# Image-Captioning

图像描述（Image-Captioning）问题融合了计算机视觉与自然语言处理两个方向，对图片输出一句话的描述。描述句子要求符合自然语言习惯，点明图像中的重要信息，涵盖主要人物、场景、动作等内容。

用简单的话就是说，输入给模型一张图像，模型输出是一句能够描述图像场景的文本句子。比如右边那张“鸟”的图片，模型就会输出 **“a bird flying over a body of water.”** 至于是中文的还是英文的，就取决于手头的数据集了。

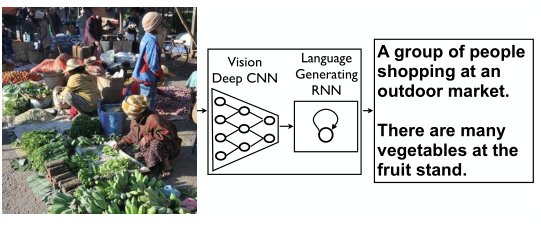
## 相关工作

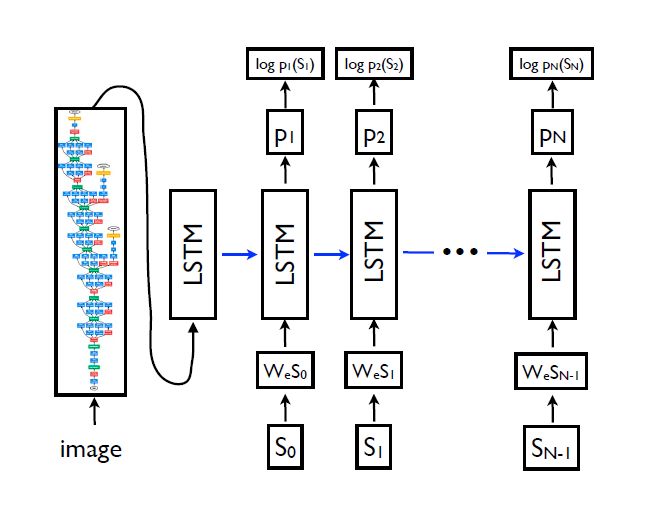
1. CNN+LSTM

主要从Google的 “**Show and Tell: A Neural Image Caption Generator**” 入手<https://arxiv.org/abs/1609.06647>

图片 image 经过 CNN 提取图像特征，最后 CNN 的输出变量是，这里取名为是因为模型的输入是 ，其中代表的是描述语句，就是文章开头那张“鸟”的描述文本 “a bird flying over a body of water.” 。这里有个问题，就是LSTM接收的输入是一个一维向量，那么就得把在高和宽的维度上做个均值处理，使它变为的大小，**它的物理意义表示图像的全局信息特征（由于是全局特征后面的论文使用注意力机制观察局部特征）**，把全局信息特征再送入到 LSTM 中。

模型结构如下：（图源论文1）





总结：其实从上图中可以很清楚的看到，图像信息只是在最开始输入进 LSTM，编码的图像信息被保存在中，后面就只有单词输入到 LSTM 中。因此编码的图像信息越往后越微茫了，提供不了图像的有效信息供 LSTM 去生成单词。

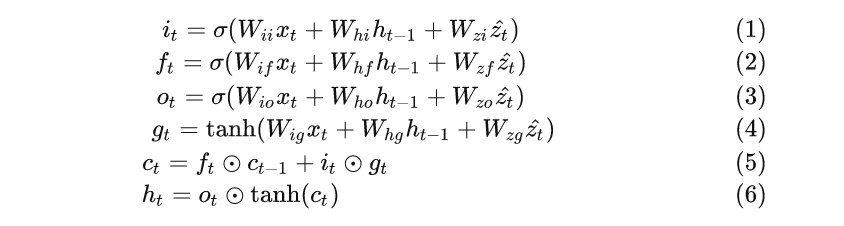
1. CNN+LSTM+attention

然后是 “**Show, Attend and Tell: Neural Image Caption Generation with Visual Attention**”<https://arxiv.org/pdf/1502.03044v3>。

这个论文是在之前说那篇 **“Show and Tell: A Neural Image Caption Generator”** 的基础上引入一个注意力模块，首先来从直觉上理解这个注意力，它的作用就是模型在每个时间步生成单词的时候，模型会关注这个单词所对应的图像区域。比如下图 (图源论文2)“狗” 的照片，模型在生成 “dog” 这个单词时，模型会着重关注图像中 “狗” 的区域，所以说下图中 “狗” 的区域要比其他地方要来的明显点。



我们通过对比与原来LSTM的计算公式来看注意力机制引入后的不同，与原来的 LSTM 6个公式相比，在变量的计算上多了一项的计算，把也编码进来了，也就是图像的信息，整体的数据流没有变化。所以可以用变量去生成单词，但是在 **“Show, Attend and Tell”** 论文中依靠了去解码单词的。



总结：整体模型的训练逻辑和推理逻辑和 LSTM 没有区别，只是输入的数据包含两种图像信息和单词信息。

1. Transformer+LSTM

“**Attention on Attention for Image Captioning**” <https://arxiv.org/pdf/1908.06954.pdf>。

Transformer中的encoder是用来编码源语句的高层表示，然后这个高层表示的特征向量送入decoder去解码出对应的目标语句，也就是decoder具有生成文本的功能，而LSTM也具有文本生成的功能，因此自然而然想到可否用LSTM替换掉decoder。

## 模型实现（1，2）

模型实现**简单NIC模型和NICwithATT模型**，但是由于计算资源有限，目前只利用模型1使用Flickr8K数据集进行了训练，模型2在接下来的时期将会使用AI challenger2017数据集进行中文图像描述资源的训练。

