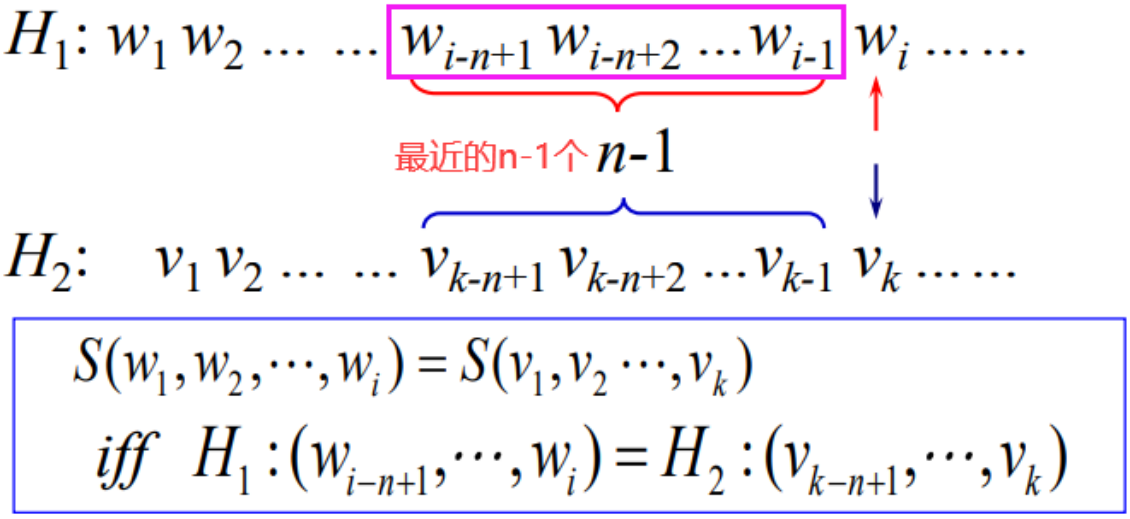
基本概念

设法减少历史基元的个数，将历史映射到等价类，使等价类的数目远远小于原来不同历史基元的数目

$$p(w_i \mid w_1, \cdots, w_{i-1}) = p(w_i \mid S(w_1, \cdots, w_{i-1}))$$

将两个历史映射到同一个等价类，当且仅当这两个历史中的最近的 n-1 个基元相同



这种情况下的语言模型称为 n-gram 模型，其实就是一个位置的概率仅与其前面的 n-1 个位置有关。

- n = 1时，基元 w_i 独立于历史，不用算条件概率
- n = 2时，bi-gram被称为1阶马尔科夫链
- n = 3时，tri-gram被称为2阶马尔科夫链
- 以此类推

为了保证条件概率在 i = 1时有意义，同时保证句子内所有字符串的概率和为1，可以在句子首尾增加开始和结束标志

分解为：

$$p(s) = \prod_{i=1}^{m+1} p(w_i \mid w_{i-n+1}^{i-1}) \qquad \ldots (5\text{-}4)$$

其中，w_i^j 表示词序列 $w_i \ldots w_j$，w_{i-n+1} 从 w_0 开始，w_0 为 <BOS>，w_{m+1} 为 <EOS>。

例子

给定句子：John read a book

增加标记：<BOS> John read a book <EOS>

Unigram：<BOS>, John, read, a, book, <EOS>

Bigram：(<BOS>John), (John read), (read a), (a book), (book <EOS>)

Trigram：(<BOS>John read), (John read a), (read a book), (a book <EOS>)

- 基于二元文法

基于2元文法的概率为：

$$p(\text{John read a book}) = p(\text{John}|\text{<BOS>}) \times p(\text{read}|\text{John}) \times p(\text{a}|\text{read}) \times p(\text{book}|\text{a}) \times p(\text{<EOS>}|\text{book})$$

