

人脸图像风格迁移

《现代图像分析》大作业

钱辰涞、赵典、马恒飞、李炳鑫

教师: 李洁

2021 年 5 月 14 日



目录

① 引入

- 人脸画像合成
- 传统的方式
- 利用深度学习的方法

② CNN 原理与风格迁移

- 风格迁移是……
- 卷积神经网络
- 风格迁移的内在原理

③ 编程实现

- 仿真环境
- 演示环节

④ 结果展示与比较

- CPU 上的实验结果
- 参数对生成图像的影响
- 在线系统的实验结果及其比较

⑤ 总结与展望



1

引入



1 引入

- 人脸画像合成
 - 传统的方式
 - 利用深度学习的方法

2 CNN 原理与风格迁移

3 编程实现

4 结果展示与比较

5 总结与展望

引入

CNN 原理与风格迁移

编程实现

结果展示与比较

总结与展望

人脸画像合成

人脸画像合成

传统的方式

1 引入

- 人脸画像合成
 - 传统的方式
 - 利用深度学习的方法

② CNN 原理与风格迁移

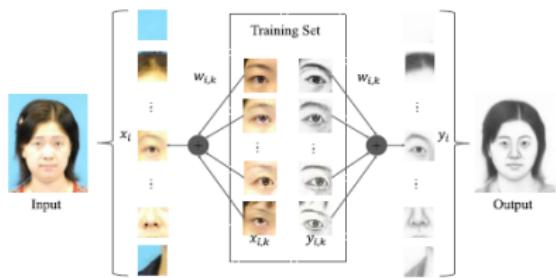
3 编程实现

4 结果展示与比较

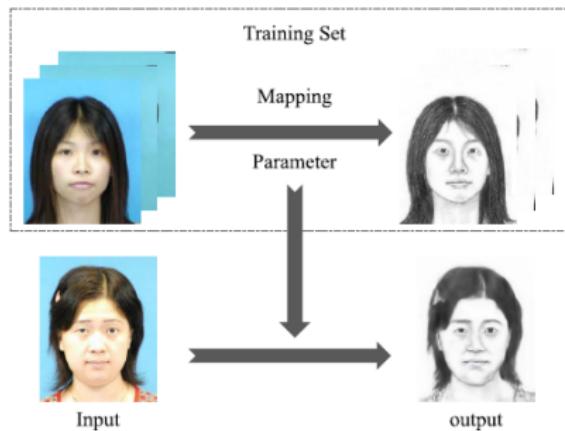
5 总结与展望

传统的方式

传统的方式



(a) Exemplar-based Model



