High-Resolution Image Synthesis with Latent Diffusion Models

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

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

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

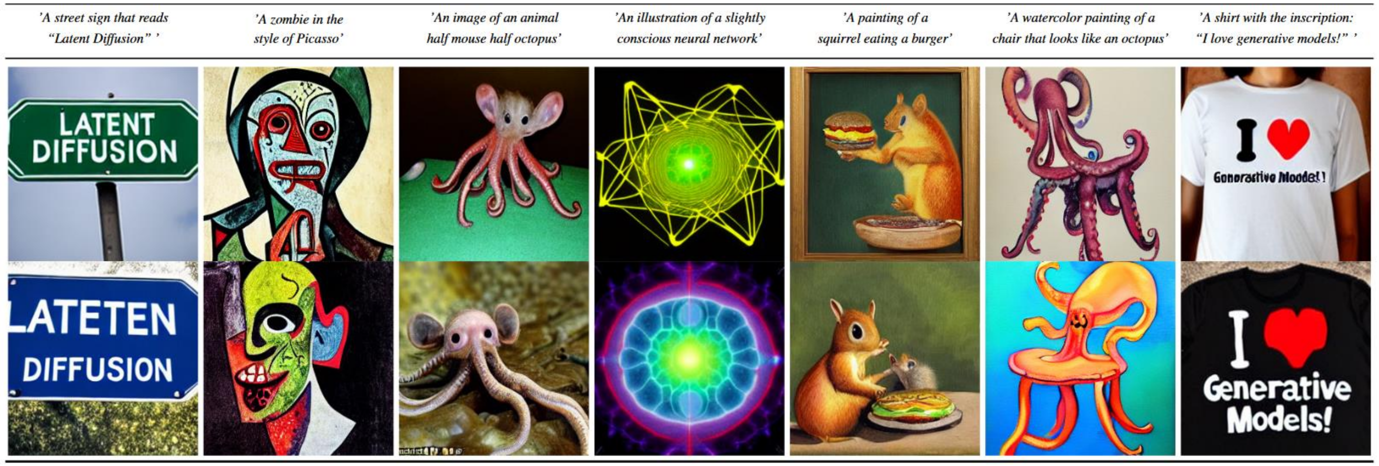
Research behind Stable Diffusion

Stable Diffusion Public Release

It is our pleasure to announce the public release of stable diffusion following our release for researchers [https://stability.ai/blog/stable-diffusion-announcement]

Image Synthesis

- Settings
  - Conditional Image Generation
    - Text-to-Img —— DALLE Series, DaVinci, Stable Diffusion
    - Inpainting —— LAMA
    - Super Resolution —— SR3
    - ……



Text2Image Example

Inpainting Example

Diffusion Models

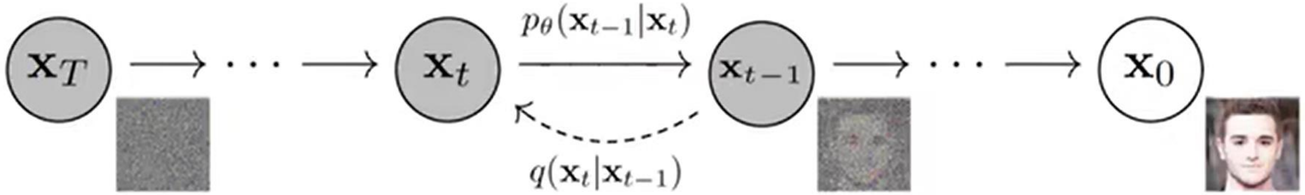
- DDPM
  - Noising and Denoising, Markov Chain

➢ Noising, No training required

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

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

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



➢ Denoising: Probabilistic Prediction

- Target: $L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\right]$
- $\boldsymbol{\varepsilon_\theta}$: U-Net
  - Whole Image **to Whole Image**