应用

在给定拼音翻译成汉字的任务中，我们可以得到目标方程

-

$$\begin{aligned}\widehat{CString} &= \arg\max_{CString} p(CString \mid Pinyin) \\ &= \arg\max_{CString} \frac{p(Pinyin \mid CString) \times p(CString)}{p(Pingyin)} \quad \text{不变的} \\ &= \arg\max_{CString} p(Pinyin \mid CString) \times p(CString) \quad \text{汉字对应的拼音是固定的，概率是1} \\ &= \arg\max_{CString} p(CString)\end{aligned}$$

如果汉字总数为 N

- 一元语法

    - 样本空间为 N
    - 只选择使用出现频率最高的汉字
- 二元语法
    - 样本空间是 N^2
    - 效果明显提高

那我们如何获得 n-gram 语法模型呢（模型里面的一些概率，需要知道才能求整个句子的概率）

参数估计

采用最大似然估计的方法，就是直接统计，用历史串出现且w_i出现的次数除以历史串出现的次数当做条件概率

对于 ***n*-gram**，参数 $p(w_i \mid w_{i-n+1}^{i-1})$可由最大似然估计求得：

说最大似然有些牵强，其实就是统计

$$p(w_i \mid w_{i-n+1}^{i-1}) = f(w_i \mid w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i})}{\sum_{w_i} c(w_{i-n+1}^{i})} \quad \ldots(5\text{-}5)$$

其中，$\sum_{w_i} c(w_{i-n+1}^{i})$ 是历史串 w_{i-n+1}^{i-1} 在给定语料中出现的次数，即 $c(w_{i-n+1}^{i-1})$，不管 w_i 是什么。

$f(w_i \mid w_{i-n+1}^{i-1})$ 是在给定 w_{i-n+1}^{i-1} 的条件下 w_i 出现的相对频度，分子为 w_{i-n+1}^{i-1} 与 w_i 同现的次数。

这样算，由于数据集不能包含所有情况的出现，会引起零概率问题，连乘有一项是0，整个结果就是0，一票否决了其他项的概率，这是不合理的。

数据平滑

思想：使零概率增大，非零概率减小，消除零概率

目标：测试样本的语言模型困惑度越小越好

困惑度

对于一个平滑的 ***n*-gram**，其概率为 $p(w_i \mid w_{i-n+1}^{i-1})$，

可以计算句子的概率：（一个句子的概率由其词项概率相乘得到）

$$p(s)=\prod_{i=1}^{m+1} p(w_i \mid w_{i-n+1}^{i-1})$$

假定测试语料 T 由 l_T 个句子构成 $(t_1,\ldots t_{l_T})$，则整个测试集的概率为：（测试语料的概率由里面所有句子的概率相乘得到）

$$p(T)=\prod_{i=1}^{l_T} p(t_i)$$

模型 $p(w_i \mid w_{i-n+1}^{i-1})$ 对于测试语料的交叉熵：（对整个测试语料）

$$H_p(T)=-\frac{1}{W_T}\log_2 p(T)$$

其中，W_T 是测试文本 T 的词数。

模型 p 的困惑度 $PP_p(T)$ 定义为：$PP_P(T)=2^{H_p(T)}$

***n*-gram 对于英语文本的困惑度范围一般为 50 ~ 1000，对应于交叉熵范围为 6 ~ 10 bits/word。**

加 1 平滑

每一种情况出现的次数加 1（对于所有的基元，每种基元出现的次数都加 1）

减值法/折扣法

修改训练样本中事件的实际计数，使实际出现的不同事件的概率之和小于1，剩余的概率量分配给未见概率

Good-Turing估计

设 N 是原来训练样本数据的大小，n_r 是在样本中正好出现 r 次的事件的数目

- 平滑：保持出现的总样本数不变，但是将一些样本数分给了在样本中没有出现过的事件
  - 没出现的样本数指的是没出现样本的数学期望

那么，$N=\sum_{r=1}^{\infty} n_r r=\sum_{r=0}^{\infty}(r+1)n_{r+1}$ 原来的 … (5-6)

设：原先出现r次的n-gram在平滑后出现r^*次

平滑后的，考虑了样本中出现 0 次的

则 $N=\sum_{r=0}^{\infty} n_r r^*$ 则 $\sum_{r=0}^{\infty} n_r r^* = \sum_{r=0}^{\infty}(r+1)n_{r+1}$

