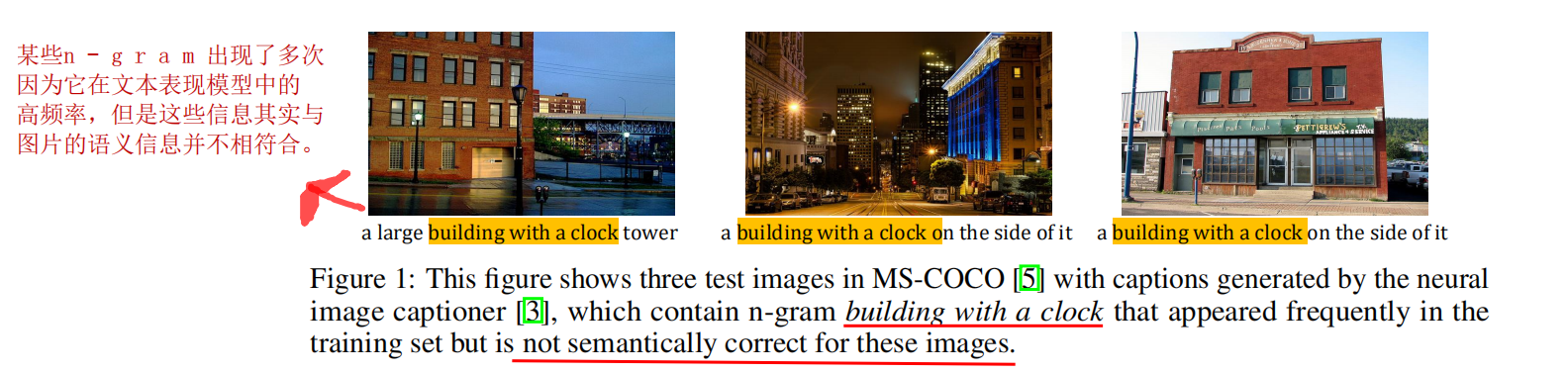
1. PekingU intern papaer analysis

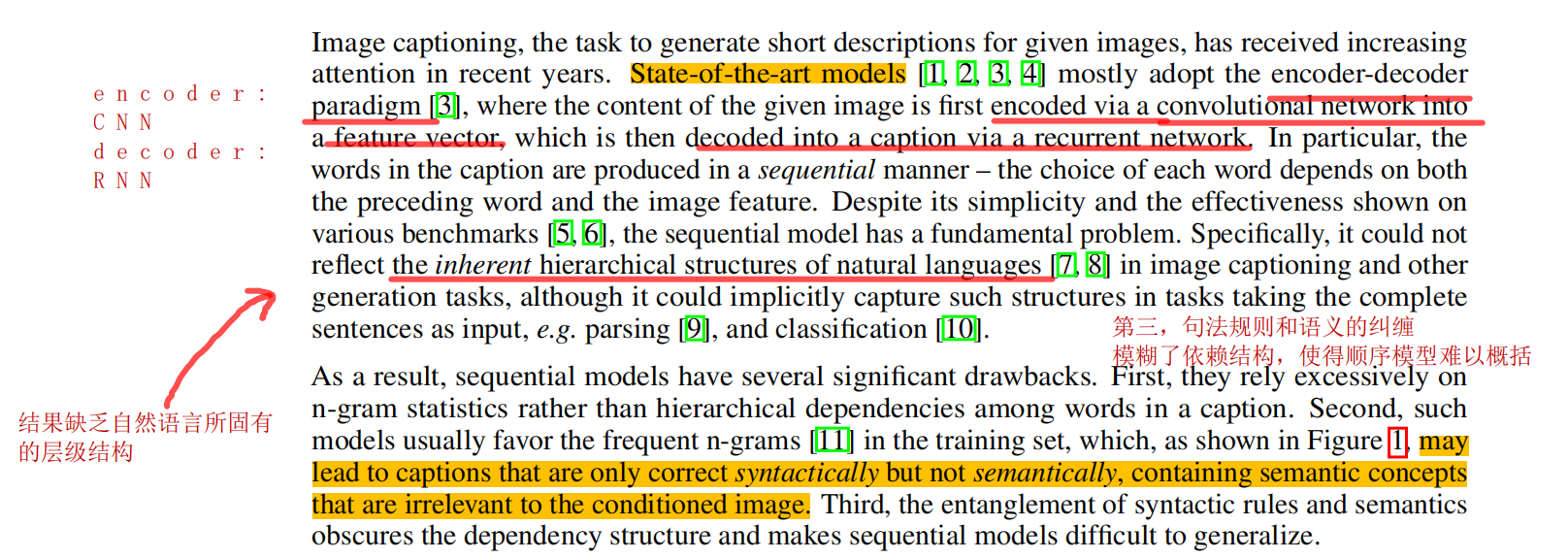
*ONE: Image caption*

A Neural Compositional Paradigm for Image Captioning

它打破了这几年做image caption所使用的框架encoder-decoder。它提到使用encoder-decoder框架生成的caption是序列生成的（逐个单词的生成），更符合n-gram规则，但是失去了句子内部的结构，比较僵硬。