第一次提交报告 Image Caption Generator

杨宝旭 罗迪 第2组

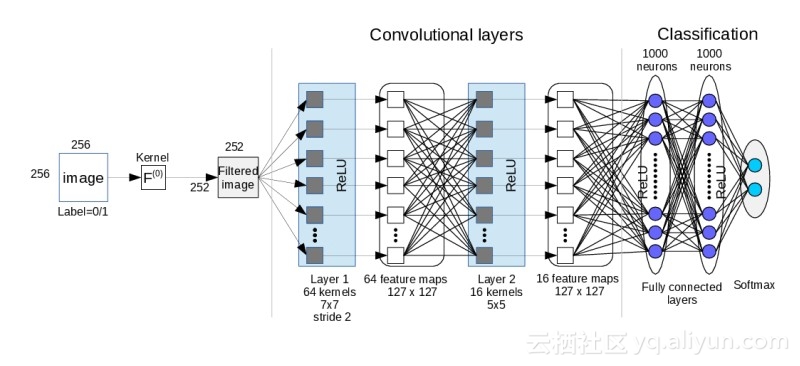
**一、论文阅读：**

首先，我们组在确定了选题是Task9 human eye之后，进一步细分确定了我们当前阶段的项目目标，即Image Caption Generator，学术上叫做基于自然语言的静态图像字幕生成问题，通俗的讲就是让机器学会“看图说话”。

首先我们组的主要任务就是阅读分析了这个领域的相关论文和进行了一些理论知识的学习，然后初步确定了要用到的模型思路。

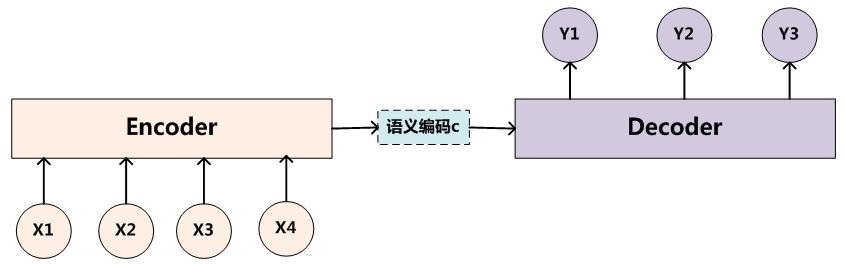
机器看图说话的英文叫做Image Caption，就是说自动从一张图片生成描述性的语句，这对于人而言可能是个小学一年级就会的很自然的过程，但对于机器而言，这个过程需要被分成三个阶段：识别图像中的物体、理解物体中的相互关系、用合理的语言表达。

首先是第一个部分识别图像中的物体，这个领域里面卷积神经网络大放异彩。通过输入图像，经过卷积计算层和池化层交替，最后接全连接层输出的方式，有效提取出图像的特征向量，并且有很好的分类效果。

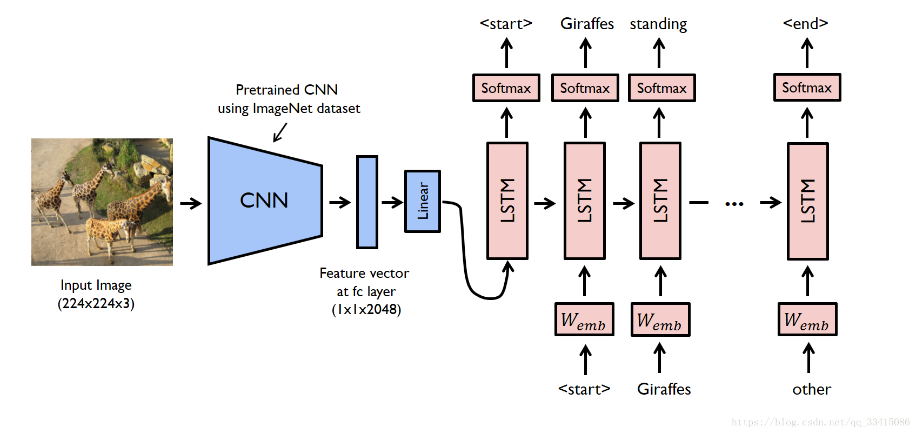


第二个部分是发掘出物体之间的相互关系，这种需要保留对之前状态的记忆，从而发掘出状态之间的联系的问题用循环神经网络可以解决，由于普通的RNN网络对于长期依赖有梯度消失的问题，这里主要用到的是改进后的LSTM网络。（遗忘门，输入门，输出门）

