语言不是由随机抽取的一些单词组成的序列，词与词之间是有联系的。这种规律用概率模型刻画。我们用语言模型来称呼单词序列的统计模型。

例1 一个人所说的话中每100个句子有一句是"OK"，则可认为 $P(OK)\approx 1\%$

例2、___和阿里、腾讯并称为中国互联网BAT三巨头。

P（百度和阿里、腾讯并称为___）≈ 1

例2揭示了一件事：上下文信息越强，词语可预测性越高。

设词语序列 $S = w_1 w_2 \cdots w_l$，根据链式法则

$$P(S) = P(w_1 w_2 \cdots w_l) = P(w_1 | <BOS>) P(w_2 | w_1) P(w_3 | w_1 w_2) \cdots P(w_l | w_1 \cdots w_{l-1})$$

即 $$P(S) = \prod_{i=1}^{l} P(w_i | w_1 \cdots w_{i-1})$$

每个条件概率都是模型的一个参数，训练模型本质上是

①确定模型参数集合 ②确定各参数的值

在n-gram语言模型中，我们用最大似然估计来算条件概率

$$P(w_i | w_1 w_2 \cdots w_{i-1}) = \frac{C(w_1 w_2 \cdots w_{i-1} w_i)}{\sum_{w_j} C(w_1 w_2 \cdots w_{i-1} w_j)}$$

C是count的缩写，分子表示$(w_1 \cdots w_i)$在语料库中出现次数

分母表示所有可能的下一个词 w_i 与前面序列组合出现总次数

如 $w_1 \cdots w_{i-1}$ 表示"我喜欢"，分子表示C(我喜欢吃)（w_i固定为"吃"）

而分母表示C(我喜欢吃)+C(我喜欢看)+C(我喜欢你)+…

即"我喜欢+任意词"的次数总和

存在问题：参数空间过大，$P(w_i | w_1 \cdots w_{i-1})$ 在训练语料中趋近于0

数据平重稀疏

n元语法

马尔可夫假设：一个词的出现仅依赖它前一个或几个词，受离它近的词影响大

$$P(\text{什么}|\text{你今天在干}) \approx P(\text{什么}|\text{干})$$

二元语法模型：一个词的出现仅依赖其前一个词

$$P(w_1 \cdots w_L) = \prod_{i=1}^{L} P(w_i|w_{i-1})$$

$$P(\text{我爱你}) = P(\text{我}|<BOS>) \times P(\text{爱}|\text{我}) \times P(\text{你}|\text{爱}) \times P(<EOS>|\text{你})$$

注：<BOS> Beginning of sequence 序列起始符

<EOS> End of sequence 序列终止符

<PAD> padding 填充

例：

二元语法模型参数$P(w_i/w_{i-1})$的最大似然估计

例

序号	w_{i-1}	w_i	$C(w_{i-1}, w_i)$	$P(w_i \mid w_{i-1})$
1	父亲	写	50	50/100= 0.5
2	父亲	看	40	40/100= 0.4
3	父亲	吃	10	10/100= 0.1
总计			100	1

$$p(w_i \mid w_{i-1}) = \frac{p(w_{i-1}w_i)}{p(w_{i-1})} = \frac{C(w_{i-1}w_i)/N}{C(w_{i-1})/N} = \frac{C(w_{i-1}w_i)}{C(w_{i-1})}$$

$C(w_aw_b) \rightarrow w_aw_b$在给定文本中出现的次数

$N \rightarrow$ 训练语料中词的个数

例、训练语料：中国万岁 中国中国 万岁中国 万岁万岁

一元语法：$P(\text{中国}) = \frac{4}{8}$ $P(\text{万岁}) = \frac{4}{8}$

二元语法：$P(\text{中国}|<BOS>) = \frac{2}{4}$ $P(\text{万岁}|<BOS>) = \frac{2}{4}$

$P(\text{中国}|\text{中国}) = \frac{1}{4}$ $P(\text{万岁}|\text{中国}) = \frac{1}{4}$

$P(\text{中国}|\text{万岁}) = \frac{1}{4}$ $P(\text{万岁}|\text{万岁}) = \frac{1}{4}$

$P(<EOS>|中国)=\frac{1}{4}$　　$P(<EOS>|万岁)=\frac{1}{4}$

$\therefore P(中国万岁)=P(中国|<BOS>)\times P(万岁|中国)\times$
$P(<EOS>|万岁)=\frac{1}{4}\times\frac{1}{4}\times\frac{1}{4}=\frac{1}{64}$

