CartoonGAN及实验

汇报人：康志清　时间：2020.12.24

目录页

第一部分

风格迁移介绍

风格迁移介绍

style constrain

p *(content image)*

a *(style image)*

x *(output image)*

content constrain

风格迁移介绍

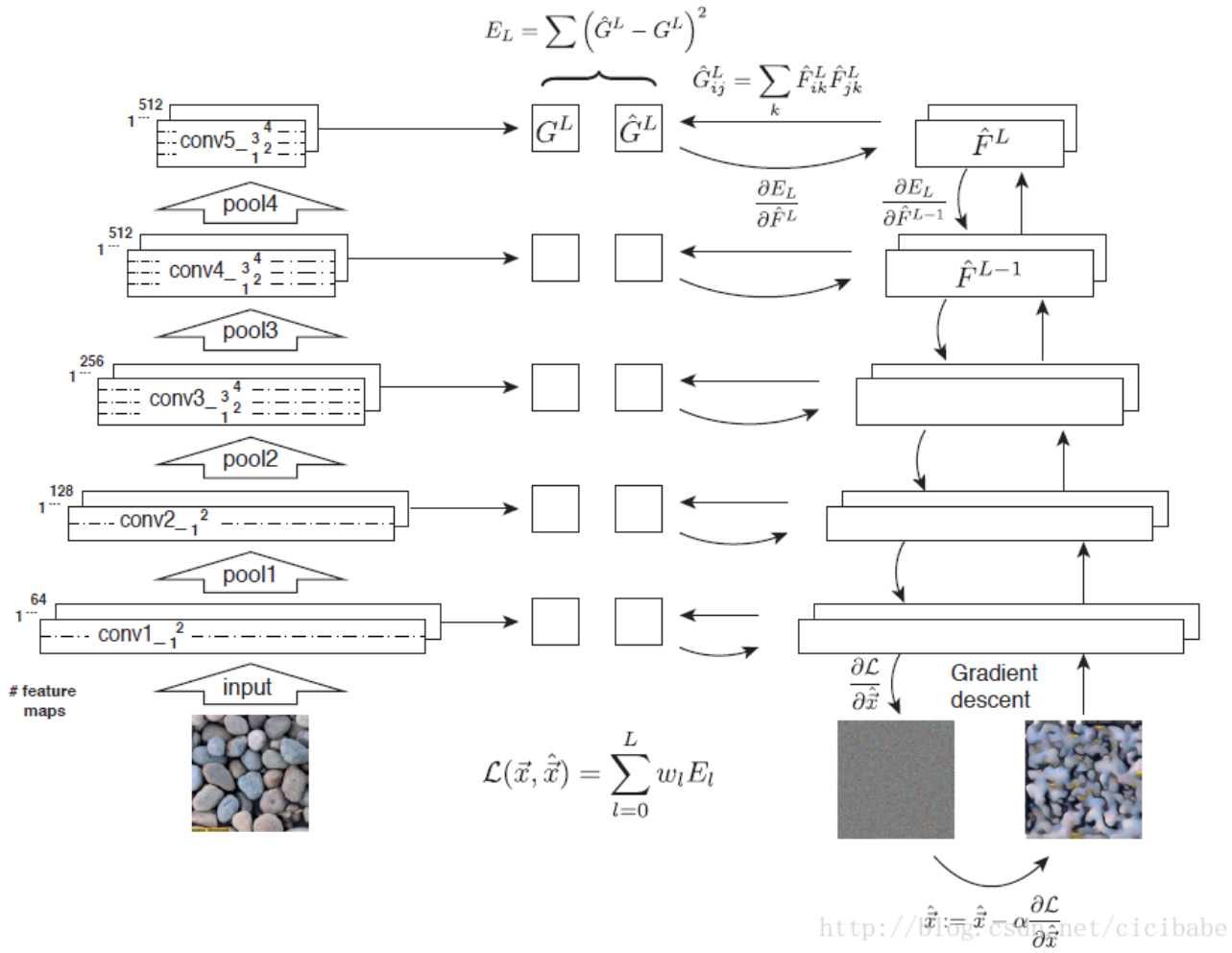
传统基于非参数的图像风格迁移：基于物理模型的绘制和纹理的合成。单一、提取底层特征，而非高层的抽象特征。

