基于区域的通用图像编辑器

陈晓桐

摘要

本文利用 CLIP 模型和掩码实现了基于区域的通用图像编辑器，并添加 DDMP 模型使得掩码区域和非掩码区域自然连接，最终生成高质量的编辑后图像。同时，设计了一个通用的掩码生成器，令用户可以根据自我的需求通过简单的操作得到掩码图像，降低了图像编辑器的使用复杂度，更利于图像编辑器的广泛应用。

关键词：图像编辑器；CLIP 模型；DDMP 模型；掩码生成器

1 引言

近年来，随着图像处理技术的提升和计算机硬件设备的更新迭代，我们见证了深度学习在计算机视觉的飞速发展。由 Goodfellow 等人[1]在 2014 年提出的生成对抗网络 (Generative Adversarial Network，GAN) 更是作为深度学习的一项重点突出工作，被广泛运用于各大领域，如数据增强[2]、图像生成[3]、图像超分辨率[4]等，且在各领域取得了惊人的效果。

深度学习主要利用多层网络结构和设置超参数来对数据样本进行学习，从而获取到样本数据集中的隐含信息，如样本特征、样本分布等，并将学到的模型运用到目标样本中。一般而言，网络的层数越深则获取到的信息越多，更有利于提高模型的泛化性。但由于在深度学习中，往往需要提供大量的数据样本用于训练，而数据收集的工作则需要耗费大量的人力资源和时间，且还要考虑到数据隐私问题，若在预先收集好的数据集上进行训练，则会侧重于数据集的特征分布，未能完全与所期望的训练样本要求一致，可能会导致模型迁移应用效果较差等问题。为解决以上问题，不少学者基于对抗网络的思想上提出了各种图像生成模型。Zhang[5-6]等人提出的以两个 GAN 堆叠形成的 StackGAN 和 StackGAN++ 模型，Zhang[7]等学者基于 GAN 额外增加了自注意力机制，提出了 SAGAN 模型。Qiao 等学者则提出了 MirrorGAN[8]，可以利用级联图像生成的全局-局部协同关注模块逐步增强生成器的多样性。

实际上，对抗网络是在博弈的思想上形成的。通过生成器和判别器组成对抗网络，生成器以判别器的结果优化图像的生成方式，而判别器则用于判别图像的真伪，两者相互博弈，当判别器不再能正确区分图像真假时，则达到博弈点，模型训练结束，认为模型已经掌握了真实数据的特征分布信息。

值得注意的是，基于 GAN 的图像生成器不仅可以用于生成训练集，还被应用于各类应用软件中。用户可以通过应用软件对输入图像进行图像风格的变化，即对指定区域进行图像编辑工作，如 AI 生成自我人物的动漫图像就收到了年轻人的广泛关注，从而实现了图像生成器商业价值。得益于 GAN 的优越性，大多图像编辑的工作都基于 StyleGAN[9]和 StyleGAN2[10]实现。

为扩大图像生成器的应用范围，满足用户更高的使用需求，则需要对图像的局部特征进行编辑工作，例如修改人物五官特征、修改图像背景信息等。2021 年，Open AI 发布了 CLIP[11]模型，使得文本可以作为监督信号来训练模型，证明文本驱动图像编辑工作的可行性，随后基于 CLIP 则提出了

StyleCLIP[12]模型。由于 StyleCLIP 生成器只能对全局图像进行编辑，也存在因图像编辑而造成图像背景改变等问题，且无法构建一个通用的图像编辑器。因此，如何在不改变非区域内图像内容的情况下，构建一个基于区域的通用图像编辑器成为了本文的重点研究工作。

2 相关工作

目前，关于图像编辑工作的实现方式主要可以分为基于 StyleGAN 的图像编辑、基于 StyleGAN2 的图像编辑和基于 StyleCLIP 的图像编辑，下文将对这三种实现方式进行详细的说明。

