

High-Resolution Image Synthesis with Latent Diffusion Models

Kenichiro Goto

Paper: Rombach et al., CVPR 2022
(CompVis, LMU Munich & Runway ML)
2025 Dec 5

Agenda

- Abstract
- モチベーション
- 提案手法
- 既存手法との比較
- Perceptual Image Compression (Stage 1)
- Latent Diffusion Models (Stage 2)
- Conditioning Mechanisms
- なぜCross Attentionで条件付けが可能になるのか?
- なぜピュアなtransformerではなくU-netなのか?
- アーキテクチャ
- 実験
- 多様なタスクにおける成果
- まとめ
- 参考文献

Abstract

本論文の提案は、Latent Diffusion Models (LDMs)

- 背景: 従来の拡散モデル (DDPM等) はピクセル空間で動作するため、学習・推論の計算コストが極めて高い。
- 提案: 学習済みのAutoencoderを用いて画像を低次元の潜在空間へ圧縮し、その空間上で拡散モデルを学習する。
- 成果
 - 画質を維持しつつ、計算コストを劇的に削減
 - 推論速度の高速化
 - Cross-Attention層の導入により、テキストやレイアウトなどの柔軟な条件付けが可能に。
 - Inpainting, Super-resolution, Text-to-ImageでSOTAまたはそれに匹敵する性能を達成。

モチベーション

前回触れたDDPMなどは、高解像度画像の生成において優れた性能を示すが、いくつか問題点がある。これらはピクセル空間で計算を行うことに起因する

1. 計算コスト: 高次元なRGB画像の各ピクセルに対してデノイズ処理を行うため、学習には大量のGPUを要し推論も低速
 - e.g. 150-1000 V in 100 days
 - モデルは各ステップで $256 \times 256 \times 3$ px分のノイズを予測しないといけない
2. 知覚的冗長性: 画像の細部（高周波成分）の学習に多くの計算リソースを消費しているが、これらは意味的な内容とは必ずしも直結しない