Galys等人2015年提出基于卷积神经网络的图像风格迁移。内容和风格分离

基于图像迭代：最大均值差异、基于马尔可夫随机场、基于深度图像类比

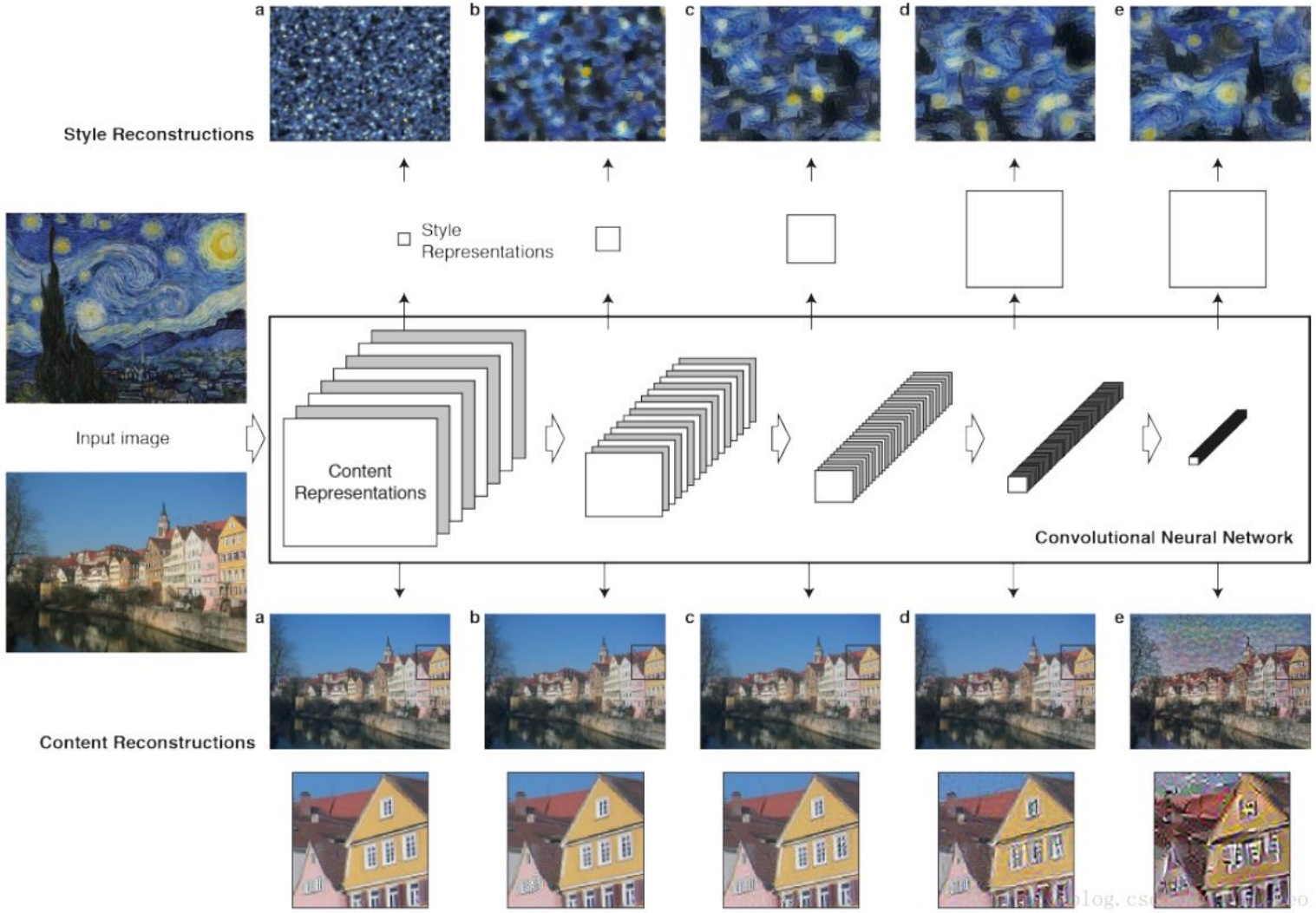
基于模型迭代：基于生成模型、基于图像重构解码器

Texture Synthesis Using Convolutional Neural Networks



