项目实训：第一天

- 确定组内分工
  - 我负责部署模型这部分。
- 了解项目需求
- 计划接下来的工作
  - 先从读论文开始
  - 阅读代码
  - 跑通预训练模型
  - 部署模型
  - 对模型做出改进，并重新训练，探究改进后的模型相较于原先的模型是否有提升

项目实训：第二天

- 阅读论文
  - 了解模型结构 — text-encoder + imgGAN。
  - 了解如何进行CA操作，以及它的优点和效果。
- 初读代码
- 目前的问题
  - 已有已经训练好的模型，但是是tensorflow规格保存的模型，tensorflow模型可以转成pytorch模型，但是需要保持模型各个参数命名一致，需要决定自己是从头训练模型，还是花费写心思转换网络参数。
  - 所给出的代码，没有text embedding这个网络，在icml2016的实现中有完整的网络结构和预训练模型，但是使用lua语言写的，正在尝试转换成python语言。
- 下一步工作
  - 部署模型

项目实训：第三天

- 阅读代码
- 解决几个问题
  - 尽可能使用别人已经训练好的模型，如果不对网络修改，仅仅重现别人的工作，是无趣的。
    - 问题：没有现成的令人满意的完整的pytorch版模型。
    - 两个解决方案
      - 服务器上搭建TensorFlow环境，使用TensorFlow预训练模型。
      - 将TensorFlow模型转成Pytorch版模型参数，这其中需要修改大量的模型结构命名。
- 下一步工作
  - 部署预训练模型

项目实训：第五天（一）

- 读关于text embedding的文章：Reed et al. 2016 — 解决细粒度文本描述和图像之间的语义gap。
  - 理解细粒度fine-grained图像分类和零样本学习zero-shot两个概念
  - 了解到几种text embedding的方法，和这几种方法性能的优劣（Word-CNN+RNN效果最好）
    - BoW +MLP
    - Word2Vec
    - Char-CNN+RNN(LSTM)
    - Word-CNN+RNN(LSTM)
  - 了解到图像文本的对称训练DS-SJE和异步训练DA-SJE两种方式和它们的性能。
- 阅读另外一篇单GAN网络的论文
  - 一篇鼻祖级别文章
  - StackGAN在它基础上堆叠了另外一个负责还原细节的GAN

项目实训：第五天（二）

- 通过阅读代码，深入了解StackGAN网络细节
  - StackGAN在还原细节时，采用了上采样和卷积方式，由于卷积具有平移不变性，因而在stage1，网络只能生成大致轮廓颜色，进一步加入在空域范围内加入condition信息后，从而能够进一步细化图像。
  - 在语义分割领域也存在类似细化的问题，我在想能否把用于解决语义分割的网络结构迁移到图像生成这个网络。
- 关于框架与训练模型
  - 我之前一直抱着尽可能使用预训练模型，可是发现，各个模型的实现所使用的框架各不相同，自己要实验TensorFlow、Torch等架构，还要学习lua语言，工作量太大。
  - 下定决心，自己训练模型。
  - 使用~~Baidu的Paddle框架~~或pytorch框架
- 下一步工作
  - 收集数据集
  - 开始编代码，训网络。

项目实训：第六天

- 在转模型参数时，取得一些进展
  - 已经能够从t7中读取参数
  - 并且在pytorch中和lua的模型也完全对应，
  - 参数大小也匹配，
  - 可是输出就是不正确
- 忙活了一天，没整出结果

项目实训：第七天

- 再次尝试转换模型参数。
- 从论文实现的GitHub源码的链接中下载好数据集，准备自己训练

项目实训：第八天

- 转模型无果
  - 我从.t7文件扒取得到模型参数，并且把它填充到pytorch内的模型类型.pth，
  - 可是经过验证，模型的输出不正确。
  - 经过两三天折腾，最终放弃
- 捋顺别人实现的代码，并且跑通程序，目前正在训练
  - 本以为训练一个模型需要几天时间，所以本着加快进度的目的，尽量不训练网络。
  - 在训练过程中发现，数据集太简单，几个小时就可以训完。
  - 教训：如果没有写成可供参考的模型，自己训！！
- 下一步工作
  - 等待训练完text encoder后，继续训练StackGAN

项目实训：第九天

- 训练模型
  - stackGAN网络是在别人工作的基础上进行的，若要重新从头开始训练，需要经过以下两个阶段：
    - 文本编码器的训练
    - GAN网络的训练
  - 目前，已经训练出CUB和flower104这两个数据集的text embedding网络，
  - 还需要训练coco数据集上的网络
- 今日工作
  - 整理coco数据集，编写coco的数据集加载部分的代码
  - 编写Xception图像编码器，和训练这个网络代码
  - 目前已经调通程序
- 下一步工作
  - 等待空出GPU后，先训练Xception，使用Xception对图片编码，进而利用图像编码向量训练文本编码向量，最后训练GAN网络。

项目实训：第十天

- 训练模型
  - 正在训练Xception，coco2017数据集有100k张图片，数据集相对cub、flower比较大，从头开始训练一个分类网络需要花费一些时间。
- 编程部分
  - 捋顺StackGAN网络框架，数据集加载和模型训练部分的代码才刚刚开始。
- 下一步工作
  - 训练完Xception后，先暂且放下coco数据集，先去把另外两个小的数据集训完。
  - 思考：
    - StackGAN多阶段训练，包括了图片编码、caption编码，和caption解码生成图像，能否将编码器和GAN网络融合成同一个网络，就如同Fast RCNN对RCNN做出的优化。

