神经图像编辑

清华大学第三十六届 “挑战杯”
学生课外学术科技作品竞赛正式文档

目　录

摘　要

图像与视觉是人类重要的交互途径，然而大部分人却不具有绘画的技能。本项目旨在为非专业人士打破绘画的技术壁垒，只用输入简笔画就可以生成、编辑图像。目前基于神经网络的图像编辑技术都有着生成结果粗糙、编辑质量不稳定而容易失去图像真实性的缺点，尤其是在动漫的领域。我们搭建了表达能力强大的生成对抗网络，取得了在生成的质量和编辑的稳定性上的提升；总结并提出了创新性的下一代网络训练方法；开发了一个简洁方便的网页应用前端，形成了一个智能图像编辑的产品雏形。我们已经能够做到在动漫人脸的领域内对人物的发色、瞳色、发型等进行自由度极高的编辑，在鞋子和手提包的图像领域内进行颜色、光泽、形状等的编辑，并将继续提升编辑能力，扩展到更多领域。

关键词　智能图像编辑，动漫人脸编辑，生成对抗网络

1　引言

图像与视觉是人类重要的交互途径，然而大部分人却不具有绘画的才能。本项目意图在于：以人工智能的手段降低非专业人士在绘图方面的技术壁垒。在我们的作品神经图像编辑中，用户给出简笔画就可以得到算法生成的与预期相似的图画；用户还可以继续完善简笔画，对图像进行符合人类直觉的图像编辑，达到更符合预期的结果。

通过简笔画生成图像的技术有着广泛的应用前景。一个例子是在基于图片的商品搜索中，消费者们脑海中已经有了一件商品的形象，而不知如何找到它，这时用户可以通过简笔画描述出想要的衣服，或者在已有的一件样品上进行更改。另一个例子是当下异常繁荣的网络小说市场。起点中文网上的作品已经有数百万之巨，其中绝大多数作品都是纯文本形式，绝大多数作家都想为自己的作品添加插图、为笔下的人物赋予视觉形象，但是由于插画的耗时与价格而只有极少数作品能有此殊荣。第三个例子是广大的兴趣爱好者团体，他们想为喜爱的作品进行二次创作，有了很好的想法却由于不具有专业的美术技能而极大地增加了创作难度、限制了作品的想象力。最后，专业的美术从业者也能从简笔画生成图像中得到便利，他们能够通过简洁的简笔画更快地展示出心中的想法、从中获得启发、减少重复性的劳动，将精力放在更富有创造性的事务上。

本作品神经图像编辑，立足于人们对于视觉表现技能的需求，通过深度神经网络架起简笔画与真实图像之间的桥梁，方便与丰富人们的生活。

2　项目背景

基于神经网络的图像编辑最早来源于 2016 提出的交互式生成对抗网络 (Interactive Generative Adversarial Network, iGAN) [12]。iGAN 提出了基于优化输入的交互式编辑框架，并在户外风景、教堂、鞋子、手提包数据集上进行了实验。iGAN 能够做到使图片随

着用户的编辑发生变化，但是它存在两个不足，一是生成的质量较低，生成的分辨率仅为 64×64 而且其内容的真假容易被人分辨；二是在编辑过程中图片的生成质量还有进一步的下滑。随后 Neural Photo Editor [1] 的提出改进了生成网络、判别网络，使得在 64×64 分辨率下的生成质量得到了提升，在人脸编辑领域得到了令人满意的结果。生成质量与编辑质量一直是这类方法的瓶颈，至今仍然处于待解决的状态。尤其是当问题变成了动漫相关的图像编辑时，由于此领域内数据集质量的低下，生成质量与编辑质量的问题变得尤为突出，成为本项目的难点之一。

另外一类与神经图像编辑相关的方法以 pix2pix 为代表 [4]，通过输入一个精确的语义地图 (semantic segmentation map) 来产生图像。语义地图指的是一个与原图像大小相等的“地图”，其中每一个像素表明了该像素所属的物体类别。编辑操作通过通过更改输入的语义地图来进行，优点在于生成的质量好于前一种方法，而且其对于物体的形态、边缘有着较高的自由度。但是这类方法的缺点主要有二：

1. 需要含有语义地图的数据集进行训练，对于数据集的要求非常高，在许多领域上都难以实现。

2. 这种编辑只能在有限的范围内改动，只能指定某个区域是什么种类的物体，而不能对物体的具体特征进行修改。

基于第二类方法，在插画线稿上色方面有着不错的成果，例如 PaintsChainer[1] 和 Style2Paint[2]。这种方法的输入是专业级别的线稿，输出是上色以后的图像，通过类似于简笔画的方式来对具体的上色进行调节。这种方法的优点继承了 pix2pix 生成质量较好的优点，但是它只能微调局部的色彩，编辑自由度较低。由于目前的技术瓶颈，这种方法还有另一个应用方面的缺点：它的输入假定了专业人士作为使用者，但是其上色的质量距离专业的水准还有一定距离，使其应用场景有些尴尬。

我们的项目立足于业余用户的简笔画，努力做到面向业余用户的简单而有效的绘图。我们主要与第一类方法进行比较，使用 iGAN [12] 和 NPE [1] 作为两个基线，其效果如图 1 和图 2 所示。这两种基线方法在各自原文的应用领域上都有着较为令人满意的结果，然而在动漫数据集 Getchu 上的表现都充分地体现出了其生成质量和编辑质量上的瓶颈。我们的项目针对这个瓶颈提出了有效的解决方法，得到了明显的提升。

