神经风格迁移

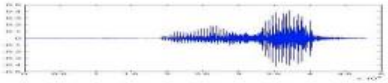
助教：顾 峥

指导教师：高 阳

Ml2020_st@163.com

什么是神经风格迁移

- Speech Recognition

$f($  $)=$ “How are you”

- Image Recognition

$f($  $)=$ “Cat”

机器学习就是找函数
——[Hung-yi Lee ML2020]

- Playing Go

$f($  $)=$ “5-5” (next move)

- Dialogue System

$f($ “How are you?” $)=$ “I am fine.”

(what the user said) (system response)

- Neural Style Transfer

F( , ) = 

神经风格迁移主要方法

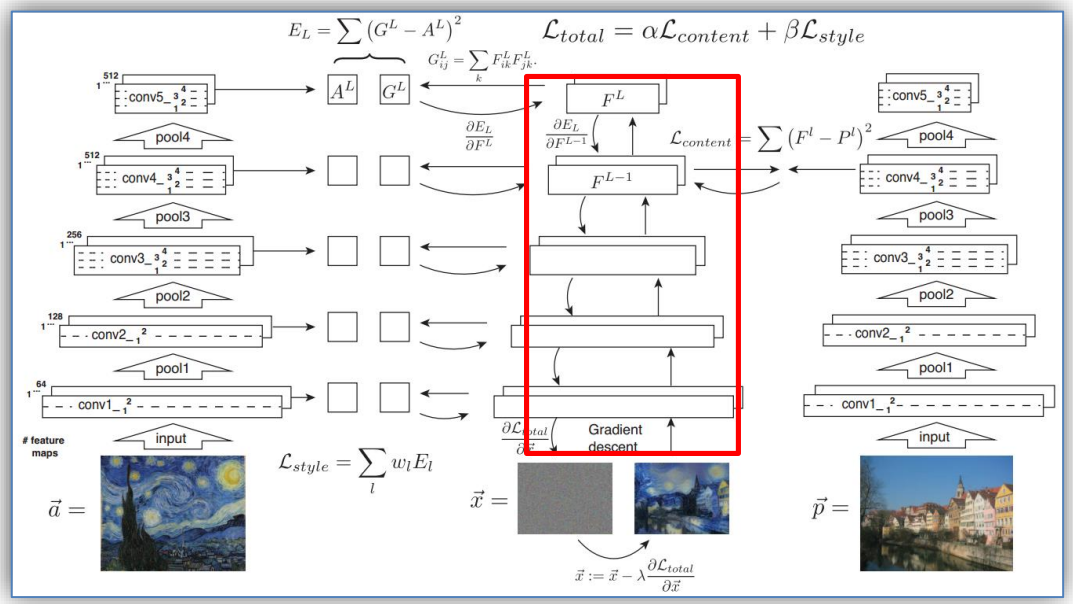
- 基于优化的方法
- 基于统计的方法
- 基于匹配的方法

基于优化的方法

- 风格、内容的特征表示

神经网络的多层级特性、Gram矩阵

代表方法：Neural Style



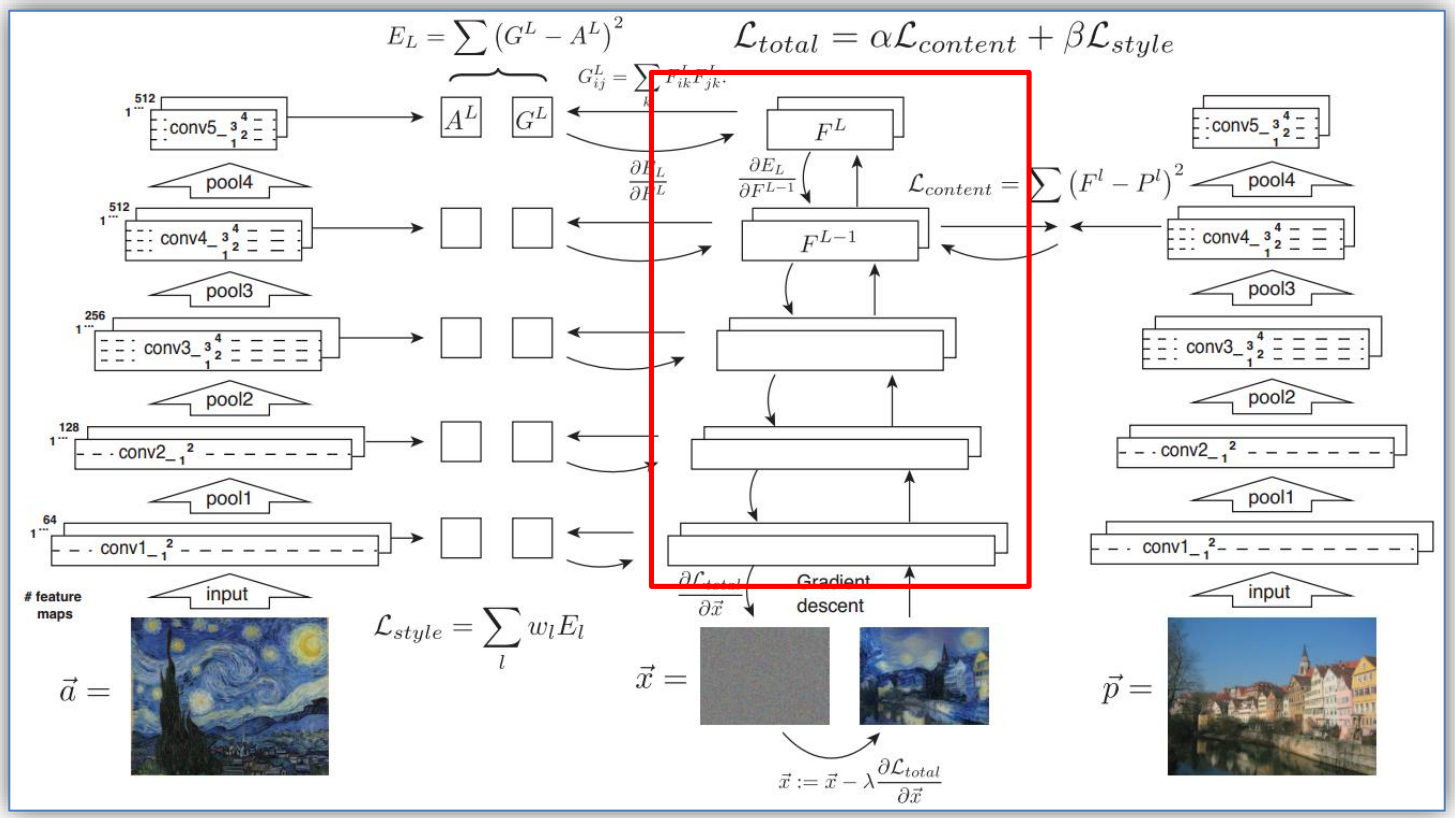
优化目标：

1. 输出图与内容图的内容差异
2. 输出图与风格图的风格差异

[Gatys L A , et al. 2016]