2.1 基于 StyleGAN 的图像编辑

StyleGAN 是一种基于样式的图像编辑生成器，应用于风格迁移领域使得用户可以根据需要对生成图像中特定属性进行编辑，如头发、眼睛等。生成器以可学习常量作为开始的输入，然后根据隐码调整每个卷积层的图像风格，从而直接控制不同尺度下图像特征的强度。再通过结合网络噪声，实现了对高级属性（人脸姿势，身份等）和随机变化（头发，眼睛等）的无监督分离，以及直观的尺寸特征混合和插值操作。StyleGAN 架构图如图 1 所示。

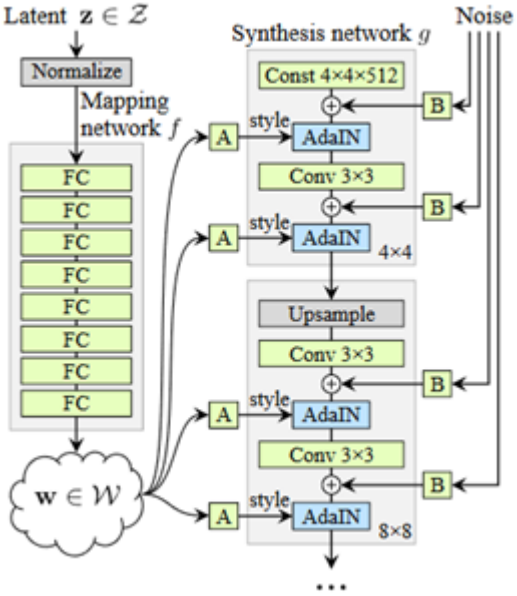

图 1: StyleGAN 架构图

由图 1 可知，StyleGAN 主要由 8 层的映射网络和 18 层的合成网络组成，其中，“A”代表学习的仿射变换，“B”代表将学习到每个通道的缩放因子应用于高斯噪声的输入。与 ProGAN[13]生成器仅通过输入层提供隐码不同，StyleGAN 生成器则引入了风格向量，利用原来的隐藏空间 $\mathcal{Z}$ 中特征之间相互关联、耦合，并通过多层感知机将输入映射到中间的隐藏空间 $\mathcal{W}$，然后在卷积层中通过连接自适应实例归一化 (AdaIN) 来控制生成器的输出，使得特征之间独立性增强，便于对特定属性进行编辑处理。同时，在评估非线性之前，还需要在卷积层后添加高斯噪声。StyleGAN 生成器生成的图片如图 2 所示。

图 2: StyleGAN 生成器生成的图片

从图 2 可以看出，StyleGAN 生成器生成的图片不仅具有较高的分辨率且十分逼真，几乎可以毫无痕迹地编辑头发、肤色等属性，使得 StyleGAN 生成器一经提出就受到了广泛的关注。

2.2 基于 StyleGAN2 的图像编辑

NVDIA 公司在 StyleGAN 的基础上，进一步提出了 StyleGAN2 生成器，通过该生成器可以生成更高分辨率的图片，并且有效修复了 StyleGAN 生成器所生成图片存在特征伪影的问题。StyleGAN 生成器的特征伪影如图 3 所示。

图 3: StyleGAN 生成器的特征伪影

特征伪影的出现主要是 AdaIN 操作造成的，由于 AdaIN 操作会对每个特征映射的均值和方差进行归一化，从而可能破坏了层与层之间传递的信息，而特征伪影则是防止信息被破坏而生成的。因此，作者为防止特征伪影的生成，重现设计了 StyleGAN 的架构，即 StyleGAN2 架构，如图 4 所示。