同时这个生成方法将caption的语义信息和句法信息混在了一起，只是在形式上符合句子的语法规则。容易出现问题，如下：

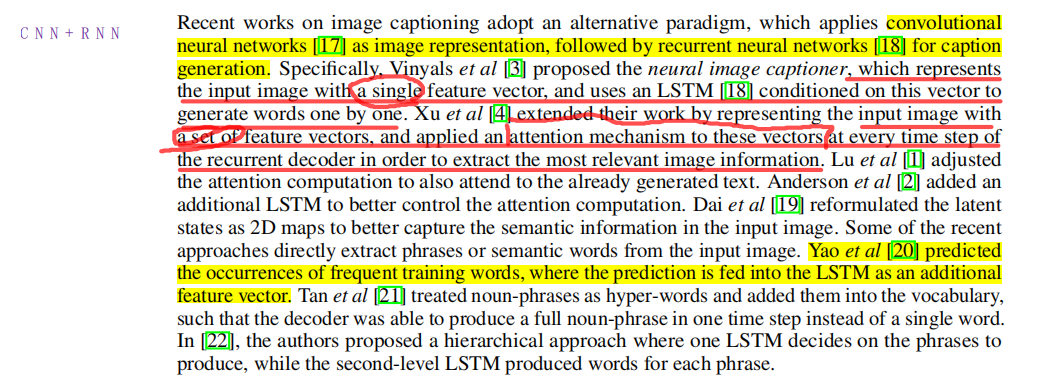


**1.问题点原文与总结：**



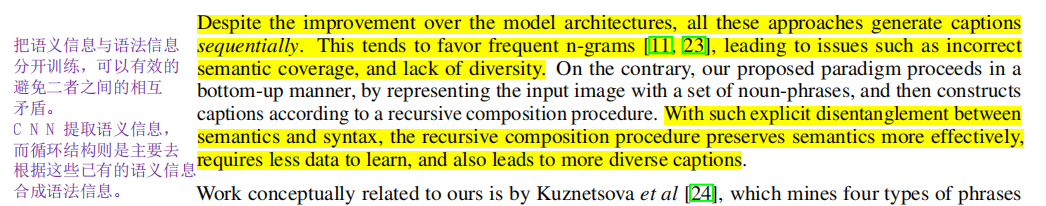
使用encoder与decoder结构，直接利用CNN 提取的feature vectors 作为encoder,利用监督学习的方法使用RNN做decoder生成图片描述。这样做的缺点有很多：