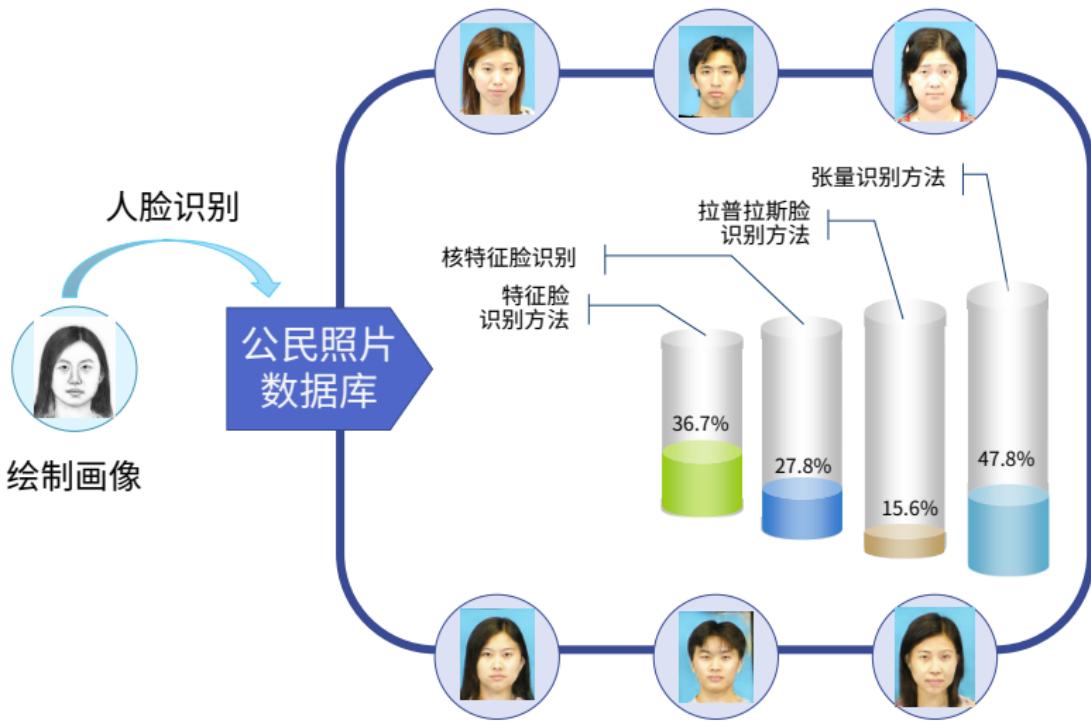
(b) Regression-based Model

图：传统的方法



传统的方式

传统的方式



① 引入

- 人脸画像合成
- 传统的方式
- 利用深度学习的方法

② CNN 原理与风格迁移

③ 编程实现

④ 结果展示与比较

⑤ 总结与展望

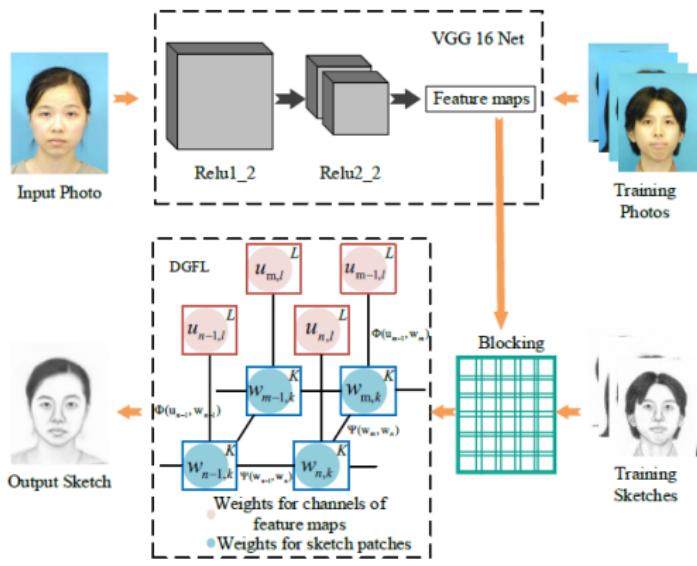


利用深度学习的方法

Deep graphical feature learning¹

Methods: exemplar-based

- general: neighbor selection and reconstruction weight representation
- novelty: combine dCNNs via DGFL framework



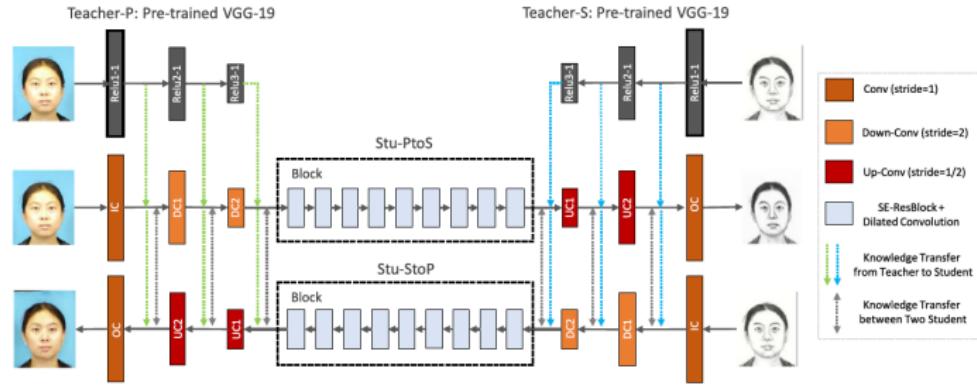
¹Zhu, M., Wang, N., Gao, X., & Li, J. (2017, August). Deep graphical feature learning for face sketch synthesis. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (pp. 3574-3580).

