二元语法中文分词数据平滑算法性能研究

刘 丹,方卫国,周 泓
LIU Dan,FANG Wei-guo,ZHOU Hong

北京航空航天大学 经济管理学院,北京 100191
School of Economy and Management,Beihang University,Beijing 100191,China
E-mail:newdan@gmail.com

LIU Dan,FANG Wei-guo,ZHOU Hong.Performance of smoothing algorithm in Chinese word segmentation by bigram. Computer Engineering and Applications,2009,45(17):33-36.

Abstract: This paper discusses the relationships between complexity and real performance based on the corpus of People's Daily of January,1998,compares the performance of multiple smoothing algorithms.The result reveals that additive smoothing is the best with 99.68% on precision,99.7% on recall in close test,and 98.64% on precision,98.74% on recall in open test.
Key words: smoothing;Chinese word segmentation;bigram

摘 要:将多种平滑算法应用于基于二元语法的中文分词,在1998年1月人民日报语料库的基础上,讨论了困惑度和实际分词性能之间的关系,对比分析各平滑算法的实际性能,结果表明,简单的加值平滑算法性能最优,封闭精度、召回率分别为99.68%、99.7%,开放精度、召回率为98.64%、98.74%。
关键词:数据平滑;中文分词;二元语法
DOI:10.3778/j.issn.1002-8331.2009.17.010 **文章编号:**1002-8331(2009)17-0033-04 **文献标识码:**A **中图分类号:**TP391

1 基于二元语法的中文分词模型

句子 S 由字序列$\{C_1,C_2,\cdots,C_t\}$组成,字以不同的方法组合到一起成为词。对 S 的切分可以有多种,S 中的第 j 种切分,称为 W_j,m_j 为 W_j 中词的个数,第 k 个词为 $w_{j,k}$($0<k\leqslant m_j$)。如果所有的词确定,则句子唯一确定,$P(S|W_j)=1$。有:

$$\hat{W}=\arg\max_{W_j}P(W_j|S)=\arg\max_{W_j}P(W_j)P(S|W_j)=\arg\max_{W_j}P(W_j)$$

应用 N 元语法,有:

$$P(W)=\prod_i P(w_i|w_{i-N+1}^{i-1})\propto\sum_i\log(P(w_i|w_{i-N+1}^{i-1}))$$

$$\hat{W}=\arg\max_{W_j}P(W_j)=\arg\max_{W_j}\sum_{i=1}^{l}\log(P(w_i|w_{i-N+1}^{i-1}))$$

分词的过程就是求上述概率公式的最优解。$P(w_i|w_{i-N+1}^{i-1})$可以通过使用训练集的数据计算最大似然估计并使用数据平滑方法求得。

N 值越大,越能反映语言的内在结构关系,但需要大量样本的支持[1]。本研究中,分词模型采用的语料库总词数为904 110,切分词典词条数为119 713,Unigram 数仅为55 102,仅占46.02%,存在明显数据稀疏现象,所以选取二元语法比较合适。对于二元语法,有:

$$\hat{W}=\arg\max_{W_j}\sum_{i=1}^{l}\log(P(w_i|w_{i-1}))$$

2 数据平滑方法

受限于训练模型的语料库规模,在语料库中未出现的语法并不一定意味着其真实概率等于0,而出现数量相同的语法概率并不一定一样;此外,零概率会向下传播,导致 $P(W)=0$,不便于计算$\hat{W}$。因此引入数据平滑技术来解决 N 元语法的数据稀疏问题,调整最大似然估计的概率值,使零概率增值,使非零概率下降,消除零概率,使分布趋向更加均匀,提高模型的整体正确率。

目前常用的数据平滑算法有加值平滑、绝对折扣法、Good-Turing、Katz Back-off、Jelinek-Mercer、One Count[2]、SGT(Simple Good-Turing)算法[3]等。

定义某词对出现的频率为 r,定义 r 的频率为 n_r。那么样本数据大小 $N=\sum_{r=1}^{\infty}n_r r$。语料的词汇量为 V。词对在训练集中未出现,则 $r=0$,n_1 是只出现过一次的词对的频率。