1. 贪心算法，不好衡量生成的句子与之前句子之间的差异，使用n-gram会让decode尽量使用freq比较高的n-gram以降低loss，从而使得生成的描述句式单一。有些词汇大量使用，因为过度在意句子的顺序性（n-gram），缺乏自然语言固有的层级结构，也就是句子的构成规律。
2. 训练时候不可避免遇到了语法规则与语义的纠缠。想要提高语义正确，必须使用n-gram里面现有的词汇与表达，这样降低的loss,但是图片的语义难以表达。
3. 所以许多句子都是句法正确，语义信息不好，而且生层的句子缺乏多样性。
4. **历史研究与发展原文与总结**



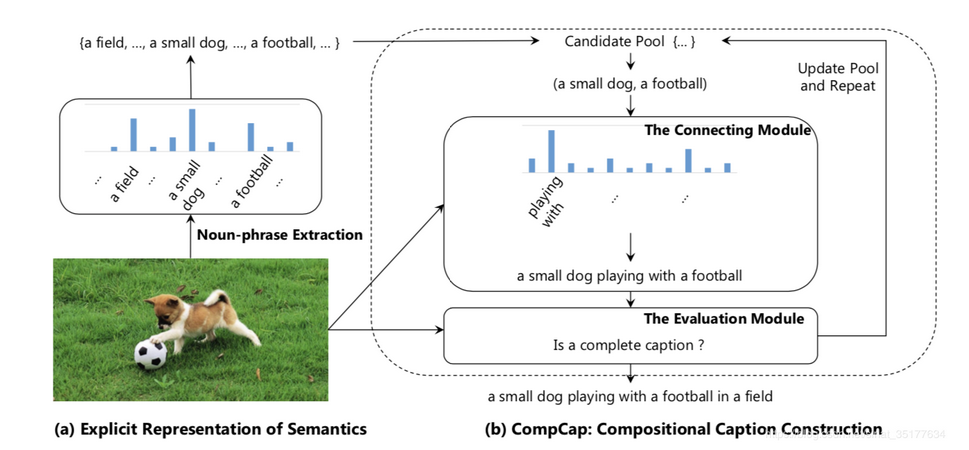
其实有许多模型做了改善，例如XU，它不是将一张图用CNN变为一个vector，而是使用CNN提取多个vector（我理解是FC层之间的output）,使用注意力机制（Attention）在不同时刻把不同的feature vectors的z作为input输入LSTM进行decode。也有人如（Anderson）使用2层LSTM,其中第一层用来控制attention(这篇论文在encode的时候也是这样做的）等等。但是这样做只会让decode的结果更准确，没有解决生成句子diversity的问题。

1. 本论文的idea:



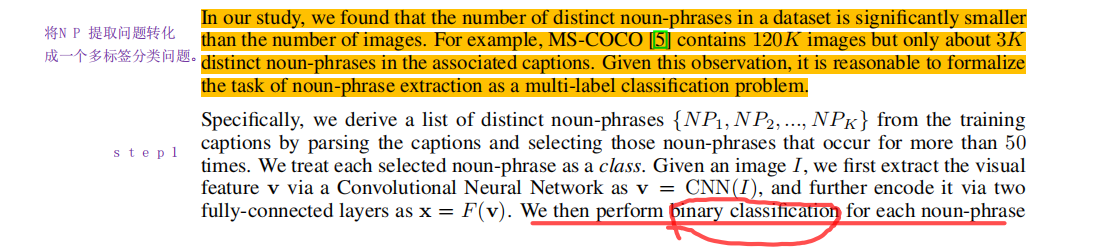
对于一张image，先提取它的语义信息，即它的短语（NP）。再使用这些NP来进行句法分析，生成完整的句子，分成两部分：C-Module，E-Module，分别用于连接两个NP，然后判断是否是完整的句子，如果是完整的句子则直接输出，如果不是更新这个NP集合，将连接后的NP加入，删除用于连接新的NP的NP，再迭代这两个module，直到输出完整的句子。

