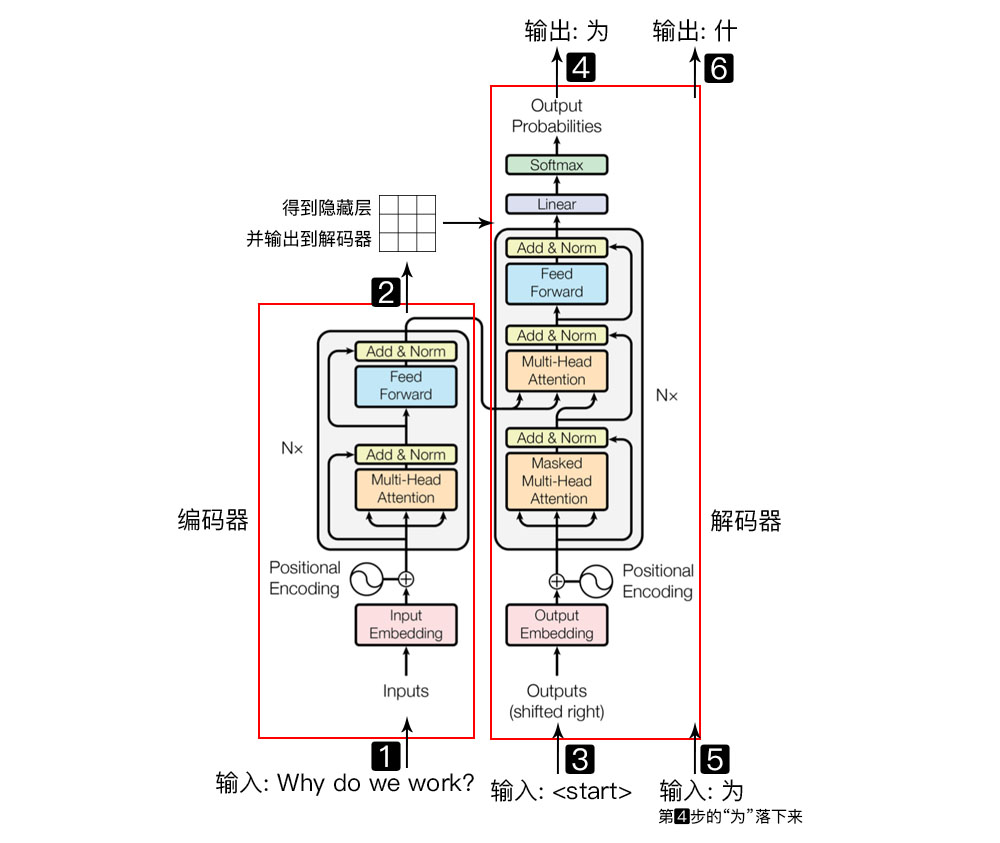
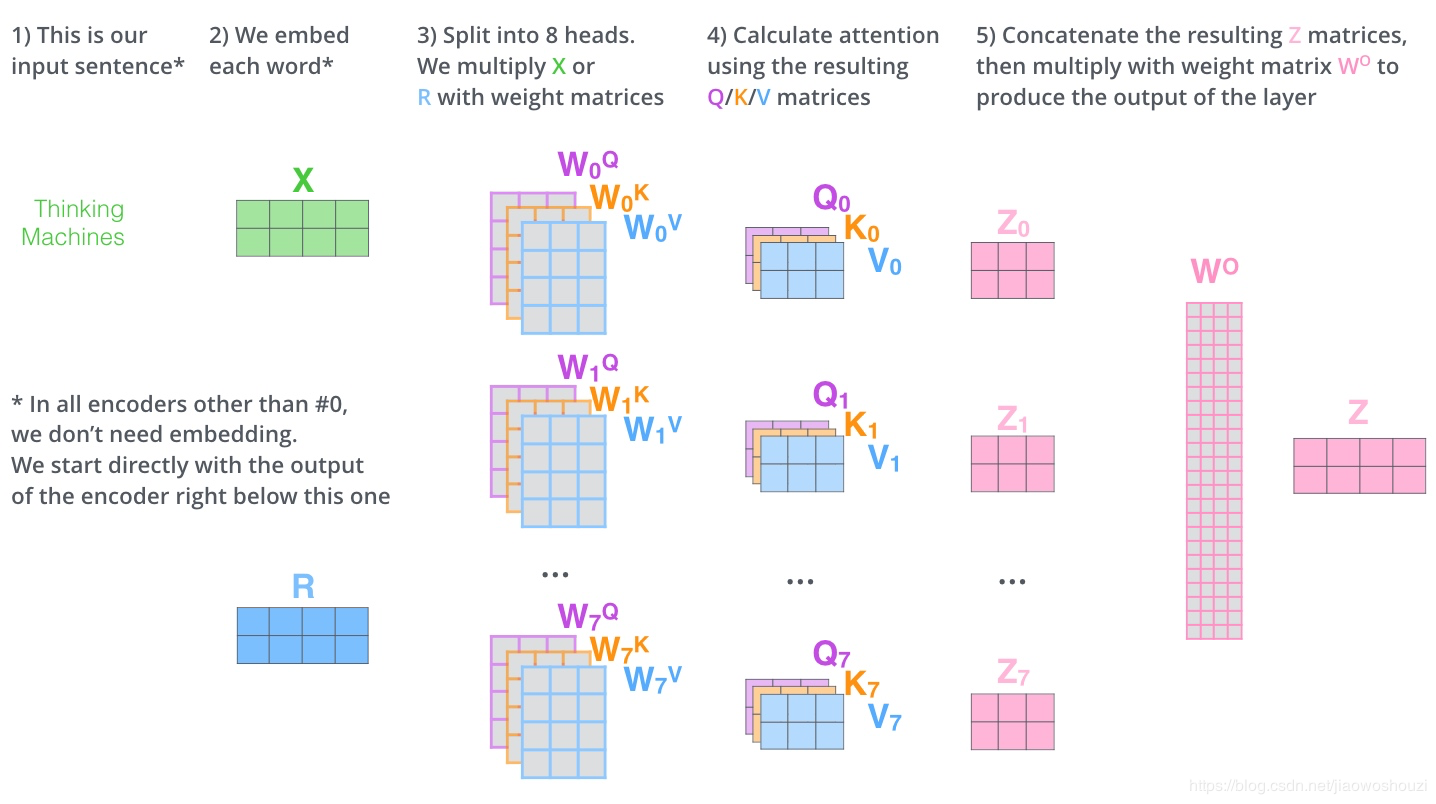
# SRTP.PICO.梅磊

## 一、将整个句子分成 PIO 三类

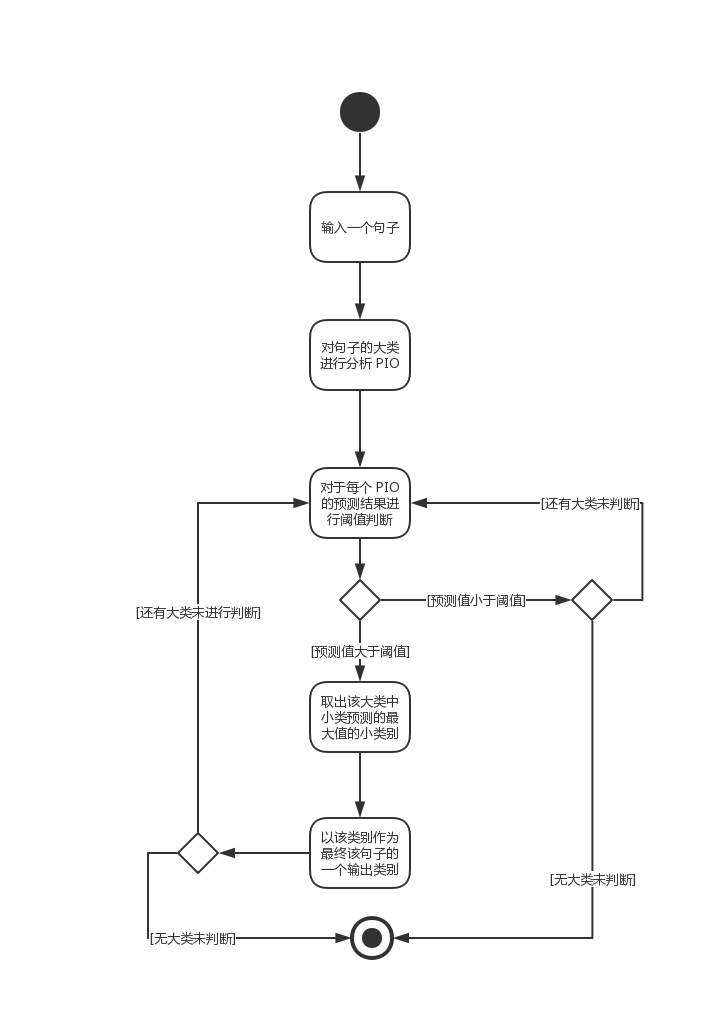
1. 本次实验中，我的实验内容是将训练集中的医学句子初步分类成 PIO 三个类别中。由于整个医学文献的句子是很多的，我们每位同学在从整个文献中收集 PIO 训练语句的时候，都比较倾向了选择不同的句子，划分到 PIO 三个类中。因此，这就导致了我们一个训练句子的多标签情况很少，对我们的后续训练带来的一定的影响。
2. 同时，由于我们的数据集一共有六位同学一起构建，所以，其中遇到了很多的差错问题。比如：有些同学将医学文献编号 pubmed 拼写成了 pembud ；将 disease 拼写成了 disaese ；或者在标签类别中有一些空格等等。这些问题，都导致了我们前期预处理训练集，出现了困难，但是我们也在一定时间后修复了这些 bug
3. 多标签分类的解决方案
   1. 首先想到的是模型的选择，一种是采用在一年前立项时采用的 BP 神经网络以及 KNN 的机器学习算法。 但是，由于技术革新过快，我们决定采用一种较新的双向长短期记忆神经网络 BiLSTM 以及一种在 18 年较为火热的 Bert 模型来完成多标签分类问题
