基于统计的词素切分算法

董兴华[1,2]，杨雅婷[1,2]，陈丽娟[1,2]，周喜[1]，吐尔洪·吾司曼[1]

（1. 中国科学院 新疆理化技术研究所，乌鲁木齐 830011； 2. 中国科学院研究生院，100190 ）

（dongxinghua0213@sina.com yangyt_xj@sina.com chenlijuanyx@tom.cn zhouxi@ms.xjb.ac.cn turghunjan@sina.com ）

摘 要：这篇论文描述了一种基于统计的词素切分算法，算法构建了一种数据结构，在该结构中语料库中的每个词都可以表示为它的词素的二叉树。因为每个词有不同的词素分割，算法选择使整体概率最高的分割，从而找到最优的词素词典和词的分割。我们用英语和维吾尔语作为实验数据，得出了较好的结果。

关键词：词素；统计分割；二叉树；维语

A Statistical Approach To Morpheme Segmentation

Dong Xinghua[1,2] , Yang Yating[1,2],Chen Lijuan[1,2],Zhou Xi[1],Turghun Osman[1]

(1.*Xingjiang Technical Institute of Physics & Chemistry, Chinese Academic of Science ,Urumqi 830011,China; 2.Graduate University Of Chinese Academic of Science, Urumqi 830011, China*)

Abstract: This dissertation describes a statistical segmentation algorithm for morpheme , the algorithm makes use of a data structure, where each distinct word form in the corpus has its own binary splitting tree. For there are different ways in which a word can be segmented, the segmentation yielding the highest probability is selected, then we can find the optimal morph lexicon and segmentation for every word. The experiments show a better result for English and Uyghur.

Key words: morpheme; statistical segmentation; binary splitting tree; Uyghur

1 引言

很多语言处理任务，如语音识别，机器翻译，信息检索等把词作为最基本的语言表示单位。例如，在信息检索中，我们把用户输入的信息借助我们事先收集的词表切分成若干关键字，使之和特定的文本内容相匹配从而检索出用户所需的信息；在统计机器翻译解码过程中，把词或短语作为翻译的基本单位。但是把词作为语言的基本的表示单位并不适用于形态变化较丰富的语言，如维吾尔语，土耳其语，芬兰语等。据统计，维吾尔语词根有三万多个，后缀有 100 多个，前缀有几十个，我们不可能收集到所有的词目，在这些语言处理任务中，有时可以把对词缀和词根（词干）的处理作为对词的处理的一种替代方案。

笔者主要参考了赫尔辛基大学开发的 morfessor 切分算法[1]，从目标语言的单语料库中推导出这种语言的词素（包括词干、前缀和后缀等词素或者近似词素）组成并用

基金项目：中国科学院“西部行动计划高新技术项目”（KGCX2-YN-507）

作者简介：董兴华(1982－)，男，博士研究生，主要研究方向：自然语言处理；杨雅婷(1985－)，女，博士生，主要研究方向：语音识别；陈丽娟(1985－)女，硕士研究生，主要研究方向：自然语言处理；周喜(1978－)，男，副研究员，主要研究方向：多语种信息处理；吐尔洪·吾司曼(1980－)，男，维吾尔族，男，乌鲁木齐人，助理研究员，主要研究领域为多语种信息处理。

于词的分割中。

2 统计模型的数学表示

一种语言一般是由组成这种语言的形态符号和语法规则组成，其中，语法规则用来描述形态符号的组合关系，我们的目标是从特定语言的目标语料库中找到一个模型 μ 来模拟这种语言的组成及语法功能，因此，这个模型应该包含这种语言的词素词典（*lexion*）和语法(*grammar*)。为了精确的描述这种语言，这个模型应该是最优的，根据贝叶斯公式：

$$\arg\max_{\mu} P(\mu \mid corpus) = \arg\max P(corpus \mid \mu) P(\mu) / P(corpus) \tag{1}$$

在这里 $P(corpus)$是常数，可以忽略这一项，根据以上描述，$P(\mu) = P(lexion, grammer)$ (2)
从等式（1）可以看出，分子分为两部分：语言的模型概率和在该模型确定的情况下语料库的概率，而等式(2)则说明语言的模型概率是这种语言的词典和语法的联合概率。下面通过词典、语法和语料库的在模型中的表示来描述其数学模型。

