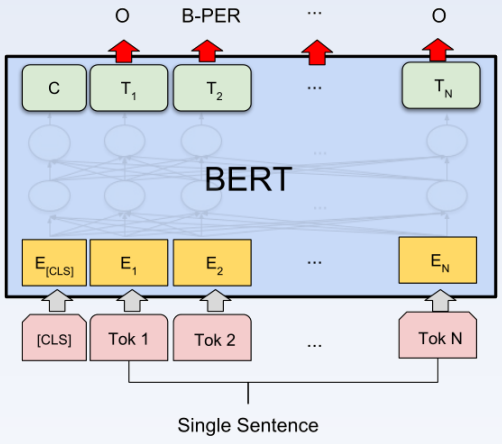
西安交通大學
XI'AN JIAOTONG UNIVERSITY

工作汇报

刘卓

2022. 05. 29

条件随机场CRF (Conditional Random Field)



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

命名实体识别NER
(Named Entity Recognition)

例如：
O：非人名地名机构名
B-PER/LOC/ORG：
　　人名/地名/机构名初始单词
I-PER/LOC/ORG：
　　人名/地名/机构名中间单词
E-PER/LOC/ORG：
　　人名/地名/机构名终止单词
S-PER/LOC/ORG：
　　人名/地名/机构名独立单词

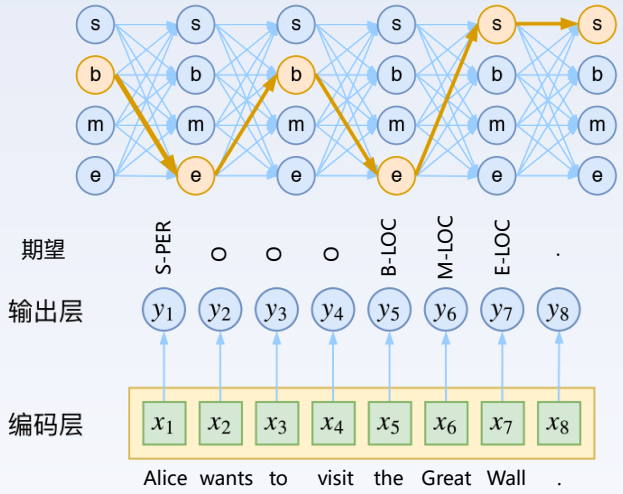
上次提到了BERT针对单句输入的一个任务，命名实体识别NER
它把每一个词对应的最终层向量送入分类器，划分为不同的标注标签，比如说人名的初始位单词、中间位单词、终止位单词等

条件随机场CRF (Conditional Random Field)

西安交通大学 XIAN JIAOTONG UNIVERSITY

命名实体识别NER (Named Entity Recognition)

例如：
O：非人名地名机构名
B-PER/LOC/ORG：
　　人名/地名/机构名初始单词
M-PER/LOC/ORG：
　　人名/地名/机构名中间单词
E-PER/LOC/ORG：
　　人名/地名/机构名终止单词
S-PER/LOC/ORG：
　　人名/地名/机构名独立单词



比如说右边这个输入序列(中文输入的最小单位可以是单个字，也可以是通过hanlp/jieba/pkuseg/stanford等分词工具处理后的词)
我们可能期待的结果是这样的
但实际上，这些标签自带一些隐含的约束，如B-I-E需要按顺序依次出现才能切出来一个词，而不能是E-I-B等顺序
如果逐个词去分类，是很难兼顾这种约束的
于是就有了CRF，比如说这张图，输入的序列有6个token，不考虑约束和概率，会有4的6次方条可能的标注序列，CRF做的就是在这些里边找到一条概率最大的合法标注序列
相比之下，逐个token进行softmax，实际上是6个4分类问题，6个token之间互不干扰
理论上这种效果是可以设计规则来实现的，但会比较复杂，计算量也比较大，CRF采用的是学习的方法

条件随机场CRF (Conditional Random Field)

