Hence, it is forced to learn to generate images from normal distributions.

https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Recap: Variational Autoencoders



- Variational Autoencoder use both, a reconstruction loss and a regularization term (KL-Divergence).

$$L = L_{\text{recon}} + L_{KL}$$

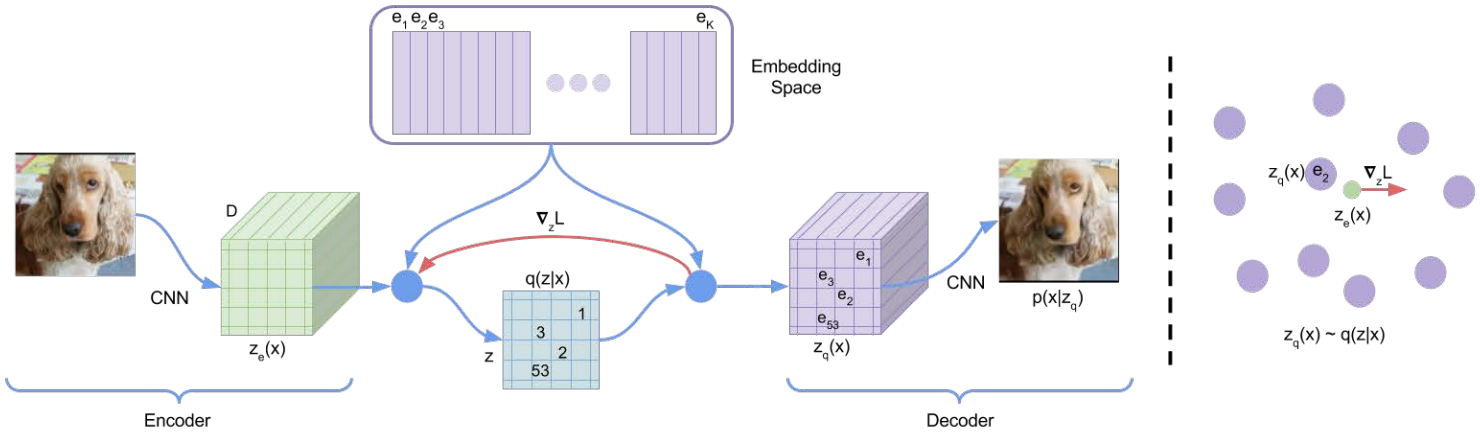
- Naturally, there is a balance between both loss terms, that can be found using a weight term.

Vector-Quantized Variational Autoencoders (VQ-VAE, van den Oord et al., 2017)

Hochschule Flensburg
University of Applied Sciences

- Having a fully continuous latent space does impose severe requirements on data, however, since the encoder now has to learn generalizing solutions for all possible combinations of latent vectors in infinite resolution.

- Vector-Quantized Variational Autoencoders (VQ-VAE) reduce the cardinality in the latent space by quantizing the latent vectors through a nearest neighbor scheme.



https://arxiv.org/pdf/1711.00937.pdf

Vector-Quantized Variational Autoencoders: An Intuition-based motivation

Hochschule Flensburg
University of Applied Sciences

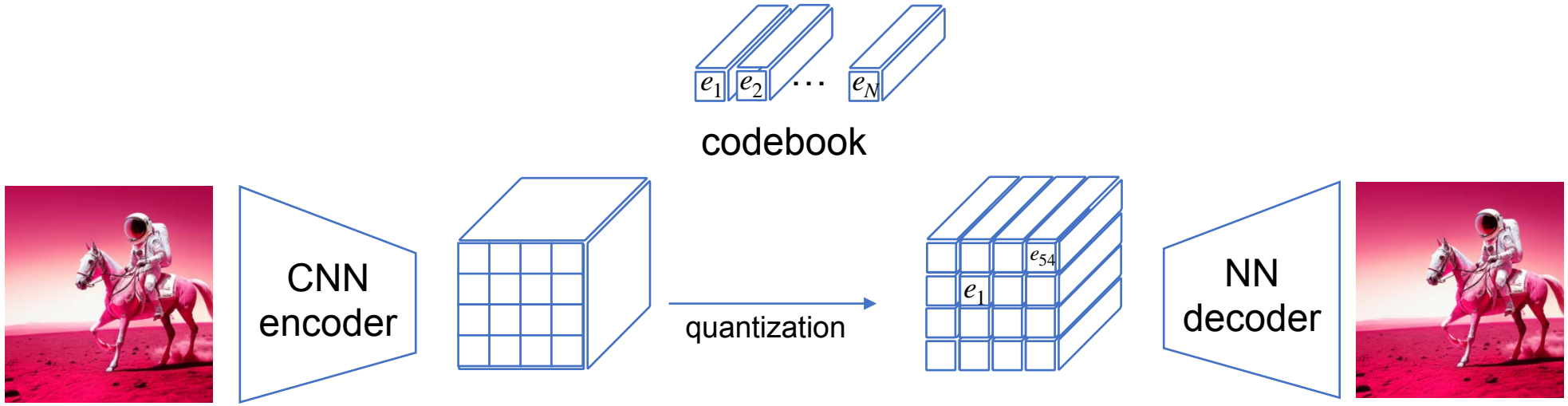
- Since the problem that VAEs are trying to solve is very complex, they can even start generating images completely ignoring the prior (i.e., the input image), which is called posterior collapse.

- Many real-world objects and object-categories are inherently discrete and/or disjoint (e.g., nose / eye, MNIST digits). Interpolating between them is contra-intuitive for image generation.

- VAEs however model that interpolation between these is inherently possible and sensible.

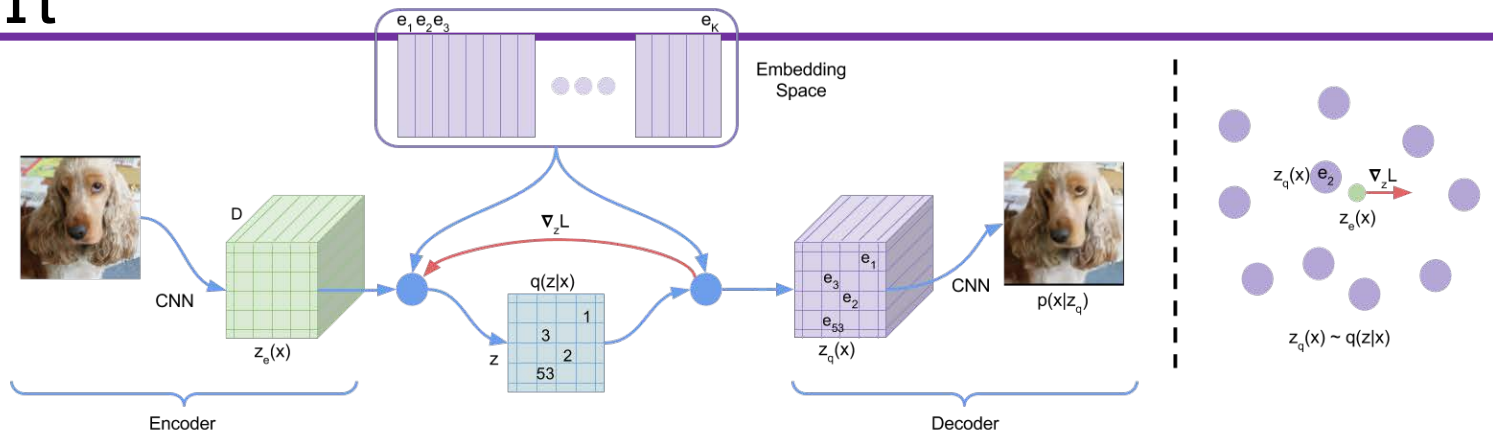
- Modeling generation from discrete variables is also much easier (since only discrete permutations are possible).

VQ-VAE: Overview



- During inference, the encoded feature vector is fed into a codebook lookup.
- This replaces the vector with the nearest vector found in the codebook.
- Note that in practice this is done for each (x,y) position in the latent space. Thus, each position will be represented by a codebook vector.
- This is then fed to the decoder.

Vector-Quantized Variational Autoencoders: Quantization



- In VQ-VAEs, the latent space is being quantized using a clustering similar to kmeans.
- This means, that we replace a continuous latent space $z_e(x)$ by a K-way categorical one.

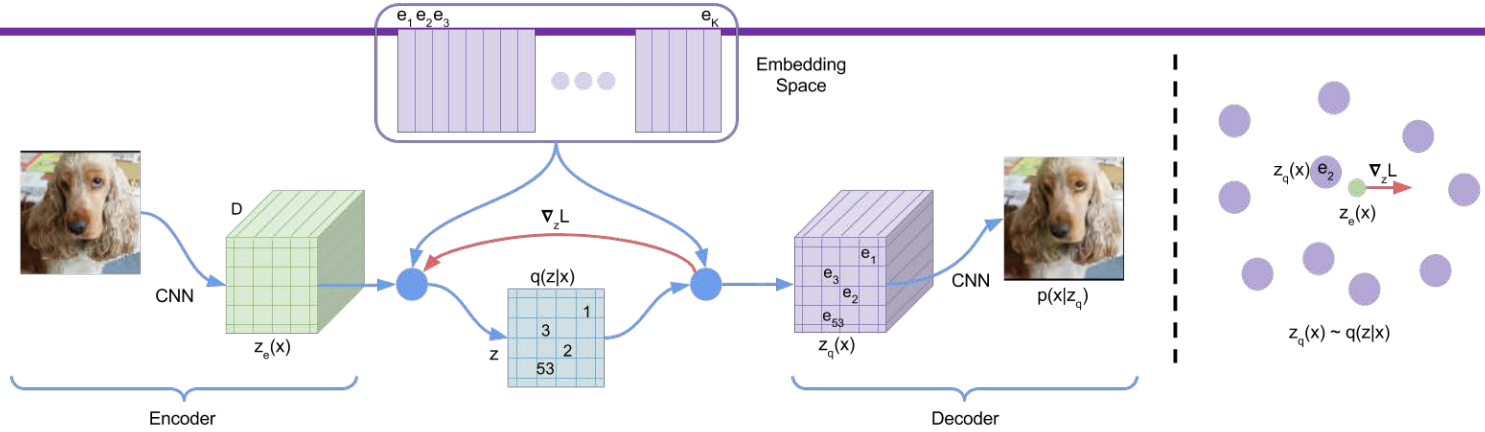
$$z_q(x) = e_k \text{ where } k = \arg\min_j \|z_e(x) - e_j\|_2$$

Codebook vectors

L2-Distance

- The latent space is divided this way into K clusters, represented by a centroid vector that will be learnt (codebook vector)

Vector-Quantized Variational Autoencoders: Training



- Now the vector quantizer codebook does not receive any gradients from the backwards pass, so the loss needs to account for this specifically (L2 loss as vector quantization objective).
- The combined loss is:

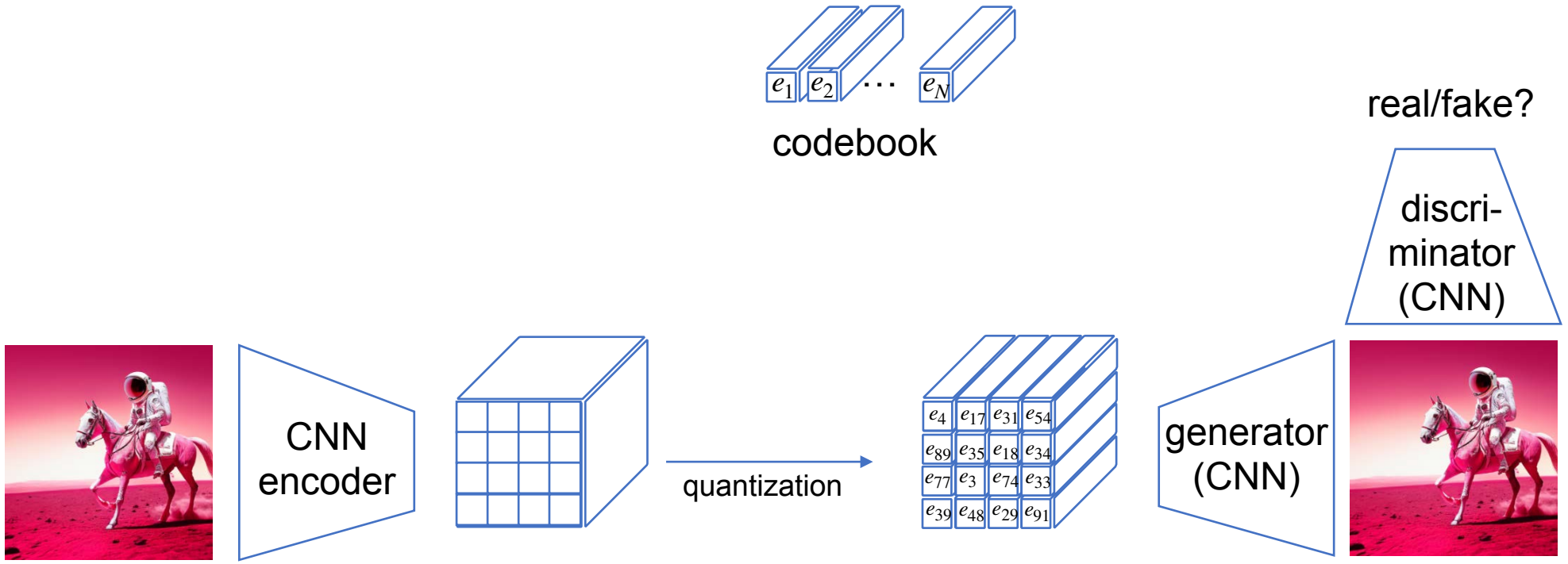
$$L = L_{\text{recon}} + \|\text{sg}\left[z_e(x)\right] - e\|_2^2 + \beta\|z_e(x) - sg\left[e\right]\|_2^2$$

reconstruction loss

Vector Quantization objective

„commitment loss“, ensuring that the embeddings don’t grow indefinitely (could happen, not ensured by VQ)

Vector-Quantized Generative Adversarial Networks (VQGAN, Esser et al., 2021)



- In VQ-GANs, we extend the VQ-VAE approach by using a discriminator (and a discriminator loss) to enhance variability and realness of the images.
- In the typical GAN sense, we then call the decoder of the vector-quantized latent space the generator.

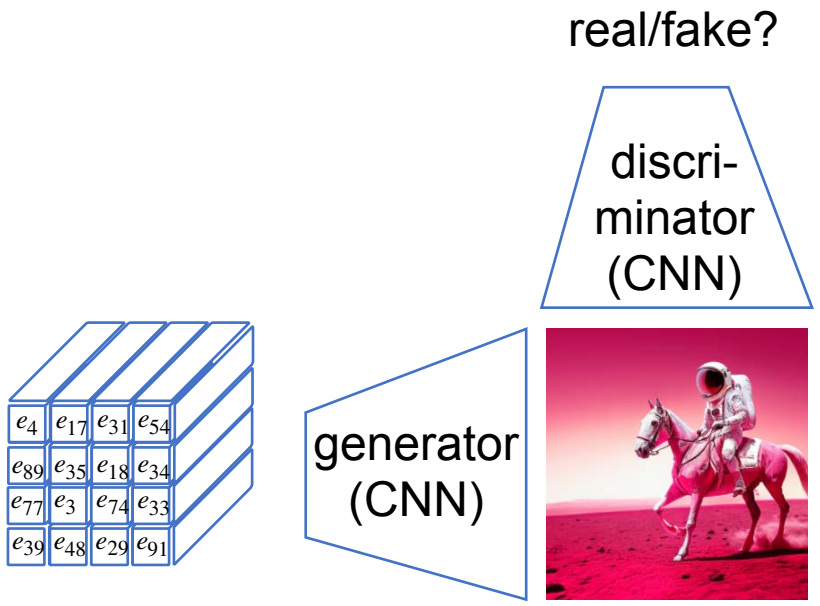
https://arxiv.org/abs/2012.09841

Vector-Quantized Generative Adversarial Networks (VQGAN, Esser et al., 2021)

- If we want to use this architecture for image generation, we have to find out, which composition of image tokens (codebook entries) makes sense.

- Clearly, the discriminator can help here, but since the generator uses the tokens directly as source of information, it would not be wise to have the generator correct image composition from random tokens.

- Hence, the question remains: How can we ensure the latent tokens to be meaningful (to decode to a meaningful image composition)?

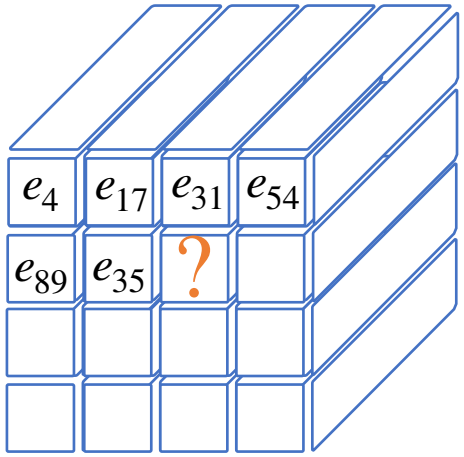


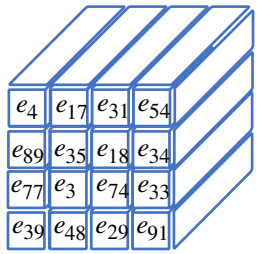
https://arxiv.org/abs/2012.09841

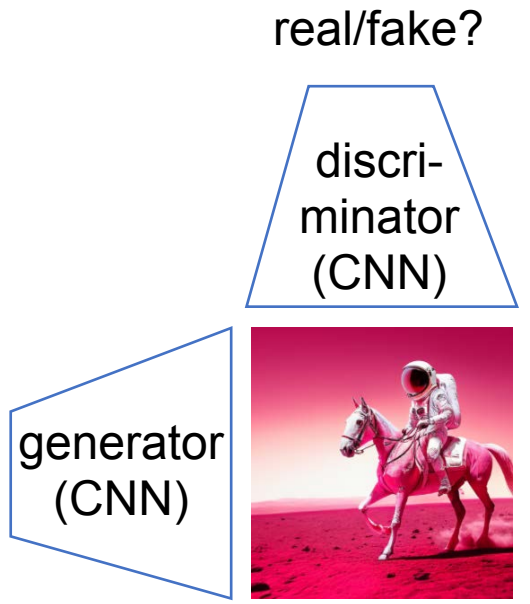
Vector-Quantized Generative Adversarial Networks (VQGAN, Esser et al., 2021)

Hochschule Flensburg
University of Applied Sciences

- Do we have a tool to learn sensible image composition?



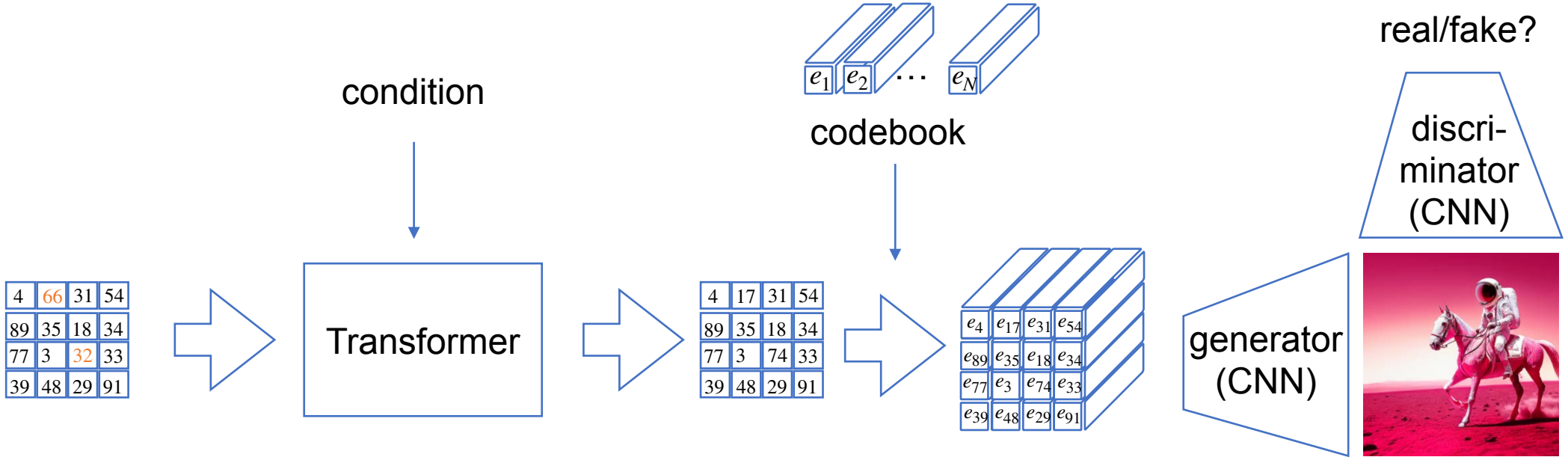




- Yes, the transformer! It can predict a suitable next token, a missing token, or even if a non-sensible token needs to be replaced!
- For this, the transformer itself is trained to predict a sensible token given the current context.
- The transformer can predict either masked tokens or noisy tokens (in which case, it also learns to differentiate sensible from noisy tokens)

https://arxiv.org/abs/2012.09841

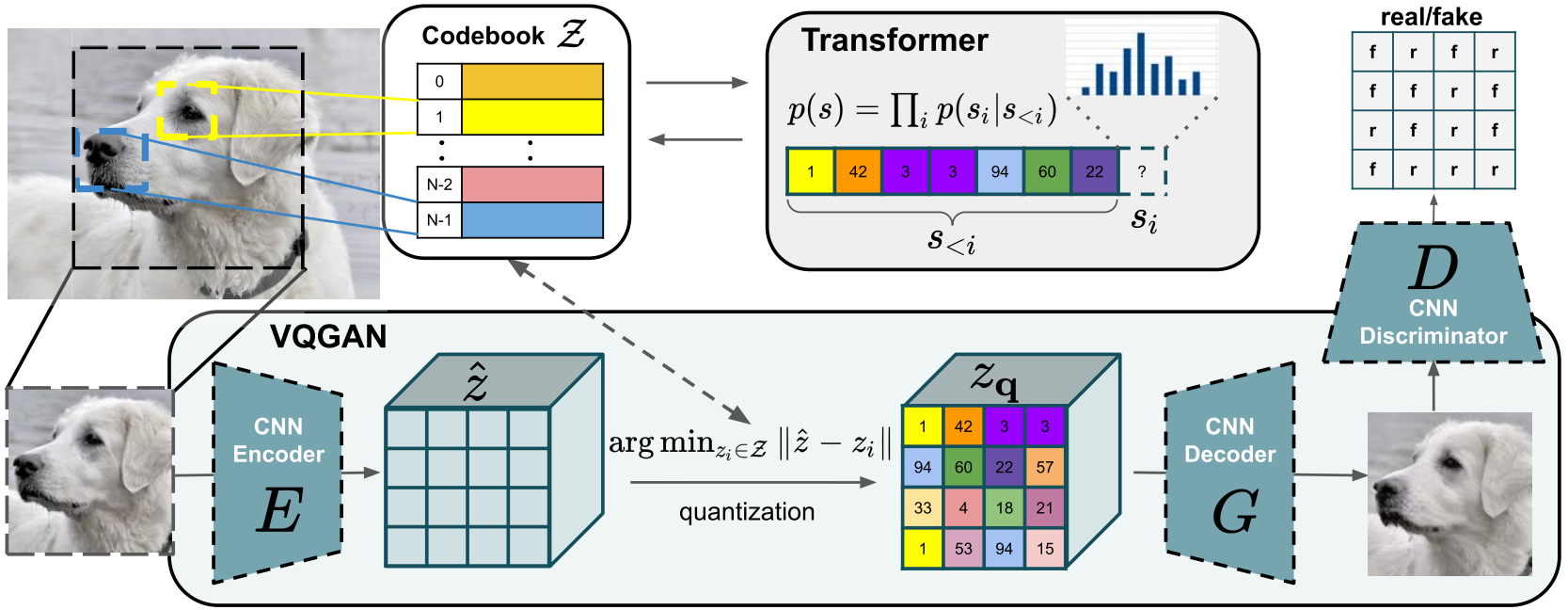
Vector-Quantized Generative Adversarial Networks (VQGAN, Esser et al., 2021)



- The transformer is a universal tool to learn sequence to sequence relations.
- It is made for making sense of tokens. Thus, a VQ latent space is ideally suited for transformers.
- The transformer is trained using a binary cross-entropy classification loss to predict the correct token of the image.

https://arxiv.org/abs/2012.09841

Vector-Quantized Generative Adversarial Networks (VQGAN, Esser et al., 2021)



- The VQGAN pipeline thus combines transformers (for creating sensible tokens that composite an image), vector quantization (to generate a discrete token-space) with a GAN-setup (discriminator and generator).
- The quantization allows to reduce the cardinality of the input space of the transformer, which is (so the title of the paper) taming the transformer.

https://arxiv.org/abs/2012.09841

Sliding window approach for image generation (in VQGAN, Esser et al., 2021)

Hochschule Flensburg
University of Applied Sciences

- The attention mechanism of size s limits the size of the generated images ($w \cdot h \leq s$).
- A strong downsampling by the encoder/decoder reduces image quality.

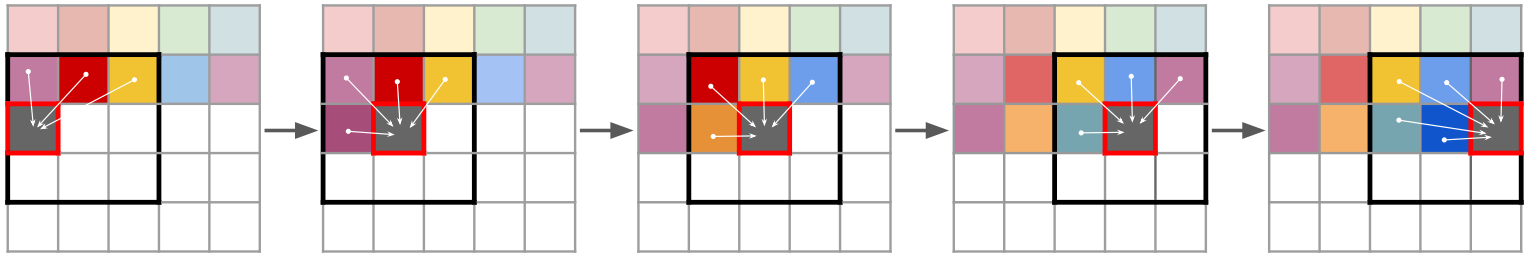


Figure 3. Sliding attention window.

- Hence, the authors propose to use a sliding window attention mechanism that allows image composition with a limited context.

https://arxiv.org/abs/2012.09841

Conditional Image Synthesis

Hochschule Flensburg
University of Applied Sciences

- In many image synthesis tasks, the user may want to add a condition to the image synthesis task.
- A condition might be a label (class), an image a segmentation, or even something completely different.