基于这两方面的知识，我们采用的是encoder-decoder编码解码框架，灵感来源是，既然机器翻译能够把各种语言翻译成另一种语言，图片经过处理后对于机器而言也只是一种语言而已，那么也应该可以用同样的方式进行“翻译”。进一步来说，采用卷积神经网络作为编码器，把输入的图片转换成一个特征向量，再把这个特征向量作为循环神经网络的输入，通过LSTM模型的解码器，得到用于描述的句子。再通过最大化给定图像产生句子的似然来输出前几个最符合的描述语句。



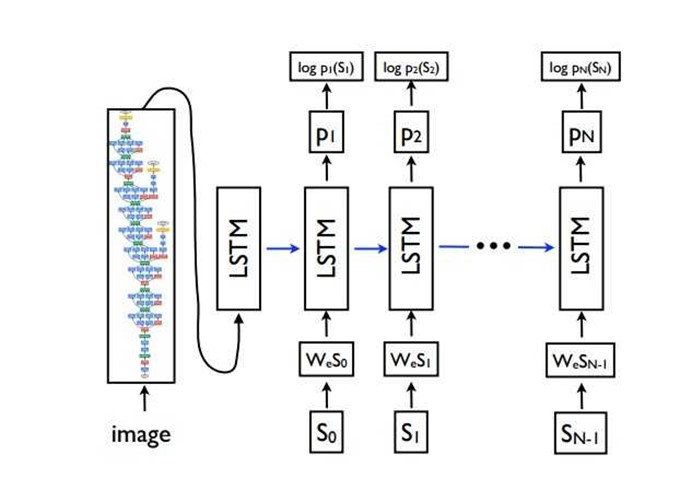
这里encoder-decoder框架本来主要用于文本的相关任务，这里引入到图片中，将图片也通过特征提取feature extract，encoder为向量形式。



**二、自己尝试实现一个简单的NIC模型**

这里我们基于Google的show and tell的原理出发，尝试实现一个简单版的NIC模型。

Google的模型被抽象出来作为image caption 的一种实现方式，被称为NIC模型。



NIC模型，其中encoder、decoder部分均可替换为多种实际模型

这里我们在自己实现是所采用的的具体数据集、模型、框架等如下：

**数据集：Flicker8k数据集** <https://www.kaggle.com/shweta2407/flickr8k-imageswithcaptions>

考虑到只是初步实现时，且时间、计算资源等较为有限，所以选取了几个常用数据集中规模相对较小的Flickr8k数据集，该数据集中一共有8000张图片（image），每张图片都有5句描述（caption）。分为Dataset和text两部分。

后续会尝试更大规模的MSCOCO、Flickr30k等数据集

**使用的框架：keras API**

因为我们刚刚接触深度学习，直接用TensorFlow写模型可能较为复杂，这里我们主要选取使用keras，其中内置有input、flatten（拉直为向量）、Dense、LSTM等封装好的函数，dropout、正则化等防止过拟合的方法也被封装成了函数，很方便我们直接搭建自己想构造的具体模型，另一方面其中还内置了一些已经训练好的经典模型，如VGG系列、inceptionV系列（均已在imagenet大规模数据集上预训练好了），可以直接调用作为模型的组件（这在我们的初步实现中也给予了很大帮助）。