基于优化的方法

- 代表方法：Neural Style



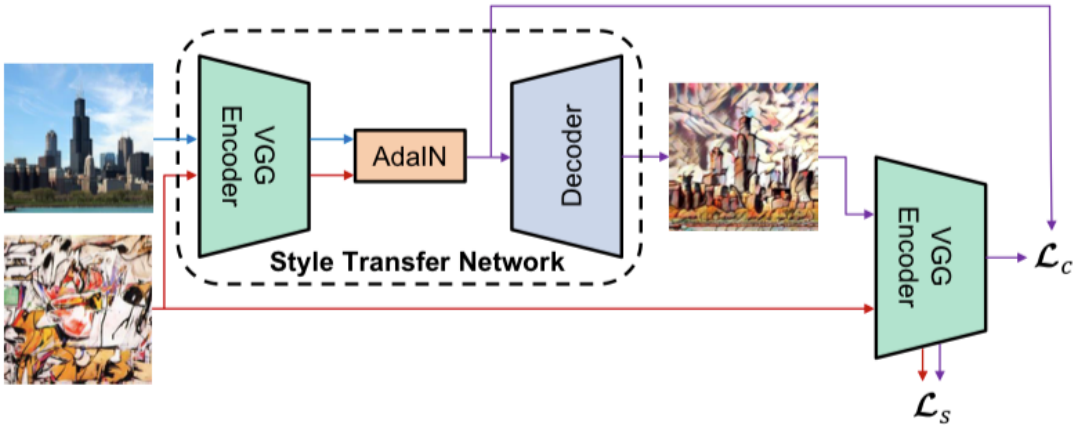
[Gatys L A , et al. 2016]

基于统计的方法

- 图像风格表示：二阶统计信息

图像风格表示为特征空间中为两个不同的分布

代表方法：AdaIN/ WCT



$$\mathrm{AdaIN}(x, y) = \sigma(y)\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \mu(y)$$

[Huang X , et al. 2016]

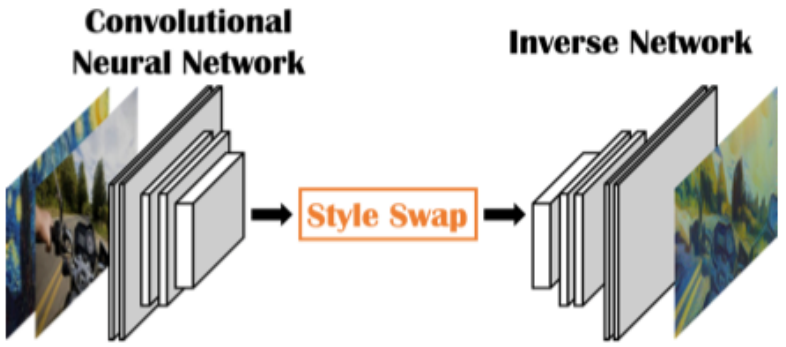
基于匹配的方法

- 能否复用已有风格图像上结构类似的区域？

图像块的相似性度量

代表方法： Style Swap

对内容图的每个小块，在风格图上寻找与之结构最相近的区域，替换到内容图上。



$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} ||\Phi(f(H_i;\theta)) - H_i||_F^2 + \lambda \ell_{TV}\left(f(H_i;\theta)\right)$$

[Chen T Q , et al. 2016]

神经风格迁移实验

- 相关文献

- Neural Style Transfer: A Review
- Image Style Transfer Using Convolutional Neural Networks
- Universal Style Transfer via Feature Transforms
- Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization

- 深度学习相关课程

- http://cs231n.stanford.edu/
- http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML20.html

神经风格迁移实验

实验任务

- 自行实现上述三种算法中的至少一种，编程时可以参考相关开源代码以加深对算法的理解，但不得抄袭；
- Bonus：分析现有算法的问题和不足，提出你的改进方法并验证。

数据采集

- 自行收集训练数据（可以是论文提供的数据集，也可以自行搜集），用实现的算法进行训练，并测试风格迁移结果。

神经风格迁移实验

- 实验报告（包括但不限于）

- 对任务的定义；
- 对所使用方法的描述和理解；
- 所使用的数据（数据来源、数据特点及难点）；
- 实验结果（多个角度、多种评价指标）；
- 实验分析，请自行对所选方法进行多角度的分析（例如：为什么选用这种方法，相比其他两种方法的优势在哪，仍存在什么问题，可以如何进行改进等）。