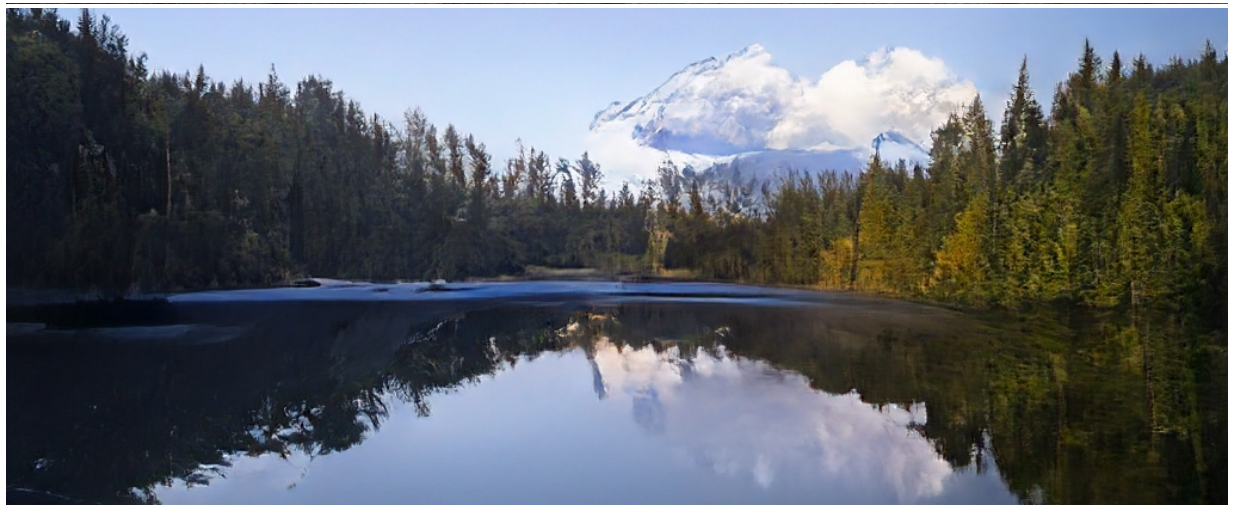
condition generated images

- The Transformer learns to utilize the information given by the condition, as it facilitates the generation task.

https://arxiv.org/abs/2012.09841

Generation of images of various size

- The sliding window approach of the image synthesis in the transformer in VQGAN also allows to perform image generation at arbitrary aspect ratios.

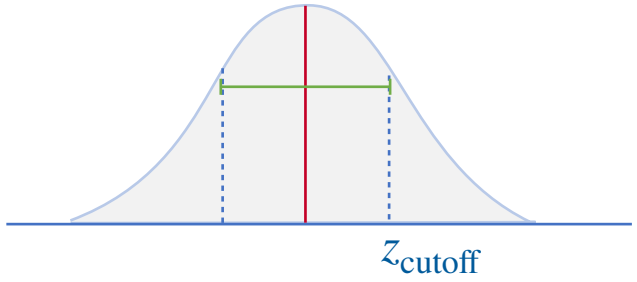


https://arxiv.org/abs/2012.09841

BigGAN (Brock et al., 2019)

- But also traditional GANs can benefit from scale, as shown by Brock et al. at ICLR 2019.
- They furthermore employ a regularization trick where they limit the outputs of the z space distribution (truncation trick) during inference.

Figure 6: Samples generated by our BigGAN model at 512×512 resolution.



- This leads to the model showing significantly less variance in the image, and producing images more similar to the original distribution.

https://arxiv.org/abs/1809.11096

Summary

- Images of high fidelity can be generated using autoencoder-style architectures with special tricks.
- Variational autoencoders regularize the latent space by modeling it probabilistically.
- Vector-quantization reduces the cardinality of the latent space.
- Vector-quantized variational autoencoders (VQ-VAE) additionally reduce the cardinality by also using spatial encoding, allowing for high quality images to be reconstructed.
- VQ-GANs employ adversarial losses to enhance image quality, but also use a transformer to generate semantically meaningful image composition.

Hochschule Flensburg
University of Applied Sciences

Test your knowledge

- How can an autoencoder be used for image generation?
- Why is a typical (non-variational) autoencoder insufficient for this task?
- Which properties to we expect from the latent space of an autoencoder so that we can use it for image generation?
- What is the main idea of a variational auto-encoder? Why is it more suitable for image generation?
- Explain the loss of the VAE. Which two terms does it consist of and why are both needed?
- Explain the role of the codebook in VQ-VAEs.
- Why is it beneficial to quantize the latent space in VQ-VAEs?
- The VQ-GAN paper is about „taming transformers“ for image generation - explain, how the transformer has been tamed in this application.
- Why is the quantization of the latent space beneficial in the VQ-GAN approach?

Hochschule Flensburg
University of Applied Sciences

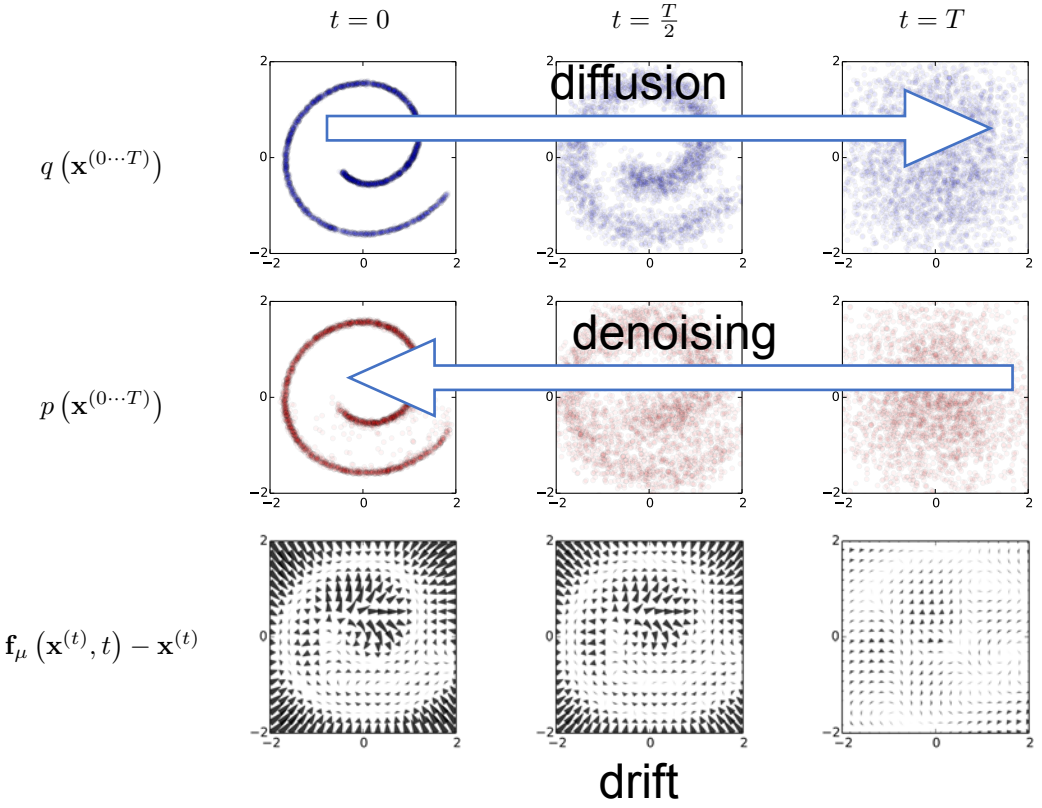
Hochschule
Flensburg
University of
Applied Sciences

Denoising-based Image Generation

Recap: Data denoising for generation

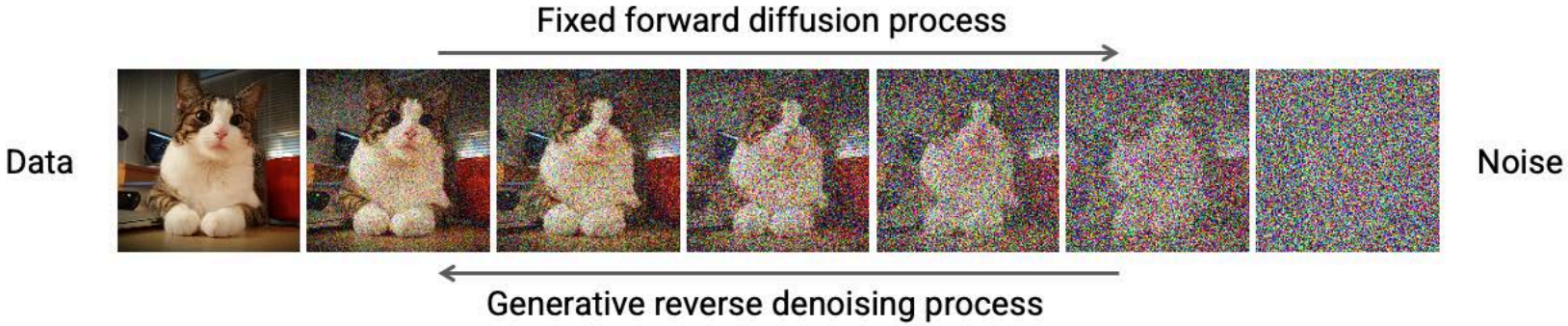
Hochschule Flensburg
University of Applied Sciences

- The idea of diffusion-based generation (and training denoising diffusion probabilistic models, DDPM) is (Sohl-Dickstein et al., 2015):
  […] to systematically and slowly destroy structure in a data distribution through an iterative forward diffusion process. We then learn a reverse diffusion process that restores structure in data, yielding a highly flexible and tractable generative model of the data.



https://arxiv.org/pdf/1503.03585.pdf

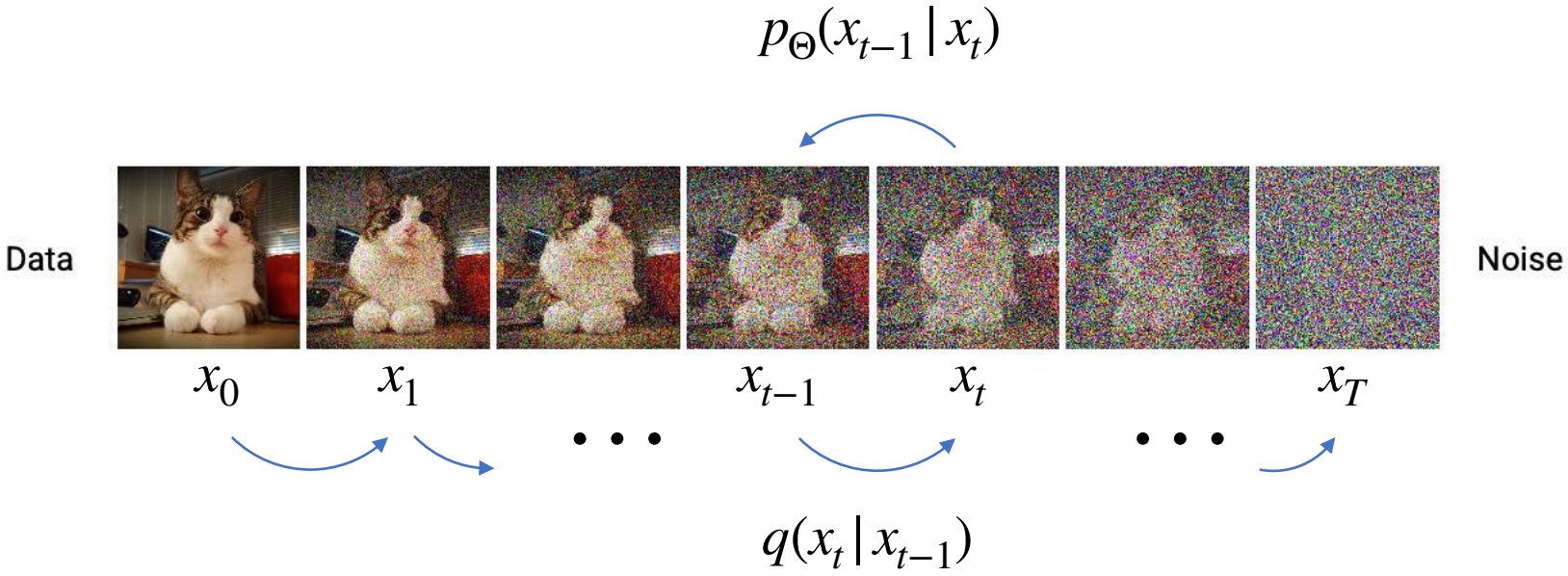
Recap: Using denoising as an image generation task



- We can formulate the image generation task as an iterative denoising task.
- The goal is to feed a model with pure noise and then gradually remove the noise until we have a clean image.
- The inverse is a stepwise process in which information dissipates - it's getting diffused.
- We can think of multiple possible permutations that would be candidates for this diffusion process - adding random noise is the simplest of them (and one with nice theoretical properties).
- Each step in this process is much simpler than directly predicting a image of the original distribution from the completely noised image.

https://cvpr2022-tutorial-diffusion-models.github.io/

Recap: Denoising and diffusion



- We thus define two processes in the scope of denoising and diffusion (noising):
  - The diffusion (forward) process: $q(x_t \mid x_{t-1})$
  - The denoising (reverse) process: $p_\Theta(x_{t-1} \mid x_t)$
- The denoising process will be carried out by a model with parameters Θ.

Noising schedule

$\bar{\alpha}_0 = 1$ $\bar{\alpha}_T = 0$

$q(x_t \mid x_{t-1}, \beta_t)$

- We can accumulate the β_t to yield an individual denoising level $\bar{\alpha}_t$ for each time step t:

$$\bar{\alpha}_t = \prod_{k=0}^{t} (1 - \beta_k)$$

- This means that we can now reformulate the noising process in a non-iterative closed form:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$$

- We call the distribution of values β_t the noising schedule for the process. It is designed, so that $\bar{\alpha}_0 = 1$ and $\bar{\alpha}_T \to 0$, i.e. $x_T = \mathcal{N}(0,1)$.

Why this works

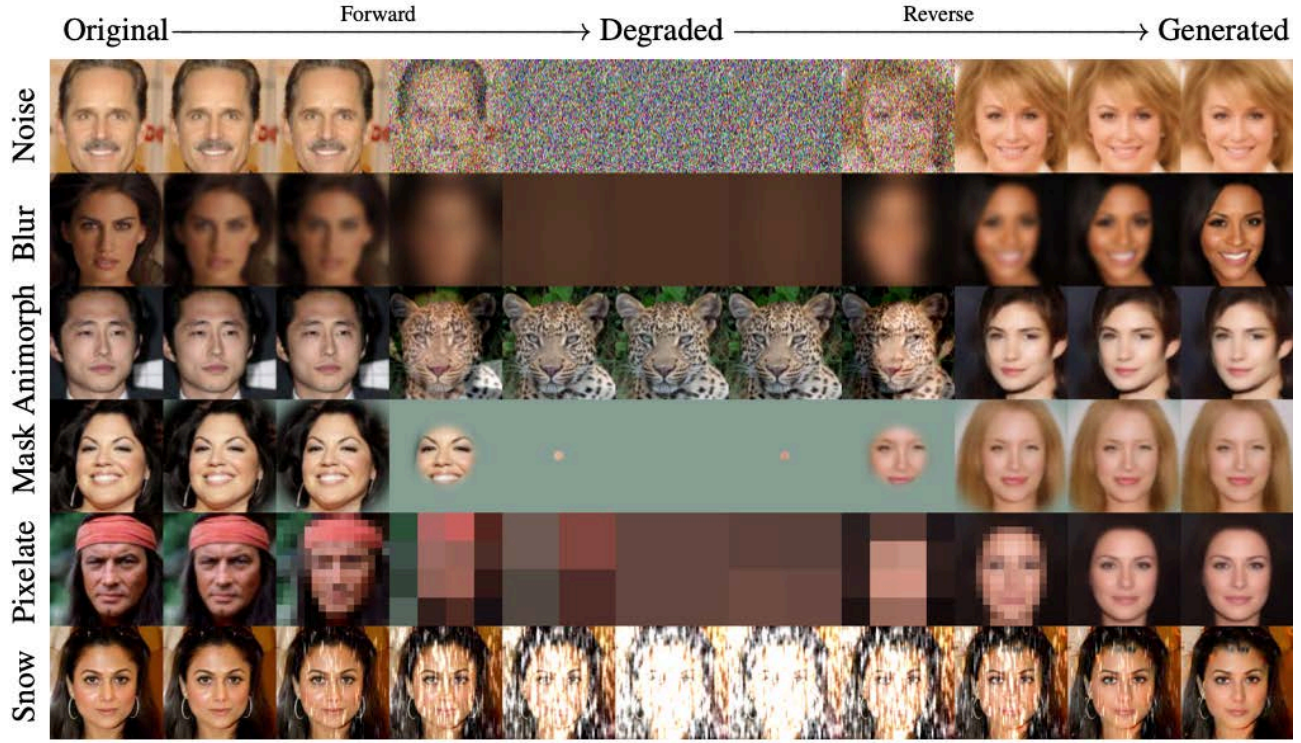
- The key idea in denoising diffusion probabilistic models (DDPMs) is that a model has learnt to invert the step of diffusion
  - for mildly noisy images
  - strongly noisy images
  - all the way until a pure noise with known distribution

- So if this is true, then we can sample from this known random distribution to start the inference process.

What about other pertubations?

Hochschule Flensburg
University of Applied Sciences

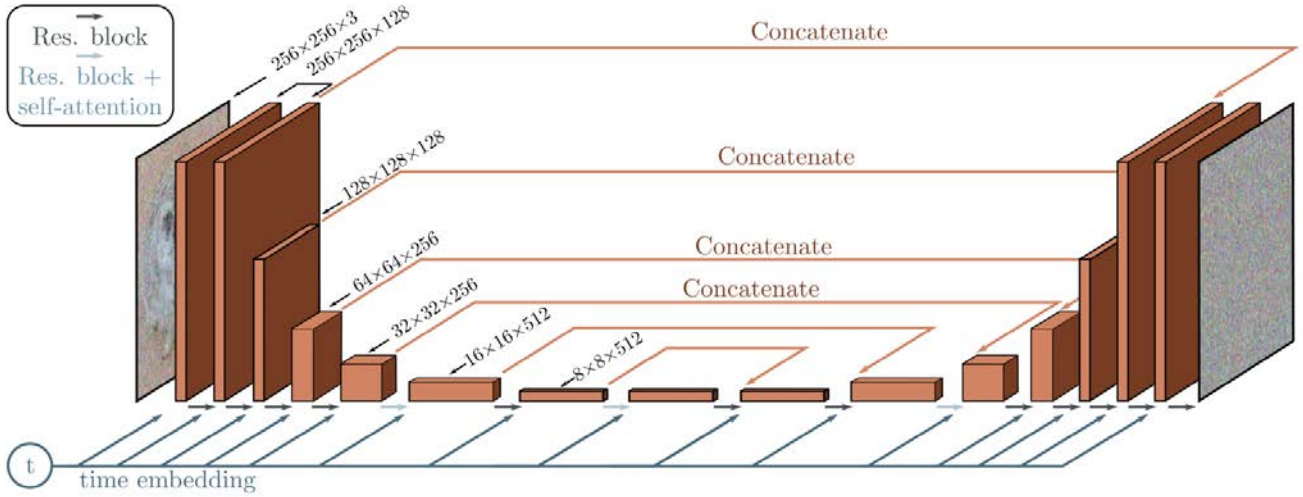
- Besides noising an image, other pertubations that can be calculated forward and reversed by a model could be also used.
- As shown by Bansal et al., 2022, this can include operations such as blurring, pixelation, masking, color desaturation, and down-sampling.
- Their work, known as Cold Diffusion, demonstrated that any invertible or approximately invertible degradation process can serve as the forward process.



https://arxiv.org/pdf/2208.09392

How the model predicts the noise

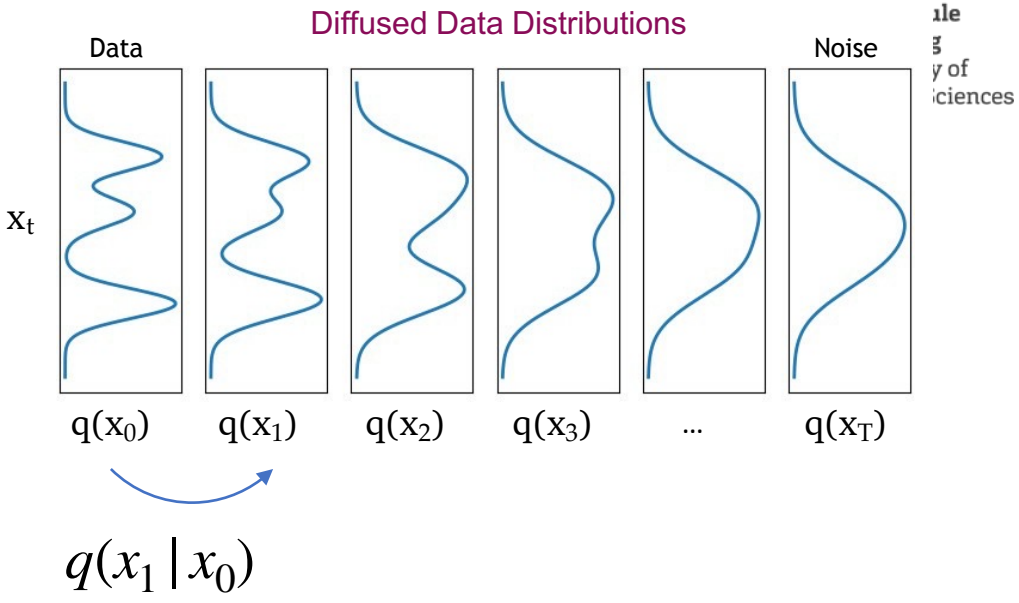
- The architecture used by the original DDPM model is a UNet-style encoder/decoder with self-attention at the bottleneck level.
- In order to help the model out in determining the magnitude of noise it will need to predict, it is informed about the timestep t using a sinusoidal embedding (like in transformers).



https://learnopencv.com/denoising-diffusion-probabilistic-models/

Data distribution from the image to the noise

- As stated, the fully noised image follows a normal distribution, while the original image follows its normal image data distribution.



- We can express the diffused data distribution $q(x_t)$ using the original data distribution $q(x_0)$ as:

$$q(x_t) = \int q(x_0) q(x_t | x_0) dx_0$$

- $q(x_t | x_0)$ is a diffusion kernel - effectively a Gaussian convolution.

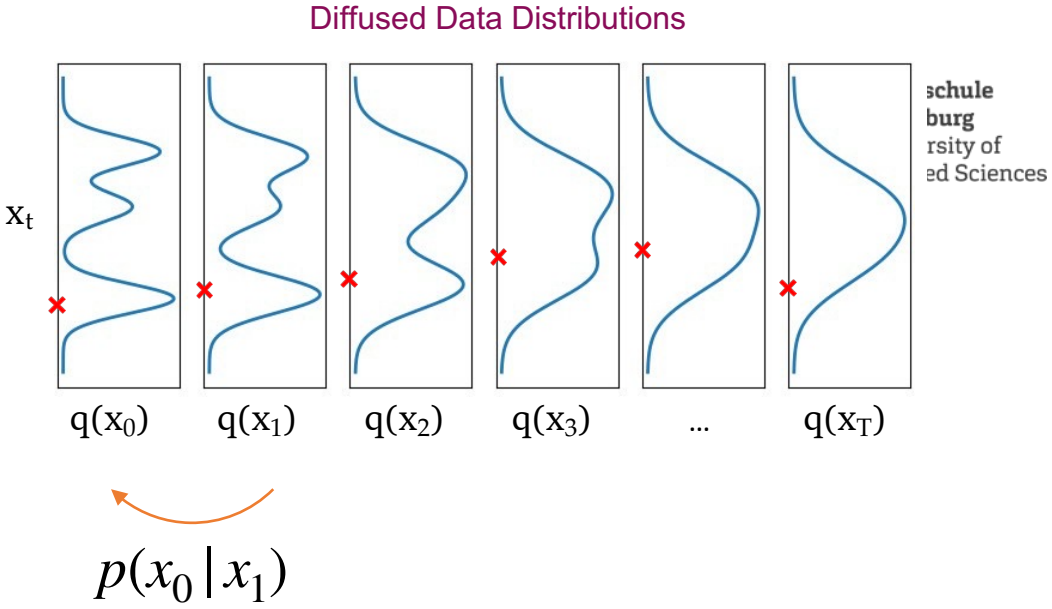
https://www.cs.unc.edu/~ronisen/teaching/fall_2022/pdf_lectures/lecture7-8_diffusion_model.pdf

Prediction of a less noisy image

- We have no analytic means of determining $p(x_{t-1} \mid x_t)$.

- But, similar to the forward process, we can model the change applied in diffusion using a normal distribution:

$$p_\Theta(x_{t-1}, x_t) = \mathcal{N}(x_{t-1}, \mu_\Theta(x_t, t), \sigma_t^2 I)$$



- The mean and standard deviation of the distribution are unknown, but can both be predicted using a neural network (in practice, we can set σ to a fixed value).
- This is very much like in VAEs, where we also used a network to predict the mean/std of a normal distribution.
- Note that all of these values (mean/std) are per coordinate of the image.

https://www.cs.unc.edu/~ronisen/teaching/fall_2022/pdf_lectures/lecture7-8_diffusion_model.pdf

Training a denoising model

- A DDPM can be trained using the following scheme proposed in the original paper:

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
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

predicted noise from noised image

noise, $\epsilon \sim \mathcal{N}(0, I)$

https://arxiv.org/pdf/2006.11239.pdf

Sampling from a DDPM

- The following algorithm describes image generation (sampling) from a denoising diffusion propabilistic model (DDPM):

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **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** $\mathbf{x}_0$

start with a random noise image
start at the fully noisy time step T,
then gradually approach t=1

predict the improved image by gradually removing the predicted noise ϵ_Θ

Summary: Diffusion-based image generation

- Besides auto-regressive image generation using transformers (such as in VQ-GAN), also diffusion-based approaches can yield high-quality generated images.

- The number of steps used for generating those images is, however, typically high (in the order of up to 1000). For inference, the number of steps can be, however, much smaller (typically 50).

- The noising schedule of the diffusion process has a significant impact on the number of diffusion steps and the image quality.

