

NLP 入门-BiLSTM+CRF 分词模型搭建

...

2021 年 1 月 26 日

目录

1 任务介绍	1
2 模型介绍	2
3 计算技巧	3
4 维特比算法在 CRF 中的应用	6
5 代码链接	8

1 任务介绍

序列标注任务。即给定一个观测序列，获取其隐藏序列标签。常见的有词性标注、分词任务、实体识别任务等。例如分词任务，给定一句话（观测序列）“我爱中国”，可以将其分成三个词：['我','爱','中国']。如果用字母‘b’ (begin) 代表词的第一个字符的标签，‘i’ (inner) 表示词的内部和结尾字符标签，则观测序列“我爱中国”对应的隐藏序列标签为：“bbbi”。上述分词任务可以采用分类的思想来完成，将句子中的每个字符分到类别‘b’或类别‘i’中，那么就可以直接采用之前的 BiLSTM 分类模型，但是该模型的交叉熵损失函数只能关注当前字符的标签，不能与前一个字符的标签联系。例如，当前字符标签为‘i’，且是第一个，可知其前面一定会有‘b’标签。交叉熵损失函数无法利用类似的约束来计算损失值，也就无法抓住这样的特征。例如“我爱中国”，直接用 BiLSTM 进行字符分类的话，可能会得到这样的结果：“ibbi”，第一个字符就出现‘i’标签是完全不合理的。所

以，此处利用条件随机场（CRF）计算损失值，可以将类似的约束融合到模型中，使结果更加准确。

2 模型介绍

1. BiLSTM，双向的 LSTM 模型，详见上一篇总结。
2. CRF，条件随机场，下面以观测序列“我爱中国”和对应的隐藏序列“bbbi”为例，介绍条件随机场。

假设有以下词表 (vocab) 和 label-id:

Instance	我爱中国
label	bbbi
vocab	{ ‘我’ :0, ‘爱’ :1, ‘中’ :2, ‘国’ :3, ‘<pad>’ :4, ‘<unk>’ :5}
label-id	{ ‘b’ :0, ‘i’ :1, ‘<start>’ :2, ‘<end>’ :3}

表 1: 说明

表 1 中‘<pad>’和‘<unk>’表示 padding 字符和未登录字符；‘<start>’和‘<end>’表示一句话的开始和结束。因此：

word_id		0	1	2	3	
word		我	爱	中	国	
label	<start>	b	b	b	i	<end>
label_id	2	0	0	0	1	3

表 2: 例子

最后得到数字化表示如表 3。

word_id	0	1	2	3		
label_id	2	0	0	0	1	3

表 3: 数字化

由于每一个字符对应的标签可能有两种，就设置一种从 word_id 到 label_id 的分数，也叫发射分数 (emit)，例如 emit[0][0] 就表示字符为 0-> ‘我’，标签为 0-> ‘b’ 的发射得分。

另外，为了加上前一个字符的约束，就加入了转移分数 (trans)，例如 $\text{trans}[0][1]$ 就表示从标签 0-> ‘b’ 转移到 1-> ‘i’ 的得分。也就是当前字符标签为 ‘i’，上一个字符标签为 ‘b’ 的得分。

要计算表 3 中隐藏序列：“2->0->0->0->1->3” 的总得分：
 $\text{scores} = \text{trans}[2][0] + \text{emit}[2][0] + \text{trans}[0][0] + \text{emit}[3][0] + \text{trans}[0][0] + \text{emit}[4][0] + \text{trans}[0][1] + \text{emit}[5][1] + \text{trans}[1][3]$ 。如果将每一个隐藏序列看做一条路径，那么除了上述表 3 正确的路径（对应正确的分词结果“我 | 爱 | 中国”）外，还有可能有其他的路径，例如“2->0->1->0->1->3”对应分词结果“我爱 | 中国”。也就是说从开始 <start> 标签到 <end> 结束标签中间有很多条路径（上面的例子有 2^4 条路径，2 个标签，观测序列长度为 4），每个路径都有一个得分。最后，计算每条路径得分公式可以如下：

$$\text{Score}(X, y) = \sum_{i=0}^n \text{trans}_{(y_i, y_{i+1})} + \sum_{i=0}^n \text{emit}_{(w_i, y_i)} \quad (1)$$