项目实训：第十二天

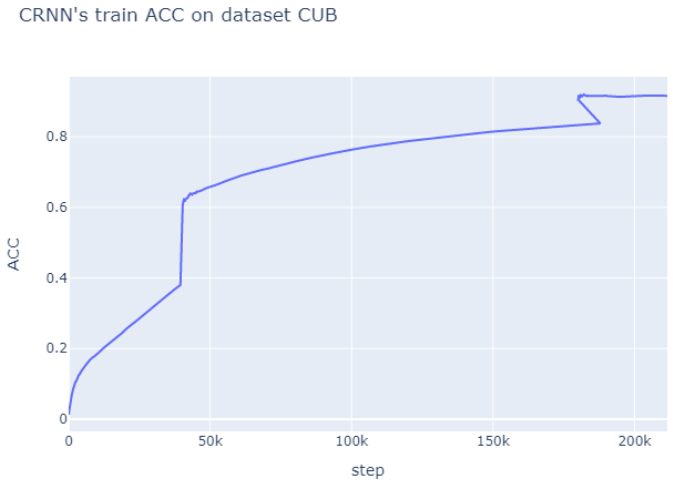
- 别人占着GPU，不能训练，这几天再等等，可以先把程序编好。

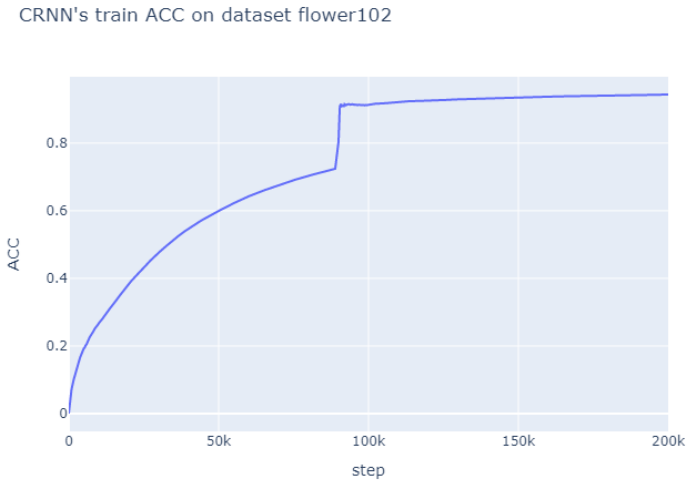
项目实训：第十三天（一）

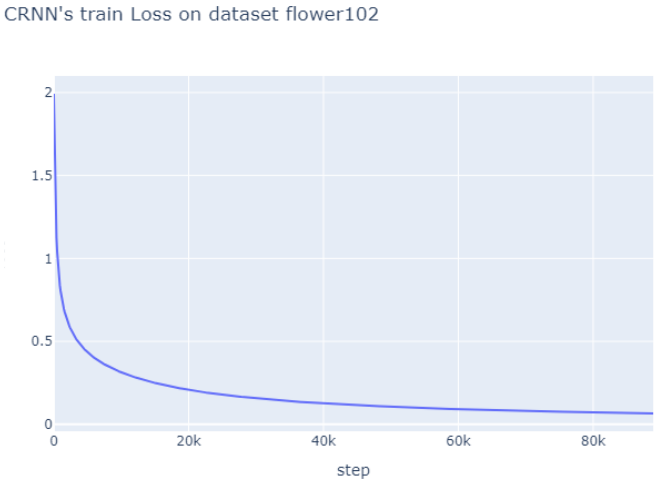
- 今天把之前训练好的编码器整理了一下
  - 把数据集中的captions转成1024维度的向量，这个向量将直接被用于训练GAN。
  - 得到caption向量化表示后，将结果可视化，结果将在后面几张PPT中展现。
- 网络训练部分
  - 今天把StackGAN调试成功了，目前正在第一阶段的训练。
  - 下面给出了目前网络所生成的图片。(real image & fake image)

项目实训：第十三天（二）

- 右图给出了训练CRNN网络时，损失和精度随着迭代次数的变化图线。
- 损失采用原文公式中所描述的那样，要求：
  - 同一个类中的图像向量和文本向量之间的距离要尽可能相近
  - 不同类的图像向量和文本向量之间的距离要尽可能远
- 这里的精度的计算方式采用了复现代码所实现的那样，两个距离最近的图像向量和文本向量属于同一类的视为正例，反之视为负例，由此计算精度。