结构如下：

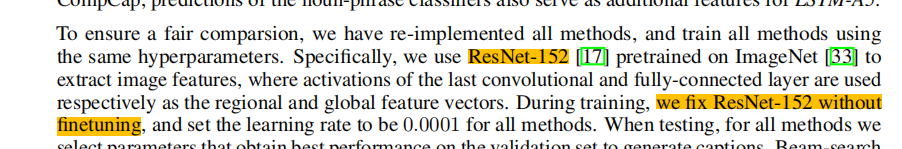


首先训练CNN做特征提取时候：

使用名字词组作为output进行监督学习，因为作者发现MS-COCO中虽然images的个数很多，但是noun phase的个数很少：

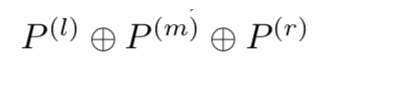


在设立作者使用的是ResNet-152，没有微调（fine-tuning）

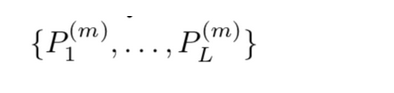


把提取NP的任务当成一个分类任务，对于每个NP设计一个分类器：  
IMG_256  
其中I为image，x为image通过CNN处理再使用两层FC得到的特征，**输出这张image包含这个NP的概率，取概率最大的几个作为image的NP集合**。实验中作者选取的是top7.

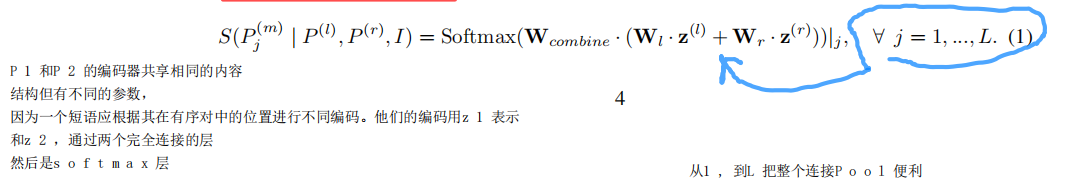
使用上述的NP集合生成完整的句子，作者称其为CompCap。在C-Module模块，对NP的集合中的每对NP进行连接，并且有不同的顺序，连接后的新NP：



Pl与Pr为左右的NP，Pm为连接词。其中连接词也是使用句法分析来得到(这里作者直接使用是Standford的NLPtoolkit这个开源库实现的《提取连接词》)：

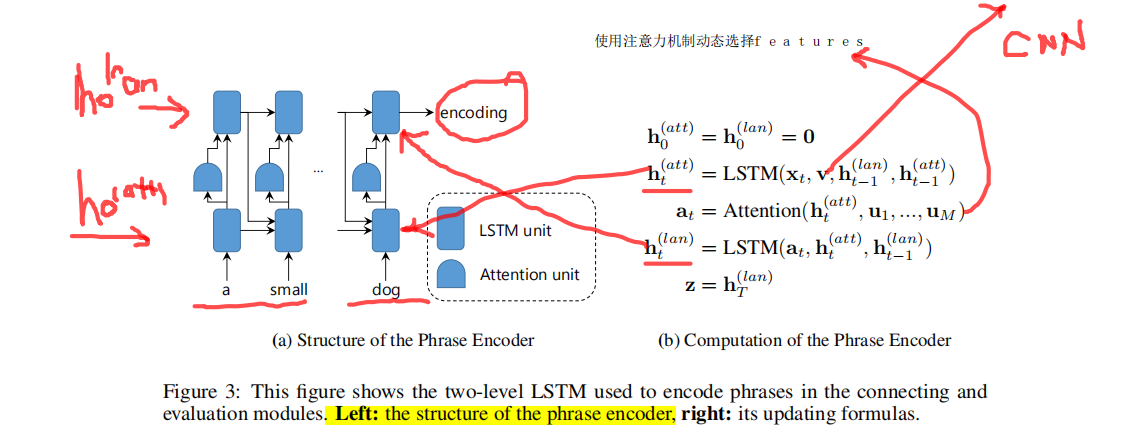


同样将得到Pm的任务当成一个分类任务：



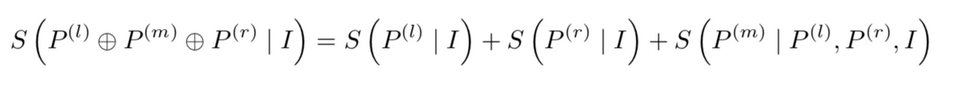
其中zl和zr是编码pl和pr短语的特征，W为参数，输出使用这个pm的概率，选择概率最大的连接词连接两个NP，作为新的NP——Pnew。

