n-gram和数据平滑

常宝宝
北京大学计算语言学研究所
chbb@pku.edu.cn

语言建模(Language Modeling)

- 从统计角度看，自然语言中的一个句子s可以由任何词串构成。不过$P(s)$有大有小。如：

  s_1= 我 刚 吃 过 晚饭

  s_2= 刚 我 过 晚饭 吃

  $P(s_1) > P(s_2)$

- 对于给定的句子s而言，通常$P(s)$是未知的。
- 对于一个服从某个未知概率分布P的语言L，根据给定的语言样本估计P的过程被称作语言建模。

语言建模

- 根据语言样本估计出的概率分布P就称为语言L的语言模型。

$$\sum_{s \in L} P(s) = 1$$

- 语言建模技术首先在语音识别研究中提出，后来陆续用到OCR、手写体识别、机器翻译、信息检索等领域。
- 在语音识别中，如果识别结果有多个，则可以根据语言模型计算每个识别结果的可能性，然后挑选一个可能性较大的识别结果。
- 汉语切分歧义消解？（借助语言模型）

语言建模

- 对于给定的句子$s = w_1 w_2 \ldots w_n$，如何计算$P(s)$?(链式规则 Chain rule)

$$P(s) = p(w_1)p(w_2 \mid w_1)p(w_3 \mid w_1 w_2) \cdots p(w_l \mid w_1 \ldots w_{l-1})$$

$$= \prod_{i=1}^{l} p(w_i \mid w_1 \ldots w_{i-1})$$

- $P(\text{“}JOHN\ READ\ A\ BOOK\text{”})$
  $= p(\text{“}JOHN\text{”}) \times p(\text{“}READ|JOHN\text{”}) \times$
  $p(\text{“}A|JOHN\ READ\text{”}) \times p(\text{“}BOOK|JOHN\ READ\ A\text{”})$
  （参数）

“Shannon Game”

“John read a _____”

- 给定一个句子中前面*n*-1个词，预测下面的词是哪个词。
- 由于语言的规律性，句子中前面出现的词对后面可能出现的词有很强的预示作用。

n-gram

$$p(w_i|w_1 w_2 \ldots w_{i-1})$$

- 为了便于计算，通常考虑的历史不能太长，一般只考虑前面n-1个词构成的历史。即：

$$p(w_i|w_{i-n+1} \ldots w_{i-1})$$

n-gram

n-gram

- *"the large green _____ ."*
  - "*mountain*"? "*tree*"?
- *"Sue swallowed the large green _____ ."*
  - "*pill*"? "*broccoli*"?

- 如果知道 "Sue *swallowed* "会缩小可选择的下一个词的范围。
- 如何选择*n*?

n-gram

- *n* 较大时
  - 提供了更多的语境信息，语境更具区别性
  - 但是，参数个数多、计算代价大、训练语料需要多、参数估计不可靠。
- *n* 较小时
  - 语境信息少，不具区别性
  - 但是，参数个数少、计算代价小、训练语料无需太多、参数估计可靠。

n-gram

- unigram (*n*=1)

  $p(w_i)$　若语言中有20000个词，则需要估计20000个参数

- bigram (*n*=2)

  $p(w_i|w_{i-1})$ 若语言中有20000个词，则需要估计20000^2个参数

- trigram (*n*=3)

  $p(w_i|w_{i-2}w_{i-1})$ 若语言中有20000个词，则需要估计20000^3个参数

- four-gram(*n*=4) 很少使用、不太现实(有时也称为digram或quadrigram)

建立 *n*-gram

- 数据准备:
  - 确定训练语料
  - 对语料进行tokenization 或切分
  - 句子边界，增加两个特殊的词<BOS>和<EOS>
    - I eat . → <BOS> I eat . <EOS>
    - I sleep . → <BOS> I sleep . <EOS>
- 参数估计
  - 利用训练语料，估计模型参数

最大似然估计(MLE)

- 选择一组参数，使得训练样本的概率最大。

  选择能使训练样本取得最大概率值得分布作为总体分布。

- 令 $c(w_1,..,w_n)$ 表示 n-gram $w_1,..,w_n$ 在训练语料中出现的次数。则

$$p_{MLE}(w_n|w_1,..,w_{n-1}) = \frac{c(w_1,..,w_n)}{c(w_1,..,w_{n-1})}$$

最大似然估计

- 假定训练语料如下

  <BOS> JOHN READ MOBY DICK <EOS>

  <BOS> MARY READ A DIFFERENT BOOK <EOS>

  <BOS>SHE READ A BOOK BY CHER<EOS>

- 则有

$$p(JOHN \mid <BOS>) = \frac{c(<BOS> \quad JOHN)}{c(<BOS>)} = \frac{1}{3} \qquad p(READ \mid JOHN) = \frac{c(JOHN \quad READ)}{c(JOHN)} = \frac{1}{1}$$

$$p(A \mid READ) = \frac{c(READ \quad A)}{c(READ)} = \frac{2}{3} \qquad p(BOOK \mid A) = \frac{c(A \quad BOOK)}{c(A)} = \frac{1}{2}$$

