

# 基于 JIT 的 diffusion 超分辨模型

---

李孟霖

December 28, 2025

# Introduction

---

# Introduction

---

## 研究背景

# 主流超分辨模型范式

## 回归型方法 (Pixel-wise Regression)

- **EDSR (CNN)**: 以 MSE / L1 为目标, PSNR/SSIM 表现稳定
- **SwinIR (Transformer)**: 更强建模能力, 当前 SR 指标基线

## 生成型方法 (Generative Models)

- **GAN-based SR** (如 SRGAN): 提升感知质量, 但训练不稳定、指标与视觉存在权衡
- **Diffusion-based SR** (如 StableSR): 逐步去噪, **高频细节表达能力强**

## 动机

Diffusion 模型在生成任务中往往具有较强的高频细节建模能力, 本任务中尝试将 **Diffusion + Transformer backbone** 引入超分辨任务, 并分析其在像素对齐与空间结构约束上的表现与局限。

# Diffusion 去噪网络的 Backbone 选择

## 常见去噪结构

- **UNet**: 多尺度卷积结构，局部归纳偏置强，在超分辨等低层视觉任务中被广泛采用
- **DiT / JiT (Transformer-based)**: 使用 ViT 结构作为去噪网络，通过 self-attention 建模全局依赖

## Transformer 的潜在优势

- **全局建模能力强**: self-attention 可直接建模远距离像素/patch 关系
- **结构灵活**: 天然适合 token-level 条件注入 (时间、类别或图像条件)
- **统一建模范式**: 避免多尺度卷积结构的复杂设计

基于以上考虑，本研究采用 **Transformer 作为 Diffusion 的去噪模块**，并进一步研究其在 **图像条件超分辨率任务**中的表现与局限。

# Diffusion 模型的去噪预测方式

## 三种参数化方式

- **$\epsilon$ -prediction**: 直接预测噪声  $\epsilon$ , 是早期 Diffusion 模型的常见形式
- **$x$ -prediction**: 预测原始干净图像  $x_0$ , 更直观, 常用于高分辨率生成
- **$v$ -prediction**: 预测速度变量  $v$ , 在不同噪声强度下具有更稳定的梯度尺度

## 本任务中的实现方式

- 网络前向输出采用  **$x$ -prediction**
- 训练时通过固定变换, 将  $x$ -prediction 转换为  **$v$ -prediction loss**
- 该做法与 JIT / EDM 等工作保持一致

## v-prediction 的定义

Diffusion 前向过程定义为：

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

v-prediction 定义为：

$$v_t = \alpha_t \epsilon - \sigma_t x_0$$

在训练中，模型预测  $\hat{x}_0$ ，并通过固定变换得到  $\hat{v}_t$ ，使用 v-prediction loss 进行优化。

## Introduction

---

### 相关工作

## JiT 的基本思想

JiT (Back to Basics: Let Denoising Generative Models Denoise) 认为，  
真实数据分布位于高维图像空间中的**低维流形**，  
使用 Transformer (ViT) 作为 Diffusion 的去噪网络。

## JiT: x-prediction 与 v-prediction

前向加噪过程

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

模型预测  $\hat{x}_0$ , 并通过固定变换构造  $v$ :

$$\hat{v}_t = \frac{\hat{x}_0 - x_t}{1 - t}, \quad v_t = \frac{x_0 - x_t}{1 - t}$$

训练目标:

