Graph to text to graph尝试

**试验任务**：我们知道GAN网络在生成图片方面有着强大的功能，由其生成的人脸和真实人脸几乎没有区别，本实验我们期望的是将这一过程运用的其他事物的生成。过程是输入一张花朵的图片，由第一个网络将其处理输出网络对图片的理解的文字，将该文字向量作为第二个网络的输入最终生成与输入图片对应的整个网络所理解的图片。

**数据集**：实验使用的是已有的标准数据集Oxford-102 flower，原本打算训练好后再用其他不同的数据集（例如猫）进行网络迁移性能的测试，但由于text-to-graph的网络表现并不好，所以最终没有做该方面的尝试。没有额外的数据标注处理任务。

数据集的组成：

7370 images for training set, where each images is annotated with 10 texts.

819 texts for testing (you must generate 1 64x64x3 image for each text)

由于GAN生成对抗网络在训练过程中需要的数据输入为图片对应的文字描述，因此在训练GAN之前要先对Oxford-102 flower数据集进行处理。

第一个子问题直接使用的数据集是Oxford-102 flower，使用RNN来做图片生成描述，可用来对文字进行判别和表示，由于它是根据图片的内容和他前一个词生成下一个词，是遵循链式规则的；

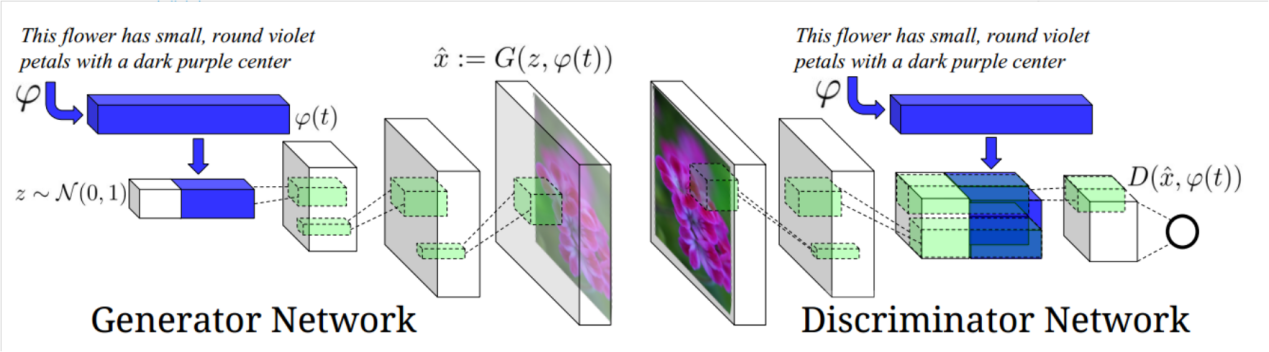
第二个子问题是使用GAN完成图片生成，模型输入为图片的文字描述。故模型使用的数据集为第一个问题的实验结果，即图片及其文字描述的数据集。

**算法原理**：

介绍GAN生成对抗网络的算法原理：

1、网络架构

基于文本的DCGAN结构：



生成对抗网络基于博弈模型，其中生成器网络（generator network）必须与其对手判别器（discriminator）竞争。训练过程可以总体表示为：生成器试图最大化鉴别器将其输入错认为正确的概率，而鉴别器引导生成器生成更逼真的图像。

模型输入为文本描述向量（上图中深蓝色的长条φ），在生成网络和判别网络中都有输入。在生成器网络中，首先我们将之前得到的文本向量，用网络全连接层对文本向量压缩，最后得到128维向量，然后和一个从正态分布中得到的随机向量进行拼接，输入生成网络中进行图像生成。

在判别网络中，首先对输入做几个stride=2的卷积，当feature map的大小变为2x2时，则又一次对文本编码结果做一个全连接层，将全连接层的结果拼接到这个大小为2x2的feature map上。然后对拼接结果做一个1x1的卷积和2x2的卷积。

DCGAN对卷积神经网络的结构做了一些改变，以提高样本的质量和收敛的速度，这些改变有：取消所有pooling层。G生成器网络中使用转置卷积进行上采样，判别器网络中用加入stride的卷积代替pooling；在判别器和生成器中均使用批归一化batch normalization等。