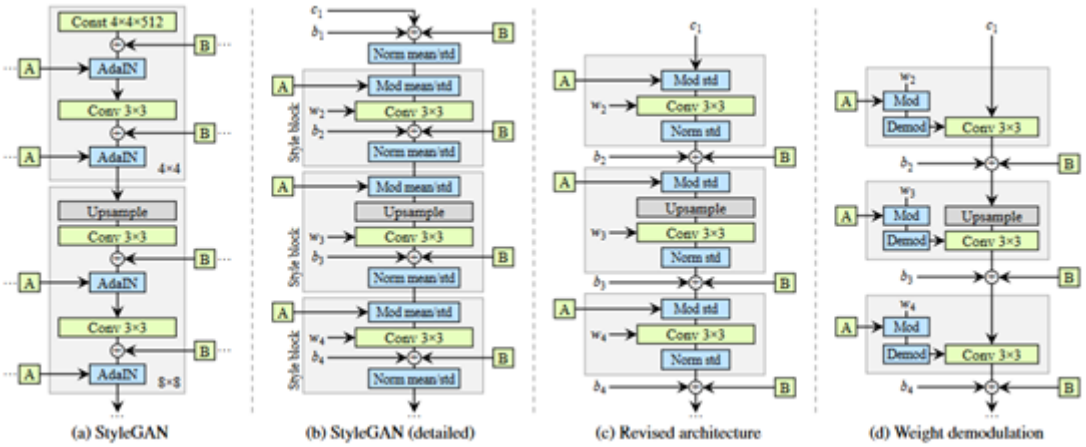

图 4: StyleGAN2 架构

图 4(a) 为 StyleGAN 的架构。图 4(b) 将 AdaIN 操作分解为实例归一化和 Style 调制，两者都需要对每个特征图的均值和标准差进行处理，且需要添加权重和偏差。图 4(c) 将原来 StyleGAN 架构中对均值的操作，并改变了噪声 B 添加的位置。图 4(d) 则将对特征图的修改转变为对卷积权重的修改，利

用解调操作替换实例归一化，并将其应用于与每个卷积层相关联的权重中。这就是 StyleGAN2 生成器的构造过程。

2.3 基于 StyleCLIP 的图像编辑

StyleCLIP 是指基于 StyleGAN 图像编辑器生成文本驱动图像的方法。该方法允许用户使用自然语言描述控制生成图像的各种属性，如颜色、纹理和整体外观等，且生成图像具有较高的质量，拥有逼真的纹理、清晰的细节和一致的风格，使得生成图像无法被肉眼直接判断是否为真实图像。StyleCLIP 的提出，是基于文本输入驱动生成图像样式的一种创新方法，推动了该领域的研究和发展。基于 StyleCLIP 的图像编辑如图 5所示。

图 5: 基于 StyleCLIP 的图像编辑

由上图可知，StyleCLIP 可以对人脸、动物、教堂等图像内容进行广泛语义操作，且文本内容可以进行具体或抽象的语义描述，如具体的语义描述“狮子”和抽象的语言描述“素颜”等。通过观察生成图像的质量，可以进一步证明 StyleCLIP 可以利用 StyleGAN 和 CLIP 模型轻松得到具有高分辨率的生成图像，具有一定的优越性，其中 CLIP 模型是指对比语言-图像预训练模型。StyleCLIP 架构如图 6 所示。

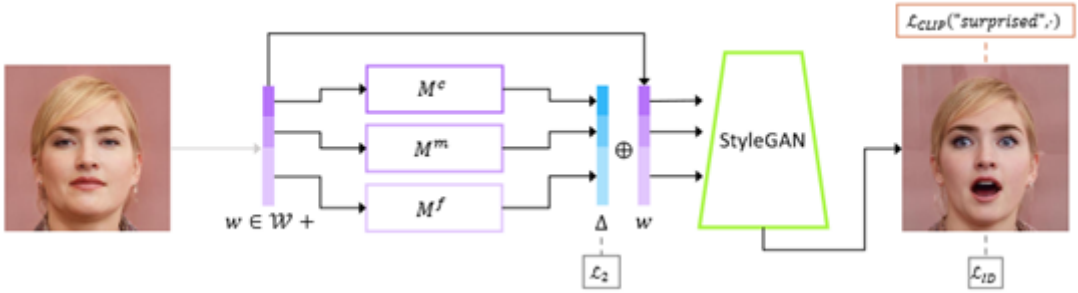

图 6: StyleCLIP 架构

StyleCLIP 由 StyleGAN 和 CLIP 模型组成，源图像作为模型的输入，得到图像的数据特征，再构造三个独立的映射函数用于计算模型残差，并以计算得到的残差和图像的数据特征来驱动 StyleGAN 模型的生成，最后基于 CLIP 和身份的损失函数来评估生成图像的质量。