所以，$r^*=(r+1)\frac{n_{r+1}}{n_r}$

实际应用中，问题都是离散的，出现次数最多的样本的出现次数不变，因为不存在n_{r+1}

平滑后，时间出现的概率如下：

那么，Good-Turing 估计在样本中出现 r 次的事件的平滑后的概率为：

$$p_r=\frac{r^*}{N} \quad \dots (5\text{-}7)$$

原样本中所有时间的概率之和为：

$$\sum_{r>0} n_r \times p_r = 1-\frac{n_1}{N}<1$$

$\frac{n_1}{N}$ 的概率剩余量可以分给没出现的事件（r=0），是没有出现过的样本的概率总和

例子

对以read开头的bi-gram进行平滑，要注意的是最后一项没变

r	n_r	r^*
1	2053	0.446
2	458	1.25
3	191	2.24
4	107	3.22
5	69	4.17
6	48	5.25
7	36	保持原来的计数7

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

因为$n_{r+1}=0$

- 以 read 开始，没有出现过的bi-gram的概率总和为 $p_0 = \frac{n_1}{N}$
- 以 read 开始，没出现过的bi-gram的个数为（假定语料库中每次词都可以出现在 read 后面）

$$n_0 = |V_T| - \sum_{r>0} n_r \quad \text{其中，} |V_T| \text{ 为语料的词汇量。}$$

- 以 read 为开始的2-gram的 概率平均为 $\frac{p_0}{n_0}$

注意：$\sum_{r=0}^{7} p_r \neq 1$

因为最后一项没变

因此，需要归一化处理：

$$\hat{p}_r = \frac{p_r}{\sum_r p_r}$$

适用于大词汇集产生的符合多项式分布的大量的观察数据

Back-off（后退/后备）方法

对每个计数 $r > 0$ 的 n 元文法的出现次数减值，把因减值而省下来的剩余概率根据低阶的 （n-1）gram分配给未见事件。

以2-gram模型为例，说明Katz平滑方法：

对于一个出现次数为 $r = c(w_{i-1}^{i})$ 的 2-gram w_{i-1}^{i}，修正其概率：

$$p_{katz}(w_i|w_{i-1}) = \begin{cases} d_r \dfrac{C(w_{i-1}w_i)}{C(w_{i-1})} & \text{if } C(w_{i-1}w_i) = r > 0 \\ \alpha(w_{i-1}) p_{\mathrm{ML}}(w_i) & \text{if } C(w_{i-1}w_i) = 0 \end{cases}$$

其中，$p_{\mathrm{ML}}(w_i)$表示 w_i 的最大似然估计概率。

公式的意思是，所有具有非零计数 r 的 2-gram都根据折扣率d_r ($0<d_r<1$)被减值了，折扣率 d_r 近似地等于 r^*/r，减值由Good-Turing估计方法预测。

那么，如何确定$\alpha(w_{i-1})$呢？ $\sum_{w_i} p_{katz}(w_i|w_{i-1}) = 1$

$$\sum_{w_{i:r=0}} p_{katz}(w_i|w_{i-1}) + \sum_{w_{i:r>0}} p_{katz}(w_i|w_{i-1}) = 1$$

$$\sum_{w_{i:r=0}} \alpha(w_{i-1}) p_{ML}(w_i) + \sum_{w_{i:r>0}} p_{katz}(w_i|w_{i-1}) = 1$$

$$\alpha(w_{i-1}) = \frac{1 - \sum_{w_{i:r>0}} p_{katz}(w_i|w_{i-1})}{\sum_{w_{i:r=0}} p_{ML}(w_i)}$$

- **绝对减值法**

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

那么，样本出现了 r 次的事件的概率可以由如下公式估计：

$$p_r = \begin{cases} \dfrac{r-b}{N} & \text{当 } r>0 \\ \dfrac{b(R-n_0)}{Nn_0} & \text{当 } r=0 \end{cases} \quad \cdots (5\text{-}10)$$