2.1 加值平滑法

加值平滑(Additive Smoothing)的基本思想是:给每一种情

基金项目:国家自然科学基金(the National Natural Science Foundation of China under Grant No.70521001)。
作者简介:刘丹(1983-),男,博士研究生,主要研究领域为知识管理,自然语言处理;方卫国(1969-),男,博士,教授,博士生导师,主要研究领域为知识管理,决策支持系统;周泓(1965-),男,博士,教授,博士生导师,主要研究领域为仿真与知识管理。
收稿日期:2009-02-02 **修回日期:**2009-03-09

况出现的次数加上一个值 δ,$0<\delta\leqslant 1$。对于 Bigram 有:

$$P(w_i|w_{i-1})=\frac{\delta+C(w_{i-1}w_i)}{V\times\delta+\sum_{w_i}C(w_{i-1}w_i)}$$

Lidstone 和 Jefferys 认为 $\delta=1$。Gale、Church 提出这种平滑算法性能很差[2]。

2.2 绝对折扣法

绝对折扣法(Absolute Discounting)原理是从每个计数 r 中减去同样的量,剩余的概率量由未见事件均分:

$$P(r)=\begin{cases}\dfrac{r-b}{N} & r>0\\ \dfrac{b(K-n_0)}{N\times n_0} & r=0\end{cases}$$

n_0为样本中未出现的事件的数目,减去的常量 $b\leqslant\dfrac{n_1}{n_1+2\times n_2}<1$,实际运用中,常用上限$\dfrac{n_1}{n_1+2\times n_2}$代替优化的 b[4]。

2.3 Good-Turing 平滑

Good-Turing 平滑,基本原理是对于在语料库中出现 r 次的事件,用 r^* 来估算概率。所有事件发生的总数不变,有 $N=\sum_{r=0}^{\infty}n_r r^*=\sum_{r=0}^{\infty}(r+1)n_{r+1}$,所以 $r^*=\dfrac{(r+1)n_{r+1}}{n_r}$,$P(r)=\dfrac{r^*}{N}$。

Good-Turing 通常不单独作为 N-gram 的平滑算法,而作为其他平滑算法的计算工具[2]。

2.4 Katz 平滑

Katz 平滑是 Good-Turing 的改进,其思想是出现零概率时,用低阶的$(n-1)$gram 的概率替代 N-gram 概率,非零概率的折扣率 d_r 基于 Good-Turing 估计。而且 Katz 观察到 r 越大时,可能有 $n_r=0$,导致 $r^*=0$,出现零概率,所以 r 大的概率不折扣,有

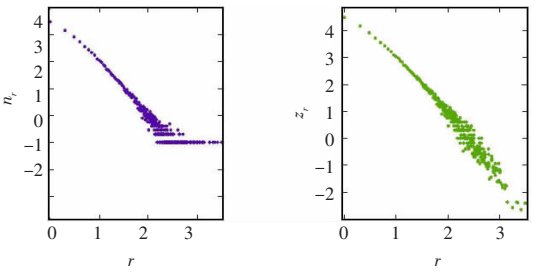
$$\hat{P}_{\text{Katz}}(r)=\begin{cases}\hat{P}_{ML}(w_i|w_{i-1}) & r>k\\ d_r\hat{P}_{ML}(w_i|w_{i-1}) & 0<r\leqslant k\\ a(w_{i-1})\hat{P}_{ML}(w_i) & r=0\end{cases}$$

且 $d_r=\dfrac{\dfrac{r^*}{r}-\dfrac{(k+1)n_{k+1}}{n_1}}{1-\dfrac{(k+1)n_{k+1}}{n_1}}$,$a(w_{i-1})=\dfrac{1-\sum_{w_i:c(w_i^{i-1})>0}p_{\text{Katz}}(w_i|w_{i-1})}{1-\sum_{w_i:c(w_i^{i-1})>0}p_{ML}(w_i)}$,Katz 建议 $k=5$。

2.5 SGT 平滑算法

SGT 平滑算法[3]也是基于 Good-Turing 算法。当 r 较小时,Good-Turing 的性能很好,直接使用 Good-Turing。r 较大时,采用线性 Good-Turing(LGT,Linear Good Turing)平滑来估算 r^*。