$$\mathcal{L} = \mathbb{E} \left[ \|\hat{v}_t - v_t\|_2^2 \right]$$

## 低清图插入方式

- 将低清图和带噪声图像按通道拼接
- 将低清图编码为 token, 和带噪声图像拼接成一个序列

# 实验

---

# 实验

---

## 性能指标

# 通道拼接 (Concat) 方案：性能指标

PSNR: 6.98      SSIM: 0.146

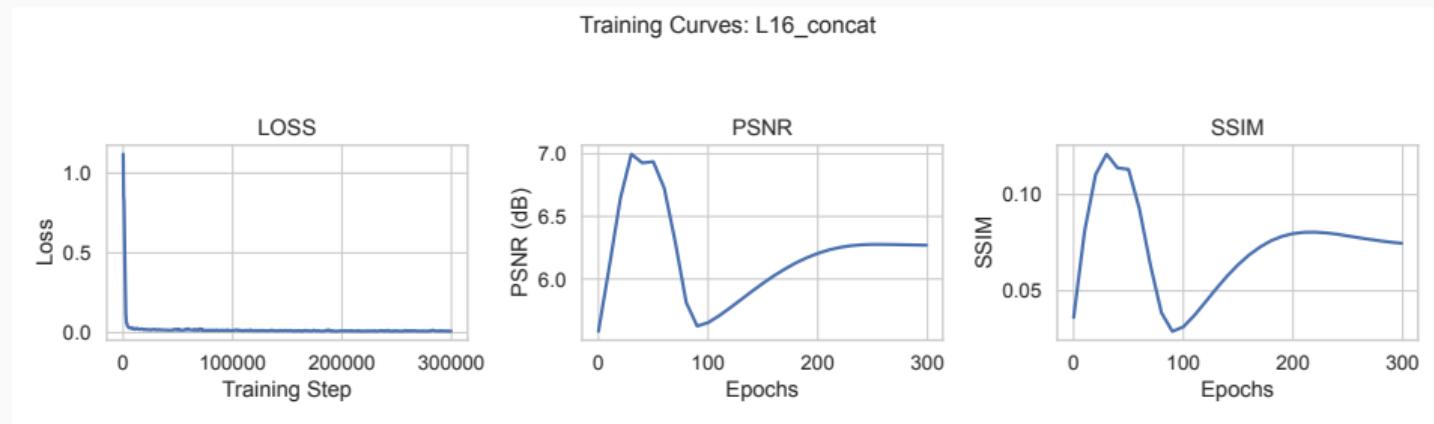