在编码是同样使用2层LSTM与注意力机制：



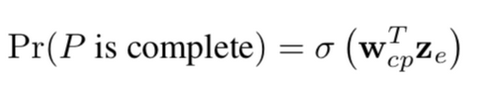
使用两层的LSTM，其中v为image整体的特征，u为image的局部特征，底层的LSTM来得到对u的attention权重，使用这个权重和u来得到加权的特征a，再输入到高层的LSTM得到隐层表示，最后一个隐层表示当作编码z。

上述过程就得到了新的NP——Pnew，但是我们一次只需要一个Pnew，则对连接的NP进行打分：



为得到三个NP的概率的和，最终选择这个总概率最大的NP为这一轮的Pnew。

1. Module用于判断这个Pnew是否为完整的句子：

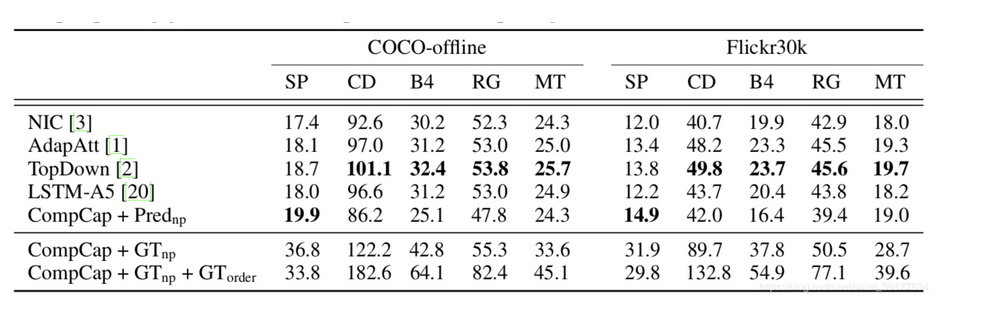


编码Pnew得到z的方式和上述相同，最终得到是完整句子的概率，设置阈值，大于阈值则输出，小于则将Pnew加入NP的集合，删除Pl、Pr。再进行CompCap过程，直到输出完整的句子，实验表示一般迭代2-3次就可以输出。

在LSTM的decode部分，作者使用是Beam-search，而不是greedy Search，我估计也是为了多样性吧，得到更多的结果（所以我觉得这一步对说明模型的性能不太好。）

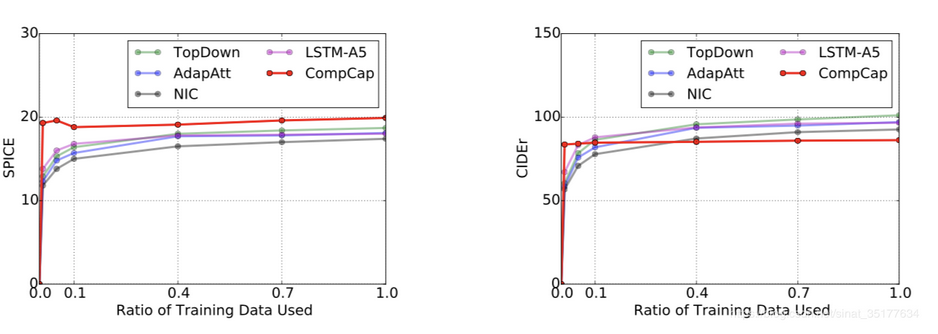
论文中选择beam-search的参数是3，也就是保留每一次最大的3个output作为下一次的input。