式中 X 表示观测序列，y 表示隐藏序列标签，i 即字符下标，n 表示观测序列一共有 n 个字符。

通过训练神经网络，使得上面的 scores 在所有路径得分的结果中出现的概率最大，这个时候一般 softmax 化，即：

$$p(\hat{y}|X) = \frac{\exp(\text{scores})}{\sum \exp(\text{Score}(X, y))} \quad (2)$$

这个概率值越大，表示预测的越准确。因此可以用该公式当做 loss，但是 $p(\hat{y}|X)$ 范围为 [0,1]，结果越趋近于 1 越好，loss 一般趋近于 0 才好，所以加个 log，此时是从负无穷趋向于 0 的，所以再加个负号，loss 就变成了越小越趋近于 0 越好了：

$$-\log(p(\hat{y}|X)) = -\text{scores} + \log\left(\sum \exp(\text{Score}(X, y))\right) \quad (3)$$

可以看出使用 CRF 需要发射分数 emit 和转移分数 trans 两个矩阵。实验中，发射分数 emit 矩阵一般为 BiLSTM 的输出，trans 矩阵一般是 CRF 层的参数，初始化之后通过训练来更新。emit 矩阵的形状为 [n,m]，n 表示观测序列长度，m 表示标签的种类数。trans 矩阵的形状为 [m,m]。

3 计算技巧

训练期间，要计算公式 (3)，需要解决两个问题：如何计算真实路径得分 (scores) 和如何计算所有路径总得分 ($\log \sum \exp(\text{Score}(X, y))$)。

1. 计算真实路径得分 scores。在上文通过例子“我爱中国”计算过，实际计算过程类似，只需要对应的 emit 矩阵和 trans 矩阵即可直接求结果。
2. 计算 $\log \sum \exp(\text{Score}(X, y)) = \log (e^{S_1} + e^{S_2} + \dots + e^{S_k})$ ，一共有 k 条路径。BiLSTM 输出可以得到发射矩阵 Emit: E_{ij} ，表示字符 w_i 的标签为 l_j 的得分；CRF 层参数可以得到转移矩阵 Trans: T_{ij} ，表示标签 l_i 变为标签 l_j 的得分。举个例子，假设有三个字符“ w_0, w_1, w_2 ”，两个标签“ l_0, l_1 ”，可视化所有路径如下：

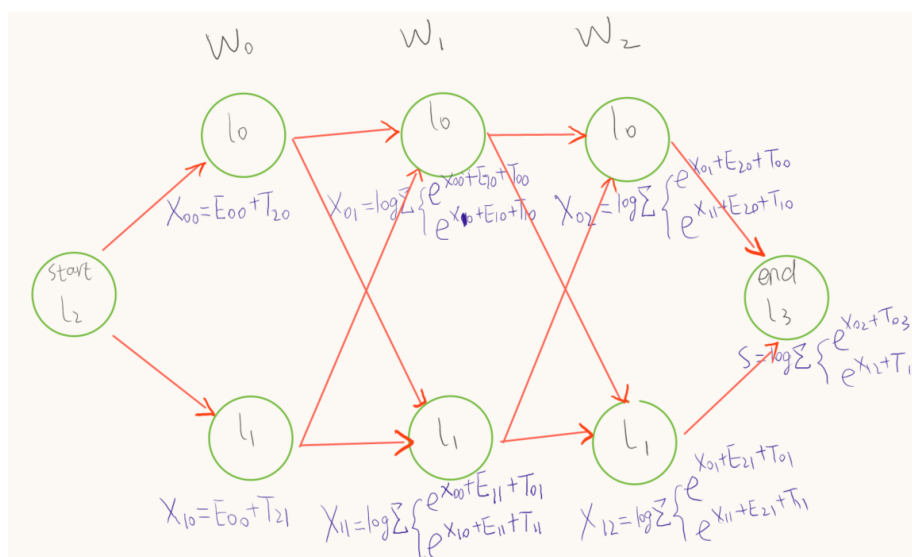


图 1: 路径

如图， X_{ij} 表示到该节点所有路径的总得分，但是图中 X_{01} 以后的都在该节点得分前加了 $\log \sum \exp()$ 函数，下面解释为何要如此做：一步一步计算：

(a) $start \rightarrow w_0$

$Pre_0 = None$ 。

$E_0 = [E_{00}, E_{01}]$

$A_0 = Pre_0 + E_1 + T_s = [E_{00} + T_{20}, E_{01} + T_{21}] = [X_{00}, X_{01}], T_s$ 即

