二．词汇与分词

2.1词汇

什么是 词；

2.2 分词算法

从20世纪80年代或更早的时候起，学者们研究了很多的分词方法，这些方法大致可以分为三大类：

 基于词表的分词方法

* 正向最大匹配法(forward maximum matching method, FMM)
* 逆向最大匹配法(backward maximum matching method, BMM)
* N-最短路径方法

 基于统计模型的分词方法

* 基于N-gram语言模型的分词方法

 基于序列标注的分词方法

* 基于HMM的分词方法
* 基于CRF的分词方法
* 基于词感知机的分词方法
* 基于深度学习的端到端的分词方法
* **各分词方法的细节**
* **正向最大匹配法(FMM)**



正向最大匹配法，顾名思义，对于输入的一段文本从左至右、以**贪心**的方式切分出当前位置上**长度最大的词**（根据已知词典）。正向最大匹配法是基于词典的分词方法，其分词原理是：单词的颗粒度越大，所能表示的含义越确切。

## 负向最大匹配法(BMM)

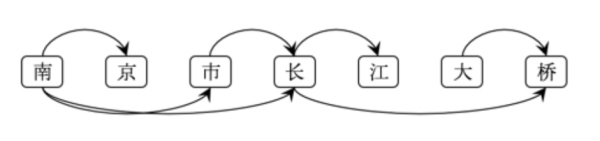


反向最大匹配法的基本原理与正向最大匹配法类似，只是分词顺序变为从右至左。容易看出，FMM或BMM对于一些有歧义的词处理能力一般。举个例子：结婚的和尚未结婚的，使用FMM很可能分成结婚/的/和尚/未/结婚/的；为人民办公益,使用BMM可能会分成为人/民办/公益。

虽然在部分文献和软件实现中指出，由于中文的性质，反向最大匹配法优于正向最大匹配法。在成熟的工业界应用上几乎不会直接使用FMM、BMM作为分词模块的实现方法。

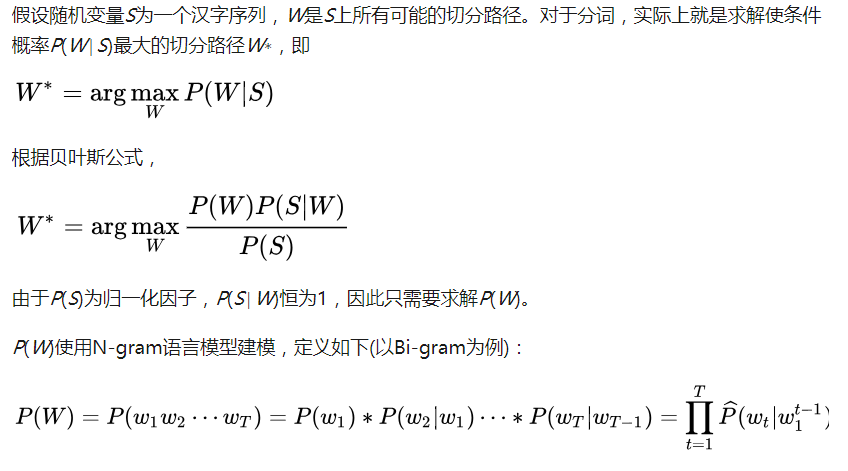
## 基于N-gram语言模型的分词方法

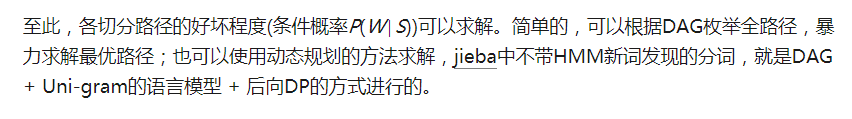
由于歧义的存在，一段文本存在多种可能的切分结果（切分路径），FMM、BMM使用机械规则的方法选择最优路径，而N-gram语言模型分词方法则是利用统计信息找出一条概率最大的路径。



上图为南京市长江大桥的全切分有向无环图(DAG)。可以看到，可能的切分路径有：

* 南京/市/长江/大桥
* 南京/市/长江大桥
* 南京市/长江/大桥
* 南京市/长江大桥
* 南京/市长/江/大桥
* 南京/市长/江大桥
* 南京市长/江/大桥
* 南京市长/江大桥
* …