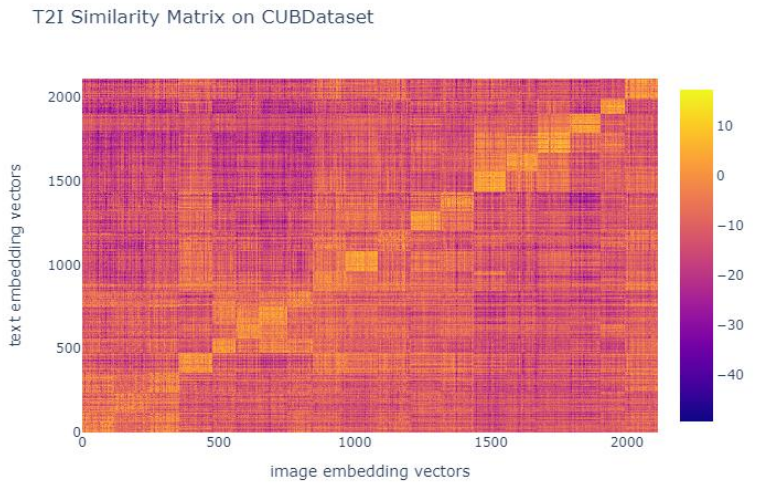


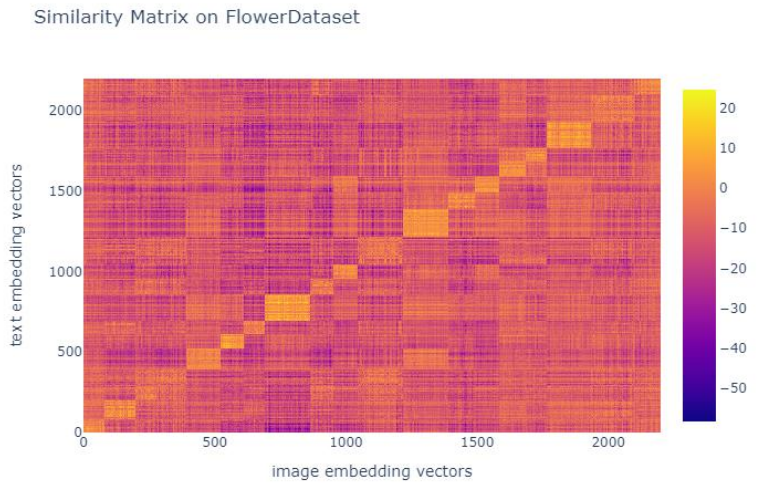


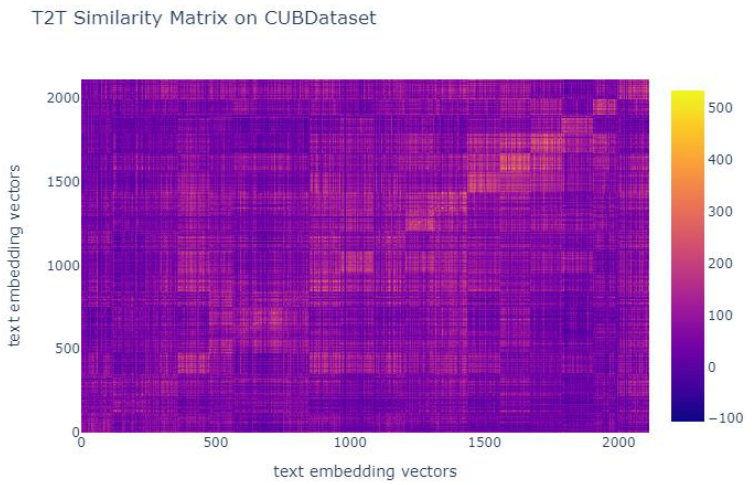


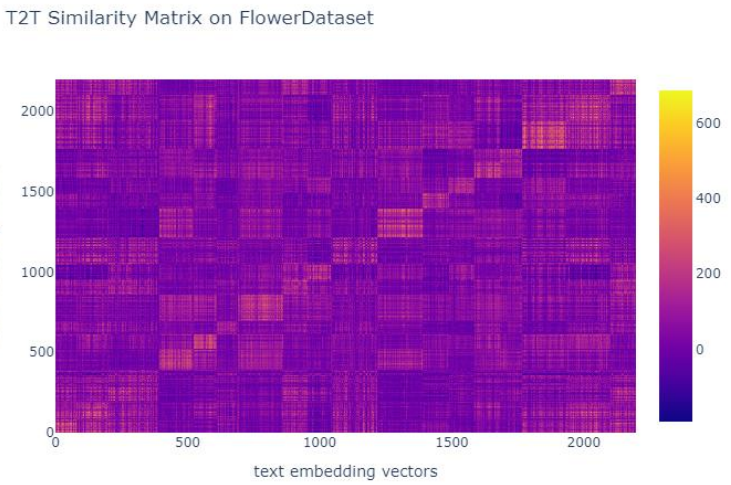
项目实训：第十三天（三）

- 右图给出了训练结果的可视化表示。
- 正如全面所说的那样，一个好的编码器，应该要把同一个类的caption和image编码到相同的子区域空间。
- 我们使用向量点积来表示向量的相似度，右图中下面两张图是文本向量和图像向量的相似度矩阵，矩阵中的第i行、j列中的元素是第i个文本向量和第j个图像向量之间的距离。下面两张图是文本对文本的相似度矩阵。
- 为了使结果更好看，我按照类别的顺序依次组织矩阵的行列，同一类放到一块，我们发现得到的相似度矩阵呈现明显的分块，这达到我们上面所说的目的。



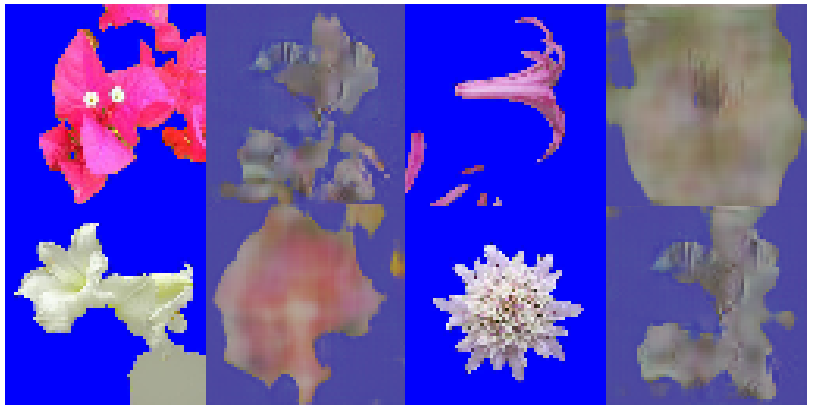






项目实训：第十四天

- 接着训练StackGAN
  - 在训练过程中发现，网络中的鉴别网络最先收敛，生成网络稳定性很差。
  - 鉴别网络的误差可以很快降到0.01这个数量级，但是生成网络从刚训练开始就上升，直至目前居高不下，一直维持在10左右。
  - 不知道是不是学习率的问题，原文中学习率每100epoch降为原来的0.5倍，而我实现中把学习率调度的代码去掉了，全程使用初始学习率，不知道这是不是导致生成网络损失波动的原因，我现在把代码改了，希望明天能得到好的结果。
- 截至目前，训练效果如下，