3 本文方法

3.1 本文方法概述

由于当前普遍的图像编辑器都只能通过用户输入的自然语言都图像的整体区域进行编辑，这可能会导致图像的背景等部分区域进行了不被用户所期望的修改，导致生成的图像不能充分的满足用户的

需求。为了更好的解决这个问题，让用户拥有更好的使用效果，Avrahami[14]等学者提出了一个基于自然语言和掩码输入对图像局部区域进行编辑的模型，该模型结合了 CLIP 模型，利用用户输入的自然语言来引导图像编辑工作，并使用 DDPM 模型来增强图像的自然性，DDPM 模型即去噪扩散概率模型，最终使得编辑后的图像能在局部区域和非局部区域之间自然连接，生成用户所需的高质量图像。同时，还提供了对图像中掩码区域进行添加、删除、替换等功能，进一步的扩大了图像编辑器的可用范围，让图像编辑器被更多的用户所广泛使用。基于区域的通用图像编辑器的实现效果如图 7 所示。

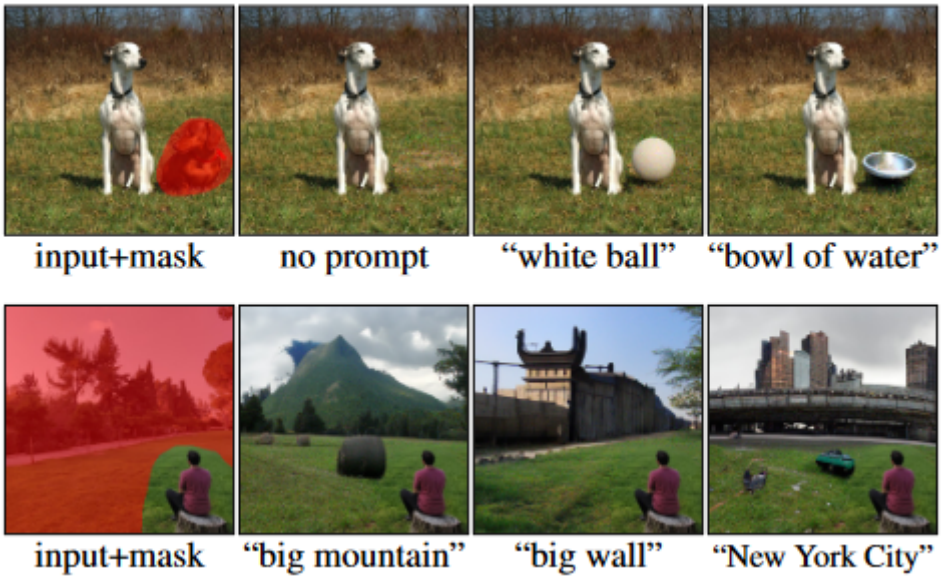

图 7: 基于区域的通用图像编辑器

3.2 CLIP 模型

关于图像分割工作，大多使用源域图像进行训练，使得训练好的分割网络无法完美地应用于目标域，通常需要进行额外的处理，添加目标域信息，以达到更好的分割效果。CLIP 模型使用来自网络的 4 亿对真实图像和文本数据进行文本匹配任务训练，利用文本作为监督信号学习图像特征，最终实现输入一段文本（或者一张图像），输出文本（图像）的向量表示。CLIP 模型如图 8 所示。