2、loss函数

GANs在训练时需要同时运行两个优化算法，我们需要为discriminator和generator分别定义一个优化器，一个用来来最小化discriminator的损失，另一个用来最小化generator的损失。即loss = d\_loss + g\_loss。Loss都采用交叉熵计算得来。

①判别器Discriminator network：

对于辨别器discriminator，其损失等于真实图片和生成图片的损失之和，即 d\_loss = d\_loss\_real + d\_loss\_fake 。在 tensorflow 中可使用以下函数：

tf.nn.sigmoid\_cross\_entropy\_with\_logits(logits=logits, labels=labels)

在计算真实数据产生的损失d\_loss\_real时，我们希望辨别器discriminator输出1；而在计算生成器生成的 ‘假’ 数据所产生的损失d\_loss\_fake时，我们希望discriminator输出0。因此对于真实数据，我们使用label=1计算代价函数来训练判别器，对于生成数据产生的d\_loss\_fake使用label=0来计算代价函数。

②生成器 Generator network

生成器经训练尝试输出能使辨别器分配接近概率1的样本。

生成器的代价函数为

g\_loss=tf.reduce\_mean(tf.nn.sigmoid\_cross\_entropy\_with\_logits(logits=d\_logits\_fake, labels=tf.ones\_like(d\_logits\_fake)))

随着以上训练的进行，判别器会逐步增强自身的判别能力，而生成器生成越来越逼真的输出，以欺骗判别器。

3、与传统训练方法相比，GAN的优势和不足

①优势

GANs是一种以半监督方式训练分类器的方法，在你没有很多带标签的训练集的时候，可以不做任何修改的直接使用网络训练；

GAN不需要马尔科夫链，模型优化仅用到反向传播，优势在于采样概率高，生成数据更接近真实数据分布；

GAN能够解决数据不足的问题，例如多视角多任务的生成等问题；

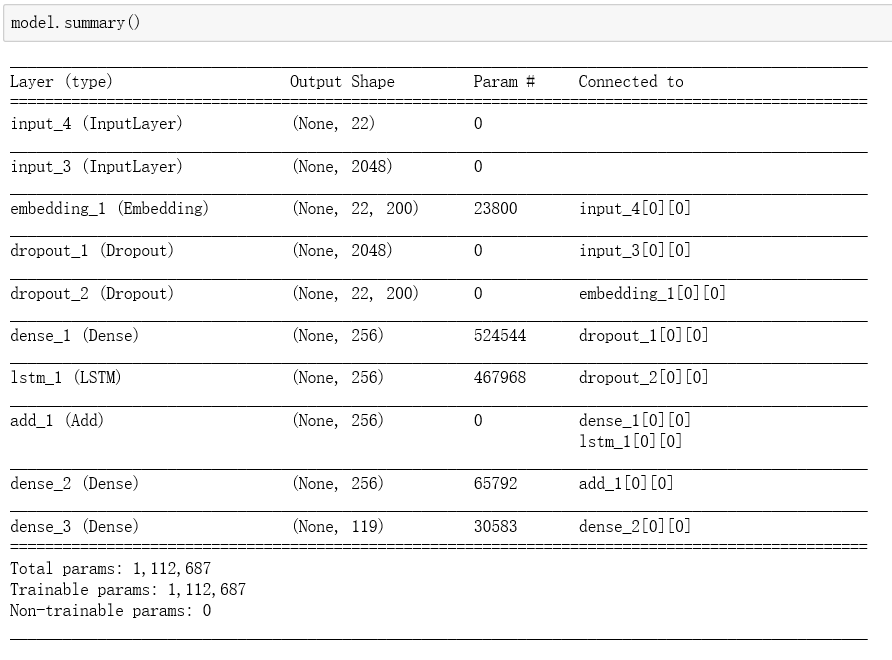
②劣势

在训练过程中，训练GAN需要达到纳什均衡，比较难以训练，存在训练不同步及梯度消失的问题，如何定义训练策略及损失函数成为关键。其中，GAN训练不同步的原因在于更新生成器G的N次，更次判别器D的1次。此外，梯度消失的原因在于当真实样本和生成样本之间具有极小重叠甚至没有重叠时，其目标函数的散度是一个常数，导致优化目标不连续。