$$p(<EOS> \mid BOOK) = \frac{c(BOOK \quad <EOS>)}{c(BOOK)} = \frac{1}{2}$$

最大似然估计

- 计算句子*JOHN READ A BOOK*的概率

  $P(JOHN\ READ\ A\ BOOK)=$
  $p(JOHN|<BOS>)p(READ|JOHN)p(A|READ)p(BOOK|A)$
  $p(<EOS>|BOOK)=0.06$

- 句子的概率表现为若干bigram参数的乘积，若句子太长，计算时，会引起下溢(underflow)，可以采用取对数并相加的方式。

  $Ln(P(JOHN\ READ\ A\ BOOK))=$
  $Ln(p(JOHN|<BOS>))+Ln(p(READ|JOHN))+Ln(p(A|READ))+Ln(p(BOOK|A))+Ln(p(<EOS>|BOOK))$
  $=Ln(1/3)+Ln(1)+Ln(2/3)+Ln(1/2)+Ln(1/2)$
  $=-2.8902$

数据稀疏问题(Data Sparsness)

- 考虑计算句子*CHER READ A BOOK*的概率。

  $c(CHER\ READ)=0$
  → $p(READ|CHER)=0$
  → $p(CHER\ READ\ A\ BOOK)=0$ (有问题)
- MLE给训练样本中未观察到的事件赋以0概率。
- 若某*n*-gram在训练语料中没有出现,则该*n*-gram的概率必定是0。
- 解决的办法是扩大训练语料的规模。但是无论怎样扩大训练语料，都不可能保证所有的词在训练语料中均出现。
- 由于训练样本不足而导致所估计的分布不可靠的问题，称为数据稀疏问题。
- 在NLP领域中，数据稀疏问题永远存在，不太可能有一个足够大的训练语料，因为语言中的大部分词都属于低频词。

Zipf 定律

- Zipf 定律描述了词频以及词在词频表中的位置之间的关系。
- 针对某个语料库，若某个词w的词频是f，并且该词在词频表中的序号为r(即w是所统计的语料中第r常用词)，则

$$f \times r = k \text{ (k是一个常数)}$$

- 若w_i在词频表中排名50，w_j在词频表中排名150，则w_i的出现频率大约是w_j的频率的3倍。
- 例：马克吐温的小说 Tom Sawyer
  - 共 71,370 词 (word tokens)
  - 出现了 8,018 个不同的词 (word types)

Zipf 定律

Word	Freq.	Use
the	3332	determiner (article)
and	2972	conjunction
a	1775	determiner
to	1725	preposition, verbal infinitive marker
of	1440	preposition
was	1161	auxiliary verb
it	1027	(personal/expletive) pronoun
in	906	preposition
that	877	complementizer, demonstrative
he	877	(personal) pronoun
I	783	(personal) pronoun
his	772	(possessive) pronoun
you	686	(personal) pronoun
Tom	679	proper noun
with	642	preposition

Tom Sawyer

Zipf 定律

Word Frequency	Frequency of Frequency
1	3993
2	1292
3	664
4	410
5	243
6	199
7	172
8	131
9	82
10	91
11–50	540
51–100	99
> 100	102

- 大部分词是低频词，3993 (50%)词(word types)仅仅出现了一次
- 常用词极为常用，前100个高频词占了整个文本的51% (word tokens)

Zipf 定律

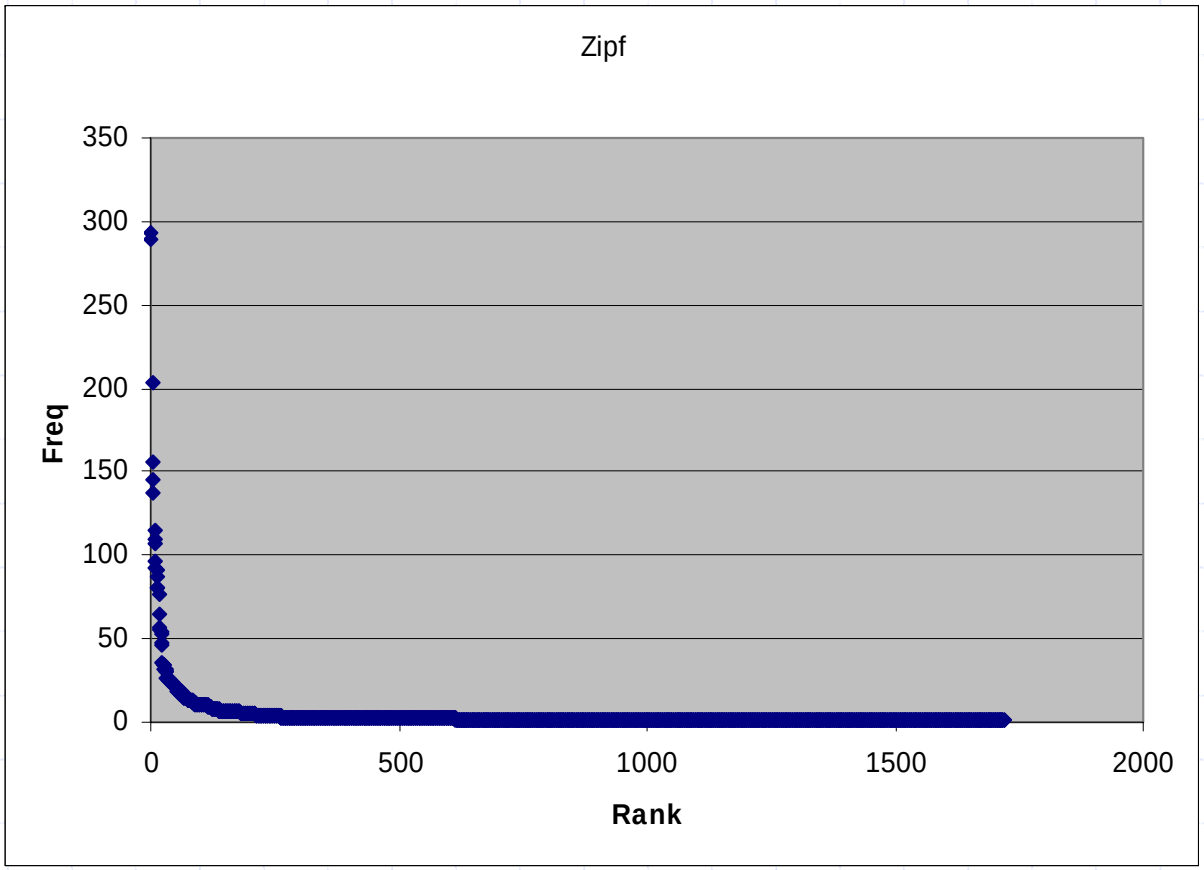
Word	Freq. (f)	Rank (r)	$f \cdot r$
the	3332	1	3332
and	2972	2	5944
a	1775	3	5235
he	877	10	8770
but	410	20	8400
be	294	30	8820
there	222	40	8880
one	172	50	8600
about	158	60	9480
more	138	70	9660
never	124	80	9920
Oh	116	90	10440
two	104	100	10400