   2. 对于 BiLSTM 网络，有 ZJY 同学来完成补充。。。
   3. 对于 BERT 模型
      1. 什么是 BERT
         * Bidirectional Encoder Representation from Transformers
      * 
      1. 编码器部分 (从输入的原始句子到中间隐藏层)
         1. 首先对原始的输入句子做一个 embedding，将一个英文的句子映射到一个指定维度的向量空间上去
         2. 利用 transformer 模型中的 position encoding 操作，从而使得语言中的位置关系可以被后续的网络识别出来
            * 其中, 表示单词处于整个句子中的位置， 表示词向量的维度
            * 最后可以得到一个 维度大小的矩阵，这个矩阵直接与每个句子所得到的嵌入矩阵做一个元素叠加即可
         3. 利用**多头自注意力机制**，训练得到减少维度的字向量
         * 
           + 对于每一个头，随机生成一个 权重矩阵
           + 将输入的矩阵与之分别点积，获得三个新的注意力矩阵
           + 对 计算每个字对应整个句子中所有字（包括本身）的相关性程度
           + 对 个头，每一个都重复上述的计算，可以获得 个结果矩阵
           + 将 拼接然后与一个参数矩阵 进行点击，使得输出的句子矩阵的维度大小不变
           + 从而获得了所需要的自注意力矩阵
         1. 将上述的输出结果 矩阵与之前的输入 做一个残差链接，使得梯度训练可以向初始层传播
         2. 最后做一个前向反馈神经网络，也就是对上述的多头自注意力机制和残差连接的重复
      2. 解码器部分
         1. 输入一个原始的句子
         2. 对其中的每一个字按顺序做 MASK
         3. 让整个上游网络来预测当前被掩盖的字的值
         4. 输出预测被掩盖的概率值
         5. 将得到的输出值重新带入句子，掩盖下一个字
         6. 重复 2 ~ 5 的循环
         7. 得到该句子每个字的概率输出
      3. 我们所做的是在上述的 BERT 的预训练模型的前提下，再额外添加了一些网络层，使之可以输出到我们想要的分类的维度上

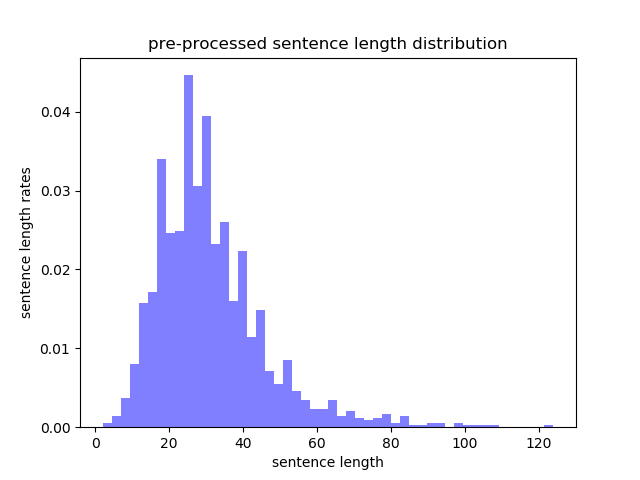