**实验细节**：

*Graph-to-text网络*

这部分使用的是已有的模型训练，我们将其原本的训练数据集改成了text-to-graph部分使用的数据集（从结果来看网络的迁移效果很好），其他没有做修改，网络结构如下：

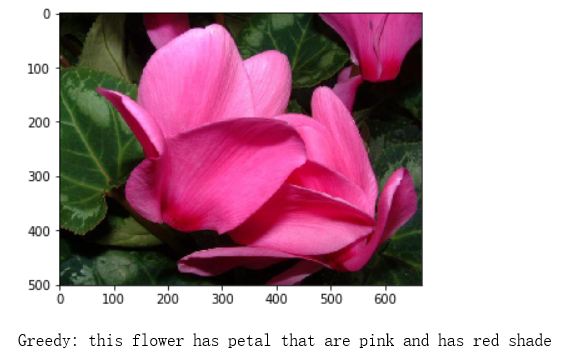


在对训练数据集的大小进行尝试后我们最终将这部分的训练数据集定为1000张flower高清图片。训练过程中损失函数稳定收敛，训练epoch数为10\*3（即分三次训练，每次的epoch为10）损失函数大小在三次训练时的衰减分别为

2.7233至1.1511， 1.0162至0.5603， 0.5462至0.3779

每次训练时收敛情况为从epoch=1到epoch=10逐渐变慢。

随机选择一张图片输入与之对应的生成文本如下：



可以看到效果不错，经过测试发现该网络训练输出的文本基本基本围绕花的petal，pistil和花的color这几个特征，而后面的text-to-graph网络的输入caption基本也是围绕这些来描述的，因此理论上只要后一个网络训练够好那么这两个网络的契合度将会很好。

Text-to-graph网络：

这部分参考链接（包括了实验中使用的完整代码和原理介绍）为

<https://nthu-datalab.github.io/ml/competitions/04_Reverse-Image-Caption/04_Reverse_Image_Caption.html>

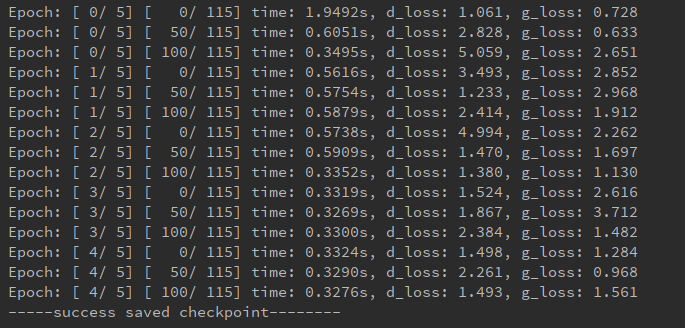
该链接中给出的代码是完整的但训练效果并不好（留了很多坑），需要使用者自主修改。网络中生成器和判别器的loss函数均为sigmoid交叉熵损失函数，优化方法为AdamOptimizer。数据集中的7370张图片均有caption与之对应，每个epoch对所以这7370张图片进行一次训练，原本代码中设定的epoch数为5。

在配置了相关的运行环境后我们按原有的代码进行训练，结果如是loss完全没有收敛的趋势。g\_loss和d\_loss始终在1到2之间起伏，有时甚至上升到4，考虑到epoch较少，我们尝试将epoch调到20，结果发现并没有改善情况，分析后认为问题在于网络的loss函数和优化器以及网络结构上，于是我们参考了GAN训练的一些经验和链接中给出的提示，主要参考了《令人拍案叫绝的Wasserstein GAN 及代码（WGAN两篇论文的中文详细介绍）》等几篇关于GAN改进的文章，我们主要关注在loss函数和优化方法的改进上。

