ETH zürich

High-Resolution Image Synthesis with Latent Diffusion Models (Stable Diffusion)

Robin Rombach, Andreas Blattman, Dominik Lorenz, Patrick Esser, Björn Ommer

Presented by: Marcus Roberto Nielsen

Seminar – Advanced Topics in Machine Learning and Data Science

Date: 27.03.2024

Generative models for image synthesis

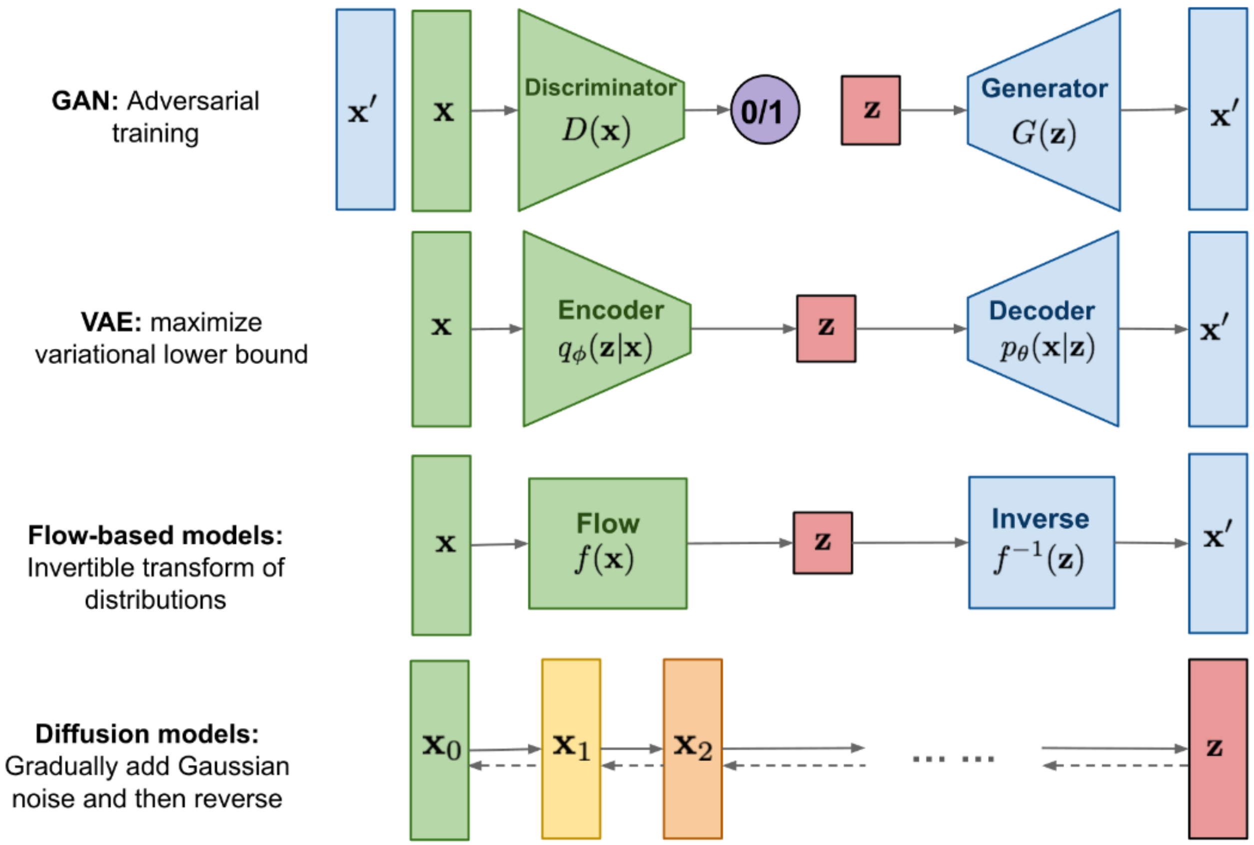


Fig. 1. Overview of different types of generative models.

Image credit: Weng, Lilian. (Jul. 2021)

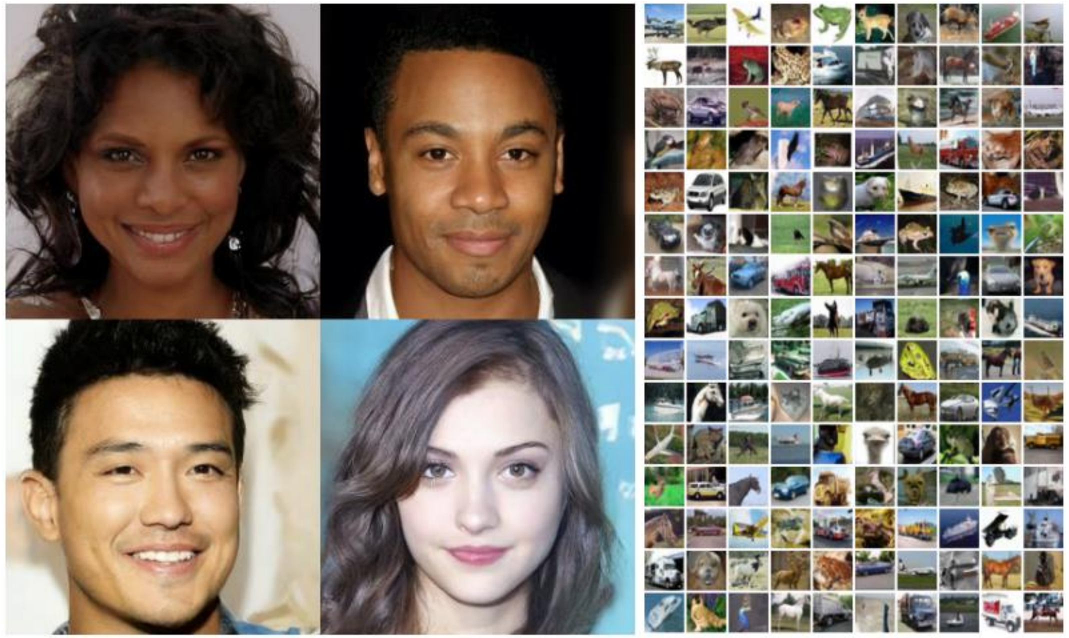


Figure 1: Generated samples on CelebA-HQ 256×256 (left) and unconditional CIFAR10 (right)