[1] https://github.com/pfnet/PaintsChainer

[2] https://github.com/lllyasviel/style2paints

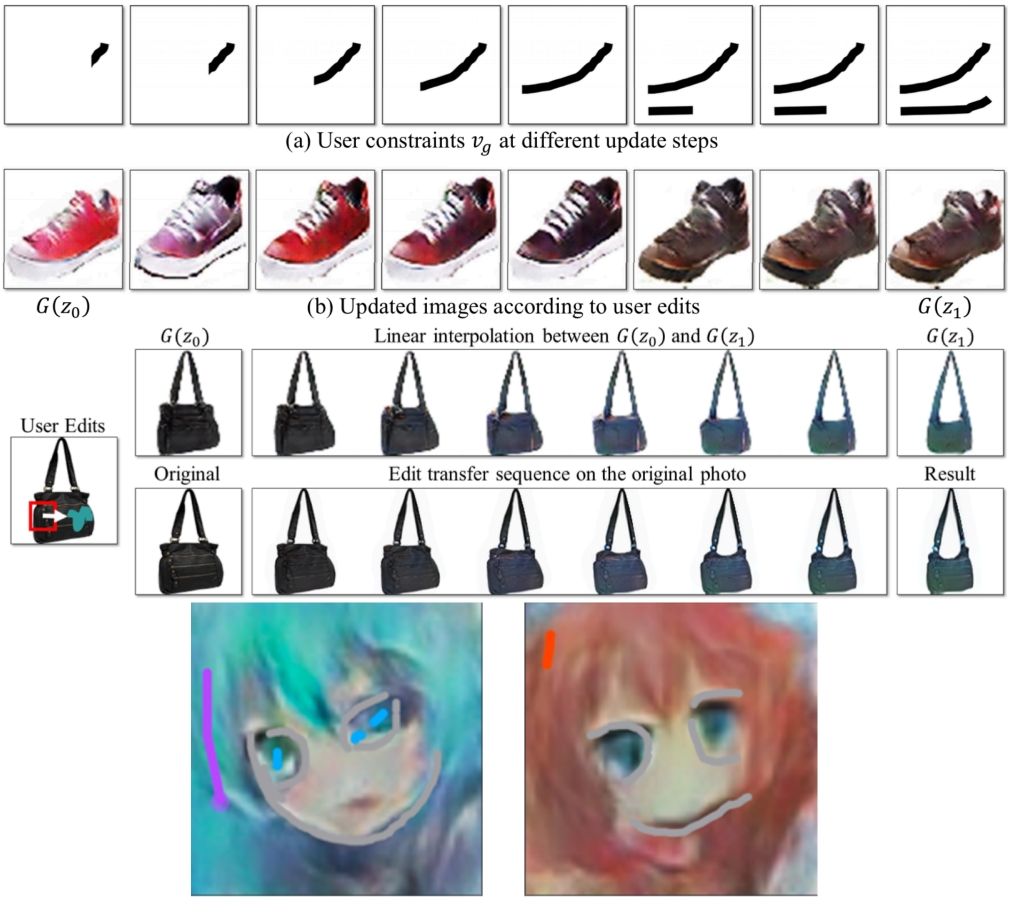

图 1: 基准线 1 iGAN [12] 的效果。上两行的图片来源于原论文，展示了随着用户添加简笔画，鞋子、手提包的颜色随之变化的过程，较为符合预期。下面的图片是在 Getchu 数据集中复现的结果，使用 iGAN 开源的代码，遵循了原文的设置。图中灰色线条是边缘简笔画，其他的彩色线条是颜色简笔画。经过试验，在编辑图像的过程中图片的真实性频繁下滑。展示的图片经过了精心挑选，达到了 iGAN 所能达到的最佳效果，但仍然十分不美观。

图 2: 基准线 2 NPE [1] 的效果。上面的图片来源于原论文，中间的图片是原图片，左右分别是经过编辑的图片。下面的图片是 NPE 在 Getchu 数据集上的复现，使用了 NPE 开源的代码，遵循了原文的设置。从左上到右下是在神经图像编辑过程中顺序产生的，用户在 1 ∼ 5 张中头发的位置画了红色线条，在 6 ∼ 8 张中画了蓝色线条（线条未显示）。头像颜色变化基本与预期相符，体现了编辑的有效性，但是生成的质量和编辑的稳定性不令人满意。

3 项目简介

神经图像编辑项目致力于实现简笔画生成或编辑图像。本项目中，针对第一类基于神经网络的图像编辑方法中的质量瓶颈问题进行了解决方法的探索。我们设计了表达能力更加强大的网络，改进并提出了自己的基于优化输入的系统框架，提升了生成质量与编辑质量。我们在动漫人脸、鞋子、手提包的数据集上进行了实验，效果较两种基线方法有了显著的提升。同时，我们完成了一个简单快捷的 Web 前端，完成了基于简笔画的智能图像编辑的产品雏形 (图 3)。最后，我们提出了一种多阶段生成网络的训练流程，能够将简笔画样例整合入网络的训练中，用以解决简笔画游离在训练数据集之外的问题，融合第二类图像编辑方法。