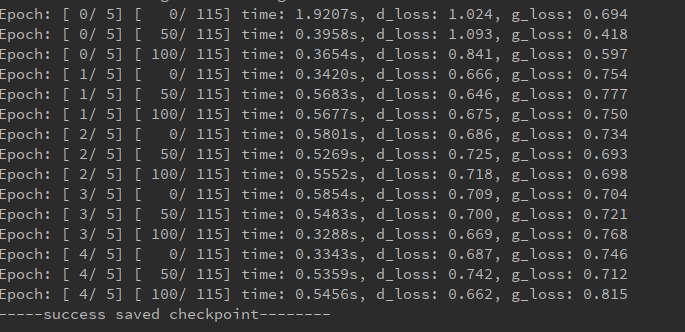
（一）优化器的改进尝试：

原本的优化器是adamoptimizer，其lr和beta1的设定值分别为0.01，0.5。我们首先尝试对这两个参数进行修改，lr由0.01改为0.001和0.0001，beta1由0.5改为0.1和0.3。实验后发现损失函数的收敛性得到明显改善。

以下是改进前loss在不同epoch中变化情况：



以下是改进后loss的变化情况（lr=0.0001，beta1=0.3）



然而尽管loss的收敛得到明显改善，生成图片的质量却没有实质性的提升，由generator生成的图片仍然像是打码后的花朵的图片，只有轮廓和颜色可以勉强辨认，甚至可以认为就是像素的一个随机整合。分析后我们认为其作用的主要是lr，但是lr取为0.0001后loss很快便达到一个较稳定的水平不再变化，可以认为此时的训练没有起到太大作用。

之后我们选择用RMSpropoptimizer代替原来的优化器adamoptimizer。在loss的收敛上的确得到进一步改善大约减少了0.1左右，但生成图片的质量大大下降，故放弃该尝试。

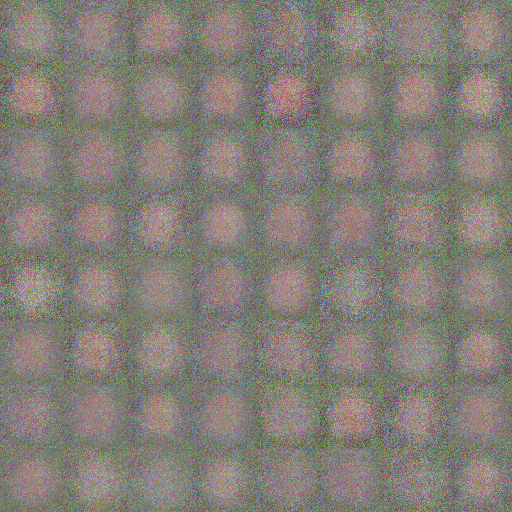
（二）Loss函数的改进尝试：

查阅资料后发现原始GAN使用Sigmoid交叉熵损失函数，容易造成梯度消失问题，使得生成器G的训练不充分。有的生成样本虽然成功欺骗了判别器D，但是其依然与真实样本的分布相差较远。Sigmoid交叉熵只管真假、不管距离，不会再惩罚这种样本，导致生成器G出现梯度消失。

目前最流行的替代sigmoid交叉熵损失函数的方法是Wasserstein GAN，该 GAN 用 Earth Mover distance ( Wasserstein-1 distance 或 EM distance) 来替换传统 GAN 的 Jensen Shannon divergence ( J-S 散度)。但是改进起来难度较大，因此我们退一步选择使用均方损失（ mean squared loss ）替代对数损失（ log loss ），对应的是LSGAN和WGAN。

效果依然不好，生成的图片像素分布更加均匀，轮廓和颜色完全消失。

以下是按自己的输入caption生成的图片的例子（生成了8\*8=64张）



**代码地址：**[**https://github.com/infdahai/AI\_Deep\_CoProj**](https://github.com/infdahai/AI_Deep_CoProj)

**小组分工情况：**

聂雷海：调研text-to-graph相关材料，给出了参考链接和相关数据集的下载链接，参与了这部分代码的前期调试工作。

程浩文：调研 graph-to-text相关材料，给出了这部分网络的代码。

邓迅：对graph-to-text和text-to-graph这两部分网络的运行和调试，根据调试结果写了实验报告实验细节部分的内容。

庄婉仪：运行text-to-graph网络，调研并编写实验报告的实验原理部分，制作答辩ppt。