在 r 较大时,可能存在 $n_r=0$,也需要计算这些 $n_r=0$ 的值,根据文[5]描述的算法,用非零 n_r 的周围的值取平均来估 n_r,n_r 的序号为 r,其周围连续的非零序号分别为 q、r、t,用 $z_r=\dfrac{2n_r}{t-q}$来代替 n_r。应用本文的训练集进行分析,图 1 显示了应用这种替换之后的结果,图 1(a)是 $\log(n_r)$和 $\log(r)$的关系,r 较大时,大量的 $n_r=1$ 的点导致图形偏向右侧。图 1(b)描述的是 $\log(z_r)$和 $\log(r)$的关系,大的 r 依然保持了线性的趋势。



(a)$\log(n_r)$与 $\log(r)$的关系　　(b)$\log(z_r)$与 $\log(r)$的关系

图 1　本文使用的训练语料的 n_r 和 z_r 对比

观察图 1(b),可以用直线对 n_r 做拟合,$\log(n_r)=a+b\log(r)$,有 $n_r=ar^b$,由 Good-Turing 算法,有 $r^*=(r+1)\dfrac{n_{r+1}}{n_r}=(r+1)\dfrac{A(r+1)^b}{A(r)^b}=r(1+\dfrac{1}{r})^{b+1}$。可使用线性回归来拟合函数。

文[3]指出,首先,使用 Good-Turing 估计,$r=t$,且当 LGT 的值和 Good-Turing 值的差大于 1.65 倍 Good-Turing 标准差时,认为这两者有显著差异,此时替换成 LGT 估计。Good-Turing 标准差可以由下式来估算:$Var(r_T^*)\approx(r+1)^2\dfrac{n_{r+1}}{n_r^2}(1+\dfrac{n_{r+1}}{n_r})$。

因此,由 SGT 算法,有 $r^*=\begin{cases}r(1+\dfrac{1}{r})^{b+1} & r\geqslant t\\ \dfrac{(r+1)n_{r+1}}{n_r} & r<t\end{cases}$。

2.6 Jelinek-Mercer 平滑

Jelinek-Mercer 算法的基本思想是用低阶 N-gram 模型对高阶N-gram 模型进行线性插值。计算公式如下:

$$P_{interp}(w_i|w_{i-N+1}^{i-1})=\lambda_{w_{i-N+1}^{i-1}}P_{ML}(w_i|w_{i-N+1}^{i-1})+(1-\lambda_{w_{i-N+1}^{i-1}})P_{interp}(w_i|w_{i-N+2}^{i-1})$$

0 阶 N-gram 模型采用均匀分布表示。

对于给定的 P_{ML} 和文本 T,$\lambda_{w_{i-N+1}^{i-1}}$ 的计算可以采用期望最大法(EM),经过迭代使得文本 T 的困惑度最小。用于计算 P_{ML} 的文本和 T 必须相互分开。通常计算方式有两种:Held-out 法(Held-out Interpolation)和删除法(Deleted Interpolation)。Held-out 法是把语料库分成两部分,一部分用来计算 P_{ML},一部分用来计算 $\lambda_{w_{i-N+1}^{i-1}}$。删除法是把训练集的不同部分交替用来计算 P_{ML} 和 $\lambda_{w_{i-N+1}^{i-1}}$,然后对计算结果做平均。

由于需要计算的 $\lambda_{w_{i-N+1}^{i-1}}$ 很多,Jelinek-Mercer 建议对 $\lambda_{w_{i-N+1}^{i-1}}$ 参数空间进行划分,属于同一类的所有 $\lambda_{w_{i-N+1}^{i-1}}$ 值相同,这样可以减少空间开销。Baul 提出可以根据 $c(w_{i-N+1}^{i-1})$对 $\lambda_{w_{i-N+1}^{i-1}}$ 进行分类。而且分类时需要保证每类最少有 $C_{\min}$ 个词以便于计算 $\lambda_{w_{i-N+1}^{i-1}}$[2]。

文[6]使用的方法就是 Jelinek-Mercer 算法的简化,所有的词都归为一类,只有一个 $\lambda_{w_{i-N+1}^{i-1}}$ 值,而且没有 0 阶 N-gram 模型。