从 start 到各个标签的转移概率，一般可初始化为 0，或者由其它方法获取。

(b) $start \rightarrow w_0 \rightarrow w_1$

$$Pre_1 = A_0 = [X_{00}, X_{01}]$$

$$E_1 = [E_{10}, E_{11}]$$

接下来计算 A_1 时为了方便, 把 Pre_1, E_1 扩展成如下形式:

$$Pre_1 = \begin{bmatrix} X_{00} & X_{00} \\ X_{01} & X_{01} \end{bmatrix} \quad E_1 = \begin{bmatrix} E_{10} & E_{11} \\ E_{10} & E_{11} \end{bmatrix} \quad (4)$$

$$A_1 = Pre_1 + E_1 + Trans$$

$$\begin{aligned} &= \begin{bmatrix} X_{00} & X_{00} \\ X_{01} & X_{01} \end{bmatrix} + \begin{bmatrix} E_{10} & E_{11} \\ E_{10} & E_{11} \end{bmatrix} + \begin{bmatrix} T_{00} & T_{01} \\ T_{10} & T_{11} \end{bmatrix} \\ &= \begin{bmatrix} E_{00} + E_{10} + T_{00} & E_{00} + E_{11} + T_{01} \\ E_{01} + E_{10} + T_{10} & E_{01} + E_{11} + T_{11} \end{bmatrix} \end{aligned} \quad (5)$$

(c) $start \rightarrow w_0 \rightarrow w_1 \rightarrow w_2$

$$Pre_2 = ?$$

上面的 A_1 矩阵四个元素刚好对应着四条路径的得分, 想迭代之前的操作的话, 从 (a) 到 (b) 的时候令 $Pre_1 = A_0$, 剩下计算步骤都是一样的, 但是上面的 A_0 只有两个元素, 想令 $Pre_2 = A_1$ 后, 使得计算步骤一样, 但是 A_1 有四个元素, 多了两个怎么办呢? 别急, 如果 $w_2 = end$, 即只有两个字的话, 一共四条路径 $[A_{00}, A_{10}, A_{01}, A_{11}]$, 计算结果, 咱们先看看最后的结果公式中的变形: $\log(e^{A_{00}} + e^{A_{10}} + e^{A_{01}} + e^{A_{11}}) = \log((e^{A_{00}} + e^{A_{10}}) + (e^{A_{01}} + e^{A_{11}})) = \log(e^{\log(e^{A_{00}} + e^{A_{10}})} + e^{\log(e^{A_{01}} + e^{A_{11}})})$

所以令 $Pre_2 = [\log(e^{A_{00}} + e^{A_{10}}), \log(e^{A_{01}} + e^{A_{11}})] = [X_{01}, X_{11}]$, 接下来的计算步骤就跟上面的一样了。

$$Pre_2 = [\log(e^{A_{00}} + e^{A_{10}}), \log(e^{A_{01}} + e^{A_{11}})] = [X_{01}, X_{11}]。$$

$$E_2 = [E_{20}, E_{21}]$$

$$A_2 = Pre_0 + E_1 + Trans = 2 * 2 \text{ 的矩阵}$$

(d) $start \rightarrow w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow end$ $Pre_3 = [X_{02}, X_{12}]。$

$$E_3 = None$$

$A_3 = Pre_3 + E_3 + T_e = [X_{02} + T_{03}, X_{12} + T_{13}]$, 这里 T_3 即从各标签到 end 标签的转移概率, 一般可初始化为 0, 或者以其它方法获得。

$$S = \log(e^{A_{30}} + e^{A_{31}})$$

总得分有了, 计算结果: $\log \sum \exp(\text{Score}(X, y)) = \log \sum (e^S) = \log(e^{A_{30}} + e^{A_{31}}) = S$

上面的过程展示了如何利用矩阵提高计算速度的技巧, 但这里是训练过程中计算损失函数的, 那么在不知道哪一条路径最好的情况下, 给定一个新的观测序列, 利用模型 BiLSTM 预测的 Emit 矩阵和模型 CRF 的 Trans 参数矩阵, 如何找到得分概率最大的路径呢? 这里就需要借用维特比算法, 是一种动态规划算法, 常用来找最短路径或最长路径。思想: 从 start 到 end 之间的路径最长, 那么从 start 到中间任意层节点的路径一定也是最长的。反证法很好证明。