RLCard

助 教: 葛振兴

指导老师：高阳

dz1933006@smail.nju.edu.cn

实验内容

- RLCard 是一个用于牌类游戏强化学习研究的开源工具包，其接口简单易用，支持多种牌类环境。
- RLCard 的目标是在强化学习与非完美信息博弈之间搭建桥梁，推动强化学习研究在多智能体、高维状态和动作空间以及稀疏奖励领域的进步。融合了中西方最流行的几种牌类游戏（包括斗地主、麻将、21 点、德州扑克、UNO 等）。
- 本次实验涉及DQN，NFSP，CFR三种在game中较为常见的方法。

环境安装

- 安装python3.5+与pip

- 安装RLCard
  - git clone https://github.com/datamllab/rlcard.git
  - cd rlcard
  - pip install -e .

  若要使用Pytorch实现的版本：
  - pip install -e .[with_torch]

环境简介

- **环境的简单使用方法**
  - 一个简易调用方法如下

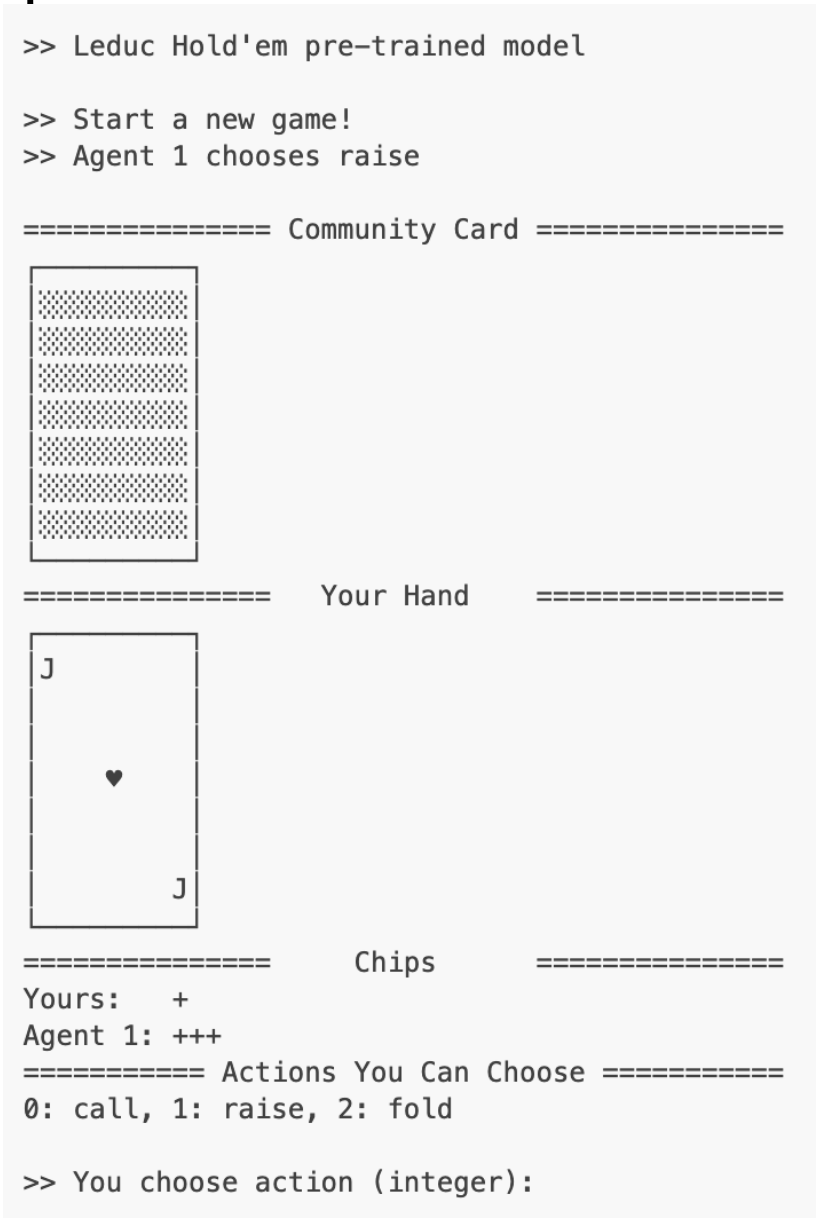
```
import rlcard
from rlcard.agents.random_agent import RandomAgent

env = rlcard.make('blackjack')
env.set_agents([RandomAgent()])

trajectories, payoffs = env.run()
```