Algorithm 1 Training

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

Algorithm 2 Sampling

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

Diffusion Models

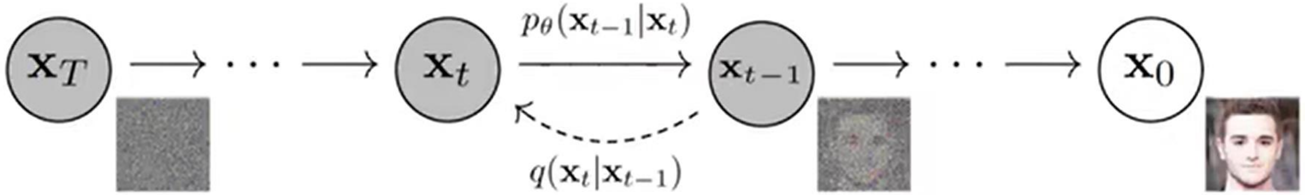
- DDPM
  - Noising and Denoising, Markov Chain

➢Noising, No training required

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

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

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



$$\begin{aligned}
q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) &= q(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \\
&\propto \exp\Big(-\frac{1}{2}\big(\frac{(\mathbf{x}_t-\sqrt{\alpha_t}\mathbf{x}_{t-1})^2}{\beta_t} + \frac{(\mathbf{x}_{t-1}-\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t-\sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big) \\
&= \exp\Big(-\frac{1}{2}\big((\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})\mathbf{x}_{t-1}^2 - (\frac{2\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{2\sqrt{\bar{\alpha}_t}}{1-\bar{\alpha}_t}\mathbf{x}_0)\mathbf{x}_{t-1} + C(\mathbf{x}_t,\mathbf{x}_0)\big)\Big)
\end{aligned}$$

where $C(\mathbf{x}_t,\mathbf{x}_0)$ is some function not involving $\mathbf{x}_{t-1}$ and details are omitted. Following the standard Gaussian density function, the mean and variance can be parameterized as follows:

$$\tilde{\beta}_t = 1/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t$$

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0) = (\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_t}}{1-\bar{\alpha}_t}\mathbf{x}_0)/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0$$

➢Denoising: Probabilistic Prediction

- Target: $L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon-\epsilon_\theta(x_t,t)\|_2^2\right]$
- $\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}$: U-Net
  - Whole Image **to Whole Image**