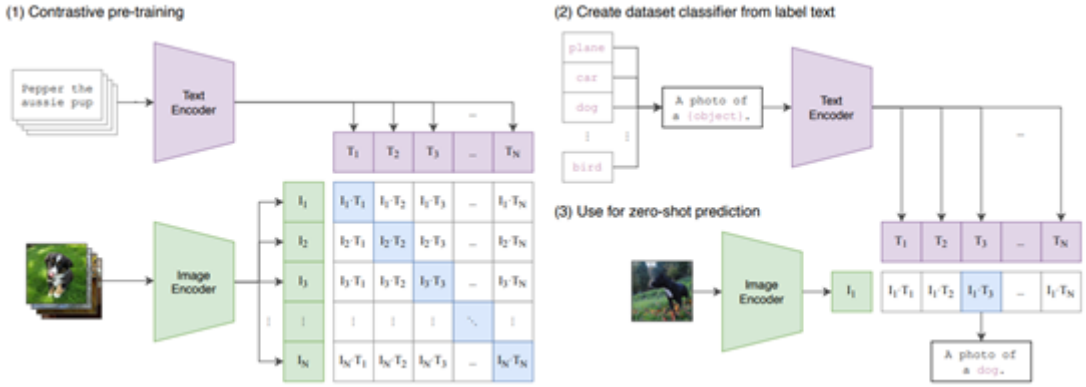

图 8: CLIP 模型

由上图可知，CLIP 模型主要分为三个阶段，分别是预训练、构建数据集分类器和预测阶段。预训练是由一个文本提取器和一个图像提取器构成，文本提取器负责提取输入文本的特征，可使用常见基于文本的 Transformer 模型，图像提取器则负责提取输入图像的特征，可使用 CNN 模型，输出图像文本对的向量表示。数据集分类器的主要任务是根据分类标签建立每个类别的文本描述，得到相应分类任务的文本特征。最后在预测阶段，可以根据输入的图像或文本，计算余弦相似度，选择相似度最高

的文本或图像作为输出结果。

因此，可以使用 CLIP 模型来指导图像编辑工作，利用用户输入的自然语言来驱动图像编辑。

3.3 DDPM 模型

DDPM 是基于扩散过程提出的，实际上就是学习参数化马尔科夫链的过程。该过程通常包括两个过程，一是前向马尔科夫噪声过程，二是马尔科夫逆向过程，也称采样过程。在前向过程中，对原始图像 x_0 逐步加入高斯噪声，直到原始图像被完全破坏，得到随机高斯噪声图像 x_T。在逆向过程中，则可以通过对图像 x_t 进行逐步去噪处理，最终得到原始图像 x_0。DDPM 模型如图 9 所示。

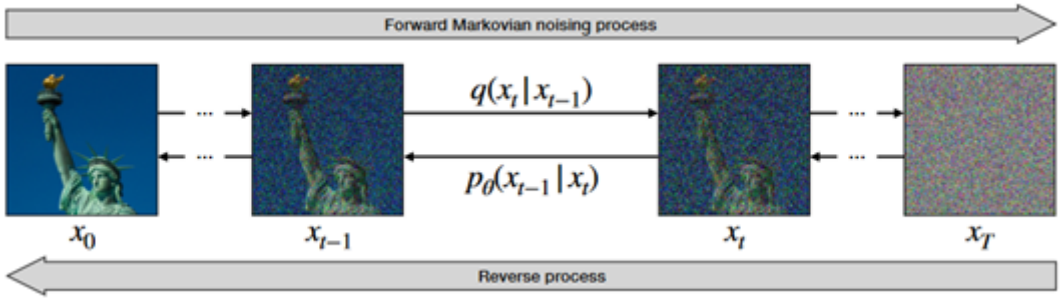

图 9: DDPM 模型

前向过程可以被定义为：

$$x_t = \sqrt{a_t}x_{t-1} + \sqrt{1-a_t}\varepsilon_{t-1} \tag{1}$$

其中，$\{a_t\}_{t=1}^{T}$ 是噪声时间表，是一个预先设置好的超参数，ε_{t-1} 是高斯噪声。

后向过程则可以被定义为：

$$x_{t-1} = \frac{1}{\sqrt{a_t}}x_t - \frac{\sqrt{1-a_t}}{\sqrt{a_t}}\varepsilon_\theta(x_t, t) + \sigma_t \tag{2}$$

其中，ε_θ 是 DDPM 模型中需要训练的噪声预测模型，σ_t 是高斯噪声。噪声预测模型的损失函数由噪声预测模型和真实噪声组成，如公式 3 所示。

$$\mathcal{L} = \|\varepsilon - \varepsilon_\theta(x_t, t)\|^2 \tag{3}$$

从而，DDPM 的训练过程就是通过在原始图像中逐步生成随机噪声，用于破坏图像，进而使用破坏图像估计噪声，通过计算估计噪声和真实噪声的损失，来恢复生成原始图像。近年来，已经有 Ho[15]和 Dhariwal[16]等学者有力地证明了 DDPM 模型可以有效应用于图像编辑工作中，得到编辑后的高质量图像。