对于 Bigram,其计算公式为:

$$P_{interp}(w_i|w_{i-1})=\lambda_1P_{ML}(w_i|w_{i-1})+\lambda_2P(w_i)+\lambda_3\frac{1}{V}$$

$$\lambda_1+\lambda_2+\lambda_3=1$$

2.7 One-Count 平滑

One-Count 平滑[2]由 Stanley F.Chen 在 1996 年提出,该方

法分成两个步骤,首先是类似加值平滑的步骤:

$$P_{one}(w_i|w_{i-N+1}^{i-1})=\frac{c(w_{i-N+1}^{i})+\alpha P_{one}(w_i|w_{i-N+2}^{i-1})}{c(w_{i-N+1}^{i-1})+\alpha}$$

按低阶模型的分布增加 n 元对的次数,增加的总和是 α。

其次,根据 Good-turing 平滑算法,增加的次数 α 应该与 n_1 成正比,有

$$\alpha=\gamma[n_1(w_{i-N+1}^{i-1})+\beta]$$

其中 $n_1(w_{i-N+1}^{i-1})=|w_i:c(w_{i-N+1}^{i-1})=1|$,$\gamma$ 和 β 为常数。

Absolute Discounting,Additive Smoothing 是对观测到的值简单地加上后者减去一个值,实现简单。Good-Turing 算法存在 r 大时,有 n_r 为 0 的问题,一般不直接使用,SGT 算法和 Katz 分别用线性拟合和回退到 n-1 元语法的方法解决这个问题,Katz 算法的性能较好,而 SGT 和 One-Count 相对容易实现。Jelinek-Mercer 用低阶 N-gram 模型对高阶 N-gram 模型进行线性插值,参数计算比较复杂。本文对上述的平滑算法在实际分词应用中的性能进行了对比分析。

3　实验设计

3.1　数据

本文所使用的语料来自北京大学计算语言学研究所和富士通研究开发中心提供的 1998 年 1 月人民日报[7],其中 1 日到 25 日的语料用作训练和封闭测试,26 日到 31 日的语料用作开放测试。

在训练时,把特征明显的词,如时间、数字、译名、人名等,用词类来代替,把词类一致的词当成一个词来处理,统计词的 Unigram 值和 Bigram 值。

表 1　数据集

	训练集	封闭测试	开放测试
句子数	151 858	151 858	13 309
词条	43 411	43 411	11 691
词次	831 994	831 994	72 116
Bigram 特征	311 768	/	/
Bigram 总次数	667 155	/	/

注:词数和 Bigram 特征都是去除标点符号,并归为词类之后的结果。

3.2　评价指标

用熵来评价语言模型,并不是模型本身具有熵值,而是用它去近似地计算语言的熵。得到的语言熵越小,说明模型表述语言的性能越好。而困惑度(PP,Perplexity)则是对应用语言模型预测语言成分出现时的难度进行度量。困惑度越大,表明可选择的范围越大,选择的难度也越大[8]。直观上,困惑度可以理解为在给定的语言模型中,某个词后面可能接的词的平均数。有

$$H(W)=-\frac{1}{n}\sum_i \mathrm{lb}P(w_i|w_{i-N+1}^{i-1})$$

$$PP=2^{H(W)}=(\prod_n P(w_i|w_{i-N+1}^{i-1}))^{-\frac{1}{n}}$$

中文分词性能采用三个指标衡量:精确率($Prec$)和召回率($Recall$),以及两者的调和平均 F 值[8]。

$$Prec=\frac{\text{正确的词数}}{\text{正确应切分的词数}}$$

$$Recall=\frac{\text{正确的词数}}{\text{实际切分的词数}}$$

$$F=\frac{2\times Prec\times Recall}{Prec+Recall}$$

采用卡耐基·梅隆大学(http://projectile.is.cs.cmu.edu/research/public/tools/segmentation/eval/index.htm)提供的工具对分词结果进行性能评估。

3.3　算法实现

分别使用加值平滑(Add)、绝对折扣法(Abs)、Katz 平滑、Jelinek-Mercer(JM)平滑、SGT 平滑、One Count 平滑算法来计算熵、困惑度,及在实际分词中的精度、召回率和 F 值。