### 1.数据集

Flickr8K（英文）和AI challenger2017（中文）

1. Flickr8K

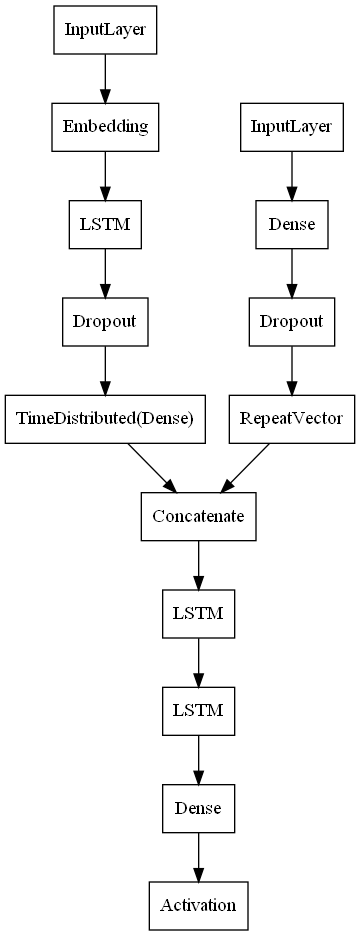
包含8000张图片，每张图片有5句英文描述。分为dataset和text两个部分，其中dataset存储的是图片文件，text存储的是描述信息。

1. AI challenger2017

数据集对给定的每一张图片有五句话的中文描述。数据集包含30万张图片，150万句中文描述。

训练集：210,000 张  
验证集：30,000 张  
测试集 A：30,000 张  
测试集 B：30,000 张

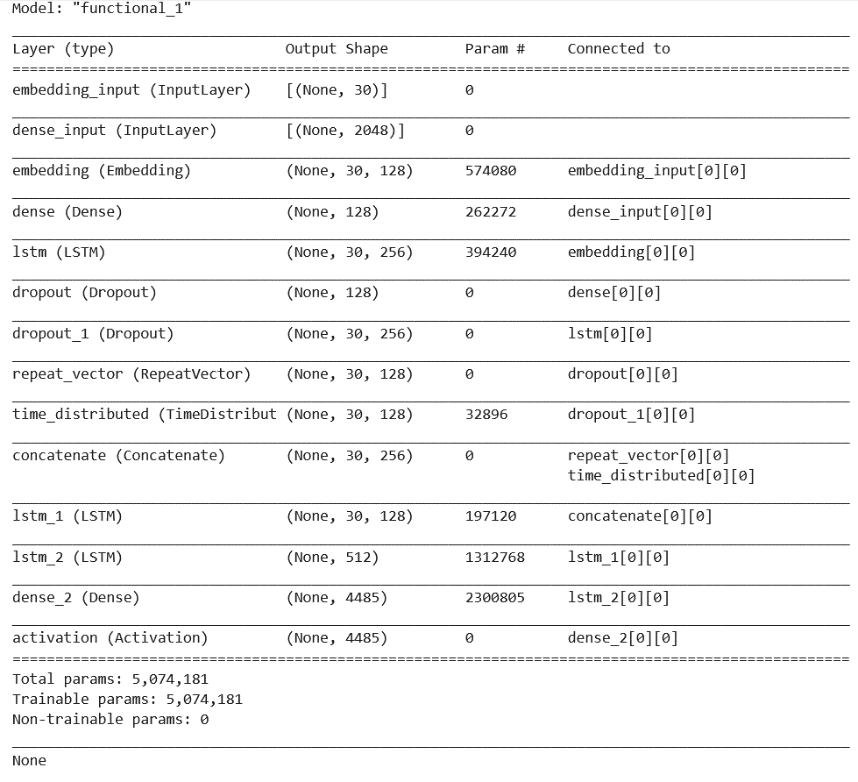
### 2.简单NIC模型实现与训练

（1）模型结构：

模型的输入数据由两个元素的列表构成，一个是文本特征，一个是图像特征。

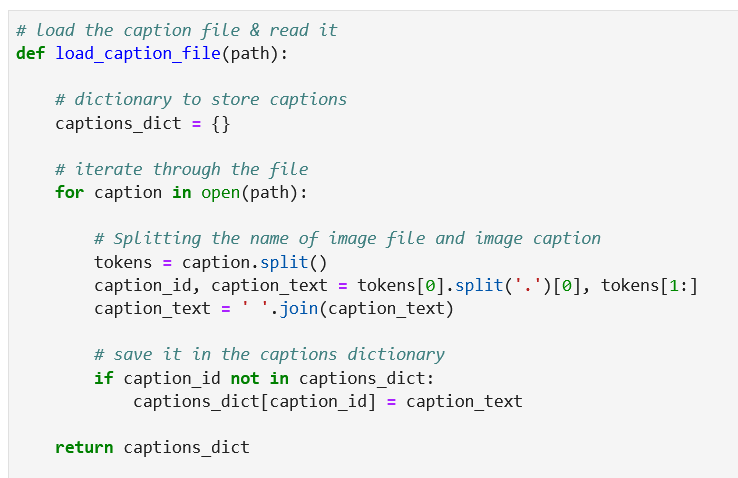
输入的文本数据是30维向量，经过Embedding词嵌入层，文本被映射到128维向量空间，Embedding层的作用是将稀疏的One-hot向量编码成维数较低的矩阵，经过LSTM层与TimeDistributed包装过的Dense层，结果为（30，128）的向量。

输入的图像数据是ResNet50提取出的2048维特征向量，经过Embedding层和RepeatVector扩充维度，得到（30，128）的向量。

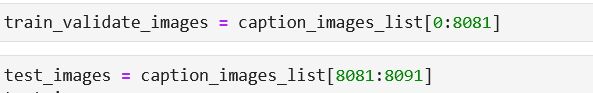
将两种特征通过Concatenate函数融合成（30，256）维特征，经过，第一层LSTM网络，第二个LSTM网络，第三个Dense层，最后通过1\*4485的全连接层输出4485维数据，维数对应预处理数据集得到的词库大小。