Hochschule
Flensburg
University of
Applied Sciences

Flow-based Models

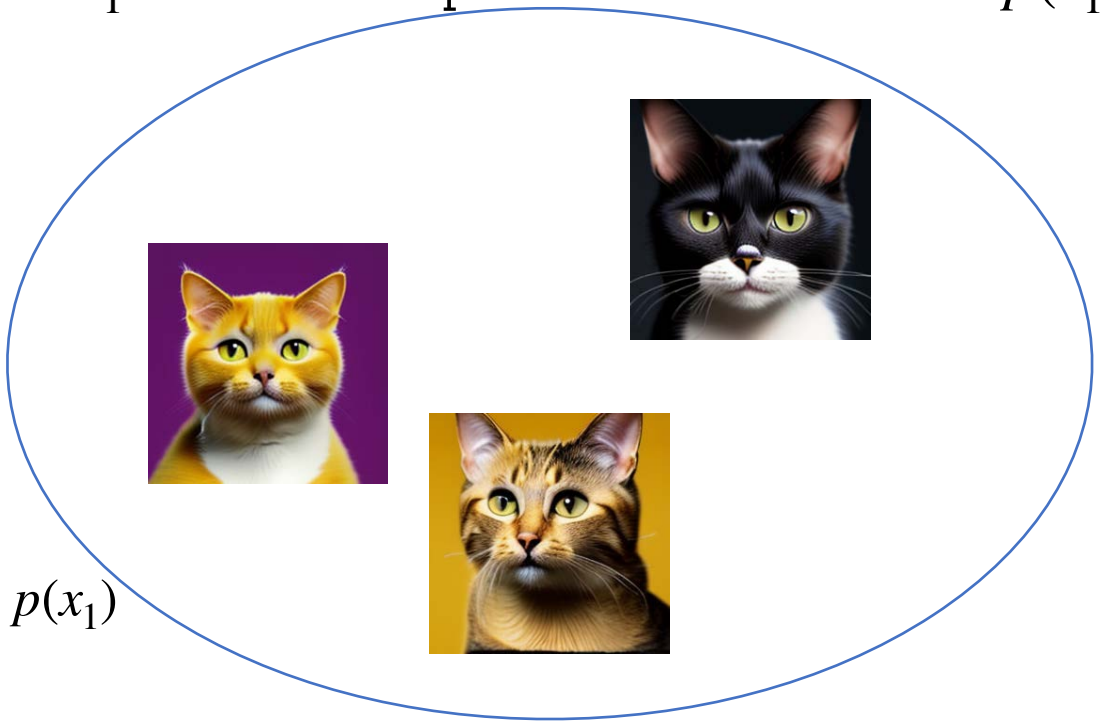
Translating noise into an image

- So, up to now we had:
  - GANs, that translate a random noise source into an image
  - Diffusion models, that also do that, but require many more steps
- But the training was a tiny bit difficult, you could say: indirect.
- Why don't we just learn the direct translation from the noise to the image?

Hochschule Flensburg
University of Applied Sciences

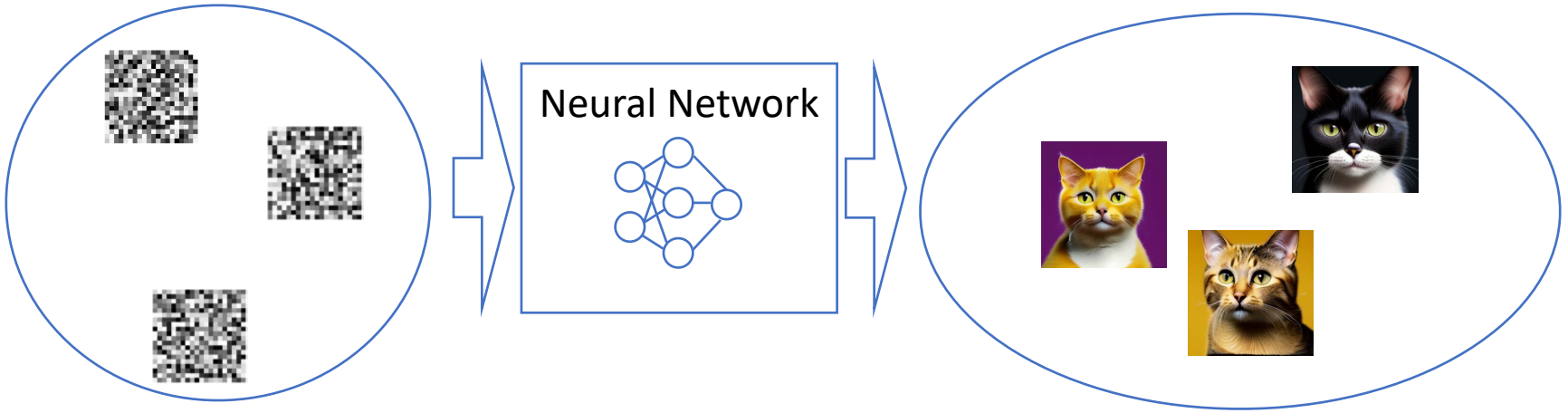
Flow Matching: Introduction

- Let's say we want to generate samples x_1 of cats.
- Ideally, we would want to sample from the distribution of all cat images $p(x_1)$.
- However, we don't know this distribution.
- But: We do have a lot of **examples** x_1 that are samples of the distribution $p(x_1)$.



Flow Matching: Idea

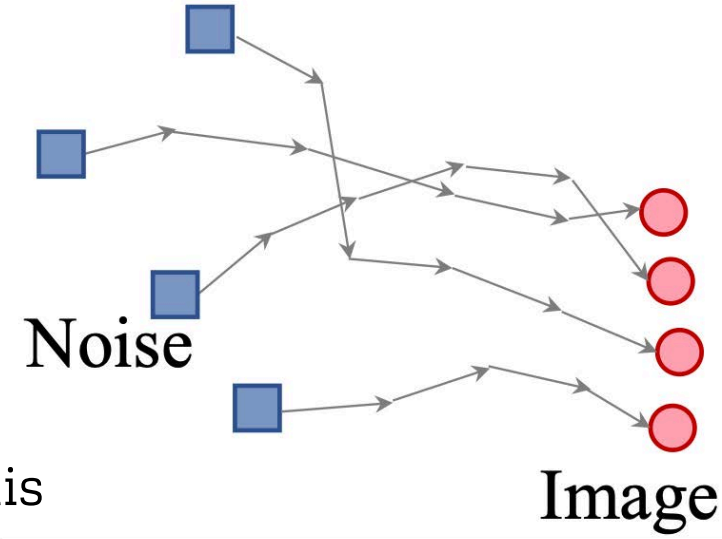
- We do **not know** the target distribution $p(x_1)$.
- But we do know a lot of other distributions, for instance, a Gaussian distribution $p(x_0) = \mathcal{N}(0,1)$ that we can easily sample from.
- Can we use a Neural Network to translate from:

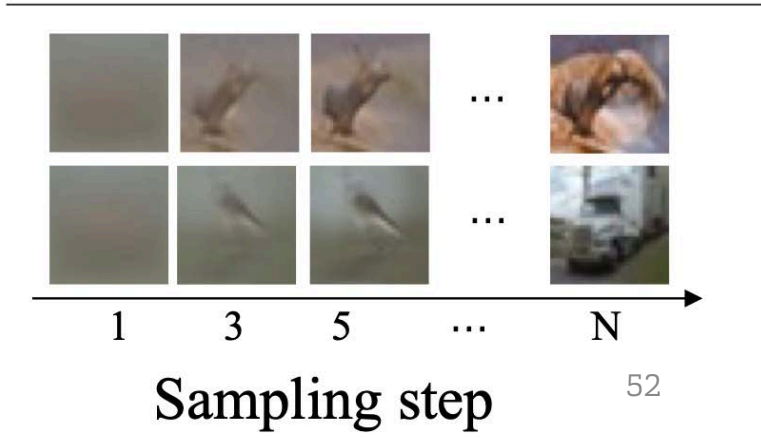


$p(x_0)$ to $p(x_1)$

Idea behind Flow-based Models

- What if we could learn a smooth, invertible path from noise to an image — and back again?
- Flow-based models do exactly that:
  they define a reversible transformation that gradually "flows" simple noise into complex data.
- Every step of the flow is mathematically invertible,
  so we can compute how likely an image is, not just generate it.
- Training is direct — we maximize the likelihood of real images under this transformation,
  no adversarial game, no noisy denoising loop.
- In other words:
  A flow-based model learns how to reshape randomness into structure.





https://arxiv.org/pdf/2311.16507v1

Flow-base Modeling

- So what we are actually interested in is to learn the function $\phi_t(x)$ that describes the morphing from the original distribution $p(x_0)$ to the target distribution $p(x_1)$.
- But what we are actually interested in is the **change** in $\phi_t(x)$ over time t.
- If we know this, we know the complete process.

Hochschule Flensburg
University of Applied Sciences

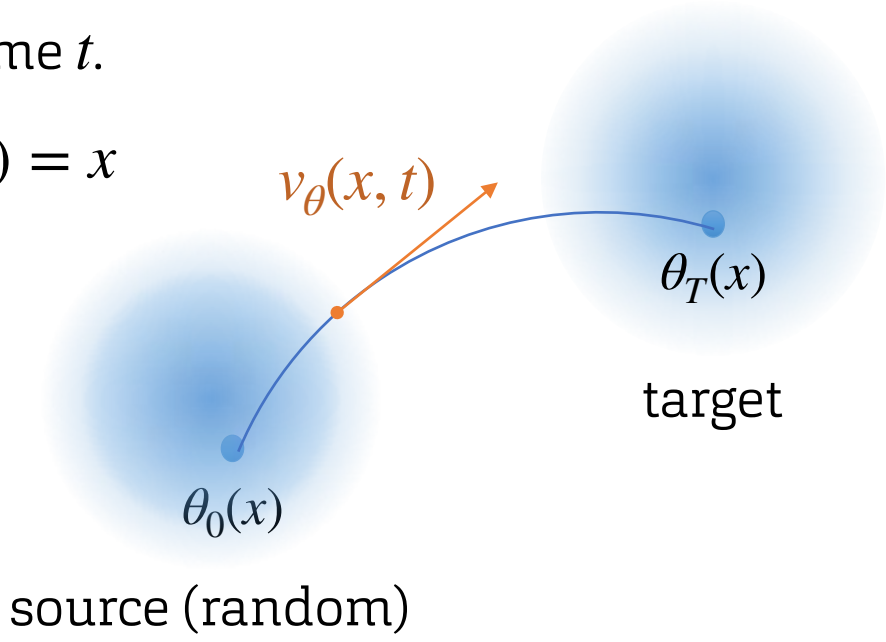
Flow Matching (Lipman et al., 2022)

Hochschule Flensburg
University of Applied Sciences

- We want to morph a random data distribution into a target data distribution, by means of a continuous morphing.
- We describe the morphing as a flow $\theta_t(x)$ per sample x over time t.
- For $t = 0$, the flow yields the original (random) sample x: $\theta_0(x) = x$
- Train a velocity field $v_\theta(x, t)$ so that integrating it over time transports samples from noise to data.

$$v_\theta(x, t) = \frac{\partial}{\partial t}\theta_t(x)$$

- The paths that we learn are invertible.

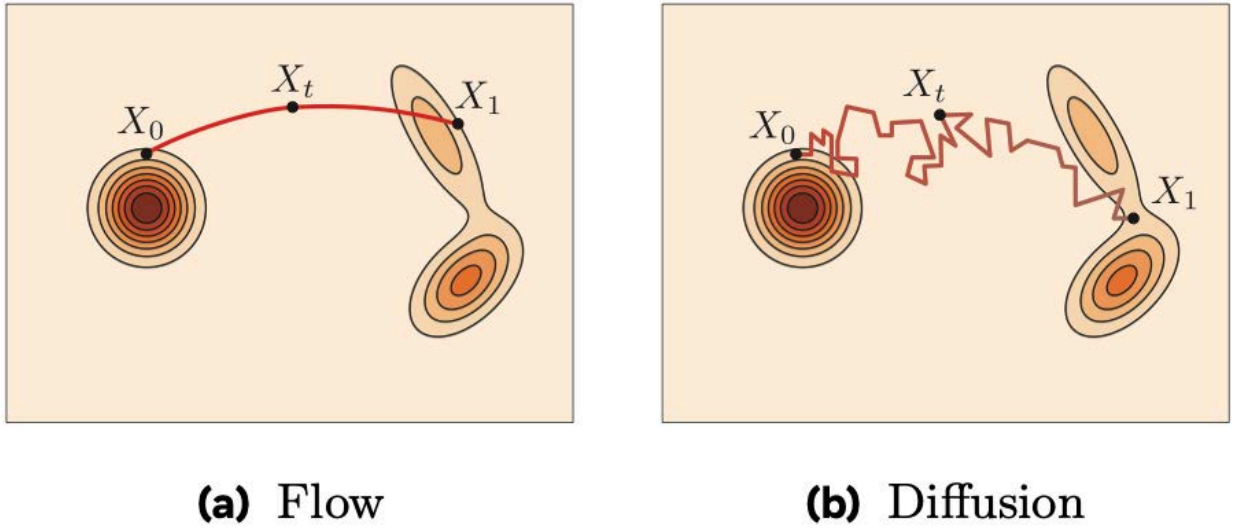


https://arxiv.org/pdf/2210.02747

Comparing Flow Matching and Diffusion

Hochschule Flensburg
University of Applied Sciences

- Both flow matching and diffusion have the same objective:
  - To learn a mapping from a simple base distribution (e.g., Gaussian noise) to a complex data distribution
    (e.g., real images, audio, etc.).
  - To enable efficient generation of new samples from this learned distribution.
- However,
  - Diffusion models learn this mapping implicitly by simulating a stochastic noising and denoising process (via an SDE).
  - Flow matching models learn it explicitly by modeling a deterministic continuous-time transformation (via an ODE).



(a) Flow **(b)** Diffusion

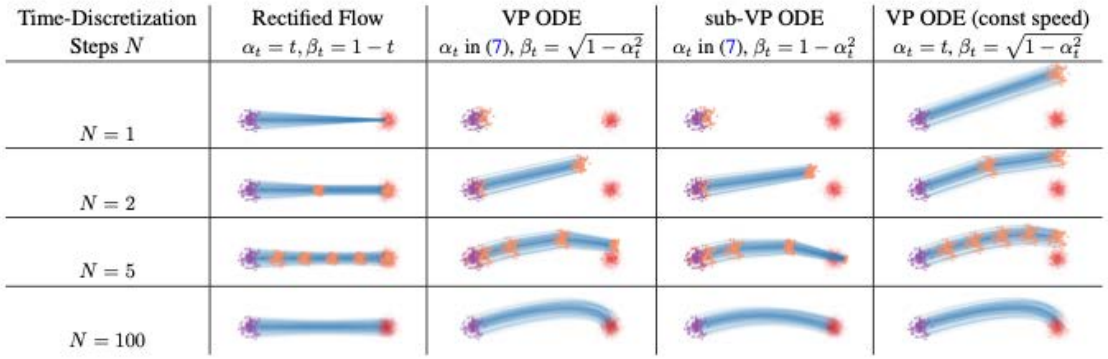
https://arxiv.org/pdf/2412.06264

Rectified Flows (Liu et al., 2022)

Hochschule Flensburg
University of Applied Sciences

- Rectified Flow is a simplified and efficient variant of flow matching that "rectifies" the paths between noise and data — turning potentially complex, nonlinear flows into straight-line trajectories in data space.
- The core idea is to make the flow a linear translation from the original data x_0 to the final image x_1, i.e., it becomes a linear interpolation:

$$\phi_t = tx_1 + (1-t)x_0$$



- If we calculate the time derivative of this $\partial X_t / \partial t$, we get:

$$\frac{\partial \phi_t}{\partial t} = x_1 - x_0$$

- Hence, in this, the velocity field becomes a **constant**.
- The trajectories are now straight, and the training becomes more stable.

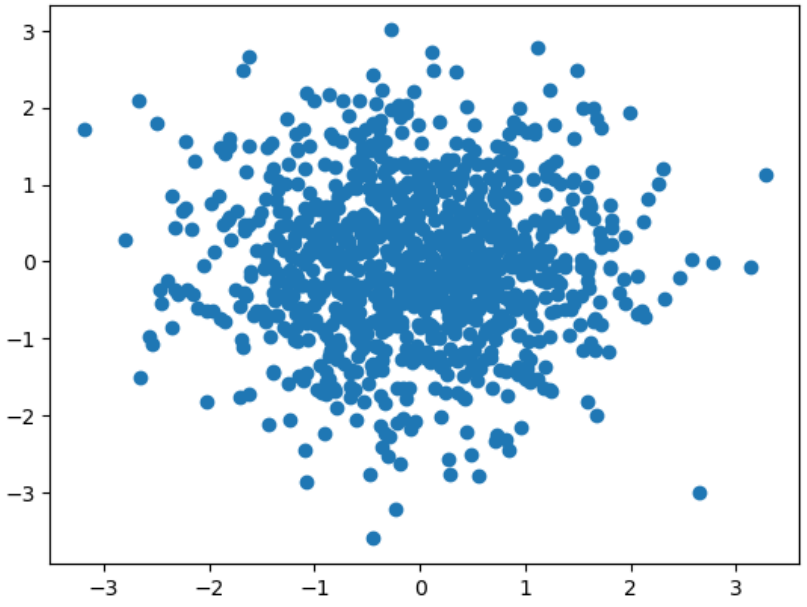
https://arxiv.org/pdf/2209.03003

Rectified Flows: Intuition

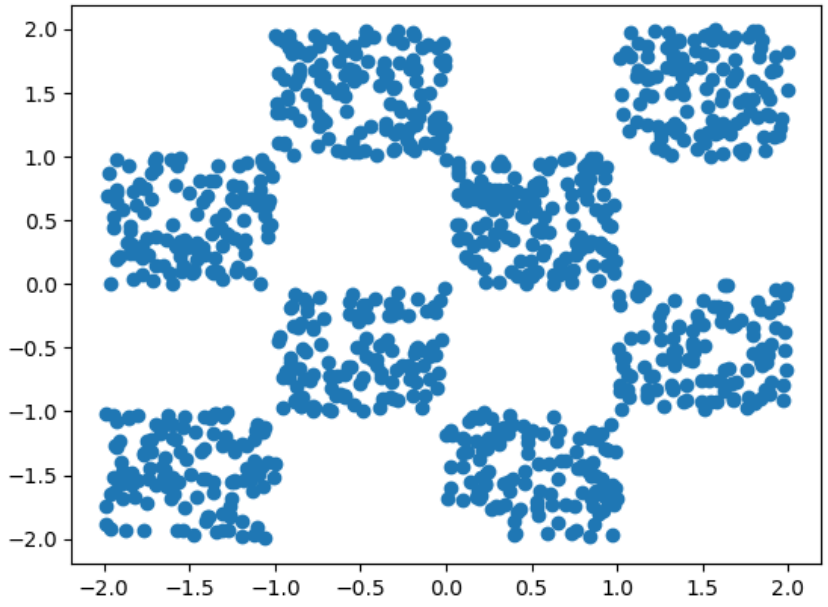
$\phi(t = 0) = x_0$ $\phi(t = 1) = x_1$

- The straight path between means that we have a direct linear interpolation between x_0 and x_1.
- This is much simpler than the original formulation, where we have some unknown flow that needs to fulfill certain criteria (invertible, continuous, etc..).

Rectified Flows: Graphical Example



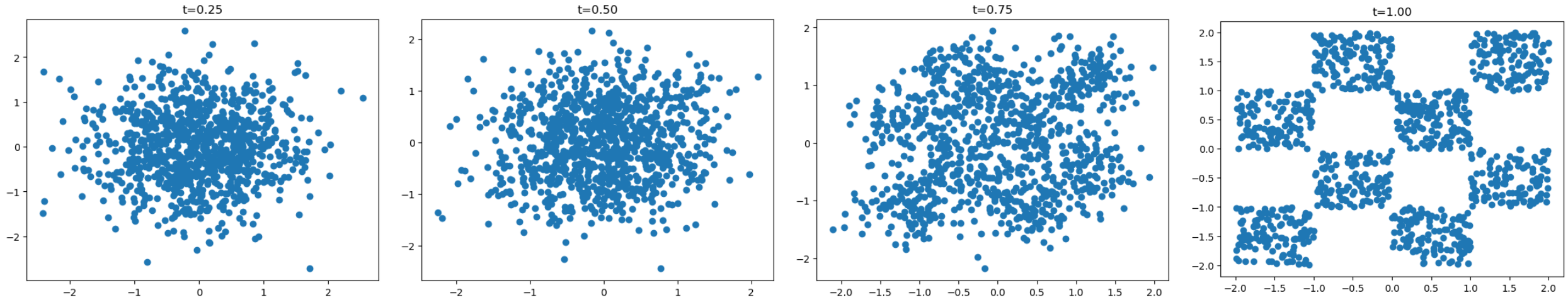
Source distribution



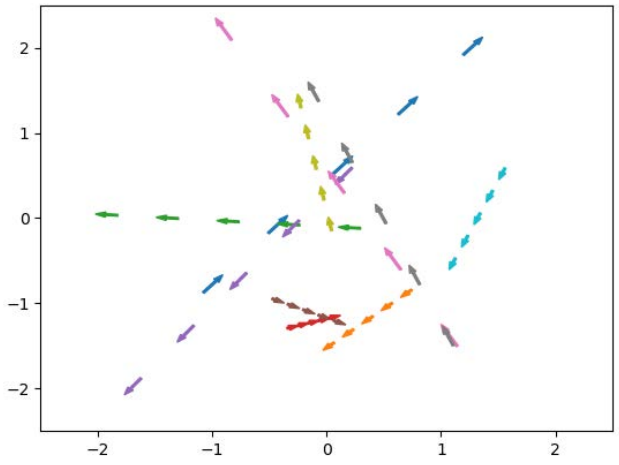
Target distribution

Rectified Flows: Graphical Example: Flow

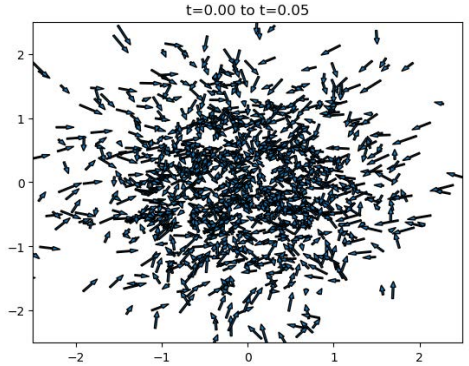
Hochschule Flensburg
University of Applied Sciences

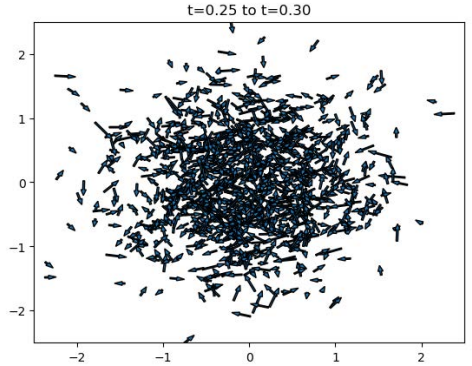


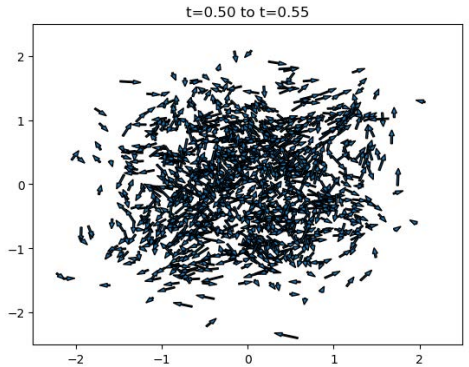
Rectified Flow: Time derivatives

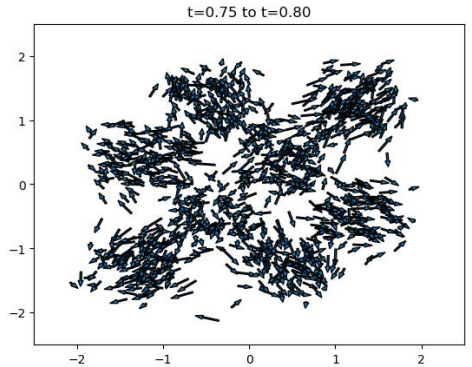


trajectory for selected points









It tells us for every point, in which direction we have to move it to go to the target distribution.

Note that in rectified flows, the derivative is **constant**.

Rectified Flows

- We can think of the derivative $\frac{\partial \phi_t}{\partial t}$ as a vector field, guiding each sample of the original distribution towards the target distribution.
- We don't know this vector field.
- But: We can use a neural network to learn it.

Rectified Flows

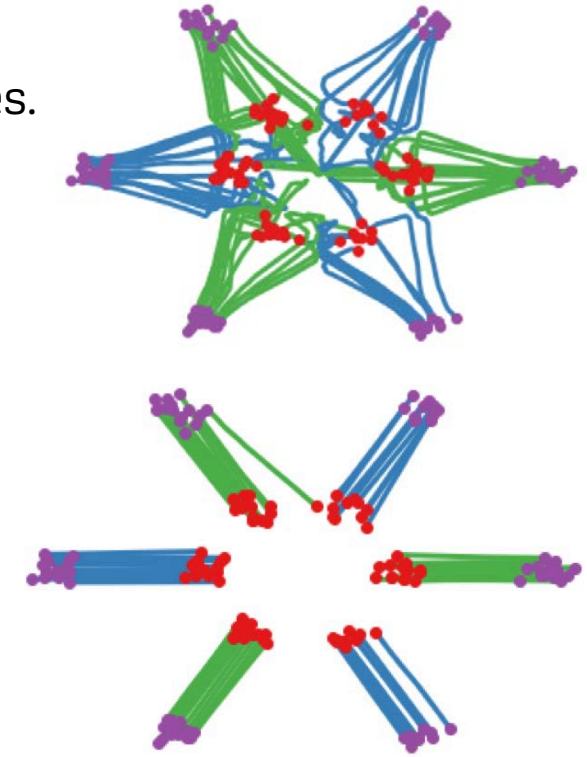
- Training:
  - Pairs noise and data samples, connects them with linear (rectified) trajectories
  - The model learns the velocity needed to move points from source to target
  - Simple and stable objective: no stochastic noise schedule required

- Sampling:
  - Starts from noise and integrates the learned flow to reach data space
  - Deterministic and fast, requiring **fewer steps** than diffusion models
  - Produces high-quality samples with smooth, consistent motion.

