**BioCreative 2 Gene Mention 任务实验记录**

刘景琛

2010年1月18日

为了测试系统对文本中基因、蛋白质名字的识别效果，使用Biocreative 2中Gene Mention任务的数据进行实验。Gene Mention任务（GM任务）的目的是要将文本中的基因及其产物（也就是蛋白质、RNA之类）的名字标注出来，结果要给出这个名字所在句子的ID和在句子中的起始结束位置。这个工作是Normalization之前的步骤，是Gene Normalization任务的基础。

**一、实验数据**

Biocreative 2中Gene Mention任务提供了15000个句子的训练数据和5000个句子的测试数据。根据官方Overview的介绍[1]，这些数据是为了一个叫AbGene tagger的项目而标注的数据。他们先用了一个贝叶斯分类器将一批文档分为富含基因名字的和不含基因名字的两类，然后分别从这两类中各抽出10000个句子进行了标注。之后他们把这20000个句子分为4组：train、test、round1、round2。这四组中富含基因和不含基因的句子都是相等数量的。在BioCreative I中，train和test被作为训练数据，round1被作为测试数据。在BioCreative II中，train、test、round1作为训练数据，round2作为测试数据。

数据的格式是这样的：训练数据和测试数据都分别有两个文件，一个文件提供未标注的原文句子，另一个提供标注结果。每一个句子都有一个ID，通过这个ID将句子和标注信息联系起来。

例如在原句文件中有一句：

BC2GM087236793 Early complement components, C1q and C4, and IgA secretory piece were absent.

在标注文件中有它对应的标注：

BC2GM087236793|26 28|C1q

BC2GM087236793|32 33|C4

BC2GM087236793|38 40|IgA

组织方提供了一个perl语言的评测程序。所以我在实验中将最后的结果输出成GM任务要求的输出格式，然后就可以使用这个标准的评测程序进行评测，这样可以与其他参加者的结果进行公平的比较。

关于每个句子的id的具体含义，我并没有找到官方的说明，但也可以大概猜得出来。在训练数据和测试数据中，句子id的形式是不一样的。在训练数据中，句子是这样的：

P00029953T0045 Characteristics of lipase activity.

P00030183T0000 Takayasu's disease: association with HLA-B5.

P00030937A0119 SGPT, SGOT, and alkaline phosphatase concentrations were essentially normal in all subjects.

用PubMed一搜就可以知道，P后面那个数字是PubMed ID。后面的T和A因为只有两种，很容易让人联想起他们那个“含基因”和“不含基因”的两个分类，但是再仔细一想，发现其实就是Title和Abstract。后面那个数字就看不出有什么特殊意义了，大概是他们自己处理时用的ID。

而在测试数据中就不是这样了。测试数据的样子是：

BC2GM086842152 Accumulation of amines in the isolated perfused rabbit lung.

BC2GM086852055 They also show that M. malmoense can be isolated from the environment which may be the source of the infection.

BC2GM后面那个数字不是任何ID。我想这是主办方为了公平起见故意将像训练数据的那种ID改成了这样一个ID的，为了隐藏哪些句子出自同一片文章和哪些句子是标题哪些是摘要等信息。可以看出主办方是希望在这个任务里将每个句子单独孤立的看待的。

**二、实验方法和结果**

**（1）字典匹配方法的实验。**

采用的字典是丁石林留下的共有234112条记录的基因蛋白质字典，其格式为：

####1

ID:104K\_THEAN~

AC:Q4U9M9~

DE:p104~104 kDa microneme-rhoptry antigen precursor~

GN:ORF@TA08425~

OS:NCBI@5874~Theileria annulata~

####2

ID:104K\_THEPA~

AC:P15711~Q4N2B5~

DE:p104~104 kDa microneme-rhoptry antigen precursor~

GN:ordered locus@TP04\_0437~

OS:NCBI@5875~Theileria parva~

ID和AC都是它的ID。DE就是它的各种描述方式，GN是基因名字（@后面那个是），另外ID里“\_”前面那个也可以作为一个名字。OS是这个蛋白所在的物种。我和丁石林以前的做法一样，把AC中的第一个作为它的ID。

对于字典中的这些名字和描述信息要进行一些标准化处理，包括转化为小写形式、去掉横杠斜杠等特殊字符、将罗马数字和希腊字母转化成标准形式。在此基础上，对字典中所有词进行一个类似于倒排文档的索引，也就是对于每一个词找出都有哪些条记录含有这个词。例如对于“104K”，它就应该至少对应Q4U9M9和P15711两条记录。

在匹配的过程中，首先对每句话的token做相同的标准化处理，然后对于每一个token，从索引结构中找出含有它的字典记录集合。然后找出相邻而且含有相同字典记录的token串，这些串就可能是蛋白名字，它们之间相同的那些字典记录就是它们对应在字典中的记录。然后再根据这些记录和一个常用词表以及一些简单规则检查一下这个名字，去掉不对的。