Algorithm 1 Training

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

Algorithm 2 Sampling

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

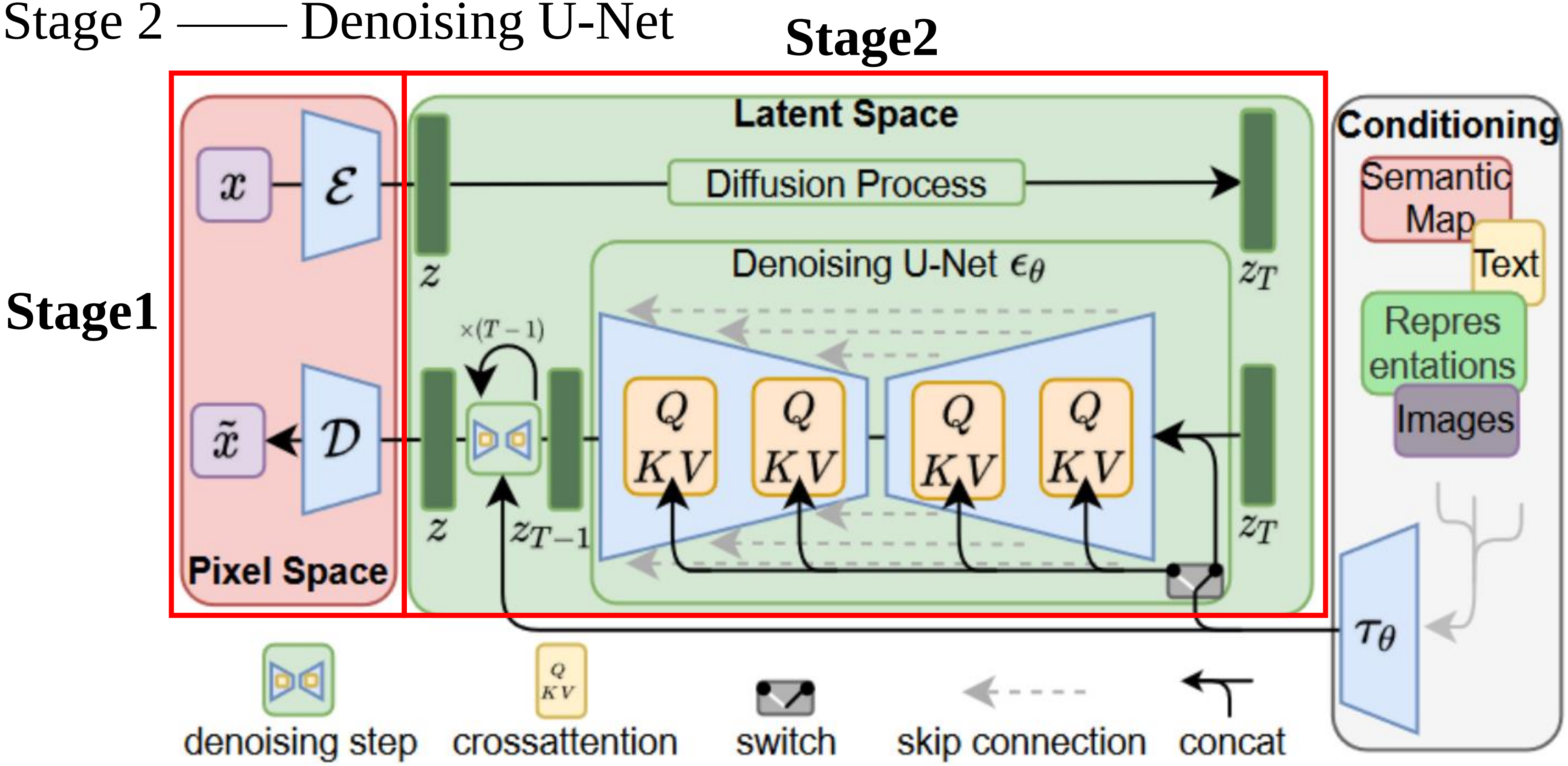
Drawbacks and Improvements

- Slow — Thousands of A100 hours for training, several hours for evaluation
  - ……
  - **DDIM：20-50 times speed up!**
    - DDPM time steps T = 1000
    - DDIM ***dim(T)* =** T/20

- Resolution Limitations
  - U-Net for whole image * T steps
  - Solution: Denoising on latent space —— LDM(Latent Diffusion Models)

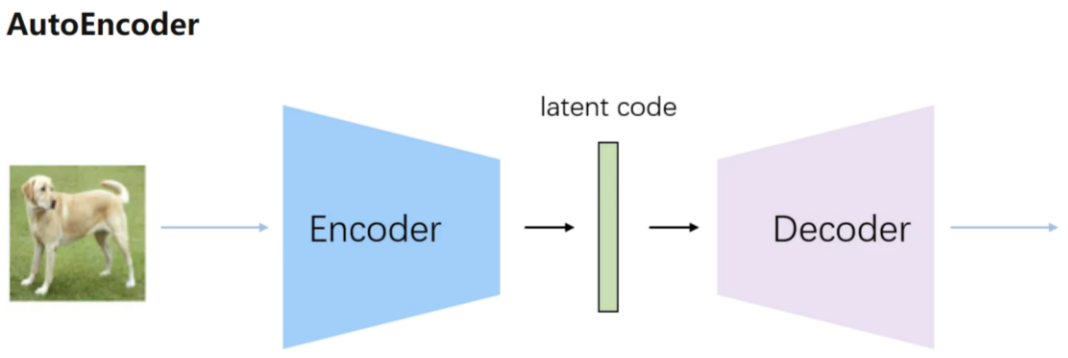
LDM

- Two Stage Training for both **Pixel Space** and **Latent Space**
- Stage 1 —— Image AutoEncoder
- Stage 2 —— Denoising U-Net