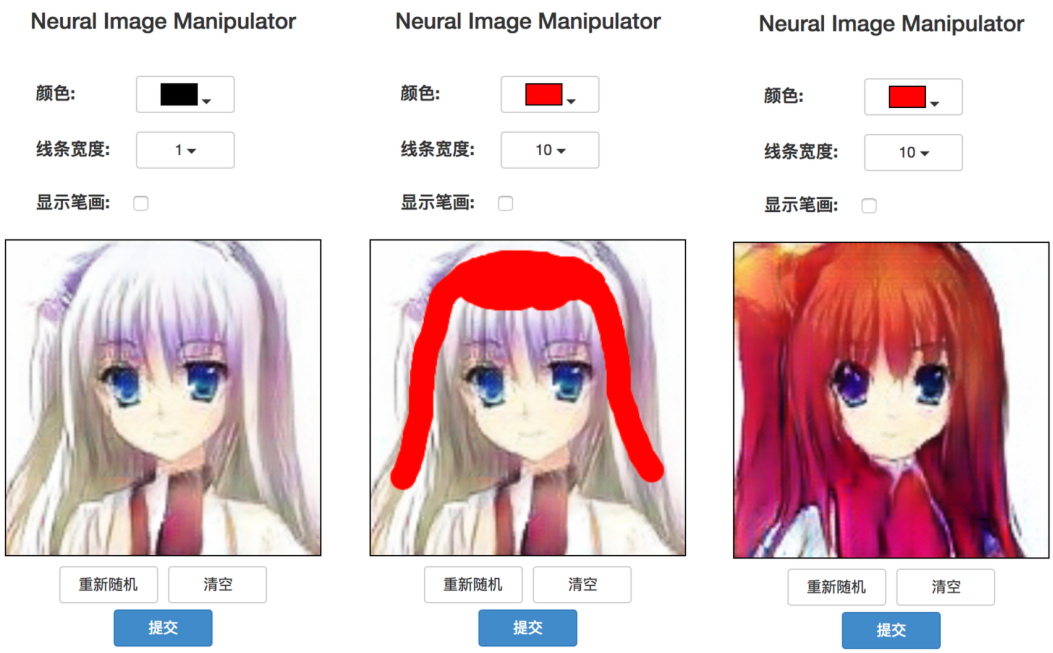

图 3: 项目效果展示。用户通过网页前端绘制简笔画，提交到后台，神经图像编辑系统会将图像向简笔画指导的方向变换一步。经过一次或多次编辑，得到一个符合用户编辑预期的图像。图片中展示的是动漫人脸编辑，此外还有鞋子、手提包编辑。

4 核心技术

神经图像编辑的核心技术分为两部分：网络的训练和图像编辑框架。在网络训练中，我们参考了 [5] 的网络设计了自己的生成网络和判别网络。在图像编辑的框架中，我们参考了第一类神经图像编辑中优化输入的方法，对其进行了改进。

4.1 图像编辑

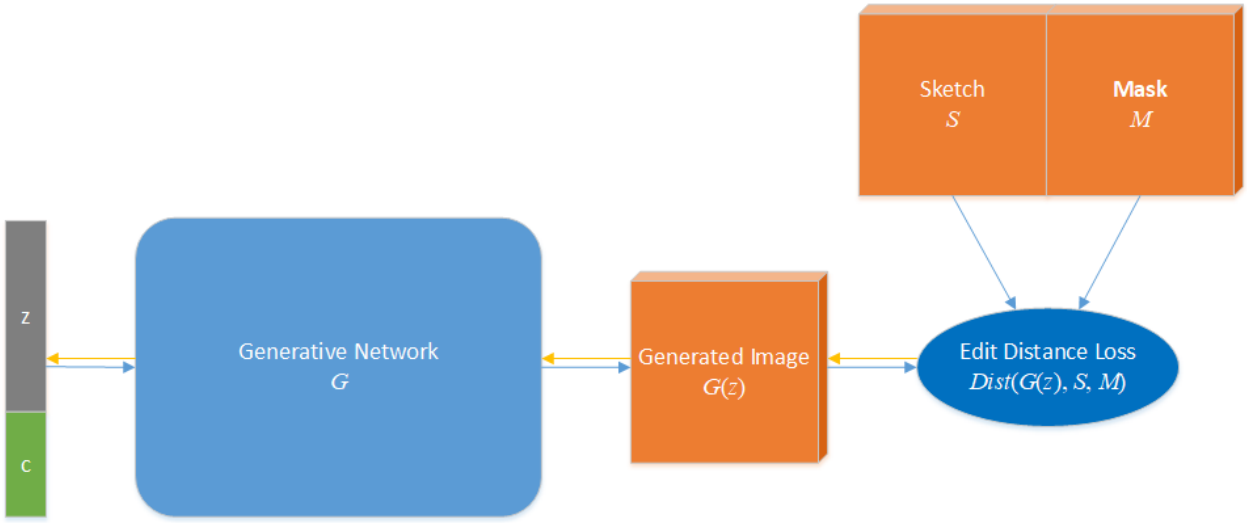

图 4: 编辑图像的实现方法。此时生成网络已经训练完成，从随机向量出发，由生成网络生成初始图像，用户给定简笔画和对应的遮罩，由编辑距离函数计算误差和导数，使用梯度下降更新 z，产生一张新的、与草图相似的图片，达到编辑图像的目的。蓝箭头表示数据流向，橙色箭头表示导数。