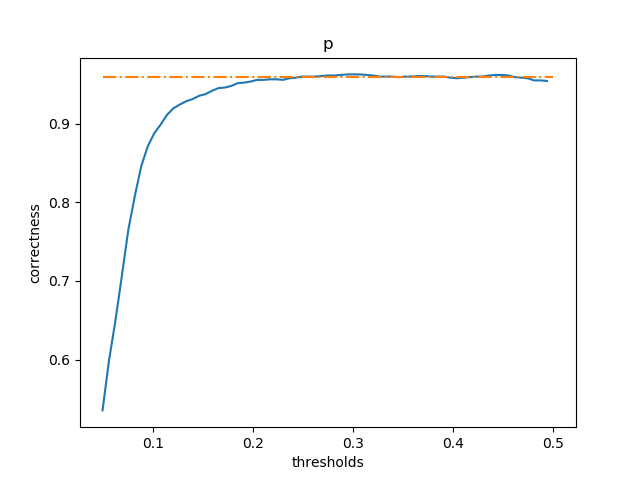
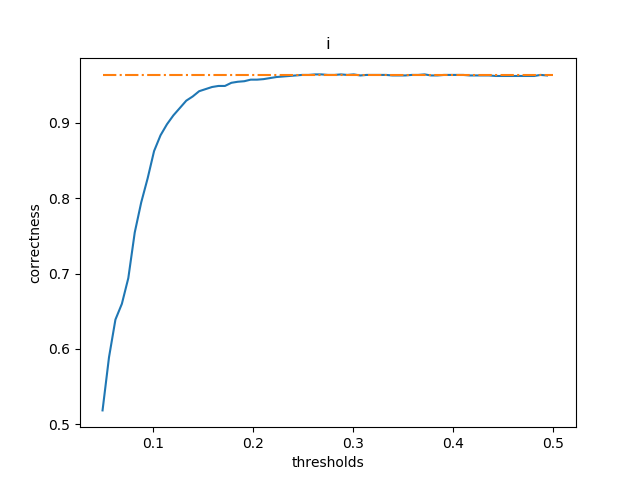
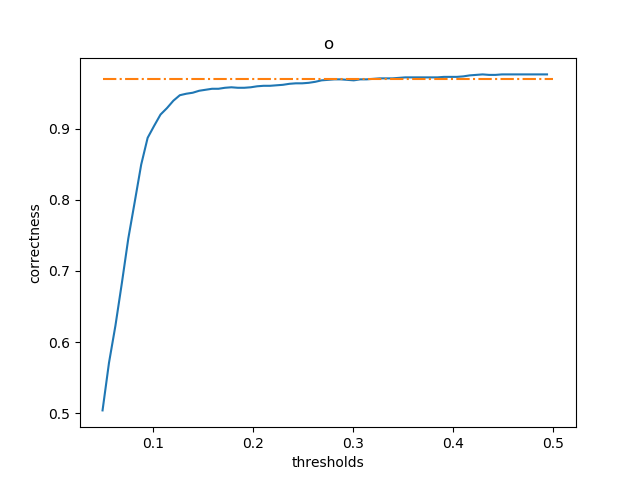
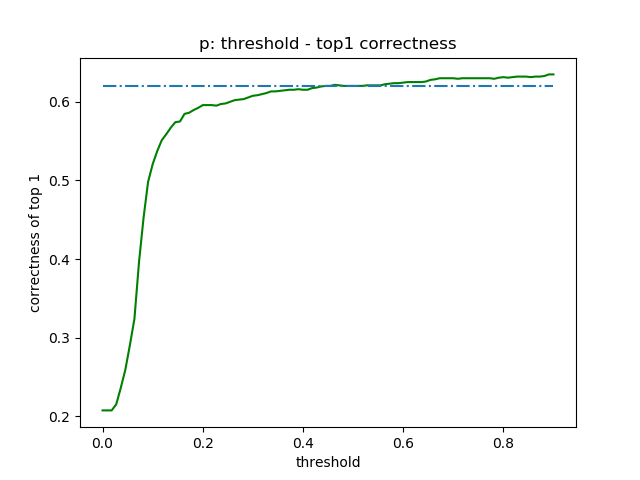
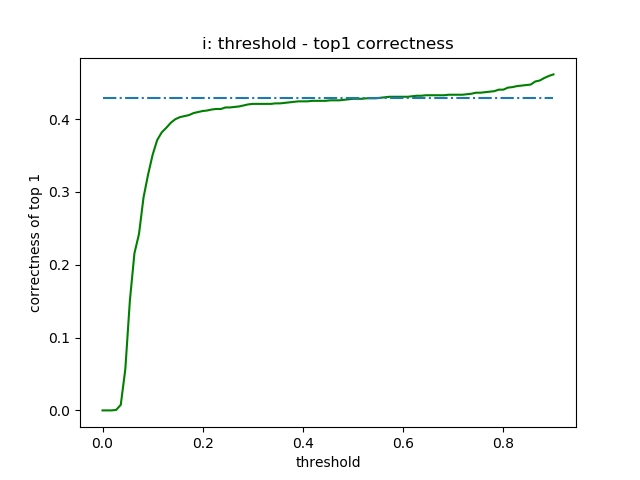
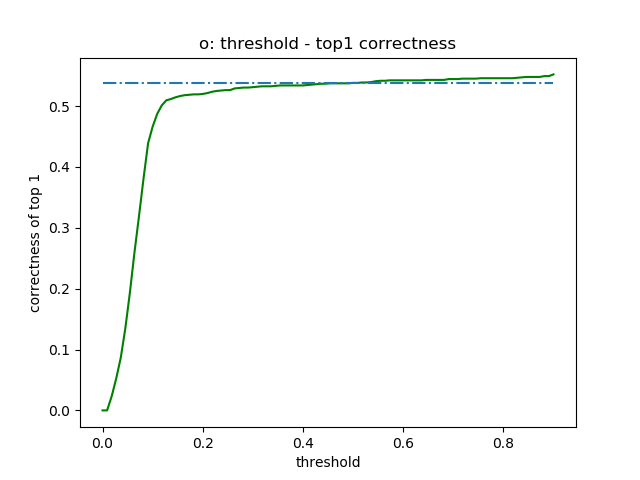
## 二、对于细粒度的实验与分析

1. 实验内容是：对于上述的长的医学句子而言，我们希望通过其中某些关键性的子句来获取到整个句子属于 PIO 的某些子类中的哪几个类别。
2. 方法一：对于每一个短句进行细粒度分类，然后将整个句子划分成若干个短句，分别经过短句分类的模型，然后取出其中最高频度的出现的分类。这种方法在实验后发现，由于 BERT 模型往往需要较长的句子，才能够分析出其中每个字对于前后的上下文关系，所以对于短语的分类效果非常不理想，故舍弃。
3. 方法二：利用类似于遮盖的技术，将原来长句中的短句遮盖掉，并将遮盖后的结果的句子，加入到训练集中，使其标签为全零。我们希望通过这种方式来训练模型，能够反向识别出某个句子中关键性的部分。比如有句子为 This is a test for 64 African women. 