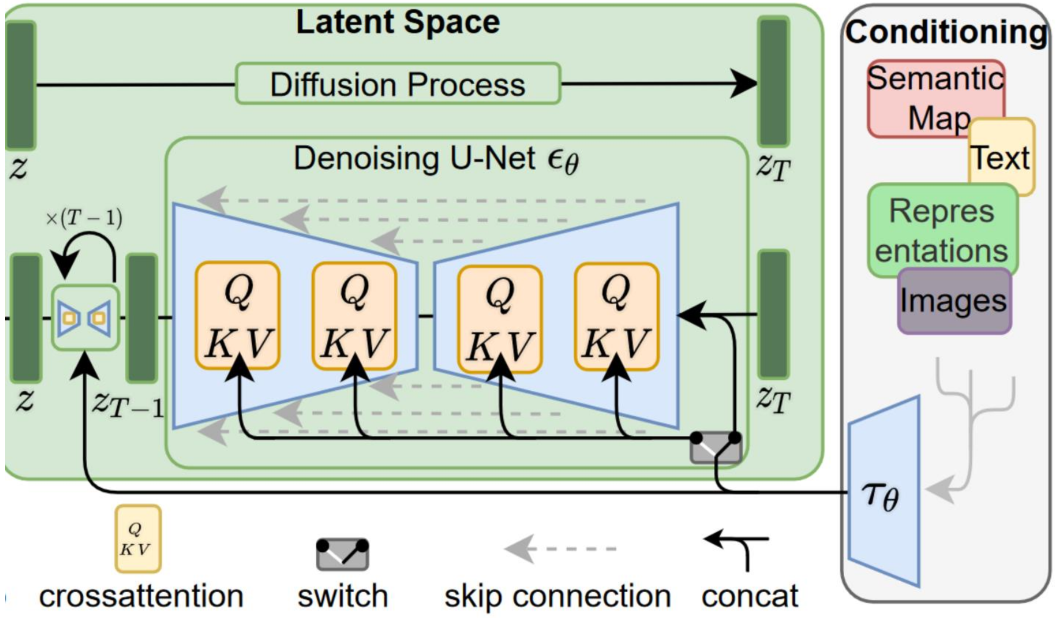
LDM

- Stage 1 —— Perceptual Image Compression
  - Encoder – Decoder model
    - Image Generation from statistics encoding and generation from distribution
    - Latent code is not high level feature(Z_c=4)
    - Load in an Gaussian Distribution and send into decoder to get image output
  - Constraint Target (Loss Function)
    - Reconstruction Loss (L2 for original image and reconstructed one)
    - LPIPS Loss (Perceptual loss for better reality)
      - Pretrained VGG 16
    - KL Loss for constraints on Latent Space



LDM

- Stage 2 —— Latent Diffusion Models
  - With Stage 1 Autoencoder frozen
  - Effect on different size latent code (with different rate of encoder)
  - Structure
    - Resnet Block with cross-attention module
    - U-Net like
  - Diffusion
    - Target:
    - DDPM $L_{LDM} := \mathbb{E}_{\mathcal{E}(x),\epsilon\sim\mathcal{N}(0,1),t}\Big[\|\epsilon - \epsilon_\theta(z_t,t)\|_2^2\Big]\,.$
    - **Can receive Conditioning**
      - Text
      - Images
      - Semantic maps
      - ……