利用深度学习的方法

Knowledge Transfer²

Methods: Knowledge Transfer

- training a smaller and fast student network with the information learned from a larger and accurate teacher network.



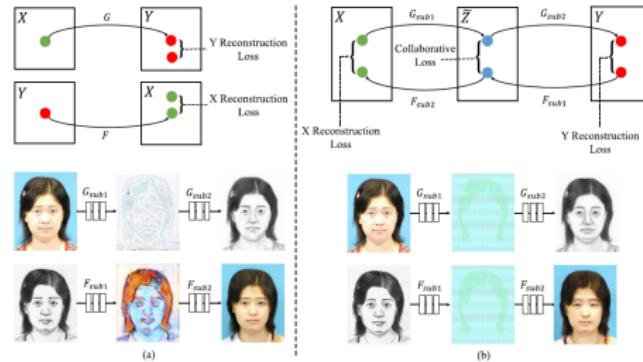
²Zhu, M., Wang, N., Gao, X., Li, J., & Li, Z. (2019, August). Face Photo-Sketch Synthesis via Knowledge Transfer. In *IJCAI* (pp. 1048-1054).

利用深度学习的方法

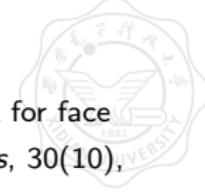
A Deep Collaborative Framework³

Methods: Regression-based

- collaborative loss that makes full use of two opposite mappings
- dCNNs, GANs, end-to-end



³Zhu, M., Li, J., Wang, N., & Gao, X. (2019). A deep collaborative framework for face photo-sketch synthesis. *IEEE transactions on neural networks and learning systems*, 30(10), 3096-3108.

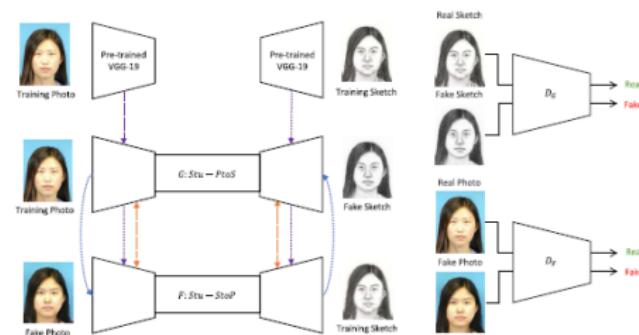


利用深度学习的方法

Knowledge Distillation⁴

Methods: Knowledge Distillation

- similar to KT
- propose a KD+ model that combines GANs with KD



⁴Zhu, M., Li, J., Wang, N., & Gao, X. (2020). Knowledge Distillation for Face Photo-Sketch Synthesis. *IEEE Transactions on Neural Networks and Learning Systems*.

2

CNN 原理与风格迁移



风格迁移是……

1 引入

2 CNN 原理与风格迁移

- 风格迁移是……
- 卷积神经网络
- 风格迁移的内在原理

3 编程实现

4 结果展示与比较

5 总结与展望



风格迁移是……

风格迁移是什么？

风格迁移是将一幅图片的内容和另一幅艺术图片的风格结合，生成一张艺术化的图片的过程。输入是一张 **内容图像 C** 和一张 **风格图像 S** ，输出是风格化的 **生成图像 G** 。⁵



⁵图片来自deeplearning.ai深度学习专项课程。

卷积神经网络

① 引入

② CNN 原理与风格迁移

- 风格迁移是……
 - 卷积神经网络
 - 风格迁移的内在原理

3 编程实现

4 结果展示与比较

5 总结与展望

卷积神经网络

CNN 是什么？

在《现代图像分析》第四章“图像增强”、第七章“图像分割”中，我们谈到了利用模板卷积的方式来进行图像平滑、锐化和边缘检测等功能。

0	1/4	0
1/4	0	1/4
0	1/4	0

■ 平滑模板

- 模板内系数全为正（表示求和、平均=>平滑）；
- 模板内系数之和为1：
 - 对常数图像 $f(m,n) \equiv c$, 处理前后不变；
 - 对一般图像, 处理前后平均亮度不变。