Image credit: Ho, et.al. (2020)

Background – Diffusion Model (DM)

Goal: *Learn data distribution* $\mathbf{x}_0 \sim q(\mathbf{x}_0)$

Forward diffusion process

- Fixed to a Markov chain

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

- Adds Gaussian noise in each timestep

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

- Admits sampling at arbitrary timestep in closed form

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

$$\alpha_t := 1 - \beta_t \qquad \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$$

Everything is Gaussian

Reverse process

- Defined as a Markov chain

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

- Diffusion Model is defined as

$$p_\theta(\mathbf{x}_0) := \int p_\theta(\mathbf{x}_{0:T})\, d\mathbf{x}_{1:T}$$

- Gaussian transitions in each timestep

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

$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$

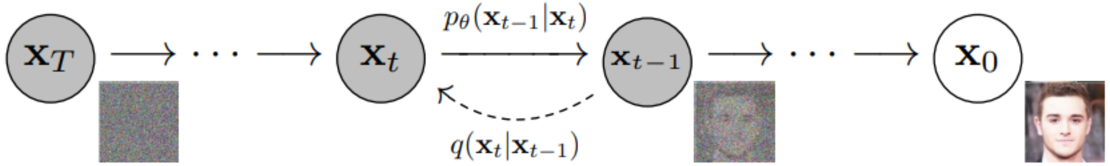


Figure 2: The directed graphical model considered in this work.

Image credit: Ho, et.al. (2020)

Background – Diffusion model

Training

- Variational bound on negative log likelihood: $\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t\geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$
- Possible to decompose training objective: $\mathbb{E}_q\Big[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)\,\|\,p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \boxed{\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)\,\|\,p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\Big]$
- Condition $q(x_{t-1}|x_t)$ on x_0 (Bayes Theorem) $q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0), \tilde{\beta}_t\mathbf{I})$

$$\approx$$

- Parameterization of reverse process $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t,t), \sigma_t^2\mathbf{I})$
- KL-divergence between two Gaussians $L_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t,t)\|^2\right] + C$

Background – Diffusion model

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

- Reparameterization trick $\mathbf{x}_t(\mathbf{x}_0, \boldsymbol{\epsilon}) = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \text{ for } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

- Rewrite parameterization $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t))\right) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$

$$L_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2\right] \rightarrow \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

Simplified objective

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

Background – Diffusion model

Recall parameterization of reverse process and training objective

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I})$$

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$$

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

Algorithm 1 Training

1: **repeat**
2: $\quad \mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $\quad t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: $\quad$ 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

Algorithm 2 Sampling

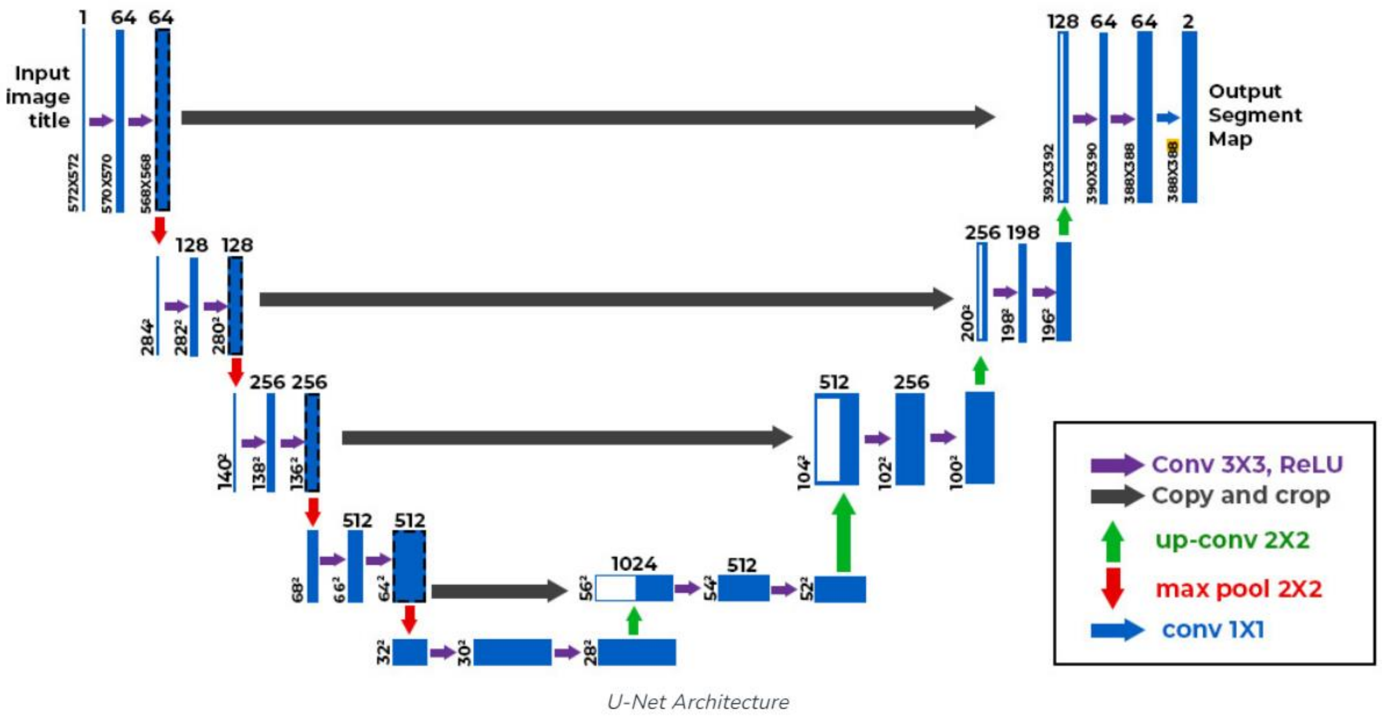
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,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

Image credit: Ho, et.al. (2020)

UNet

- *Contracting path*
  - encoder layers
  - capture contextual information
  - reduce the spatial resolution