$P(y_1, \dots, y_n | x_1, \dots, x_n)$

称输入序列为X，某一特定标签序列为Y_i

即求$P(Y_i|X)$

现假设：

$$P(Y_i|X) = \frac{e^{s(X,Y_i)}}{\sum_{Y_n} e^{s(X,Y)}}$$

$$s(X, Y_i) = \sum_{j=1}^{n-1} A(y_j, y_{j+1}) + \sum_{j=1}^{n} P(y_j|x_j)$$

转移分数　　发射分数

$$Loss = -\log P(Y_{gt}|X) = s(X, Y_{gt}) - \log \sum_{Y_n} e^{s(X,Y)}$$

现在要求的就是在上述约束下，给定序列x1-xn，预测标签序列是y1-yn的概率，如果y1-yn的排列不合理，概率会是一个很小的值

CRF做了一些假设，首先是假设这个概率是一个类似于softmax的分布，然后假设对于每一种标签序列Yi，都有一个从约束和预测两个角度评价它是否合理的分数s

s由两部分组成

一个是转移分数来衡量约束，这个可以理解为第i个标签接第i+1个标签的概率(其实不能说是概率，因为总和不一定为1)，这个是需要我们学习的参数，一般会随机初始化

另一个是发射分数来衡量预测，这就是模型比如说BERT这些预测的第i个token对应标签是Yi序列中对应位置的那个的概率，我们已经得到了

在BERT这类模型得到token标签之后，我们再加一个CRF层，Loss设为ground truth标签的负对数，从而来学习转移分数

条件随机场CRF (Conditional Random Field)

$$Loss = -\log P(Y_{gt}|X) = s(X, Y_{gt}) - \log \sum_{Y_n} e^{s(X,Y)}$$

令$Z(X) = \sum_{Y_n} e^{s(X,Y)}$

令Z_j^k表示j时刻所有以标签k为终点的路径分数的指数和

令$Z_j = [Z_j^1, \dots, Z_j^k]$，则有$Z(X) = \mathrm{sum}(Z_n)$

$$s(X, Y_i) = \sum_{j=1}^{n-1} A(y_j, y_{j+1}) + \sum_{j=1}^{n} P(y_j|x_j) \quad \sum_{Y_{n-1}} e^{s(X, Y_{\{1,\dots,n-1\}})} \rightarrow \sum_{Y_n} e^{s(X, Y_{\{1,\dots,n\}})}$$

所有终点y为标签k的总和　　所有终点y为标签k的总和

$$s(X, Y_{\{1,\dots,n\}}) = \sum_{j=1}^{n-2} A(y_j, y_{j+1}) + \sum_{y_{n-1}, y_n}^{k^2} A(y_{n-1}, y_n) + \sum_{j=1}^{n-1} P(y_j|x_j) + P(y_n|x_n)$$

现在的问题就是怎么算P(Yi|X)的分母，或者说Loss的第二项
之前也说了，如果有k个标签，n个token，这就会有k^n条可能的标签序列，遍历求和是不大现实的
于是它这里考虑用递推关系来计算
列公式之前举个例子，我现在已知1到n-1标签序列的分数指数和，参考这个分数的计算公式，该如何求1到n标签序列的分数指数和呢
答案是选择已知标签序列里，所有以某个标签k为n-1时刻终点的分数指数和，乘以下一时刻也就是第n个标签yn的预测概率，再乘以从yn-1到yn的转移分数
yn-1和yn都各自有k个值待选，把它们一共的k^2个组合项累加起来，就是要求的下一时刻的分数指数和，这也就是上边用终点标签进行定义的原因
CRF别的部分网上都有资料，但这部分没找到详细过程，思考了好久才推出来，但还是感觉不很好理解

条件随机场CRF (Conditional Random Field)

$$
\begin{aligned}
Z_j^1 &= \left(Z_{j-1}^1 e^{A(1,1)} + Z_{j-1}^2 e^{A(2,1)} + \cdots + Z_{j-1}^k e^{A(k,1)}\right) e^{P(1|x_j)} \\
Z_j^2 &= \left(Z_{j-1}^1 e^{A(1,2)} + Z_{j-1}^2 e^{A(2,2)} + \cdots + Z_{j-1}^k e^{A(k,2)}\right) e^{P(2|x_j)} \\
&\quad \cdots \\
Z_j^k &= \left(Z_{j-1}^1 e^{A(1,k)} + Z_{j-1}^2 e^{A(2,k)} + \cdots + Z_{j-1}^k e^{A(k,k)}\right) e^{P(k|x_j)}
\end{aligned}
$$

