LDM

A Preprint

1 Contributions

The main paper's contributions:

- LDM's preserve more detail that transformers and scale efficiently
- Performance across different tasks(unconditional synthesis, class-conditional ImageNet, inpainting, super-resolution, and text-to-image generation while lowering compute costs)
- Unlike joint encoder-prior structures in LDM autoencoder and Diffusion model both are trained separately from each other
- Enables convolutional sampling to generate coherent images up to 1024^2 px
- Cross-attention conditioning which overall improves conditioning

2 Main

2.1 Perceptual image compression

Train a convolutional autoencoder E, D to learn a low-dimensional latent space that preserves perceptually relevant structure. With perceptual loss plus a patch-based adversarial loss ensuring reconstructions lie on the image manifold.

Latent $z = E(x) \in \mathbb{R}^{h \times w \times c}$ downsamples by $f = \frac{H}{h}$

Two regularization were introduced:

KL-reg: a mild KL-reg penalty toward $\mathcal{N}(0; I)$ VQ-reg: uses vector quantization layer absorbed into the decoder

2.2 Latent Diffusion

A diffusion model is trained on the latent space via the reweighted denoising objective

$$\mathbb{E}_{z,\epsilon \sim N(0;I),t} ||\epsilon - \epsilon_\theta(z_t, t)||_2^2$$

Backbone ϵ_θ is a time-conditional U-net operating on latents.

2.3 Conditioning

Cross-attention layer is inserted in U-net structure, mapping conditional inputs y hrough a domain encoder ρ_θ into key/value vectors.

3 More details

Autoencoder:

Encoder: stack of 2D convolution residual blocks with GroupNorm and SiLu

Decoder: symmetric upsampling path (nearest-neighbor upsampling + conv) mirroring the encoder.

Loss may be described as combination of Reconstruction loss, Perceptual loss(VGG network features used) and Patch-based adversarial loss(with a PatchGAN discriminator that pushes reconstructions onto the natural-image manifold)

$$\mathcal{L}_{\text{rec}} = \left\| x - D\big(E(x)\big) \right\|_2^2, \tag{1}$$

$$\mathcal{L}_{\text{perc}} = \left\| \phi(x) - \phi\big(D(E(x))\big) \right\|_2^2, \tag{2}$$

$$\mathcal{L}_{\text{adv}} = \log D_\psi(x) \; + \; \log\big(1 - D_\psi\big(D(E(x))\big)\big), \tag{3}$$

UNet Architecture (Overall U-shape)

- Downsampling path: Four resolutions, each consisting of
  - Two residual blocks (GroupNorm → SiLU → Conv)
  - Self-attention at the coarsest three resolutions (32, 16, 8px)
  - Strided-convolution downsampling
- Bottleneck:
  - One residual block
  - Self-attention
- Upsampling path: Mirror of the downsampling path, using nearest-neighbor upsampling + Conv, with skip-connections from corresponding downsampling layers.

Time Conditioning (FiLM)

Each residual block is modulated by a learned embedding of timestep t:

1. Compute sinusoidal embedding
$$\text{SinEmb}(t) \;\in\; \mathbb{R}^d.$$
2. Pass through an MLP to obtain scale–shift vectors
$$(\gamma_t,\, \beta_t) \;\in\; \mathbb{R}^d \times \mathbb{R}^d.$$
3. In each block, after GroupNorm, apply FiLM:
$$h \;\mapsto\; \gamma_t \odot h \;+\; \beta_t.$$

Hyperparameters

- Diffusion steps: $T = 1000$ (linear β_t schedule)
- Model size: $\approx 274\,\text{M}$ parameters
- Base channels: 224
- Channel multipliers: $\{1, 2, 3, 4\}$
- Attention resolutions: 32, 16, 8 px
- Batch size: 48 (CelebA–HQ), 42 (FFHQ), ...
- Training iterations: 410k–1.9M
- Learning rate: 5×10^{-5}–9.6×10^{-5}

Algorithms

Algorithm 1 Training Latent Diffusion UNet

Pretrained autoencoder (E, D), UNet ϵ_θ, noise schedule $\{\beta_t\}_{t=1}^{T}$ each training step Sample minibatch $\{x^{(i)}\}_{i=1}^{N}$ $z_0^{(i)} \leftarrow E\big(x^{(i)}\big)$ for all i Sample $t \sim \text{Uniform}\{1, \dots, T\}$ Sample noise $\epsilon^{(i)} \sim \mathcal{N}(0, I)$ $z_t^{(i)} \leftarrow \sqrt{\alpha_t}\, z_0^{(i)} + \sqrt{1-\alpha_t}\, \epsilon^{(i)}$ $\hat{\epsilon}^{(i)} \leftarrow \epsilon_\theta\big(z_t^{(i)}, t\big)$ Compute $\mathcal{L} = \dfrac{1}{N} \sum_i \big\| \epsilon^{(i)} - \hat{\epsilon}^{(i)} \big\|_2^2$ Update $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$

Algorithm 2 Sampling from Latent Diffusion Model

Trained UNet ϵ_θ, decoder D, schedules $\{\alpha_t, \beta_t\}$ $z_T \sim \mathcal{N}(0, I)$ $t = T, T-1, \dots, 1$ $\hat{\epsilon} \leftarrow \epsilon_\theta(z_t, t)$ $\mu_\theta \leftarrow \dfrac{1}{\sqrt{1-\beta_t}} \Big(z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\, \hat{\epsilon} \Big)$ $z_{t-1} \sim \mathcal{N}\big(\mu_\theta,\, \beta_t I\big)$ return $\hat{x} = D(z_0)$