其中，n_0 为样本中未出现的事件的数目。b 为减去的常量，$b \le 1$。

- N 是各种 r 的和
- R 是所有可能事件的数目，当事件为 n-gram 时，如果统计基元为词，且词汇集大小为 L，则 $R = L^n$
- 对于出现了的事件，每个都贡献了 $\dfrac{b}{N}$ 的概率给没有出现的事件（和为 n_0）

线性减值法

从每个计数 r 中减去与该计数成正比的量

$$p_r = \begin{cases} (1-\alpha)\dfrac{N_r}{N} & \text{当 } r>0 \\ \dfrac{\alpha}{n_0} & \text{当 } r=0 \end{cases} \quad \cdots (5\text{-}12)$$

自由参数 α 的优化值为：$\dfrac{n_1}{N}$

每个存在计数贡献概率为 $\alpha * \dfrac{N_r}{N}$，对 r 求和，得到存在计数总的贡献概率为 α

绝对减值法产生的 n-gram 通常优于线性减值法

总结

- **Good-Turing 法：**对非0事件按公式削减出现的次数，节留出来的概率均分给0概率事件。
- **Katz 后退法：**对非0事件按Good-Turing法计算减值，节留出来的概率按低阶分布分给0概率事件。
- **绝对减值法：**对非0事件无条件削减某一固定的出现次数值，节留出来的概率均分给0概率事件。
- **线性减值法：**对非0事件根据出现次数按比例削减次数值，节留出来的概率均分给0概率事件。

删除插值法

用低阶语法估计高阶语法

- 3-gram不能准确估计时用 2-gram 代替
- 下面同理

$$p(w_3 \mid w_1 w_2) = \lambda_3 p'(w_3 \mid w_1 w_2) + \lambda_2 p'(w_3 \mid w_2) + \lambda_1 p'(w_3) \quad \ldots (5\text{-}13)$$

$$\text{其中，} \lambda_1 + \lambda_2 + \lambda_3 = 1$$

将训练语料分为两部分，从原始语料中删除一部分作为留存数据

第一部分用于估计 $p'(w_3 \mid w_1 w_2)$，$p'(w_3 \mid w_2)$ 和 $p'(w_3)$。

第二部分用于计算 $\lambda_1, \lambda_2, \lambda_3$：使语言模型对留存数据的困惑度最小。

语言模型的自适应

训练语言模型时的语料往往来自不同的领域，语言模型对训练文本的类型，主题等都很敏感

n元语言模型的独立性假设的前提在很多情况下不成立

基于缓存的语言模型

问题：在文本中刚刚出现过的一些词在后边句子中再次出现的可能性往往较大，大于n-gram模型预测的概率

解决的思路：语言模型通过 n-gram 的线性插值求得

$$\hat{p}(w_i \mid w_1^{i-1}) = \lambda \hat{p}_{Cache}(w_i \mid w_1^{i-1}) + (1-\lambda)\hat{p}_{n-gram}(w_i \mid w_{i-n+1}^{i-1})$$

缓存预测的　　n-gram预测的

- 通常的做法，认为前面 K 个词在缓存中，一个词的概率等于其在缓存中出现的概率

$$\hat{p}_{Cache}(w_i \mid w_1^{i-1}) = \frac{1}{K}\sum_{j=i-K}^{i-1} I_{\{w_j=w_i\}} \quad \ldots (5\text{-}15)$$

w_i在前面K个词中出现多少次

其中，I_ε为指示器函数(indicator function)，如果词重复出现(w_j=w_i)，则$I_\varepsilon=1$，否则$I_\varepsilon=0$。

  - 缺陷：缓存中词的重要性独立于该词与当前词的距离
- 改进：缓存中每个词对当前词的影响随与该词的距离的增大呈指数级衰减
  - 缓存中不止存 K 个，而是存前面所有的词，影响程度随距离增大而衰减

$$\hat{p}_{Cache}(w_i \mid w_1^{i-1}) = \beta\sum_{j=1}^{i-1} I_{\{w_i=w_j\}} e^{-\alpha(i-j)} \quad \ldots(5\text{-}16)$$