アプローチ

- Perceptual Compression
- Semantic Generation

にプロセスを分離する。

「画像の大半は知覚的詳細であり、意味的・概念的な部分は圧縮後も残っているはずである」[2] という発想からきている

提案手法

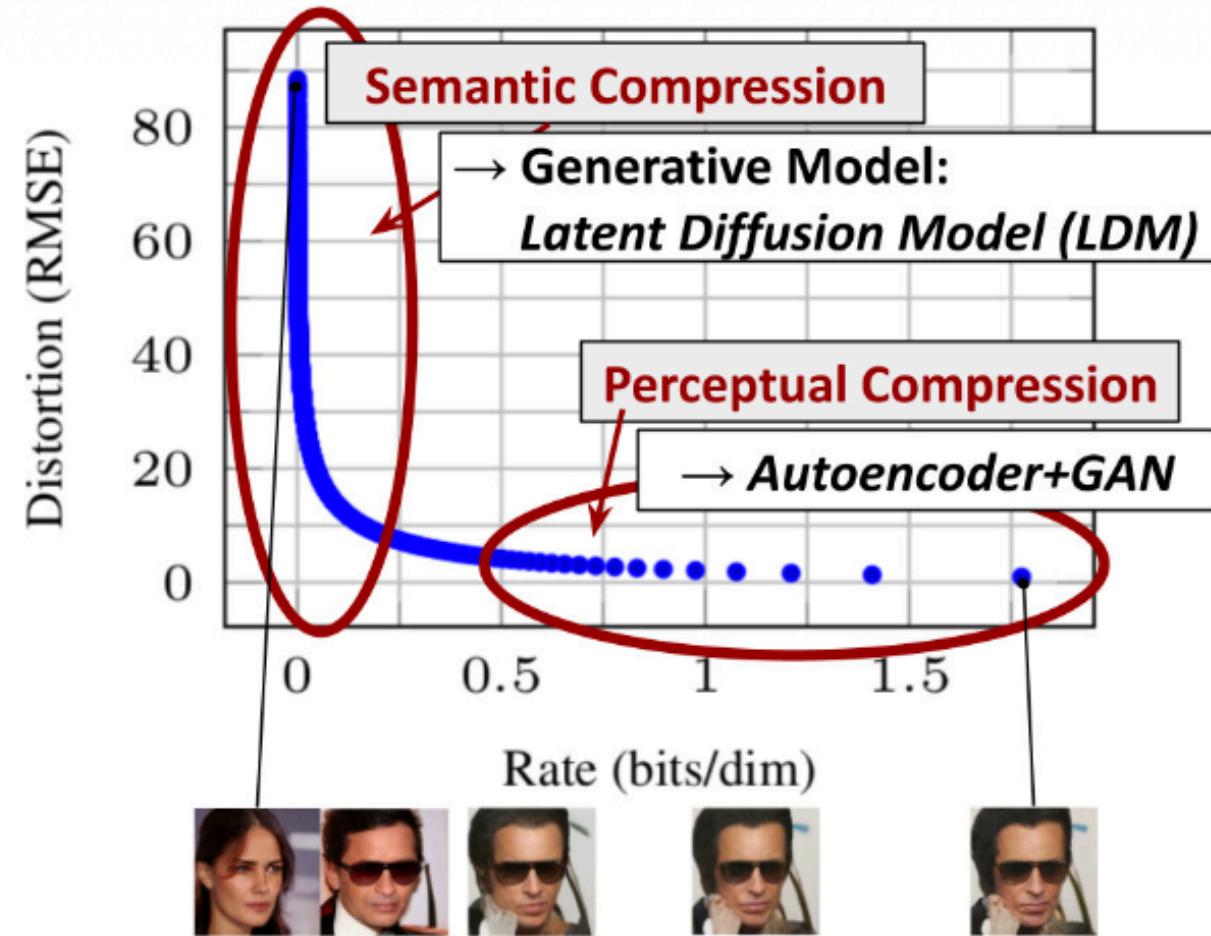
LDMは大きく2つのステージに分かれる。

1. Perceptual Compression

- ピクセル空間 x を知覚的に等価な低次元の潜在空間 z に圧縮する。
- ピクセル空間での冗長性・高周波成分を取り除き、計算効率を上げる
- 形状・色・意味的な構造などhigh-levelな情報は維持

2. Latent Diffusion

- 圧縮された潜在空間 z 上で拡散過程を行う。
- 空間的な次元が減るため、高解像度画像の生成も効率的に行える



既存手法との比較

Method	Latent Type	Pros	Cons
VQGAN + Transformer	Discrete latent	高品質; transformer由来の柔軟性	でかい(>1B)ので遅い
Pixel-based DDPM	Pixel	最も忠実	高コスト
LDM	Continuous latent	高速・高画質・汎用的	Autoencoderに依存 [2]

[2] Weng, Lilian. (Jul 2021). What are diffusion models? Lil'Log.
<https://lilianweng.github.io/posts/2021-07-11-diffusion-models/>.

Perceptual Image Compression (Stage 1)

学習済みAutoencoder $(\mathcal{E}, \mathcal{D})$ を利用する

- Encoder \mathcal{E} : 画像 $x \in \mathbb{R}^{H \times W \times 3}$ を潜在表現 $z = \mathcal{E}(x) \in \mathbb{R}^{h \times w \times c}$ にエンコード
 - ダウンサンプリング係数 $f = 2^m$ where $m \in \mathbb{N}$
- Decoder \mathcal{D} : 潜在表現から画像を再構成 $\tilde{x} = \mathcal{D}(z)$
- 正則化: 潜在空間の分散を抑えるため、(i)KL正則化または(ii)VQ正則化を使用

Encoderは一度学習すれば、様々なタスクのDMsの学習に**再利用可能**

Latent Diffusion Models (Stage 2)

潜在空間 z に対する拡散モデル

- Forward Process: z にガウシアンノイズを徐々に追加し、純粋なノイズにする (DDPMと同様)。
- Reverse Process: ノイズから z を復元するよう、Denoising U-Net ϵ_θ を学習。

目的関数

ピクセル x ではなく、潜在変数 $z (= \mathcal{E}(x))$ に対して最適化を行う。

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

- t : タイムステップ
- z_t : ノイズが加わった時点 t の潜在表現
- ϵ_θ : ノイズ予測ネットワーク (Time-conditional U-Net)