推广：n元语法模型：一个词的出现依赖其前n-1个词

$P(w_1\cdots w_l)=\prod_{i=1}^{l}P(w_i|w_{i-(n-1)}\cdots w_{i-1})$

n的选择{ 大　约束多，辨别强
　　小　训练集数据多，更可靠

理论上n越大越好，实际上二元和三元运用得多

Laplace平滑

最大似然估计的问题：训练语料不可能囊括所有合理的词语搭配，一旦出现未在训练语料中囊括的词语搭配，那么其P就为0，模型完全无法预测

Laplace平滑：$P_{MLE}(w_i|w_{i-1})=\frac{C(w_{i-1}w_i)}{\sum_{w_i}C(w_{i-1}w_i)}=\frac{C(w_{i-1}w_i)}{C(w_{i-1})}$　（简化）

↓

$P_{Lap}(w_i|w_{i-1})=\frac{C(w_{i-1}w_i)+1}{\sum_{w_i}[C(w_{i-1}w_i)+1]}=\frac{C(w_{i-1}w_i)+1}{C(w_{i-1})+|V|}$

$|V|$表示词汇表中单词个数

例：

- **p(Father read a book)**

Father read Holy Bible
Mother read a text book
He read a book by Grandpa

$V = \{Father, read, \cdots, Grandpa, <BOS>, <EOS>\}$
$|V| = 13$

$$p(Father\ read\ a\ book)$$
$$= p(Father \mid <BOS>) \times p(read \mid Father) \times p(a \mid read) \times p(book \mid a) \times p(<EOS> \mid book)$$

MLE:

$$= \frac{1}{3} \times 1 \times \frac{2}{3} \times \frac{1}{2} \times \frac{1}{2} = 0.06$$

LAP:

$$= \frac{2}{16} \times \frac{2}{14} \times \frac{3}{16} \times \frac{2}{15} \times \frac{2}{15} \approx 0.00006$$

- **p(Grandpa read a book)**

Father read Holy Bible
Mother read a text book
He read a book by Grandpa

$V = \{Father, read, \cdots, Grandpa, <BOS>, <EOS>\}$
$|V| = 13$

$$p(Grandpa\ read\ a\ book)$$
$$= p(Grandpa \mid <BOS>) \times p(read \mid Grandpa) \times p(a \mid read) \times p(book \mid a) \times p(<EOS> \mid book)$$
$$= \frac{1}{16} \times \frac{1}{14} \times \frac{3}{16} \times \frac{2}{15} \times \frac{2}{15} \approx 0.000015$$

Laplace平滑一般形式：

$$P_{lap}(w_i \mid w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i}) + 1}{\sum_{w_i} C(w_{i-n+1}^{i}) + |V|}$$

$$= \frac{C(w_{i-n+1}^{i}) + 1}{C(w_{i-n+1}^{i-1}) + |V|}$$

其中 $w_i^j = w_i \cdots w_j$

在前一个例子中，我们发现laplace平滑后句子出现概率非常小，这是由于直接加1会将概率稀释得很厉害，所以我们有：

Lidstone法则：

$$P_{Lid}(w_i \mid w_{i-(n-1)}^{i-1}) = \frac{C(w_{i-(n-1)}^{i-1}) + \delta}{C(w_{i-(n-1)}^{i-1}) + \delta|V|}$$

其中 δ 是一个很小的正数，如0.1，0.01等

比Laplace更温和、实用

一元先验平滑：给所有同一个"保底概率"，无论其是否在训练集中出现过。

$$P(w_i) = \frac{C(w_{i-1}w_i) + mP(w_i)}{C(w_{i-1}) + m}$$

Good-Turning 估计

对于任何一个出现 r 次的词(这记作 n 元语法)都假设其发生 r^* 次

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

n_r 表示训练语料中出现 r 次词的数量

r	0	1	2	3	4	...
n_r	n_0	n_1	n_2	n_3	n_4	...
r^*	$\frac{n_1}{n_0}$	$2\frac{n_2}{n_1}$	$3\frac{n_3}{n_2}$	$4\frac{n_4}{n_3}$	$5\frac{n_5}{n_4}$	...