- *Expansive path*
  - decoder layers
  - decode encoded information



U-Net Architecture

Image credit: Aditya Taparia (2023)

- Due to the UNet backbone of DMs, they offer excellent inductive biases for spatial data

Motivation – Latent Diffusion Models

- Diffusion models achieves state-of-the-art synthesis results on image data
- Powerful, yet simple model architecture

Problems

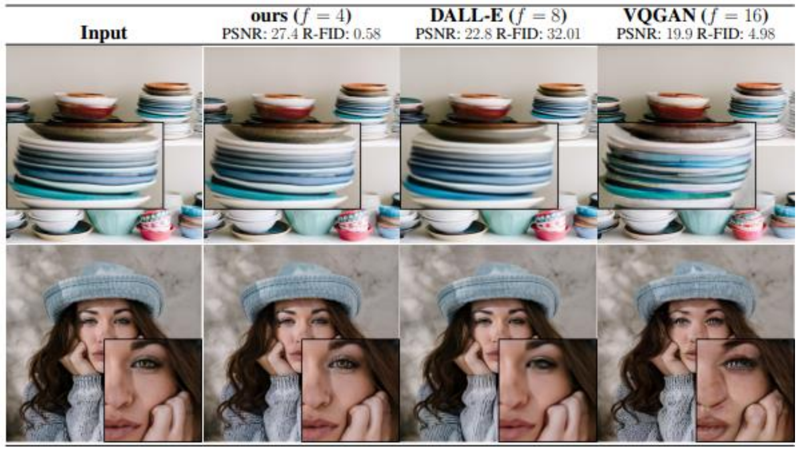
- Mode-covering behaviour (likelihood-based model)
- Operates directly in the high-dimensional pixel space
- Requires massive computational resources
- Expensive in time and memory

Proposed Method: *Latent Diffusion Models*

- Operates in a lower-dimensional latent space
- Reduces resource consumption for both training and sampling
- Detail preservation

Main contributions of the paper

1. LDMs scales more gracefully to higher dimensional data
2. Reducing computational costs, while retaining competitive performance
3. Reducing inference costs compared to pixel-based diffusion approaches
4. Does not require a delicate weighting of reconstruction and generative abilities
5. Can be applied in a convolutional fashion
6. Enables multi-modal training via cross-attention



LDMs require less aggressive downsampling

Image credit: Rombach, et.al. (2022)

Analysis of trained Diffusion Models in pixel space

Two-stage learning process:

1. ***Perceptual compression***
   - Removes high-frequency details
   - Learns a little semantic variation
2. ***Semantic compression***
   - Learns the semantic and conceptual composition of the data

Idea:

- *Find a perceptually equivalent, but computationally more suitable space to train diffusion models for high-resolution image synthesis*

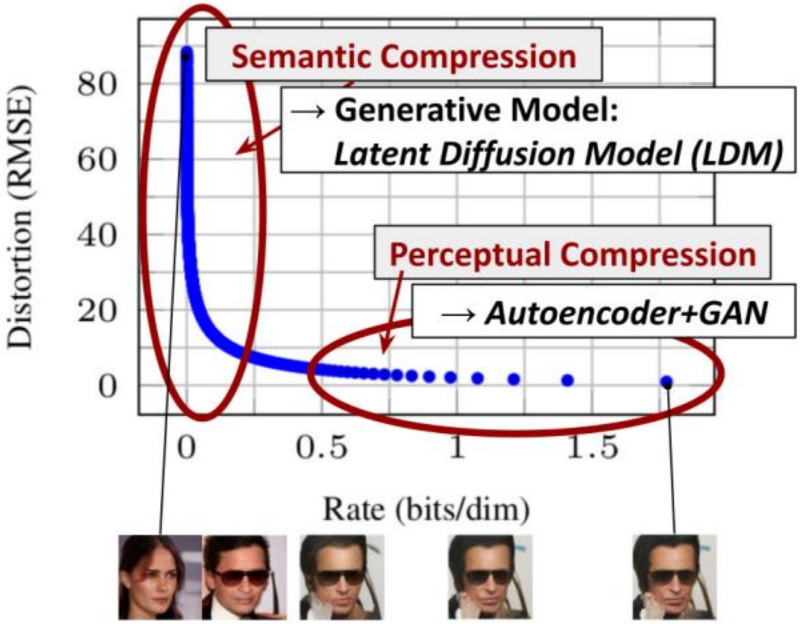


Image credit: Rombach, et.al. (2022)

Latent diffusion models (LDMs)

Goal: *Learn data distribution* $z_0 \sim p(z_0)$

Two-phased learning process

1. Train an autoencoder to obtain a low-dimensional latent space

Pixel-space — Latent space — Pixel-space

$$x \in \mathbb{R}^{H \times W \times 3} \xrightarrow{\mathcal{E}(x)} z \in \mathbb{R}^{h \times w \times c} \xrightarrow{\mathcal{D}(z)} \tilde{x} \in \mathbb{R}^{H \times W \times 3}$$

Downsampling factor: $f = H/h = W/w$

Model architecture

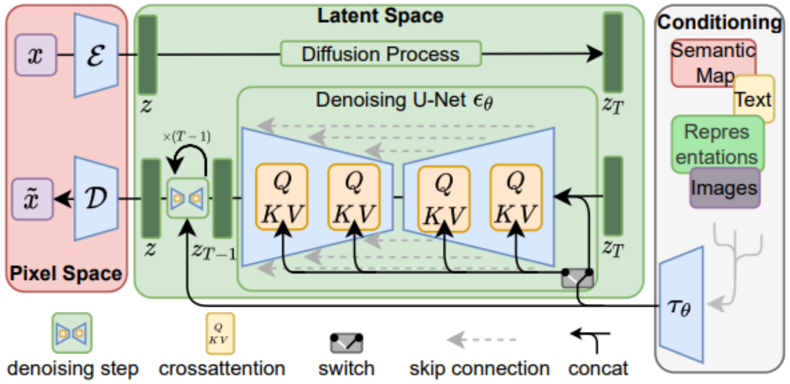


Image credit: Rombach, et.al. (2022)