0	-1	0
-1	5	-1
0	-1	0

■ 锐化模板

- 模板内系数有正有负, 表示差分运算；
- 模板内系数之和为1：
 - 对常数图像 $f(m,n) \equiv c$, 处理前后不变；
 - 对一般图像, 处理前后平均亮度不变。

0	-1	0
-1	4	-1
0	-1	0

■ 边缘检测模板

- 模板内系数有正有负, 表示差分运算；
- 模板内系数之和为0：
 - 对常数图像 $f(m,n) \equiv c$, 处理后为0；
 - 对一般图像, 处理后为边缘点。



卷积神经网络

CNN 是什么？

为了让能检测各种方向的线条，利用图像卷积实现更多的效果，我们可以通过深度学习来训练这个卷积模板，当然他不一定是 3×3 的，可以是 5×5 甚至更高维度的。一般我们选择奇数维的卷积模板。

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

显然对于上面的卷积模板有 9 个参数需要学习。



卷积神经网络

CNN 的主要组件

对于一个简单的 CNN 网络主要有以下几种类型的中间层：

- **CONV**: Convolution
- **POOL**: Pooling
- **FC**: Fully connected

卷积层的基本参数

$filter(f)$ 模板的尺寸

$Padding(p)$ 主要分“Valid Convolution”和“Same Convolution”

$Strided(s)$ 卷积模板两次移动之间的跨越方格数

$$(n \times n) * (f \times f) = \left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor$$