我们将其中属于 po.size 的部分遮盖掉得到 This is a test for African women 并让其标签属性为全 0，从而训练模型认识其中某些部分。这种方法再试验后，也是失败了，主要原因在我看来，是因为对于句子其他部分的偏差不断累积，反而超出了对于其中缺失部分的识别强度。
4. 最终我们使用直接训练整个长句子，搭配首要阈值的分析方法来解决问题。

## 三、模型求解

* 算法流程图如下



* 至于大类与小类别的分类预测器的构建，构建的方法已经在第一部分的 BERT 模型中讲述完毕了，主要在实践过程中，对于句子长度这个超参数要人为确定，其他均使用 BERT 模型的默认结构即可
* // TODO STUB
* // 在这里要加入 zjy 之前的图，有点难看？？
* 实际过程中我们还去掉了无用的标点符号，并将数字保留以及做统一的小写化操作
* 
* 对于模型对大的类别 PIO 的分类结果情况
* 
* 不考虑细粒度标签时，P 标签的阈值取：0.346，最终正确率结果：0.954
* 
* 不考虑细粒度标签时，I 标签的阈值取：0.243，最终正确率结果：0.969
* 
* 不考虑细粒度标签时，O 标签的阈值取：0.281，最终正确率结果：0.957
  + 如何选定阈值，使用的方法是：计算出对于某个阈值之后的所有正确率，对于该阈值的均方差取到最小值
* 对于模型每个大类下小类别的阈值与正确率的关系曲线
* 
* 当考虑细粒度时，P 标签的阈值取：0.45，最终正确率结果：0.62
* 
* 当考虑细粒度时，I 标签的阈值取：0.53，最终正确率结果：0.43
* 
* 当考虑细粒度时，O 标签的阈值取：0.45，最终正确率结果：0.54

## 四、实验总结

1. 可以从结果中看到，对于训练集的大类 PIO 的分类是我们比较容易高准确度实现的，但是对于细粒度的分类，却十分难以达到好的结果
2. 细粒度分类难以达成的原因：
   1. 细粒度的所有种类，一共有30种以上，而对于在一开始构建的数据集中，经过统计后，所有的细粒度一共不重复的出现的有 20 种，所以也就是说，并不是所有细粒度都可以被训练得到，甚至对于一些极其少出现的细粒度，我们也不得不训练，这反而影响了整个模型的精度
   2. 人工标注的训练集本身存在较多歧义，对于一些细粒度标签比如 医疗过程中病人的治疗表现 与 医疗过程中病人的治疗结果 很多情况是有覆盖的情况的，所以导致了各个参数之间不独立，相互影响
   3. 同时，模型本身也具有一定的缺陷，在处理多标签的过程中，我们采用的是多个二分类结合使用的方法，因此，可能会导致两个细粒度之间的关联性被破坏，影响模型精度
3. 总结。。。