2. Train DMs in the learned latent space

Training objective for DMs

$$L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\right]$$

Pixel-space

Training objective for LDMs

$$L_{LDM} := \mathbb{E}_{\mathcal{E}(x),\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(z_t, t)\|_2^2\right]$$

Latent space

Autoencoder model – Regularization of latent space

Adversarial training objective

$$L_{\text{Autoencoder}} = \min_{\mathcal{E},\mathcal{D}} \max_{\psi} \left(L_{rec}(x, \mathcal{D}(\mathcal{E}(x))) - L_{adv}(\mathcal{D}(\mathcal{E}(x))) + \log D_{\psi}(x) + L_{reg}(x; \mathcal{E}, \mathcal{D}) \right)$$

Reconstruction loss: $L_{rec}(x, \mathcal{D}(\mathcal{E}(x)))$

Adversarial loss: $-L_{adv}(\mathcal{D}(\mathcal{E}(x))) + \log D_{\psi}(x)$

Regularization term: $L_{reg}(x; \mathcal{E}, \mathcal{D})$

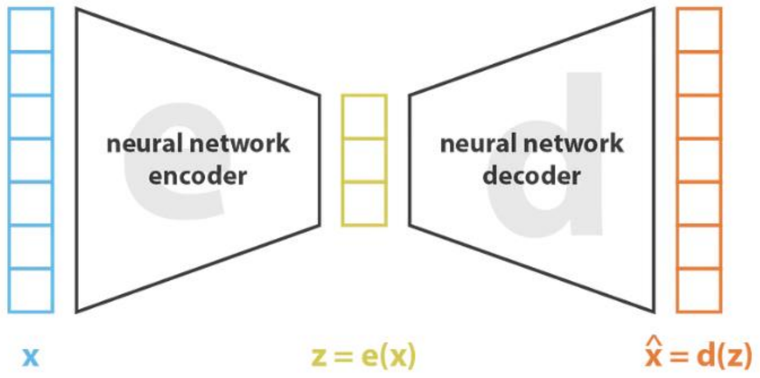


Image credit: Joseph Rocca (2019)

KL-regularization

- Imposes a slight KL-penalty towards a standard normal on the learned latent

VQ-regularization

- Learns a codebook of |Z| different exemplars

Transformers and Cross-Attention

- Attention mechanism

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

- Multi-head Attention

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

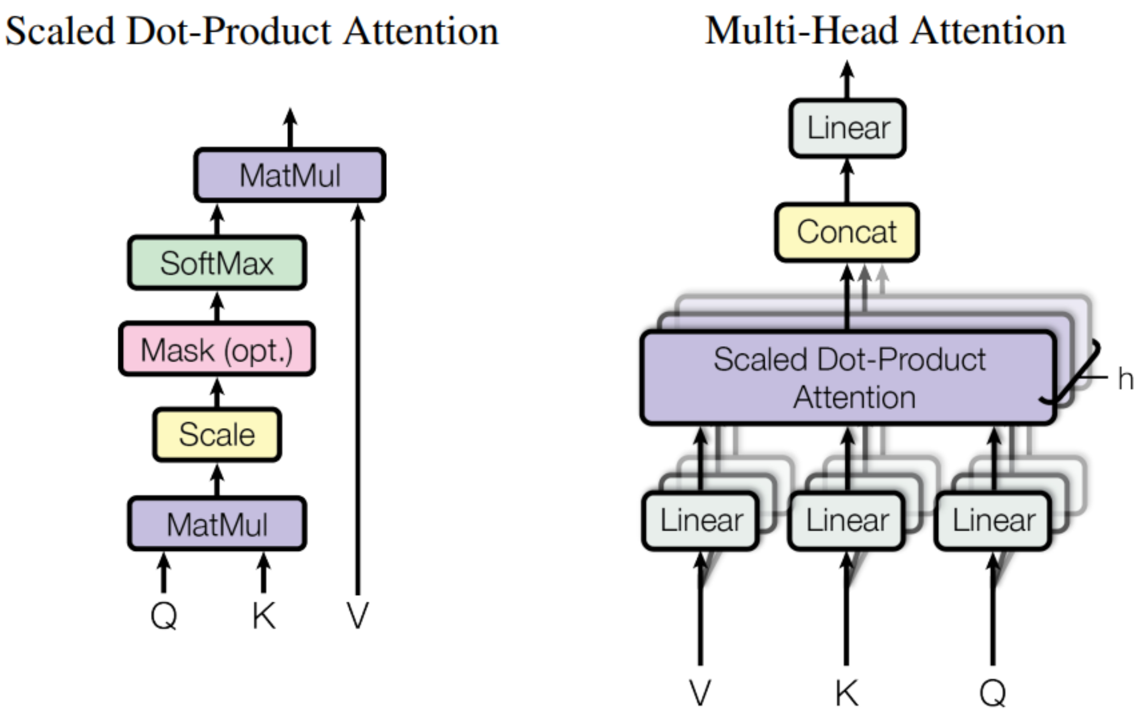


Image credit: Vaswani, et.al. (2017)

Conditional Latent Distance Models

Conditioning mechanisms

- DMs can also model conditional distributions $p(z|y)$
- Pre-processing → domain specific encoder
- Augment the UNet backbone with a cross-attention mechanism

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)\cdot V, \text{with}$$

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

- **Training objective**

$$L_{LDM} := \mathbb{E}_{\mathcal{E}(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \boxed{\epsilon_\theta(z_t,t,\tau_\theta(y))}\|_2^2\right]$$