**注：运行使用TensorFlow2.0后端，且(如果没有的话)需要pip install nltk包**

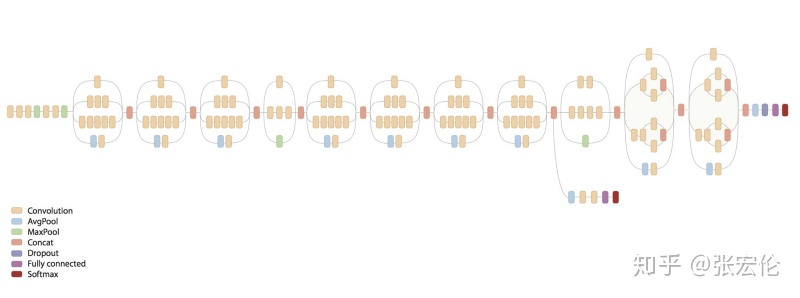
**具体模型简介：**

（因为有具体代码文件，这里的具体实现只展示部分截图）

**补充对代码文件复现的注意事项：**如果老师需要尝试复现进行检查的话，请将数据解压后放在datasets文件夹中，分为和Flickr8k\_Dataset、Flickr8k\_text两个文件夹分别存放图片和txt文件（如果解压后文件夹名字不对还请老师修改一下，保证路径正确）。model-params文件夹中内置了一个训练好的目前的current\_best\_model.h5,代码文件中测试部分使用的是该训练效果最好的模型（权重）。实际训练时也会产生存储的模型h5文件，文件名标注了loss信息。而put-your-image-here文件架用于存储测试用的图片，可以用其他图片替换掉其中的example.jpg（保持命名不变可不用修改代码中的路径）

**①图片的encoder部分 inceptionV3**

即一个图片的feature extract任务，Google模型中使用CNN模型实现的，这里用的是keras内置的训练好的inceptionV3模型作为具体实现。（代码部分会自动申请下载权重文件，需要一小段时间）



提取图像特征的深度学习卷积神经网络模型大同小异，其主要原理基本都是多层依次进行

卷积和最大值池化的过程。

卷积即依靠卷积核（每个卷积核代表一种特征）计算图片中每一片像素矩阵与卷积核进行相应卷积计算，实际上每个卷积核即提取出图片的一种特征。

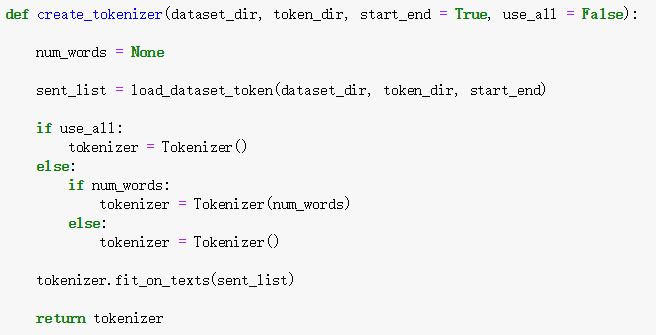
最大值池化则是起到了减少模型参数的作用。（提取反应特征块中的主要部分（相邻多块中的最大值））

可以看到上边的图示，经过不断卷积池化，不断提取出更多特征。Inception系列的网络的一个特征就是很深。当然也可以尝试用CNN、VGG等

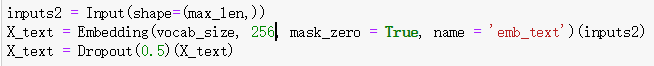


这里keras内置了在imagenet数据集上训练好的inceptionV3模型，且该模型的训练想达到较好的精度，规模也需要比较大，这里直接使用内置的模型，大大减少了我们自己预训练所需要的时间。

**②描述文本的词语序列的encoder：一个word embedding层 (及前面的tokenizer构建)**



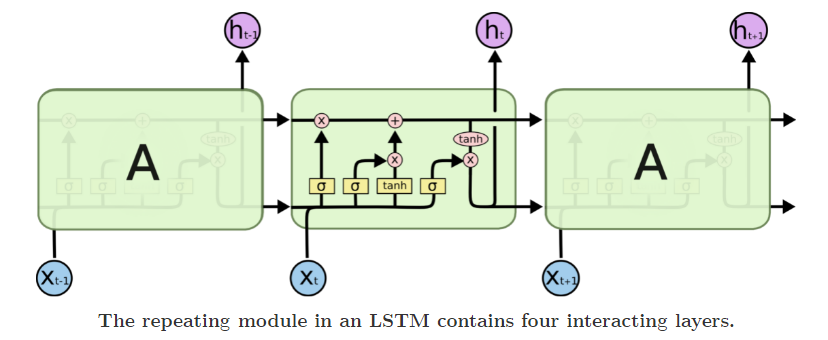
Tokenizer分词将完整的句子分成词语，同时也方便构建vocabulary词汇库。