（2）模型训练过程：

a.数据预处理

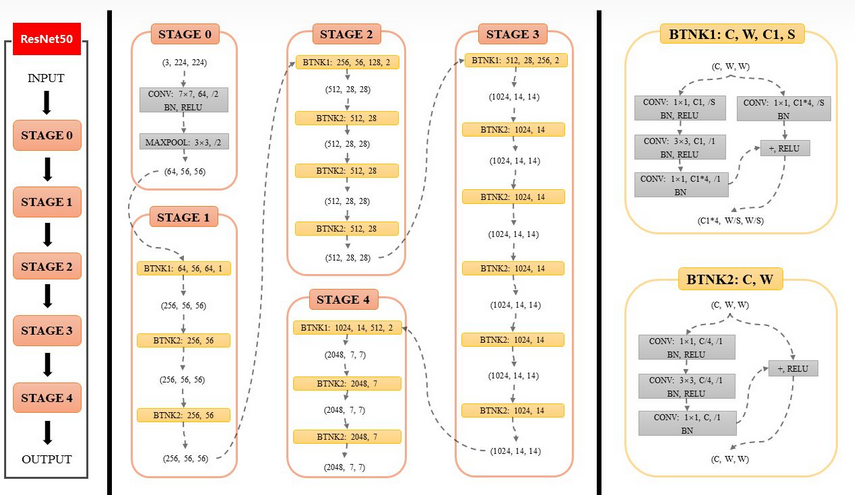
这个部分包括数据读入，caption标签分离和标签清洗。

b.测试集验证集训练集划分



c.图像特征提取

使用ResNet50模型来提取图像特征，网络结构如下：

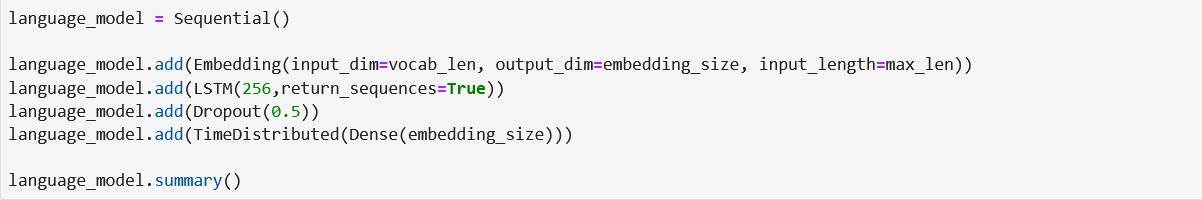
 上图可划分为左、中、右3个部分，三者内容分别如下：

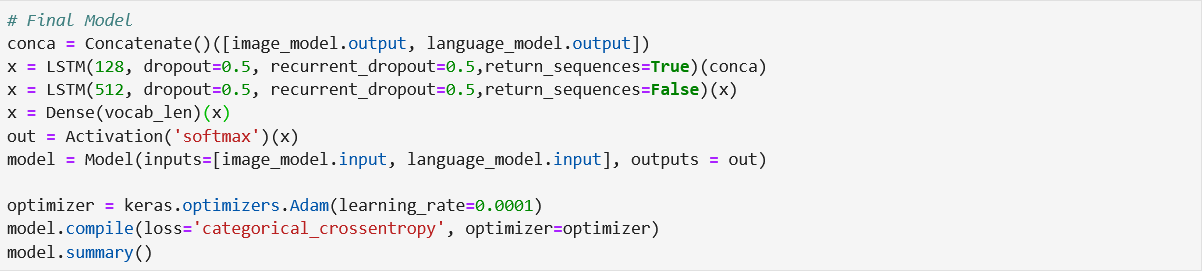
ResNet50整体结构；ResNet50各个Stage具体结构；Bottleneck具体结构

d.构建网络模型

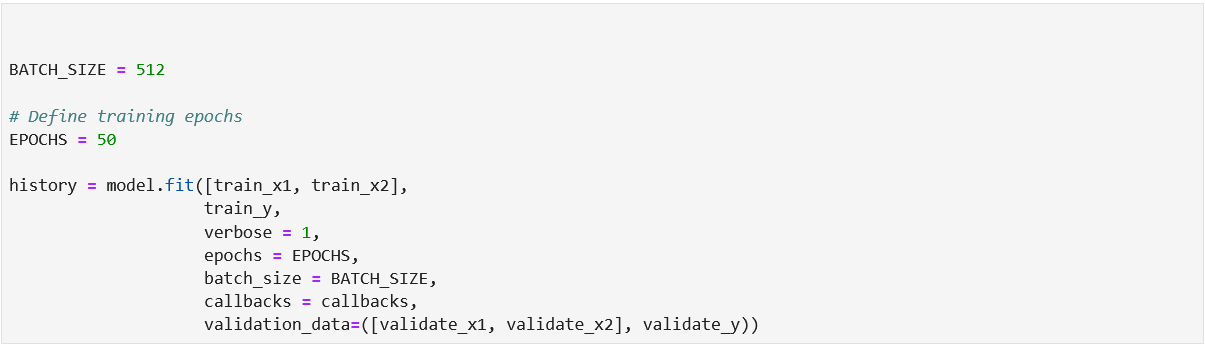
分别构建图像模型和语言模型，然后再通过concatenate函数合起来，然后在接上两个LSTM模型，最后再接上词库大小维度的dense层，通过softmax函数输出最后结果的概率。