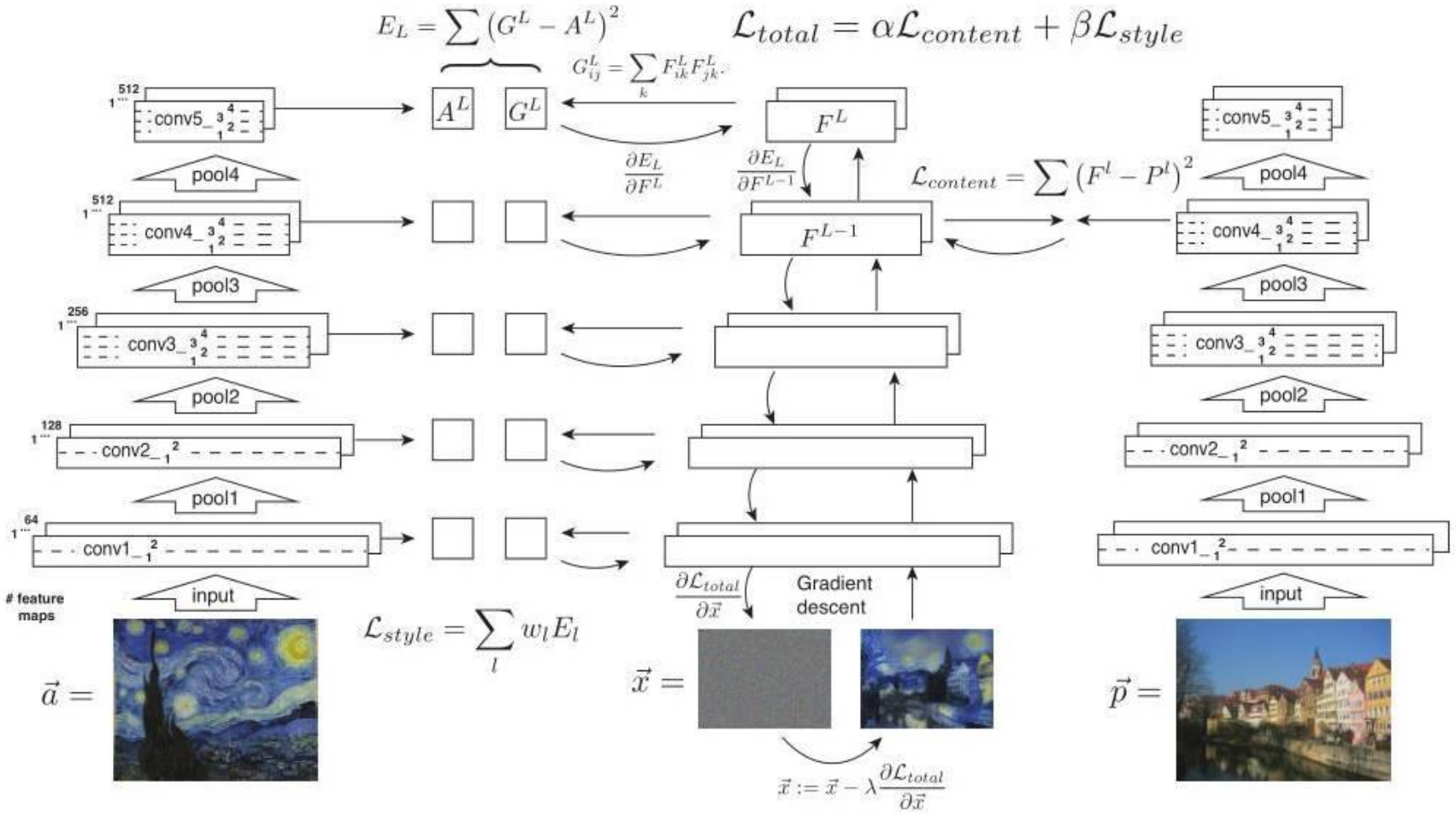
Gatys, Leon, Alexander S. Ecker, and Matthias Bethge. "Texture synthesis using convolutional neural networks." Advances in neural information processing systems 28 (2015): 262-270.