从1开始，不止在缓存中存K个词

其中，α 为衰减率，β为归一化常数，以使得：$\sum_{w_i \in V}\hat{p}_{Cache}(w_i \mid w_1^{i-1})=1$，$V$为词汇表。

基于混合方法的语言模型

问题：训练数据往往是不同领域的，但是测试语料一般是同源的。为了获得最佳心梗，模型必须适应各种不同类型的语料对其的影响

处理方法：将语言模型划分成 n 个子模型，整个模型通过线性插值得到

$$\hat{p}(w_i \mid w_1^{i-1}) = \sum_{j=1}^{n}\lambda_j \hat{p}_{M_j}(w_i \mid w_1^{i-1}) \quad \ldots(5\text{-}17)$$

其中，$0 \le \lambda_j \le 1$，$\sum_{j=1}^{n}\lambda_j = 1$

λ值可以通过 EM 算法计算出来。

基本方法：

- 将训练语料聚类
- 模型运行时识别测试语料的主题
- 确定适当的训练语料自己，分别建立特定的语言模型
- 插值获得整个语言模型

EM迭代计算插值系数（感觉可以不看）

- 对 n 个类，随机初始化插值系数 λ
- 根据整个模型计算概率和期望
- 更新

第 r 次迭代，第 j 个语言模型在第 $i\ (i \le n)$ 类上的系数：

$$\lambda_{ij}^{r} = \frac{\lambda_{ij}^{r-1} p_{ij}(w \mid h)}{\sum_{i=1}^{n} \lambda_{ij}^{r-1} p_{ij}(w \mid h)}$$

其中，h 为历史。

第j个模型在所有类上的

- 不断迭代，直到收敛

基于最大熵的语言模型

结合不同信息源的信息构建一个语言模型，每个信息源提供一组关于模型参数的约束条件。

在所有满足约束的模型中，选择熵最大的模型。

比如有两个信息源，f，g 分别是两个模型定义的函数（约束）

$$\hat{p}_{\mathrm{M}_1}(w_i \mid w_1^{i-1}) = f(w_i, w_{i-1})$$

$$\hat{p}_{\mathrm{M}_2}(w_i \mid w_1^{i-1}) = g(w_i, w_{i-2})$$

- 这两个信息源都分别提供了一组约束
- 我们并不是让这些公式对所有可能的历史都成立，而是更宽松的限制。即它们在训练数据上的平均成立即可

$$E(\hat{p}_{M_1}(w_i \mid w_1^{i-1}) \mid w_{i-1} = a) = f(w_i, a)$$

$$E(\hat{p}_{M_2}(w_i \mid w_1^{i-1}) \mid w_{i-2} = b) = g(w_i, b)$$

语言模型应用举例

基于语言模型的分词方法

最基本的做法是以词为独立的统计基元，但效果不大好

- **方法描述：**

设对于待切分的句子$S = z_1 z_2 \ldots z_m$，$W = w_1 w_2 \ldots w_k$ $(1 \le k \le n)$ 是一种可能的切分。那么，

$$\begin{aligned}\hat{W} &= \arg\max_W p(W \mid S) \\ &= \arg\max_W p(W) \times p(S \mid W) \\ &\cong \arg\max_W p(W)\end{aligned}$$

最基本的做法是以词为独立的统计基元，但效果不佳。

具体实现时，将词分类，将词序列转成词类序列

进一步做如下约定，把一个可能的词序列 W 转换成词类序列 $C = c_1 c_2 \cdots c_N$，即：

- **专有名词：人名PN、地名LN、机构名ON分别作为一类；**
- **实体名词中的日期 dat、时间tim、百分数per、货币mon 等作为一类；**
- **对词法派生词MW和词表词LW，每个词单独作为一类。**