DDPMとの比較

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

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

Conditioning Mechanisms

LDMの強力な特徴は、テキスト、画像、レイアウトなど多様な入力 y で生成を制御できる点。

Cross-Attention の導入

U-Netの中間層にCross-Attention機構を組み込み条件 y を注入する

1. ドメイン固有のエンコーダ τ_θ (例: BERT, CLIP) で y を中間表現に変換。
2. U-Netの特徴マップ $\varphi_i(z_t)$ と Attention をとる。

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

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

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

- $\tau_\theta(y) \in \mathbb{R}^{M \times d_\tau}$
- $\varphi_i(z_t) \in \mathbb{R}^{N \times d_\epsilon^{(i)}} \rightarrow \mathbb{R}^{N \times d_i}$
- $W_Q^{(i)} \in \mathbb{R}^{d \times d_\epsilon^{(i)}} \rightarrow \mathbb{R}^{d_i \times d}, \quad W_K^{(i)} \in \mathbb{R}^{d \times d_\tau} \rightarrow \mathbb{R}^{d_\tau \times d}, \quad W_V^{(i)} \in \mathbb{R}^{d \times d_\tau} \rightarrow \mathbb{R}^{d_\tau \times d}$

- i : U-netのブロックのインデックス。つまり各ブロックに対して Q, K, V の projection が存在する
- M : テキスト埋め込みを生成するエンコーダ (Transformer) の出力 token 長 (CLIPなら $M = 77$)
- d_τ : 1 token embeddingあたりの次元数 (CLIP では 768 固定)
- N : 画像の空間位置の総数 ($= h \times w$)。attentionに入る前にこのような空間方向の圧縮をするのは典型的
- d : attention headの次元。ハイパーパラメータ

以上を組み込んで、最終的な loss は

$$L_{DDPM} := \mathbb{E}_{\mathcal{E}(x), \epsilon \sim \mathcal{N}(0,1), t} [||\epsilon - \epsilon_\theta(x_t, t, \tau_\theta(y))||_2^2] \quad (3)$$

なぜCross Attentionで条件付けが可能になるのか?

- 画像側特徴 $\varphi(\mathbf{z}) \rightarrow \text{Query}$
- テキスト側特徴 $\tau(\mathbf{y}) \rightarrow \text{Key / Value}$

Attention(Q, K, V) で画像のどの空間位置がテキストのどの単語と対応すべきかという依存関係を学習

- Key-Value はテキストの意味を持っており、Query は 画像の生成途中の特徴マップを指す
- $\text{softmax}(\mathbf{Q}\mathbf{K}^\top)$ により、位置ごとに関連単語が選ばれる
これによりテキストと画像の対応 (つまり条件付け) が自然に形成される

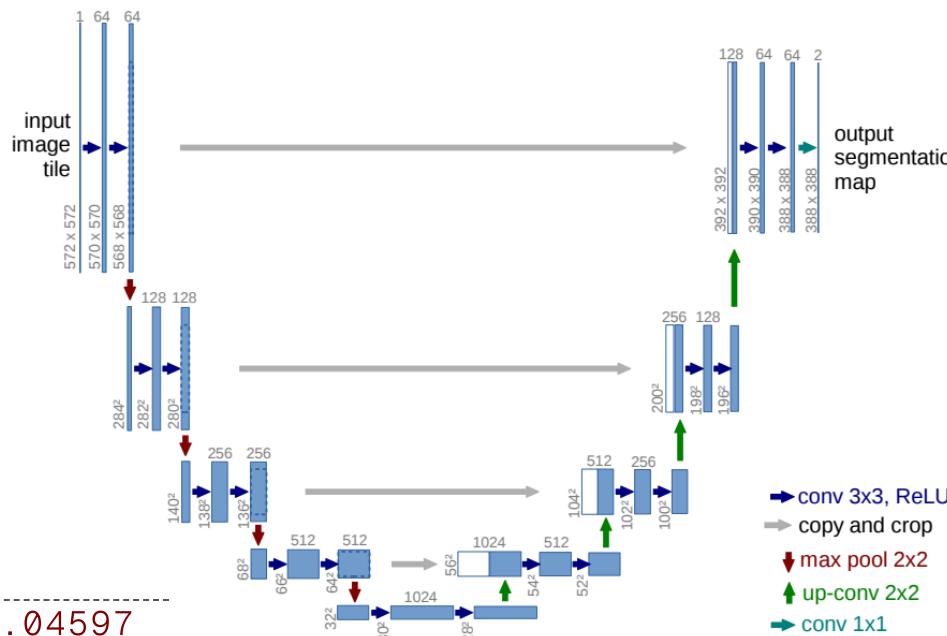
なぜピュアなtransformerではなくU-netなのか?