Model architecture

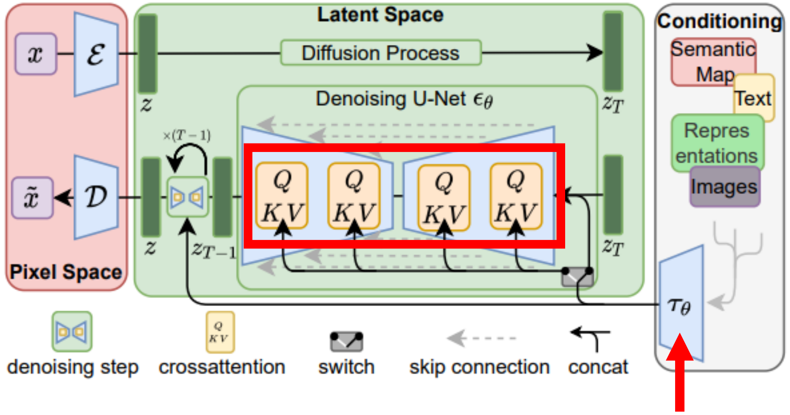
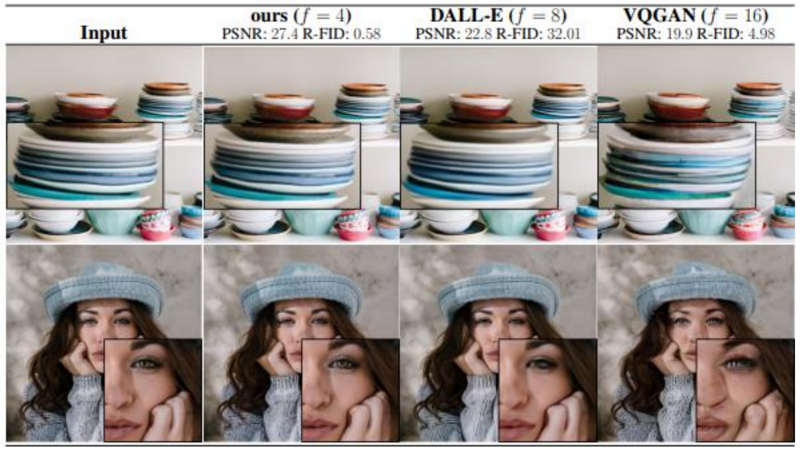


Image credit: Rombach, et.al. (2022)

Advantages of the learning process for Latent Diffusion Models

1. Train the universal autoencoding stage only once
2. Does not require excessive spatial compression
3. Efficient image generation from the latent space with a single network pass
4. Does not require a delicate weighting of reconstruction and generative abilities
5. Reduces computational demands
6. Exploits the inductive bias of DMs
7. Obtain general-purpose compression models



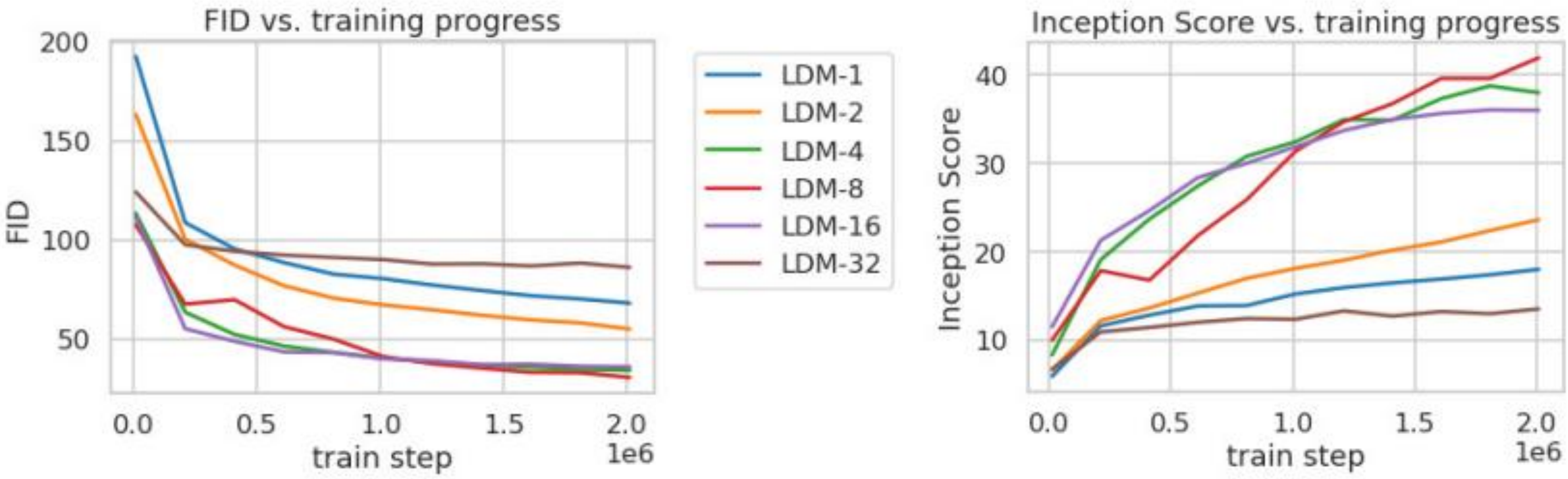
LDMs require less aggressive downsampling

Image credit: Rombach, et.al. (2022)