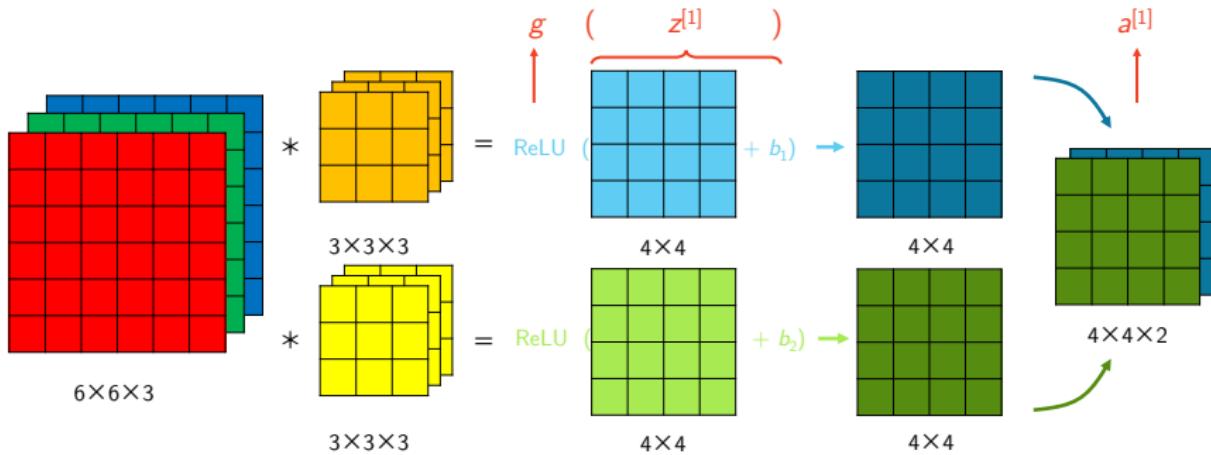
卷积神经网络

卷积层 (CONV) 的基本参数



卷积神经网络

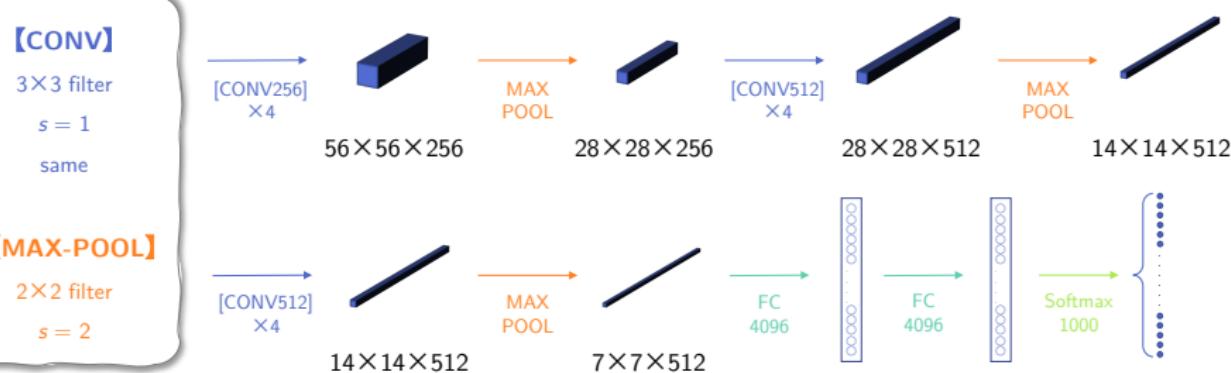
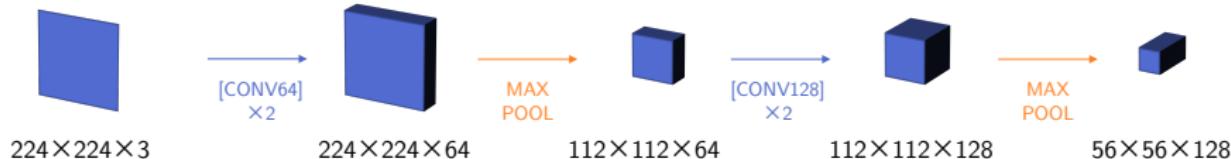
CNN 如何运作?



对于利用 $3 \times 3 \times 3$ 的卷积模板作卷积需要 $(3 \times 3 \times 3 + bias) = 28$ 个参数，如果产生的通道数为 2，则需要 2 个这样的模板，即需要 $28 \times 2 = 56$ 个参数。往往需要的通道数会更多。

卷积神经网络

一个经典的 CNN 网络——VGG-19⁶



⁶Simonyan, K. , and A. Zisserman . "Very Deep Convolutional Networks for Large-Scale Image Recognition." *Computer Science* (2014).

卷积神经网络

CNN 在学什么？

显示整个验证数据中随机的特征图子集的前 9 个激活项，使用去卷积网络方法投影到像素空间。⁷



(a) Layer 1



(b) Layer 2



(c) Layer 3



(d) Layer 4



(e) Layer 5

第 1 层主要检测边缘，第 2 层是一些形状、纹理。第 3 层开始检测更复杂的图案。网络越深，同一激活项内部的多样性越多。

⁷MD Zeiler, and R. Fergus . "Visualizing and Understanding Convolutional Neural Networks." *European Conference on Computer Vision* (2013).



风格迁移的内在原理

① 引入

② CNN 原理与风格迁移

- 风格迁移是……
- 卷积神经网络
- 风格迁移的内在原理

③ 编程实现

④ 结果展示与比较

⑤ 总结与展望

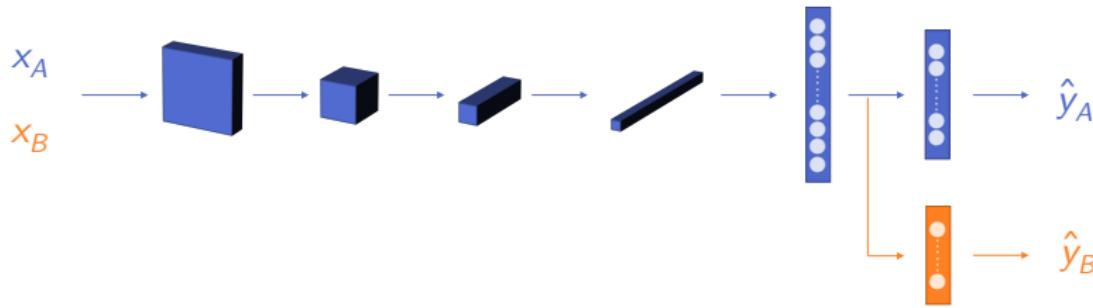


风格迁移的内在原理

迁移学习

迁移学习 (Transfer Learning)

利用在不同 Task 上训练的网络，将其应用于新的 Task。



- 两个任务有着相同的输入；
- 相比于 B，A 有更多的数据；
- 迁移学习的一大特征就是共享权重。



风格迁移的内在原理

风格迁移中的代价函数⁸

风格迁移中的代价函数 (cost function)

$$J(G) = \alpha J_{\text{content}}(C, G) + \beta J_{\text{style}}(S, G) \quad (1)$$

如何利用代价函数找到生成图像 G ?