Word	Freq. (f)	Rank (r)	$f \cdot r$
turned	51	200	10200
you'll	30	300	9000
name	21	400	8400
comes	16	500	8000
group	13	600	7800
lead	11	700	7700
friends	10	800	8000
begin	9	900	8100
family	8	1000	8000
brushed	4	2000	8000
sins	2	3000	6000
Could	2	4000	8000
Applausive	1	8000	8000

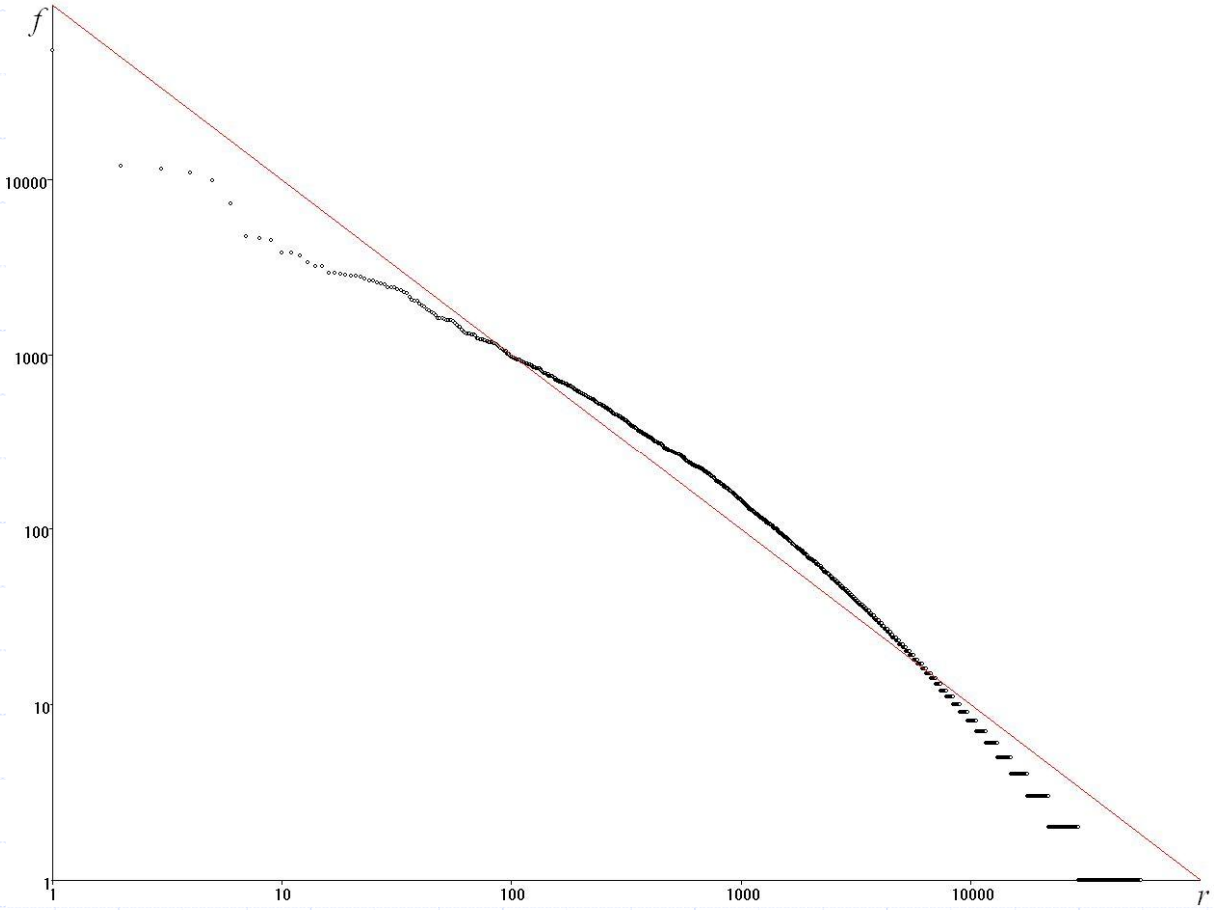
- $k \approx$ 8000-9000
- 有例外
  - 前3个最常用的词
  - r = 100 时