加值平滑做了两组结果,参数为:Add1 的参数 δ=1,Add2 的参数 $\delta=10^{-5}$。

根据 Katz 的建议,Katz 平滑算法的参数取:k=5。

Jelinek-Mercer 算法使用期望最大法来估计模型参数,有两种实现。第一种实现方法 JM1 参考文[6]的做法,对 $\lambda_{w_{i-1}^{i-1}}$ 的计算过程做简化,所有的词都归为一类,只有一个 $\lambda_{w_{i-N+1}^{i-1}}$。第二种实现方法 JM2 按 $c(w_{i-N+1}^{i-1})$ 分类计算 $\lambda_{w_{i-1}^{i-1}}$。

通过对训练集的数据进行分析和拟合,SGT 平滑算法的参数取:b=-2.374,t=13。

3.4　实验结果

以 1998 年人民日报 1 月 1 日到 25 日的语料为统计样本,分别使用各平滑算法,建立了汉语词的 Bigram 模型,应用所建立的这些模型估算的信息熵及其困惑度如表 2。

表 2　各平滑算法的熵及困惑度

平滑算法	Add2	OneCount	Katz	Abs
熵	6.077	6.156	7.015	7.041
困惑度	67.532	71.313	129.346	131.689
平滑算法	JM2	JM1	Add1	SGT
熵	11.667	11.866	12.946	18.270
困惑度	3 253.552	3 734.502	7 892.135	316 226.8

可以观察到,使用同样的训练库,不同的平滑算法的熵差别较大,最小的是加值算法的 6.077,最大的是 SGT 算法的 18.270。

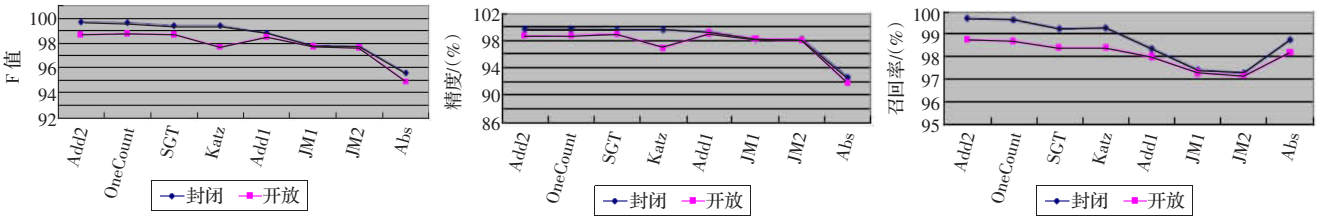
应用上述平滑算法在实际中文分词中的结果如图 2。

同样的,在分词的实际应用中,不同的平滑算法对结果的影响较大,F 值最大和最小之间相差 0.041,精度最大和最小间相差 7.06%。封闭测试中,Add2、OneCount、SGT、Katz 的 F 值均大于 0.99,最优值为 0.996 8;除 Abs 外,在开放测试中,F 值均大于 0.97,最优值为 0.987 2。Katz 算法的开放测试和封闭测试结果差距较大。

在所使用的平滑算法中,OneCount 和 Add2 性能最优;其次是 SGT、Katz 和 Add1,以及 JM 插值法;虽然绝对折扣法的召回率较高,但由于精度低,导致性能相对较差。

另外,与文[8]结论相似,通常情况下,语言模型的困惑度越小,错误率越低。但是也有相反的情况出现,即模型的困惑度虽然很小,但预测的误差却较大,因此,模型的最终质量还要由实际应用的效果来检验。如 SGT 算法,困惑度很高,但是分词性能却不低。

与 Gale、Church 结论相反,在本文的训练集中,加值平滑法的性能相对最优,且参数 δ 越小,困惑度越低。



(a)F 值 (b)精度 (c)召回率

图 2 各平滑算法在分词应用中的性能对比

表 3 各平滑算法的分词性能

方法	封闭			开放		
	精度/(%)	召回率/(%)	F	精度/(%)	召回率/(%)	F
Add2	99.68	99.70	**0.996 8**	98.64	98.74	0.986 8
OneCount	99.67	99.63	0.996 4	98.75	98.70	**0.987 2**
SGT	99.61	99.25	0.994 2	98.97	98.39	0.986 7
Katz	99.55	99.28	0.994 1	96.98	98.39	0.976 8
Add1	99.30	98.37	0.988 3	99.01	97.98	0.984 9
JM1	98.24	97.39	0.978 1	98.17	97.26	0.977 1
JM2	98.20	97.26	0.977 2	98.14	97.14	0.976 3
Abs	92.62	98.75	0.955 8	91.84	98.18	0.949 0