- ① 随机生成生成图像 G
- ② 使用梯度下降，最小化 $J(G)$

$$G \leftarrow G - \frac{\partial J(G)}{\partial G}$$



⁸Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "A neural algorithm of artistic style." *arXiv preprint arXiv:1508.06576* (2015).

风格迁移的内在原理

代价函数之一——内容代价函数

- 选择在第 ℓ 层来计算内容代价函数 (Content cost);
- 使用一个 pre-trained 的 ConvNet，在这里选择的是 VGG-19;
- 设 $a^{[\ell]}(C)$, $a^{[\ell]}(G)$ 表示第 ℓ 层 内容图像 C 和 生成图像 G 的激活项。当 $a^{[\ell]}(C)$, $a^{[\ell]}(G)$ 相似时，认为 内容图像 C 和 生成图像 G 的内容是相似的。

通过刻画激活项之间的相似度，来得到内容代价函数。

内容代价函数 (Content cost)

$$J_{\text{content}}(C, G) = \frac{1}{4 \times n_H^{[\ell]} \times n_W^{[\ell]} \times n_C^{[\ell]}} \left\| a^{[\ell]}(C) - a^{[\ell]}(G) \right\|^2 \quad (2)$$

风格迁移的内在原理

代价函数之二——风格代价函数

将风格 (style) 定义为激活项在不同通道之间的相关性 (correlation)。

相关性 (correlation)

更高层次的特征是否倾向于在图像的一部分中同时出现。而这种高层次可以体现为图像的各个通道。

比如，橙色和垂直线条是否会同时出现，同时出现或是不同时出现的频率又是怎样的。

于是，用通道与通道之间的关系来刻画相关性，进而刻画风格代价函数。



风格迁移的内在原理

代价函数之二——风格代价函数

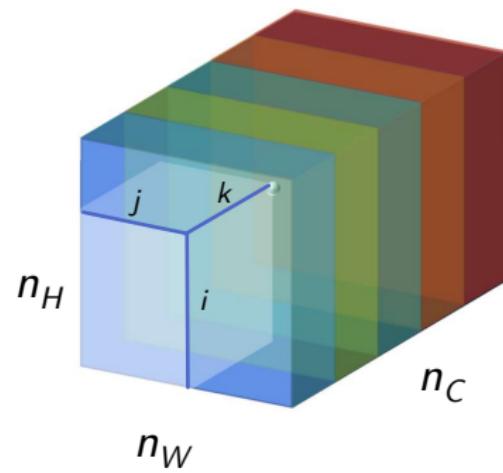
设在 (i, j, k) 点处的激活项为 $a_{i,j,k}^{[\ell]}$ ，我们用风格矩阵 (style matrix) 来刻画通道与通道之间的相关性。

风格矩阵 (Style matrix 或 Gram matrix)

- $G^{[\ell]}$ 是一个 $n_c \times n_c$ 维的方阵。

$$G_{kk'}^{[\ell](S)} = \sum_i^{n_H^{[\ell]}} \sum_j^{n_W^{[\ell]}} a_{ijk}^{[\ell](S)} a_{ijk'}^{[\ell](S)}$$

$$G_{kk'}^{[\ell](G)} = \sum_i^{n_H^{[\ell]}} \sum_j^{n_W^{[\ell]}} a_{ijk}^{[\ell](G)} a_{ijk'}^{[\ell](G)}$$