Zipf 定律

Tom Sawyer 第1-3章



Zipf定律



1998年1月份人民日报语料统计情况，Zipf定律对汉语而言也大致成立。

Zipf定律

- Zipf定律告诉我们
  - 语言中只有很少的常用词
  - 语言中大部分词都是低频词(不常用的词)
- Zipf的解释是 Principle of Least effort
  (讲话的人和听话的人都想省力的平衡)
  - 说话人只想使用少量的常用词进行交流
  - 听话人只想使用没有歧义的词(量大低频)进行交流
- Zipf 定律告诉我们
  - 对于语言中的大多数词，它们在语料中的出现是稀疏的
  - 只有少量词语料库可以提供它们规律的可靠样本。

数据稀疏问题

- Balh 等人的工作
  - 用150 万词的训练语料训练trigram模型
  - 测试语料（同样来源）中23%的trigram没有在训练语料中出现过。
- 对语言而言，由于数据稀疏的存在，MLE 不是一种很好的参数估计办法。
- 解决办法: 平滑技术
  - 把在训练样本中出现过的事件的概率适当减小
  - 把减小得到的概率密度分配给训练语料中没有出现过的事件
  - 这个过程有时也称为discounting(减值)

平滑技术

- 目前已经提出了很多数据平滑技术，如：
  - Add-one 平滑
  - Add-delta 平滑
  - Witten-Bell平滑
  - Good-Turing平滑
  - Church-Gale平滑
  - Jelinek-Mercer平滑
  - Katz平滑
  - ……

Add-one 平滑 (Laplace's law)

- 规定任何一个*n*-gram在训练语料至少出现一次（即规定没有出现过的*n*-gram在训练语料中出现了一次）
- 则: *new_count*(*n*-gram) = *old_count*(*n*-gram) + 1

- 没有出现过的*n*-gram的概率不再是0

Add-one 平滑

2nd word

1st word

	I	want	to	eat	Chinese	food	lunch	…	Total (N)
I	8	1087	0	13	0	0	0		3437
want	3	0	786	0	6	8	6		1215
to	3	0	10	860	3	0	12		3256
eat	0	0	2	0	19	2	52		938
Chinese	2	0	0	0	0	120	1		213
food	19	0	17	0	0	0	0		1506
lunch	4	0	0	0	0	1	0		459
…									

未平滑的 bigram 频次

	I	want	to	eat	Chinese	food	lunch	…	Total
I	.0023 (8/3437)	.32	0	.0038 (13/3437)	0	0	0		1
want	.0025	0	.65	0	.0049	.0066	.0049		1
to	.00092	0	.0031	.26	.00092	0	.0037		1
eat	0	0	.0021	0	.020	.0021	.055		1
Chinese	.0094	0	0	0	0	.56	.0047		1
food	.013	0	.011	0	0	0	0		1
lunch	.0087	0	0	0	0	.0022	0		1
…									

未平滑的 bigram 概率

Add-one 平滑

平滑后的 bigram 频次

	I	*want*	*to*	*eat*	*Chinese*	*food*	*lunch*	…	***Total (N+V)***
I	~~8~~ 9	~~1087~~ 1088	1	14	1	1	1		~~3437~~ 5053
want	~~3~~ 4	1	787	1	7	9	7		2831
to	4	1	11	861	4	1	13		4872
eat	1	1	23	1	20	3	53		2554
Chinese	3	1	1	1	1	121	2		1829
food	20	1	18	1	1	1	1		3122
lunch	5	1	1	1	1	2	1		2075

	I	*want*	*to*	*eat*	*Chinese*	*food*	*lunch*	…	***Total***
I	.0018 (9/5053)	.22	.0002	.0028 (14/5053)	.0002	.0002	.0002		1
want	.0014	.00035	.28	.00035	.0025	.0032	.0025		1
to	.00082	.00021	.0023	.18	.00082	.00021	.0027		1
eat	.00039	.00039	.0012	.00039	.0078	.0012	.021		1
Chinese	.0016	.00055	.00055	.00055	.00055	.066	.0011		1
food	.0064	.00032	.0058	.00032	.00032	.00032	.00032		1
lunch	.0024	.00048	.00048	.00048	.00048	.0022	.00048		1

平滑后的 bigram 概率 $p(w_1|w_2)$

Add-one 平滑

$$P_{Add1}(w_1\ w_2\ \ldots w_n) = \frac{C\,(w_1\ w_2\ \ldots w_n) + 1}{N + V}$$

N: 训练语料中所有的*n*-gram的数量(token)