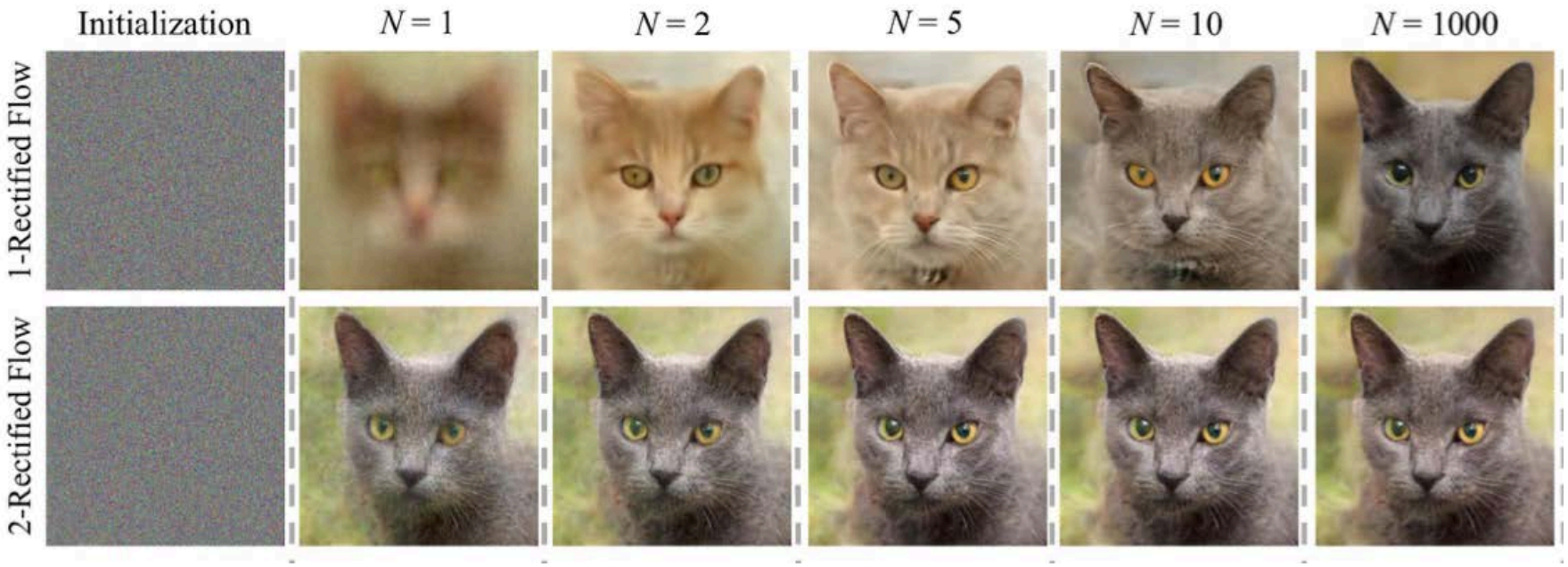
Rectified Flows: Reflow (Rectification)

Hochschule Flensburg
University of Applied Sciences

- In inference, we thus predict the next step in the flow from the previous one.
- However, these generated trajectories may deviate from the straight paths seen during training — meaning the model operates out of distribution relative to its training assumption of linear interpolation.
- To address this mismatch, we retrain the model on the flows it actually generates.
- This process is called Reflow (or Rectification).
- In each reflow iteration:
  - The current model generates new trajectories from noise to data.
  - These trajectories are treated as new training samples.
  - The model is retrained to better fit these realistic (curved) paths.
- As a result, the learned flow becomes progressively straighter and more consistent with its own inference dynamics.



Rectified Flows: Examples



- The more straightforward the trajectory, the easier it is to sample with a smaller number of steps.
- The straighter trajectory in the network with Reflow applied twice (2-rectified flow) allows for better images at a smaller amount of sampling steps.

Summary: Flow Matching and Rectified Flow

Hochschule Flensburg
University of Applied Sciences

- Both aim to transform noise into data via a learned continuous flow.
- Flow Matching learns a velocity field to model this transformation deterministically.
- In practice, learned trajectories can curve away from ideal straight paths.
- Rectified Flow assumes straight-line paths between noise and data during training.
- Reflow (Rectification) retrains the model on its own generated flows to correct mismatches.
- Each reflow iteration (1-rectified, 2-rectified) makes trajectories straighter and sampling faster.
- Overall: Rectified Flow = simpler, faster, and more self-consistent flow matching.

Hochschule
Flensburg
University of
Applied Sciences

Conditional Image Generation

From Label-Conditional to Text-Conditional

- We can now generate realistic images, conditioned on all kinds of labels (see VQ-GAN approach).

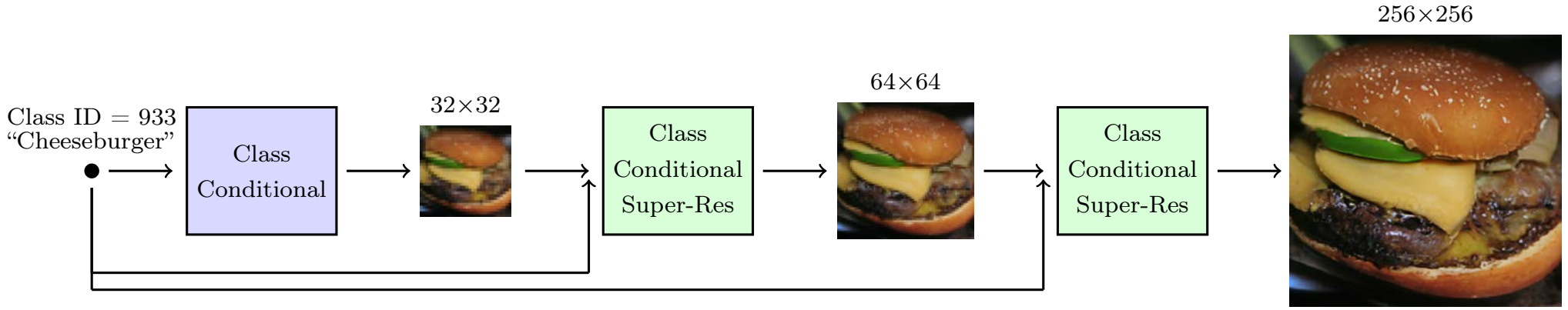
- But it would be even more interesting to combine other modalities in this generative process.

- So:
  - How can we create embeddings that encapsulate the semantics of text?
  - How can we combine these with the image generation process?
  - Which additional tricks do we need for high fidelity image generation?

Cascaded diffusion models for image generation (Ho et al., JMLR, 2022)

Hochschule Flensburg
University of Applied Sciences

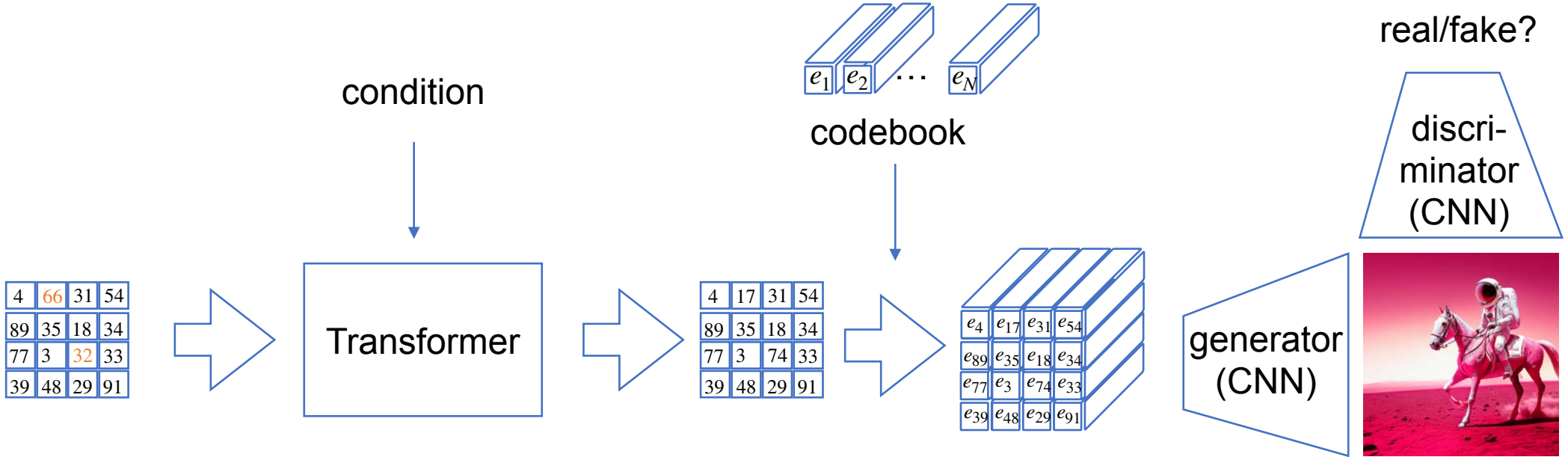
- Diffusion models can also be utilized in a cascaded way to achieve conditional super-resolution.



- In this architectural setup, each model learns to add more semantically meaningful details in a higher resolution.

https://arxiv.org/abs/2106.15282

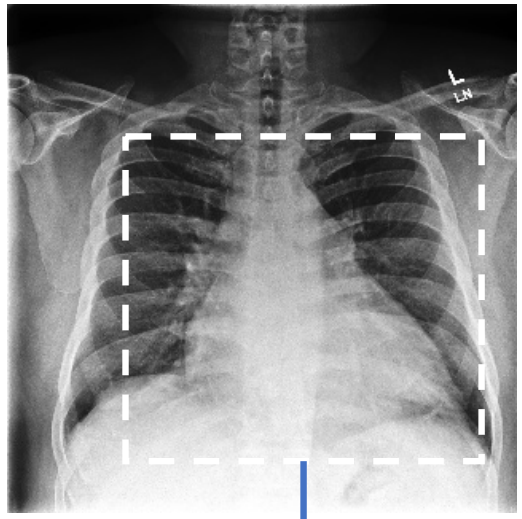
Vector-Quantized Generative Adversarial Networks (VQGAN, Esser et al., 2021)



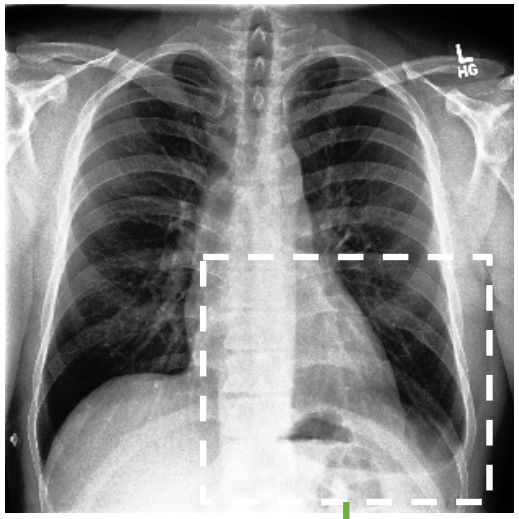
- In VQ-GANs we use transformers to incorporate the condition to denoise the latent space tokens.
- The condition (e.g., an embedded image label) helps the transformer to denoise the input tokens.

How could we use text as a condition?

Matching images and texts



Severe **cardiomegaly** is noted in the image with enlarged…

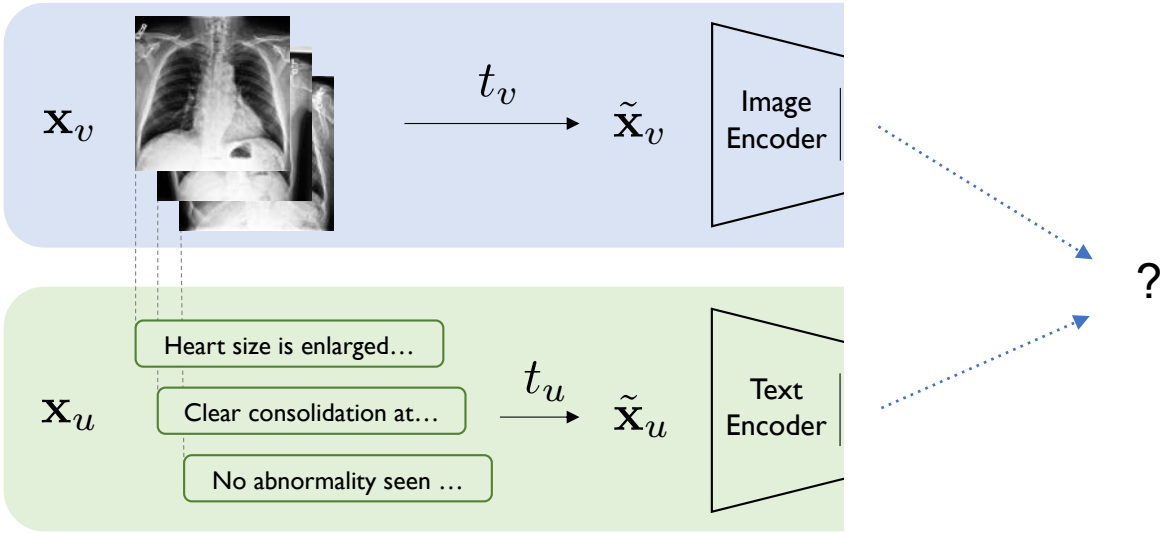


Radiograph shows **pleural effusion** in the right…

- Especially in medical imaging, we have an abudance of textual descriptions for medical images (e.g., x-rays and medical reports).
- However, we often lack the correspondence between the words and the image parts (e.g., bounding boxes or instance segmentations)

https://arxiv.org/pdf/2010.00747.pdf

Contrastive Learning of Medical Visual Representations from Paired Images and Text

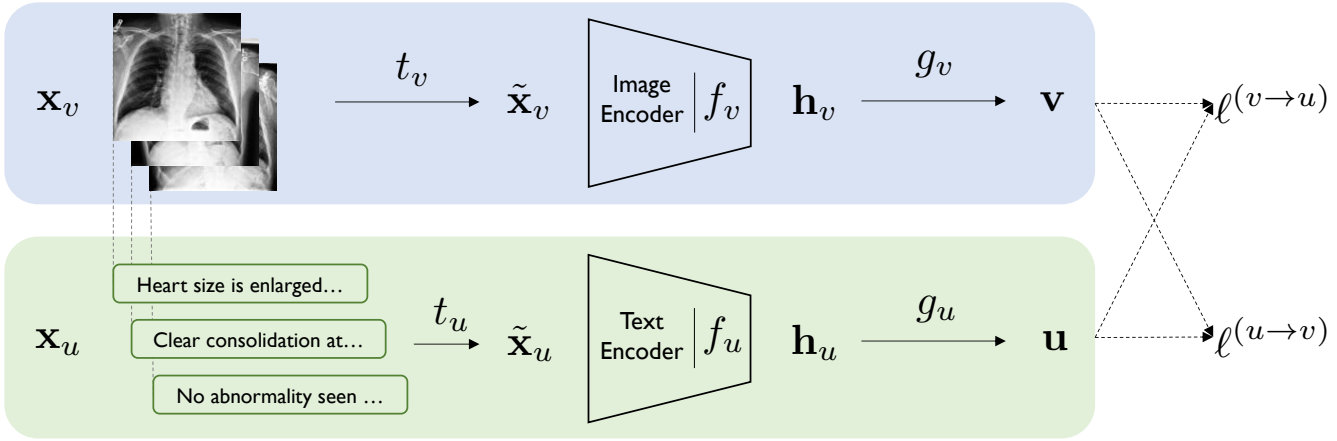


- If we combine a text-encoder and an image encoder, we can maybe find corresponding embeddings that are similar for both, the text and the image.
- Idea: Find corresponding pairs of text and image by minimizing their distance in latent space.
- This form of representation learning might hence be a very interesting pretext task for object recognition.

https://arxiv.org/abs/2010.00747

ConVIRT: Details

- The authors of ConVIRT (Zhang et al., 2020) hence used two encoders:
  - ResNet50 as image encoder
  - BERT as text encoder, in this case, ClinicalBERT (Alsentzer et al., 2019), an encoder that was trained on a medical text corpus.



- Behind each encoder, they use a non-linear projection head $g(\cdot)$
- They use a bidirectional contrastive loss function with temperature τ-scaling based on the cosine similarity $\langle u, v \rangle$ (like SimCLR):

$$L = -\log \frac{\exp\left(\langle v_i, u_i \rangle / \tau\right)}{\sum_{k=0}^{N} \exp\left(\langle v_i, u_k \rangle / \tau\right)}$$

https://arxiv.org/abs/2010.00747

ConVIRT: How well does it work?

(a) Linear classification

Method	RSNA (AUC) 1%	RSNA (AUC) 10%	RSNA (AUC) all	CheXpert (AUC) 1%	CheXpert (AUC) 10%	CheXpert (AUC) all	COVIDx (Accu.) 10%	COVIDx (Accu.) all	MURA (AUC) 1%	MURA (AUC) 10%	MURA (AUC) all
General initialization methods											
Random Init.	55.0	67.3	72.3	58.2	63.7	66.2	69.2	73.5	50.9	56.8	62.0
ImageNet Init.	82.8	85.4	86.9	75.7	79.7	81.0	83.7	88.6	63.8	74.1	79.0
In-domain initialization methods											
Caption-Transformer	84.8	87.5	89.5	77.2	82.6	83.9	80.0	89.0	66.5	76.3	81.8
Caption-LSTM	89.8	90.8	91.3	85.2	85.3	86.2	84.5	**91.7**	75.2	81.5	84.1
Contrastive-Binary-Loss	88.9	90.5	90.8	84.5	85.6	85.8	80.5	90.8	76.8	81.7	85.3
ConVIRT (Ours)	**90.7**	**91.7**	**92.1**	**85.9**	**86.8**	**87.3**	**85.9**	**91.7**	**81.2**	**85.1**	**87.6**

- The training was performed on the MIMIC-CXR database (217k image-text pairs) and on in in-house dataset consisting of 48k image-text-pairs.
- A linear classifier was trained from the pre-trained embeddings on the RSNA dataset, the CheXpert dataset and other datasets, with various degrees of data percentage used to investigate data efficiency.
- The results indicate that the pretraining extremely benefits the recognition task (especially compared to ImagNet pre-trained weights).

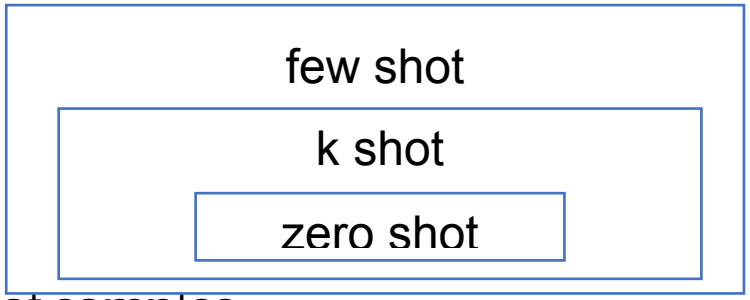
ConVIRT: Zero shot performance

Method	Image-Image Retrieval			Text-Image Retrieval		
	Prec@5	Prec@10	Prec@50	Prec@5	Prec@10	Prec@50
Random	12.5	12.5	12.5	12.5	12.5	12.5
ImageNet	14.8	14.4	15.0	–	–	–
In-domain initialization methods						
Caption-Transformer	29.8	28.0	23.0	–	–	–
Caption-LSTM	34.8	32.9	28.1	–	–	–
Contrastive-Binary-Loss	38.8	36.6	29.7	15.5	14.5	13.7
ConVIRT (Ours)	**45.0**	**42.9**	**35.7**	**60.0**	**57.5**	**48.8**
Fine-tuned						
ConVIRT + CheXpert Supervised	56.8	56.3	48.9	–	–	–

Table 2: Zero-shot image-image and text-image retrieval results on the CheXpert 8×200 datasets. *Random* shows results from a random guess; *ConVIRT + CheXpert Supervised* shows results from further fine-tuning the pretrained weights with supervised training data. Text-image retrieval results are not obtained for some methods due to the lack of text encoders.

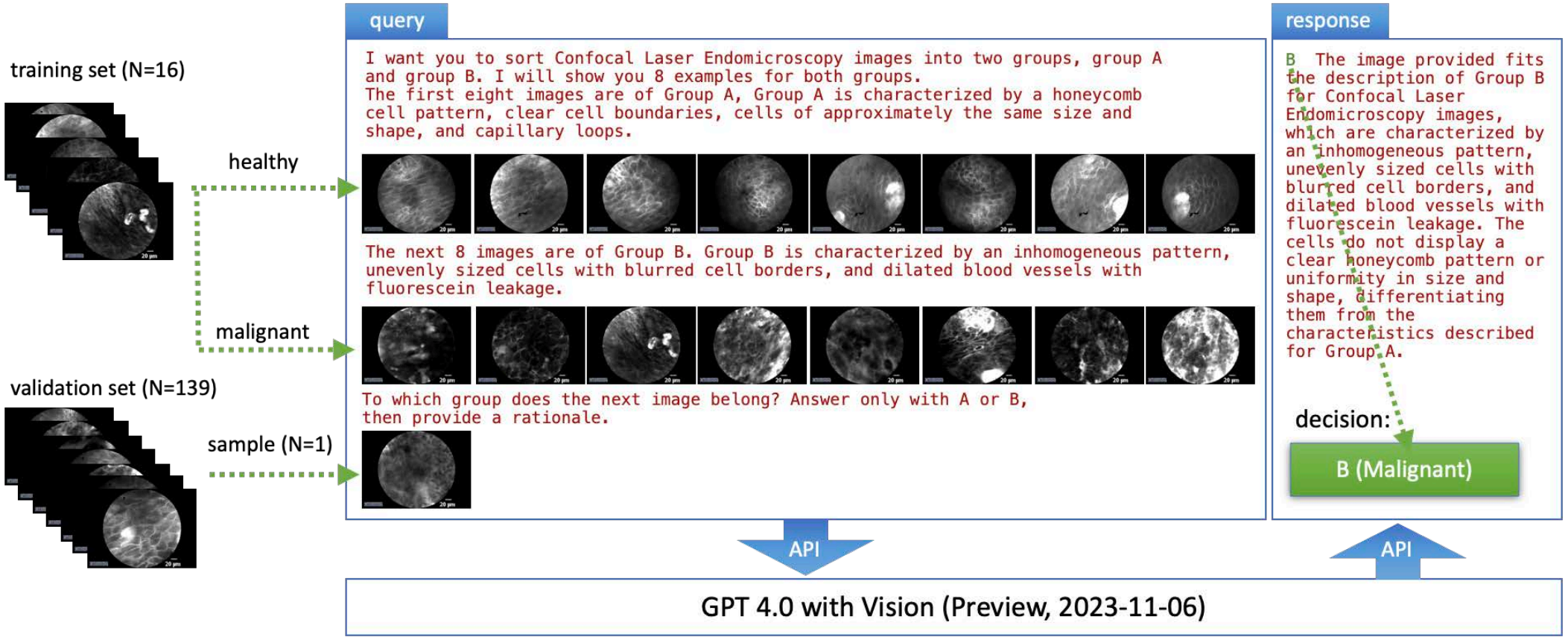
- The authors also investigate how well the system works on a zero-shot task, i.e., without using any labels of the target task and domain.
- For this, they used image-image retrieval and text-image retrieval:
  - Image-image retrieval feeds a reference image and a few other (candidate) images to the encoder and ranks all according to their similarity metric (cosine similarity) in latent space.
  - We can use this to calculate a precision at various levels of recall (Prec@Recall), like in AP calculation.
  - Text-image retrieval is similar but uses a textual query.

Zero-Shot / Few-Shot / k-shot learning and Few-shot prompting



- In large vision and language models often we want to investigate the zero shot / few shot capabilities.
- In the few shot scenario, we only take a small (in the 1-2 digits order) number of samples.
- A special case of few-shot learning is k-shot learning, where exactly k samples of a class are used during training.
- A special case of k-short learning is zero-shot learning, where we have no samples of a class/task during training.
- We differentiate this against few-shot prompting, where the model has not seen any data samples of the target task / domain during training, but receives such data as examples during inference.

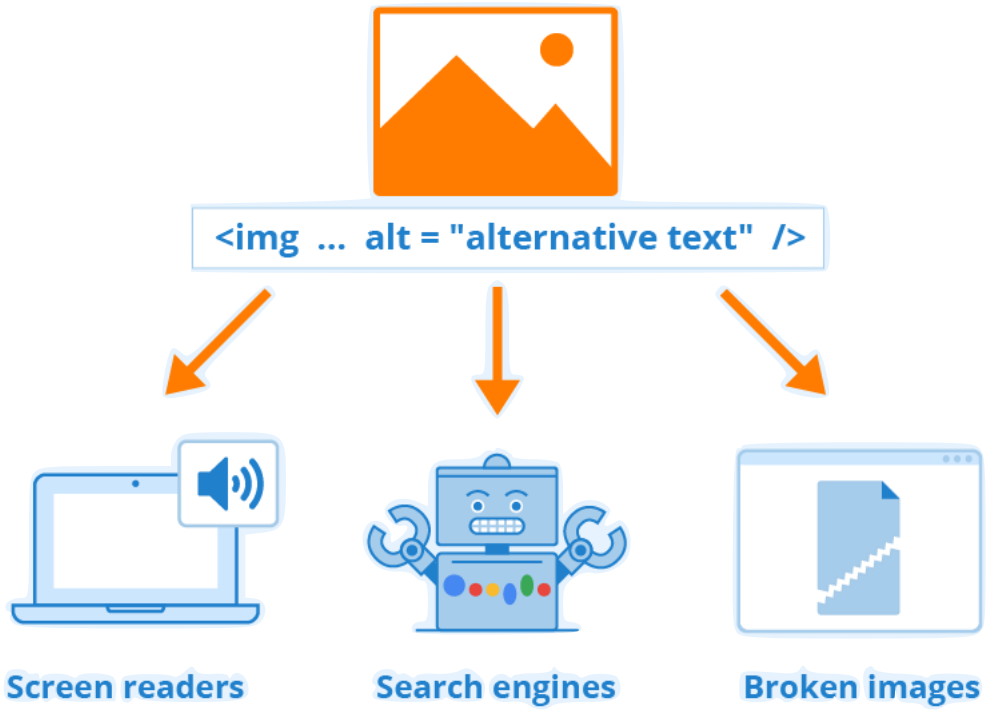
Example from our research: Few shot prompting (Sievert et al., 2024)



Using Natural Language as Supervision for Images at Scale

Hochschule Flensburg
University of Applied Sciences

- Another way we could use natural language (e.g., texts) is as a label for images where a categorical label (i.e., a class) is missing.
- In fact, we have huge (available) datasets for images that have captions:



Many images on websites are equipped with an „alternative text“

These kind of texts could be used as some kind of image labels!

