

HW3

赵延顺

2024 年 11 月 16 日

1 实验目标

- 1、使用条件 GAN 网络生成图像；
- 2、使用 GAN 网络实现人脸对齐。

2 条件 GAN 网络生成图像

2.1 Requirements

首先下载数据集，运行代码：

```
bash download_dataset.sh
```

于是得到了 *cityscapes* 数据集，并将其分成了训练集和测试集。

2.2 Training

由于网络的输入是图像的某种特征，而输出是原始的图像，这使得该问题很有难度。因此，针对该问题的处理，使用 U-net 结构作为生成器的主体。U-net 架构如下：

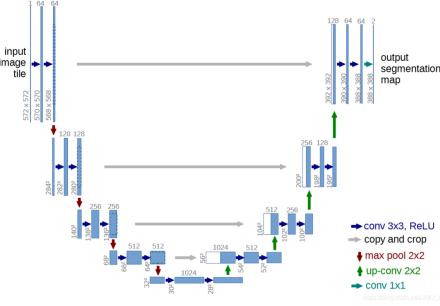


图 1: U-net

不过不同的是，U-net 的上采样使用转置卷积。而对于判别器，使用 U-net 的下采样部分，并将特征图拉直为向量接一个全连接网络输出。

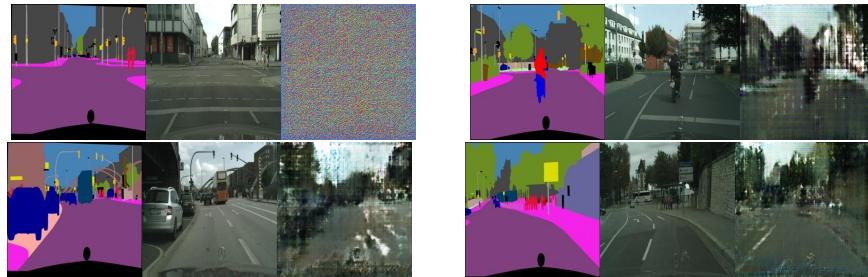
网络编写完成之后，运行代码：

Python train.py

当网络训练完成之后，即可用来评估模型。

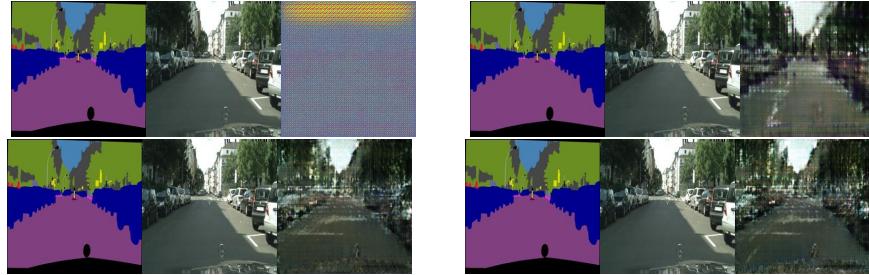
2.3 Results

在训练集上的效果为：



这 4 张图分别为训练 0 轮、400 轮、800 轮、1600 轮的效果。

在验证集上的效果为：



这 4 张图分别为训练 0 轮、400 轮、800 轮、1600 轮的效果。

2.4 Discussion

首先需要注意的是，编写的网络中，应尽量平衡 GAN 的生成器和判别器的参数量。生成器或判别器过强或过弱都会使得模型的能力下降。

另外，在处理的过程中，除了条件 GAN 本身的 loss 之外，还引入了 L1 损失以增强生成图片的质量。

从结果来看，效果一般，猜想可能的原因如下：

1、过拟合严重。cityscapes 数据集很小，而为了针对困难的任务，模型的参数量特别大。导致在训练后期，发现判别器的 loss 已经很小了，导致模型能力不能更进一步。

2、没有很好的平衡 L1 loss 和条件 GAN loss，导致模型的训练失衡。

3 使用 GAN 网络实现人脸对齐

直接运行 Github 上的 GragGAN 源项目。效果如下：



将人脸从大笑变为了微笑。