k^2个项

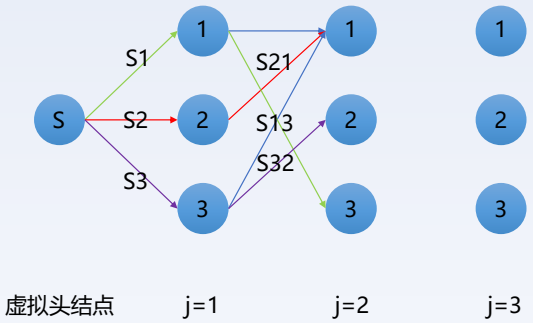
$$Z_j = Z_{j-1} A \oplus P_j$$

$$Z_j = \left[Z_j^1, \ldots, Z_j^k\right], P_j = \left[e^{P(1|x_j)}, \ldots, e^{P(k|x_j)}\right]$$

$$A = \left[\left[e^{A(1,1)}, \ldots, e^{A(1,k)}\right], \ldots, \left[e^{A(k,1)}, \ldots, e^{A(k,k)}\right]\right]$$

这是对应的递推公式，可以看出的确是有k^2个项
至于Z1，直接按定义算就可以了，也就是对第1个时间步所有转移分数和发射分数之和求指数
因为j=1之前没有标签，所以转移分数也就都没有，Z1实际上也就是对发射分数之和进行指数
于是P(Yi|X)和Loss就都可以算了

条件随机场CRF (Conditional Random Field)



至于转移分数训练完后，怎么从k^n条路径中找到计算概率最大的，肯定还是不可能所有路径的结果都求一遍再比较，需要用到维特比算法，是一个动态规划，主要是提前剪枝的思想

主要用一个例子来说明吧，这是一个n=3，k=3的序列标注，为了便于说明，加一个虚拟头结点S

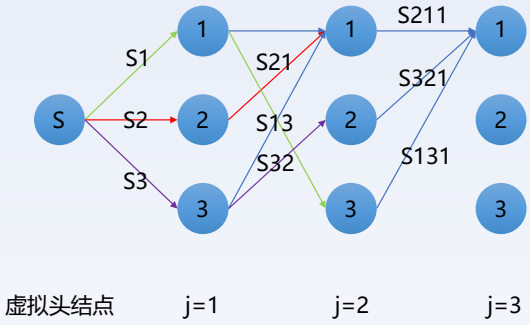
首先看j=1这一个时间步的所有标签，找源结点到这一层每个标签的最高概率，概率刚才说了都是能算的，这里没有其它可比较的，也就是S1、S2、S3

然后是j=2，对这一层的标签1来说，到它有3条路径，S11、S21、S31，假设其中S21的概率最大，那我们就只保留S21这条路径，其它的都删掉，后续任何经过j=2，k=1的序列都只走S21这条路

原因很好理解，在所有经过j=2，k=1的序列中，概率最大的序列肯定不会经过S11与S31，经过S21的序列一定是最大的，这就起到了一个提前剪枝的作用，省去了很多重复计算的操作

所以说j=2这一层有3个标签，能找出3条到这一层概率最大的序列，我们假设是S21、S32、S13

条件随机场CRF (Conditional Random Field)



同理，在j=3时，对k=1，只能考虑S211、S131、S321，此外，最终落脚于k=2和k=3还各自有3条，最终比较这9条，可以得到概率最大的标注序列
可见很多不必要的概率计算我们就省略了，比如说S111、S221这些，都因为提前剪枝而排除了
以上就是CRF算法的内容，使用的时候一般是在BERT这类实体标签分类模型之后加一个CRF层，pytorch有包可以调
此后一段时间的计划是，多涉猎一些pytorch代码的实现，包括而不限于text-CNN、BERT-BiLSTM-CRF、BERT白化、first-last-avg等，感觉这方面的功底还是很薄弱，以及项目、刷题、准备考试等，抽空了解下最基础的vit和vivit