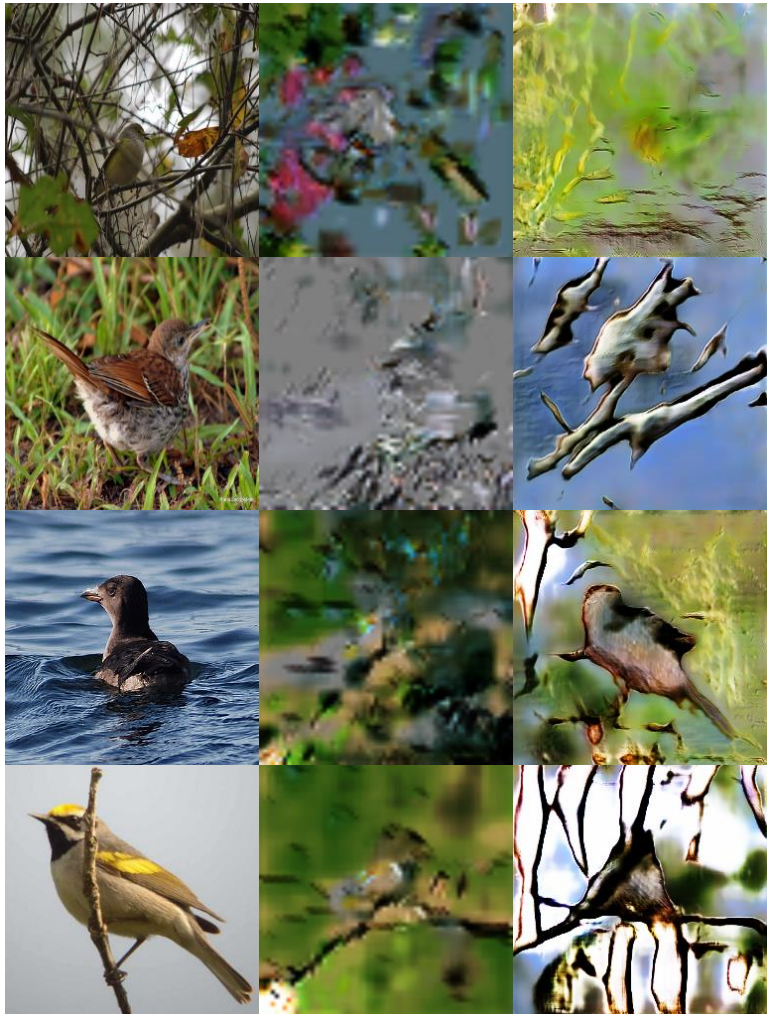


项目实训：第二周总结

- 第一周尝试转模型参数无果
- 第二周从头开始训练
  - 目前，我已经得到CUB数据集和flower102数据集上的文本编码器，
  - 没有找到coco数据集上的图像编码器，打算自己训练，使用Xception作为图像编码器，之前由于训练速度太慢，只训练到16epoch。
  - 这两天在数据集CUB和Flower102训练StackGAN。
- 目前存在的问题有：
  - 正如前面所说的那样，StackGAN中的生成网络部分不收敛。