4 维特比算法在 CRF 中的应用

还是以上面的例子说明, 参考下面图 2,

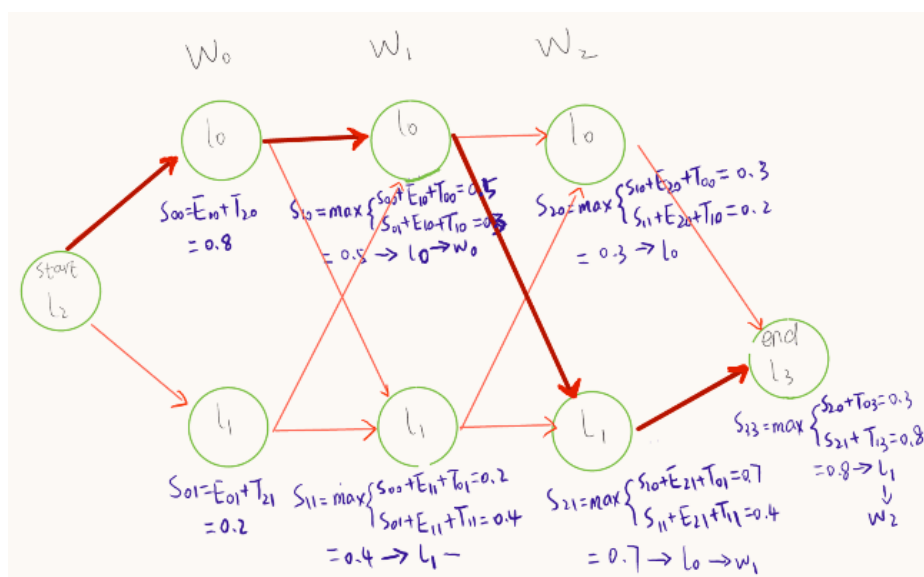


图 2: viterbi 路径

1. $start \rightarrow w_0$
 $Pre_0 = None$
 $E_0 = [E_{00}, E_{01}]$

$A_0 = [X_{00}, X_{01}]$, 假设 $A_0 = [0.8, 0.2]$, 最大值 0.8 下标为 0, 如果只有一个字的话, 很明显这个字的标签就是 $0 \rightarrow$ ‘b’。

2. $start \rightarrow w_0 \rightarrow w_1$

$$Pre_1 = A_0 = [X_{00}, X_{01}]$$

$$E_1 = [E_{10}, E_{11}]$$

按照上面的计算方法有:

$$A_1 = \begin{bmatrix} E_{00} + E_{10} + T_{00} & E_{00} + E_{11} + T_{01} \\ E_{01} + E_{10} + T_{10} & E_{01} + E_{11} + T_{11} \end{bmatrix}$$

目前跟之前的计算都是一样的, 但是下一步令 $Pre_2 = [\max(A_{00}, A_{10}), \max(A_{01}, A_{11})]$,

如果上述的 $A_1 = \begin{bmatrix} 0.5 & 0.3 \\ 0.2 & 0.4 \end{bmatrix}$, 那么

$$Pre_2 = [0.5, 0.4]。$$

同时为了记录下最大值的路径, 设置矩阵 P 记录每一列最大值出现的行标, 所以 $p_0 = (0, 1)$

3. $start \rightarrow w_0 \rightarrow w_1 \rightarrow w_2$ $Pre_2 = [\max(A_{00}, A_{10}), \max(A_{01}, A_{11})] = [0.5, 0.4]$

$$E_2 = [E_{20}, E_{21}]$$

按照上面的计算方法有:

$$A_2 = \begin{bmatrix} 0.3 & 0.7 \\ 0.2 & 0.4 \end{bmatrix}$$

路径 $p_1 = (0, 0)$

4. $start \rightarrow w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow end$

$$Pre_3 = [0.3, 0.7]$$

$$E_3 = None$$

$$A_3 = [0.3, 0.8]$$

, 最大值 0.8, 表示上一个字符 w_2 的标签为 1, 接下来回溯找最佳路

径。已知 $P = [p_0, p_1] = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$

知道 w_2 的标签为 1, 就取 p_1 下标为 1 (从 0 开始) 的标签是 0, 即

w_1 的标签为 0, 接着取 p_0 下标为 0 的标签是 0, 即 w_0 的标签为 0,

所以最佳路径即: “ $0 \rightarrow 0 \rightarrow 1$ ”, 即 “b \rightarrow b \rightarrow i”, 即 “ $w_0 | w_0 w_1$ ”

5 代码链接

1. ...