3.4 损失函数定义

首先，我们利用 CLIP 模型来指导图像中掩码的编辑工作，并结合掩码外区域共同构成该模型的损失函数，再添加 DDPM 模型来减少两区域之间的差异性，最终得到无缝连接的编辑后的图像。因此，基于区域的通用图像编辑器的架构如图 10所示。

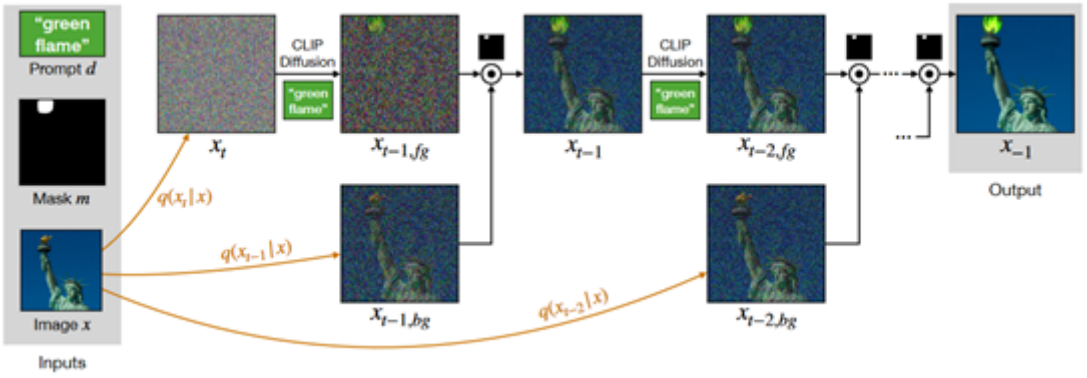

图 10: 基于区域的通用图像编辑器的架构

图像掩码区域基于 CLIP 模型的损失函数 $\mathcal{D}_{CLIP}$ 可以被定为：

$$\mathcal{D}_{CLIP}(x,d,m) = D_c\left(CLIP_{img}(x \odot m), CLIP_{txt}(d)\right) \tag{4}$$

其中 D_c 表示余弦距离，x 为给定图像，d 是用户输入的自然语言，m 是掩码，$(x \odot m)$ 是特定的乘法，表示乘法两边的对象应尽可能相似。使用上式可以有效的利用 CLIP 模型来指导特定区域的编辑设置。为了保障非掩码区域的完整性，我们需进一步添加背景的损失函数 $\mathcal{D}_{bg}$，可以被定义为：

$$\begin{aligned} \mathcal{D}_{bg}\left(x_1, x_2, m\right) &= d\left(x_1 \odot (1-m), x_2 \odot (1-m)\right) \\ d\left(x_1, x_2\right) &= \frac{1}{2}\left(\text{MSE}\left(x_1, x_2\right) + \text{LPIPS}\left(x_1, x_2\right)\right) \end{aligned} \tag{5}$$

其中，MSE 是图像之间像素差的 L2 范数，LPIPS 是学习感知图像块相似性的度量，x_i 是经过去噪扩散过程的图像。最终，该模型的损失函数 $\mathcal{D}$ 由掩码区域的 CLIP 损失函数 $\mathcal{D}_{CLIP}$ 和背景损失函数 $\mathcal{D}_{bg}$ 共同构成，可以被定义为：

$$\mathcal{D} = \mathcal{D}_{CLIP} + \lambda \mathcal{D}_{bg} \tag{6}$$

4 复现细节

4.1 与已有开源代码对比

在复现的过程中，由于每次都需要根据所需编辑的区域调整掩码的结构，这对用户来说是十分不方便的，因此，我们在该模型中添加了新的功能，考虑生成设计通用的掩码生成器，使得用户可以通过简单的操作就可以得到相应的掩码图像。同时，对模型中损失函数的参数进行调整，比较不同的参数结果，寻求最佳参数。

