扩散模型在图像重建与生成任务中的应用

李孟霖 2311067

November 10, 2025

DDPM

Denoising Diffusion Probabilistic Models

NeruIPS 2020

基本思路

Denoising Diffusion Probabilistic Models(DDPM) 通过逐步添加噪声破坏图像，再用反向过程逐步去噪重建图像。

模型学习从纯噪声去噪得到图片的过程，从而实现由任意纯高斯噪声生成图片的功能。

该模型结合了变分下界优化和 Langevin 动力学的去噪得分匹配，实现高质量图像生成。

DDPM

数学原理

前向过程 (Forward Process):
给定原始图像 $x_0 \sim q(x_0)$，前向过程逐步添加高斯噪声，构成一个固定的马尔可夫链：

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}), \quad q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

可直接从 x_0 采样出任意时刻 x_t：

$$q(x_t|x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$$

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

反向过程 (Reverse Process):
从纯噪声 $x_T \sim \mathcal{N}(0, I)$ 开始，反向过程学习一个去噪马尔可夫链：

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t), \quad p_\theta(x_{t-1}|x_t) = \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

DDPM 损失函数构造与简化

目标：利用变分下界（ELBO），设计可优化的训练损失函数，以学习从噪声 x_T 逐步生成图像 x_0 的反向过程。

1. **变分下界（ELBO）分解：**

$$\log p_\theta(x_0) \geq \text{ELBO} = -[L_T + \sum_{t=2}^{T} L_{t-1} + L_0]$$

- $-L_T$ 是固定常数，可忽略
- $-L_{t-1}$ 是主要的训练目标项（KL 散度）
- $-L_0$ 是重建项（有时也忽略）

DDPM 损失函数构造与简化

2. **反向过程建模为高斯分布：**

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

- 其中 σ_t^2 为固定超参数，不学习
- μ_θ 为模型（如 U-Net）输出

DDPM 损失函数构造与简化

3. **损失函数重写为预测后验均值：**

$$L_{t-1} = \mathbb{E}\left[\frac{1}{2\sigma_t^2}\|\mu_\theta(x_t,t) - \tilde{\mu}_t(x_t,x_0)\|^2\right] + C$$

4. **使用重参数化将问题转化为噪声预测：**

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0,I)$$

最终损失函数简化为：

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{x_0,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(x_t,t)\|^2\right]$$