字典匹配的结果见表3第一行。可以看到字典匹配加简单过滤规则的效果是不行的，必须使用CRF这样的给予学习的方法。

我把标准化处理后的字典和字典的索引信息都存在了文件中，在程序初始化时可以快速读入。整个匹配过程中，虽然使用的词表很大，但是由于使用了索引结构，速度很快。

**（2）CRF方法的实验**

主要进行的实验是使用CRF进行序列标注的方法。CRF工具包使用的是CRF++，主要是考虑它的速度比较快。

使用的基本的feature包括：

字形类feature，是否首字母大写、是否全大写、是否包含数字、是否是特殊符号等等。这样的feature是通过检查一个token是否符合某一个正则表达式，给出01的结果，如表1所示：

|  |  |  |
| --- | --- | --- |
| 正则表达式 | 说明 | 举例 |
| [A-Z].\* | 大写开头 | Kinase |
| [A-Z][a-z]\* | 大写开头且后面都是小写 | Kinase |
| [A-Z]+ | 多个大写字母 | SOX |
| [a-z]+ | 多个小写字母 | protein |
| [A-Za-z]+ | 仅有英文字母 | gene |
| .\*[0-9].\* | 包含数字 | p53a |
| [\\-0-9]+[\\.\\,]+[0-9]\\.\\,]+ | 浮点数 | 3.14 |
| [A-Za-z0-9]+ | 仅有字母数字 | p23 |
| [ivxdlcm]+|[IVXDLCM]+ | 罗马数字 | iiv |
| .\*[A-Z] | 大写结尾 | kappaB |
| .\*[A-Z].\*[a-z].\*|.\*[a-z].\*[A-Z].\* | 大小写混合 | RalGDS |
| [A-Z] | 一个大写字母 | A |
| [^A-Z]\*[A-Z][^A-Z]\* | 只有中间一个大写字母 | kDa |
| [A-Z]{2} | 两个大写字母 | IL |
| [A-Z]{3} | 三个大写字母 | CSF |
| [A-Z]{4,} | 四个以上大写字母 | RESULT |
| [0-9] | 一个数字 | 1 |
| [0-9]{2} | 两个数字 | 22 |
| [0-9]{3} | 三个数字 | 234 |
| [0-9]{4,} | 四个以上数字 | 234353 |
| [\\,\\.\\;\\:\\?\\!\\-\\+\\'\\\"\\'] | 特殊字符 | . |
| [A-Z]|[a-z] | 一个字母 | a |
| [ATCGU]+|[atcgu]+ | 碱基序列 | ACTGACT |

表1 字形特征

词表类feature，按照一些词表，查找一个token是否是核酸、氨基酸、希腊字母等等。通过检查一个token是否在一个词表中存在，给出01的结果。如表2所示。

|  |  |
| --- | --- |
| 词表 | 举例 |
| 希腊字母 | beta |
| 氨基酸 | tyrosine |
| 氨基酸缩写 | Ser |
| 氨基酸缩写+位置 | Ser150 |
| 核甘 | Thymine |
| 核苷酸 | ATP |
| 核酸 | cDNA |

表2 词表特征

其他一些feature，包括：用“a”替换小写字母，“A”替换大写字母，“1”替换数字，将token表示成“Aa1”这样的形式；根据token的长度给出“短中长”这样的等级标签；token本身和它的stem。

以上这些feature主要仿照的是第二名的队伍[2]和[3]。

用CRF对token序列进行BIO标签标注完成后，经过分析发现结果中存在很多的边界错误，而且其中很多都是形如cyclin D只识别出cyclin这样的情况。这样的错误可以用简单的规则很容易地纠正。于是加入一个后处理步骤，向右寻找像“D /1/ E2”这样的标号，扩展右边界，这样的扩展也能处理“cyclin A and D”这样的情况。另外，一些括号也对结果产生不利影响，有些结果中只含有一个左括号而没有右括号，这样的结果将括号及后面的去掉。另外，还有“cyclin A and cycling D”这样的情况，Biocreative数据在这样的情况中标注有很多不一致，有的时候将它标为一个，有的时候拆成两个。我们从训练数据中训练得到的模型倾向于把它标成一个，最后经过权衡，还是决定将它拆成两个。

使用上述方法得到的实验结果如表3第二行所示。

**（3）向CRF中加入字典特征实验**

这里考虑将字典匹配的结果和CRF标注过程进行融合，方法是将字典匹配的结果作为特征加入CRF模型中。我尝试了两种加入方法：一种是对每一个token，根据是否有字典中的记录包含它，给出01的特征（称为字典特征1）；第二种是根据字典匹配的结果，对token序列标上BIO标签，将每个token得到的标签作为一维特征（称为字典特征2）。对加入字典特征后的CRF进行实验的结果见表3后三行。