A Neural Algorithm of Artistic Style



Gatys L A, Ecker A S, Bethge M. A neural algorithm of artistic style[J]. arXiv preprint arXiv:1508.06576, 2015.

Image Style Transfer Using Convolutional Neural Networks(CVPR 2016)



Gatys L A, Ecker A S, Bethge M. Image style transfer using convolutional neural networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 2414-2423.

基于图像迭代

基于图像迭代

- 2015 Gatys L A, Ecker A S, Bethge M. A neural algorithm of artistic style
  （第一篇，用CNN完成风格迁移）
- 2016 Gatys L A, Bethge M, Hertzmann A, et al. Preserving color in neuralartistic style transfer
- 2016 Li Chuan, Wand M. Combining Markov random fields and convolutional neural networks for image synthesis
- 2017 Liao Jing, Yao Yuan, Yuan Lu, et al. Visual attribute transfer through deep image analogy
- .....

基于模型迭代

基于模型迭代

- 2016 Justin J，Alexandre A，Li Feifei． Perceptual losses for real-time style transfer and super-resolution
  （为某种特定风格生成模型-基于Gatys 的文章-感知损失=内容+风格+平滑）
- 2016 Ulyanov D, Vedaldi A, Lempitsky V. Instance normalization: the missing ingredient for fast stylization
  （使用实例归一化代替批量归一化）
- 2016 Wang Xin, Oxholm G, Zhang Da, et al. Multimodal transfer: a hierarchical deep convolutional neural network for fast artistic style transfer [J]
  （多尺度风格迁移颜色亮度生成高清图片）
- 2017 Huang Haozhi，Wang Hao，Luo Wenhan，et al．Real-time neural style transfer for videos
  （自适应实例归一化）
- 2018 Chen Y, Lai Y K, Liu Y J. Cartoongan: Generative adversarial networks for photo cartoonization
- .....

第二部分

CartoonGAN

CartoonGAN（2018年CVPR）

使用不成对的图片集训练GAN（单向Cycle）

基于GAN的架构中提出了两个简单而有效的损失函数

$$\mathcal{L}(G,D)=\mathcal{L}_{adv}(G,D)+\omega\mathcal{L}_{con}(G,D)$$

引入一个预训练过程，先用$\mathcal{L}_{\text{con}}(G,D)$单独一个损失来训练G，大概训练10个epoch，目的是使生成的图片初始后就能保证原真实场景的内容