U-Net [4] は局所性を強く持つアーキテクチャであり、そのinductive biasを利用するため

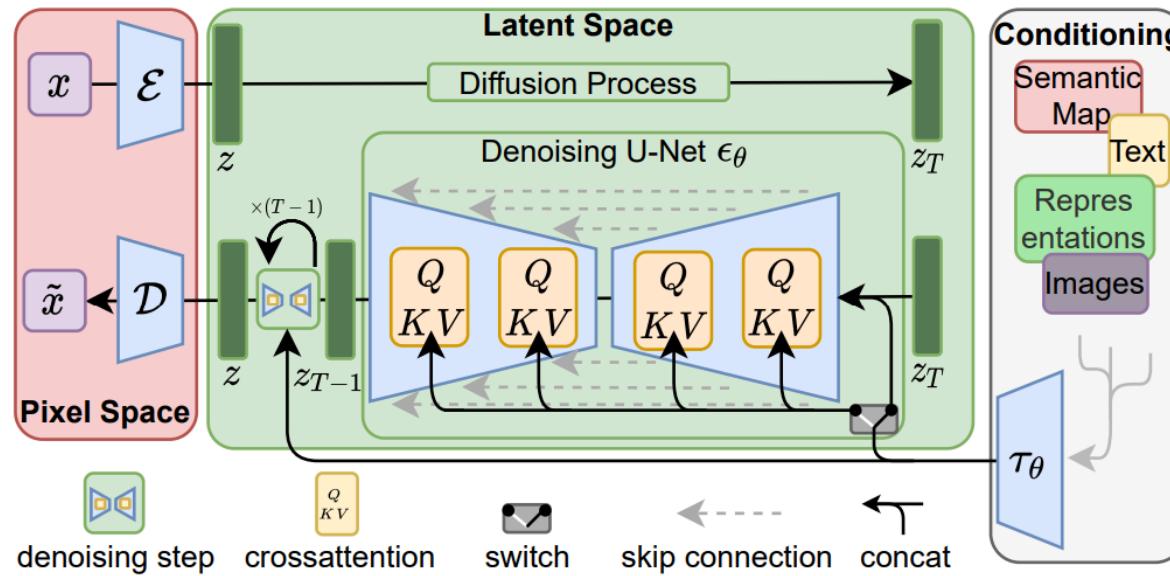
- Encoderで粗い情報を取得しDecoderで細かい情報を復元、skip connectionで両者を統合
- 画像の構造（エッジ・パート・形状）を自然に扱える

ピュアなTransformerはfully-connected attentionのため、画像の局所構造を学習するには大量データ・計算が必要

LDMはlatent spaceで計算し比較的低解像度なのでCNNのinductive biasが効率的に働く



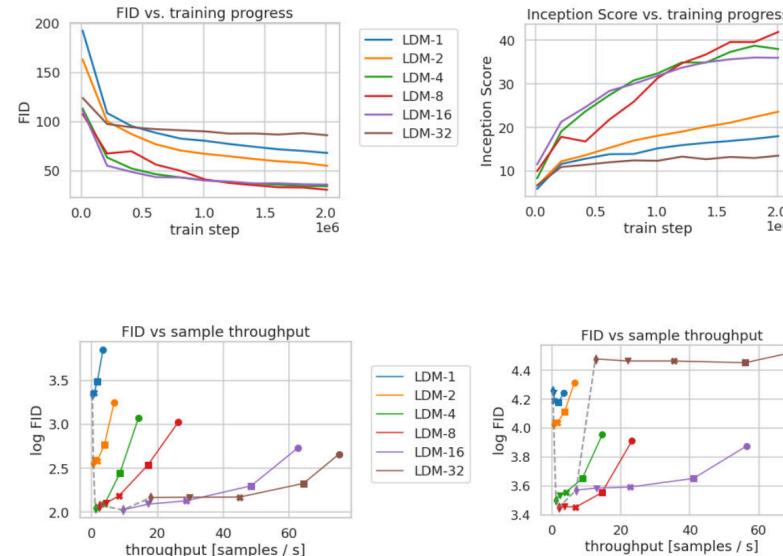
アーキテクチャ



1. Pixel Space: 入力画像 $x \xrightarrow{\mathcal{E}}$ Latent Space Representation: z
2. Diffusion Process: $z \xrightarrow{\text{Noise}} z_T$
3. Denoising U-Net: $z_T \xrightarrow{\text{Denoise}} z$
 - ここに Conditioning (Text, Semantic Map等) が Cross-Attention で入る。
4. Pixel Space: $z \xrightarrow{\mathcal{D}}$ 出力画像 \tilde{x}