结论：模型训练目标是拟合噪声 ϵ，本质上是一个时间条件的**去噪任务**。

algorithm

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

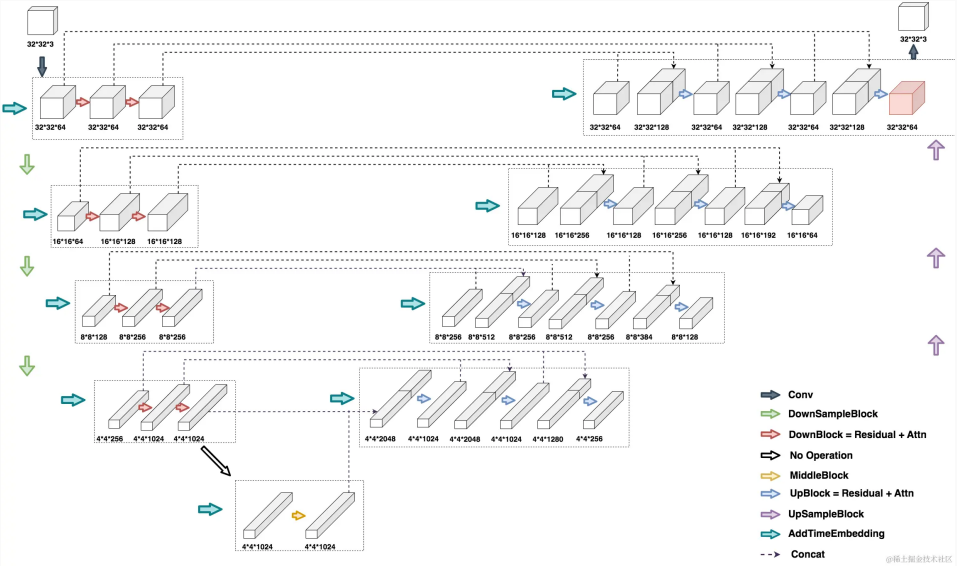
Algorithm 2 Sampling

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

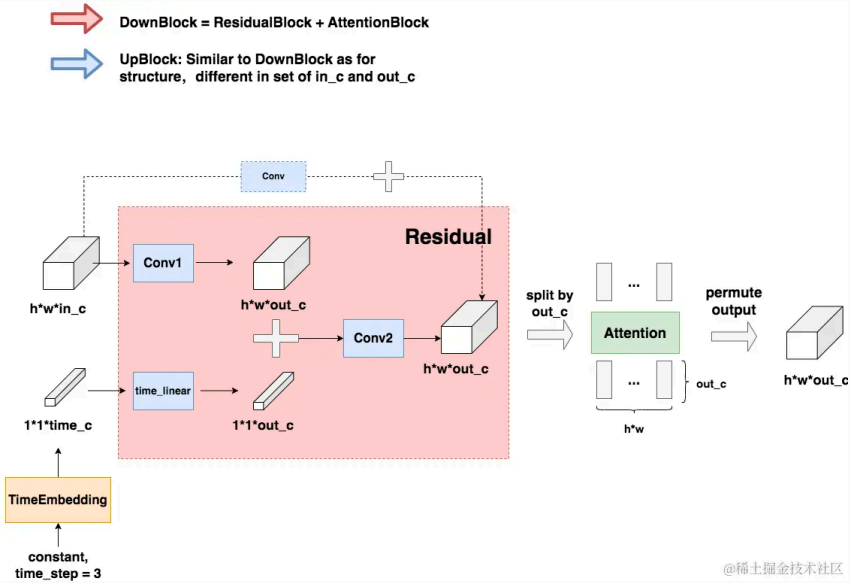
DDPM

模型结构

DDPM 模型结构（U-Net）



时间步条件的注入方式



DDPM 示例

Figure 3.DDPM 推理示例

VAE 压缩模块

VAE 在图像压缩中的核心机制

- **结构概览：**
  - 使用编码器 $E_\phi(x)$ 将高维图像 $x \in \mathbb{R}^{H\times W\times 3}$ 映射为低维潜向量 z
  - 潜空间分布为 $q_\phi(z|x) = \mathcal{N}(\mu(x), \sigma^2(x))$
  - 解码器 $D_\theta(z)$ 将 z 重建为图像 $\hat{x} = D(z)$
- **压缩意义：**
  - z 是图像的紧凑编码，维度远小于原始图像
  - 潜空间引入正则化（如 KL 散度），避免过度拟合，提升压缩泛化能力
  - 使用连续高斯向量（而非固定码本），更适合感知一致性场景
- **与传统 Autoencoder 的区别：**
  - VAE 学习的是 **分布**而非固定编码，压缩后可进行随机采样
  - 提供更好的鲁棒性与潜变量结构控制
- **压缩 vs 生成：**
  - 虽然 VAE 为生成设计，但编码器结构天然适合压缩任务
  - 实际应用中，如图像重建、感知压缩、表示学习常借助其潜空间能力

Latent Diffusion Model

Latent Diffusion Model (LDM): **功能概览**

- **计算高效**：
  - 在**潜空间**（latent space）中进行扩散建模，避免在高维像素空间中处理。
  - 显著**降低训练与推理成本**，支持低资源环境运行。
- **灵活的条件生成（**Conditional Generation**）**：
  - 引入 Cross-Attention **模块**，实现多种输入条件的控制：
    - 文本（Text-to-Image）
    - 边界框（Layout-guided generation）
    - 掩膜图像（Inpainting）
    - 类别标签（Class-conditional generation）
- **多任务适应性强**：
  - 同一模型适用于：
    - 无条件图像生成（Unconditional Generation）
    - 图像修复（Image Inpainting）
    - 图像增强与重建（e.g., Super-resolution）

Latent Diffusion Model

LDM Autoencoder

LDM 中的 Autoencoder：压缩图像感知结构

目的：将图像 $x \in \mathbb{R}^{H\times W\times 3}$ 压缩为结构化潜空间 $z \in \mathbb{R}^{h\times w\times c}$，便于高效训练扩散模型。

结构组成：

- Encoder $E(x) \to z$：多层卷积和残差模块（细节参见补充材料或代码）
- Decoder $D(z) \to \tilde{x}$：恢复图像
- 下采样比例 $f = H/h = W/w$, 例如 $f = 4, 8, 16$

训练目标：

- 使用 **感知损失（perceptual loss）**：保持图像语义细节
- 使用 PatchGAN **对抗损失**：增强局部真实性，避免模糊
- 最小化重建误差 $\mathcal{L}_{\text{rec}}(x, \tilde{x})$