CartoonGAN网络结构

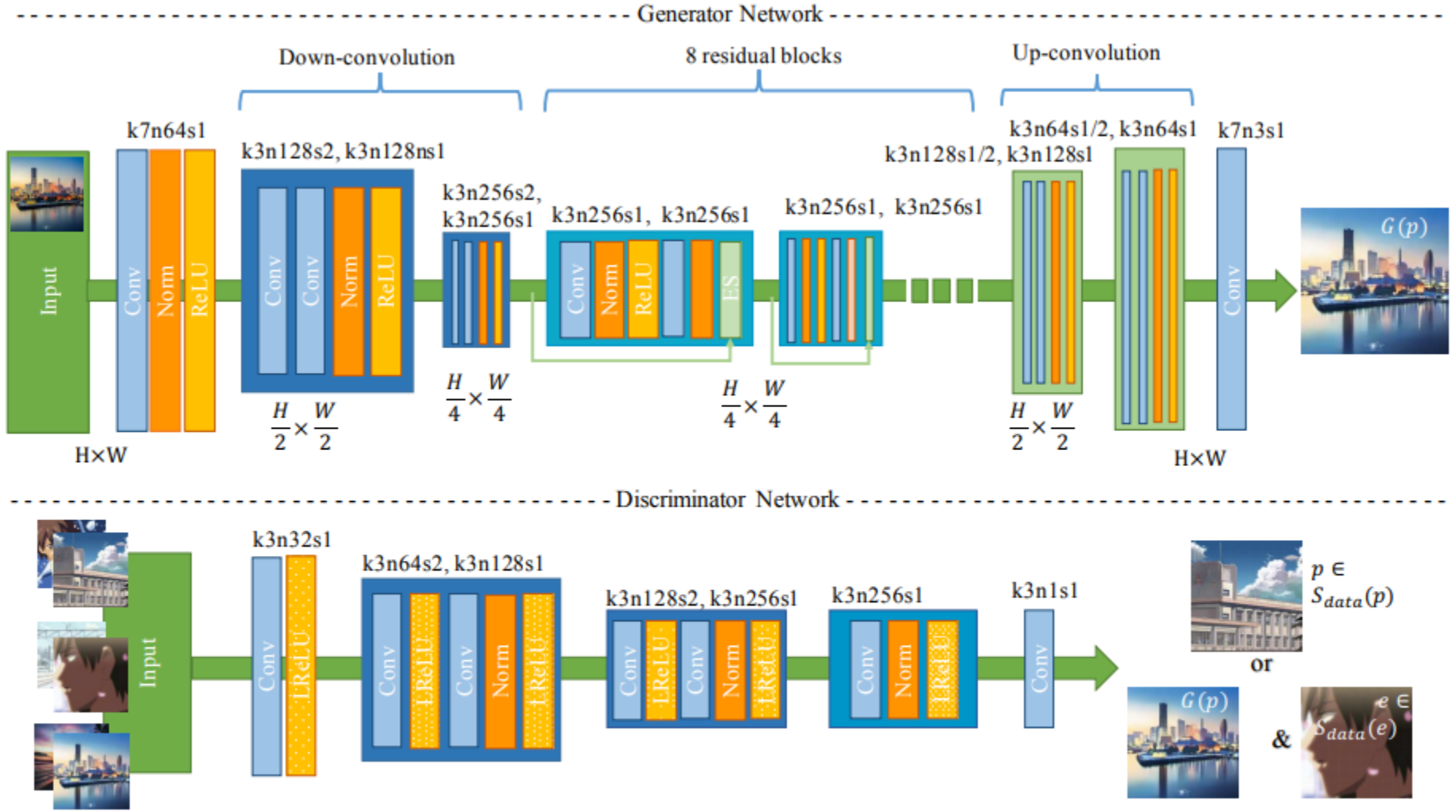


Figure 2. Architecture of the generator and discriminator networks in the proposed CartoonGAN, in which k is the kernel size, n is the number of feature maps and s is the stride in each convolutional layer, 'norm' indicates a normalization layer and 'ES' indicates elementwise sum.

损失函数

$$L(G,D) = L_{adv}(G,D) + \omega L_{con}(G,D)$$

$$\begin{aligned} L_{adv}(G,D) = & \mathbb{E}_{c_i \sim S_{data}(c)}[logD(c_i)] \\ & + \mathbb{E}_{e_j \sim S_{data}(e)}[1 - logD(e_j)] \\ & + \mathbb{E}_{p_k \sim S_{data}(p)}[\log(1 - D(G(p_k))] \end{aligned}$$

$$L_{con}(G,D) = \mathbb{E}_{p_i \sim S_{data}(p)}[||VGG_l(G(p_i) - VGG_l(p_i)||_1]$$

第三部分

实验

实验环境

- python 3.6.5
- pytorch 1.7.0
- opencv 4.4.0
- GPU：NVIDIA DGX工作站 4核Tesla单卡V100（32G显存）

数据集

1 目录结构

```
.
├── src_data
│   ├── test
│   └── train
└── tgt_data
    └── train
```