実験

ダウンサンプリング係数 f が生成品質と効率にどう影響するか？



- f が小さい (e.g. 1, 2): 圧縮が足りずピクセル空間に近いので計算コスト削減効果が薄い。学習も遅い
- f が大きすぎる (e.g. 32): 情報が失われすぎてしまい、画質が停滞する
- 最適なバランス: $f \in \{4, 8, 16\}$ が最も良いトレードオフを示した (より左下)

LDM-4 や LDM-8 が、従来のPixel-based DM (LDM-1) よりも低いFIDと高いスループットを達成

多様なタスクにおける成果

1. Text-to-Image Synthesis (Fig. 1)

- LAION-400Mデータセットで学習
- 1.45Bパラメータのモデルで、ARモデルやGANと同等以上の性能
- ユーザー入力プロンプトに対して忠実で高解像度な画像を生成可能

2. Inpainting (Fig. 2)

- 欠損部分の補完。高解像度でも整合性の取れた補完が可能
- U-netでの畳み込み的なサンプリングにより、 512^2 px以上の解像度にも対応

3. Super-Resolution (Fig. 3)

- 低解像度画像を入力条件として連結して学習する
- SR3 (Pixel-based DM) に匹敵するFIDを達成しつつ推論は高速

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

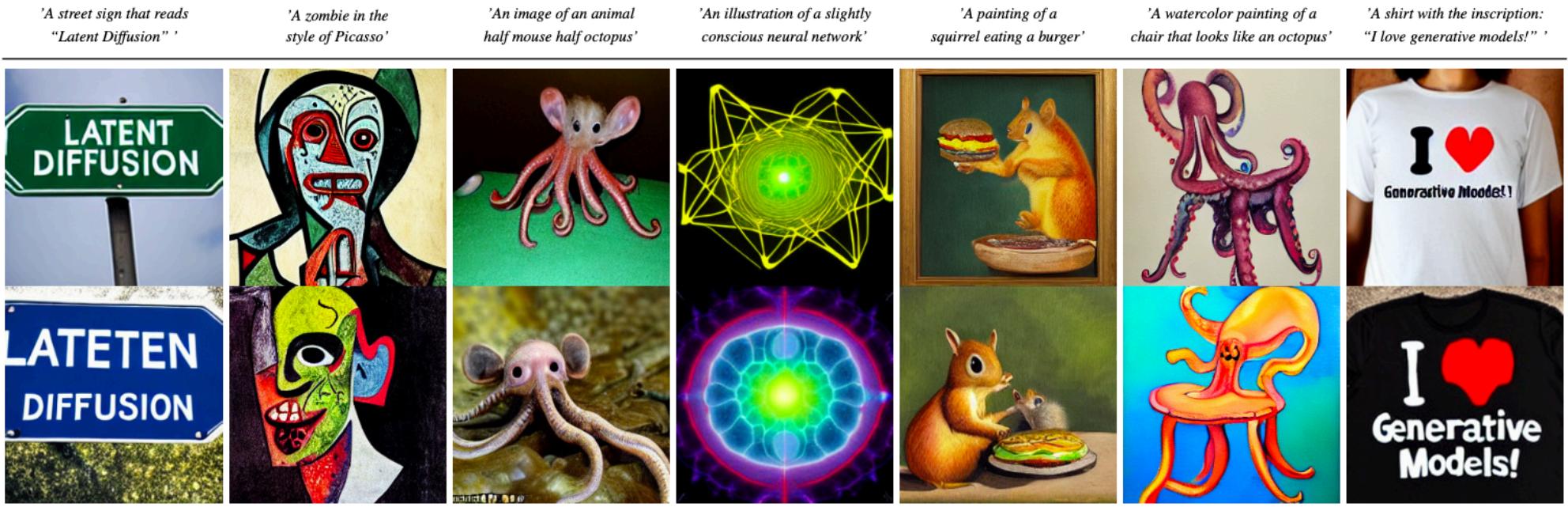


Fig 1.