Figure 1. L16 通道拼接方案在验证集上的性能曲线

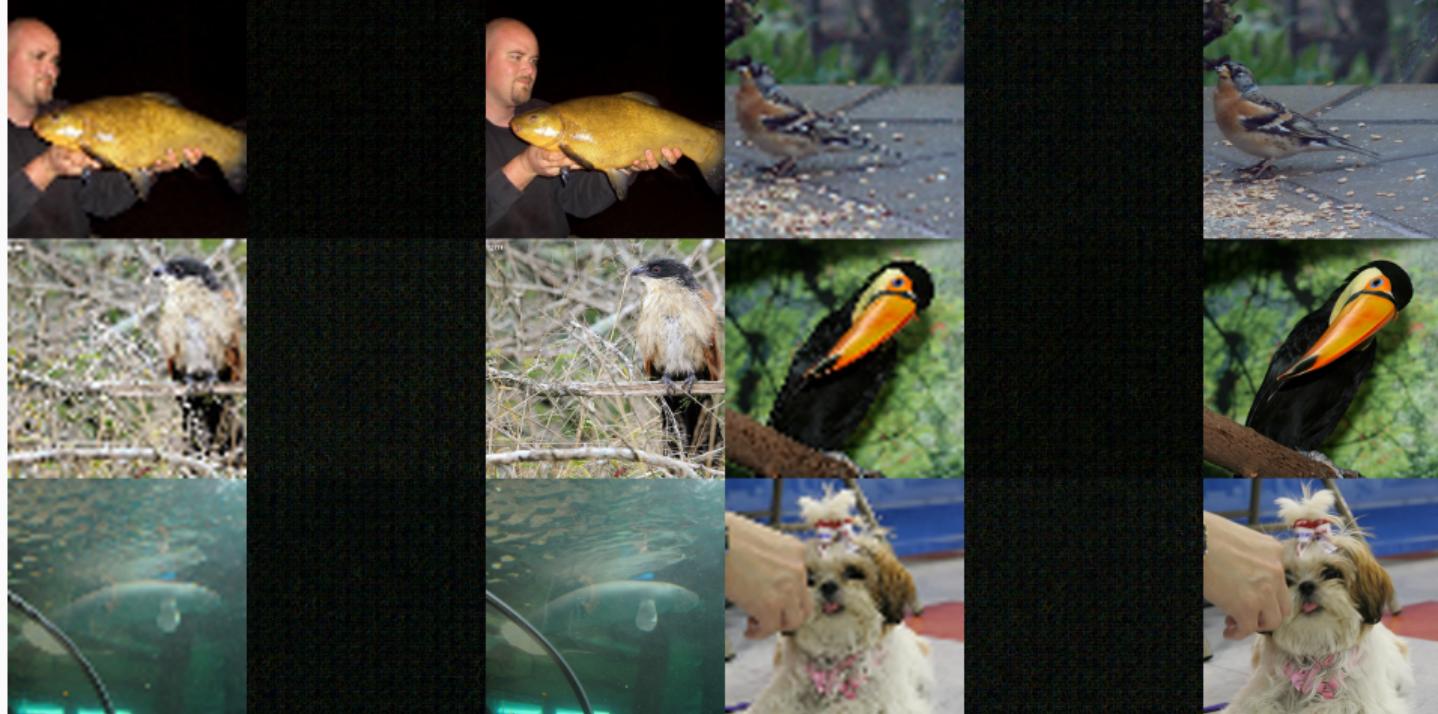


Figure 2.L16 通道拼接方案效果图

# Token 拼接方案：性能指标

PSNR: 20.47

SSIM: 0.617

参数量: 131M

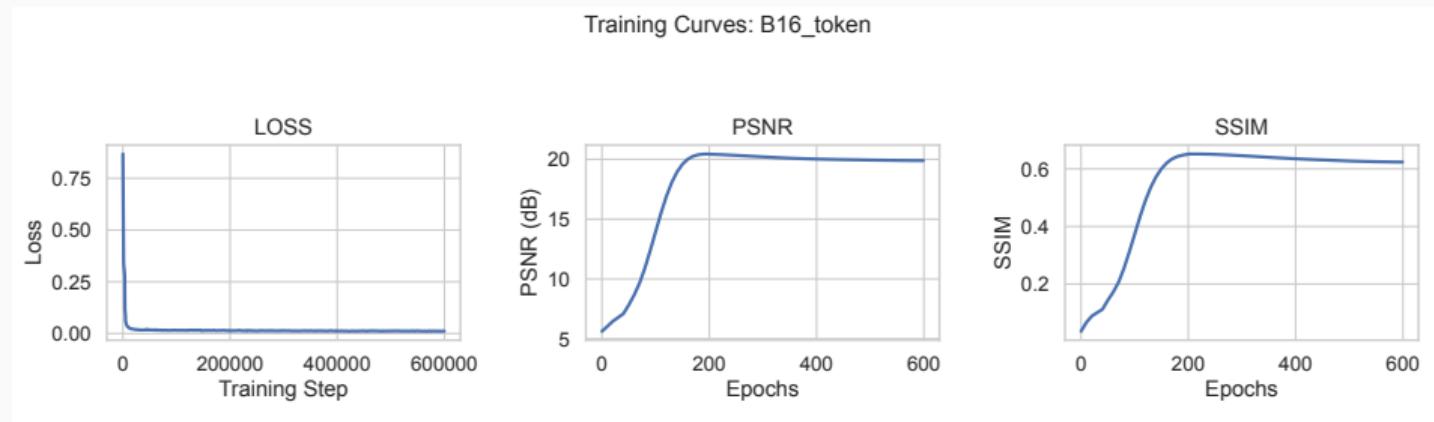


Figure 3. B16 Token 拼接方案在验证集上的性能曲线

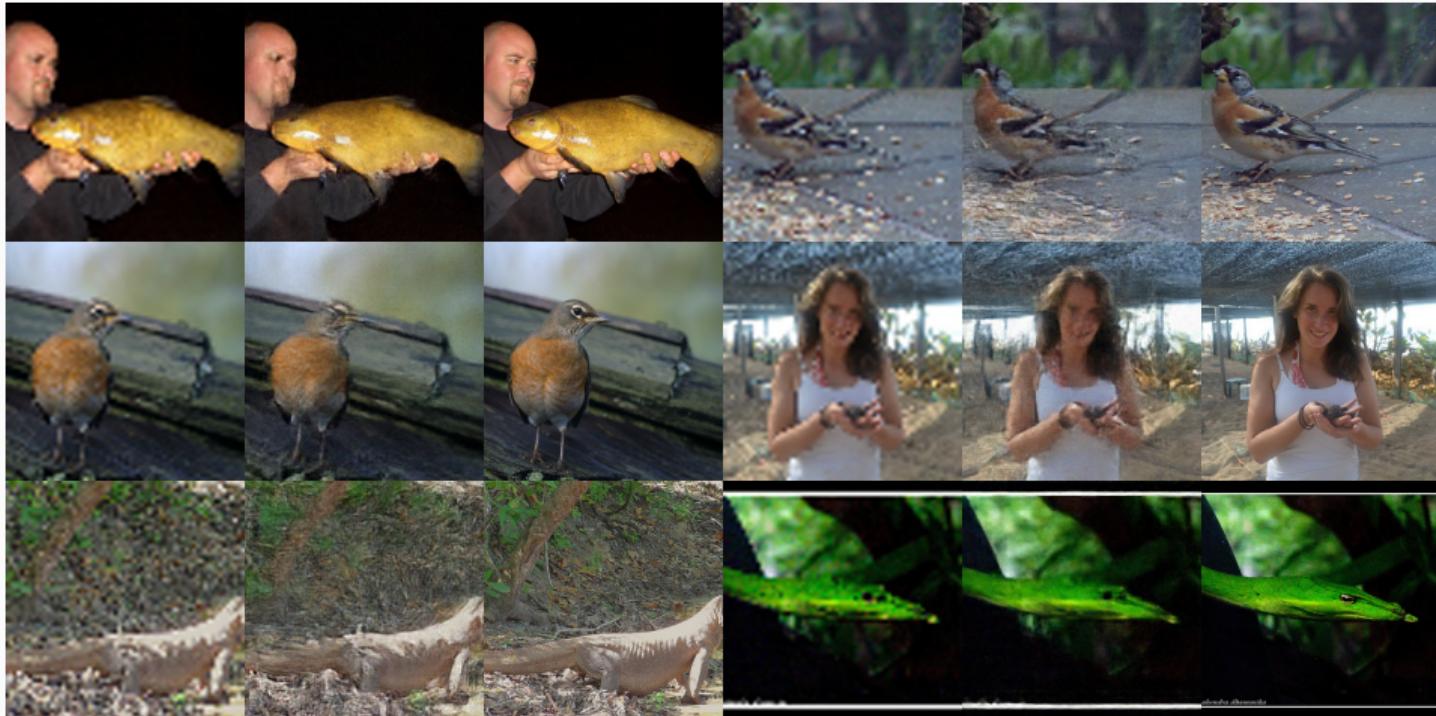