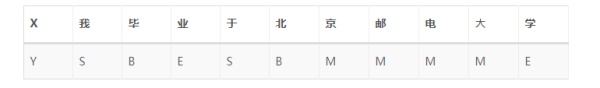
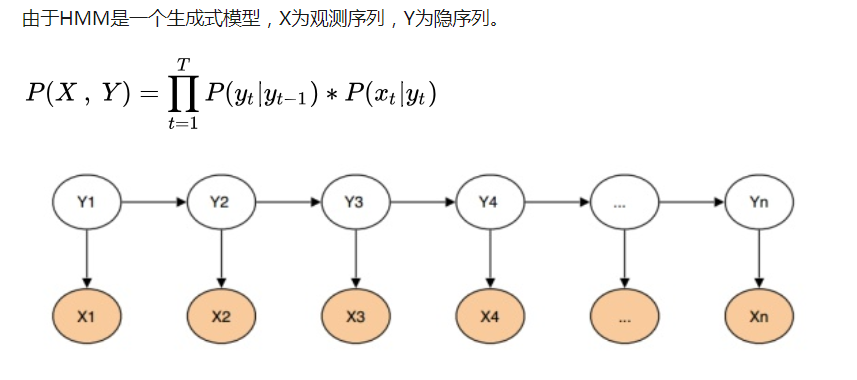


**基于HMM的分词方法**

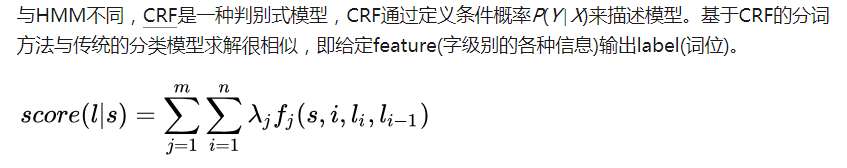
接下来介绍的几种分词方法都属于由字构词的分词方法，由字构词的分词方法思想并不复杂，它是将分词问题转化为字的分类问题（序列标注问题）。从某些层面讲，由字构词的方法并不依赖于事先编制好的词表，但仍然需要分好词的训练语料。

规定每个字有4个词位：

* 词首 B
* 词中 M
* 词尾 E
* 单字成词 S

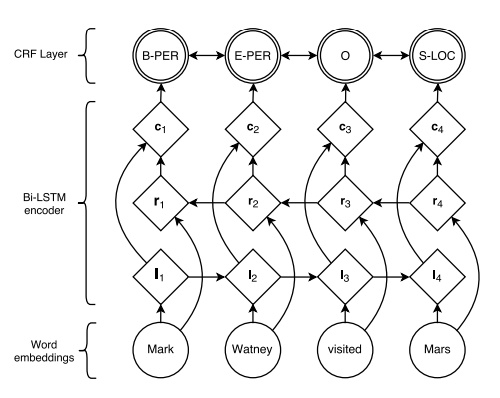
  
  
对应HMM 解码问题，模型参数(转移矩阵，发射矩阵)可以使用统计方法计算得到，原始文本为输出序列，词位是隐状态序列，使用Viterbi算法求解即可。

## 基于CRF的分词方法



## 基于深度学习的端到端的分词方法

最近，基于深度神经网络的序列标注算法在词性标注、命名实体识别问题上取得了优秀的进展。词性标注、命名实体识别都属于序列标注问题，这些端到端的方法可以迁移到分词问题上，免去CRF的特征模板配置问题。但与所有深度学习的方法一样，它需要较大的训练语料才能体现优势。



2.3 词典结构

2.4 新词发现

在中文分词领域里，还有一个比分词歧义更令人头疼的东西——未登录词。中文没有首字母大写，专名号也被取消了，这叫计算机如何辨认人名地名之类的东西？更惨的则是机构名、品牌名、专业名词、缩略语、网络新词等等，它们的产生机制似乎完全无规律可寻。最近十年来，中文分词领域都在集中攻克这一难关。自动发现新词成为了关键的环节。

 挖掘新词的传统方法是，先对文本进行分词，然后猜测未能成功匹配的剩余片段就是新词。这似乎陷入了一个怪圈：分词的准确性本身就依赖于词库的完整性，如果词库中根本没有新词，我们又怎么能信任分词结果呢？此时，一种大胆的想法是，首先不依赖于任何已有的词库，仅仅根据词的共同特征，将一段大规模语料中可能成词的文本片段全部提取出来，不管它是新词还是旧词。然后，再把所有抽出来的词和已有词库进行比较，不就能找出新词了吗？