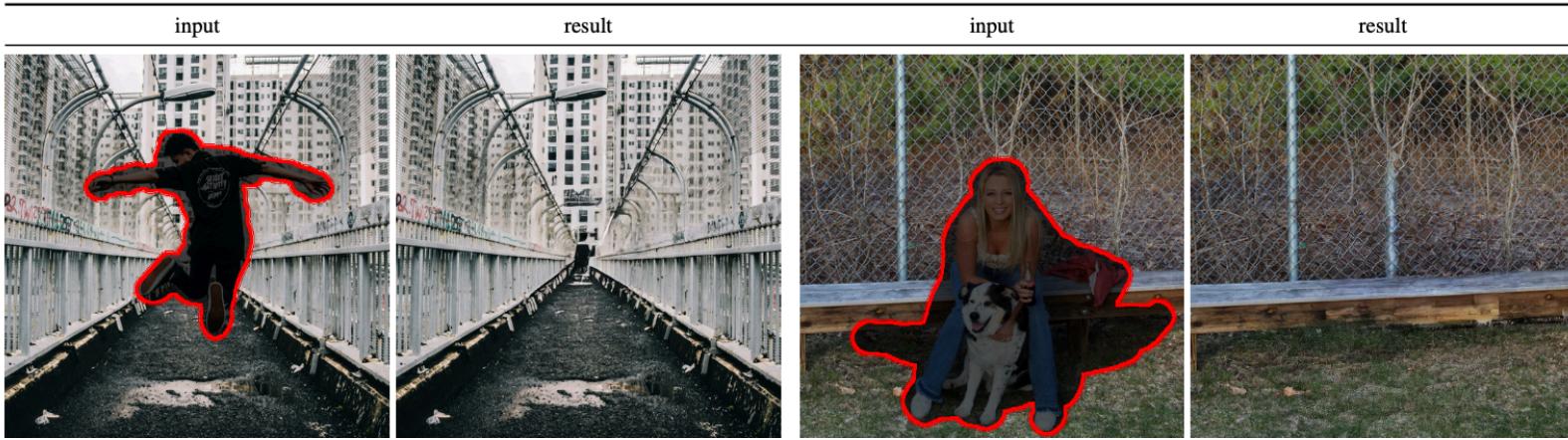


Fig 2. 512^2 pxでのInpainting

Method	FID ↓	IS ↑	PSNR ↑	SSIM ↑	Nparams	[$\frac{\text{samples}}{\text{s}}$] (*)
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
<i>LDM-4</i> (ours, 100 steps)	<u>2.8[†]/4.8[‡]</u>	166.3	24.4 ± 3.8	0.69 ± 0.14	169M	4.62
emphLDM-4 (ours, big, 100 steps)	2.4[†]/4.3[‡]	<u>174.9</u>	24.7 ± 4.1	0.71 ± 0.15	552M	4.5
<i>LDM-4</i> (ours, 50 steps, guiding)	<u>4.4[†]/6.4[‡]</u>	<u>153.7</u>	25.8 ± 3.7	0.74 ± 0.12	<u>184M</u>	0.38

Fig. 3. パラメータ数を大きく抑えつつ、FIDでSR3を上回る

まとめ

- Latent Diffusion Models (LDMs) を提案
 - 画像生成プロセスをPerceptual CompressionとLatent Diffusionに分離
 - ピクセル空間ではなく潜在空間で拡散モデルを学習することで、計算コストを大幅に削減しつつ高解像度・高画質を達成
- Cross-Attention による柔軟な条件付けメカニズムを導入し、Text-to-Imageなど多様なタスクでSOTA級の性能

参考文献

- [1] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 10684–10695. <https://arxiv.org/abs/2112.10752>
- [2] Weng, Lilian. (Jul 2021). What are diffusion models? Lil'Log. <https://lilianweng.github.io/posts/2021-07-11-diffusion-models/>.
- [3] Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. Proceedings of the European Conference on Computer Vision (ECCV), 694–711. <https://arxiv.org/abs/1505.04597>