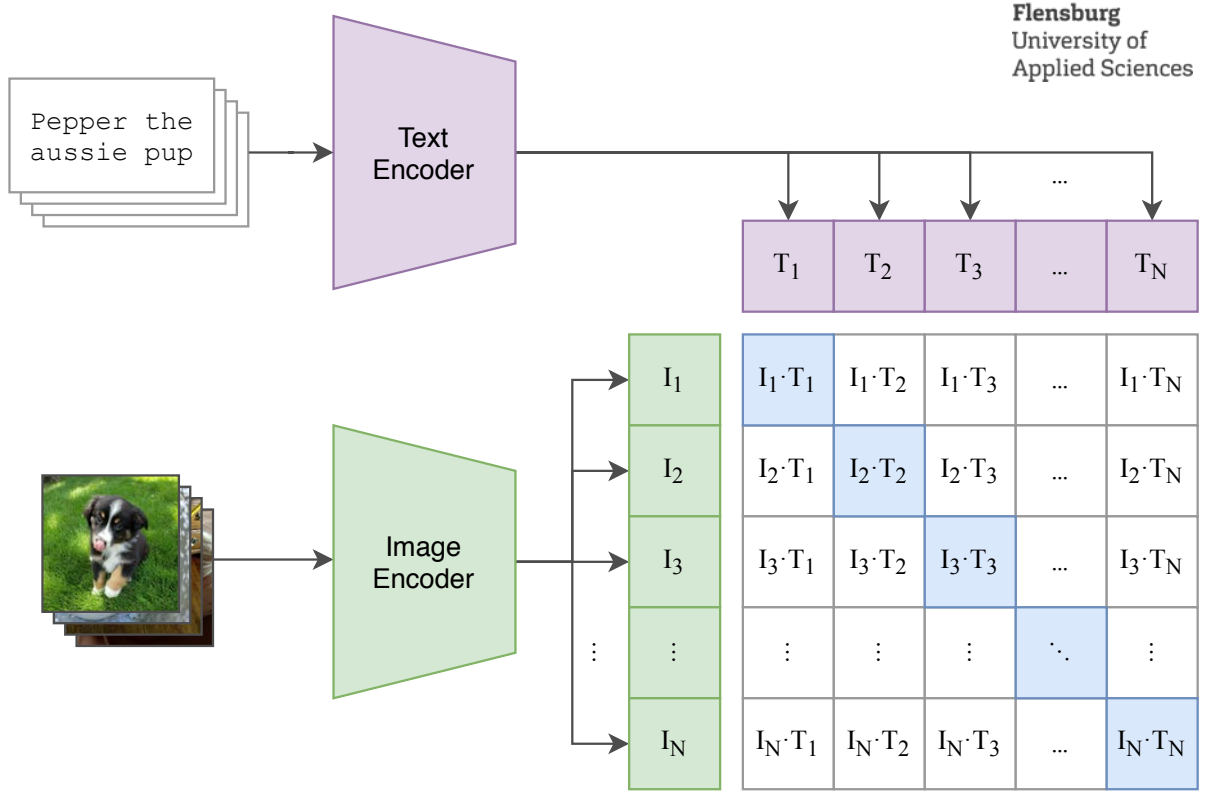
This could also be a very nice pretext task for representation learning.

https://www.seobility.net/en/wiki/ALT_Attributes

Contrastive Language-Image Pretraining (CLIP, Radford et al., 2021)

Hochschule Flensburg
University of Applied Sciences

- Besides thinking of text as just another data source containing a label information, we could also extend our thinking by realizing that text can express a much wider set of visual concepts than that.



- Using a dataset of 400 Million images curated from the internet, the authors of CLIP reiterate on the ConVIRT concept and scale it up significantly.

- The idea is that a wide range of tasks (inclusind OCR, geo-localization, action recognition) could be learnt by the model by using this pretext task.

https://arxiv.org/abs/2103.00020

CLIP: Algorithm

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

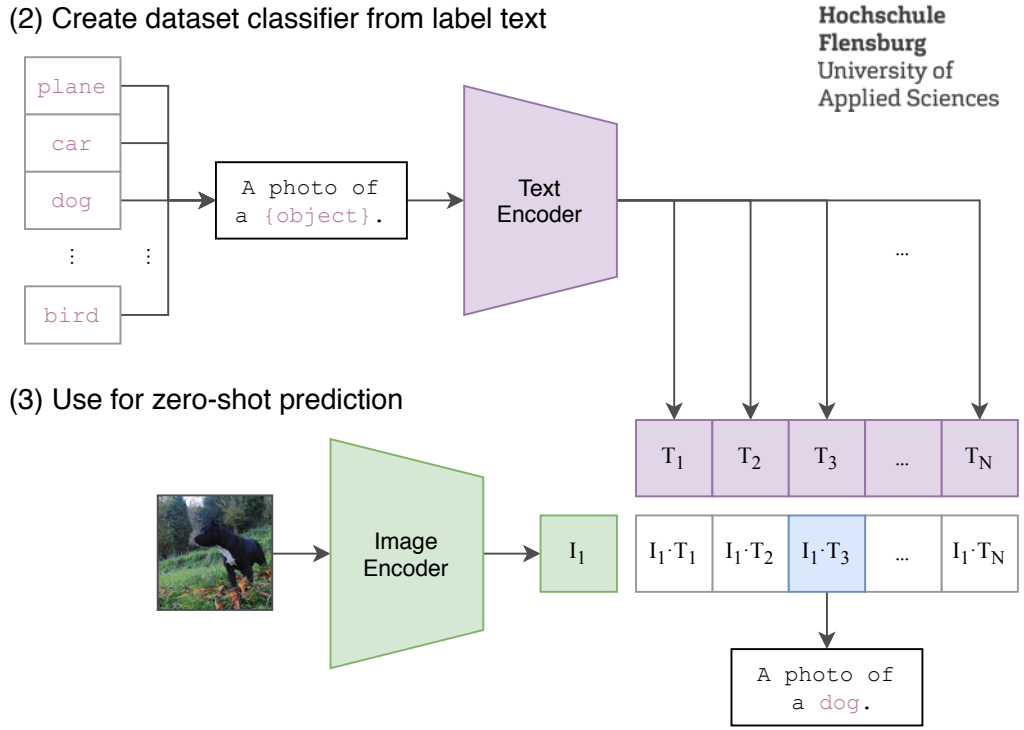
Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

- CLIP uses a linear projection (W_i, W_t) from the image/text encoder into a joint latent space.
- There, they use cosine similarity, scaled with a learnt temperature parameter t.
- Then they employ the cross-entropy loss (with softmax, not directly shown in the code), effectively a different formulation of the InfoNCE loss (softmax-log-loss).

https://arxiv.org/abs/2103.00020

CLIP: Zero-Shot application

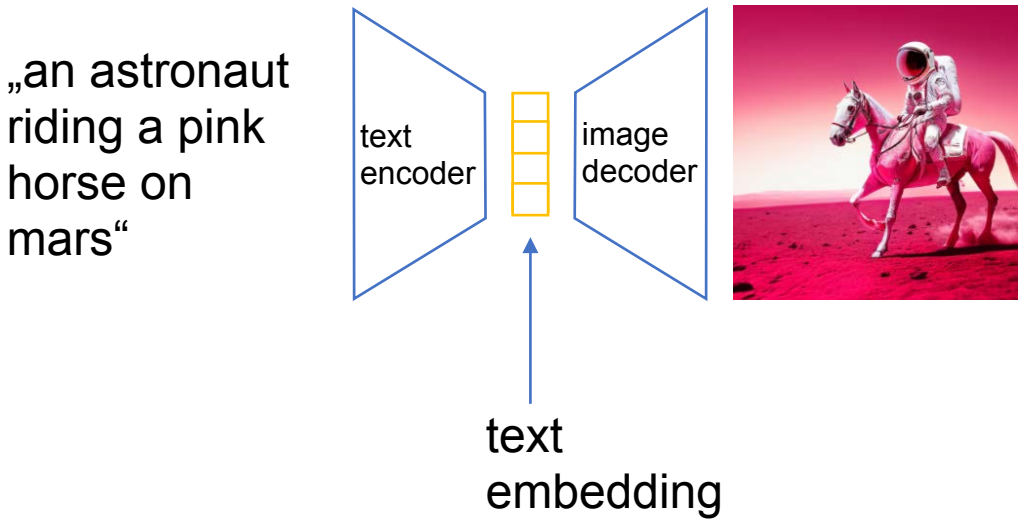
- For zero-shot application, the authors use a text-image retrieval task, effectively encoding a single image and multiple text prompts.
- The text prompts are created as „a photo of a class name}“ for this task, according to the class names of the target (downstream task) dataset.
- The text embedding with the highest (cosine) similarity will be chosen as winner. The results can also again be ranked according to cosine similarity.



https://arxiv.org/abs/2103.00020

Generating text-conditional images

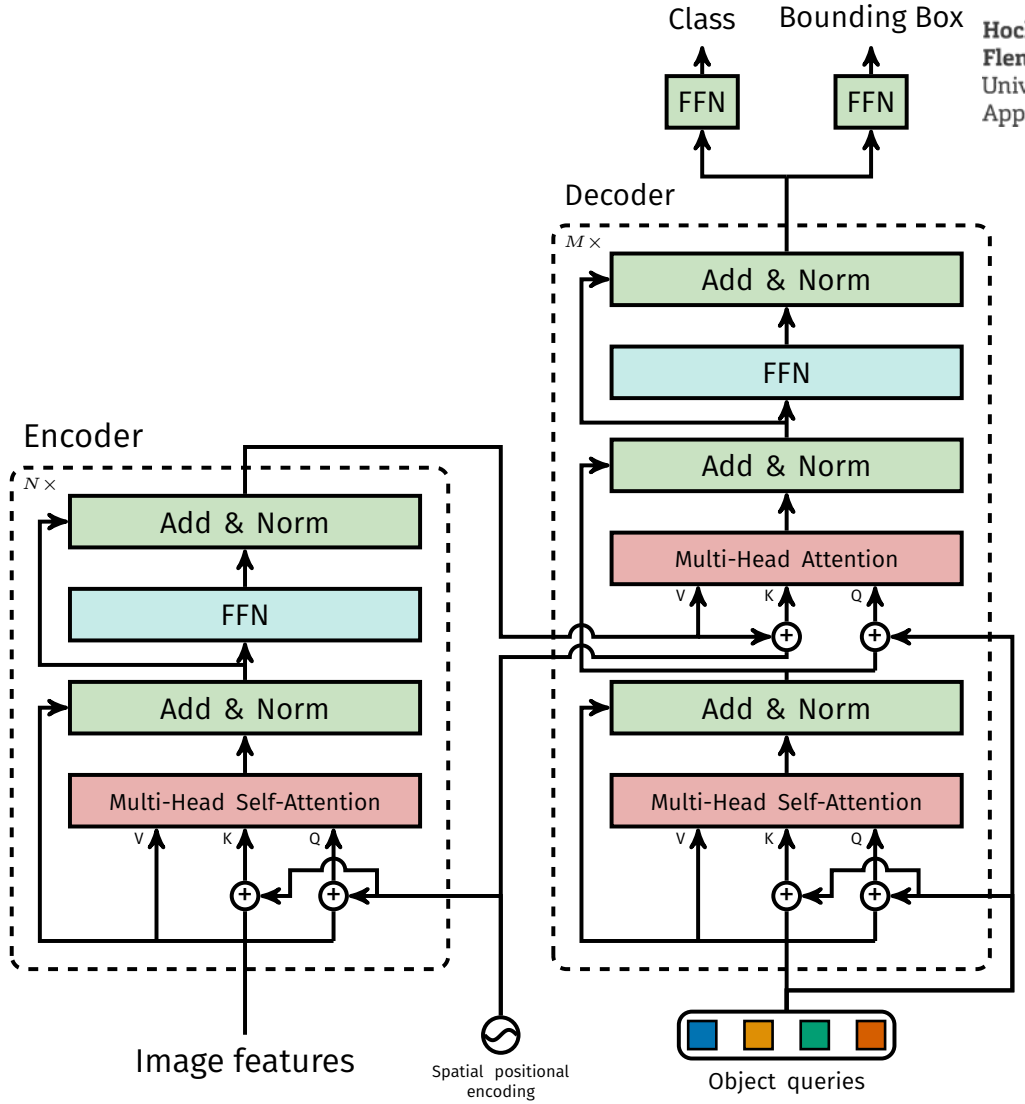
- Since we now know how to represent textual embeddings for image interpretation, we could also use this embedding as conditioning for image generation.



Feeding conditioning from a transformer

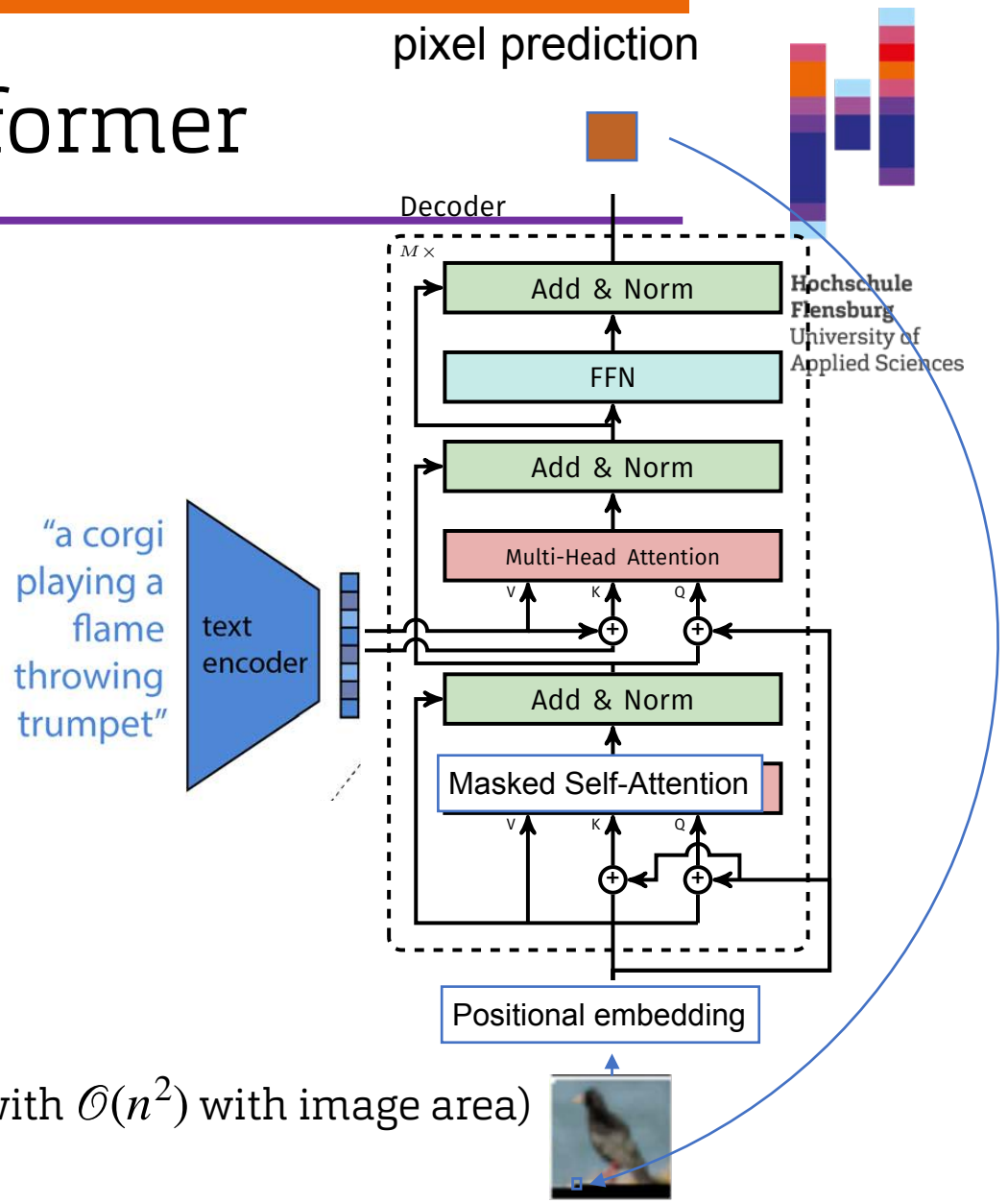
- Let's revist the idea we discovered in transformers:
  - We can use cross-attention to feed features with a global field of view
  - The self-attention / cross-attention blocks will assign the relevant information needed for the prediction task.

- Could we also use this principle to generate images?



Feeding conditioning from a transformer

- We could use the CLIP-generated text embedding (with positional embeddings) to condition a transformer.
- We could then predict the complete image in an auto-regressive way (like in LLMs).
- This way, the transformer can ensure to
  - include the conditioning when generating the image
  - adhere to the complete generated image context and thus generate consistent images
- Problem: This is infeasible for large images (self-attention growing with $\mathcal{O}(n^2)$ with image area)

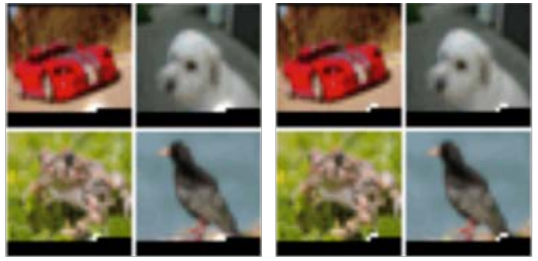


https://arxiv.org/pdf/1904.10509.pdf

Sparse Transformers (Child et al., 2019)

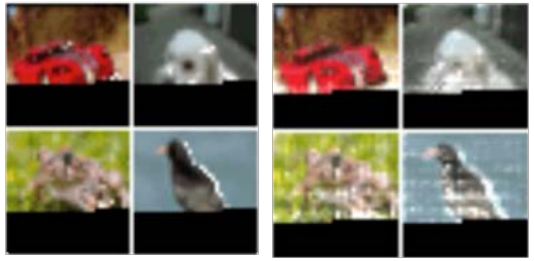
Hochschule Flensburg
University of Applied Sciences

- When looking into the attention maps of a 128-layer transformer for image generation, Child et al. found a specialization of the layers (attention as bright pixels):



Early layers:
mostly local attention
(—> like convolutions)

Some middle layers (19-20)
split attention in horizontal / vertical direction to ensure consistency



Some layers showed global data-dependent access patterns

Late layers show high sparsity, only activating for specific input patterns

- Apparently transformer heads have a high degree of specialization and are often sparsely activated.

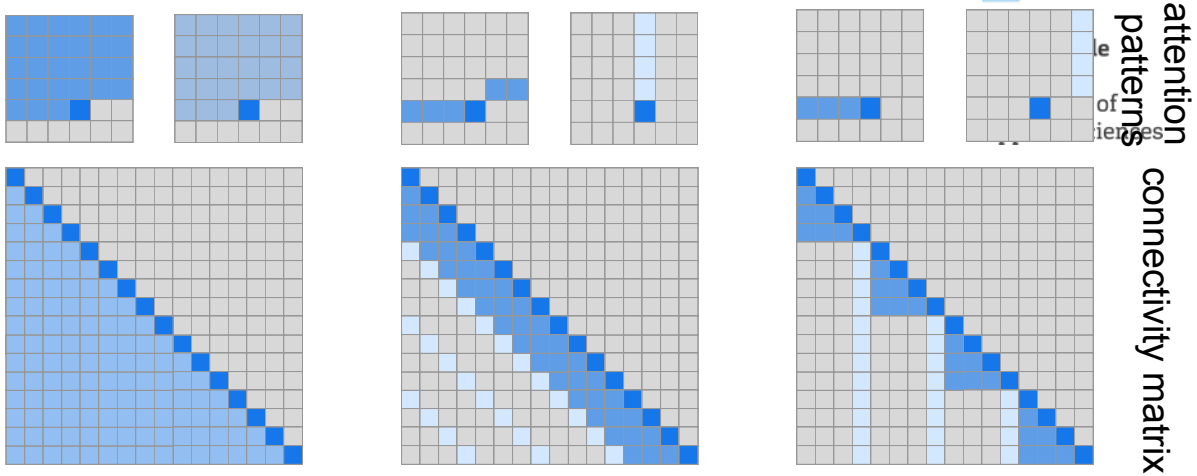
https://arxiv.org/pdf/1904.10509.pdf

Sparse Transformers (Child et al., 2019)

- If we only calculate attention on a sparse pattern, we can save a lot of compute (and parameters).

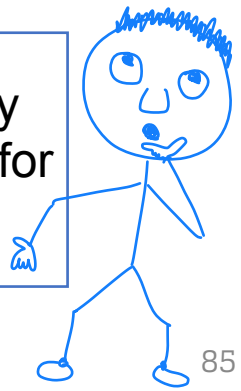
- The authors propose to reduce the image parts the transformer can attend to strongly, so that each of the n pixels can only attend to $\sqrt{n}$ other pixels.

- Effectively, this yields a complexity of $\mathcal{O}(n\sqrt{n})$.



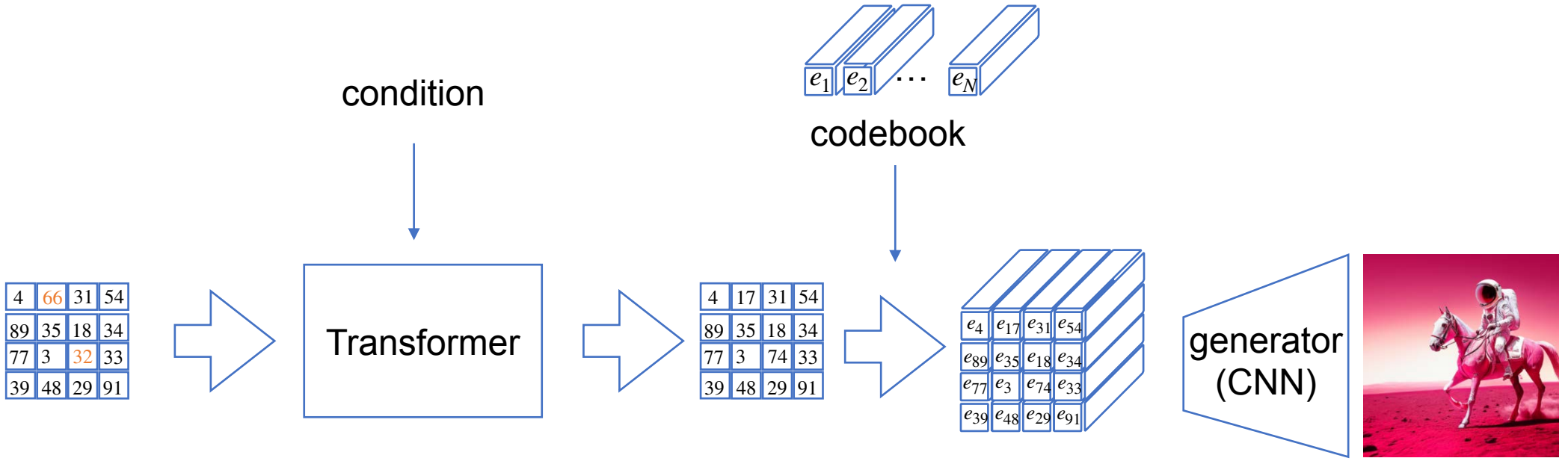
(a) Transformer (b) Sparse Transformer (strided) (c) Sparse Transformer (fixed)

Nice idea for reducing the complexity of the transformer, but not sufficient for large (>128x128) images.



https://arxiv.org/pdf/1904.10509.pdf

Let's revisit the idea of VQ-VAE / VQ-GAN

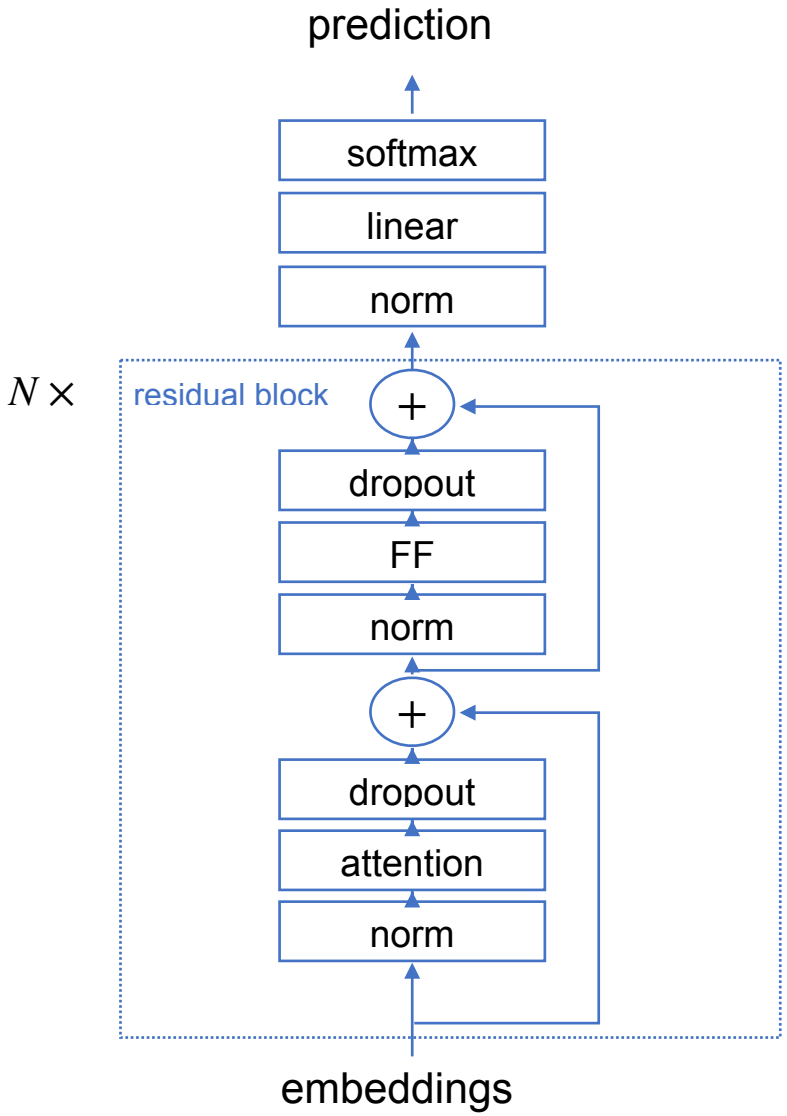


- We need to **compress** the latent image such that using a transformer for image generation becomes feasible again.
- The VQ-VAE employed in VQ-GAN fulfilled the same purpose, and it additionally performed a tokenization.