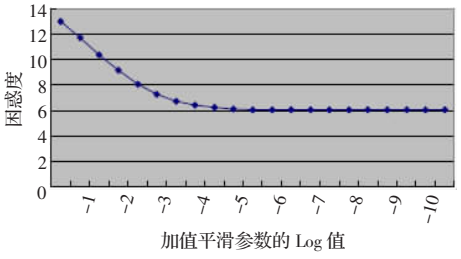


图 3 加值平滑算法参数与困惑度关系

4 结语

N 元语法在中文分词中有着广泛的应用,而数据平滑算法是 *N* 元语法的核心。本文对比多种平滑算法在基于二元语法的中文分词中的应用,以 1998 年 1 月的人民日报语料库为实验对象,发现:通常来说,分词性能与困惑度一致,但也存在反例,困惑度高的算法反而在实际应用中性能较好。虽然 Gale、Church 指出加值平滑性能最差,但是当加值平滑算法所加的值足够小时,性能反而最优。因此选择平滑算法时,应针对实际应用综合比较。

今后的研究,拟使用更大的语料库,引入更复杂的统计语言信息,如三元语法、长距离信息等,进一步验证各平滑算法的性能,提高分词质量。

参考文献:

[1] 黄建中,王肖雷.Katz 平滑算法在中文分词系统中的应用[J].计算机工程,2004,30(12):371-372.

[2] Chen S F,Goodman J.An empirical study of smoothing techniques for language modeling[D].Cambridge:Harvard University,1996.

[3] Gale W A,Sampson G.Good turing frequency estimation without tears[J].Journal of Quantitative Linguistics,1995,2(3).

[4] Ney H,Essen U.Estimating small probabilities by leaving-one-out[C]//Proc Euro Speech,1993:2239-2242.

[5] Church K,Gale W.A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams[J].Computer Speech and Language,1991,5(1):19-54.

[6] 吴春颖,王士同.基于二元语法的 N 最大概率中文粗分模型[J].中文信息学报,2007,27(12):2903-2905.

[7] Yuh J.An adaptive and learning control system for underwater robots[C]//13th World Congress International Federation of Automatic Control,San Francisco,1996,A:145-150.

[8] 张仰森,曹元大,俞士汶.语言模型复杂度度量与汉语熵的估算[J].小型微型计算机系统,2006,27(10):1931-1934.

(上接 15 页)

[2] ZHAO Hui-qun,YAN Ren-xiang,Gao Yuan.Component-based Design and Implementation of DSS' Model[J].Computer Engineering,2000.

[3] GE Zhi-yuan,WANG Yong-xian.A study on adaptive modeling of DSS and its application[J].Journal of Management Sciences in China,2000.

[4] Hui Xiao-bin.An integrative framework for decision support system based on data mining and its approach to modeling[J].Computer Engineering and Applications,2002.

[5] Yang Qiu-fen,Chen Yue-xin.Ontology methodology summary[J].Computer Application and Study,2002,4.

[6] Berners Lee T,Hendler J,L assila O.The semantic Web[J].Scientific American,2001,284(5):34-43.

[7] Gruber T.Towards principles for the design of ontologies used for knowledge sharing[J].International Journal of Human-Computer Studies,1995,43(5/6):907-928.

[8] Stumme G.Using ontologies and formal concept analysis for organizing business knowledge[EB/OL].(2001).http://citeseer.nj.nec.com/stummer01using.html.

[9] Needleman M H.RDF:The resource description framework[J].Standards Update,2001,27(1).

[10] Leon S,Ehud S.The art of prolog[M].Massachusetts,USA:MIT Press,1986.

[11] Gomez-perez A,Benjamins V R.Overview of knowledge sharing and reuse components:Ontologies and problem-solving methods[C]//Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-solving Methods(KRR5),Stockholm,Sweden,1999.