Autoencoder 的正则化与训练策略

潜空间正则化目的：避免高方差、非结构化的 latent 表示，增强生成稳定性。

两种正则化方式：

- KL **正则（KL-reg.）：**
  - 将 $q_E(z|x) = \mathcal{N}(\mu, \sigma^2)$ 推近标准高斯 $\mathcal{N}(0, I)$
  - 类似 VAE，权重非常小（$\sim 10^{-6}$），避免损害重建质量
- VQ **正则（VQ-reg.）：**
  - 使用矢量量化 codebook，构造离散 latent 空间（类似 VQGAN）
  - 量化层嵌入解码器首层，编码器输出为连续 z

完整训练目标函数：

$$\mathcal{L}_{\text{AE}} = \min_{E,D} \max_{\psi} \left[\mathcal{L}_{\text{rec}}(x, D(E(x))) - \mathcal{L}_{\text{adv}} + \log D_\psi(x) + \mathcal{L}_{\text{reg}} \right]$$

判别器： Patch-based GAN 判别器 D_ψ（仅在训练时使用）

Latent Diffusion Model

Conditioning Mechanisms

LDM 的条件生成机制：灵活控制图像合成

目标：扩散模型学习条件分布 $p(z \mid y)$，在潜空间中实现**多模态控制**（如文本、图像、语义图等）。

核心思想：

- 将条件信息 y 融入扩散模型 $\epsilon_\theta(z_t, t, y)$ 中。
- 引入专门的 **条件编码器** $\tau_\theta(y)$，将 y 投影为中间表示。
- 使用 Cross-Attention **模块**，将 $\tau_\theta(y)$ 融入 UNet 的中间层。

适用任务：

- 文本到图像（text-to-image）
- 图像修复（inpainting）
- 图像翻译（image-to-image）
- 语义图控制合成

Cross-Attention 条件注入机制与训练目标

Cross-Attention 注入方式：

- 条件编码器输出 $\tau_\theta(y) \in \mathbb{R}^{M \times d_\tau}$。
- 在 UNet 中层使用 Attention：

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

- 投影定义：

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

训练目标函数（条件扩散损失）：

$$\mathcal{L}_{\text{cond}} = \mathbb{E}_{x,y,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2\right]$$