我们使用训练好的生成网络 G 实现对于图像的编辑工作。

一个生成网络 $G(z,c)$ 接受随机向量 z（遵循高斯分布）和条件向量 c 作为输入，生成图像 $I=G(z,c)$。同时用户给定编辑图像 I'，例如一副简笔画，则可以据此给出生成图像与编辑图像之间的距离 $Dist(G(z,c),I')$。设 I 在像素点 (i,j) 处的 RGB 颜色向量为 I_{ij}。设 M_{ij} 为像素点 (i,j) 处的遮罩向量，含义是用户编辑的部分是否包含像素点 (i,j)。M_{ij} 的值只可能为全 0 或全 1，定义如下：

$$M_{ij}=\begin{cases}[0,0,0], & \left\|I'_{ij}\right\|=0\\ [1,1,1], & \left\|I'_{ij}\right\|\neq 0\end{cases} \tag{1}$$

则我们可以基于已有的 M_{ij}、I_{ij} 和 I'_{ij} 定义原生成图像与编辑图像之间的距离，这里的距离定义为两幅图中每个用户编辑的像素点距离的 L1-norm 除以一个合理的代表用户修改总点数的值。这样定义的目的是为了仅仅对用户编辑的位置进行修改，且提供与修改像素点数无关的距离函数，避免用户修改的点过少时效果变差。即定义

$$Dist(I,I')=\frac{\sum_{ij}\left|(I_{ij}-I'_{ij})\cdot M_{ij}\right|}{1+\sum_{ij}\frac{|M_{ij}|}{\sqrt{3}}} \tag{2}$$

假设 G 已经得到，本文的基本目标是求使 $Dist(G(z,c),I')$ 最小的 z 和 c，即

$$\arg\min_{z,c} Dist(G(z,c),I') \tag{3}$$

为求这样的 z 和 c，先给出随机的合法初值 z_0 和 c_0，再将距离函数对 z_0 和 c_0 求偏导：

$$\Delta z_0 = \frac{\partial}{\partial z} Dist(G(z_0, c_0), I') \tag{4}$$

$$\Delta c_0 = \frac{\partial}{\partial c} Dist(G(z_0, c_0), I') \tag{5}$$

之后，在 z 和 c 的反函数域上以学习率 lr 使用梯度下降 (Gradient Descent) 算法进行学习，并限制 $\tilde{z}_1$ 和 $\tilde{c}_1$ 在合法的边界中：

$$\tilde{z}_1 = \operatorname{arctanh}(z_0) - lr \cdot \Delta z_0 \tag{6}$$

$$\tilde{c}_1 = -\log\left(\frac{1}{c_0} - 1\right) - lr \cdot \Delta c_0 \tag{7}$$

最后，将 $\tilde{z}_1$ 和 $\tilde{c}_1$ 变换回原来的函数域，得到新的值 z_1 和 c_1：

$$z_1 = \tanh(\tilde{z}_1) \tag{8}$$

$$c_1 = \sigma(\tilde{c}_1) \tag{9}$$

其中 σ 为 sigmoid 函数。重复这样的学习，即可使 $Dist(G(z, c), I')$ 不断减小，直到生成符合要求的图片。

4.2 网络结构

本项目中对于多种网络结构进行了尝试，最终确定了和 [5] 类似的网络结构与训练方法。我们采用了使用多层 Residual Block [2] 搭建的深层生成器和判别器，具体结构如图 5。网络的主要特点是生成器中使用 Subpixel Conv [10] 作为上采样函数，判别器中不使用 Batch Normalization [3]。生成器和判别器的层数都达到了一百层，表达能力强大。