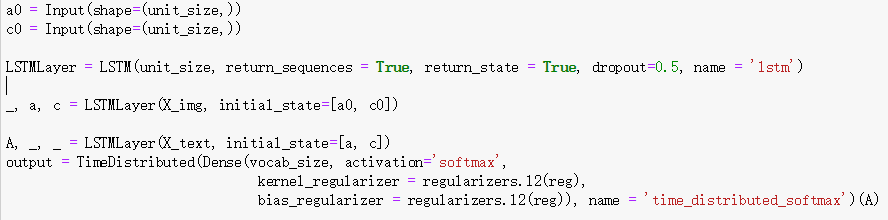
Embedding层将词语列表（向量）转化成一个唯一对应的数字向量

Dropout为防止过拟合的神经元概率失活的方法

**③decoder部分，根据已有信息不断预测出描述词语：LSTM**



LSTM模型相对于RNN解决了长期记忆的问题（即保存了较长向量信息中较早出现的信息对下一个词语的预测贡献作用）

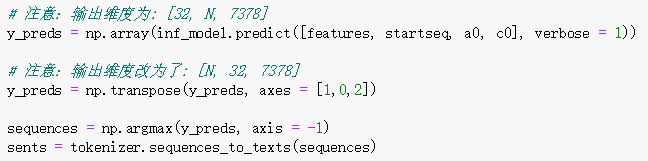


整个模型的思路实际上是根据输入的图片，从中提取出特征，转化成向量，

用LSTM根据输入的向量，不断预测出词语，且每次预测出的词语，也作为输入参与下一个词语的预测

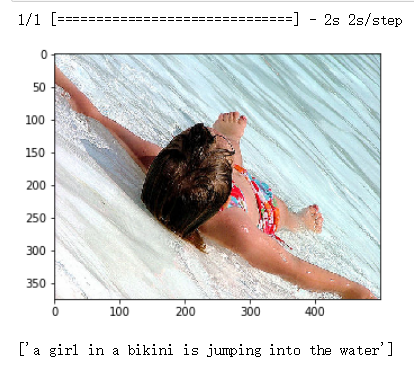
**④最终模型预测出的其实是多个结果，并且有相应的概率值等，涉及了一个搜索问题**

一般由greedy search 和beam-search两种算法，这里主要使用的第一种。（代码中beam-search的方法也有定义函数，但实际测试时并未使用）



这里用的是greedy search （实现主要是代码中的argmax）

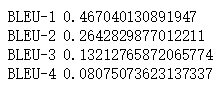
**模型效果展示：**

****

这里展示的是一个训练了很多epoch的best\_model(一个epoch大概需要5到10分钟左右)

但是由于训练数据集大小（图片和文字均有限）有限，所以实际上能够被识别出来的物体仅限数据集中多次出现的物体，并且模型预测时，能够预测出来的词语的词汇也受限于所有的文字描述中的词语。

（附上BLEU评价方法下的得分：

）

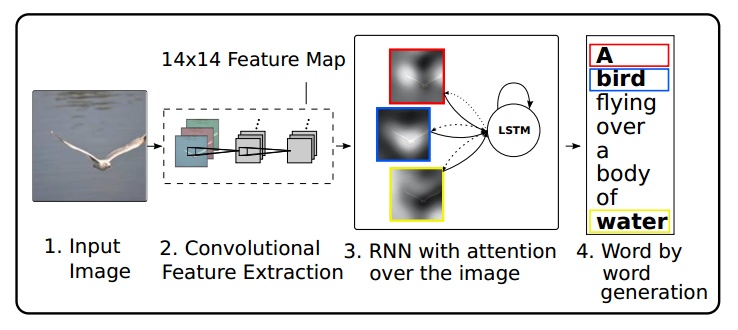
（只是一个初次尝试，精度还上不了台面，就不展开讨论了）

**三、后期展望及其他论文方法的实践**

后期除了会在一些更大规模的数据集上尝试训练（如MSCOCO、Flickr30k）外，还准备尝试一些其他模型结构。

① with attention的一些模型结构

## 如：Att-CNN+LSTM



在feature extractin和 LSTM之间增加了attention机制（还有待深入学习）

Attention机制还有一些具体不同实现模型结构如 soft attention、adaptive attention等，都有待下一步去了解。总的目的还是为了进一步提高精度。

②后续还会尝试一些应用上的扩展，如尝试实现输入一个图片序列，返回一整段文字等

可能会在此模型基础上增加对图像的中的文本描述向量之间的逻辑关系的模型进行相关训练。