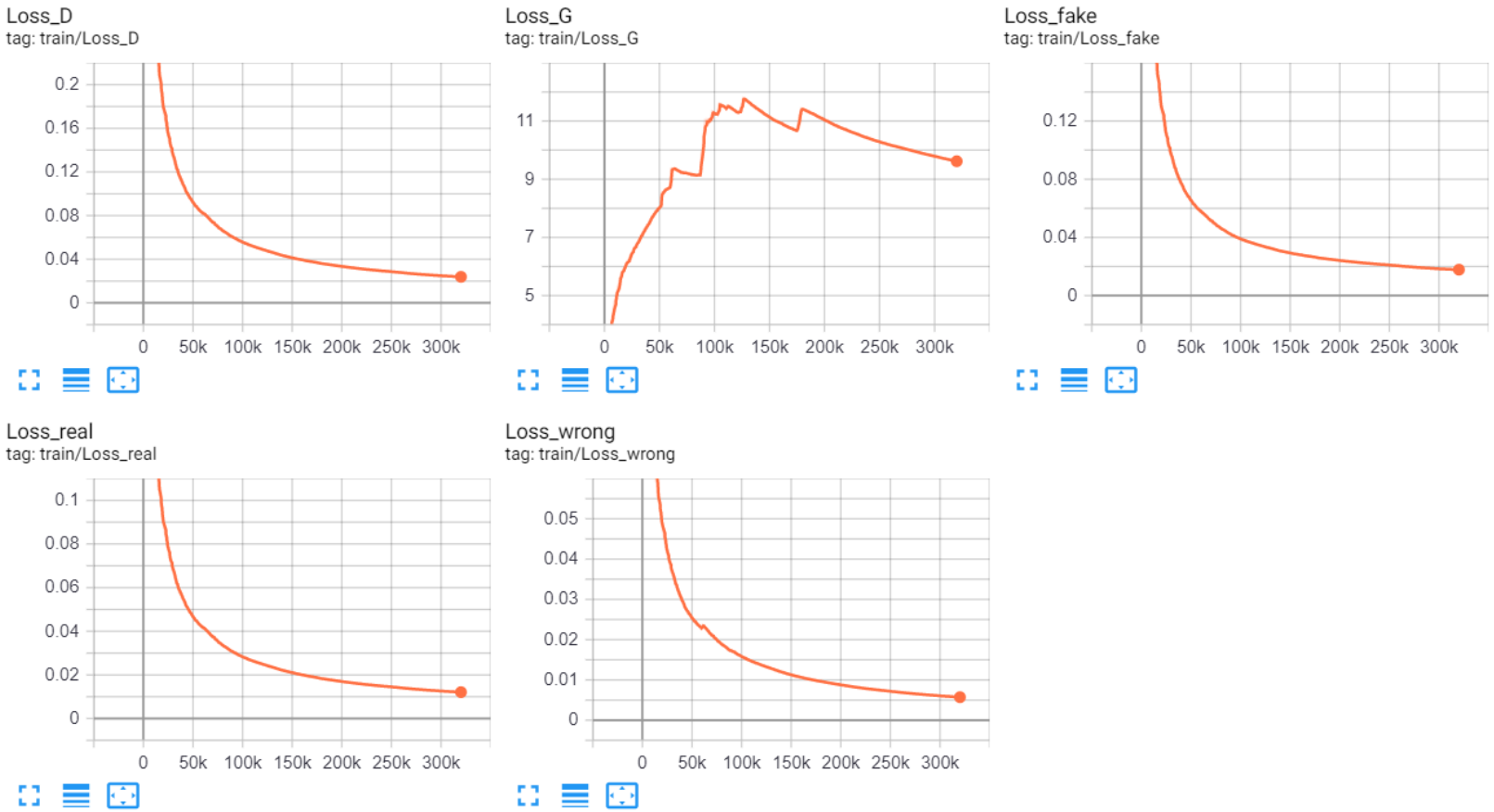
项目实训：第十五天

- 网络训练
  - 把学习率调小，仍然不收敛。
  - 我找不出错误。
  - 编码器使用的是自己训练的编码器，不知道是不是编码器的问题。
  - 不知道是网络本身固有的问题，还是自己编码的问题。
  - 结果两天的训练，stackGAN在CUB数据集上的第一阶段训练到545个epoch，并开始训练第二阶段。
- 截至目前，训练效果如右，（原图-stage1-stage2）
- 找到另外关于GAN的文章，HDGAN这个有pytorch版本的预训练模型，如果训练不出来的话，我就去看看这个模型。



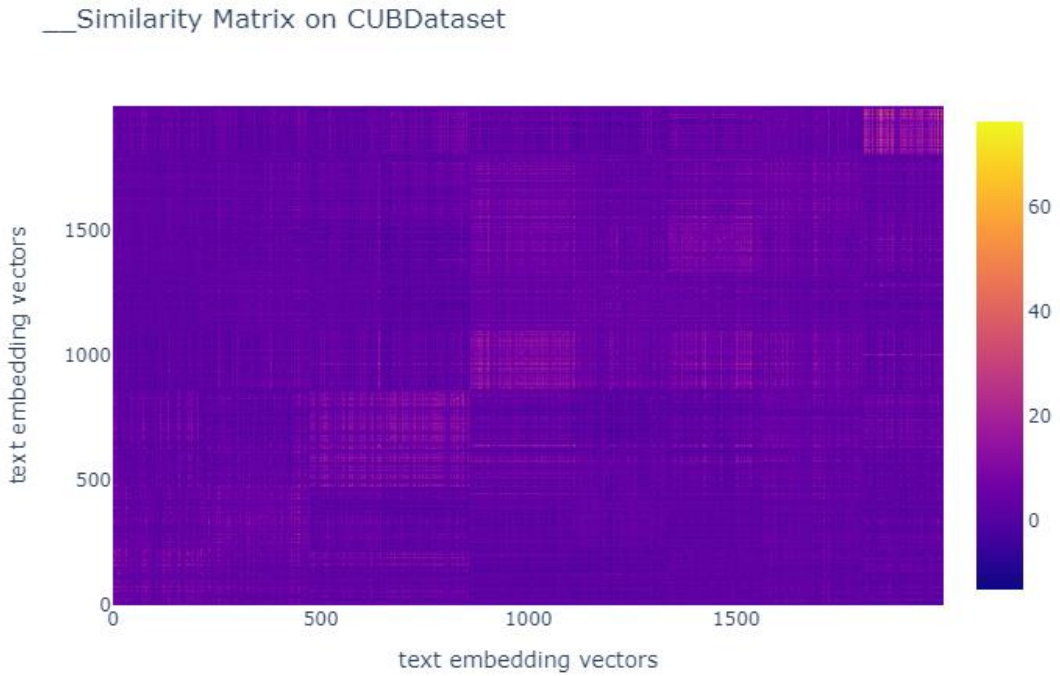
项目实训：第十六天（一）

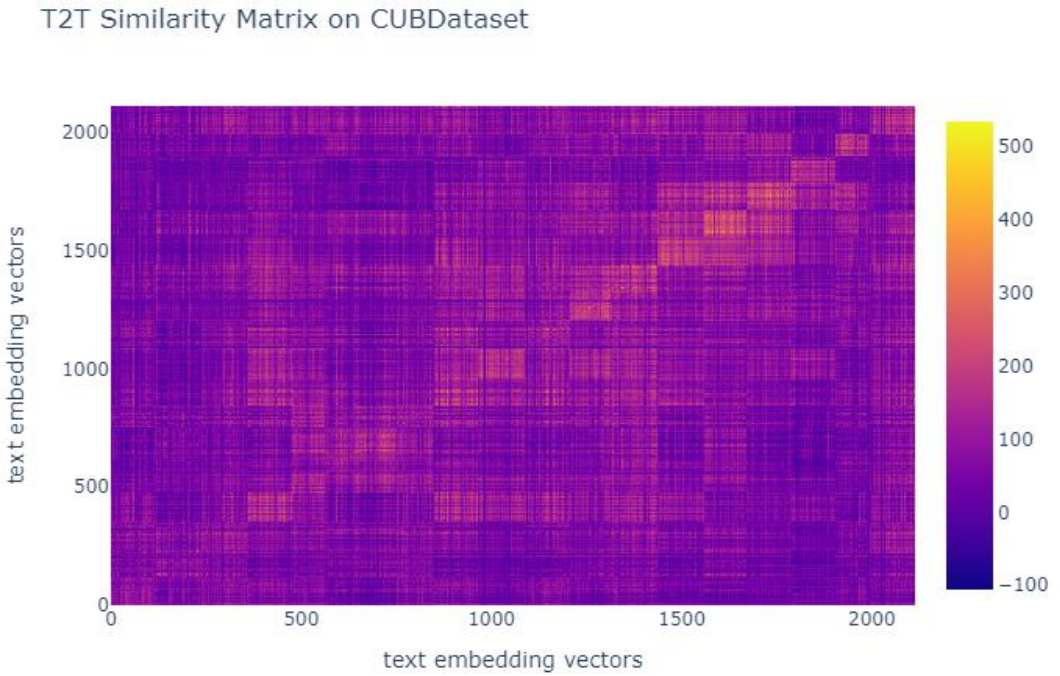
- 网络训练
  - 昨天在CUB数据集上训练的效果并不好，
  - 今天转移到flower102这个数据集，仍然存在和cub一样的问题。
  - 如右图所示，这是StackGAN在训练过程中各个损失的曲线图，注意到，这已经经过平滑。
  - 刚开始的时候，鉴别网络快速收敛，生成网络的损失在增加，而后波动，之后慢慢收敛到一个损失之较大的状态。