DALL-E: https://arxiv.org/pdf/2102.12092.pdf

DALL-E (Ramesh et al., 2021)

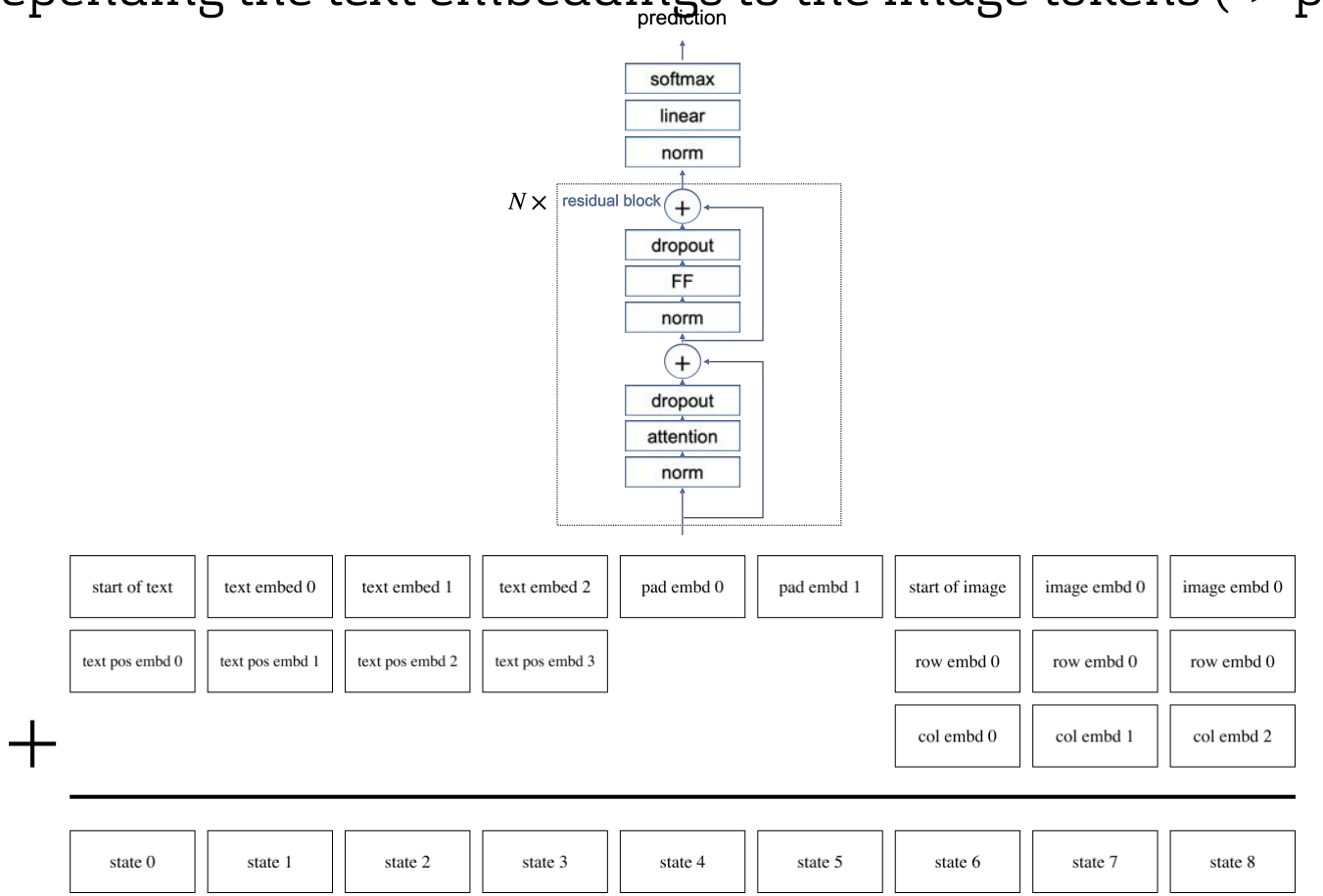
- The DALL-E architecture performs zero-shot text-to-image using a decoder-only sparse transformer architecture.
- The transformer works in the latent space of a quantized/discrete variational auto-encoder (dVAE).
- The transformer is used to predict auto-regressively new image tokens.



DALL-E: https://arxiv.org/pdf/2102.12092.pdf

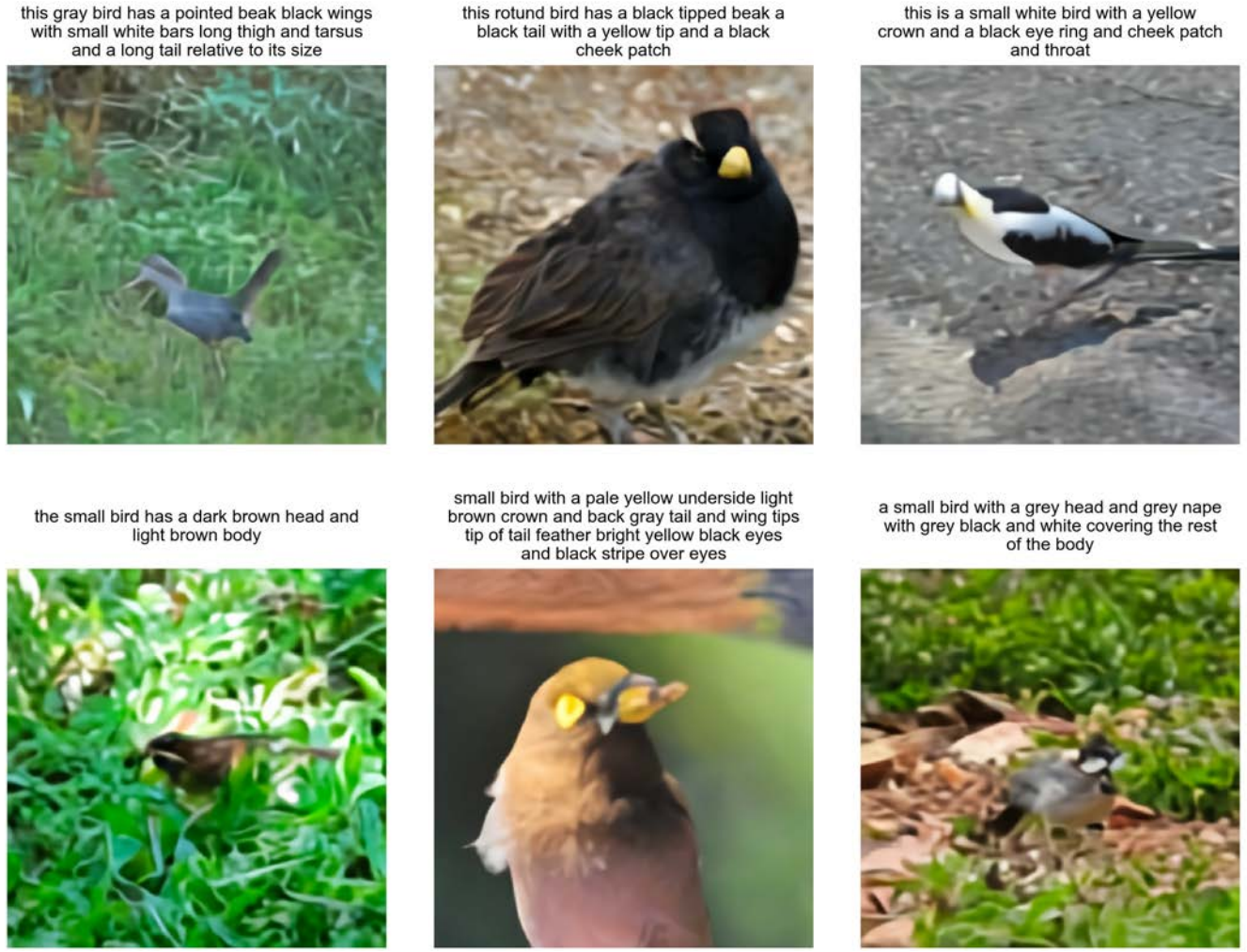
DALL-E (Ramesh et al., 2021)

- Instead of using cross-attention from text embeddings, they combine text and image embeddings by prepending the text embeddings to the image tokens (-> pooling).



DALL-E: https://arxiv.org/pdf/2102.12092.pdf

DALL-E: Examples



- While showing remarkably aligned examples, the FID / IS scores fall short of previous (class-conditional) works.

Model	Codebook Size	dim $\mathcal{Z}$	FID/val	FID/train
VQVAE-2	64×64 & 32×32	512	n/a	~ 10
DALL-E [59]	32×32	8192	32.01	33.88
VQGAN	16×16	1024	7.94	10.54
VQGAN	16×16	16384	4.98	7.41
VQGAN*	32×32	8192	1.49	3.24
VQGAN	64×64 & 32×32	512	1.45	2.78

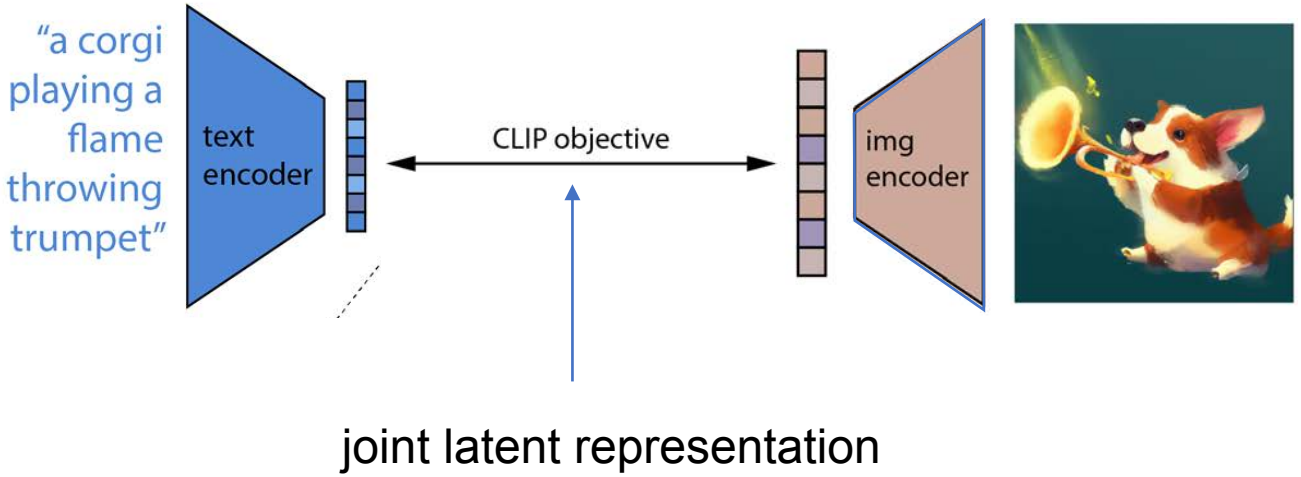
Table 5. FID on ImageNet between reconstructed validation split and original validation (FID/val) and training (FID/train) splits. *trained with Gumbel-Softmax reparameterization as in [59, 29].

https://arxiv.org/pdf/2102.12092.pdf

https://arxiv.org/abs/2012.09841

Utilizing the CLIP latents for image generation

- The CLIP latents already were trained to yield a joint latent representation for text and image.



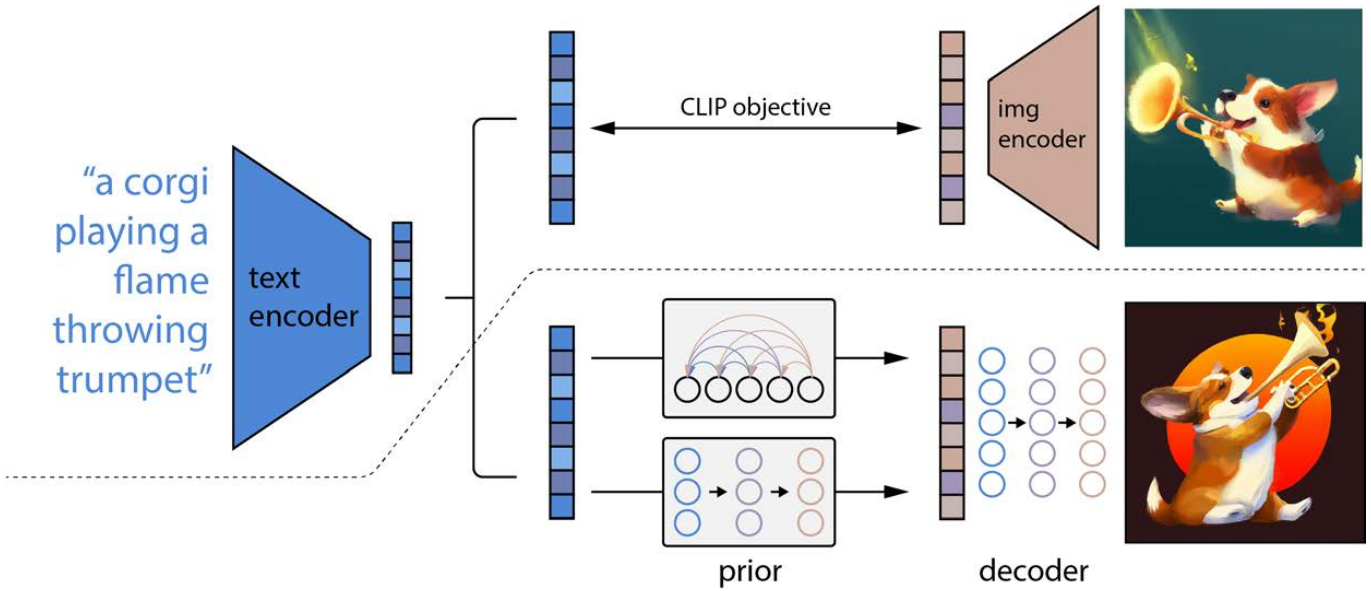
- So it should be possible to decode an image from this joint representation, if we feed a text, if we just „reverse“ the image encoder.

https://cdn.openai.com/papers/dall-e-2.pdf

Image generation using CLIP latents (UnCLIP ≙DALL-E 2, Ramesh et al., 2022)

Hochschule Flensburg
University of Applied Sciences

- In fact, image and text often have a very different level of detail, so the text embeddings necessarily have much less detail than the image imbeddings.



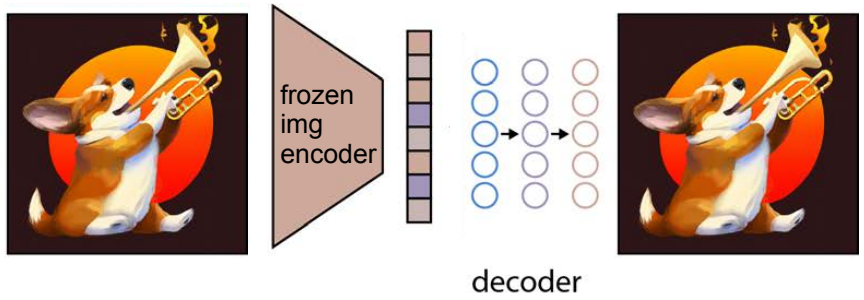
- However, we can use the joint latents to train a prior (which translates text to image embeddings), which can then be decoded to an image again.

https://cdn.openai.com/papers/dall-e-2.pdf

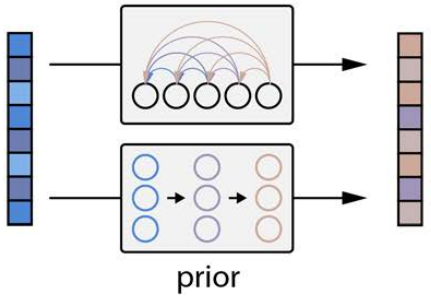
Image generation using CLIP latents (UnCLIP ≙DALL-E 2, Ramesh et al., 2022)

Hochschule Flensburg
University of Applied Sciences

- We can train the image decoder in the typical auto-encoder fashion:



- We hence can use the previously trained CLIP image encoder (frozen) and train a decoder for it.
- Then, we can train a network („prior“) to translate CLIP text embeddings to CLIP image embeddings:



Note that we can use either autoregressive prediction or diffusion-based approaches for the „prior“.

https://cdn.openai.com/papers/dall-e-2.pdf

Interpolation in latent space of CLIP

- Since the CLIP model produces a convex latent space, we can also interpolate the embeddings using spherical interpolation now:

a photo of a cat → an anime drawing of a super saiyan cat, artstation

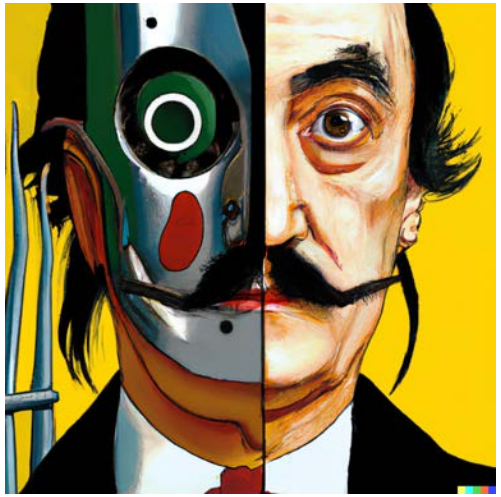
a photo of a victorian house → a photo of a modern house

a photo of an adult lion → a photo of lion cub

a photo of a landscape in winter → a photo of a landscape in fall

https://cdn.openai.com/papers/dall-e-2.pdf

UnCLIP (DALL-E 2): Results



vibrant portrait painting of Salvador Dalí with a robotic half face

a shiba inu wearing a beret and black turtleneck

an espresso machine that makes coffee from human souls, artstation

panda mad scientist mixing sparkling chemicals, artstation

- DALLE-2 yielded a high image fidelity, even for large images.

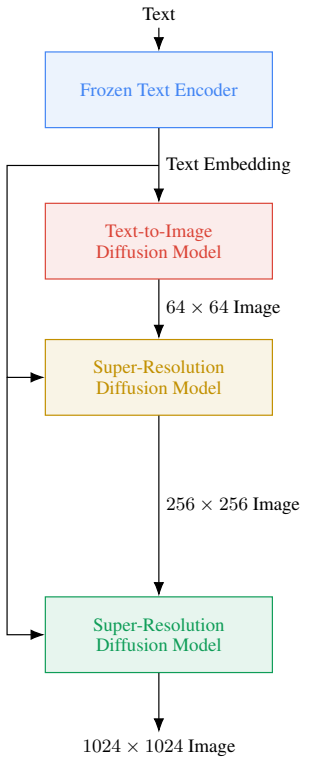
Model	FID	Zero-shot FID
AttnGAN (Xu et al., 2017)	35.49	
DM-GAN (Zhu et al., 2019)	32.64	
DF-GAN (Tao et al., 2020)	21.42	
DM-GAN + CL (Ye et al., 2021)	20.79	
XMC-GAN (Zhang et al., 2021)	9.33	
LAFITE (Zhou et al., 2021)	8.12	
Make-A-Scene (Gafni et al., 2022)	**7.55**	
DALL-E (Ramesh et al., 2021)		$\sim$ 28
LAFITE (Zhou et al., 2021)		26.94
GLIDE (Nichol et al., 2021)		12.24
Make-A-Scene (Gafni et al., 2022)		
unCLIP (AR prior)		10.63
unCLIP (Diffusion prior)		**10.39**

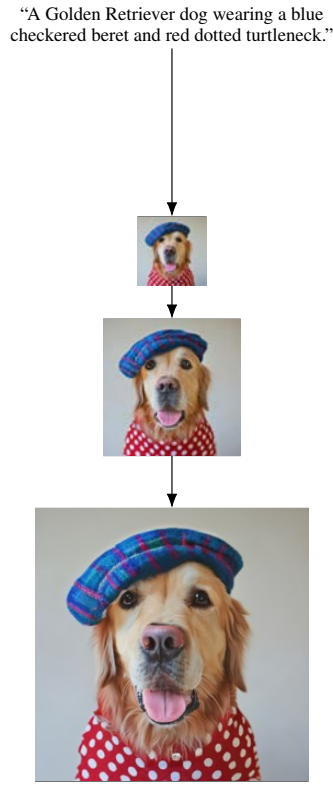
- The results for the auto-regressive and the diffusion-based prior are similar.

https://cdn.openai.com/papers/dall-e-2.pdf

Imagen (Saharia et al., 2022)

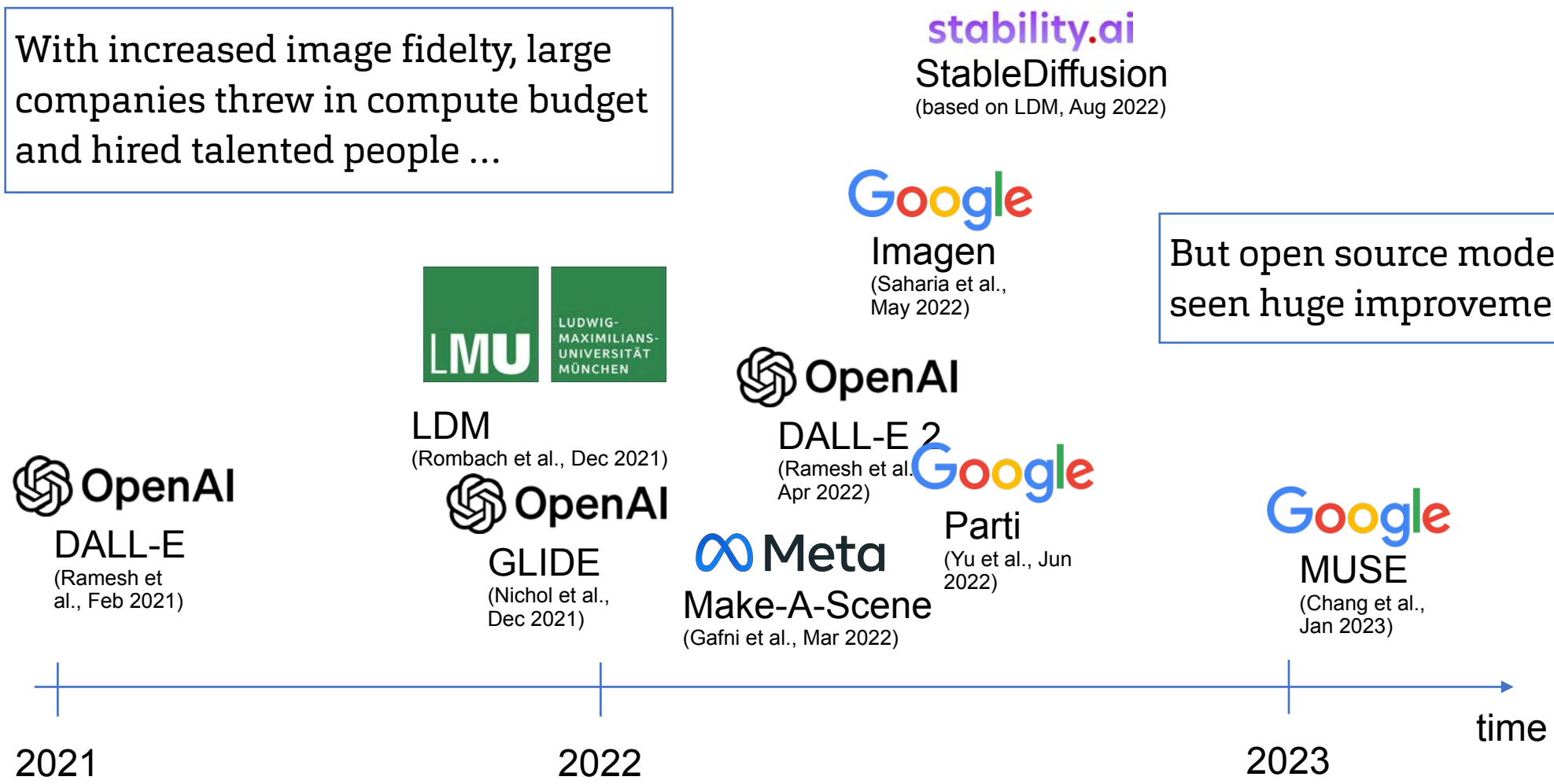
- Imagen reiterated on the idea of cascaded diffusion models (Ho et al., 2022).
- The architecture was a U-Net style encoder/decoder model used for denoising.
- They use T5 (text to text transfer transformer) as text embedding model.
- The used cross-attention to add the conditioning at each resolution.





Proprietory and open source models

With increased image fidelty, large companies threw in compute budget and hired talented people ...



But open source models also have seen huge improvements ...

Open Datasets: LAION

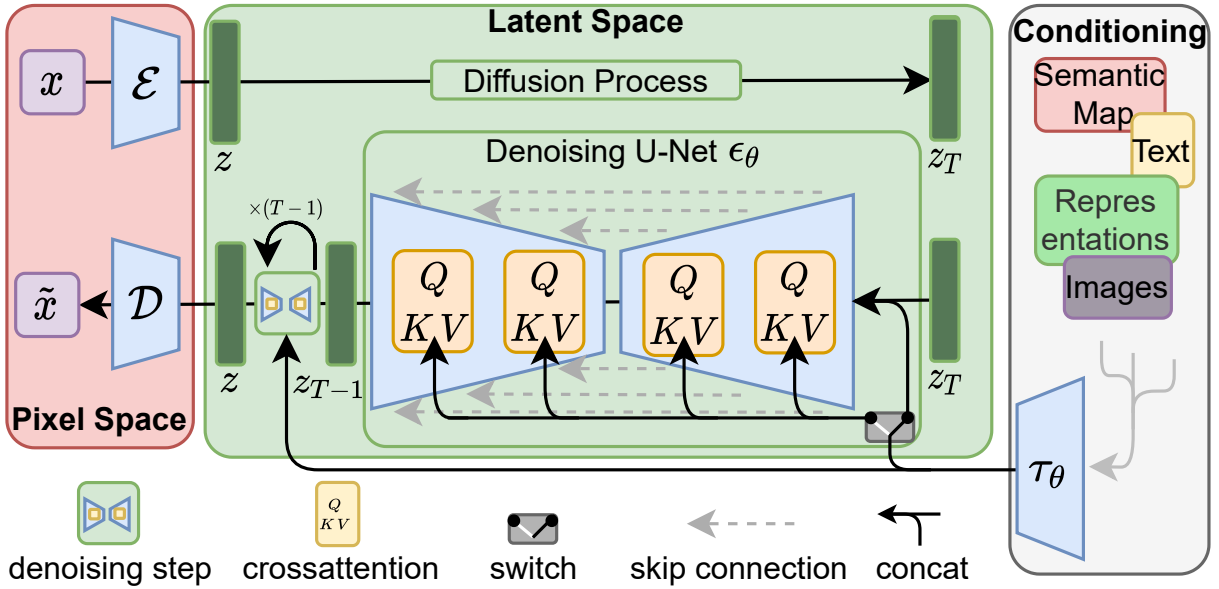
LAION

Hochschule Flensburg
University of Applied Sciences

- The Large-scale Artificial Intelligence Open Network (LAION e.V.) is a non-profit organization based in Hamburg that has set the goal of providing large-scale datasets to the public.
- The most massive dataset (LAION-5B) contains URLs and textual descriptions from 5.85 billion images found on the web (CommonCrawl).
- They additionally provide tools for NSFW detection (pornography, voilence, …) to filter images unsuitable for model training and an open source version of CLIP (OpenCLIP).
- The datasets have recently been criticized for containing a small amount of CSAM (child abuse) material, which apparently was not filtered out by the automatic filters.
(https://stacks.stanford.edu/file/druid:kh752sm9123/ml_training_data_csam_report-2023-12-23.pdf)

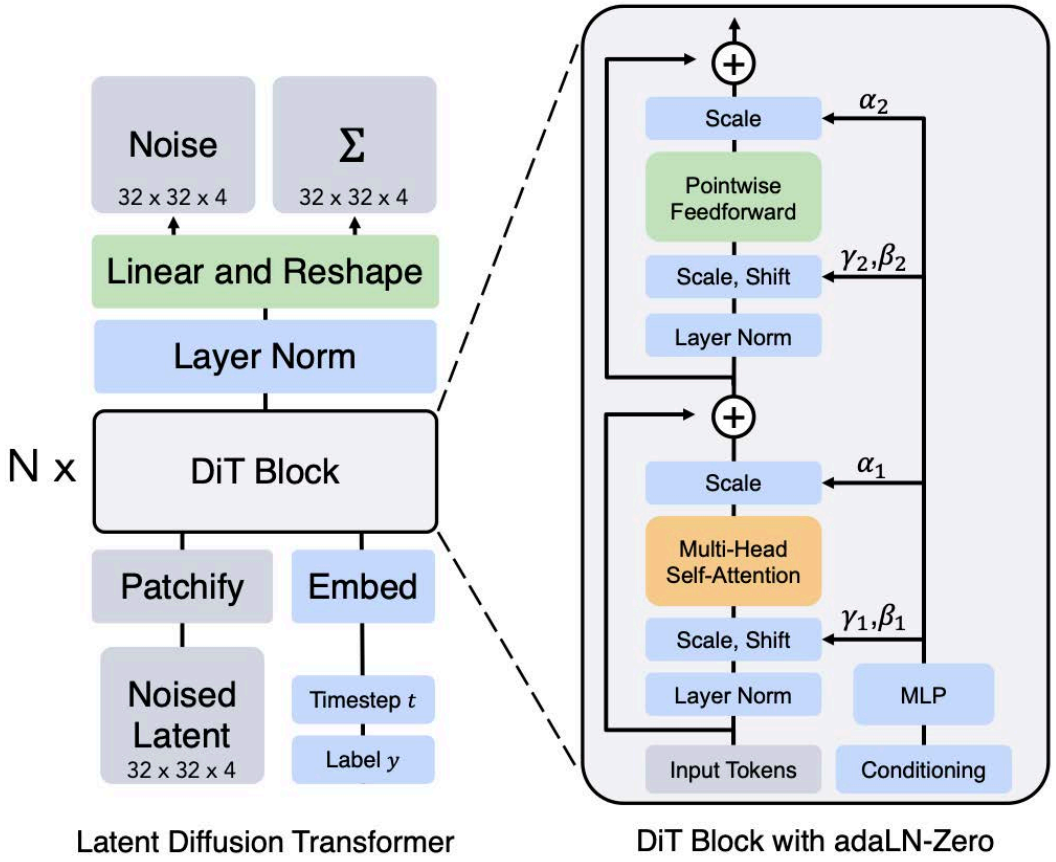
Latent Diffusion Models (Rombach et al., 2021)

- Rombach et al. proposed to perform the diffusion/denoising directly in latent space (latent diffusion model).
- Since the denoising/diffusion process requires a significant computational power, reducing it to a compressed latent space saves lots of time in the training of the model.
- Conditioning can be applied using semantic maps, texts, images and more.