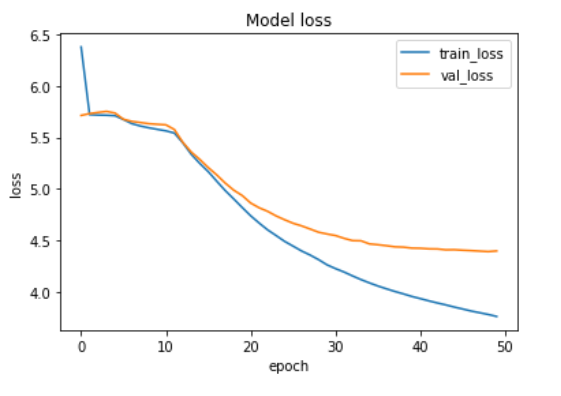




e.模型训练



f.模型评估

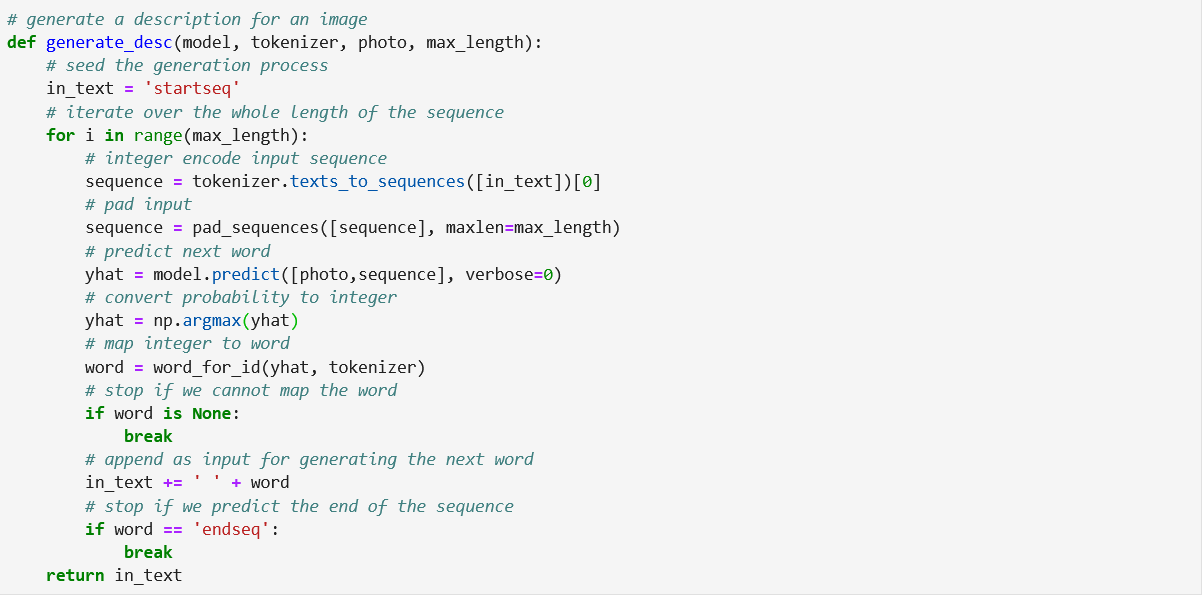


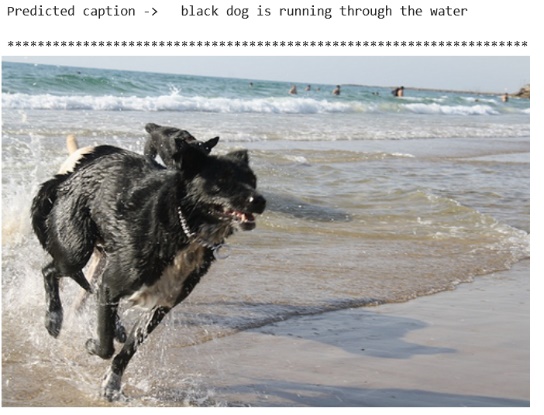
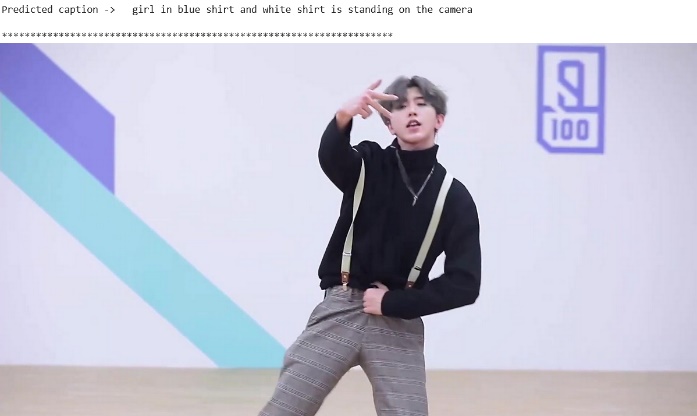
g.结果可视化

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。Beam-search将在with attention模型中使用。