2 数据来源

1、其他论文提供的数据集

2、爬虫（自己编写的）

3、其他类型的数据集

https://kangzhiqing.com/post/datashow/

所有实验目录

OHAZEBIG_30_2000_16_results	2020-12-22 ...	root	root	drwxr-xr-x
OHAZE_30_2000_16_results	2020-12-21 ...	root	root	drwxr-xr-x
deHaze_20_2000_16_results	2020-12-20 ...	root	root	drwxr-xr-x
sword_20_2000_16_results	2020-12-20 ...	root	root	drwxr-xr-x
sword_20_400_16_results	2020-12-19 ...	root	root	drwxr-xr-x
Paprika_20_400_16_res8_res...	2020-12-19 ...	root	root	drwxr-xr-x
HayaoBig_20_200_32_0.0005...	2020-12-18 ...	root	root	drwxr-xr-x
Paprika_15_200_32_0.001_re...	2020-12-18 ...	root	root	drwxr-xr-x
HayaoBig_20_200_32_0.0005...	2020-12-17 ...	root	root	drwxr-xr-x
HayaoBig_15_200_32_0.0004...	2020-12-17 ...	root	root	drwxr-xr-x
Paprika_20_400_16_results	2020-12-16 ...	root	root	drwxr-xr-x
Paprika_1000_results	2020-12-15 ...	root	root	drwxr-xr-x
kzq_100_3000_results	2020-12-15 ...	root	root	drwxr-xr-x
kzq_1000_results	2020-12-14 ...	root	root	drwxr-xr-x
HaYao_10000_results	2020-12-13 ...	root	root	drwxr-xr-x
HaYao_1000_results	2020-12-13 ...	root	root	drwxr-xr-x
kiwato_10000_results	2020-12-13 ...	root	root	drwxr-xr-x
kiwato2_results	2020-12-12 ...	root	root	drwxr-xr-x
kiwato_results	2020-12-12 ...	root	root	drwxr-xr-x

Name	Size (KB)	Last m...	Owner	Group	Access
..					
394_epoch_Paprika_20_400_...	193	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	265	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	249	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	298	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	232	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	245	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	290	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	234	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	338	2020-12-16 ...	root	root	-rw-r--r--
394_epoch_Paprika_20_400_...	239	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	244	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	265	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	307	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	289	2020-12-16 ...	root	root	394_epoch_Paprika_20_400_16_test_
392_epoch_Paprika_20_400_...	239	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	193	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	277	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	337	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	187	2020-12-16 ...	root	root	-rw-r--r--
392_epoch_Paprika_20_400_...	199	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	265	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	232	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	245	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	250	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	264	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	293	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	296	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	254	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	248	2020-12-16 ...	root	root	-rw-r--r--
390_epoch_Paprika_20_400_...	202	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	193	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	308	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	249	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	239	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	277	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	279	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	268	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	345	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	205	2020-12-16 ...	root	root	-rw-r--r--
388_epoch_Paprika_20_400_...	183	2020-12-16 ...	root	root	-rw-r--r--
386_epoch_Paprika_20_400_...	193	2020-12-16 ...	root	root	-rw-r--r--
386_epoch_Paprika_20_400_...	268	2020-12-16 ...	root	root	-rw-r--r--
386_epoch_Paprika_20_400_...	296	2020-12-16 ...	root	root	-rw-r--r--

```
52G     ./pytorch-CartoonGAN
```

实验一（Hayao100 10000epoch）

数据集特别小，无论是原图还是风格图，都只有100张左右。训练很快。训练了10000次。1s/epoch

训练原图　　1000epoch　　3000epoch

实验一（Hayao100 10000epoch）

测试原图　　1000epoch　　3000epoch

实验二（Hayao 2000 200epoch）

数据集扩充，无论是原图还是风格图，都增加到2000张左右。但训练很慢。训练了200个epoch。300s/epoch

测试原图　　预训练结束　　100epoch　　200epoch

实验二（Hayao 2000 200epoch）

我们修改了预训练的次数10->20，并扩充了数据集

1、只用内容损失预训练20次的重建图比再用对抗训练200次后的结果似乎看上去好一些

2、训练太慢，迭代200次，动漫风格不够抽象

3、训练过程种出现棋盘效应：最后上采样过程中，选用反卷积进行上采样当卷积核大小不能被步长整除时，反卷积就会出现重叠问题，插零的时候，输出结果会出现一些数值效应，就像棋盘一样