LDM

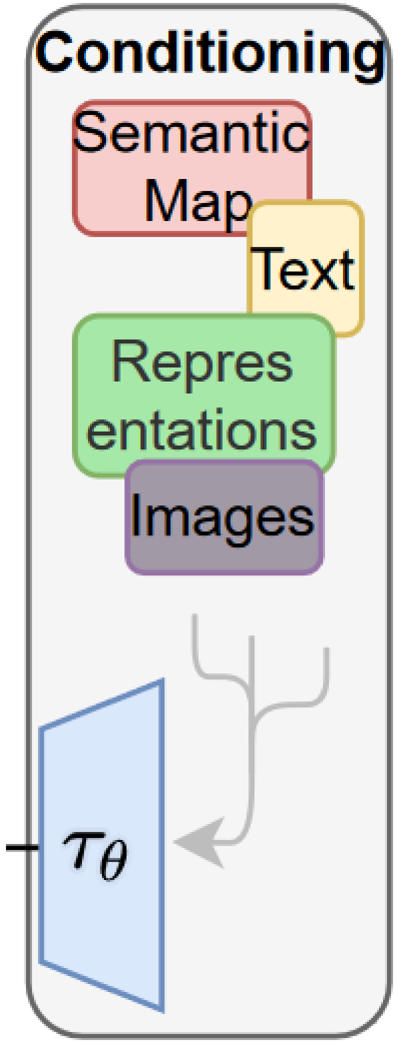
- Conditioning Mechanisms
  - Different(domain specific) τ_θ for different modalities
  - Using Attention to let conditioning effects on generation process:

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

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

  - Optimization target comes to:

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



Experiments

- On Perceptual Compression Tradeoffs
  - Ablation on different downsampling factor $f \in \{1, 2, 4, 8, 16, 32\}$
- Image Generation with Latent Diffusion
  - On four regular datasets
- Conditional Latent Diffusion
  - Text to Image
  - Layout to Image
  - Semantic Map to Image
- SR and Inpainting

Experiments

- On Perceptual Compression Tradeoffs

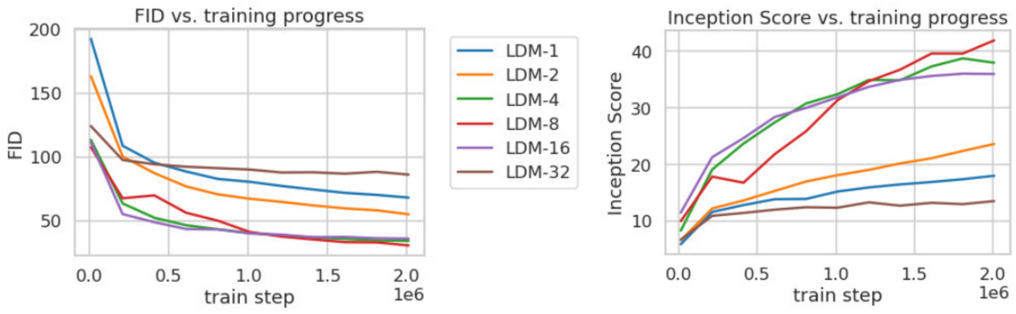


Figure 6. Analyzing the training of class-conditional *LDMs* with different downsampling factors f over 2M train steps on the ImageNet dataset. Pixel-based *LDM-1* requires substantially larger train times compared to models with larger downsampling factors (*LDM-*{*4-16*}). Too much perceptual compression as in *LDM-32* limits the overall sample quality. All models are trained on a single NVIDIA A100 with the same computational budget. Results obtained with 100 DDIM steps [84] and $\kappa = 0$.