Experiments – Perceptual Compression Tradeoffs

Training

Evaluated on the ImageNet dataset



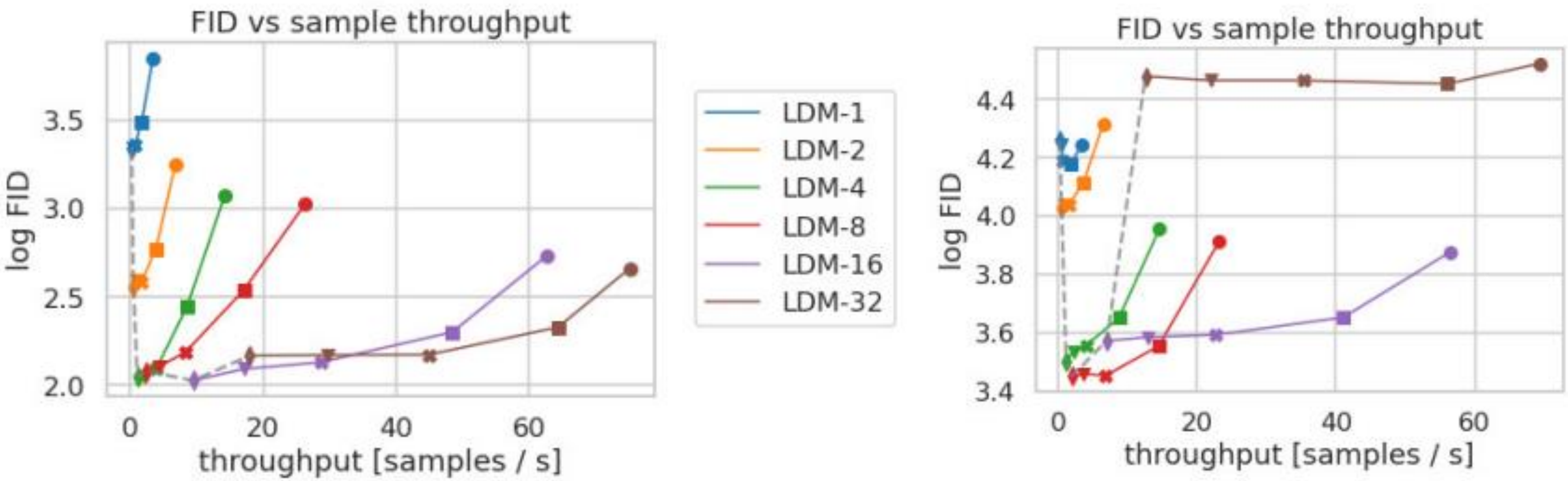
- Low perceptual compression (i.e. low f) → large train times
- High perceptual compression (i.e. high f) → limits overall sample quality

- Optimal compression tradeoff: LDM-{4-16}

Experiments – Perceptual Compression Tradeoffs

Sampling

CelebA-HQ dataset *ImageNet dataset*



Different markers indicate {10,20,50,100,200} sampling steps using DDIM from right to left

- Low perceptual compression (i.e. low f) → lower sample throughput
- High perceptual compression (i.e. high f) → limits overall sample quality, higher sample throughput

- Optimal compression rate: LDM-{4-16} (left) and LDM-{4-8} (right)

Experiments – Effects of regularization

KL-Regularization vs. VQ-Regularization

- Better reconstruction capabilities (KL)
- Better sample quality (KL)

f	$\|\mathcal{Z}\|$	c	R-FID ↓	R-IS ↑	PSNR ↑	PSIM ↓	SSIM ↑
16 *VQGAN* [23]	16384	256	4.98	–	19.9 ± 3.4	1.83 ± 0.42	0.51 ± 0.18
16 *VQGAN* [23]	1024	256	7.94	–	19.4 ± 3.3	1.98 ± 0.43	0.50 ± 0.18
8 *DALL-E* [66]	8192	-	32.01	–	22.8 ± 2.1	1.95 ± 0.51	0.73 ± 0.13
32	16384	16	31.83	40.40 ± 1.07	17.45 ± 2.90	2.58 ± 0.48	0.41 ± 0.18
16	16384	8	5.15	144.55 ± 3.74	20.83 ± 3.61	1.73 ± 0.43	0.54 ± 0.18
8	16384	4	1.14	201.92 ± 3.97	23.07 ± 3.99	1.17 ± 0.36	0.65 ± 0.16
8	256	4	1.49	194.20 ± 3.87	22.35 ± 3.81	1.26 ± 0.37	0.62 ± 0.16
4	8192	3	0.58	224.78 ± 5.35	27.43 ± 4.26	0.53 ± 0.21	0.82 ± 0.10
$4^{\dagger}$	8192	3	1.06	221.94 ± 4.58	25.21 ± 4.17	0.72 ± 0.26	0.76 ± 0.12
4	256	3	0.47	223.81 ± 4.58	26.43 ± 4.22	0.62 ± 0.24	0.80 ± 0.11
2	2048	2	0.16	232.75 ± 5.09	30.85 ± 4.12	0.27 ± 0.12	0.91 ± 0.05
2	64	2	0.40	226.62 ± 4.83	29.13 ± 3.46	0.38 ± 0.13	0.90 ± 0.05
32	KL	64	2.04	189.53 ± 3.68	22.27 ± 3.93	1.41 ± 0.40	0.61 ± 0.17
32	KL	16	7.3	132.75 ± 2.71	20.38 ± 3.56	1.88 ± 0.45	0.53 ± 0.18
16	KL	16	0.87	210.31 ± 3.97	24.08 ± 4.22	1.07 ± 0.36	0.68 ± 0.15
16	KL	8	2.63	178.68 ± 4.08	21.94 ± 3.92	1.49 ± 0.42	0.59 ± 0.17
8	KL	4	0.90	209.90 ± 4.92	24.19 ± 4.19	1.02 ± 0.35	0.69 ± 0.15
4	KL	3	0.27	227.57 ± 4.89	27.53 ± 4.54	0.55 ± 0.24	0.82 ± 0.11
2	KL	2	0.086	232.66 ± 5.16	32.47 ± 4.19	0.20 ± 0.09	0.93 ± 0.04

Table 8. Complete autoencoder zoo trained on OpenImages, evaluated on ImageNet-Val. † denotes an attention-free autoencoder.

Experiments – Image Generation with Latent Diffusion

Setup

- Train unconditional models of 256^2 images on CelebA-HQ, FFHQ, LSUN-Churches and – Bedrooms dataset

Evaluation metrics

1. sample quality (FID score)
2. coverage of data manifold (Precision and Recall)

CelebA-HQ 256 × 256			
Method	FID ↓	Prec. ↑	Recall ↑
DC-VAE [63]	15.8	-	-
VQGAN+T. [23] (k=400)	10.2	-	-
PGGAN [39]	8.0	-	-
LSGM [93]	7.22	-	-
UDM [43]	7.16	-	-
LDM-4 (ours, 500-s†)	**5.11**	0.72	0.49

FFHQ 256 × 256			
Method	FID ↓	Prec. ↑	Recall ↑
ImageBART [21]	9.57	-	-
U-Net GAN (+aug) [77]	10.9 (7.6)	-	-
UDM [43]	5.54	-	-
StyleGAN [41]	4.16	0.71	0.46
ProjectedGAN [76]	**3.08**	0.65	0.46
LDM-4 (ours, 200-s)	4.98	**0.73**	**0.50**