2.1 词典

词典包含语料库中出现的不同的词素，如果一个词典包含 M 个不同的词素 $\mu_1, \mu_2 \cdots \mu_M$，那么这个词典的概率为：

$$P(lexion) = M! P(properties(\mu_1), \cdots, properties(\mu_M)) \tag{3}$$

这意味着词典可以表示成各个词素属性集合的联合概率。等式(3)包含因子 M! 是因为在包含 M 个不同的词素词典中，词素有 M! 种不同的排列的顺序，不管以何种顺序出现，词典是相同的。为了使模型简化，我们认为词素的属性 $properties(\mu)$ 只和样本语料库中词素出现的频率和组成词素的字母相关。假设词素的频率和字母组成是独立的，我们用 f 表示词素的频率，s 表示组成词素字母串，我们可以得出如下的表示：

$$P(properties(\mu_1), \cdots properties(\mu_M)) = P(f_{\mu 1}, \cdots, f_{\mu M}) P(s_{\mu 1}, \cdots, s_{\mu M}) \tag{4}$$

$P(f_{\mu 1} \cdots f_{\mu M})$ 为词素频率的概率，一种计算它的方法是我们把它作为一个整体来考虑，假设在样本语料库中有 M 个不同的词素，而词素的总数为 N，问题转化为求一种特定频率分布的概率。我们假定所有的频率分布是等可能，那么一种特定频率分布的概率是用 1 除以 M 个正整数（M 个不同的频率）加起来等于 N 的方法数。如果我们把 N 个词素按照字母表的顺序排列，每一个词素可以表示成一个二进制数（用 0 和 1 表示）。我们把频率大于 1 的同一词素放在同一行中，起初我们把这 N 个位全部初始化为 0，在词素发生改变的地方我们把它的位变为 1，而每个词素没有发生变化的地方（和以前的词素等同）保持它的位不变。在 N 个位中要把 M 个位转化为 1 共有 $\binom{N}{M}$ 种方法，因为第一个位确定转化为 1，忽略它总共的方法为 $\binom{N-1}{M-1}$，因此频率分布的概率为：

$$P(f_{\mu 1},\ldots,f_{\mu M}) = 1/\binom{N-1}{M-1} = \frac{(M-1)!\,(N-M)!}{(N-1)!} \tag{5}$$

另一种计算词素频率分布概率的方法是为每一个词素的概率分别分配一个概率。我们假定词素的频率分布是独立的，因此有：

$$P(f_{\mu 1},\ldots,f_{\mu M}) = \prod_{i=1}^{M} P(f_{\mu i}) \tag{6}$$

$P(f_{\mu i})$ 的推导过程主要用到了 Zipf[2]定律，得到的表达形式如下：

$$P(f_{\mu i}) = f_{\mu i}^{\log 2(1-h)} - (f_{\mu i}+1)^{\log 2(1-h)} \tag{7}$$

其中 h 为语料库中只出现一次的词的概率。
在实验中我们发现这两种计算词素频率分布概率的方法差别不大。

我们假定一个词素包含的字符串与另一个词素包含的字符串是相互独立的，则等式（4）中的因子 $P(s_{\mu 1},\ldots,s_{\mu M})$ 则变为每个词素字符串概率的乘积：

$$P(s_{\mu 1},\ldots,s_{\mu M}) = \prod_{i=1}^{M} P(s_{\mu i}) \tag{8}$$

进一步假定词素字符串中的字母也是相互独立的，那么词素字符串的概率就转化为字符串各个字母概率的乘积：

$$P(s_{\mu i}) = \prod_{j=1}^{l_{\mu i}} P(c_{i,j}) \tag{9}$$

在等式（9）中 $l_{\mu i}$ 表示词素 μ_i 的长度，$P(c_{i,j})$ 可以通过样本语料库中各个字母的相对频率来计算。

对词素长度 $l_{\mu i}$ 的评估有两种方法，一种方法是假定词典中的每个词素都有一个结尾符#，并且结尾符是字母表的一部分，词素长度为 l 的概率可以通过下式计算：

$$P(l) = [1-P(\#)]^{l} \cdot P(\#) \tag{10}$$

在等式（10）中，$P(\#)$ 是语料库中词素结尾符的概率，等式右边的乘积可以解释为首先选择除词素结尾符（#）l 个字母，然后选择结尾符的概率。由于这是一个指数分布，词素的长度 l 的概率随着 l 的增加而降低。另一种方法是应用 gamma 分布或者泊松分布分别计算词典中的词素长度概率[3]。

2.2 语法

语法可以被看做语言单元相互结合的规则，为了简化模型，暂时不考虑语法，那么 $P(lexion, grammar)$ 变为 $P(lexion)$，对一个词素来说，缺少语法信息意味着出现在它前面或后面的任词素概率都是等概率的，而且这个词素在一个词中出现的位置也是等概率的。当然也可以建立更为复杂的数学模型来模拟这些语法规则。