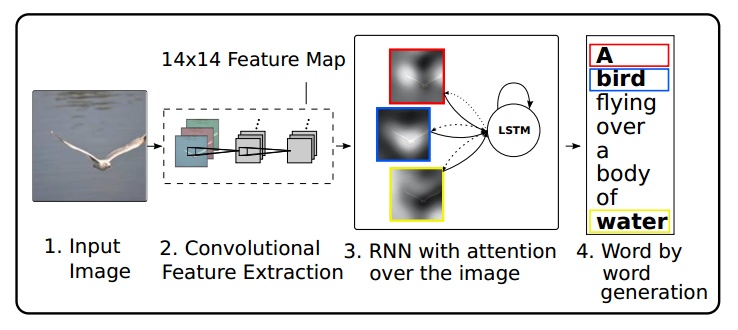
这里给出几个预测结果：

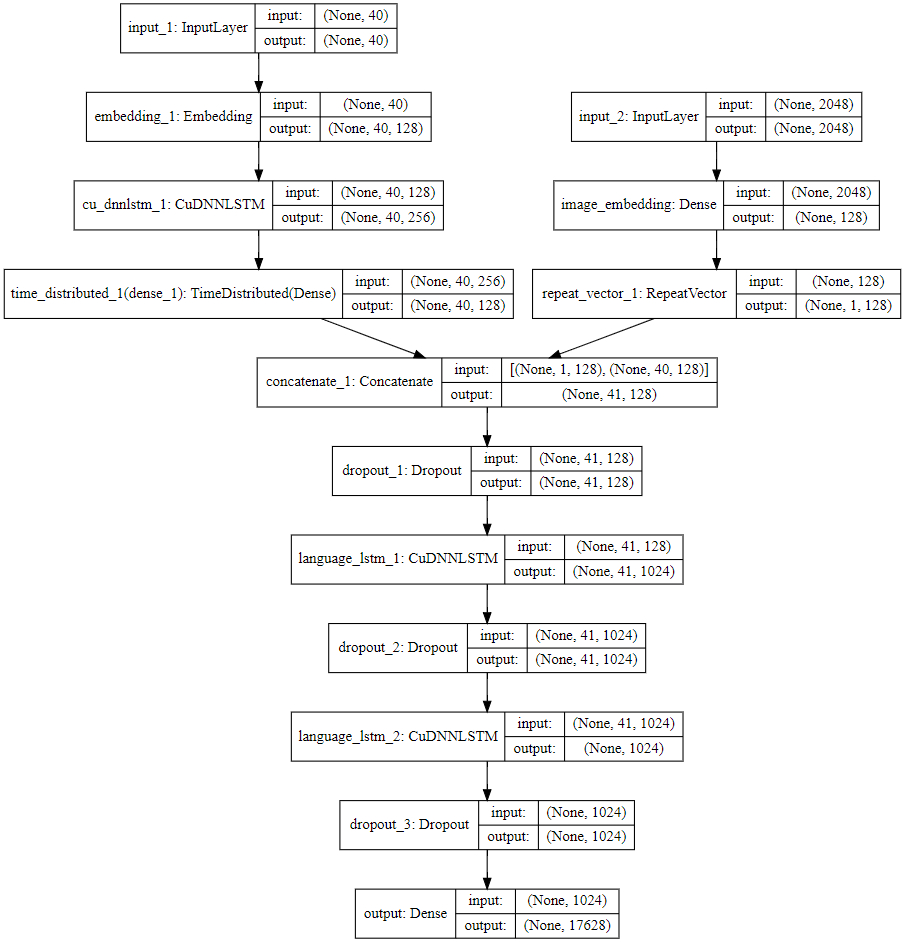
可以发现模型的精度还是不够高，由于训练数据集大小（图片和文字均有限）有限，所以实际上能够被识别出来的物体仅限数据集中多次出现的物体，并且模型预测时，能够预测出来的词语的词汇也受限于所有的文字描述中的词语。并且该模型仅仅只关注图像的整体特征，局部特征信息基本被丢失。所以模型预测生成精度还是过低。

### NICwithATT模型

该模型完成了模型构建的代码，还没有开始进行模型训练，所以暂未提交至Github

1. 模型结构





模型的输入数据由两个元素的列表构成，一个是文本特征，一个是图像特征。

输入的文本数据是40维向量，经过Embedding词嵌入层，文本被映射到128维向量空间，Embedding层的作用是将稀疏的One-hot向量编码成维数较低的矩阵，经过LSTM层与TimeDistributed包装过的Dense层，结果为（40，128）的向量。

输入的图像数据是ResNet50提取出的2048维特征向量，经过Embedding层和RepeatVector扩充维度都，得到（1，128）的向量。

将两种特征通过Concatenate函数融合成（41，128）维特征，经过第一个Dropout层防止过拟合，第一层LSTM网络，第二个Dropout层，第二个LSTM网络，第三个Dropout层，最后通过1024\*17628的全连接层输出17628维数据，维数对应预处理数据集得到的词库大小。