LSUN-Churches 256 × 256			
Method	FID ↓	Prec. ↑	Recall ↑
DDPM [30]	7.89	-	-
ImageBART [21]	7.32	-	-
PGGAN [39]	6.42	-	-
StyleGAN [41]	4.21	-	-
StyleGAN2 [42]	3.86	-	-
ProjectedGAN [76]	**1.59**	0.61	0.44
*LDM-8** (ours, 200-s)	4.02	**0.64**	**0.52**

LSUN-Bedrooms 256 × 256			
Method	FID ↓	Prec. ↑	Recall ↑
ImageBART [21]	5.51	-	-
DDPM [30]	4.9	-	-
UDM [43]	4.57	-	-
StyleGAN [41]	2.35	0.59	0.48
ADM [15]	1.90	**0.66**	**0.51**
ProjectedGAN [76]	**1.52**	0.61	0.34
LDM-4 (ours, 200-s)	2.95	**0.66**	0.48

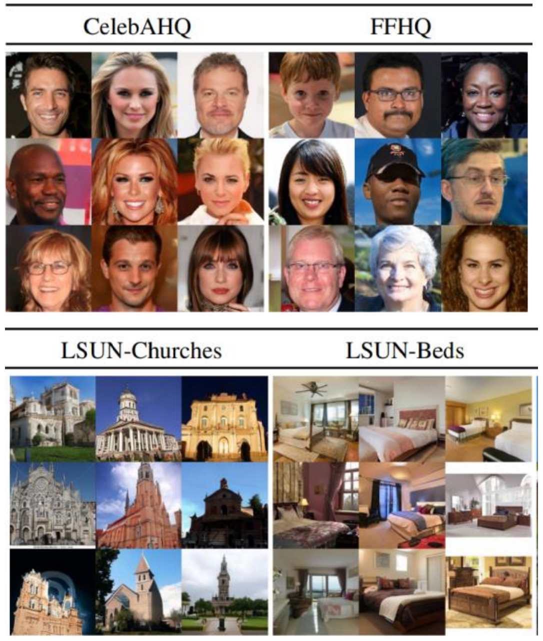


Image credit: Rombach, et.al. (2022)

Experiments – Conditional Latent Diffusion

Class-conditional ImageNet

- Using downsampling factor f=4

ImageNet

Image credit: Rombach, et.al. (2022)

Method	FID↓	IS↑	Precision↑	Recall↑	N_{params}	
BigGan-deep [3]	6.95	$203.6_{\pm 2.6}$	**0.87**	0.28	340M	-
ADM [15]	10.94	100.98	0.69	**0.63**	554M	250 DDIM steps
ADM-G [15]	4.59	186.7	0.82	0.52	608M	250 DDIM steps
LDM-4 (ours)	10.56	$103.49_{\pm 1.24}$	0.71	0.62	400M	250 DDIM steps
LDM-4-G (ours)	**3.60**	**$247.67_{\pm 5.59}$**	**0.87**	0.48	400M	250 steps, c.f.g [32], $s = 1.5$

Experiments – Conditional Latent Diffusion

Transformer Encoders for LDMs

- Text-to-image modeling
  - 1.45B parameter KL-regularized LDM conditioned on language prompts
  - BERT-tokenizer
  - Domain specific encoder as a transformer
  - MS-COCO validation set

Text-to-Image Synthesis on LAION. 1.45B Model.

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

Text-Conditional Image Synthesis

Method	FID ↓	IS↑	N_{params}	
CogView† [17]	27.10	18.20	4B	self-ranking, rejection rate 0.017
LAFITE† [109]	26.94	<u>26.02</u>	75M	
GLIDE* [59]	<u>12.24</u>	-	6B	277 DDIM steps, c.f.g. [32] $s = 3$
Make-A-Scene* [26]	**11.84**	-	4B	c.f.g for AR models [98] $s = 5$
LDM-KL-8	23.31	$20.03_{\pm 0.33}$	1.45B	250 DDIM steps
*LDM-KL-8-G**	12.63	$\mathbf{30.29}_{\pm 0.42}$	1.45B	250 DDIM steps, c.f.g. [32] $s = 1.5$

Experiments – Conditional Latent Diffusion

Convolutional Sampling Beyond 256^2

- Concatenate spatially aligned conditioning information to the input of ϵ_θ
- LDMs can serve as a general purpose image-to-image translation model
- Useful for semantic synthesis, super-resolution and image inpainting

Figure 9. A *LDM* trained on 256^2 resolution can generalize to larger resolution (here: 512×1024) for spatially conditioned tasks such as semantic synthesis of landscape images. See Sec. 4.3.2.

Image credit: Rombach, et.al. (2022)

Experiments – Conditional Latent Diffusion

Super-Resolution with Latent Diffusion (LDM-SR)

- Condition on low-resolution images via concatenation, i.e. τ_θ is the identity function
- Fix the image degradation to a bicubic interpolation with 4x downsampling
- Autoencoding model pretrained on OpenImages (VQ-reg.)

Figure 10. ImageNet 64→256 super-resolution on ImageNet-Val. *LDM-SR* has advantages at rendering realistic textures but SR3 can synthesize more coherent fine structures. See appendix for additional samples and cropouts. SR3 results from [72].

Image credit: Rombach, et.al. (2022)

Method	FID ↓	IS ↑	PSNR ↑	SSIM ↑	N_{params}	$[\frac{\text{samples}}{s}](*)$
Image Regression [72]	15.2	121.1	**27.9**	**0.801**	625M	N/A
SR3 [72]	5.2	**180.1**	26.4	0.762	625M	N/A
LDM-4 (ours, 100 steps)	$2.8^{\dagger}/4.8^{\ddagger}$	166.3	24.4 ± 3.8	0.69 ± 0.14	**169M**	4.62
emphLDM-4 (ours, big, 100 steps)	$\mathbf{2.4}^{\dagger}/\mathbf{4.3}^{\ddagger}$	174.9	24.7 ± 4.1	0.71 ± 0.15	552M	4.5
LDM-4 (ours, 50 steps, guiding)	$4.4^{\dagger}/6.4^{\ddagger}$	153.7	25.8 ± 3.7	0.74 ± 0.12	184M	0.38