Figure 4.B16 token 拼接方案效果图

# Token 拼接方案：性能指标

PSNR: 21.10

SSIM: 0.634

参数量: 462M

Training Curves: L16\_token

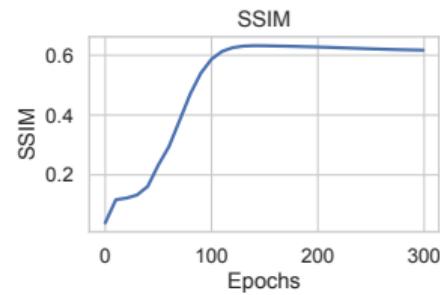
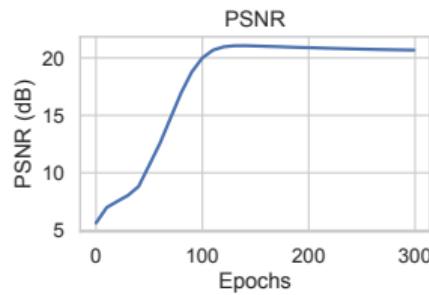
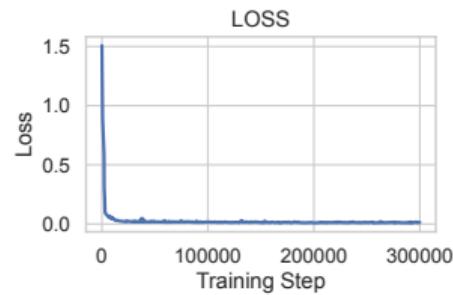