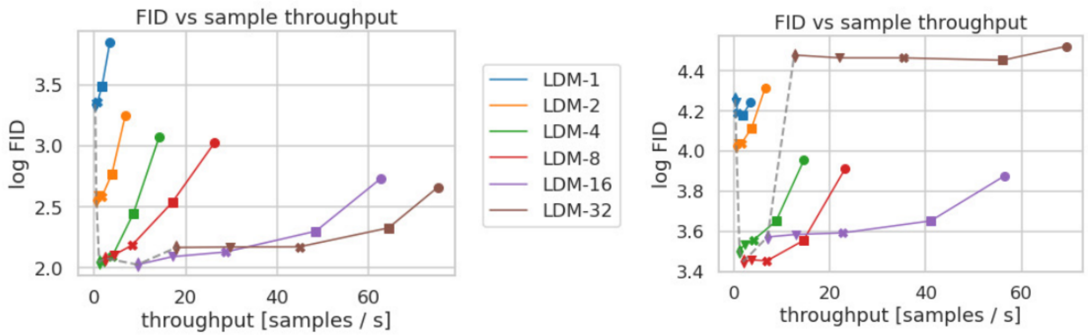


Figure 7. Comparing *LDMs* with varying compression on the CelebA-HQ (left) and ImageNet (right) datasets. Different markers indicate $\{10, 20, 50, 100, 200\}$ sampling steps using DDIM, from right to left along each line. The dashed line shows the FID scores for 200 steps, indicating the strong performance of *LDM-*{*4-8*}. FID scores assessed on 5000 samples. All models were trained for 500k (CelebA) / 2M (ImageNet) steps on an A100.

Experiments

- Image Generation with Latent Diffusion

CelebA-HQ 256 × 256				FFHQ 256 × 256			
Method	FID ↓	Prec. ↑	Recall ↑	**Method**	FID ↓	Prec. ↑	Recall ↑
DC-VAE [63]	15.8	-	-	ImageBART [21]	9.57	-	-
VQGAN+T. [23] (k=400)	10.2	-	-	U-Net GAN (+aug) [77]	10.9 (7.6)	-	-
PGGAN [39]	8.0	-	-	UDM [43]	5.54	-	-
LSGM [93]	7.22	-	-	StyleGAN [41]	4.16	0.71	0.46
UDM [43]	7.16	-	-	ProjectedGAN [76]	**3.08**	0.65	0.46
LDM-4 (ours, 500-s†)	**5.11**	0.72	0.49	*LDM-4* (ours, 200-s)	4.98	**0.73**	**0.50**

LSUN-Churches 256 × 256				LSUN-Bedrooms 256 × 256			
Method	FID ↓	Prec. ↑	Recall ↑	**Method**	FID ↓	Prec. ↑	Recall ↑
DDPM [30]	7.89	-	-	ImageBART [21]	5.51	-	-
ImageBART [21]	7.32	-	-	DDPM [30]	4.9	-	-
PGGAN [39]	6.42	-	-	UDM [43]	4.57	-	-
StyleGAN [41]	4.21	-	-	StyleGAN [41]	2.35	0.59	0.48
StyleGAN2 [42]	3.86	-	-	ADM [15]	1.90	**0.66**	**0.51**
ProjectedGAN [76]	**1.59**	0.61	0.44	ProjectedGAN [76]	**1.52**	0.61	0.34
*LDM-8** (ours, 200-s)	4.02	**0.64**	**0.52**	*LDM-4* (ours, 200-s)	2.95	**0.66**	0.48

Experiments

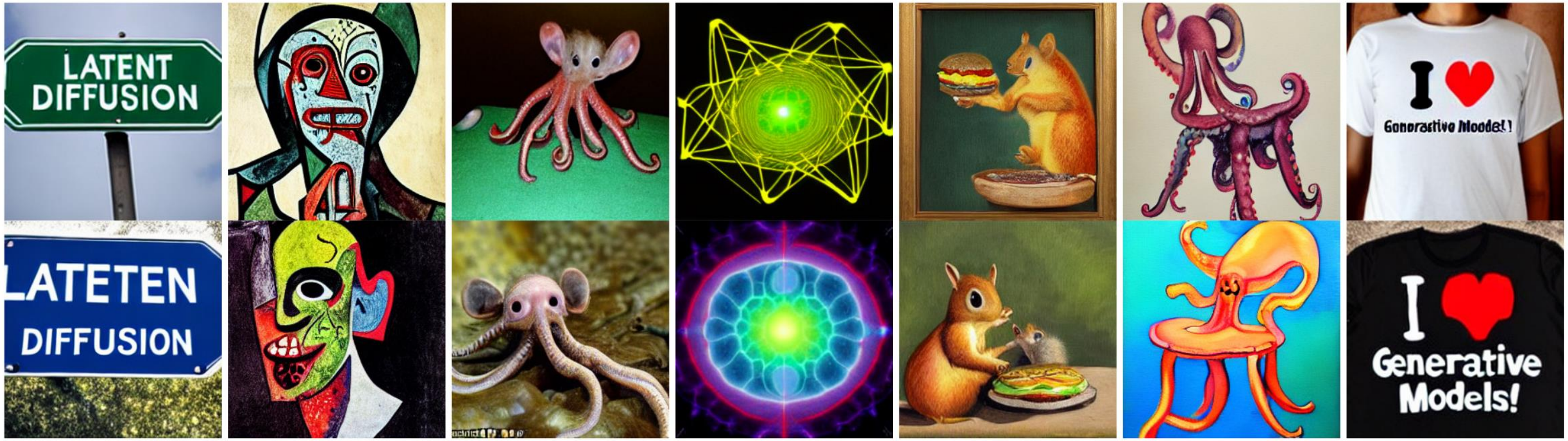
- Conditional Latent Diffusion
  - Text to Image
    - BERT Tokenizer， Train on LAION-400M