4.2 实验环境搭建

创建相对的虚拟环境，安装 ftfy、regex、matplotlib、lpips、kornia、torch 等相应的包，并使用 CUDA 进行训练。实验环境搭建如图 11所示。

```
scipy              1.9.3
setuptools         65.5.0
six                1.16.0
torch              1.12.1
torchaudio         0.12.1
torchvision        0.13.1
tqdm               4.64.1
typing_extensions  4.3.0
```

图 11: 实验环境搭建

4.3 界面分析与使用说明

只需要在终端输入 python 命令，且给定输入图像、自然语言以及掩码图像即可。界面分析与使用说明如 12图所示。

图 12: 界面分析与使用说明

4.4 创新点

本文的创新点在于实现了一个基于区域的通用图像编辑器，并在该基础上设计了一个掩码生成器，使得用户可以通过简单的操作就可以得到相应的掩码图像，避免了用户在使用图像编辑器时需要花费大量的时间生成掩码图像，降低了图像编辑器的使用复杂性，有利于图像编辑器的广泛推广。

5 实验结果分析

使用新设计的通用掩码生成器对图像掩码区域内的内容进行修改，实验结果如图 13所示。

图 13: 实验结果

由上图可知，生成图像已经具有很高的质量，但仔细观察仍然可以看出具有编辑痕迹。而原论文所提交的实验结果则如 14图所示。

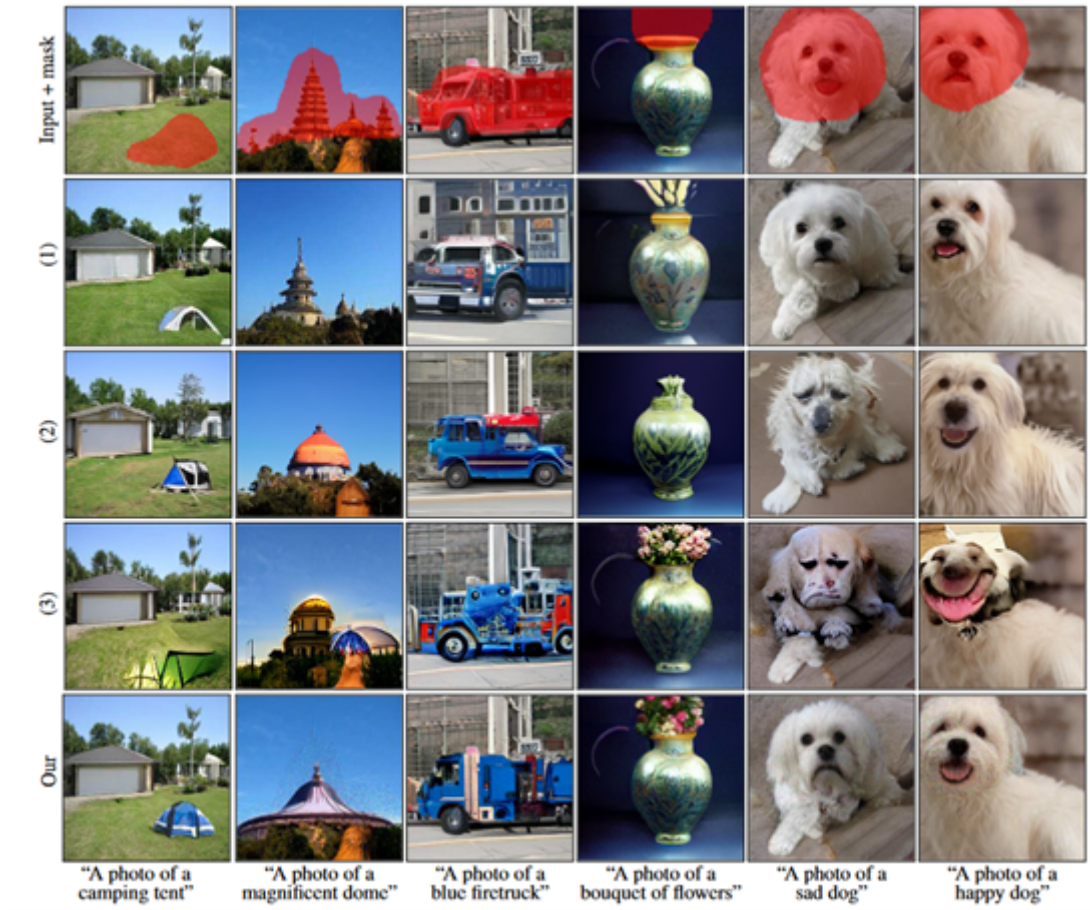

图 14: 原论文所提交的实验结果

通过对比图 13和 14，不难发现我们的工作在生成质量上仍与原工作存在着一定的差距，这可能是由于生成掩码的形状过于规则有关，后续可以通过添加随机过程来改变掩码的规则性，从而提升生成图片的质量。

6 总结与展望

通过本次基于区域的通用图像编辑器的复现，我发现图像编辑器在整体图像编辑工作上已经能很好的满足用户需求，并投入到了实际的商业应用中，且得到了广泛用户的关注和喜爱。但针对特定区域的图像编辑工作还有待提升，本文提出了 DDPM 模型可以在一定程度上推进该工作的发展，但仍然存在很多的不足之处，未来，希望能在 DDPM 模型上进一步改进，或重新设计模型的损失函数，以求生成更高质量的图像。

参考文献

[1] GOODFELLOW I, POUGET-ABADIE J, MIRZA M, et al. Generative adversarial networks[J]. Communications of the ACM, 2020, 63(11): 139-144.

[2] ANTONIOU A, STORKEY A, EDWARDS H. Data augmentation generative adversarial networks[J]. arXiv preprint arXiv:1711.04340, 2017.

[3] SAXENA S, TELI M N. Comparison and analysis of image-to-image generative adversarial networks: A survey[J]. arXiv preprint arXiv:2112.12625, 2021.

[4] CHAN K C, WANG X, XU X, et al. Glean: Generative latent bank for large-factor image super-resolution [C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 14245-14254.

[5] ZHANG H, XU T, LI H, et al. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks[C]//Proceedings of the IEEE international conference on computer vision. 2017: 5907-5915.