Latent Diffusion Model

LDM **模型结构**

LDM 模型整体架构图

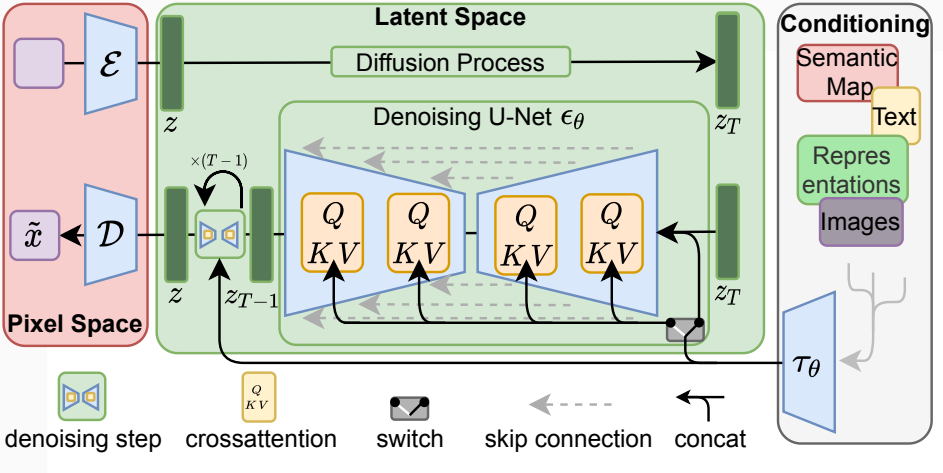


Figure 4.LDM 模型整体架构图

Stable SR: 基于 stable diffusion 的超分辨模型

Stable SR 模型结构

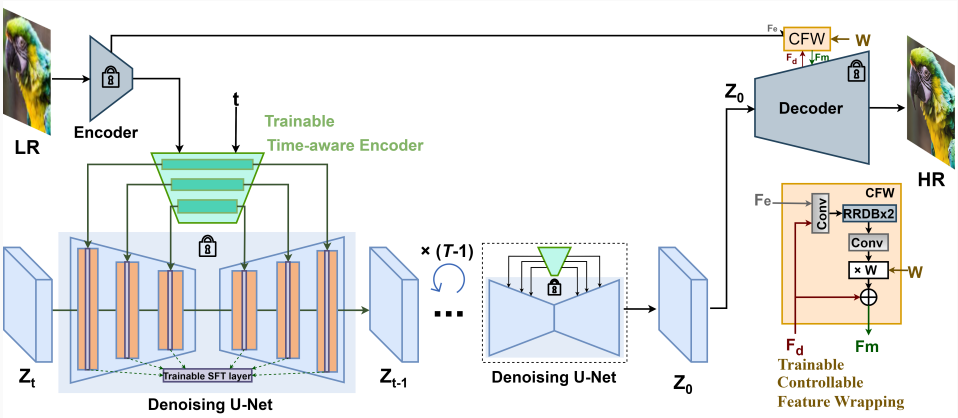


Figure 5.Stable SR 模型结构图

SFT 调制机制公式详解

调制公式：

$$\hat{F}^n_{\text{dif}} = (1 + \alpha^n) \odot F^n_{\text{dif}} + \beta^n$$

含义解析：

- F^n_{dif}：原始 UNet 在第 n 层生成的特征图。
- F^n：由 LR 图像提取的结构特征（通过 Time-Aware Encoder 获得）。
- M^n_θ：一个小型卷积网络，用于从 F^n 生成调制参数：

$$(\alpha^n, \beta^n) = M^n_\theta(F^n)$$

- α^n：** 缩放系数 **，控制每个通道的增强或抑制。
- β^n：** 偏置项 **，引入可控偏移以增强表达能力。

调制作用：

- 逐像素、逐通道地对原始特征图进行 **仿射变换**。
- 引入 LR 图像的多尺度语义信息，精细控制特征图生成过程。

Time-Aware Encoder：EncoderUNetModelWT（概念与结构）

结构

半个 UNet 编码器：按 时间步 t 条件化的多尺度特征抽取器；无解码头。返回各分辨率的特征并统一到 `out_channels`，供后续重建/融合模块使用。

输入 / 输出

- 输入：$\mathbf{x} \in \mathbb{R}^{N \times C \times H \times W}$，时间步 t（或 `timesteps`）
- 输出：`dict`，键为分辨率（如 "64"、"32"…），值为 $\mathbb{R}^{N \times \texttt{out\_channels} \times h \times w}$ 的特征张量

重要组件

- Time Embed：MLP(`model_channels` $\to 4\times$) 注入所有 ResBlock/Attention
- **编码路径**：按 `channel_mult` 堆叠 `num_res_blocks`；命中 `attention_resolutions` 时插自注意力
- Bottleneck：ResBlock $\to$ Attn $\to$ ResBlock
- **统一投影**：对每个尺度的特征用时序条件 ResBlock 映射到 `out_channels`

EncoderUNetModelWT

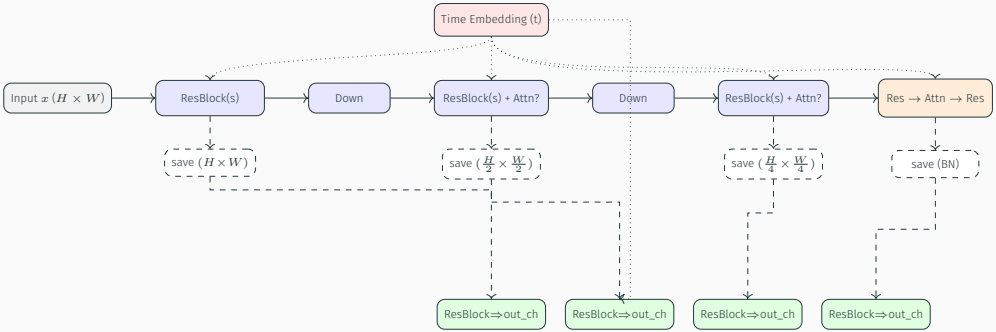
整体思路

编码路径逐层下采样并在指定分辨率插入注意力；每当分辨率变化，就“截取”下采样前的特征；瓶颈处理后把所有尺度通过 `fea_tran` 统一到 `out_channels`，以分辨率字符串为键打包返回。

实现细节提示

- `attention_resolutions` 用下采样倍率 `ds` 判断是否插入 Attn
- `fea_tran` 与收集到的尺度一一对应（长度断言）
- 支持 `fp16/fp32` 转换（主要作用于编码与瓶颈）

EncoderUNetModelWT：结构示意图



注：虚线箭头表示“保存该尺度特征”；点划线表示时间嵌入注入。

谢谢大家！