词序列变类序列

dat/ tim/ PN/ 在/ LN/ 发表/ 讲话/ ，/ 决定/ 从/ tim/ 起/ ON/ 将/ 大/ 规模/ 招收/ 研发/ 人员/ ，/ 其中/ ，/ per/ 将/ 从/ 山东/ 大学/ 所/ 培养/ 的/ 应届/ 博士/ 或/ 硕士/ 毕业生/ 中/ 选拔/ ，/ 年薪/ 不/ 低于/ mon/ 。

求概率

那么，
$$\hat{C} = \arg\max_{C} p(C \mid S)$$
$$= \arg\max_{C} p(C) \times p(S \mid C) \quad \ldots (5\text{-}22)$$

语言模型　　生成模型

$p(C)$ 可采用三元语法：

$$p(C) = p(c_1) \times p(c_2 \mid c_1) \prod_{i=3}^{N} p(c_i \mid c_{i-2} c_{i-1}) \quad \ldots (5\text{-}23)$$

$$p(c_i \mid c_{i-2} c_{i-1}) = \frac{count(c_{i-2} c_{i-1} c_i)}{count(c_{i-2} c_{i-1})} \quad \ldots (5\text{-}24)$$

生成模型在满足独立性假设的条件下，可近似为：

$$p(S \mid C) \approx \prod_{i=1}^{N} p(s_i \mid c_i) \quad \ldots (5\text{-}25)$$

该公式的含意是，任意一个词类 c_i 生成汉字串 s_i 的概率只与自身有关，而与其上下文无关。

分词与词性标注一体化

假设句子S是由单词串组成：

$$W = w_1 w_2 \cdots w_n \quad (n \geq 1)$$

单词 $w_i\,(1 \leq i \leq n)$ 的词性标注为 t_i，即句子S相应的词性标注符号序列可表达为：

$$T = t_1 t_2 \cdots t_n$$

那么，分词与词性标注的任务就是要在S所对应的各种切分和标注形式中，寻找 T 和 W 的联合概率 $p(W, T)$为最优的词切分和标注组合。

- **基于词性的三元统计模型**

$$p(W,T) = p(W \mid T) \times p(T) \approx \prod_{i=1}^{n} p(w_i \mid t_i) \times p(t_i \mid t_{i-1}, t_{i-2}) \quad \ldots(5\text{-}26)$$

- **基于单词的三元统计模型**

$$p(W,T) = p(T \mid W) \times p(W) \approx \prod_{i=1}^{n} p(t_i \mid w_i) \times p(w_i \mid w_{i-1}, w_{i-2})$$

- **分词与词性标注一体化模型**

$$p^*(W,T) = \alpha \prod_{i=1}^{n} p(w_i \mid t_i) \times p(t_i \mid t_{i-1}, t_{i-2}) + \beta \prod_{i=1}^{n} p(t_i \mid w_i) \times p(w_i \mid w_{i-1}, w_{i-2})$$

舍弃一部分没有用的

$$p(W,T)=p(T\mid W)\times p(W)\approx\prod_{i=1}^{n}p(t_i\mid w_i)\times p(w_i\mid w_{i-1},w_{i-2})$$ … (5-27)

只取后半部分

$p(t_i\mid w_i)$ 对分词无帮助，且在分词确定后对词性标注又会增添偏差。

因此，可以仅取(5-27) 中的语言模型部分，而舍弃词性标注部分，并令(5-28) 中的$\alpha=1$，仅保留加权系统 β，于是，(5-28) 式成为：

基于词性的原始式子

在原始式子上增加了对分词的约束

$$\hat{p}(W,T)=\prod_{i=1}^{n}p(w_i\mid t_i)\times p(t_i\mid t_{i-1},t_{i-2})+\beta\prod_{i=1}^{n}p(w_i\mid w_{i-1},w_{i-2})$$ …(5-29)

计算 β

在确定β 系数值时，可以根据词典中词汇w的个数和词性t 的种类数目，取二者之比，即：

β=词典中词w的个数 / 词性t 的种类数。

在系统实现时，首先对训练文本进行预处理，将人名、地名和数字串先识别出来，然后用规定的符号分别予以替代，最后再计算相应的条件概率。