项目实训：第十六天（二）

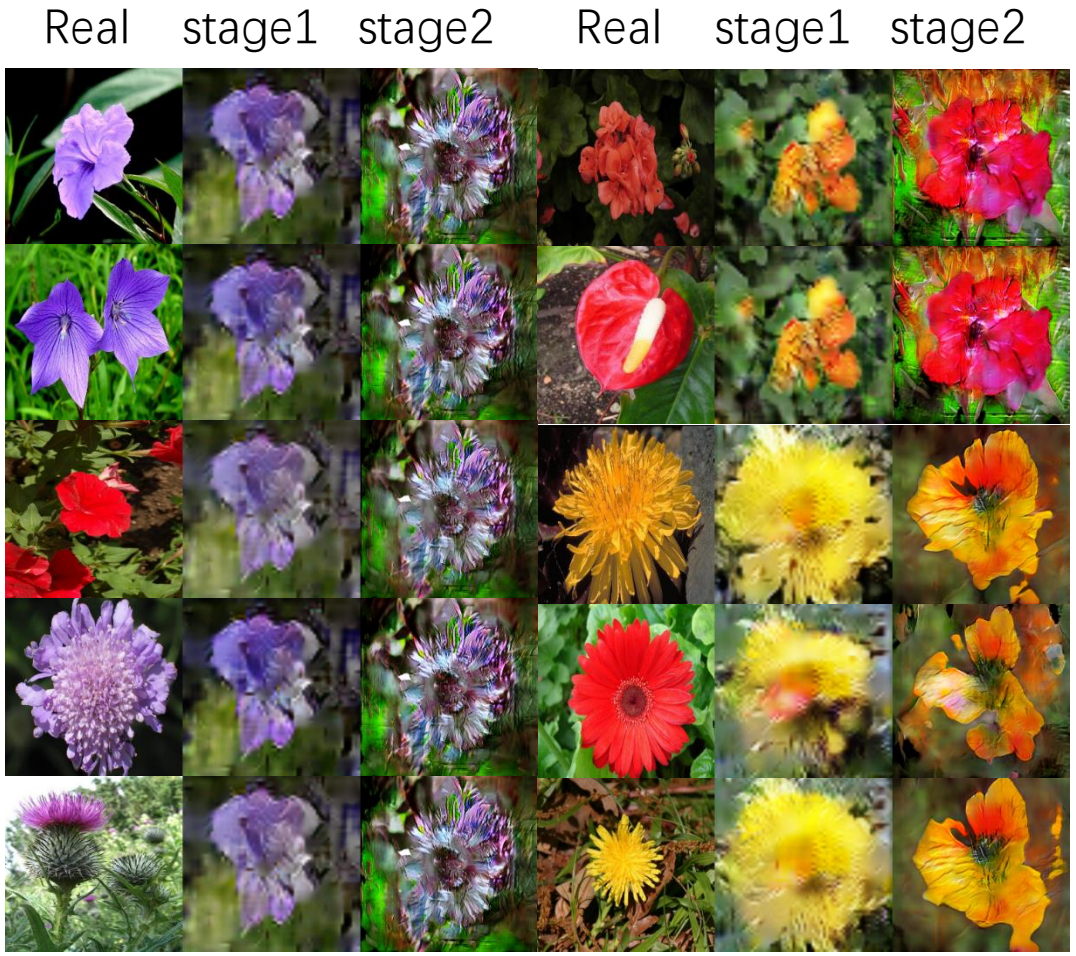
- 我在想是不是编码器的问题，我找到原文中所使用的文本编码向量，将它按照前面所说的那样，作出相似度矩阵图像，
- 如下，右侧是我们的编码得到的向量，左侧是人家使用的编码器向量。
- 我们的类与类之间的界限更加清除，人家的比较模糊。





项目实训：第十六天（三）

项目实训：第十七天



- 通过结果可以基本断定编码器确实问题。
- 自己训练的编码器确实存在问题。

- 今天在看HDGAN，它是StackGAN##，它所使用的编码器仍然是cvpr2016那篇文章的编码器，
- 如果不使用人家的编码器，别人预训练好的模型都不能使用。
- 刚才在安装torch7，没成功。

项目实训：第十八天

- 像icml2016那篇文章（暂且称作SingleGAN）、StackGAN、HDGAN这些网络都是在cvpr2016这篇文章的编码器基础上做的工作，我们需要它的编码器，如果得到了这个编码器，后续的工作就会非常简单（因为可以使用别人已经训练好的模型），在之前已经提到过，cvpr2016的那篇CRNN编码器使用的是lua语言写的，模型是.t7格式的，尝试转模型、配置torch7的这些方法均没有成功，自己训练的CRNN又出现莫名的问题。
- 我又想到另外一种解决方案，目前存在caption到embedding的编码结果，存在pytorch版本的CRNN编码器代码，唯一缺少的是编码器模型（参数），如果像前几天自己训练模型，后续的GAN网络都必须自己训练，我们可以从另外一个角度出发，使用caption当输入，别人的embedding当label，来训练我们自己的编码器encoder，
  - 给定caption，我们的编码器得到的embedding向量为