（2）模型训练：使用40个G的AIchallenger2017数据集，暂未完成

## 三、StageⅡ计划

第二阶段主要任务分为2个：

1.实现基于Transformer的图像描述模型，并对比与之前2个模型（CNN+LSTM、CNN+LSTM+ATT）在相同数据集上的效果。

2.使用更大规模的数据集（如AIChanllenger2017、MSCOCO、Flickr30k）进行模型训练。

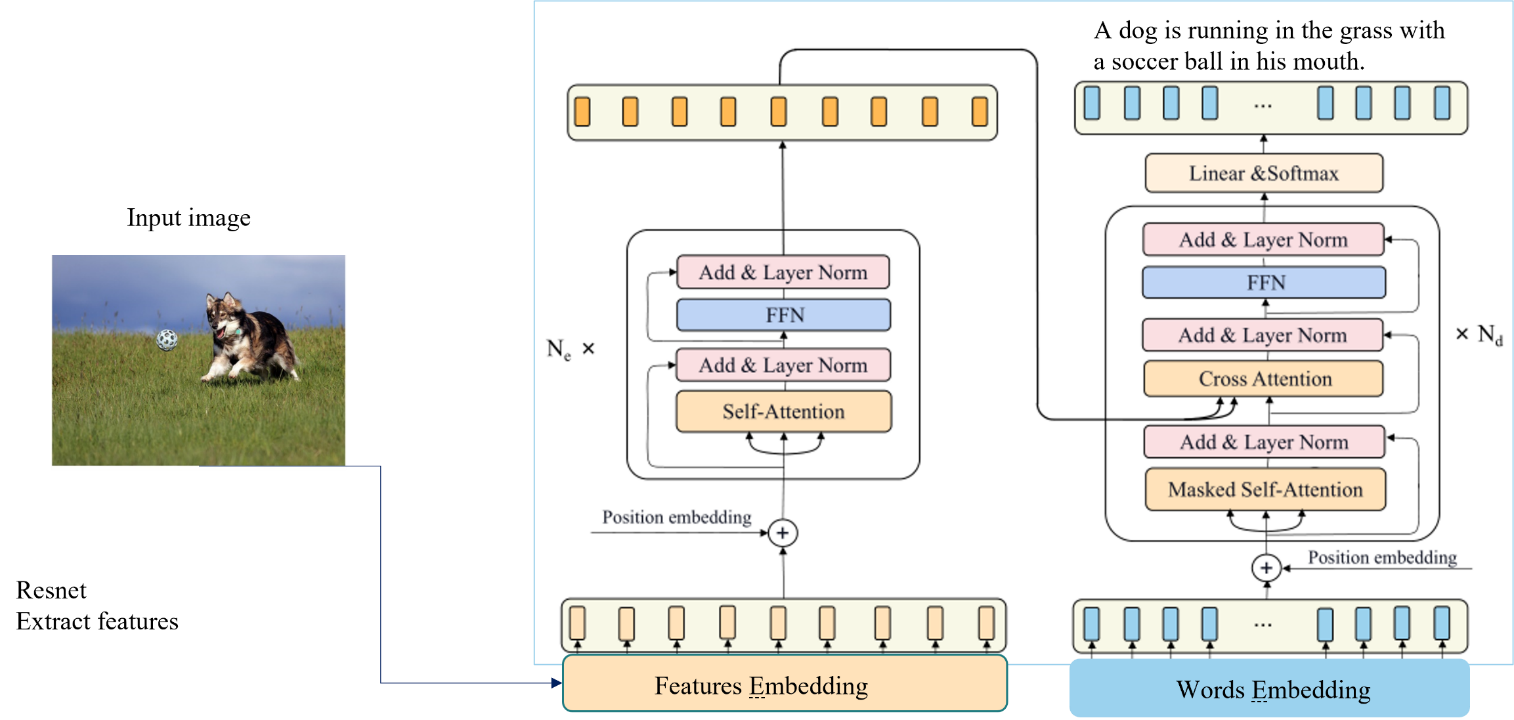
项目展望：

最终将模型应用：APP或者网页前端或者客户端。由于是个人项目，自己在这些方面知识的欠缺以及时间计算资源有限等问题这部分完成难度将会比较大。

或者将图像描述和生成网络结合，进行根据图片进行AI作画/作诗等。

# Stage Ⅱ

## 模型构建（CNN+transformer）

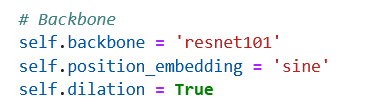
1. 模型结构：

模型主要由两个部分组成：encoder+decoder。

Encoder的输入是图片特征构成的Features Embedding(d\_model=512)，这个特征由Resnet101提取得到。然后在这个输入之后加入Positional Encoding(位置编码)，然后是N(N=6)个编码器层（Encoder layer），每个编码器都有着相同的结构：第一层Multi head self-attention(N\_head=8)，然后是FFN，两个子层都添加了一个残差连接+layer normalization的操作。编码器将输出的向量作为解码器Cross Attention的输入Q。在解码器也有一个相同的输入，这个输入是对图片描述的词转化为特征向量，然后是相同的解码器结构：一个Masked Multi head self-attention(N\_head=8)层，第二个子层是Multi head self-attention(N\_head=8)，最后一个FFN，每个子层同样也用了residual以及layer normalization。模型的输出由Decoder的输出简单的经过softmax得到。

