链滴

# crf++ 里的特征模板

作者：netblind

原文链接：https://ld246.com/article/1543380116044

来源网站：链滴

一:

Unigram和Bigram模板分别生成CRF的状态特征函数S_l(y_i,x,i) 和转移特征函数t_k(y_{i-1},y_i,x,i) 其中 y_i 是标签， x 是观测序列， i 是当前节点位置。每个函数还有一个权值，具体请参考CRF相关料。

crf++模板定义里的%x[row,col]，即是特征函数的参数x 。

举个例子。假设有如下用于分词标注的训练文件：

<pre><code>
北 N B
京 N E
欢 V B
迎 V M
你 N E
</code></pre>

其中第3列是标签，也是测试文件中需要预测的结果，有BME 3种状态。第二列是词性，不是必须的。

特征模板格式：%x[row,col]。x可取U或B，对应两种类型。方括号里的编号用于标定特征来源，row示相对当前位置的行，0即是当前行；col对应训练文件中的列。这里只使用第1列（编号0），即文字。

1、Unigram类型

每一行模板生成一组状态特征函数，数量是LN 个，L是标签状态数。N是此行模板在训练集上展开后唯一样本数，在这个例子中，第一列的唯一字数是5个，所以有LN = 3*5=15。

例如：U01:%x[0,0]，生成如下15个函数：

<pre><code>
func1 = if (output = B and feature=U01:"北") return 1 else return 0
func2 = if (output = M and feature=U01:"北") return 1 else return 0
func3 = if (output = E and feature=U01:"北") return 1 else return 0
func4 = if (output = B and feature=U01:"京") return 1 else return 0
...
func13 = if (output = B and feature=U01:"你") return 1 else return 0
func14 = if (output = M and feature=U01:"你") return 1 else return 0
func15 = if (output = E and feature=U01:"你") return 1 else return 0
</code></pre>

这些函数经过训练后，其权值表示函数内文字对应该标签的概率（形象说法，概率和可大于1）。

又如 U02:%x[-1,0]，训练后，该组函数权值反映了句子中上一个字对当前字的标签的影响。

2、Bigram类型

与Unigram不同的是，Bigram类型模板生成的函数会多一个参数：上个节点的标签 。

生成函数类似于：

<pre><code>func1 = if (prev_output = B and output = B and feature=B01:"北") return 1 else eturn 0</code></pre>

这样，每行模板则会生成 LLN 个特征函数。经过训练后，这些函数的权值反映了上一个节点的标签当前节点的影响。

每行模版可使用多个位置。例如：U18:%x[1,1]/%x[2,1]

字母U后面的01，02是唯一ID，并不限于数字编号。如果不关心上下文，甚至可以不要这个ID。

二:

根据CRF++包中给出的例子可以较为清楚的知道有那些模板构建方法。模板构建分为两类，一类是Unigram标注，一类是B i gram标注。

Unigram模板是比较常用的模板，这类模板提取的信息较为全面，组成的模板数量也比较多；Bigra模板比较简单，一般是当前词和前面一个词的自动组合生成的Bigram特征集合。

主要介绍Unigram模板

```
<pre><code>
Unigram
U00:%x[-2,0]
U01:%x[-1,0]
U02:%x[0,0]
U03:%x[1,0]
U04:%x[2,0]
U05:%x[-2,0]/%x[-1,0]/%x[0,0]
U06:%x[-1,0]/%x[0,0]/%x[1,0]
U07:%x[0,0]/%x[1,0]/%x[2,0]
U08:%x[-1,0]/%x[0,0]
U09:%x[0,0]/%x[1,0]
</code></pre>
```

这是CRF++例子中给出的模板，一共有 9 个模板，先看第一个模板，表示当前词和前面的第二个词组的特征，以‘小明今天穿了一件红色上衣’为例，符合CRF++处理格式的这句话应该变成如下形式：

```
<pre><code>小  B
明  I
今  B
天  I
穿  S
了  S
一  B
件  I
红  B
色  I
上  B
衣  I </code></pre>
```

假设我们有三个标记tag， B （表示一个词的开头那个字）， I （表示一个词的结尾那个字）， S （示单个字的词），先看第一个模板U00:%x[-2,0],第一个模板产生的特征如下：

如果当前词是‘今’，那-2位置对应的字就是‘小’，

每个特征对应的字如下：

<pre><code>U00:%x[-2,0]======>小
U01:%x[-1,0]======>明
U02:%x[0,0]======>今
U03:%x[1,0]======>天
U04:%x[2,0]======>穿
U05:%x[-2,0]/%x[-1,0]/%x[0,0]======>小/明/今
U06:%x[-1,0]/%x[0,0]/%x[1,0]======>明/今/天
U07:%x[0,0]/%x[1,0]/%x[2,0]======>今/天/穿
U08:%x[-1,0]/%x[0,0]======>明/今
U09:%x[0,0]/%x[1,0]======>今/天 </code></pre>

根据第一个模板U00:%x[-2,0]能得到的转移特征函数如下：

<pre><code>func1=if(output=B and feature=’U00:小‘) return 1 else return 0 </code><pre>

其中output=B 指的是当前词（字）的预测标记，也就是’今‘的预测标记，每个模板会把所有可能标记输出都列一遍，然后通过训练确定每种标记的权重，合理的标记在训练样本中出现的次数多，对的权重就高，不合理的标记在训练样本中出现的少，对应的权重就少，但是在利用模板生成转移特征数是会把所有可能的特征函数都列出来，由模型通过训练决定每个特征的重要程度。

<pre><code>func2=if(output=I and feature=’U00:小’) return 1 else return 0
func3=if(output=S and feature=’U00:小) return 1 else return 0 </code></pre>

得到三个特征函数之后当前这个字’今‘的特征函数利用第一个模板就得到全不了，然后扫描下一个‘天’，以’天‘字作为当前字预测这个字的标记tag,同样会得到三个特征函数：

<pre><code>
func4=if(output=B and feature=’U00:明’) return 1 else return 0
func5=if(output=I and feature=’U00:明’) return 1 else return 0
func6=if(output=S and feature=’U00:明’) return 1 else return 0 </code></pre>

特征函数中的feature指的是当前词的-2位置对应的词或对应的词的特征，因为在这里没有其他特征，所以用字本身做特征，有的会有词性，那feature就会是’明‘这个字对应的词性而不是字本身了。

这个feature的作用就是确定模板所确定的当前词和临近词