要想从一段文本中抽出词来，我们的第一个问题就是，怎样的文本片段才算一个词？大家想到的第一个标准或许是，看这个文本片段出现的次数是否足够多。我们可以把所有出现频数超过某个阈值的片段提取出来，作为该语料中的词汇输出。不过，光是出现频数高还不够，一个经常出现的文本片段有可能不是一个词，而是多个词构成的词组。在人人网用户状态中，“的电影”出现了 389 次，“电影院”只出现了 175 次，然而我们却更倾向于把“电影院”当作一个词，因为直觉上看，“电影”和“院”凝固得更紧一些。

 为了证明“电影院”一词的内部凝固程度确实很高，我们可以计算一下，如果“电影”和“院”真的是各自独立地在文本中随机出现，它俩正好拼到一起的概率会有多小。在整个 2400 万字的数据中，“电影”一共出现了 2774 次，出现的概率约为 0.000113 。“院”字则出现了 4797 次，出现的概率约为 0.0001969 。如果两者之间真的毫无关系，它们恰好拼在了一起的概率就应该是 0.000113 × 0.0001969 ，约为 2.223 × 10-8 次方。但事实上，“电影院”在语料中一共出现了 175 次，出现概率约为 7.183 × 10-6 次方，是预测值的 300 多倍。类似地，统计可得“的”字的出现概率约为 0.0166 ，因而“的”和“电影”随机组合到了一起的理论概率值为 0.0166 × 0.000113 ，约为 1.875 × 10-6 ，这与“的电影”出现的真实概率很接近——真实概率约为 1.6 × 10-5 次方，是预测值的 8.5 倍。计算结果表明，“电影院”更可能是一个有意义的搭配，而“的电影”则更像是“的”和“电影”这两个成分偶然拼到一起的。

令 p(x) 为文本片段 x 在整个语料中出现的概率，那么我们定义“电影院”的凝合程度就是 p(电影院) 与 p(电) · p(影院) 比值和 p(电影院) 与 p(电影) · p(院) 的比值中的较小值，“的电影”的凝合程度则是 p(的电影) 分别除以 p(的) · p(电影) 和 p(的电) · p(影) 所得的商的较小值。

文本片段的自由运用程度也是判断它是否成词的重要标准。如果一个文本片段能够算作一个词的话，它应该能够灵活地出现在各种不同的环境中，具有非常丰富的左邻字集合和右邻字集合。

我们用信息熵来衡量一个文本片段的左邻字集合和右邻字集合有多随机。考虑这么一句话“吃葡萄不吐葡萄皮不吃葡萄倒吐葡萄皮”，“葡萄”一词出现了四次，其中左邻字分别为 {吃, 吐, 吃, 吐} ，右邻字分别为 {不, 皮, 倒, 皮} 。根据公式，“葡萄”一词的左邻字的信息熵为 - (1/2) · log(1/2) - (1/2) · log(1/2) ≈ 0.693 ，它的右邻字的信息熵则为 - (1/2) · log(1/2) - (1/4) · log(1/4) - (1/4) · log(1/4) ≈ 1.04 。可见，在这个句子中，“葡萄”一词的右邻字更加丰富一些。

我们不妨就把一个文本片段的自由运用程度定义为它的左邻字信息熵和右邻字信息熵中的较小值。

 在实际运用中你会发现，文本片段的凝固程度和自由程度，两种判断标准缺一不可。只看凝固程度的话，程序会找出“巧克”、“俄罗”、“颜六色”、“柴可夫”等实际上是“半个词”的片段；只看自由程度的话，程序则会把“吃了一顿”、“看了一遍”、“睡了一晚”、“去了一趟”中的“了一”提取出来，因为它的左右邻字都太丰富了。

我们把文本中出现过的所有长度不超过 d 的子串都当作潜在的词（即候选词，其中 d 为自己设定的候选词长度上限，设定的值为 5 ），再为出现频数、凝固程度和自由程度各设定一个阈值，然后只需要提取出所有满足阈值要求的候选词即可。

2.5 实现

Xincifaxian

五．

词性，语块与命名实体

5.1 词性标注

Seq-lstm

5.2 短语识别

5.3 命名实体识别

5.4 实现