V: 所有的可能的不同的*n*-gram的数量(type)

Add-one 平滑

- 训练语料中未出现的*n*-gram的概率不再为0，而是一个大于0的较小的概率值。
- 但由于训练语料中未出现*n*-gram数量太多，平滑后，所有未出现的*n*-gram占据了整个概率分布中的一个很大的比例。
- 因此，在NLP中，Add-one给训练语料中没有出现过的*n*-gram分配了太多的概率空间。
- 认为所有未出现的*n*-gram概率相等，这是否合理？
- 出现在训练语料中的那些*n*-gram，都增加同样的频度值，这是否公平？(低频、高频)

- AP 语料数据 (Church and Gale, 1991)
  - 语料共含有 22,000,000 个bigrams (token)
  - 语料中共出现了 273,266 个词(type) (因此共有 74,674,306,760个可能的 bigrams (type))
  - 74,671,100,000 bigrams 在语料中没有出现
  - 根据Add-one平滑，每个未出现过的 bigram 平均出现 0.000295次

Freq. from training data

Freq. from held-out data

Add-one smoothed freq.

too high

too low

f_{MLE}	$f_{empirical}$	$f_{add\text{-}one}$
0	0.000027	**0.000295**
1	0.448	0.000274
2	1.25	0.000411
3	2.24	0.000548
4	3.23	0.000685
5	4.21	0.000822

- 所有未出现过的bigram的概率分布 =

  (74,671,100,000 x 0.000295) / 22,000,000 ~**99.96** !!!!

Add-delta 平滑 (Lidstone's law)

- 不是加 1,而是加一个小于1的正数 λ

$$P_{AddD}(w_1\ w_2\ \ldots w_n) = \frac{C(w_1\ w_2\ \ldots w_n) + \lambda}{N + \lambda V}$$

- 通常 λ = 0.5，此时又称为Jeffreys-Perks Law或ELE

$$P_{ELE}(w_1\ w_2\ \ldots w_n) = \frac{C(w_1\ w_2\ \ldots w_n) + 0.5}{N + 0.5\ V}$$

- 效果比Add-one好，但是仍然不理想

留存估计(Held-out Estimation)

- 留存数据(Held-out data)
  - 把训练语料分作两个部分:
    - **训练语料(training set):** 用于初始的频率估计。
    - **留存语料(held out data):** 用于改善最初的频率估计

- 对于每一个 n-gram $w_1...w_n$ 计算:
  - $C_{tr}(w_1...w_n)$ $w_1...w_n$ 在训练语料中出现的频率。
  - $C_{ho}(w_1...w_n)$ $w_1...w_n$ 在留存数据中出现的频率。

留存估计

- 令:
  - r = 某个 n-gram 在训练语料中出现的频率
  - N_r = 在训练语料中出现了r次的不同的 n-gram 的个数。
  - T_r = 所有在训练语料中出现了r次的 n-gram 在留存语料中出现的频率之和。

$$T_r = \sum_{\{w_1 \cdots w_n \mid C_{tr}(w_1 \cdots w_n) = r\}} C_{ho}(w_1 \cdots w_n)$$

  - T = 留存语料中所有的n-gram数(token)
- 则:

$$P_{ho}(w_1 \cdots w_n) = \frac{T_r}{T} \times \frac{1}{N_r}, \quad r = C_{tr}(w_1 \cdots w_n)$$

留存估计

$$P_{ho}(w_1 \cdots w_n) = \frac{T_r}{T} \times \frac{1}{N_r}, \quad r = C_{tr}(w_1 \cdots w_n)$$

所有在训练语料中出现r次的n-gram 在留存语料中的概率。

由于在训练语料中共有 N_r 不同的出现了r次的 n-gram，让它们等概率分布

- 例如：假定
  - r=5 并且有10个不同的 n-gram (types) 在训练语料中出现了5次，即: $N_5 = 10$
  - 这10个不同的n-gram在留存语料中共出现了20次，即:$T_5 = 20$
  - 留存语料中共包含2000 个 n-gram (token)

$$P_{ho}(\text{an } n\text{-gram with } r = 5) = \frac{20}{2000} x \frac{1}{10} = 0.001$$

删除估计 (Deleted Estimation)

- 如果有很多训练语料的话，可以使用留存估计
- 如果训练语料不多的话，可以…
  - 把训练语料分成两个部分 part 0 和 part 1
  - 把 part 0 作为训练语料，把 part 1 作为留存语料进行建模
  - 再用 part 1 作为训练语料，把 part 0 作为留存语料进行建模
  - 对所得到的两个模型加权平均，求得最后的模型

$$p_{del}(w_1 \ldots w_n) = \frac{T_r^{01} + T_r^{10}}{N(N_r^0 + N_r^1)}, \qquad c(w_1 \ldots w_n) = r$$

  - 这样的方法通常称作删除估计或双向交叉验证 (Deleted Estimation or two-way cross validation)