1. 模型实现：
2. 特征提取：backbone模块：

Config：

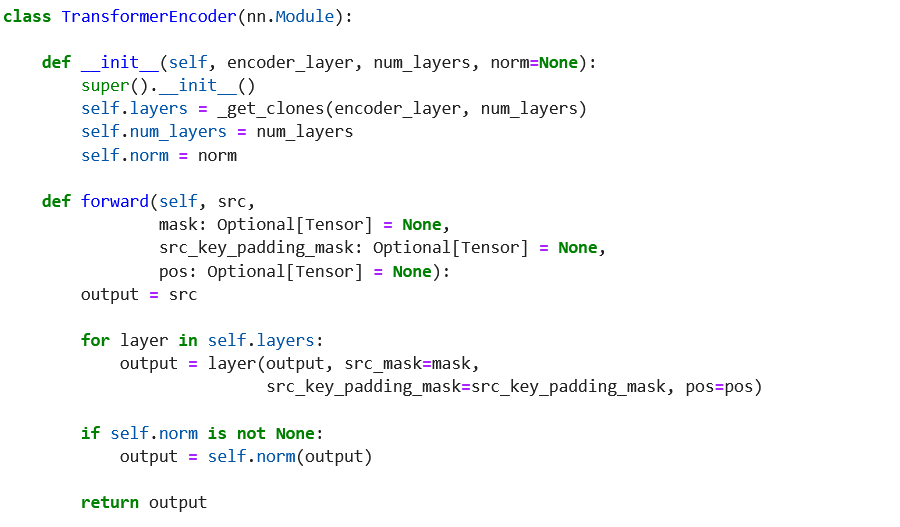


使用预训练Resnet模型：

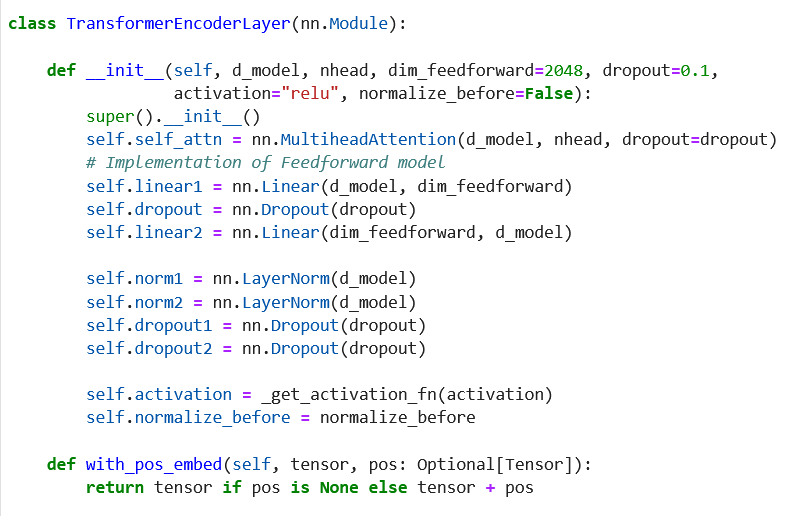


1. Transfomer：

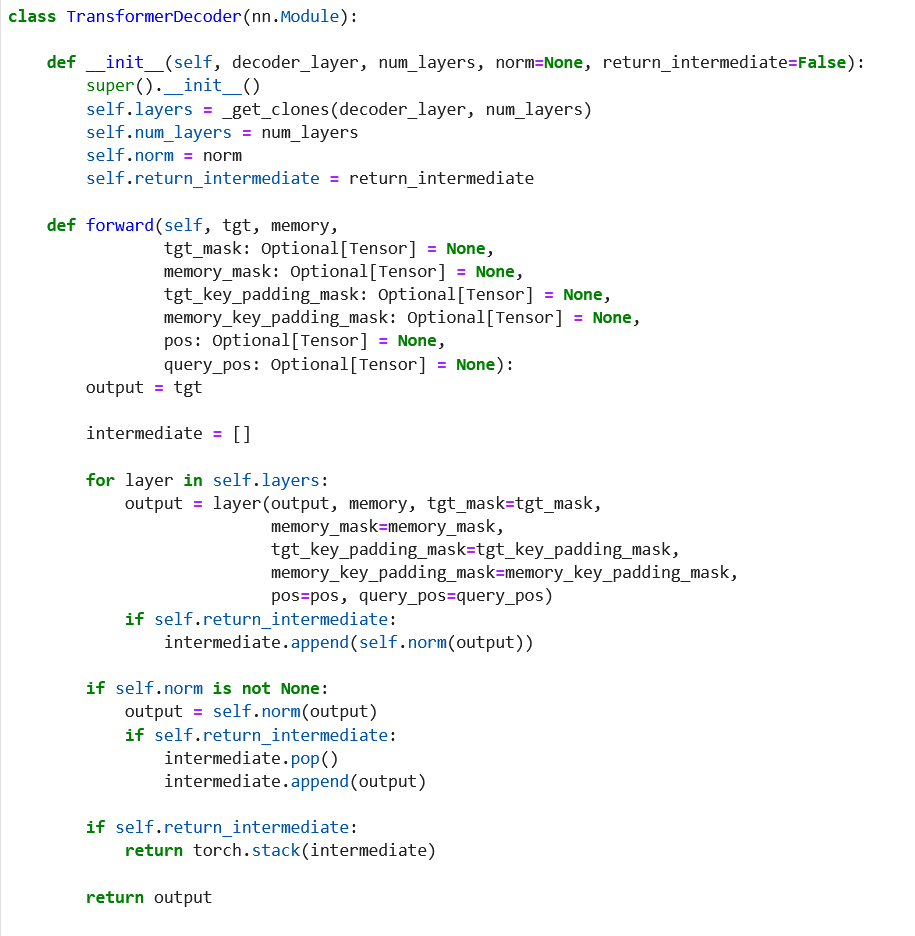
Encoder：



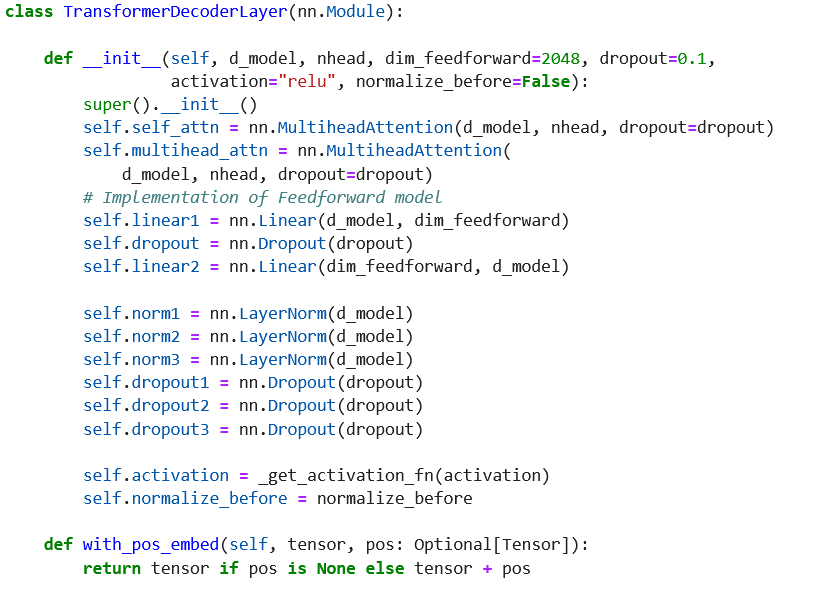
Encoder-layer：



Decoder：



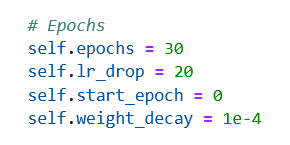
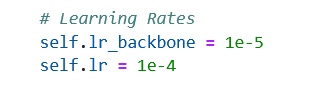
Decoder-layer：



