**第二次实验报告**

**所选任务：Task7(HumanEye)**

* 任务内容：从视觉信息中生成文本信息
* 所选数据集：COCO，训练集大小：13G，验证集大小：6G，总共19G

**第一阶段**

* 论文调研：将该任务视为image caption任务，调研了该领域的相关论文，决定从最基础的show and tell模型开始实现，在第二第三阶段不断对模型进行优化改进。
* 数据预处理：该数据集的caption是json格式，预处理部分将有效信息抽取出来，剔除冗余信息，生成一个字典，并建立一个单词与数字索引的双向字典(实际实现方式为两个字典)，并将出现次数小于5的次数忽略，加入关键字< ukn >，用于编码和解码阶段，最后再将图片名与字幕组合起来生成一个tuple，方便data loader的构建。
* 构建dataset类和data\_loader：继承torch.utils.data.Dataset类，重新写 \_\_init \_\_，\_\_get\_item\_\_，\_\_len\_\_三个函数。\_\_init\_\_函数用于初始化数据集的各种参数，\_\_get\_item\_\_函数用于取数据集里的一个元素，\_\_len\_\_函数则是返回数据集的大小。由于数据集过大，无法一次性将图片读入内存，于是在\_\_get\_item\_\_函数里面单个读入，最后返回图片，同样的，在\_\_get\_item\_\_函数里还进行了对字幕的处理，将原来的单词转化为数字索引，并加入开始标识符和结束标识符。最后，还重写了collate\_fn函数。由于字幕长度不一致，所以无法转化为tensor，于是对此进行了补0操作，并转化为tensor类型返回。

**第二阶段**

* 更新数据预处理方式：更新preprocess.py文件，里面包含Vocabulary，将原来的双向字典，即两个字典，打包成一个类，使得程序更加健壮，不至于把变量裸露在外。同时，将Vocabulary类变量vocab和caption与image\_name的组合data保存到磁盘，使得数据预处理步骤和模型训练分开，为模型训练节省时间。
* 增加network.py文件：里面包含模型的主要网络结构，完全复现show\_and\_tell模型。Encoder部分为ResNet152(去除最后一层fc)+AvgPooling+Linear+BN；Decoder部分为LSTM结构，inference采用贪心算法，每一步(单词)都选概率最大的那个词。
* 增加train.py文件：训练模型的文件，先对数据进行预处理(若已生成data.pkl和vocab.pkl则可注释这句话)和数据增强，再生成data\_loader，encoder，deocder，最后开始训练，打印信息并保存模型。损失函数为交叉熵，优化函数为Adam。
* 增加inference.py文件：用于测试时生成字幕的文件，每次输入都是一张图片(图片路径可在配置文件里改)，首先加载数据和模型，接着将数据送入模型，使用贪心算法得到每个词对应的索引，最后将索引转化为具体单词并输出结果，即一句话。若在非终端跑该代码，还可以显示原图进行比较。
* 增加show\_and\_tell.yaml文件：该文件为配置文件，保存各种数据与模型的路径，还有模型和训练时的参数。在不同机器下跑该代码需要修改配置文件的参数以适应不同环境。