环境简介

- **Demo**
  - RLCard支持人机交互

更多简介可参考

- https://github.com/datamllab/rlcard
- http://rlcard.org/



实验要求

- **仔细阅读一下论文内容：**
  - https://daiwk.github.io/assets/dqn.pdf
  - https://arxiv.org/pdf/1603.01121.pdf
  - http://papers.nips.cc/paper/3306-regret-minimization-in-games-with-incomplete-information.pdf
- **基本要求：**自己实现复现工具箱中的3种基本方法,可以参考框架中的代码，但不得抄袭。推荐在agent/目录下包装成agent
  - 深度 Q 网络（DQN；Silver et al. 2016）
  - 神经虚拟自我博弈（NFSP；Heinrich and Silver 2016）
  - 反事实遗憾最小化（CFR；Zinkevich et al. 2008）
- **扩展要求：**
  - 选择3种基线方法中的某个主要方向或多个方向交叉进行拓展研究，性能超过基线方法。

实验提交

- **提交完成上述算法设计和代码，报告内容包括但不限于：**
  - DQN，NFSP，CFR三种算法的原理
  - 结合代码描述
    - 每个算法的具体实现方式
    - 改进的算法的原理与方法
    - 一些你认为的重要变量/参数的意义
  - 展示实现的算法效果（包含但不限于胜率，收敛速度等等）

评分标准

- 本实验旨在于希望同学们学习并掌握基本算法原理的同时进行代码实现，因此评分标准设计如下：
- **基础得分**
  - 完成作业要求，提交代码 （50%）
    - 基础要求 （30%）
    - 扩展要求 （20%）
  - 按要求完成报告 （50%）
    - 能够清楚的解释实验涉及的算法 （20%）
    - 展示核心代码并加以讲解设计思路与重要内容 （20%）；
    - 展示实验结果 （10%）

星际争霸

陈佳瑞

chenjiarui0096@163.com

多智能体强化学习


LAC 124 BOS 123
OT 58.5 17
TNT


神速之靴
唯一被动：【+30移动速度】
517
抗魔披风
+90法抗
+105
回复生命与法力
60
恢复
25

多智能体强化学习的难点

1. 部分可观察

$$Q(s,a) \Rightarrow Q(o,a)$$

- 无法根据全局状态进行决策，动作可能是局部最优，真正能找到最优解的$Q(\boldsymbol{s},\boldsymbol{a})$未知

$$\boldsymbol{s}=[o_1,o_2,\dots,o_n] \quad \boldsymbol{a}=[a_1,a_2,\dots,a_n]$$

2. 不稳定性

$$r=R(\boldsymbol{s},\boldsymbol{a})$$

- 由于无法知道其他智能体的o，当前智能体对于同样的(o,a)，其他智能体的o可能不一样，因此实际上的$\boldsymbol{s}$不一样，从而导致r可能不同

- 其他的智能体也在学习，策略在不断变化，当前智能体对于同样的(o,a)，即使$\boldsymbol{s}$相同，得到的r也可能不同，因为联合动作$\boldsymbol{a}$不同