Figure 5. L16 Token 拼接方案在验证集上的性能曲线

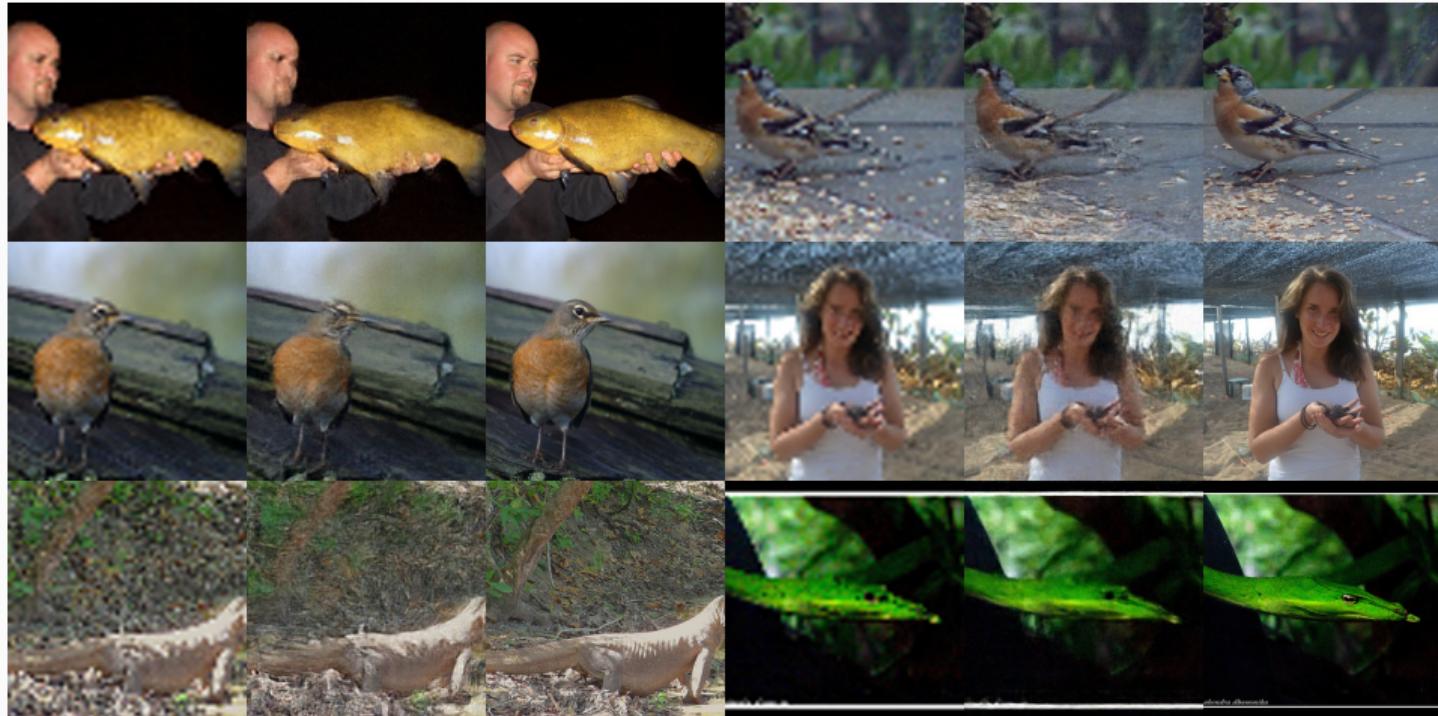


Figure 6.L16 token 拼接方案效果图

## 结果分析

---

谢谢大家！