Diffusion Transformers (DiT, Peebles & Xie, 2023)

- Another recent idea was to use transformers as scalable architecture for image generation.
- In the diffusion transformers (DiT) paper, the authors show that this provides for a good tradeoff between image quality and parameter scale.
- They operate the (reverse) diffusion process in the latent space of a VAE.



Latent Diffusion Transformer

DiT Block with adaLN-Zero

https://arxiv.org/pdf/2212.09748

Diffusion Transformers (DiT, Peebles & Xie, 2023)

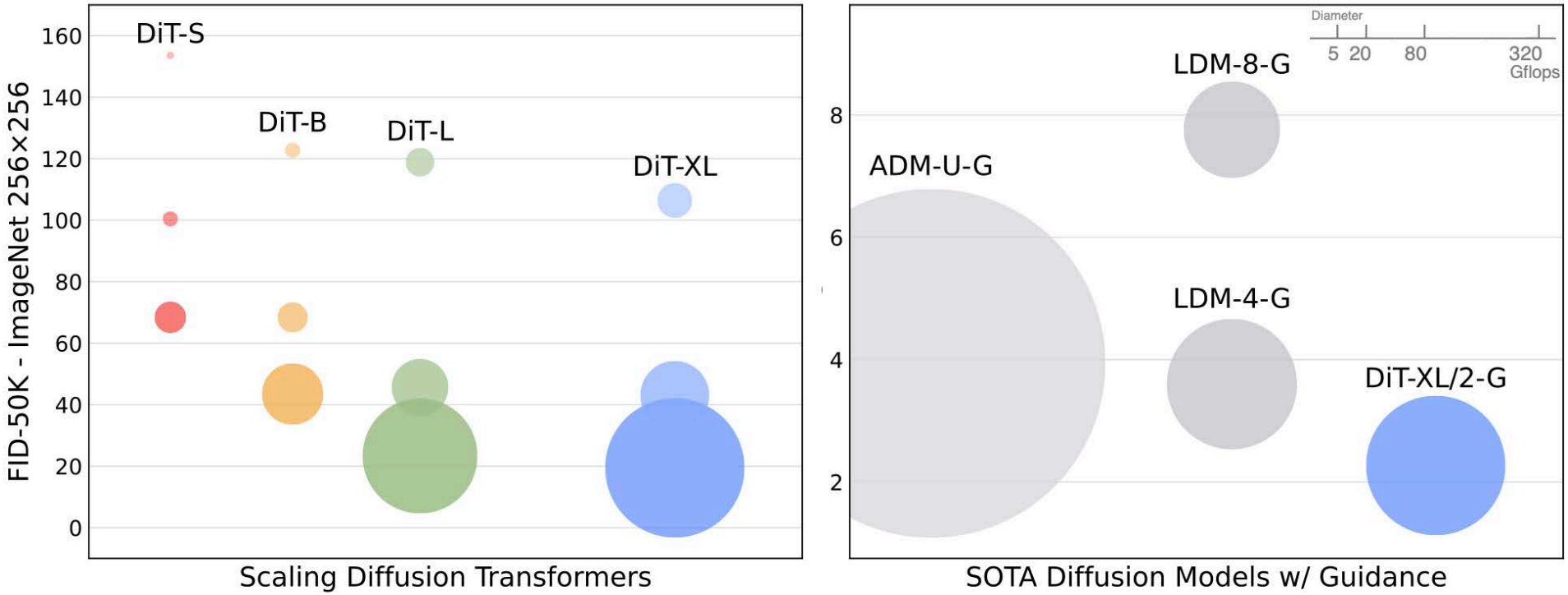


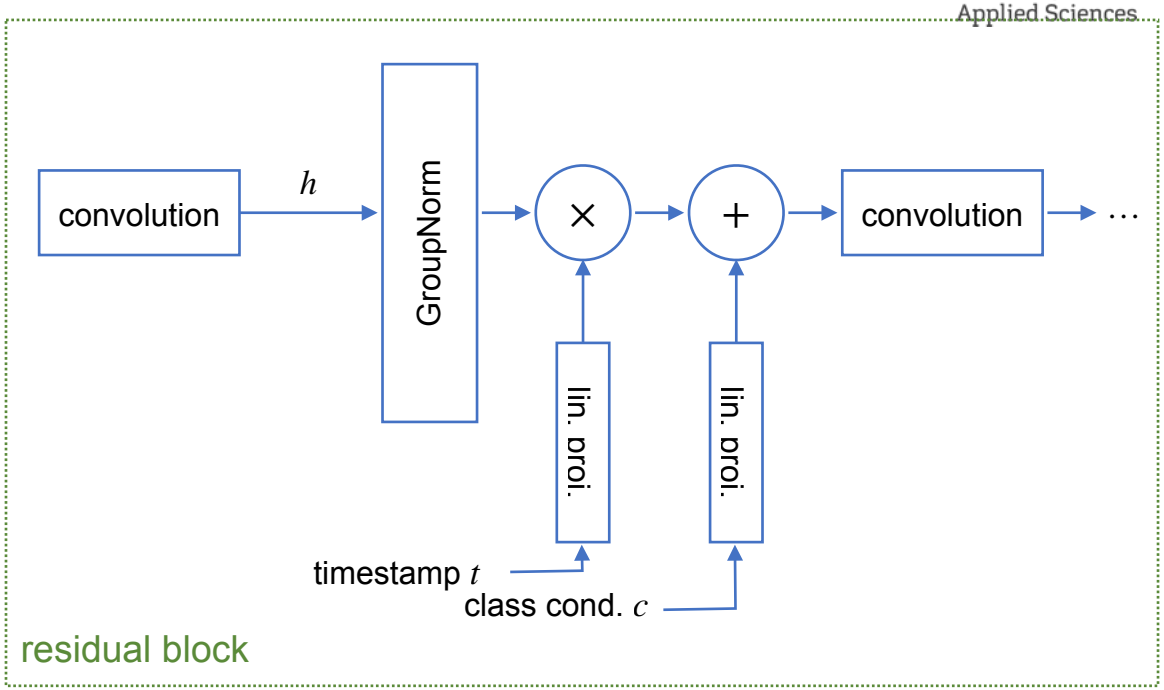
Figure 2. **ImageNet generation with Diffusion Transformers (DiTs).** Bubble area indicates the flops of the diffusion model. *Left:* FID-50K (lower is better) of our DiT models at 400K training iterations. Performance steadily improves in FID as model flops increase. *Right:* Our best model, DiT-XL/2, is compute-efficient and outperforms all prior U-Net-based diffusion models, like ADM and LDM.

https://arxiv.org/pdf/2212.09748

Conditioning using Adaptive GroupNorm (AdaGN, Dhariwal et al., 2021)

Hochschule Flensburg
University of Applied Sciences

- Adaptive GroupNorm was a novel method to incorporate class-conditioning.
- It works by simply using embeddings generated as linear projection of the timestamp and class.
- The model uses the information, since it helps in the noise prediction process.

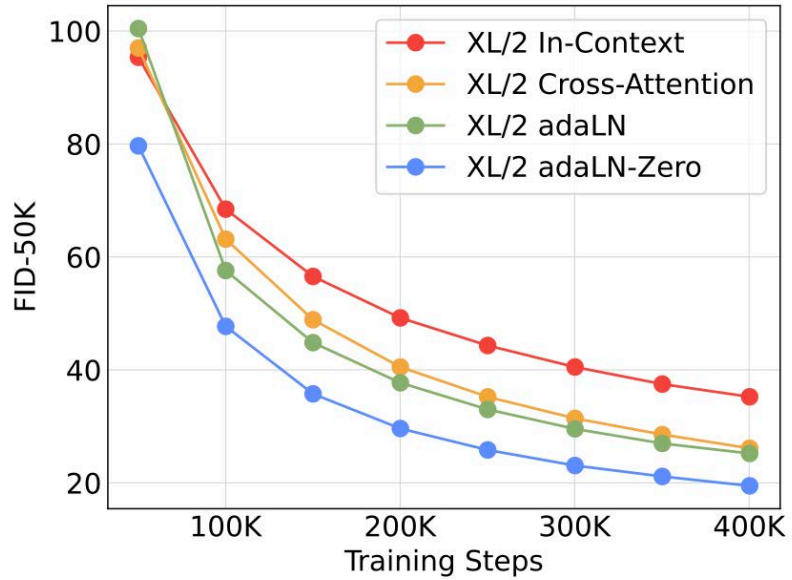


https://openreview.net/pdf?id=AAWuCvzaVt

Adaptive Layer Normalization

Hochschule Flensburg
University of Applied Sciences

- Adaptive Layer Normalization is another way of incorporating the conditioning to the model.
- It modifies the standard layer normalization, which is an in-model normalization technique that aims to normalize the distribution of hidden activations across channels for each token.
- In standard LayerNorm, the normalized activations are then rescaled and shifted by learned parameters that stay the same for every input.
- Adaptive LayerNorm replaces these fixed parameters with **conditioning-dependent** parameters.

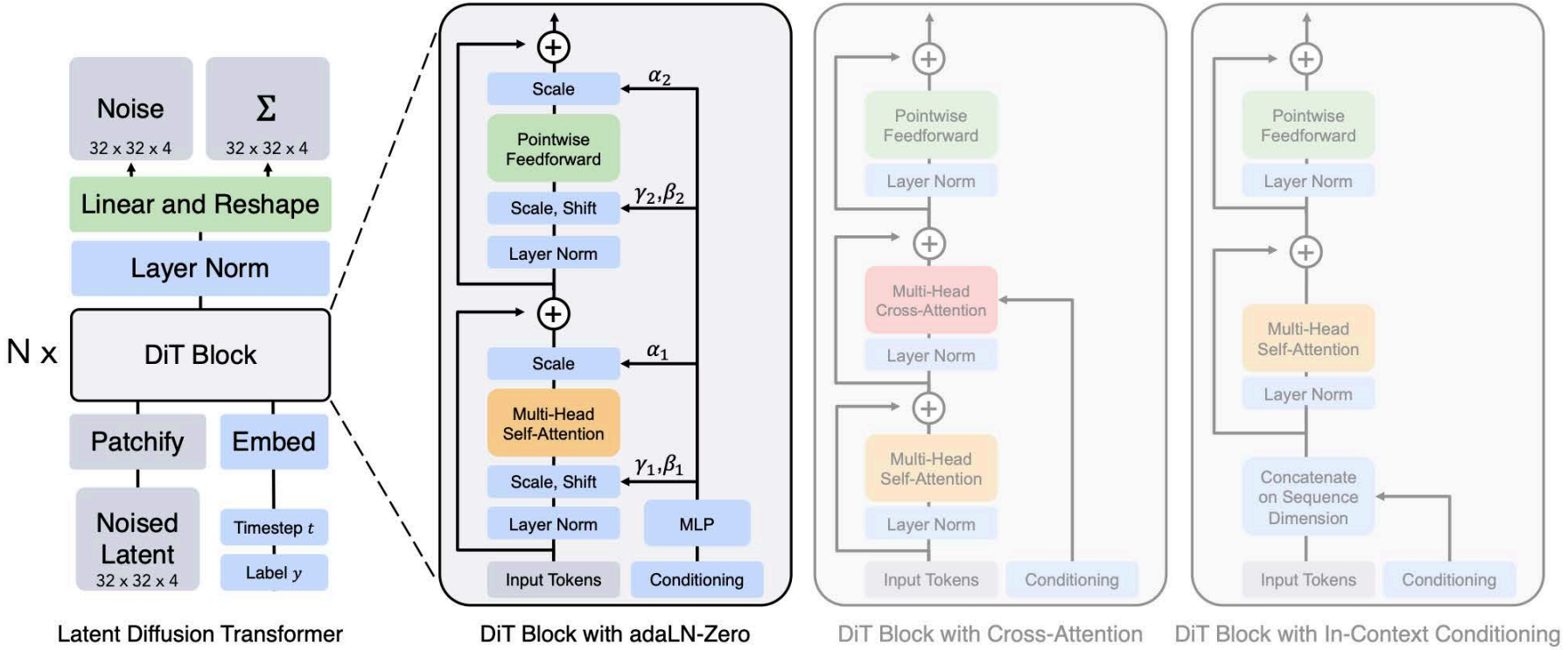


https://arxiv.org/pdf/2212.09748

Conditioning: Comparing three ideas (DiT paper)

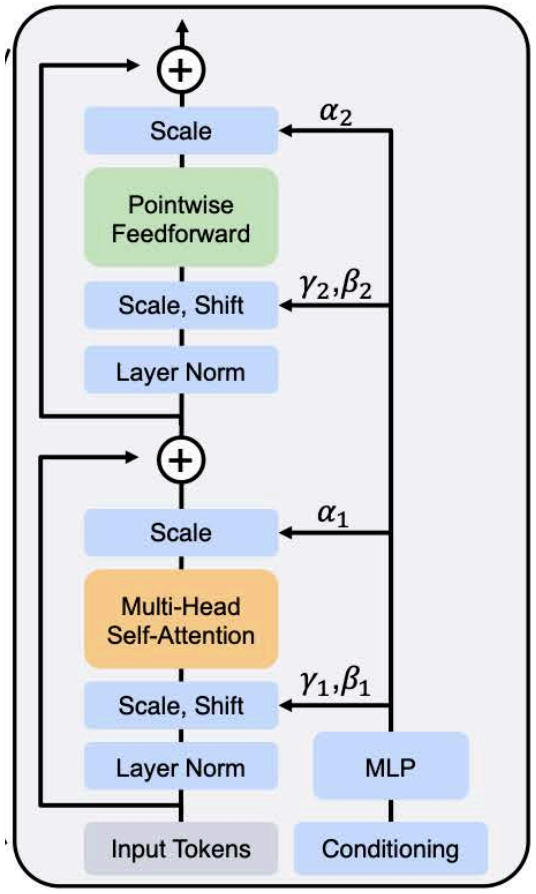
Hochschule Flensburg
University of Applied Sciences

- In principle, conditioning can be provided to the model by various ways.
- In the DiT paper, they compared:
  - adaptive layer norm (best-suited as of their paper)
  - cross-attention
  - in-context cond-itioning (concatenate input and text tokens)



https://arxiv.org/pdf/2212.09748

Adaptive Layer Normalization



DiT Block with adaLN-Zero

- Instead of a constant scale and shift, the model feeds the conditioning information (such as the diffusion timestep, text embedding, or class label) through a small neural network to produce dynamic scale and shift values.
- This means the model can adapt its internal feature processing depending on what is being generated, at every layer.
- Adaptive LayerNorm therefore supports fine-grained control, since the conditioning influences not just the input or attention layers, but the entire depth of the network.
- In diffusion transformers, this conditioning mechanism is especially useful, because global self-attention allows the injected signal to influence the entire image representation efficiently.
- The AdaLN-Zero variant initializes the learned scale and shift to zero, so each residual block initially contributes nothing. During training, the conditioning gradually “turns on” the behavior of each block, which stabilizes optimization and improves sample quality.

Stable Diffusion 3 (Esser et al., 2024)

Hochschule Flensburg
University of Applied Sciences

- Stable Diffusion 3 builds upon Rectified Flows using Transformers.
- In the Stable Diffusion 3 paper, the authors aim at answering the question:
  How do we formulate the interplay between noise and image, i.e., the flow between noise and image.
- They evaluate different functions (flow trajectories) for this:
  - Rectified Flows (Liu et al., 2022) : $z_t = (1-t)x_0 + t\epsilon \quad (\epsilon \sim \mathcal{N}(0,1))$
  - EDM (Karras et al., 2022): $z_t = x_0 + b_t\epsilon$
  - Cosine (Nichol & Dhariwal, 2021) : $z_t = \cos\left(\frac{\pi}{2}t\right)x_0 + \sin\left(\frac{\pi}{2}t\right)\epsilon$
  - LDM (Rombach et al., 2022): $z_t = a_t x_0 + \sqrt{1-a_t^2}\epsilon$

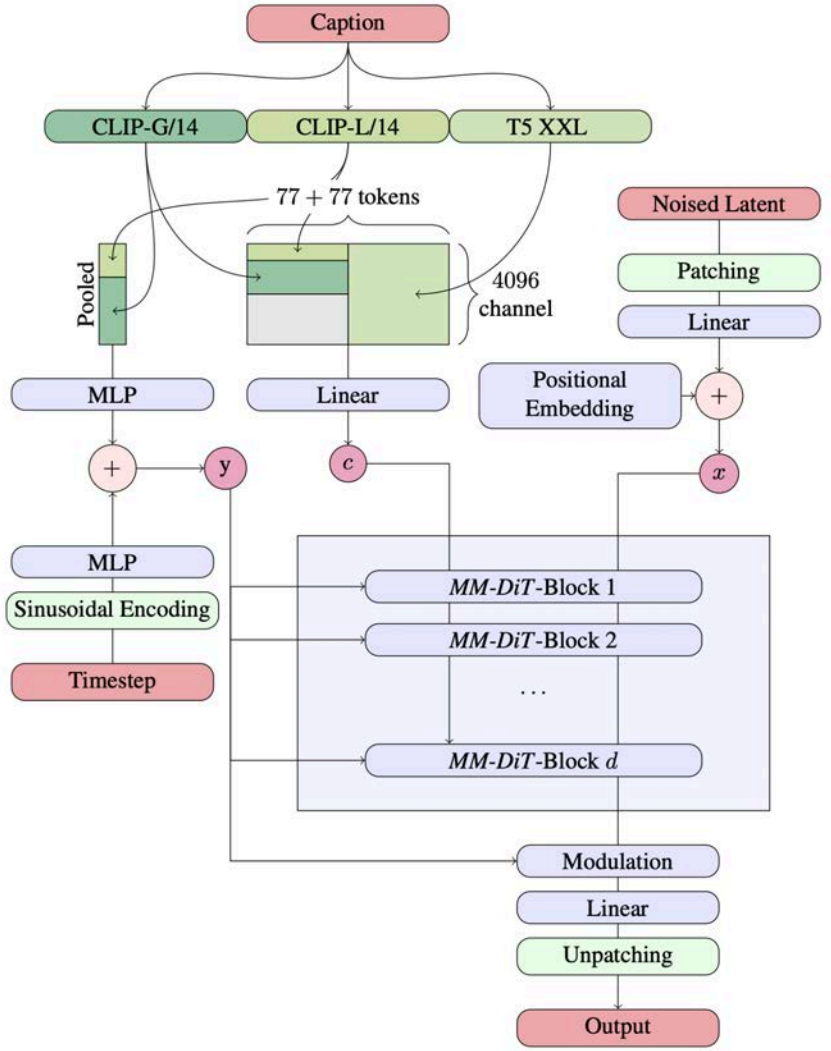
SD3: SNR Samplers

- All flow trajectories define a signal-to-noise ratio, between image and noise ϵ.
- The target is the prediction of the velocity term $u = x_0 - \epsilon$ (rectified flow matching), which is, however, easier for:
  - t=0 (pure data), as the velocity then points towards the mean of the (known) noise
  - t=1 (pure noise), as the velocity then points towards the mean of the data
- In-between, it's much harder for the model to guide the flow into the right direction.
- So this is why we should sample with higher probability from the middle of [0,1].

Hochschule Flensburg
University of Applied Sciences

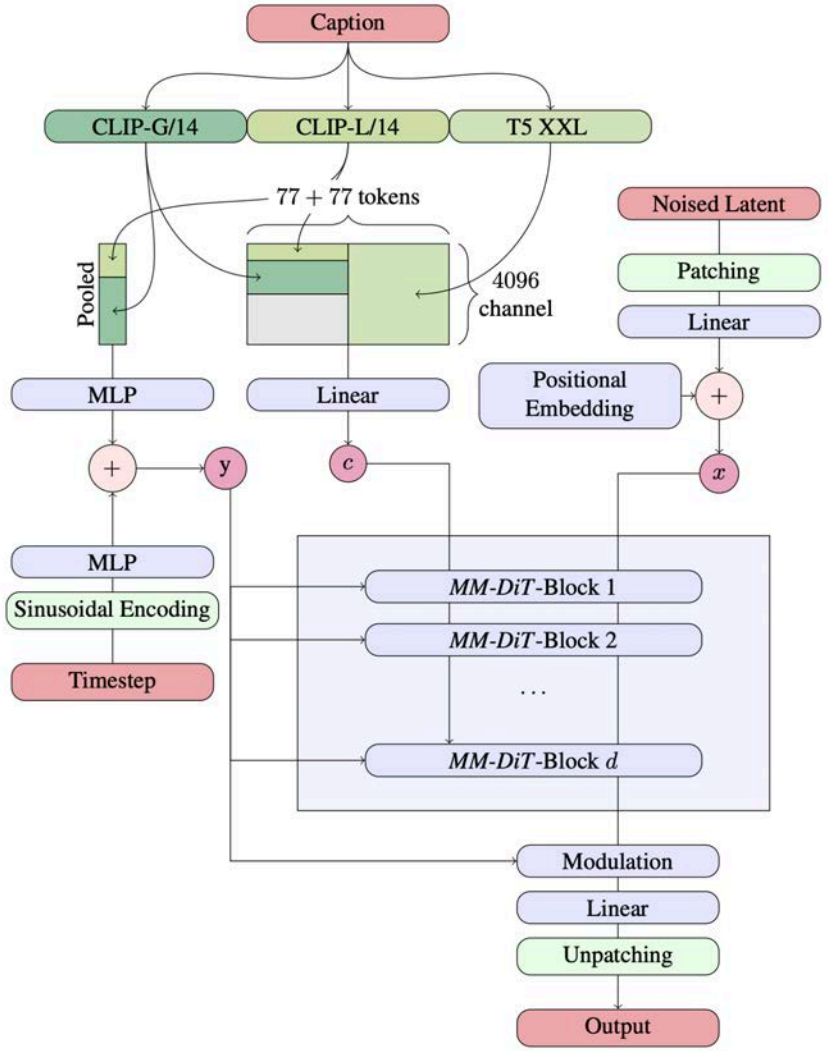
SD3: Architecture

- The SD3 architecture is a rectified flow transformer working in the latent space.
- It is built on the diffusion transformer (DiT) architecture.
- They use three different text embedding models (CLIP-G/14, CLIP-L/14, T5 XXL) to get an even better textual context information.
- Here, the CLIP text embeddings are considered image-related text embeddings, while the T5-XXL (a language model) embeddings form only text embeddings.

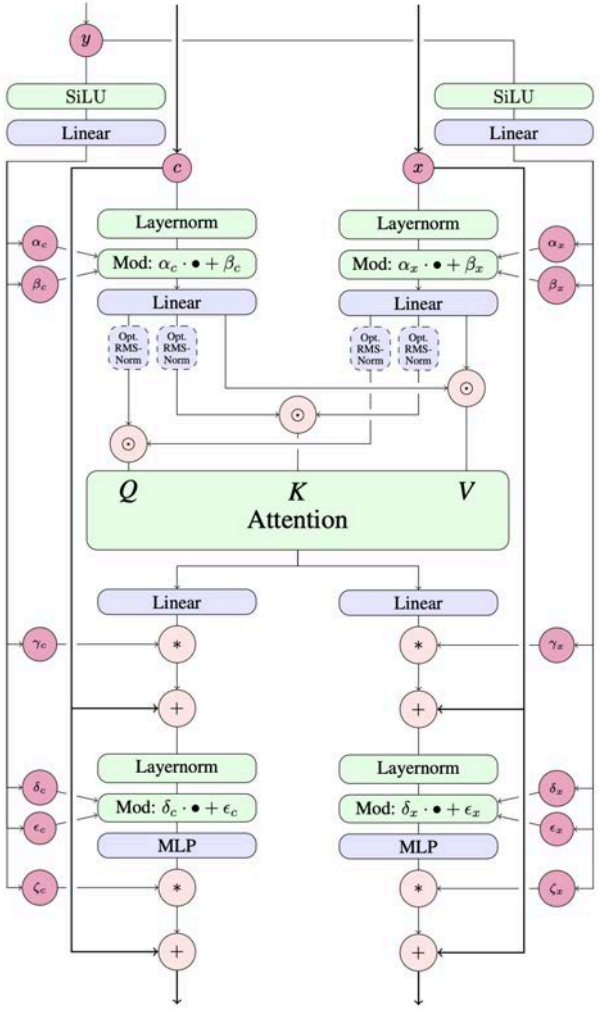


Text and image-related text embeddings

- In CLIP, we find a strong embeddings for those text parts that are **visually grounded** in the image.
- However, this means that those text embeddings being underrepresented in the visual representation are often missing.
- This is why the embeddings of a text model (T5-XXL) are used in SD3 additionally.
- Both are treated differently: The CLIP-based embeddings (and time embeddings) are used fine-grained at each DiT block, while the text embeddings from T5-XXL are used as global conditioning, providing broad semantic structure that guides the model's interpretation of the prompt.