解决方法

1. 通信

所有智能体在进行决策之前，先向其他智能体发送信息，并且接收其他智能体的信息（要发送的信息的内容很关键），然后再做出相应的决策。

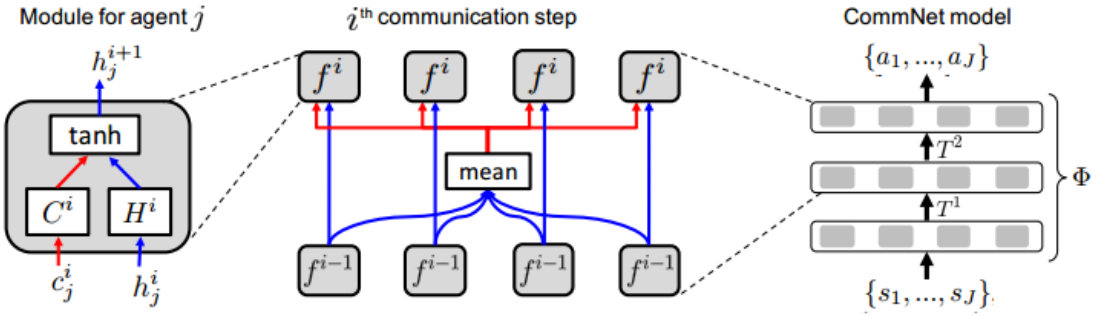
2. 集中式训练、分散式执行

每个智能体在选择动作的时候只根据自己的$Q(o,a)$选择动作，但是为了保证智能体之间能够合作，就需要让他们一起进行学习，朝着同一个目标更新自己的$Q(o,a)$。

比如一个小组出去执行任务，执行的时候每个人自己进行决策，但是执行结束后大家聚在一起学习，一起讨论什么时候该做出什么样的决策，这样下一次执行任务的收益就会更高。

具体算法

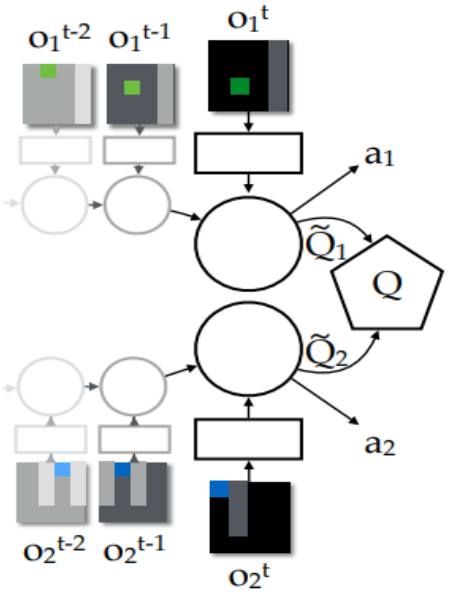
1. 通信——CommNet



2. 集中式训练、分散式执行——VDN

$$Q(\boldsymbol{s}, \boldsymbol{a}) = \sum_{i=1}^{n} Q_i(o^i, a^i)$$

$$L(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - Q(\boldsymbol{s}, \boldsymbol{a})\right)^2$$



效果展示

训练环境与要求

1. 实验要求

- 在星际争霸微观平台SMAC上训练智能体合作对战；
- 在地图‘3m’上，以难度值为3进行训练，达到60%以上的胜率。

2. 提前安装训练环境

- pysc2 https://github.com/deepmind/pysc2
- smac https://github.com/oxwhirl/smac

3. 参考资料

- https://arxiv.org/abs/1812.11794
- https://arxiv.org/abs/1605.07736
- https://arxiv.org/abs/1706.05296

。

具体要求

- 地图为3m，难度为3，实现CommNet或VDN，自行选择其他MARL算法也可，达到60%以上的胜率，提交相关**代码**与**报告** (10`)
- 对实现算法进行改进与优化 (4`)
- 报告中报告中详细阐述算法原理(2`)
- 报告中展示算法结果曲线图，分析自己的实验结果(2`)
- 报告中指出当前算法的不足之处与，并介绍自己的改进方法(2`)