Good-Turing 平滑

- 主要思想:
  - 利用高频率*n*-gram的频率调整低频的*n*-gram的频率。

$$r^* = (r+1)\frac{N_{r+1}}{N_r}$$

出现了 r+1 次的n-gram个数

出现了 r 次的n-gram个数

Turing estimate

  - 分配给所有未出现的*n*-gram的概率空间。

$$N_0 \frac{r_0^*}{N} = \frac{N_1}{N}$$

Good-Turing 平滑

- 在估计频度为r的n-gram的概率p_r时，如果数据集中没有频度为r+1的n-gram怎么办？此时，N_{r+1}=0导致p_r=0
- 解决办法：对N_r进行“平滑”，设$S(.)$是平滑函数，$S(r)$是N_r的平滑值。(Good-Turing estimate)

$$r^* = (r+1)\frac{S(r+1)}{S(r)}$$

- 如何选择$S(.)$?

r	N_r
1	120
2	40
3	24
4	13
5	15
6	5
7	11
8	2
9	2
10	1
12	3
14	2
15	1
16	1
17	3
19	1
20	3
21	2
23	3
24	3

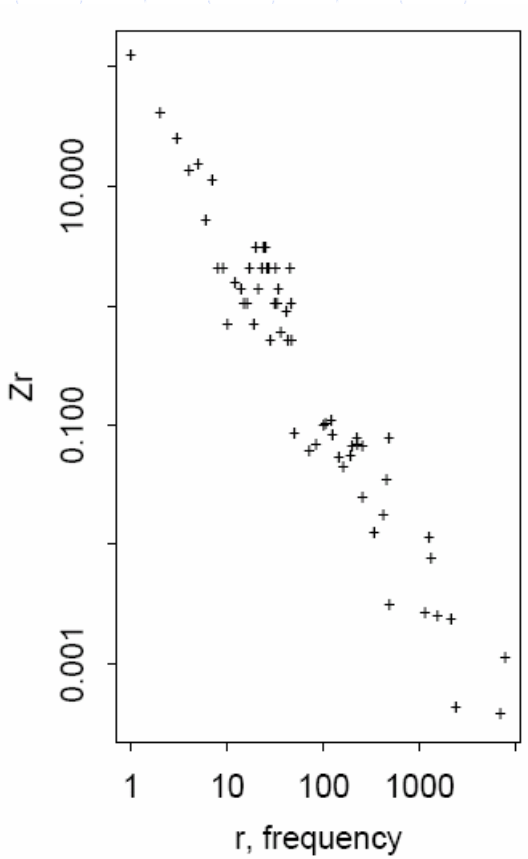
简单Good-Turing平滑[1]

- $log(r)$和$log(N_r)$的关系
- r较大时，很多N_r是0，N_r不可靠。采用下面简单的办法对N_r进行修正。

$$Z_r = \frac{N_r}{0.5(t-q)}$$

这里$t > r > q$, 并且$N_t \neq 0$, $N_r \neq 0$, $N_q \neq 0$

若 $t > x > r$ $r > x > q$，则$N_x = 0$



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

简单Good-Turing 平滑

- 关于r^*和r的关系 (r>1)
  - $r^* < r$
    - p_r=r*/N，要把一部分概率空间分配给未出现的n-gram
  - 随着r增大，r^*越接近r，即r^*/r趋近于1
    - r越大，p_{MLE}越可靠，分配出去的概率应该越小。
- 线形平滑