实验结果如下：

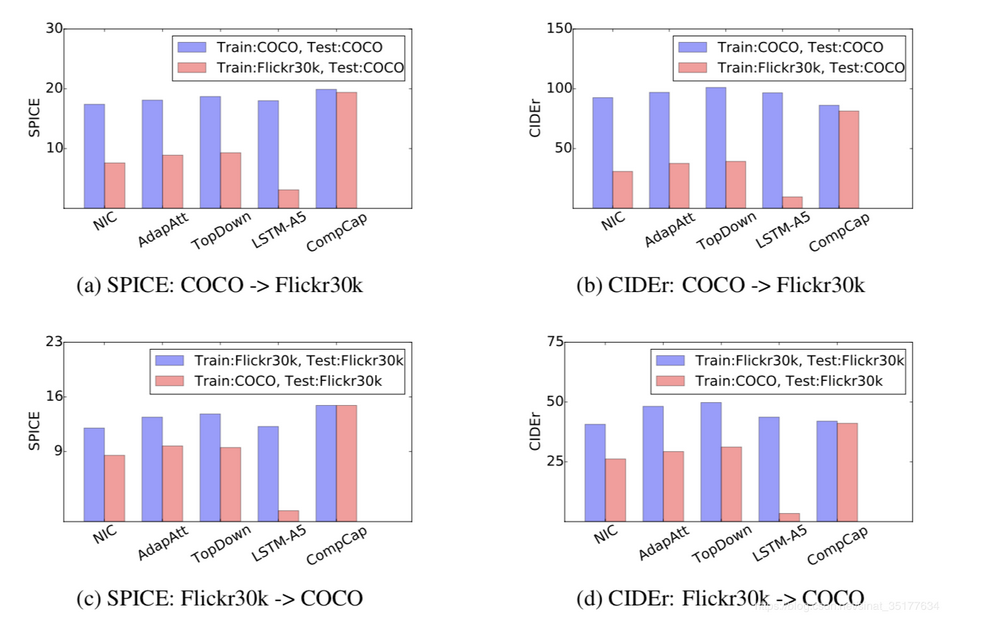


可以看到方法在大部分指标中虽然不如up-down方法，但是SPICE指标最高，这个指标更代表了caption的语义信息，而其他的指标更符合n-gram。并且如何使用真实的np集合，和真实的NP顺序得到的效果很高。

同时这个方法在使用较少的训练集时同样有效：



方法使用较少的data就可以得到较好的效果，而其他的方法需要大量的训练数据。并且在跨数据集的实验上也取得了较好的效果：

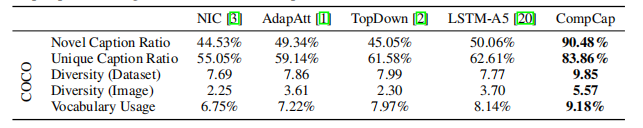


因为是语义与语法分开的，与一个训练集中的描述关系没有之间那么紧密。所以泛化好，不会出现一个训练集work,另一个训练集不work情况。

在一个数据集上训练，另一个数据集上测试效果远高与其他的方法。

这两个结果都归功于将语义分析和句法分析分开，虽然数据少，虽然数据集不同，但是得到NP大体相似，并且句法分析并不关心语义的信息，只是用于将NP构成句子，因此虽然数据集少，虽然跨数据集依然有较好的效果。

并且在生成caption**多样性方**面也不错：



可以看到可以用不同的NP顺序来生成不同的caption，同时还可以用不同的NP集合来生成不同的caption。

综上，虽然在大部分硬指标上不是最好的，但是它的泛化性，多样性更强，和数据集的关联较少。并且提出了新的image caption框架，给我们以启发，可以不是用encoder-decoder框架，将语义分析和句法分析分离可能更好。

**论文存在的疑惑：**

关于Model E判断句子完整性这一点，使用LSTM进行encode之后使用一个activation funcation之后output一个结果，这个如何训练呢？是给机器看完整的句子与生成的不完整的句子，像GAN那样去train吗？作者没有说清楚。