$$v'_{emb} = encoder(caption)$$

  - 我们希望我们得到的v'_{emb}能够和别人的编码器的输出v_{emb}相同，所以设置损失函数

$$loss = SmoothL1Loss(v_{emb}, v'_{emb})$$

  - 目前上述想法已经完成编码，正在等待训练。

项目实训：第十九天（一）

- 昨天的方法成功！！
- 目前，我所能够部署成功的模型有：

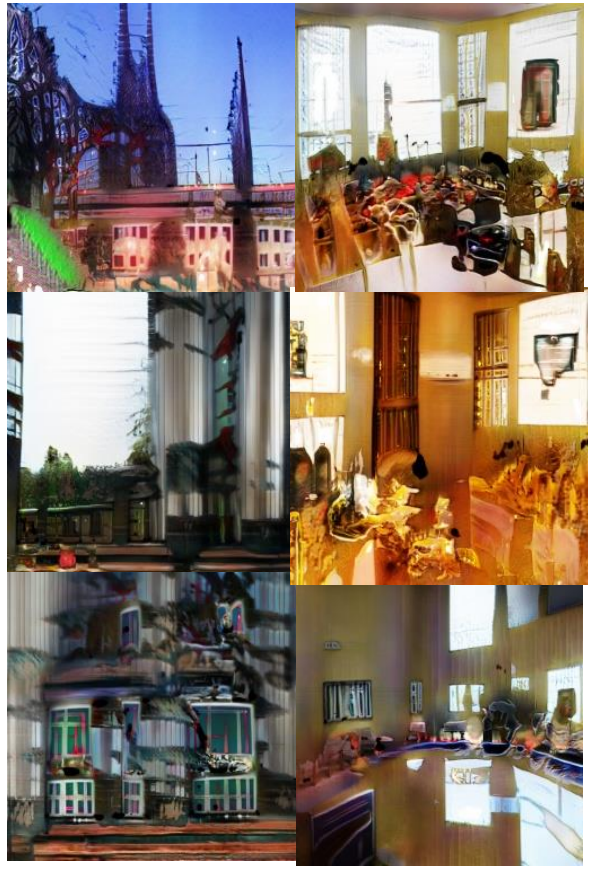
	CUB	Flower102	COCO
SingleGAN			
StackGAN			√
HDGAN	√	√	?

  - 其中，HDGAN的COCO数据集上面的captions embedding结果，没有下载下来，
  - StackGAN的CUB和Flower102的预训练模型是TensorFlow版本的训练模型。
  - SingleGAN的全部预训练模型均为.t7格式。
- 下一步工作
  - 和后台对接。
  - 训练在CUB数据集和Flower102数据集上训练SingleGAN和StackGAN（如果时间够的话）。

项目实训：第十九天（二）

HDGAN-CUB

HDGAN-Flower102



HDGAN-COCO

项目实训：第二十天（一）

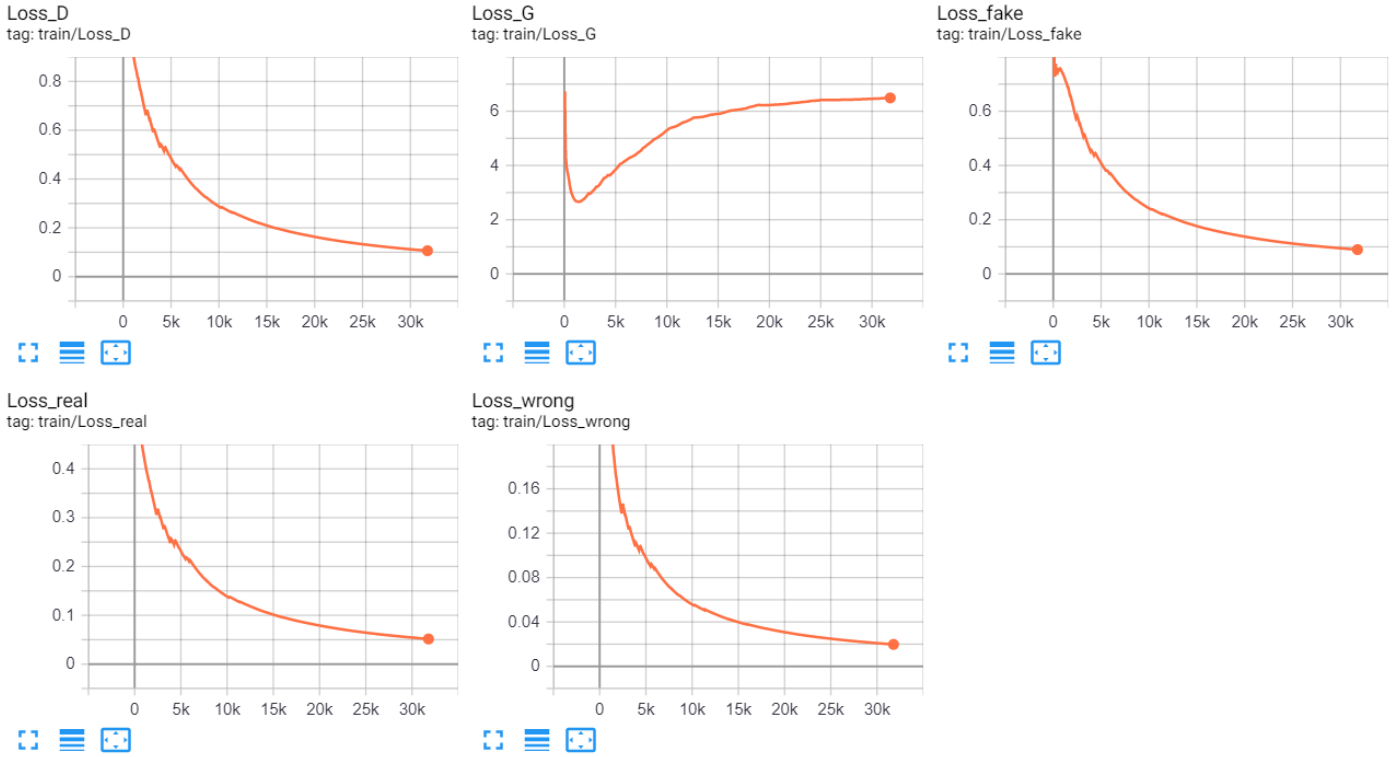
- 目前，我所能够部署成功的模型有：

	CUB	Flower102	COCO
SingleGAN			
StackGAN			?
HDGAN	√	√	√

  - StackGAN的COCO数据集模型找不出错误，模型加载正确，模型超参也完全对应，可是第二阶段就出不来想要的结果，第一阶段生成的图片是正常的。
- 训练网络
  - 使用HDGAN的embedding结果来重新训练StackGAN，
  - 反思我前一周自己训练的embedding为什么效果不好。
  - 抱着训练不完的态度，我开始在Flower102上面训练StackGAN。