[6] ZHANG H, XU T, LI H, et al. Stackgan++: Realistic image synthesis with stacked generative adversarial networks[J]. IEEE transactions on pattern analysis and machine intelligence, 2018, 41(8): 1947-1962.

[7] ZHANG H, GOODFELLOW I, METAXAS D, et al. Self-attention generative adversarial networks[C]//International conference on machine learning. 2019: 7354-7363.

[8] QIAO T, ZHANG J, XU D, et al. Mirrorgan: Learning text-to-image generation by redescription[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 1505-1514.

[9] KARRAS T, LAINE S, AILA T. A style-based generator architecture for generative adversarial networks[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 4401-4410.

[10] KARRAS T, LAINE S, AITTALA M, et al. Analyzing and improving the image quality of stylegan[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 8110-8119.

[11] RADFORD A, KIM J W, HALLACY C, et al. Learning transferable visual models from natural language supervision[C]//International Conference on Machine Learning. 2021: 8748-8763.

[12] PATASHNIK O, WU Z, SHECHTMAN E, et al. Styleclip: Text-driven manipulation of stylegan imagery [C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 2085-2094.

[13] KARRAS T, AILA T, LAINE S, et al. Progressive growing of gans for improved quality, stability, and variation[J]. arXiv preprint arXiv:1710.10196, 2017.

[14] AVRAHAMI O, LISCHINSKI D, FRIED O. Blended diffusion for text-driven editing of natural images [C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 18208-18218.

[15] HO J, JAIN A, ABBEEL P. Denoising diffusion probabilistic models[J]. Advances in Neural Information Processing Systems, 2020, 33: 6840-6851.

[16] DHARIWAL P, NICHOL A. Diffusion models beat gans on image synthesis[J]. Advances in Neural Information Processing Systems, 2021, 34: 8780-8794.