2.3 语料库

语料库中的每个词形都可以看做词典中词素的序列。一个词可能有很多种不同的

分割，我们的目标是找最可能的分割。在语言的模型确定的情况下语料库发生的概率可以被表达为：

$$P(corpus \mid \mu) = \prod_{j=1}^{W} \prod_{k=1}^{n_j} P(\mu_{j_k}) \tag{11}$$

W 为语料库中单词的个数，n_j 为第 j 个单词词素的个数，$P(\mu_{j_k})$ 为第 j 个单词第 k 个词素的概率，可以通过下式来计算：

$$P(\mu_j) = \frac{f_{\mu j}}{N} = \frac{f_{\mu j}}{\sum_{j=1}^{M} f_{\mu j}} \tag{12}$$

等式（12）中 N 为 M 个不同的词素的和。

3 搜索算法

我们用贪婪算法来寻找最优的词典和每个词的分割。最初，语料库中每一个词作为一个词素，因为一个词可能有很多不同的分割，根据以上的描述，模型选择使模型概率最大的分割作为最优分割，并把相应的词素保存在词典中。

因为计算机对数据存储的精度是有限的，在操作中，我们用对数的形式来存储概率，这样概率乘积的计算就变成了概率的和。

搜索算法主要利用一种数据结构，在该数据结构中，语料库中每个不同的词形被表示成二叉树。如图 1 是英语单词”reopened”和“openminded”的二叉树。

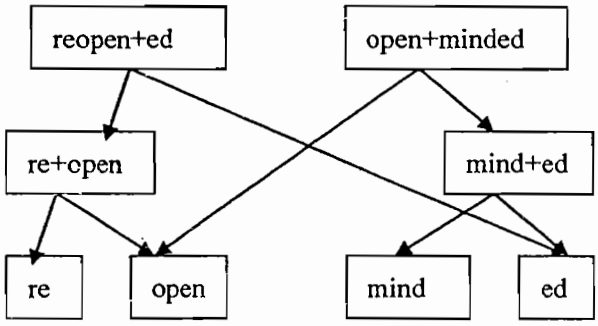

图 1　英语单词 reopened 和 openminded 的二叉树表示

二进制树的叶节点不能被分割，它们被存储在词典中，叶节点是唯一对模型的概率有影响的节点，其余的节点仅仅用在搜素的过程中。每个节点的数据结构中也存储着当前单词或者字串在语料库中出现的频率。子节点中存储的频率总是等于它的父节点的和。例如在图 1 中，词素 open 的频率等于其父节点 reopen 和 openminded 之和。搜素算法的伪代码如下：

```
split(node) //node 表示一个单词或或一个单词的字串
{
        //移除当前节点 node 的数据结构表示和数值表示//
```

如果 node 已经存在数据结构表示，对 node 子树中的所有节点 m 执行以下循环

{

count(m)=count(m)-count(node);//count(m)表示节点中数据的频率

如果当前节点是叶节点，则

从 $L(corpus \mid \mu)$ 和 $L(f_{\mu 1}, \ldots, f_{\mu M})$ 中扣除其相应的“贡献率”。

如果 count(m)为 0，则

从数据结构中移除 m;

如果 m 是叶节点则从 $L(s_{\mu 1}, \ldots, s_{\mu M})$ 中移除相应的“贡献率”;

}

//把当前单词本身当做词素添加到词典中//

把当前节点 node 和其相应频率作为叶节点存储到数据结构中;

把其相应的“贡献率”增加到 $L(corpus \mid \mu)$、$L(f_{\mu 1}, \ldots, f_{\mu M})$ 和 $L(s_{\mu 1}, \ldots, s_{\mu M})$ 中;

把 $[L(\mu \mid corpus), node]$ 作为最优的”解决方案“，并把其相应的概率存储在 bestsolution 中;

//把当前节点分成两个字串，并记录其概率最大的分割方式//

从 $corpus \quad (\mu \mid corpus)$ 中扣除 node 的贡献率，但 node 节点任然保存在数据结构中;

保存现在的 $corpus \quad (\mu \mid corpus)$ 和数据结构;

对所有的子串 pre 和 suf，如果满足 pre o suf =node,则 //o 表示“联合“

{

对[pre , suf]中的任何一个节点 subnode,执行以下循环:

{

若果 subnode 在数据结构中，则对 subnode 子树中的所有节点 m,执行以下循环:

{

count(m)=count(m)+count(node);