项目实训：第二十天（二）

- 右图是我刚刚训练得到的损失曲线，这次使用的HDGAN的文本编码结果，略微修改了超参数。
- 对比前三、四天的那个损失曲线，生成网络损失值居高不下也许就是网络自身的原因。
- 这次换了文本编码，结果相比之前训练的要好，至少它不会生成综合了多种不同种类的花特征的图片。

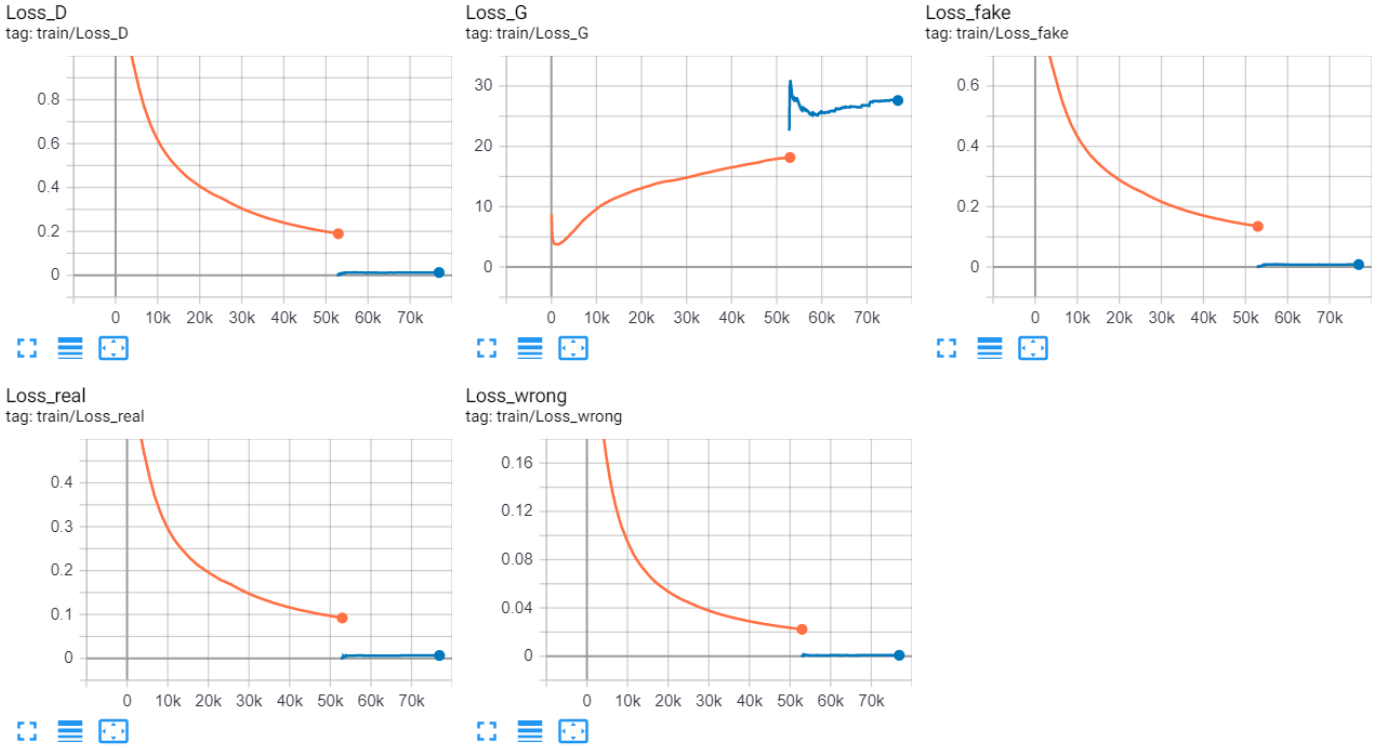


项目实训：第二十一天（一）

- 论文及编码部分
  - 今天在看MirrorGAN，这个网络相较于以往的网络做出的突出改进有：
    - 大量引入attention机制，无论是在上采样卷积还是在特征图与condition向量结合的时候，均使用了attention机制。
    - 为保证生成图像的语义一致性和完整性，它引入了另外一个损失，将生成的图片经过I2T逆操作得到生成图片的caption，将这个caption和ground truth的caption相比较，计算损失。
  - 编码部分，把MirrorGAN的网络结构的代码认真过了一遍，掌握了网络结构。
  - 但是作者以及复现者均没有提供生成网络的预训练模型，这个网络无法部署。
- 训练网络
  - 我无法根据损失判断网络是否应该停止训练，StackGAN第二阶段只训练到300个epoch。
  - 注意到我在数据集Flower102训练StackGAN网络的时候，没有使用数据增强，这可能是导致我训练效果差的原因的一个方面。
  - 接着在CUB数据集上训练StackGAN，并使用数据增强。

项目实训：第二十一天（二）

- 右边给出了StackGAN在Flower数据集上第二阶段的损失图像，
- 中间中断了一次，由于使用了数据平滑，所以中间出现断层。
- 下面给出了最终生成的图片。
- 效果并不理想。



项目实训：第二十二天

- 项目逐渐接近尾声，之后不计划训练网络，
- 最近，我做的工作逐渐偏离项目主题，目前我得到了几种模型，但最终只被小组录用了一个HDGAN的COCO模型。
- 下一步工作：帮助小组写文档，做出一些收尾工作。

项目实训：第二十三天

- 今日工作
  - 帮助小组制作PPT
  - 编写模型部分的代码维护文档
- 其它
  - 我简单看了看StackGAN++和AttnGAN
  - 了解到我之前StackGAN训练不稳定可能绝大部分原因不却决于自己代码有问题或者编码器的问题，StackGAN++在Introduction部分简单说了导致GAN不稳定的原因。
  - 我发现StackGAN++和HDGAN结构几乎相同，MirrorGAN的attention机制那部分网络也取自于AttnGAN。