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$0.33	1.45B	250 DDIM steps
*LDM-KL-8-G**	12.63	**30.29**$\pm$0.42	1.45B	250 DDIM steps, c.f.g. [32] $s = 1.5$

Table 2. Evaluation of text-conditional image synthesis on the 256 × 256-sized MS-COCO [51] dataset: with 250 DDIM [84] steps our model is on par with the most recent diffusion [59] and autoregressive [26] methods despite using significantly less parameters. †/*:Numbers from [109]/ [26]

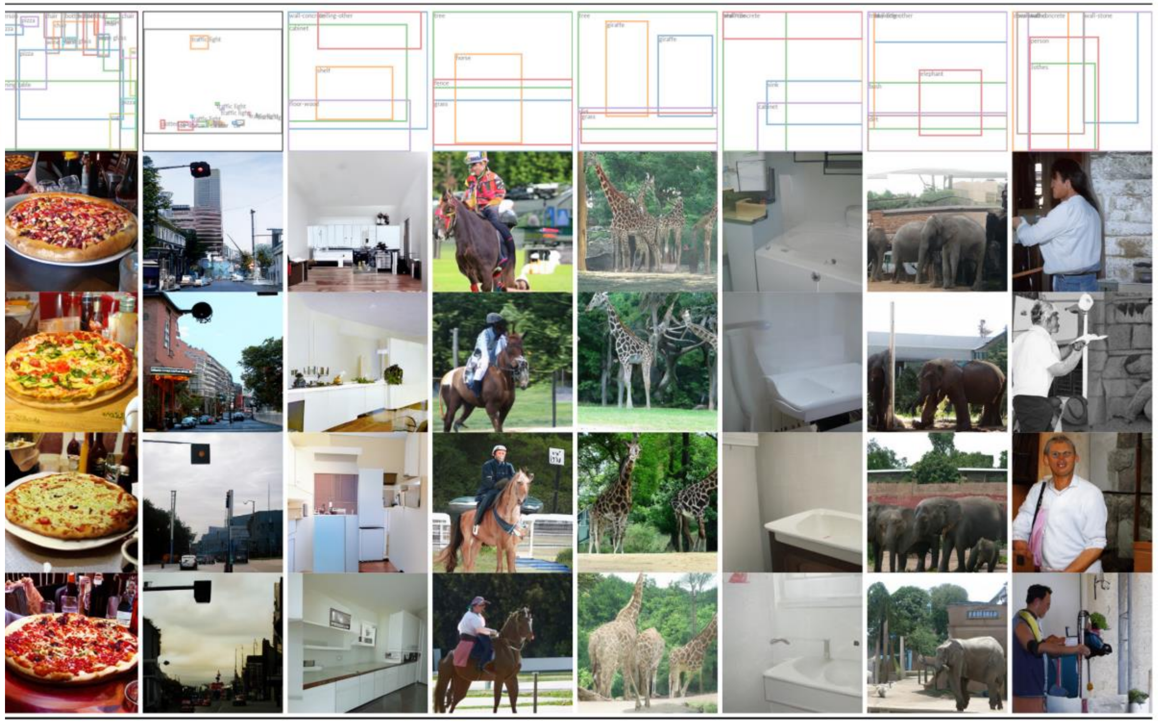
Text-to-Image Synthesis on LAION. 1.45B Model.