如果 m 是叶节点，把其相应的“贡献率”增加到 $L(corpus \mid \mu)$、$L(f_{\mu 1}, \ldots, f_{\mu M})$ 中。

}

否则，执行

{

把 subnode 和其相应的频率数增加到数据结构中;

把其相应的“贡献率”增加到 $L(corpus \mid \mu)$、$L(f_{\mu 1}, \ldots, f_{\mu M})$ 和 $L(s_{\mu 1}, \ldots, s_{\mu M})$ 中。

}

如果 $corpus \quad (\mu \mid corpus) > bestsolution$, 则把 $[L(\mu \mid corpus), pre, suf]$ 作为最优的”解决方案“，并把其相应的概率存储在 bestsolution 中;

恢复存储的数据结构和 $corpus \quad (\mu \mid corpus)$;

}

```
}
        //选择最优的分割（或者没有分割）//
        选择产生 bestsolution 的分割（或者没有分割），并相应地更新数据结构和
corpus  (μ | corpus  ):
        如果一个满足 pre  o  suf  =node 的分割被选择，则:
        使 node 节点为 pre 和 suf 的父节点;
        //继续分割子串//
        splitnode(pre);
        splitnode(suf);
}
```

在搜索过程中，语料库中所有不同的词串被随机的传递给 split 函数，产生当前词串的二叉树。在 *split* 函数中，首先词串本身作为一个词串被添加到词典中。然后对组成当前词的两个子串进行概率评估，选择使模型概率最高的分割(也可能不分割)。如果一个词被分成两个子串，则迭代地对其子串进行分割，当把子串分为更小的串不再使模型的概率提高时，停止分割。在语料库中所有不同的词被处理一次之后，它们再一次被传递到 split 函数重新分割，这个过程继续进行下去，直到模型的概率的提高小于某个阈值时停止分割。

4 相关实验

为了对得到的结果进行评测，我们定义如下评测指标：

1. 精度（*precision*）:正确分割的词素数占总的分割的词素之比，即

$$\text{Precision} = \frac{\text{正确分割的形态素数}}{\text{总的分割的形态素数}} \times 100\%$$

2. 召回率（*recall*）:正确分割的词素占总的正确的词素之比，即

$$\text{Recall} = \frac{\text{正确分割的形态素}}{\text{总的正确的形态素之比}} \times 100\%$$

评测可以分为按符号（token）评测和按类型（type）评测；如果按符号评测，词频高的词对评测结果影响较大，如果按类型评测，则每个不同的词形对评测的影响是相同的。我们在试验中用到的汉语和维吾尔语的语料如下：

表 2 训练预料信息

	句子数	符号数	类型数
英语	25 万	4355212	132329
维吾尔语	8 万+35 万维语词典	2472476	223903

按符号评测，结果如下：

表 3 按符号评测结果

	精度	召回率
英语	81.2%	76.5%
维吾尔语	72.4%	63.4%

按类型评测：

表 4 按类型评测结果

	精度	召回率
英语	75.5%	72.3%
维吾尔语	65.5%	62.3%

查看切分结果时我们发现英语种很多错误主要发生在英语的转化词和专有名词上，如”cities”被切分为“citie+s”，“Mary”被切分为“Mar+y”，另一个主要原因是组成该词的子词在训练语料库中出现的频率较高引起切分错误，如“area”被切分为“are+a”，”team”被切分为“tea+m”，其中单词“a”和“tea”在语料库中出现的频率相对较高。除英语中出现的相应错误外，维语切分结果中另一类错误主要是外来词的错误切分。例如“ئامېيلوفا”被切分为“ئام + يېلو + فا”，“تاربنتىن”被切分为“تا + ربن + تىن”。外来词作为一个独立的词不应该切分。

5 结束语

本文介绍了一种基于统计的词素分割算法，并以英语和维吾尔语作为训练语料给出了实验数据和实验结果。文章中的数学模型没有考虑语法规则，但可以建立更为复杂的数学模型来模拟这些规则，例如可以建立数学模型来模拟不同的词素之间的连接关系和词素在词中的位置。

参考文献

[1] Mathias Creutz and Krista Lagus. 2005. Unsupervised Morpheme Segmentation and Morphology Induction from Text Corpora Using Morfessor 1.0. Publications in Computer and Information Science, Report A81, Helsinki University of Technology, March. URL:http://www.cis.hut.fi/projects/morpho/

[2] GK Zipf, Human Behavior and the Principle of Least Effort (Addison-Wesley, 1949).

[3] Mathias Creutz. 2003. Unsupervised segmentation of words using prior distributions of morph length and frequency. In Proc. ACL'03, pages 280–287, Sapporo, Japan.