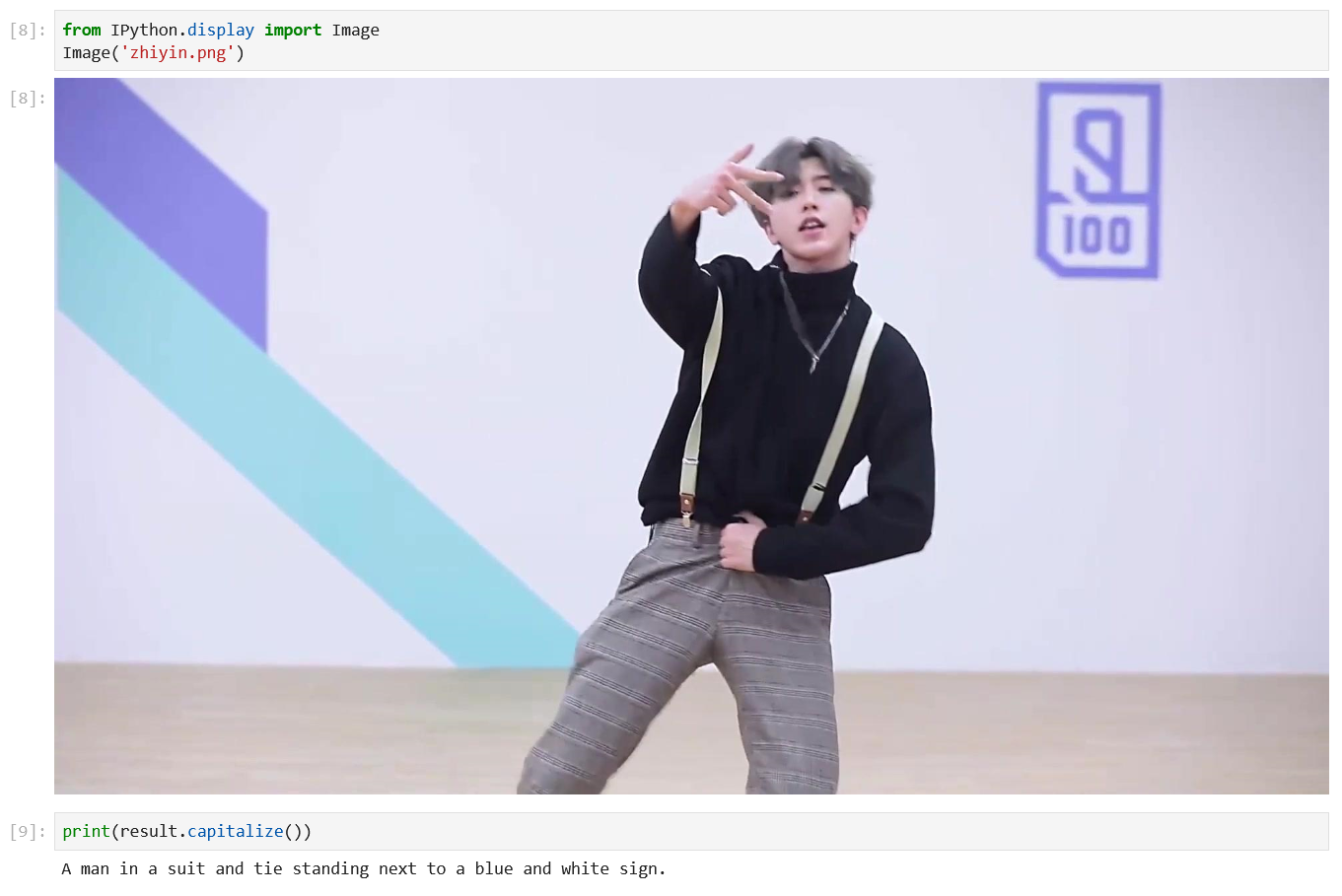
1. 模型训练：
2. 数据集：MSCOCO2017

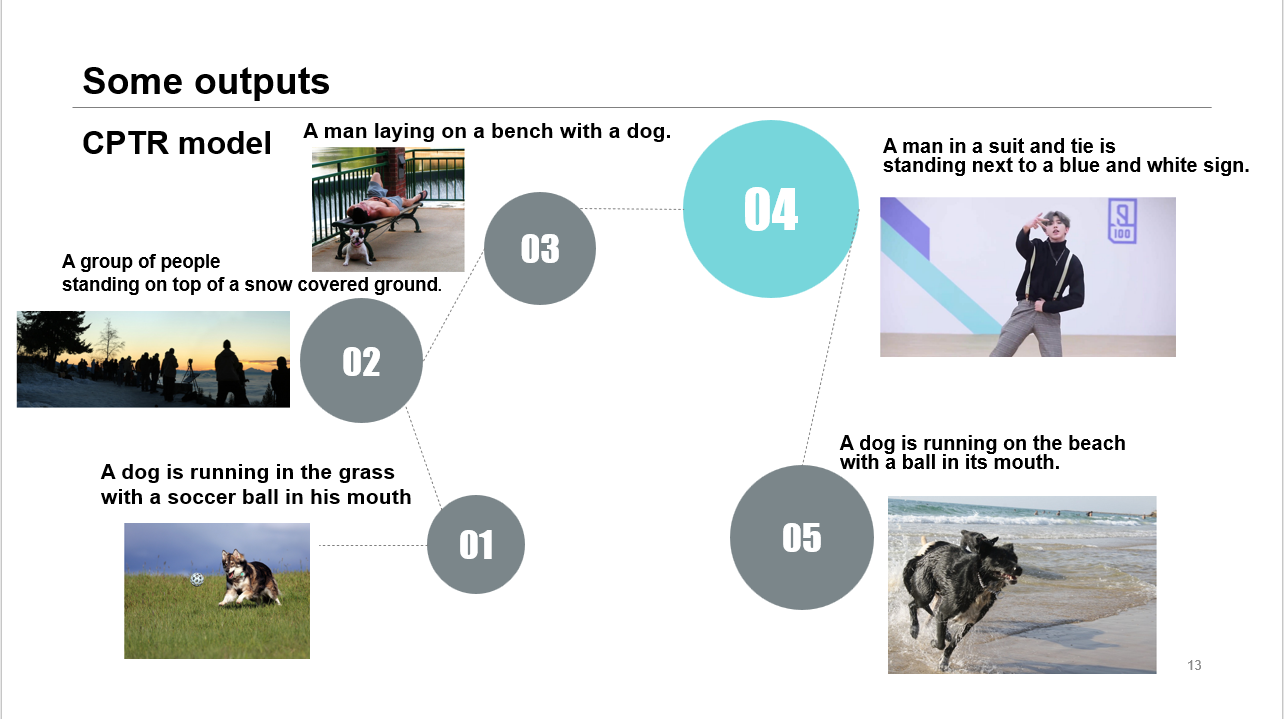


1. 模型训练config

1. 模型预测结果展示：





## 二、模型对比

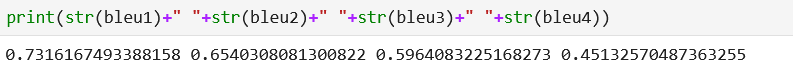
1. 评测指标：BLEU

BLEU (其全称为Bilingual Evaluation Understudy), 其意思是双语评估替补。所谓Understudy (替补)，意思是代替人进行翻译结果的评估。尽管这项指标是为翻译而发明的，但它可以用于评估一组自然语言处理任务生成的文本。

在NLTK中, 其允许用户显式指定不同的N-grams的权重以便来计算BLEU的值。这使得用户可以灵活的计算不同类型的BLEU值, 比如独立的BLEU或者累积的BLEU。

2. 模型对比：

CNN+LSTM



CNN+Transformer