实验二（Paprika 2000 1000epoch）

原图　　预训练之后　　250个epoch　　500个epoch　　论文给的模型

实验三（Paprika 2000 1000epoch 修改残差块的结构和数量）

两层恒等残差结构改成三层，将残差块的数目由8个改成10个

```
self.conv1 = nn.Conv2d(channel, channel, kernel, stride, padding)
self.conv1_norm = nn.InstanceNorm2d(channel)
self.conv2 = nn.Conv2d(channel, channel, kernel, stride, padding)
self.conv2_norm = nn.InstanceNorm2d(channel)
# 新加一层
# self.conv3 = nn.Conv2d(channel, channel, kernel, stride, padding)
# self.conv3_norm = nn.InstanceNorm2d(channel)
```

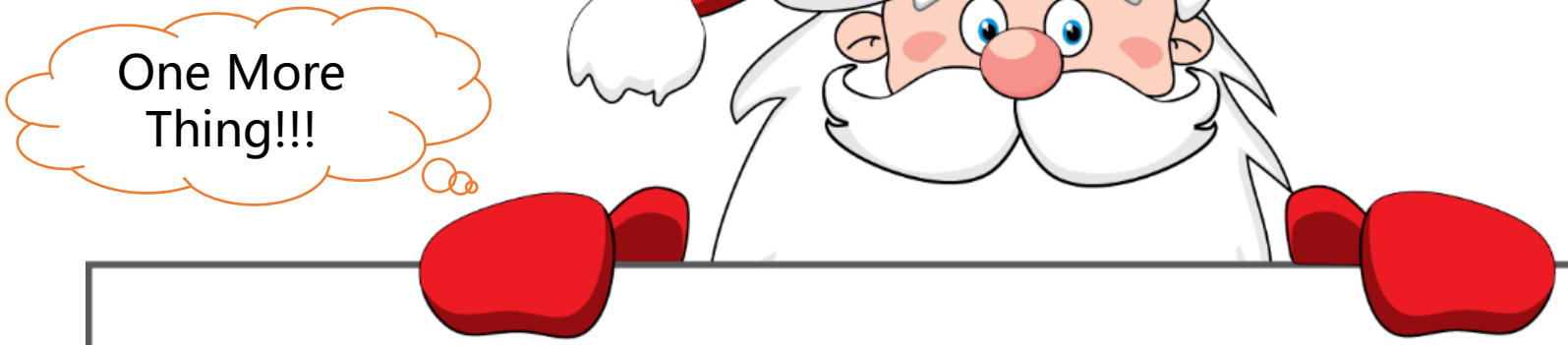
```
# nb = 8 ==> 10
def __init__(self, in_nc, out_nc, nf=32, nb=10):
    super(generator, self).__init__()
    self.input_nc = in_nc
    self.output_nc = out_nc
    self.nf = nf
    self.nb = nb
```

实验三（Paprika 2000 1000epoch 修改残差块的结构和数量）

上面为没修改，下面为修改过的

实验四（SwordArtOnLine 2000epoch）

原图　　预训练　　500　　1000　　2000



？ 我们是否能够把雾当作一种风格，
将其和内容分离开来，用于去雾领域。

Just Try it ➡

实验五（RESIDE 去雾 薄）

原图　　预训练　　500　　1000

实验六（OHAZE 去雾 浓）

原图　　预训练　　500　　1000

实验七（Reside 生成雾 薄）

原图　　预训练　　500　　1000

第四部分

感想和未来工作

感想

1、要想实验做得出来做得快做得好，还得看GPU的算力，只有迭代得快，我们才能在短周期内对整个网络进行调整，不然参数调整一下，训练需要很多天，才能知道参数调整对于整个网络的影响，实验的周期就会比较长。

2、对于网络的理解还是不太够，感觉调整网络结构比较困难和盲目，需要实验结果来验证想法，所以第一点真的很重要。调参真是门技术活！！！

3、每次实验一定要好好记录下，实验做多了容易混淆

4、数据集的选取和处理对于实验结果有很大的影响，要迁移的目标风格的图片尽可能风格一致

5、实践出真知，做了实验才能更好地理解论文所讲的内容

未来工作（基于风格迁移的图像去雾）

- 采用灰度图去雾验证网络去雾的效果
- 基于该模型，增加风格损失函数
- 采用CycleGAN
- .....


圣诞快乐
谢谢~