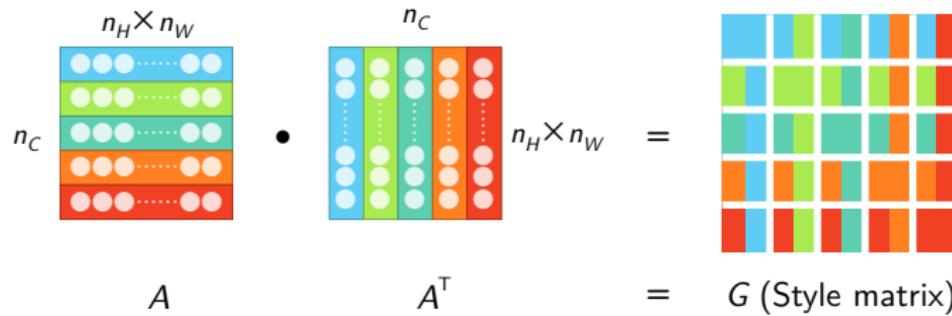


如果两通道之间越相关，那么计算出风格矩阵中的值越大。



风格迁移的内在原理

代价函数之二——风格代价函数



风格代价函数 (Style cost)

$$J_{\text{style}}^{[\ell]}(S, G) = \frac{1}{(2n_H^{[\ell]} \times n_W^{[\ell]} \times n_C^{[\ell]})^2} \|G^{[\ell](S)} - G^{[\ell](G)}\|^2 \quad (3)$$

$$J_{\text{style}}(S, G) = \sum_{\ell} \lambda^{[\ell]} J_{\text{style}}^{[\ell]}(S, G) \quad (4)$$

3

编程实现



仿真环境

① 引入

② CNN 原理与风格迁移

③ 编程实现

- 仿真环境
- 演示环节

④ 结果展示与比较

⑤ 总结与展望



仿真环境

仿真环境

- OS 环境：Windows 10
- 处理器类型：x86_64
- CPU 型号：i7-8550U
- Python 版本：3.8.5
- 深度学习框架：tensorflow
- tensorflow 版本：2.4.1



演示环节

① 引入

② CNN 原理与风格迁移

③ 编程实现

- 仿真环境
- 演示环节

④ 结果展示与比较

⑤ 总结与展望



演示环节

演示环节——风格迁移所需函数

① 内容代价函数

② 风格代价函数

- 风格矩阵 (style matrix)
- 计算单层风格代价函数
- 设置权重，并求解总体风格代价函数

③ 风格迁移总代价函数



演示环节

演示环境——实现风格迁移

- ① 加载 内容图像 C 、 风格图像 S
- ② 初始化 生成图像 G
- ③ 加载 Pre-trained VGG-19 模型
- ④ 计算风格迁移总代价函数
 - 计算内容代价函数
 - 计算风格代价函数
- ⑤ 定义 Optimizer 和学习率，进行优化



4

结果展示与比较



CPU 上的实验结果

① 引入

② CNN 原理与风格迁移

③ 编程实现

④ 结果展示与比较

- CPU 上的实验结果
- 参数对生成图像的影响
- 在线系统的实验结果及其比较

⑤ 总结与展望



CPU 上的实验结果

人脸图像的画像风格

以学习率为 0.001，跑 21001 epochs。从早上 8 点跑到晚上 8 点 40。



图: Content



图: Style



CPU 上的实验结果

人脸图像的梵高风格

以学习率为 0.001，跑 10001 epochs。从下午 3 点半跑到晚上 9 点半。

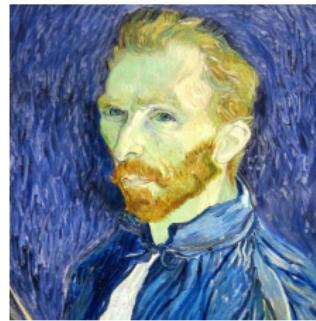


图: Content

图: Style



参数对生成图像的影响

① 引入

② CNN 原理与风格迁移

③ 编程实现

④ 结果展示与比较

- CPU 上的实验结果
- 参数对生成图像的影响
- 在线系统的实验结果及其比较

⑤ 总结与展望

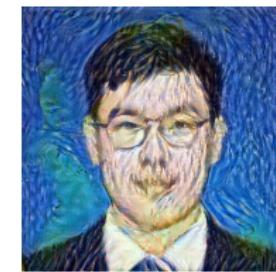


参数对生成图像的影响

调整风格代价函数各层权重对生成图像 G 的影响

以学习率为 0.03，内容权重为 10，风格权重 40，各跑 1001 epochs。

	深层权重高	均衡	浅层权重高
block1_conv1	0.0	0.2	0.5
block2_conv1	0.0	0.2	0.4
block3_conv1	0.1	0.2	0.1
block4_conv1	0.4	0.2	0.0
block5_conv1	0.5	0.2	0.0