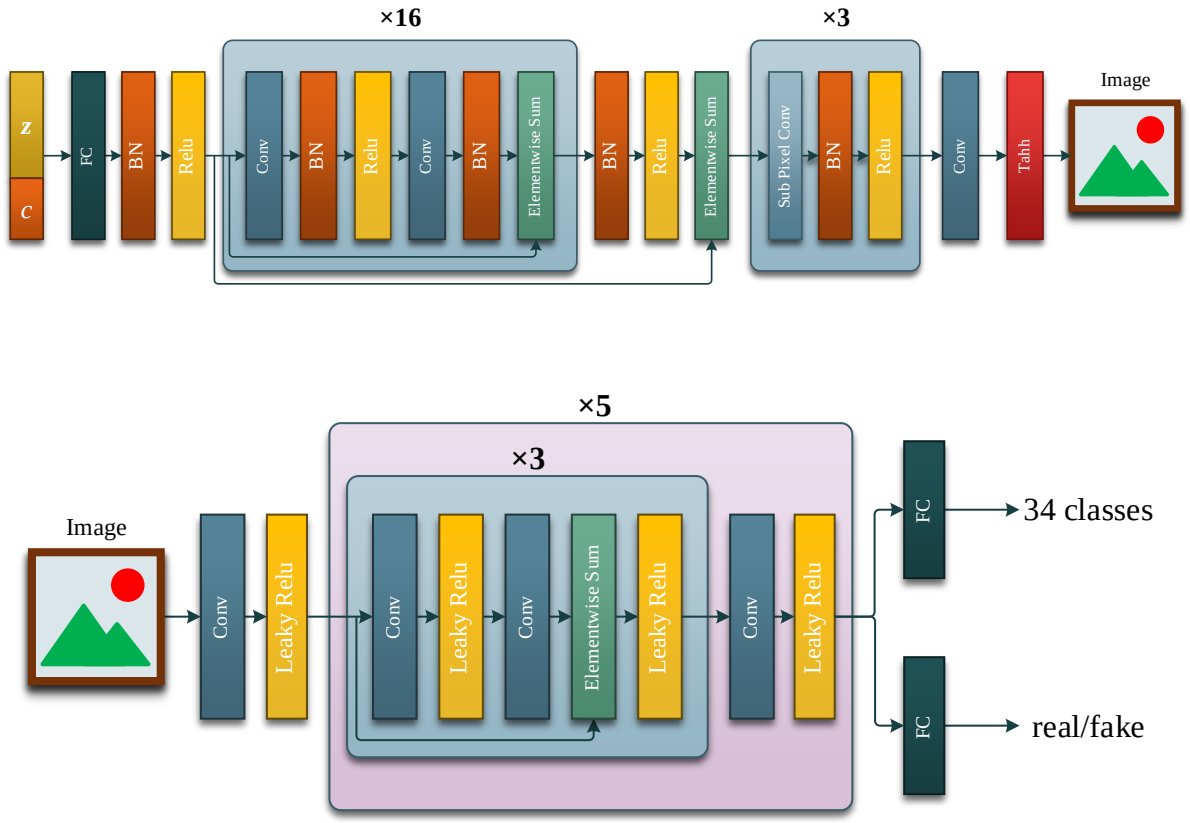

图 5: 上面是生成器模型，下面是判别器模型。生成器输入是随机向量与条件向量，输出是生成的图片。判别器接受图片输入，或是真实图片或是生成图片，输出有两个分支，一个表示真/假，一个是推断出来的条件向量。

4.3 网络训练

我们对多种训练方法进行了尝试，最终使用 DRAGAN [7] 加上辅助分类器 ACGAN [8] 的训练方法进行。

设真实数据集为 D_R，一个训练样例可以表示成 $x \sim P_{data}$。生成器接受随机向量 z 和条件向量 c 的输入，且 $z \sim P_{noise}$，$c \sim P_{cond}$，P_{cond} 表示给定标签下的先验分布。$\mathcal{L}_{adv}$，$\mathcal{L}_{gp}$ 和 $\mathcal{L}_{reg}$ 分别表示 adversarial，gradient penalty 和 weight regularization 的损失函数系数。

使用 DRAGAN 的训练可以用如下公式表达：

$$\mathcal{L}_{adv}(D) = -\mathbb{E}_{x\sim P_{data}}[\log D(x)] - \mathbb{E}_{z\sim P_{noise}, c\sim P_{cond}}\big[\log(1 - D(G(z,c)))\big] \tag{10}$$

$$\mathcal{L}_{cls}(D) = \mathbb{E}_{x\sim P_{data}}\big[\log P_D[label_x|x]\big] + \mathbb{E}_{x\sim P_{noise}, c\sim P_{cond}}\Big[\log\big(P_D[c|G(z,c)]\big)\Big] \tag{11}$$

$$\mathcal{L}_{gp}(D) = \mathbb{E}_{\tilde{x}\sim P_{perturebed\_data}}\Big[\big(\|\nabla_{\tilde{x}} D(\tilde{x})\|_2 - 1\big)^2\Big] \tag{12}$$

$$\mathcal{L}_{adv}(G) = \mathbb{E}_{x\sim P_{noise}, c\sim P_{cond}}\big[\log D(G(z,c))\big] \tag{13}$$

$$\mathcal{L}_{cls}(G) = \mathbb{E}_{x\sim P_{noise}, c\sim P_{cond}}\big[\log P_D[c|G(z,c)]\big] \tag{14}$$

$$\mathcal{L}_{DRAGAN}(D) = \mathcal{L}_{cls}(D) + \lambda_{adv}\mathcal{L}_{adv}(D) + \lambda_{gp}\mathcal{L}_{gp}(D) + \lambda_{reg}\mathcal{L}_2(D) \tag{15}$$

$$\mathcal{L}_{DRAGAN}(G) = \lambda_{adv}\mathcal{L}_{adv}(G) + \mathcal{L}_{cls}(G) + \lambda_{reg}\mathcal{L}_2(D) \tag{16}$$

其中 $\mathcal{L}_2$ 是 L2 正则化损失，$\tilde{x}$ 是经过扰动的真实数据 (σ_x 为 x 的标准差)，

$$\tilde{x} = x + \frac{1}{2}\alpha\sigma_x \tag{17}$$

$$\alpha \sim U(0,1) \tag{18}$$

使用 Adam [6] 优化器迭代交替优化 $\mathcal{L}_{DRAGAN}(D)$，$\mathcal{L}_{DRAGAN}(G)$ 完成训练。

4.3.1 演示模型的训练

最终我们确定的训练设置是：Adam [6] 优化器以 2×10^{-4} 的初始学习率进行 5×10^4 的迭代，然后在接下来 5×10^4 的迭代中线性降低学习率至 1×10^{-5}。网络使用 128 维的随机向量 z 和 34 维的条件向量 c，前者服从 $\sigma = 1, \mu = 0$ 的正态分布，后者在保证类互斥条件下由均匀分布随机产生。各个 loss 的系数是 $\lambda_{gp} = 1.0$，$\lambda_{reg} = 10^{-4}$，$\lambda_{adv} = 2.0$。batchsize = 64。训练在 Titan X GPU 下需要花费大约 40 小时的时间。