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F |
| 字典匹配 | 0.4710 | 0.4118 | 0.4393 |
| 无字典特征CRF | 0.8641 | 0.7976 | 0.8295 |
| 加入字典特征1CRF | 0.8664 | 0.8100 | 0.8372 |
| 加入字典特征2CRF | 0.8643 | 0.8043 | 0.8332 |
| 加入两种字典特征CRF | 0.8694 | 0.8041 | 0.8355 |

表3 实验结果

**三、实验结果错误分析**

为了明确系统目前存在的主要问题和改进方向，我对实验结果做了错误分析。主办方给出的标准评测程序可以给出评测的详细信息，能够列出所有的False Positive（FP）和False Negative（FN）。通过观察这些详细信息，可以看到出现的问题主要有以下几类：

1. 左边界错误：

FP|BC2GM098918087|128 146|B1A-thymidine kinase

FN|BC2GM098918087|132 146|thymidine kinase

1. 右边界错误：

FP|BC2GM083962510|227 232|p637OC

FN|BC2GM083962510|227 236|p637OC-luc

1. 双边界错误：

FP|BC2GM091632436|1 12|rare tRNA-Arg

FN|BC2GM091632436|5 17|tRNA-Arg(CCU)

FP|BC2GM040715083|111 139|NADP-malic enzyme-type C4 plants

FN|BC2GM040715083|116 126|malic enzyme

1. 单独的FP和FN：就是不是边界错误，而就是识别错或没有识别到。

我们可以将这几种不同的错误形式从所有错误中分辨出来。同时主办方也提供了所有参赛队的提交结果，我将比我们结果好的7个队的结果做同样的分析，可以进行一些对比。如表4所示：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 右边界 | 左边界 | 双边界 | 边界错误总数 | 单独FP | 单独FN |
| Rank 1 | 116 | 107 | 43 | 266 | 477 | 642 |
| Rank 2 | 133 | 118 | 31 | 282 | 402 | 706 |
| Rank 3 | 92 | 95 | 27 | 214 | 795 | 551 |
| Rank 4 | 135 | 108 | 63 | 306 | 507 | 637 |
| Rank 5 | 108 | 110 | 25 | 243 | 693 | 606 |
| Rank 6 | 105 | 93 | 27 | 225 | 972 | 480 |
| Rank 7 | 141 | 146 | 25 | 312 | 512 | 805 |
| 我们 | 146 | 122 | 35 | 303 | 510 | 950 |

表4 错误分类统计

从表中可以看出，我们与其他较好结果的主要差距在于单独的FN，这也符合我们结果中recall比较低的情况。而在FN中也有不同的情况，我觉得比较重要的是名字中包含token的数量。我们对“单独FN”这一项做进一步的分析，看看在不同token数下FN的数量，如表5所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | >3 |
| Rank 1 | 290 | 151 | 86 | 115 |
| Rank 2 | 362 | 167 | 82 | 95 |
| Rank 3 | 287 | 134 | 60 | 70 |
| Rank 4 | 328 | 135 | 68 | 86 |
| Rank 5 | 320 | 142 | 76 | 68 |
| Rank 6 | 203 | 123 | 68 | 86 |
| Rank 7 | 408 | 212 | 92 | 93 |
| 我们 | 476 | 239 | 103 | 132 |

表5 FN错误按token数分类统计

从上面可以看到，主要的差距在于token较少的FN错误，token越少越明显。我们可以从另一个角度来分析这些结果。结合两个表看，只有一个token的FN越少，全部的单独FN页越少，或者可以说，一个token的FN和多个token的FN，是同增同减的关系，而不是此消彼长的关系。

这说明，对于一个单独的token识别它是否是蛋白质名字或者是其一部分是一个很重要的部分。前面这些参赛队大都使用了CRF，根据我对CRF的理解，我觉得原因是这样的：CRF的高明之处在于综合考虑前后位置的特征信息，这种特性在确定名字边界的时候是比较有优势的，但是CRF非常需要一些feature能够比较准确地定位到序列的某些位置。例如“P. falciparum DHFR active site”中，DHFR这个词是一个很重要的位置，如果根据一些特征能够让CRF确信DHFR的标签应该是B或I而不会是O，那么CRF强大的边界确定功能就会发生作用，DHFR本身的信息也会被周围的位置利用，因而有很大可能成为一个正确结果。其实如果结果的边界稍微有些错误对我们来说也没关系，因为我们最终是要做BioCreative III的GN任务，即使边界有误也很可能能够正确得到ID。