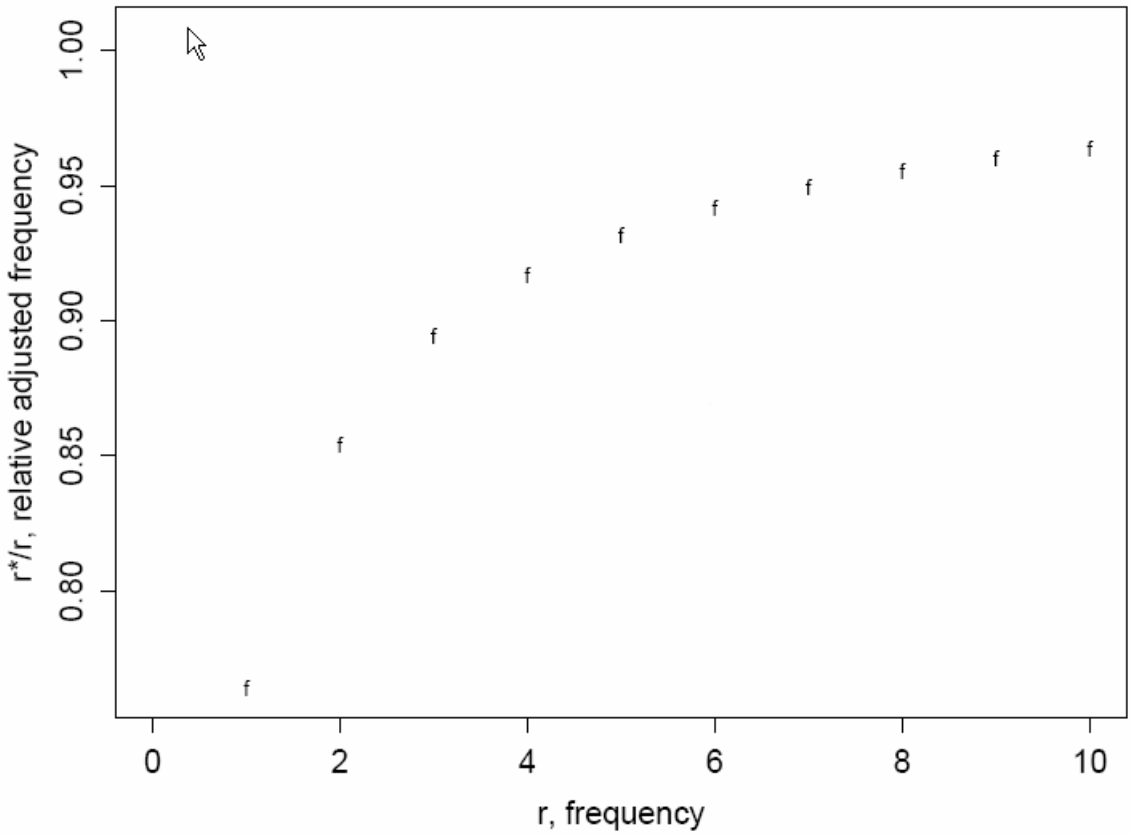
$$log(Nr) = a + b\,log(r)$$

- 这意味着

$$N_r = Ar^b \quad ,(a = log(A))$$

简单Good-Turing平滑

- 给定一组$log(Z_r)$和$log(r)$，通过线形回归的办法确定参数a和b
- 只有$b < -$



简单Good-Turing平滑

- Turing估计比较可靠，因此应尽可能使用Turing估计，当Turing估计和Good-Turing估计差别不大时，转向使用Good-Turing估计，直至结束。
- 若Turing估计和Good-Turing的差小于1.65倍的Turing估计标准差时，可认为二者差别不大。
- Turing估计的方差可近似定义为：

$$Var(V_T^*) \approx (r+1)^2 \frac{N_{r+1}}{N_r^2}(1+\frac{N_{r+1}}{N_r})$$

- 归一

$$p_r = (1-\frac{N_1}{N})\frac{p_r^{unnorm}}{\sum_{r\geq 1} N_r \cdot p_r^{unnorm}} \qquad r \geq 1$$

组合估计

- 到目前为止，所有未出现的 *n*-gram 概率估计值均相等 (均匀分布)
  - 例如，下面的bigram若均未出现过
    - journal of $P_{unsmoothed}(of \mid journal) = 0$
    - journal from $P_{unsmoothed}(from \mid journal) = 0$
    - journal never $P_{unsmoothed}(never \mid journal) = 0$
  - 前面的所有平滑技术均给它们一个相等的概率值
- 但实际上, “journal of ” 概率应该更大.
  - “of” 频率高于“from” & “never”
  - unigram 概率应为 $P(of) > P(from) > P(never)$

组合估计

- 依据
  - unigram model 数据稀疏问题比 bigram model 小
  - bigram model 数据稀疏问题比 trigram model 小
  - …
- 高阶*n*-gram的概率估计可以求助于低阶*n*-gram的概率估计值
- 因此可以把不同阶的*n*-gram 模型组合起来产生一个更好的模型。

简单线形插值(Simple Linear Interpolation)

- 把不同阶别的*n*-gram模型线形加权组合

$$P_{li}(w_n|w_{n-2},w_{n-1}) = \lambda_1 P(w_n) + \lambda_2 P(w_n|w_{n-1}) + \lambda_3 P(w_n|w_{n-2},w_{n-1})$$

其中 $0 \le \lambda_i \le 1$, $\Sigma_i \lambda_i = 1$

- 又称:
  - 有限混合模型(finite mixture model)
  - 删除插值模型(deleted interpolation)
- λ_i可以根据试验凭经验设定。也可以通过应用某些算法确定，例如EM算法。

简单线形插值(Simple Linear Interpolation)

- 在简单线形插值法中，权值 λ_i 是常量
- 不管高阶模型的估计是否可靠，低阶模型均以同样的权重被加入模型，这并不合理。
- 解决办法是让 λ_i 成为历史的函数 h

Jelinek-Mercer平滑[1]

- Jelinek and Mercer提出如下的插值模型。

$$p_{JM}(w_i \mid w_{i-n+1}^{i-1}) = \lambda_{w_{i-n+1}^{i-1}} p_{ML}(w_i \mid w_{i-n+1}^{i-1}) + (1-\lambda_{w_{i-n+1}^{i-1}}) p_{JM}(w_i \mid w_{i-n+2}^{i-1})$$

- 递规的终止：
  - 终止于平滑后的unigram模型 $p_{smoothed}(w_i)$
  - 对unigram继续采用MLE估计，继续递归终止于所谓的0元模型。0元模型定义为均匀分布。

$$p_{JM}(w_i) = 1/|V|$$

- 如何得到 $\lambda_{w_{i-n+1}^{i-1}}$?使用参数训练算法。 (Baum-Welch)

Jelinek-Mercer平滑

- 参数 $\lambda_{w_{i-n+1}^{i-1}}$ 太多，需要海量数据来训练。必须减少参数个数。
- Jelinek和Mercer建议依据某种策略把参数 $\lambda_{w_{i-n+1}^{i-1}}$ 划分成若干个类(bucket)，每个类内的参数取同样的值。
  - 依据 $c(w_{i-n+1}^{i-1})$ 来分类：把 $c(w_{i-n+1}^{i-1})$ 分成若干个区间，每个区间内对应的参数 $\lambda_{w_{i-n+1}^{i-1}}$ 取同样的值。
  - 依据 下面的原则分类

$$\frac{c(w_{i-n+1}^{i-1})}{|w_i : c(w_{i-n+1}^{i}) > 0|}$$

[1] Frederick Jelinek and Robert L. Mercer. 1980. Interpolated estimation of Markov source parameters from sparse data. In Proceedings of the Workshop on Pattern Recognition in Practice, Amsterdam, The Netherlands: North-Holland, May.