图 6 展示了我们使用的演示模型的训练 loss 变化：

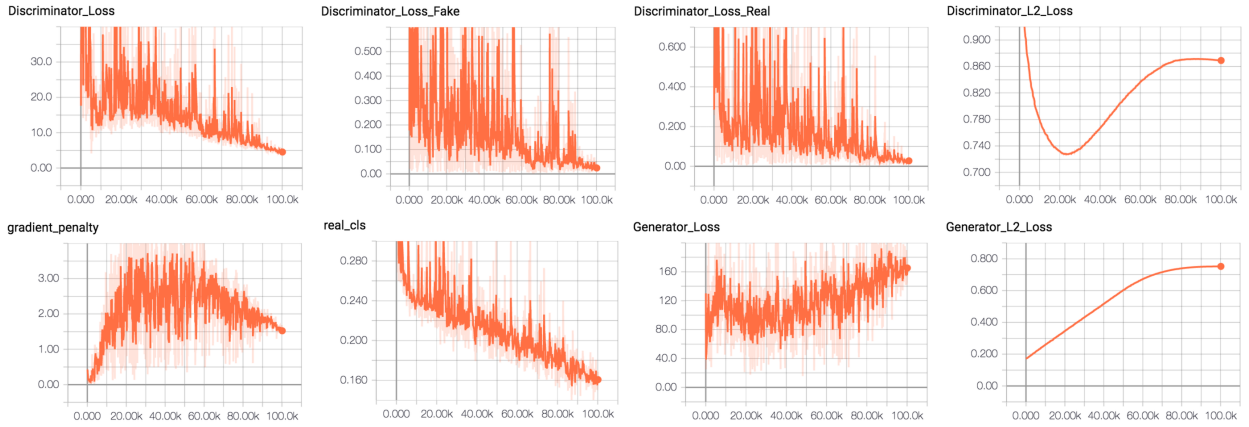

图 6: 演示模型的训练 loss 图。左上角开始的 Discriminator Loss 是判别器的总损失，其右的 Fake 和 Real 损失分别是判别器识别生成图片和真实图片的损失。左下角的 gradient penalty 如上文所述，其右是判别器分类图片的交叉熵损失。再右边是 Generator Loss 的总损失，最右边是两个网络的正则损失。

可以看出生成器的损失除了刚刚开始的震荡较为剧烈以外，整体在稳步下降。查看固定随机数的生成结果，如图 7 所示。从每一个随机种子生成的图片可以看到生成的不断完善、生成结果质量的稳定上升。

图 7: 演示模型的训练中生成图像的演化。从左到右四张图片分别是 25k, 50k, 75k, 100k 迭代时的生成结果。每一张图片包含 16 个固定随机数生成的结果。

4.4 下一代技术

我们经过实践总结，与导师讨论提出了创新的网络结构与训练方案。当前方案中，简笔画未出现在训练过程中，导致一些编辑不符合预期的情况。我们通过将生成网络分为两个阶段，将简笔画整合入网络训练的过程。流程如图 8 所示。这种训练方案除了能够充分利用简笔画信息，达到更好的收敛点以外，还有更好的扩展性，将 G_1 的输入换成语义地图等就可以支持第二类基于神经网络的图像编辑。目前还没有相关的研究将简笔画直接用于训练，也没有将生成分为多个阶段的相关研究，下一代技术创新而充满挑战。

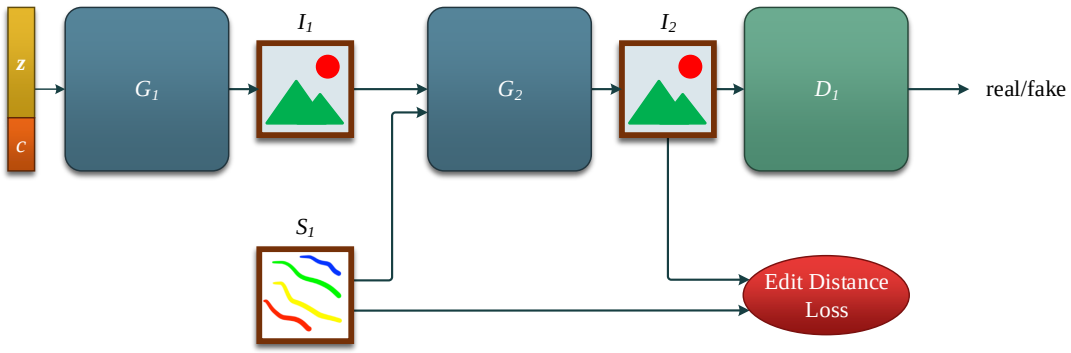

图 8: 两阶段生成网络训练流程。第一阶段的生成接受随机数输入，第二阶段的生成接受第一阶段的结果与一个草图，输出编辑以后的图片。该图片需要保证编辑距离尽可能小，同时进入判别器进行 GAN 训练。