	SR on ImageNet	
User Study	Pixel-DM ($f1$)	*LDM-4*
Task 1: Preference vs GT ↑	16.0%	**30.4%**
Task 2: Preference Score ↑	29.4%	**70.6%**

Experiments – Conditional Latent Diffusion

Image Inpainting with LDMs

- Latent diffusion models improves sample throughput and sample quality for image inpainting tasks

Model (*reg.*-type)	train throughput samples/sec.	sampling throughput† @256	@512	train+val hours/epoch	FID@2k epoch 6
LDM-1 (no first stage)	0.11	0.26	0.07	20.66	24.74
LDM-4 (*KL*, w/ attn)	0.32	0.97	0.34	7.66	15.21
LDM-4 (*VQ*, w/ attn)	0.33	0.97	0.34	7.04	14.99
LDM-4 (*VQ*, w/o attn)	0.35	0.99	0.36	6.66	15.95

User Study	Inpainting on Places LAMA [88]	*LDM-4*
Task 1: Preference vs GT ↑	13.6%	**21.0%**
Task 2: Preference Score ↑	31.9%	**68.1%**

Figure 11. Qualitative results on object removal with our *big, w/ ft* inpainting model. For more results, see Fig. 22.

Image credit: Rombach, et.al. (2022)

Related work

Generative Models for image synthesis

- Generative Adversarial Networks (GANs)

  + efficient sampling of high-dimensional images and good perceptual quality

  - difficult to optimize and doesn’t capture full data distribution (mode-collapse)

- Variational autoencoders (VAEs)

  + efficient synthesis of images

  - worse sample quality than GANs

- Autoregressive models (ARM)

  + strong performance in density estimation

  - computationally demanding architecture and sequential sampling process

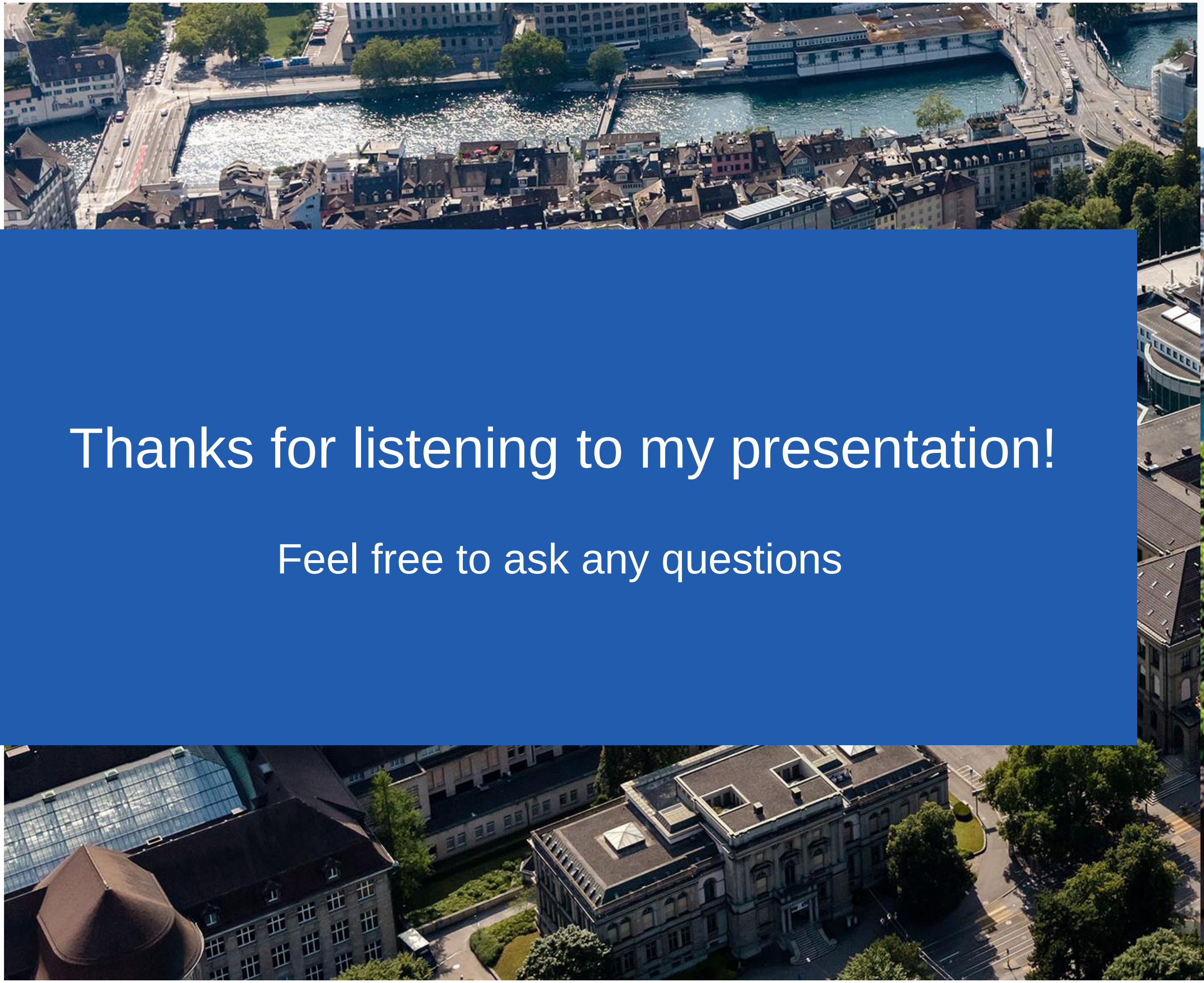
Assessment of paper and model

Pros

- Ground-breaking paper in the field of image synthesis
- Offers a method for powerful high-resolution image generation using fewer computational resources
- Applications in a diverse range of settings (cross-attention and conditioning mechanisms)
  - Text-to-image generation
  - Image inpainting
  - Image super-resolution

Cons

- Some segments are explained very briefly with few details regarding implementation
- Slower sampling speed compared to GANs
- Limited performance in cases where fine-grained accuracy in pixel-space is crucial (e.g. superresolution)



Thanks for listening to my presentation!

Feel free to ask any questions