生成图像 G



参数对生成图像的影响

改变内容与风格的相对权重对生成图像 G 的影响

以学习率为 0.03, 风格代价函数各层权重均为 0.2, 各跑 1001 epochs。

$$J(G) = \alpha J_{\text{content}}(C, G) + \beta J_{\text{style}}(S, G)$$

	内容权重高	风格权重高
内容权重 α	40	10
风格权重 β	10	40




生成图像 G



在线系统的实验结果及其比较

① 引入

② CNN 原理与风格迁移

③ 编程实现

④ 结果展示与比较

- CPU 上的实验结果
- 参数对生成图像的影响
- 在线系统的实验结果及其比较

⑤ 总结与展望



在线系统的实验结果及其比较

应用网页端进行人脸图片风格迁移

由于利用 CPU
跑这个代码实
在很慢，所以 我
们选择使用类似
相同算法的网站
(deeprift.io)。一
般每张图片处理
10-20 分钟左右。



在线系统的实验结果及其比较

三种人脸图像转画像

相信大家看到这里看到我的脸快要吐了，我也没有必要再放内容图像和风格图像了。所以下面我们直接上结果。大家做好心理准备，有点吓人。



(a) CPU, 21000 epochs



(b) deepart.io



(c) sketchai.cn

5

总结与展望



风格迁移

风格迁移中的代价函数 (cost function)

$$J(G) = \alpha J_{\text{content}}(C, G) + \beta J_{\text{style}}(S, G) \quad (5)$$

内容代价函数 (Content cost)

$$J_{\text{content}}(C, G) = \frac{1}{4 \times n_H^{[\ell]} \times n_W^{[\ell]} \times n_C^{[\ell]}} \left\| a^{[\ell]}(C) - a^{[\ell]}(G) \right\|^2 \quad (6)$$

风格代价函数 (Style cost)

$$J_{\text{style}}(S, G) = \sum_{\ell} \lambda^{[\ell]} \frac{1}{\left(2n_H^{[\ell]} \times n_W^{[\ell]} \times n_C^{[\ell]}\right)^2} \left\| G^{[\ell]}(S) - G^{[\ell]}(G) \right\|^2 \quad (7)$$

风格迁移

- 风格迁移指给定一个 **内容图像 C** 和一个 **风格图像 S** 可以生成一个艺术图像 G 。
- 它使用基于 Pre-trained 的 ConvNet 的隐藏层激活项。
- 内容代价函数 (Content cost) 是用一个隐藏层的激活值计算的。
- 一个层的风格代价函数 (Style cost) 使用该层激活项的风格矩阵 (Gram Matrix) 计算。总体风格代价函数是通过几层风格代价函数加权获得的。
- 与开始介绍的迁移学习不同的是，通过风格迁移总代价函数优化算法更新的是像素值而不是神经网络的参数，其输出是合成新图像。



迁移学习

迁移学习 (transfer learning) 不光在图像领域有所发展，同时也在其他领域有很应用。例如在通信领域，迁移学习和 MIMO 相结合，目前有和半监督学习的 AMC(Automatic Modulation Classifier)⁹，对频分多址的大规模 MIMO 系统进行下行链路信道预测¹⁰，对低分辨率毫米波 MIMO 的特定场景信道估计¹¹等。

⁹Wang, Yu, et al. "Transfer learning for semi-supervised automatic modulation classification in ZF-MIMO systems." *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 10.2 (2020): 231-239.

¹⁰Yang, Yuwen, et al. "Deep Transfer Learning-Based Downlink Channel Prediction for FDD Massive MIMO Systems." *IEEE Transactions on Communications* 68.12 (2020): 7485-7497.

¹¹Alves, Wesin, et al. "Deep Transfer Learning for Site-Specific Channel Estimation in Low-Resolution mmWave MIMO." *IEEE Wireless Communications Letters* (2021).

Q&A

电子工程学院 电子信息工程

钱辰涞 (18020100016)

Blog: <https://levitate-qian.github.io/>