这样便平滑了低频次的频率，避免了"出现0次的词概率为0"

总词数验证：$\sum r n_r = n_1 + 2n_2 + 3n_3 + \cdots = N$

修正后：$\sum r^* n_r = n_1 + 2n_2 + 3n_3 + \cdots = N$

总词数无变化

$$P_{GT}(w) = \frac{r^*}{N} = (r+1)\frac{n_{r+1}}{n_r N}$$

缺陷：在实际统计中，会发现一个现象：低频次出现次数反高于高频词，且高频词数量波动大，这会导致公式低估低频词而严重失真

改进：Katz平滑

只对 $r \le k$ 的词修正 $r > k$ 的词直接用原次数 (k 一般取 5)

即
$$\begin{cases} r = 0 & r^* = \frac{n_1}{n_0} \\ 1 \le r \le k & r^* = \dfrac{(r+1)\frac{n_{r+1}}{n_r} - r\frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \\ r > k & r^* = r \end{cases}$$

$$P_{katz}(w) = \begin{cases} \frac{r^*}{N} & r \le k \\ \frac{r}{N} & r > k \end{cases}$$

绝对折扣

$$P_{abs}(w) = \begin{cases} \frac{r-\delta}{N} & r > 0 \\ \frac{(\sum_{r=1}^{\infty} n_r)\delta}{n_0 N} & r = 0 \end{cases}$$

$r>0$：这个词出现次数减去一个固定值 δ（通常很小）

再除以总词数 N

$r=0$：未登录词的概率 = 所有词扣掉的总折扣除以未登录词的数量与总词数之积，即将扣掉的概率平均分给未登录词

r（次数）	n_r（词数）	$P_{MLE}(w)$	$P_{abs}(w)$ （δ=0.01）
0	16	0	(40-16)×0.01÷16÷50=0.0003
1	10	0.02	0.99/50=0.0198
2	6	0.04	1.99/50=0.0398
3	5	0.06	2.99/50=0.0598
4	2	0.08	3.99/50=0.0798
5	1	0.10	4.99/50=0.0998
总计	V=40 N=50		

线性折扣

$$P_{LD}(w) = \begin{cases} \frac{(1-\alpha)r}{N} & r > 0 \\ \frac{\alpha}{n_0} & r = 0 \end{cases}$$

α 是折扣比例（通常很小）

$r>0$：每个词按比例 $(1-\alpha)$ 打折

r=0：将α平均分给未登录词

组合估计

假定我们要在一堆语料上构建三元语法模型

其中 C(he send the) =0 C(he send thou)=0

无论是 Lap、GT、abs都有：

P(the | he send) = P(thou | he send)

但是直觉上P(the | he send)更大一些

这是因为在二元语法模型中

P(the | send) > P(thou | send)

低阶模型可以给高阶模型提供有效信息。

常用组合估计模型 { 线性插值(混合一、二、三元)
回退(三元优先，二元次之，一元最后)

线性插值

三元：

$$P_{interp}(W_i|W_{i-2}W_{i-1}) = \lambda_3 P_{ML}(W_i|W_{i-2}W_{i-1}) + \lambda_2 P_{ML}(W_i|W_{i-1}) + \lambda_1 P_{ML}(W_i)$$

$0 \le \lambda_i \le 1$ 且 $\sum \lambda_i = 1$

假设C(he send the)= C(he send thou)=0

$$p_{interp}(the \mid he\ send) = \lambda_3 \frac{C(he\ send\ the)}{C(he\ send)} + \lambda_2 \frac{C(send\ the)}{C(send)} + \lambda_1 \frac{C(the)}{N}$$

V || V V

$$p_{interp}(thou \mid he\ send) = \lambda_3 \frac{C(he\ send\ thou)}{C(he\ send)} + \lambda_2 \frac{C(send\ thou)}{C(send)} + \lambda_1 \frac{C(thou)}{N}$$

二元

$$P_{interp}(w_i|w_{i-1}) = \lambda P_{ML}(w_i|w_{i-1}) + (1-\lambda) P_{ML}(w_i)$$

语言模型性能评价

- 外部评估：将语言模型用于其它任务，以任务的指标衡量模型（问题：不够客观）
- 内部评估：困惑度（问题：不够精确）

困惑度（perplexity）：测试集概率的倒数，直观来说，就是模型预测某句话的下一个词，猜得准，困惑度低

$$PP(W) = P(w_1 \cdots w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1 \cdots w_N)}}$$

$$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1 \cdots w_{i-1})}} \qquad (W = w_1 \cdots w_N)$$

对于二元语法：$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}}$

PP(W)是对W这个句子的困惑度，而模型的困惑度会综合许多PP(W)来计算（如取均值）

例、句子由完全随机的数字（0~9）构成，则困惑度是多少

即：每个位置出现数字i的概率为 $\frac{1}{10}$

$$PP(W) = \sqrt[N]{\left(\frac{1}{10}\right)^N} = 10$$