4.5 数据集

我们使用的动漫人脸数据集取自 Getchu[3] 网站。该网站汇聚了大量高质量动漫人物立绘，且图片中仅有单人，背景为白色，很适合作为训练数据。我们使用从该网站上爬取了 22000 张图片，经过 OpenCV [4] 识别人脸，裁剪、缩放等操作得到 128×128 大小的数据集。然后，我们对采集到的图像使用 illustration2vec [9] 进行分类，选取其中与我们的任务相关的 34 类 (illustration2vec 有 512 个类别)，可编码一个 34 维的向量，即条件向量 c，如表 1 所示。向量的第 i 维取值为 0 或 1，如果是 1 表示该图片具有编号为 i 的特征，且同一大类的各个子类中最多只能有一个 1。

鞋子数据集来源于 [11]，手提包数据集来源于 [12]。这两个数据集大小为 128×128，数据干净、质量高，适合 GAN 的训练。

[3] http://getchu.com

[4] https://opencv.org

图 9: Getchu 数据集样例。采用 OpenCV 识别人脸，经过裁剪、缩放到 128×128 的大小。

编号	0	1	2	3	4	5	6	7
类别	发色							
	金色	棕色	黑色	蓝色	粉色	紫色	绿色	红色
编号	8	9	10	11	12	13	4	15
类别	发色					发长		发型
	银色	白色	橙色	天蓝色	灰色	长发	短发	双马尾
编号	16	17	18	19	20	21	22	23
类别	发型		是否脸红	是否微笑	是否张嘴	是否戴帽子	是否有缎带	是否戴眼镜
	卷发	马尾						
编号	24	25	26	27	28	29	30	31
类别	眼睛颜色							
	蓝色	红色	棕色	绿色	紫色	黄色	粉色	天蓝色
编号	32	33						
类别	眼睛颜色							
	黑色	橙色						

表 1: 条件向量的构成。共有 10 个大类和 34 个小类，向量值的某一维为 1，表示具有该维所对应的那个类别的特征。

5 效果展示

图 10: 神经图像编辑在动漫人脸上的编辑效果展示，共六组，每组三幅图。每组中，左中右分别为原图、原画加上简笔画图、结果图。从上到下的编辑方式分别是更换发色、更换瞳色、更换发型。生成图片的质量良好，编辑后图像基本符合预期，且质量未有下滑。

图 11: 神经图像编辑在鞋子上的编辑效果展示，格式同上图。从上到下分别是更换鞋子颜色或光泽、编辑鞋子形状、基于整体草图的编辑。同样可以观察到良好的生成质量，稳定的编辑质量，同基线 1 图 1 进行比较可以看出生成质量上的提升。

图 12: 神经图像编辑在手提包图片上的编辑效果展示，格式同上图。第一行左边是改变颜色，右边是添加光泽，第二行是调整手提包大小，第三行是修改手提包带子形状。同基线 1 图 1 进行比较可以看出生成质量上的提升。

将编辑操作应用于训练好的网络，我们得到了效果优秀的图像生成和编辑效果。如图 10，图 11 和图 12 所示，我们的模型能够生成高质量的图片，并对其进行质量稳定的编辑，而且能够完成多种类的编辑，包括颜色、光泽、形态、增减等，自由度很高。

图 13: 编辑过程中质量的稳定性展示。最右侧是给出的简笔画，前五幅图中，从左到右是原图在简笔画的指导下逐渐变化的过程，发色由原来的白色变化到简笔画所制定的红色。过程中图片的质量没有下降，同时未被编辑部分基本保持了不变，如脸型、眼睛、表情等。与图 2 的编辑过程相比，可以看出神经图像编辑对于编辑稳定性的提升。

在动漫人脸数据集上与两个基线相比（图 1，图 2），我们的效果提升非常明显。首先是生成人物的质量，在分辨率、真实性、画面细节等都有着优势。其次是编辑过程的稳定性出色，如图 13 所展示的，在人物一步步向简笔画指导的方向变换时，图片质量看不出下降。反观图 2 中 NPE 在动漫人脸上面的编辑，过程中的图片一度失去人脸的形态。在鞋子、手提包编辑中，与图 1 中原文的效果相比，在生成图片的分辨率和质量上也有着提升。