回退模型(back-off model)

- 基本思想
  - 在高阶模型可靠时，尽可能的使用高阶模型
  - 必要时，使用低阶模型
- 回退模型的一般形式

$$p_{smooth}(w_i \mid w_{i-n+1}^{i-1}) = \begin{cases} \hat{p}(w_i \mid w_{i-n+1}^{i-1}) & \text{if } c(w_{i-n+1}^{i}) > 0 \\ \alpha(w_{i-n+1}^{i-1}) \cdot p_{smooth}(w_i \mid w_{i-n+2}^{i-1}) & \text{if } c(w_{i-n+1}^{i}) = 0 \end{cases}$$

- 参数 $\alpha(w_{i-n+1}^{i-1})$ 是归一因子，以保证

$$\sum_{w_i} p_{smooth}(w_i \mid w_{i-n+1}^{i-1}) = 1$$

Katz平滑[1]

- 以bigram为例，令 $r = c(w_{i-1}w_i)$
- 若r>0，则 $p_{katz}(w_i \mid w_{i-1}) = d_r \cdot p_{ML}(w_i \mid w_{i-1})$

  $d_r (\in (0,1])$ 称为折扣率(discount ratio)，
  不同的r具有不同的折扣率。
- 给定 w_{i-1}，从r>0的bigrams中扣除的概率为：

$$S(w_{i-1}) = 1 - \sum_{w_i \in M(w_{i-1})} p_{katz}(w_i | w_{i-1}),$$

其中$M(w_{i-1}) = \{w_i | c(w_{i-1}w_i) > 0\}$

[1] Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. IEEE Transactions on Acoustics, Speech and Signal Processing, ASSP-35(3):400-401, March.

Katz平滑

- 对于给定的 w_{i-1} ，令

  $Q(w_{i-1}) = \{w_i \mid c(w_{i-1}w_i) = 0\}$

- 如何把 $S(w_{i-1})$ 分配给集合$Q(w_{i-1})$ 中的那些元素？
  - 回退到unigram模型
- 对于$w_i \in Q(w_{i-1})$，若$p_{ML}(w_i)$比较大，则应该分配更多的概率给它。所以，若$r = 0$，则：

$$p_{katz}(w_i \mid w_{i-1}) = \frac{p_{ML}(w_i)}{\sum_{w_j \in Q} p_{ML}(w_j)} \times S(w_{i-1})$$

Katz平滑

- 对于bigram模型，Katz平滑为

$$p_{katz}(w_i \mid w_{i-1}) = \begin{cases} d_r \cdot p_{ML}(w_i \mid w_{i-1}) & \text{if } r > 0 \\ \alpha(w_{i-1}) \times p_{ML}(w_i) & \text{if } r = 0 \end{cases}$$

$$\text{其中} \alpha(w_{i-1}) = \frac{1 - \sum_{w_j \in M} p_{katz}(w_j \mid w_{i-1})}{\sum_{w_j \in Q} p_{ML}(w_j)}$$

Katz平滑

- 如何计算d_r?
  - 若$r > k$, 不折扣，即$d_r = 1$ (Katz提出$k = 5$)
  - 若$0 < r \leqslant k$, 按照和Good-Turing估计同样的方式折扣，即按照r^*/r进行折扣。严格说，要求d_r满足

$$1 - d_r = \mu(1 - \frac{r^*}{r})$$

  - 根据Good-Turing估计，未出现的n元组估计出现频次是n_1

$$\sum_{r=1}^{k} n_r(1 - d_r)r = n_1$$

Katz平滑

- 具体而言，若$0 < r \leqslant k$，有 $d_r = \dfrac{\dfrac{r^*}{r} - \dfrac{(k+1)n_{k+1}}{n_1}}{1 - \dfrac{(k+1)n_{k+1}}{n_1}}$
- bigram的Katz平滑模型最终可描述为:

$$p_{Katz}(w_i \mid w_{i-1}) = \begin{cases} c(w_{i-1}w_i)/c(w_{i-1}) & r > k \\ d_r c(w_{i-1}w_i)/c(w_{i-1}) & k \geq r > 0 \\ \alpha(w_{i-1})\, p_{Katz}(w_i) & r = 0 \end{cases}$$

- n-gram模型的Katz平滑可依此类推。

回退模型和插值模型

- 在回退模型和线形插值模型中，当高阶*n*-gram未出现时，使用低阶*n*-gram估算高阶*n*-gram的概率分布。
- 在回退模型中，高阶*n*-gram一旦出现，就不再使用低阶*n*-gram进行估计。
- 在线形插值模型中，无论高阶*n*-gram是否出现，低阶*n*-gram都会被用来估计高阶*n*-gram的概率分布。