SD3: Fine-grained Layer Adaptation



- They use adaptive layer norm at every step of the transformer block.
- CLIP-based embeddings are injected at each DiT transformer block
- This allows local, token-level conditioning throughout the denoising process
- Enhances visual grounding: shapes, colors, textures, spatial relations
- • Works alongside timestep embeddings for dynamic control over generation
- Enables detailed interpretation of prompt elements visible in the final image"

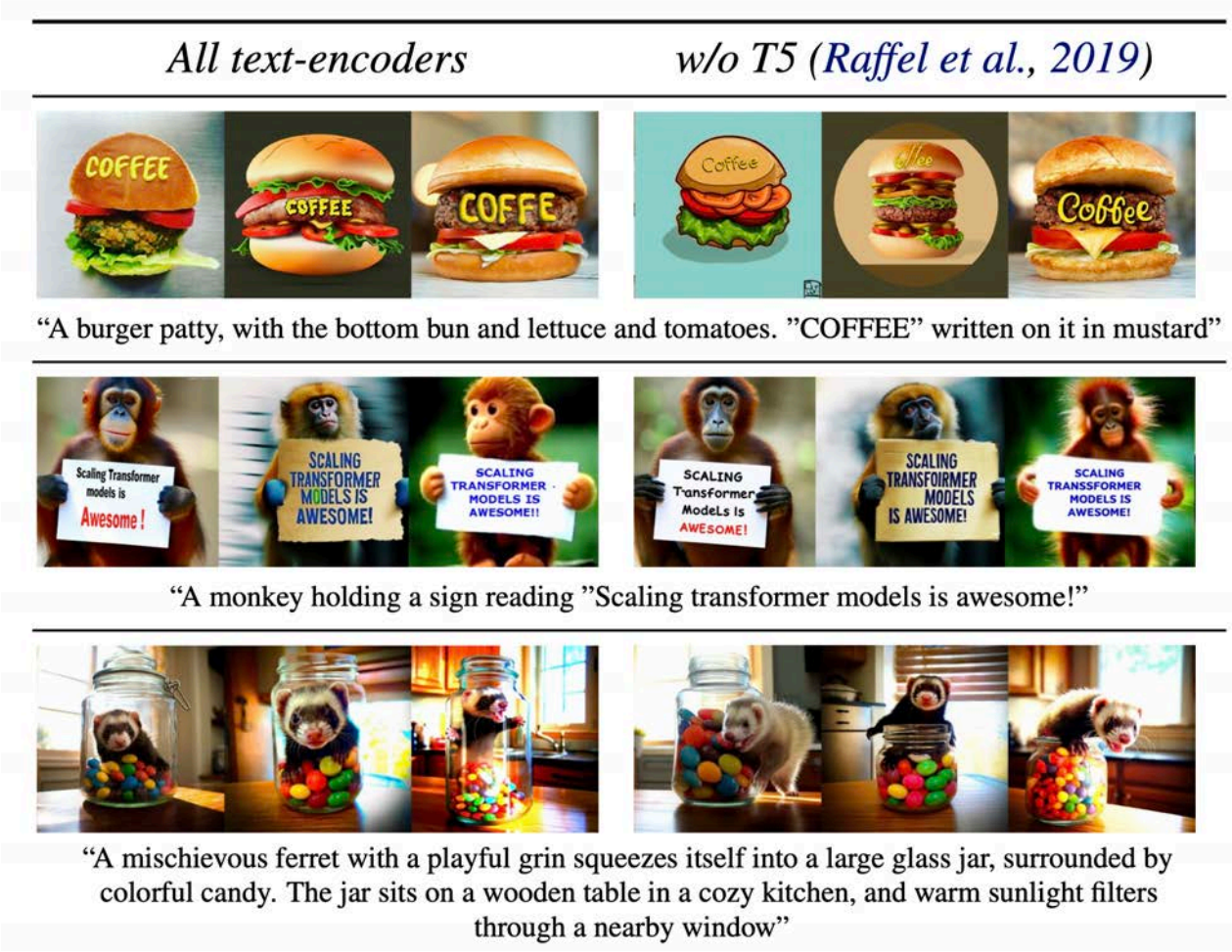
Hochschule Flensburg
University of Applied Sciences

SD3: Results

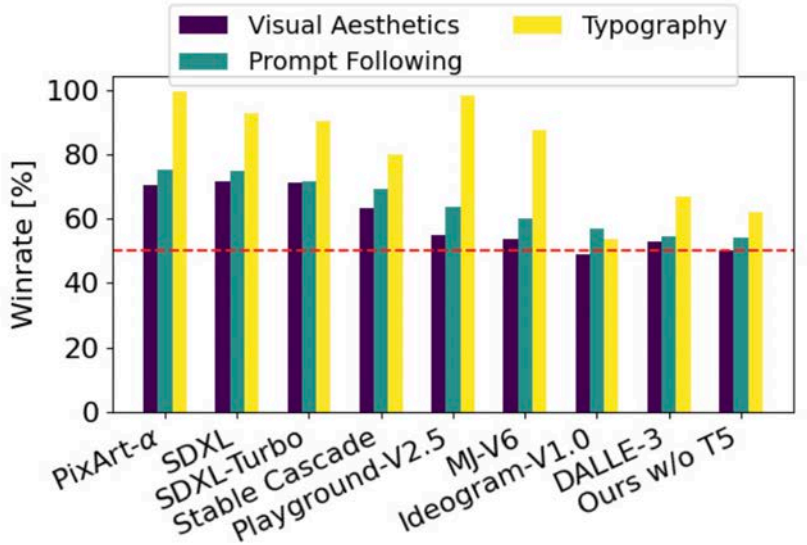
an old rusted robot wearing pants and a jacket riding skis in a supermarket.

smiling cartoon dog sits at a table, coffee mug on hand, as a room goes up in flames. “This is fine,” the dog assures himself.

SD3: Results



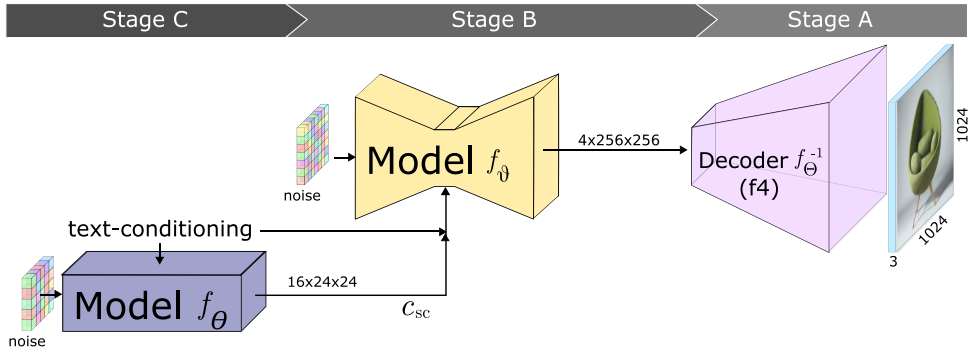
- SD3 performs strongly especially in the context of typography.
- This is especially thanks to the T5 model's embeddings.



Hochschule Flensburg
University of Applied Sciences

Even stronger latent compression: Würstchen (Pertinas et al., ICLR 2024)

- Since image composition (generating a coarse image from the text) is the most difficult task in image generation, it usually needs more diffusion steps to generate high-fidelity images.



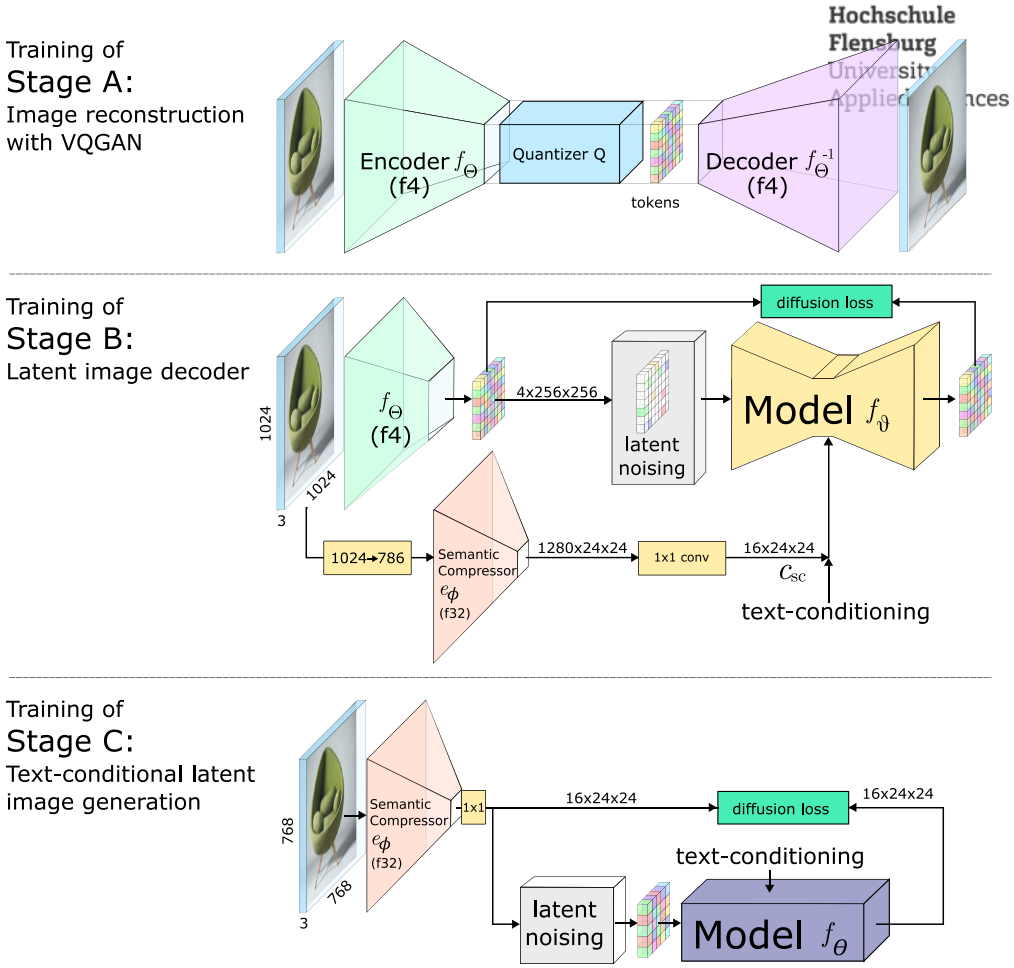
- So it is sensible to perform this task within a strongly compressed (semantic) latent space (saves lots of compute time in inference and training)

- The inference is performed by
  - generating a text-conditional strongly-compressed latent image (DDPM, Stage C)
  - denoising higher resolution latents (DDPM, Stage B)
  - decoding these latents to an image (Stage A).

https://arxiv.org/pdf/2306.00637.pdf

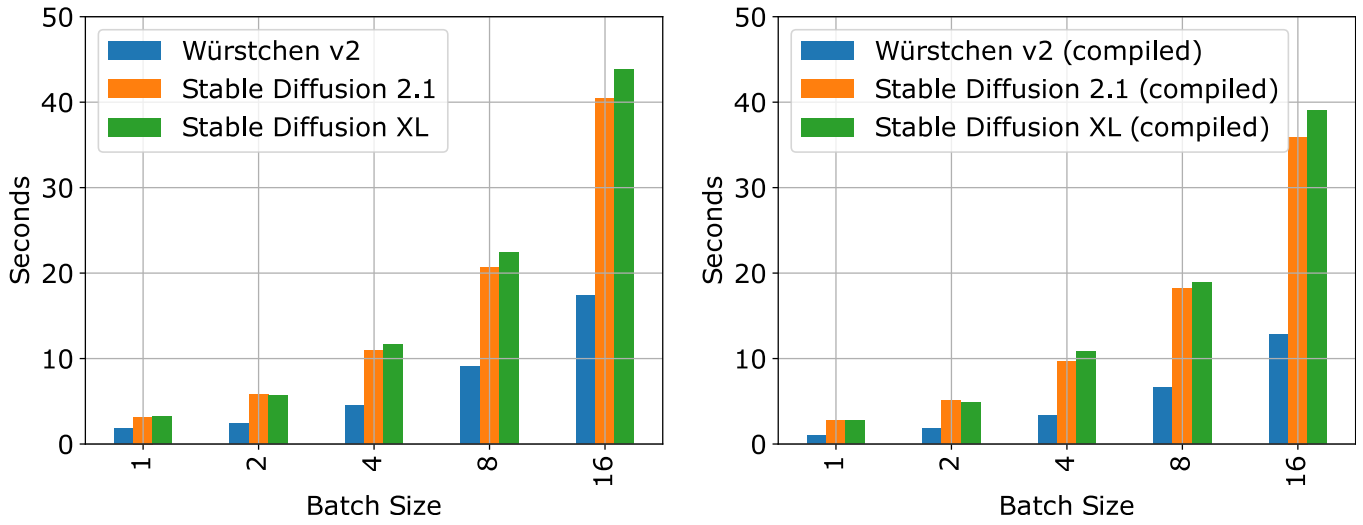
Würstchen (Pertinas et al., ICLR 2024)

- Training of this model is performed in reverse order of inference:
  - First the image decoder is trained (using a VQGAN objective)
  - Then we train another model to denoise a latent image from strongly compressed latents which are created by feeding the image into an EfficientNet encoder
  - Finally, we train another denoising model to construct the highly compressed latent image.



Würstchen: Inference time

- Balancing the image composition task (which is performed at low resolution using more diffusion steps) and the image superresolution task (needs less steps at higher resolution) yields a better compromise in computational performance.



Summary: Text-conditional Image Generation

Hochschule Flensburg
University of Applied Sciences

- While conditional GANs are a viable method to generate text-conditional images, recent success in diffusion models shows that these are superior for this task.
- Cascaded approaches can balance the computational load in training and inference towards composition (low scale) and adding details (high scale).
- Denoising can be performed in an auto-regressive way (using transformers) as well as using a plain encoder-decoder architecture (with cross-attention for conditioning).
- The availability of large-scale datasets has advanced the field of open source image generation models in recent years significantly.

Hochschule
Flensburg
University of
Applied Sciences

Applications

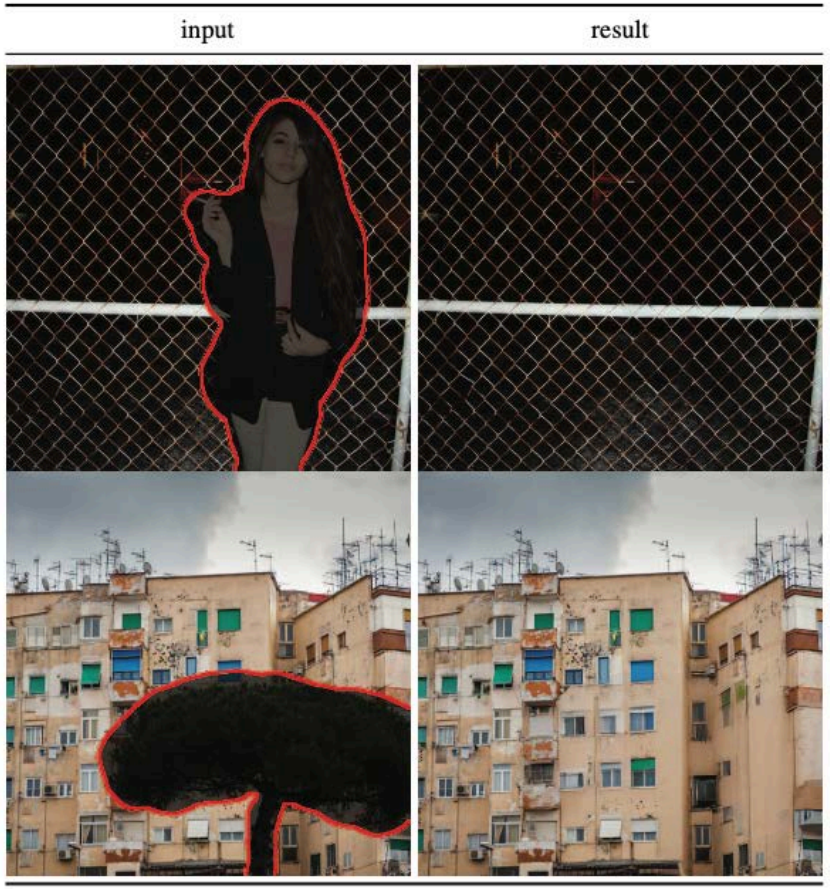
Super-Resolution

- One application for generative models is to create images that are much larger then the input image.
- We commonly refer to this application as "super-resolution"
- While there are many smart super-resolution approaches that do only utilize the data provided, generative superresolution uses the knowledge about common patterns in the training set.

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9878449

Inpainting

- Inpainting is the task of filling a defined part of the image with content.
- We do this because
  - Parts of the image are corrupted
  - We want to compose a new image.
- Inpainting can be only conditional to the remaining image, but also to additional text.

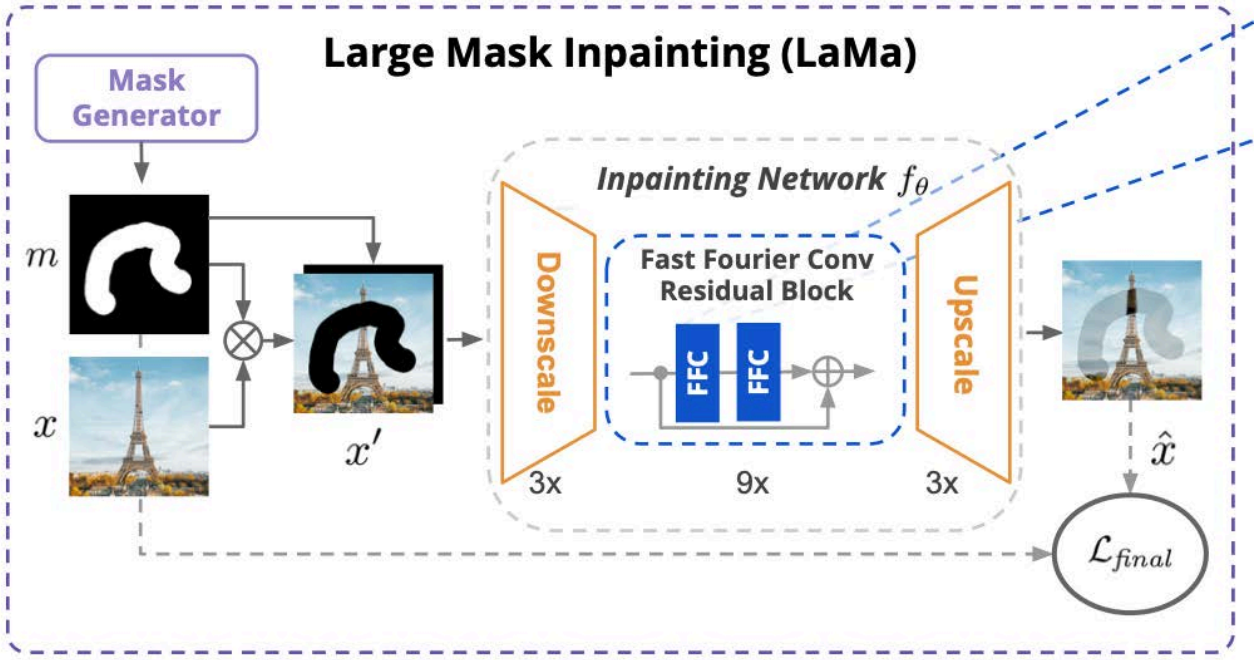


Hochschule Flensburg
University of Applied Sciences

Large Mask Inpainting (LaMa, Suvorov et al., 2021)

Hochschule Flensburg
University of Applied Sciences

- One of the first approaches in large-scale inpainting using masks.
- Was trained specifically on input images and masks generated for this task.
- Uses a latent space inpainting network for the generation of images.
- This approach was also followed in the Stable Diffusion model line to derive specific inpainting models.
  (https://huggingface.co/stabilityai/stable-diffusion-2-inpainting)



https://arxiv.org/pdf/2109.07161

Multi-Conditioning

- Besides conditioning on texts, we can also condition on image embeddings.
- Conditioning on images can also be used to generate image variations.

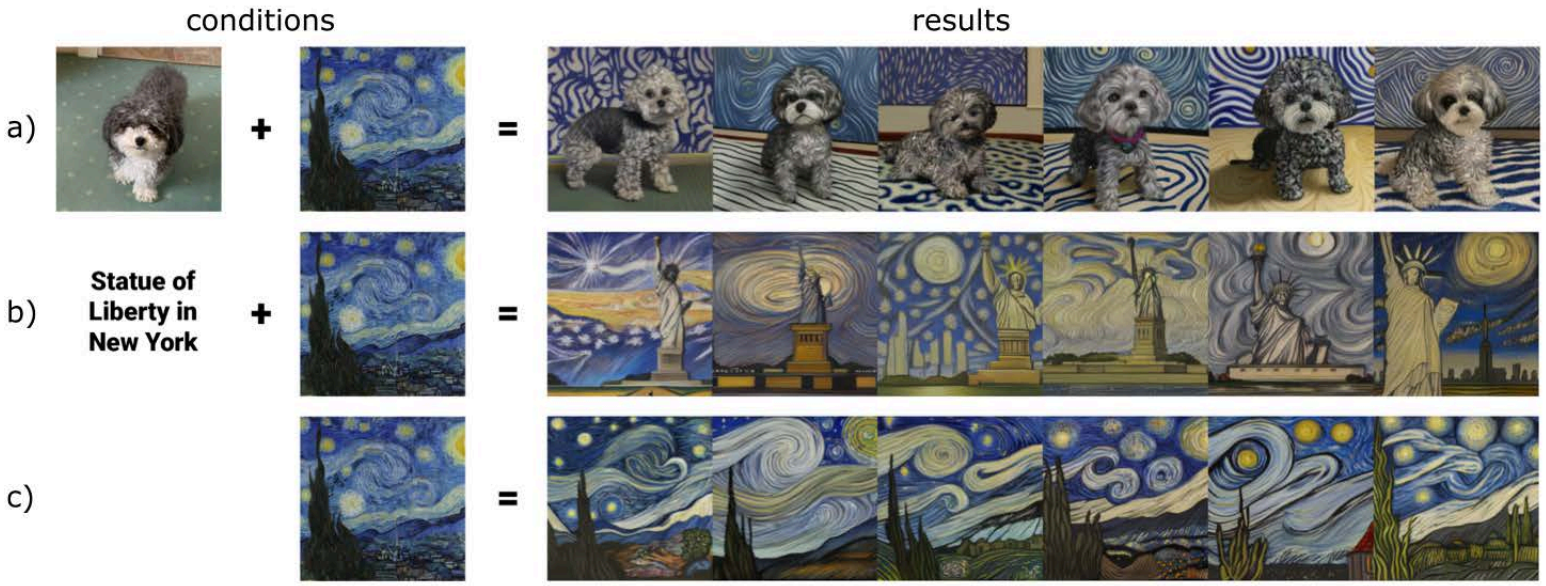
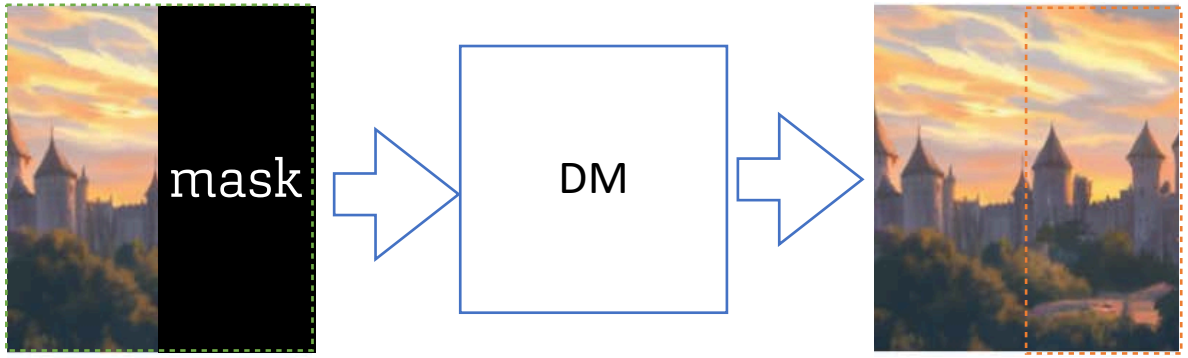


Figure 6: Image manipulation with Paella. a) shows image-conditional image generation, b) shows text-conditional image generation and c) shows image variations.

https://arxiv.org/pdf/2211.07292

Outpainting

- Now that we know that we can use masks to generate missing parts in an image, this can also be used in outpainting, i.e., image extension.



Local editing

- Besides mask-conditional image modifications, we can also perform image editing purely controlled by textual prompts.
- Prompts can specify what to change while preserving overall style, composition, and identity.
- Diffusion models can iteratively refine or “rewrite” specific semantic elements (e.g., replace sky with sunset, change clothing color, add objects).
- Guidance strength controls how strongly the model deviates from the original content.

(a) Input image

(b) *“remove the thing from her face”*

(c) *“she is now taking a selfie in the streets of Freiburg, it’s a lovely day out.”*

(d) *“it’s now snowing, everything is covered in snow.”*

Hochschule Flensburg
University of Applied Sciences

https://arxiv.org/pdf/2506.15742

Test your knowledge

Hochschule Flensburg
University of Applied Sciences

- What do the metrics Inception Score and Fréchet Inception Distance measure?
- Why are both not ideal for judging image quality?
- What's the main idea behind diffusion denoising probabilistic models?
- Why do diffusion models require so many steps for image generation?
- Why is it a good idea to leverage textual descriptions of images rather than image labels?
- What's the difference between few shot and zero shot applications?
- How can I classify an image with the CLIP model on a dataset with previously unknown classes?
- Why can't CLIP text embeddings be used directly to predict images?
- Why is it beneficial to do text-to-image in compressed latent spaces?
- Image denoising can be performed using a U-Net-style model or an auto-regressive VQ-GAN-style transformer approach. Which is more beneficial for large images and why?