从上面丰富的展示材料可以看出，神经图像编辑较为成功地提升了生成质量与编辑质量。

5.1 网页应用

我们基于训练好的模型，开发了一个基于 B/S 架构的网页应用，可以通过 Web 浏览器进行在线图像编辑。

5.1.1 网页端

我们的网页端界面如图 14 ~ 图 16 所示，能够运行在任何一款现代浏览器上。

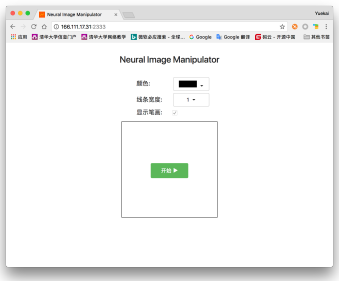

图 14: 开始页面

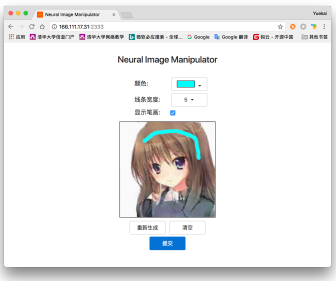

图 15: 输入笔画

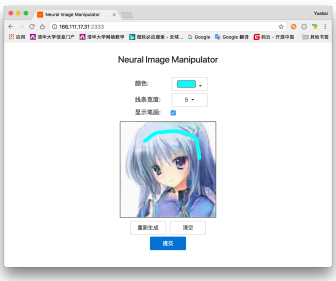

图 16: 点击提交后

首次打开网页，点击“开始按钮”，服务器会返回一张初始生成的图像。之后用户可以使用画笔工具，简单地勾勒出图片期望的变化方向，点击“提交”后，服务器就会返回在原图基础上，加上用户编辑条件后生成的图像；点击“重新生成”后，服务器会重新生成一张图片；点击“清空”能清空用户画的所有的笔画。此外，网页还支持更改画笔颜色、粗细，并能切换笔画的显示，以方便用户编辑。

该网页主要用 HTML 和 JavaScript 实现，并使用了 jQurey[5] 和 Bootstrap [6] 等库。图片编辑框使用了 HTML5 的 `<canvas>`，点击提交后，会将编辑框中的笔画图转成 Base64[7] 编码，使用 HTTP 请求发给服务器，之后服务器会返回生成图像的 Base64 编码，前端再将其作为图片编辑框的背景，显示给用户。

需要注意的是，用户每次点提交时，服务器是在该用户当前图像的基础上，再根据用户提供的笔画生成新图像，而且需要支持多用户同时编辑，即需要知道该用户当前的编辑状态。我们的做法是，服务器生成图像给前端时，会附带此时的 z 和 c 向量。前端将其保存下来，下次点提交时再发给服务器，服务器就会根据前端提供的 z 和 c 生成图像，以达到保存用户编辑状态的效果。

[5] http://jquery.com

[6] http://getbootstrap.com

[7] https://en.wikipedia.org/wiki/Base64

5.1.2 服务器端

由于训练模型使用的是 Python，所以我们选择了基于 Python 的 Django 框架搭建服务器端。

服务器端接收前端发来的笔画图，计算出 sketch 和 mask 矩阵，再根据前端发来的 z 和 c，放入之前训练好的网络中，就能得到生成的图像。然后将其和修改后的 z 和 c 返回给前端。

实际测试时，模型的运行效率非常快，生成一张图像的时间仅需十几毫秒，主要延时在于网络传输。因此，就算是多个用户同时访问网页进行编辑，服务器也完全可以承受住这样的负载。

6 结论与展望

神经图像编辑项目致力于通过简笔画生成或编辑一个图像，针对第一类基于神经网络的图像编辑方法中质量瓶颈问题进行了解决方法的探索，主要贡献如下：

1. 设计了表达能力更加强大的网络，改进了基于优化输入的编辑框架。
2. 在动漫人脸、鞋子、手提包的数据集上进行了实验，相较于两个基线提升了生成质量与编辑质量。
3. 提出了多阶段训练方法，将简笔画整合入网络的训练中。
4. 开发了基于 B/S 架构的网页应用，支持动漫人脸、鞋子、手提包的图像编辑，形成了产品雏形。

接下来我们主要的发展方向有三点，一是前端界面的改进，二是生成、编辑质量的进一步提高，三是编辑领域的扩展。

前端的改进有如下计划：

1. 网页端的改进。丰富网页端的功能，提供多种模型的选择，支持在手机上访问网页。工作量在一周左右，预计三月完成。
2. Android 端的应用。初步将网页端的功能进行移植即可，预计五月份完成。

技术改进的主要的步骤有：

1. 收集用户草图数据与增广数据集。预计三月完成。
2. 实现二阶段生成网络训练流程。预计四月完成。

3. 扩展输入的形态，支持第二类神经图像编辑框架。预计五月完成。

编辑领域扩展的优先级有如下计划：

1. 全身动漫人物的编辑。预计四月份完成。

2. 动漫场景的编辑。预计五月份完成。

3. 衣服、裤子的编辑。预计五月份完成。

我们会继续努力与挑战，让神经网络更加善解人意，让每个人都能用视觉与图像的方式展示自己的思想，让画师能够免于重复性的劳动，让神经网络架起简笔画与真实世界的桥梁。

参考资料

[1] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. Neural photo editing with introspective adversarial networks. 2016.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[3] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[4] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

[5] Y. Jin, J. Zhang, M. Li, Y. Tian, H. Zhu, and Z. Fang. Towards the automatic anime characters creation with generative adversarial networks. 2017.

[6] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *Computer Science*, 2014.

[7] N. Kodali, J. Abernethy, J. Hays, and Z. Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

[8] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016.

[9] M. Saito and Y. Matsui. Illustration2vec: a semantic vector representation of illustrations. In *SIGGRAPH Asia 2015 Technical Briefs*, pages 380–383, 2015.

[10] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016.

[11] A. Yu and K. Grauman. Fine-grained visual comparisons with local learning. In *Computer Vision and Pattern Recognition (CVPR)*, Jun 2014.

[12] J. Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, pages 597–613, 2016.