'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!" '*



Experiments

- Conditional Latent Diffusion
  - Layout to Image
  - Semantic map to Image



layout-to-image synthesis on the COCO dataset

Experiments

- Super Resolution & Inpainting

User Study	SR on ImageNet Pixel-DM ($f1$)	SR on ImageNet *LDM-4*	Inpainting on Places LAMA [88]	Inpainting on Places *LDM-4*
Task 1: Preference vs GT ↑	16.0%	**30.4%**	13.6%	**21.0%**
Task 2: Preference Score ↑	29.4%	**70.6%**	31.9%	**68.1%**

Table 4. Task 1: Subjects were shown ground truth and generated image and asked for preference. Task 2: Subjects had to decide between two generated images. More details in E.3.6

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**	<u>26.4</u>	<u>0.762</u>	625M	N/A
LDM-4 (ours, 100 steps)	<u>2.8</u>†/<u>4.8</u>‡	166.3	$24.4_{\pm 3.8}$	$0.69_{\pm 0.14}$	**169M**	4.62
emphLDM-4 (ours, big, 100 steps)	**2.4**†/**4.3**‡	<u>174.9</u>	$24.7_{\pm 4.1}$	$0.71_{\pm 0.15}$	552M	4.5
LDM-4 (ours, 50 steps, guiding)	4.4†/6.4‡	153.7	$25.8_{\pm 3.7}$	$0.74_{\pm 0.12}$	<u>184M</u>	0.38

Table 5. $\times 4$ upscaling results on ImageNet-Val. (256^2); †: FID features computed on validation split, ‡: FID features computed on train split; *: Assessed on a NVIDIA A100

Model (*reg.*-type)	train throughput samples/sec.	sampling throughput† @256	sampling throughput† @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

Table 6. Assessing inpainting efficiency. †: Deviations from Fig. 7 due to varying GPU settings/batch sizes *cf*. the supplement.

Method	40-50% masked FID ↓	40-50% masked LPIPS ↓	All samples FID ↓	All samples LPIPS ↓
LDM-4 (ours, big, w/ ft)	**9.39**	<u>0.246</u>$\pm$ 0.042	**1.50**	<u>0.137</u>$\pm$ 0.080
LDM-4 (ours, big, w/o ft)	12.89	0.257$\pm$ 0.047	2.40	<u>0.142</u>$\pm$ 0.085
LDM-4 (ours, w/ attn)	11.87	0.257$\pm$ 0.042	2.15	<u>0.144</u>$\pm$ 0.084
LDM-4 (ours, w/o attn)	12.60	0.259$\pm$ 0.041	2.37	<u>0.145</u>$\pm$ 0.084
LaMa [88]†	12.31	**0.243**$\pm$ 0.038	2.23	**0.134**$\pm$ 0.080
LaMa [88]	12.0	**0.24**	2.21	<u>0.14</u>
CoModGAN [107]	<u>10.4</u>	0.26	<u>1.82</u>	0.15
RegionWise [52]	21.3	0.27	4.75	0.15
DeepFill v2 [104]	22.1	0.28	5.20	0.16
EdgeConnect [58]	30.5	0.28	8.37	0.16

Table 7. Comparison of inpainting performance on 30k crops of size 512 × 512 from test images of Places [108]. The column *40-50%* reports metrics computed over hard examples where 40-50% of the image region have to be inpainted. †recomputed on our test set, since the original test set used in [88] was not available.

Thanks for watching !