综上所述，我们需要一些好的feature来帮助CRF定位一个个单独的token，在我们的系统里，我们是希望用字典匹配得到的信息来完成这个任务的。但是目前效果并不好，我认为目前主要问题还在于字典上。目前的字典有这样一个问题：里面存储的记录的名字和描述都是一些比较正式的形式，而缺乏这些蛋白质在文章中出现时可能的比较简单的形式。以我们的一个FN错误“PHO”为例，这个我们没有识别出来，但是在我们的字典中搜索PHO，有很多记录包含它：

P15710|pho4~phosphate repressible phosphate permease~pho 4 ncu09564~

Q751E8|pho85~negative regulator of the pho system~serine threonine protein kinase pho85~pho85 agl242c~

Q9HGY5|pho85~capho85~negative regulator of the pho system~serine threonine protein kinase pho85~pho85~

Q6FKD4|pho85~negative regulator of the pho system~serine threonine protein kinase pho85~pho85 cagl0l12474g~

Q6BRY2|pho85~negative regulator of the pho system~serine threonine protein kinase pho85~pho85 deha0d14256g~

Q92241|pho85~negative regulator of the pho system~serine threonine protein kinase pho85~pho85 klla0d11990g~

Q6C7U8|pho85~negative regulator of the pho system~serine threonine protein kinase pho85~pho85 yali0d25190g~

P17157|pho85~negative regulator of the pho system~serine threonine protein kinase pho85~pho85 ypl031c~

这里用“~”分隔不同的名字和描述。可以看到虽然有很多有PHO，但是没有一条记录有PHO这么一个单独的名字，像这样：

P15710|pho4~phosphate repressible phosphate permease~pho 4 ncu09564~pho~

在这种情况下，在字典匹配的最后，面对pho这样，有记录包含它，却在字典中不是一个单独名字的情况，有两种选择：将它们都算作是一个结果，那么precision会下降；把它们扔掉，recall会下降。其实在加入CRF中的两个字典特征中，特征1就是提供类似于前一种的特征信息，只要这个token有记录含有它，就是1，然而由于准确率太低，CRF经过学习就会认为这一维特征是“不值得信任的”，因而也起不了太大作用；而特征2所用的字典匹配结果则是通过后一种策略得到的。

统计一下所有的单独FN中，有多少其中至少一个token在字典中是出现了的，结果见表6。从结果中可以看到，我们使用的字典覆盖率是很高的。

|  |  |  |
| --- | --- | --- |
| Token数 | 至少有一个token在字典中的单独FN数 | 单独FN数 |
| 1 | 339 | 476 |
| 2 | 216 | 239 |
| 3 | 100 | 103 |
| >3 | 129 | 132 |
| 全部 | 784 | 950 |

表6 单独FN被字典覆盖的统计

这样看来我觉得我们要提高性能，需要做的工作就应该从这个方面入手，就是提供给CRF更能准确指示出蛋白质名字中比较核心的那么1、2个词的特征。我们可以通过改进字典的质量和字典匹配的策略来完成这一目标，或者寻找其他更好的特征。

对于影响Precision的单独FP错误也做一下类似的分析。按照token数统计一下数量，结果见表7。由这个结果可以看出，我们对于一个单独的token是否是蛋白质的识别正确率还是可以的。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | >3 |
| Rank 1 | 274 | 146 | 34 | 23 |
| Rank 2 | 167 | 135 | 55 | 45 |
| Rank 3 | 283 | 227 | 138 | 147 |
| Rank 4 | 282 | 133 | 45 | 47 |
| Rank 5 | 317 | 230 | 87 | 59 |
| Rank 6 | 495 | 244 | 130 | 103 |
| Rank 7 | 224 | 147 | 80 | 61 |
| 我们 | 213 | 166 | 81 | 50 |

对于FP中被字典覆盖的情况，也统计不同token数的FP中至少有一个token在字典出现的个数，结果见表6。可以看到识别错误的结果被字典覆盖率非常高，这也说明了我们字典过于广泛，正确率不足，这首先将导致字典特征在学习的过程中会被模型认为不应信任，进而对结果产生不了很大的指导意义，同时也会导致一定的FP错误。

|  |  |  |
| --- | --- | --- |
| Token数 | 至少有一个token在字典中的单独FN数 | 单独FN数 |
| 1 | 171 | 213 |
| 2 | 151 | 166 |
| 3 | 79 | 81 |
| >3 | 48 | 50 |
| 全部 | 449 | 510 |

表6 单独FP被字典覆盖的统计

**参考文献**

[1] Larry Smith. Overview of BioCreative II gene mention recognition.

[2] Cheng-Ju Kuo. Rich Feature Set, Uniﬁcation of Bidirectional Parsing and Dictionary Filtering for High F-Score Gene Mention Tagging.

[3] Ryan McDonald and Fernando Pereira. Identifying gene and protein mentions in text using conditional random fields.