

机器翻译原理与方法

第五讲 基于词的统计机器翻译方法

刘群

中国科学院计算技术研究所

liuqun@ict.ac.cn

中国科学院计算技术研究所2011年秋季课程

内容提要

- 为翻译建立概率模型
- IBM 的信源信道模型
- 语言模型 – – n 元语法模型
- 翻译模型 – – IBM 模型1-5
- 词语对齐算法
- 解码算法
- Candide 系统
- Egypt 工具包与 Giza++
- 机器翻译自动评价

为翻译建立概率模型

- 假设任意一个英语句子 **e** 和一个法语句子 **f**, 我们定义 **f** 翻译成 **e** 的概率为:

$$P(e|f)$$

其归一化条件为:

$$\sum_e P(e|f) = 1$$

- 于是将 **f** 翻译成 **e** 的问题就变成求解问题:

$$\hat{e} = \operatorname{argmax}_e P(e|f)$$

内容提要

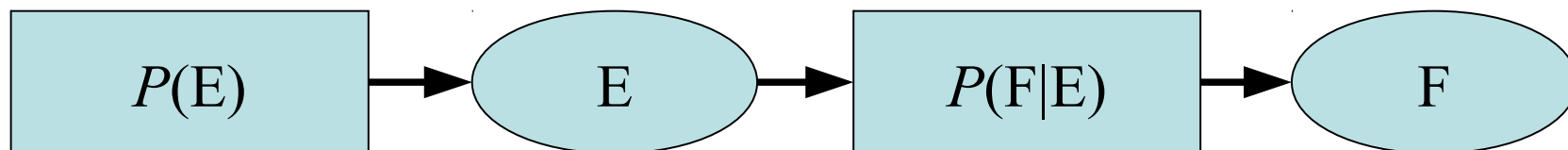
- 为翻译建立概率模型
- **IBM 的信源信道模型**
- 语言模型 – – n 元语法模型
- 翻译模型 – – **IBM 模型1-5**
- 词语对齐算法
- 解码算法
- **Candide 系统**
- **Egypt 工具包与 Giza++**
- 机器翻译自动评价

信源信道模型 (1)

- 信源信道模型又称噪声信道模型，是由 IBM 公司的 Peter F. Brown 等人于1990年提出来的：

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, Paul S. Roossin, *A Statistical Approach to Machine Translation*, Computational Linguistics, 1990

信源信道模型 (2)



- 假设我们看到的源语言文本 F 是由一段目标语言文本 E 经过某种奇怪的编码得到的，那么翻译的目标就是要将 F 还原成 E ，这也就是就是一个解码的过程。
 - 注意，在信源信道模型中：
 - 噪声信道的源语言是翻译的目标语言
 - 噪声信道的目标语言是翻译的源语言
- 这与整个机器翻译系统翻译方向的刚好相反

统计机器翻译基本方程式

$$E = \operatorname{argmax}_E P(E) P(F|E)$$

- P.Brown 称上式为统计机器翻译基本方程式
 - 语言模型: $P(E)$
 - 翻译模型: $P(F|E)$
- 语言模型反映 “ E 像一个句子 ” 的程度: 流利度
- 翻译模型反映 “ F 像 E ” 的程度: 忠实度
- 联合使用两个模型效果好于单独使用翻译模型, 因为后者容易导致一些不好的译文。

语言模型与翻译模型

- 考虑汉语动词“打”的翻译：有几十种对应的英语词译文：
打人，打饭，打鱼，打毛衣，打猎，打草稿，……
- 如果直接采用翻译模型，就需要根据上下文建立复杂的上下文条件概率模型
- 如果采用信源—信道思想，只要建立简单的翻译模型，可以同样达到目标词语选择的效果：
 - 翻译模型：不考虑上下文，只考虑单词之间的翻译概率
 - 语言模型：根据单词之间的同现选择最好的译文词

统计机器翻译的三个问题

- 三个问题：
 - 语言模型 $P(E)$ 的建模和参数估计
 - 翻译模型 $P(F|E)$ 的建模和参数估计
 - 解码（搜索）算法

内容提要

- 为翻译建立概率模型
- IBM 的信源信道模型
- 语言模型 – – **n 元语法模型**
- 翻译模型 – – **IBM 模型1-5**
- 词语对齐算法
- 解码算法
- **Candide 系统**
- **Egypt 工具包与 Giza++**
- 机器翻译自动评价

语言模型

- 统计语言模型把一种语言理解成是产生一个句子的随机事件。在统计语言模型看来，对于一种语言，任何一个句子都是可以接受的，只是接受的可能性（概率）不同
- 语言模型给出任何一个句子的出现概率：

$$P(E=e_1e_2...e_3)$$

$$\text{归一化条件: } \sum_E P(E)=1$$

- 统计语言模型实际上就是一个概率分布，它给出了一种语言中所有可能的句子的出现概率

N 元语法模型—概念辨析

- N 元语法模型： N-Gram Model。
- 所谓 **N-Gram**， 指的是由 **N 个词组成的串**， 可以称为“N 元组”， 或“N 元词串”。
- 基于 N-Gram 建立的语言模型， 称为“N 元语法模型 (N-Gram Model)”。
- Gram 不是 Grammar 的简写。在英文中， 并没有 N-Grammar 的说法。
- 在在汉语中， 单独说“N 元语法”的时候， 有时指“N 元组 (N-Gram)”， 有时指“N 元语法模型 (N-Gram Model)”， 请注意根据上下文加以辨别

N 元语法模型一定义

- N 元语法模型（ N-gram Model）

$$P(w) = \prod_{i=1}^n P(w_i | w_1 w_2 \dots w_{i-1})$$
$$\approx \prod_{i=1}^n P(w_i | w_{i-N+1} w_{i-N+2} \dots w_{i-1})$$

- 假设：单词 w_i 出现的概率只与其前面的 N-1 个单词有关

N 元语法模型—举例

- **N=1 时：一元语法模型**
 - 相当于词频表，给出所有词出现的频率
- **N=2 时：二元语法模型**
 - 相当于一个转移矩阵，给出每一个词后面出现另一个词的概率
- **N=3 时：三元语法模型**
 - 相当于一个三维转移矩阵，给出每一个词对儿后面出现另一个词的概率
- 在自然语言处理中，**N 元语法模型**可以在汉字层面，也可以在单词层面，还可以在概念层面……

内容提要

- 为翻译建立概率模型
- IBM 的信源信道模型
- 语言模型 - - n 元语法模型
- 翻译模型 - - IBM 模型1-5
- 词语对齐算法
- 解码算法
- Candide 系统
- Egypt 工具包与 Giza++
- 机器翻译自动评价

翻译模型

- 翻译模型 $P(F|E)$ 反映的是一个源语言句子 E 翻译成一个目标语言句子 F 的概率
- 由于源语言句子和目标语言句子几乎不可能在语料库中出现过，因此这个概率无法直接从语料库统计得到，必须分解成词语翻译的概率和句子结构（或者顺序）翻译的概率

翻译模型与对齐

- 翻译模型的计算，需要引入隐含变量：
对齐 A :

$$P(F|E) = \sum_A P(F, A|E)$$

- 翻译概率 $P(F|E)$ 的计算转化为对齐概率 $P(F, A|E)$ 的估计
- 对齐：建立源语言句子和目标语言句子的词与词之间的对应关系和句子结构之间的对应关系

词语对齐的表示 (1)

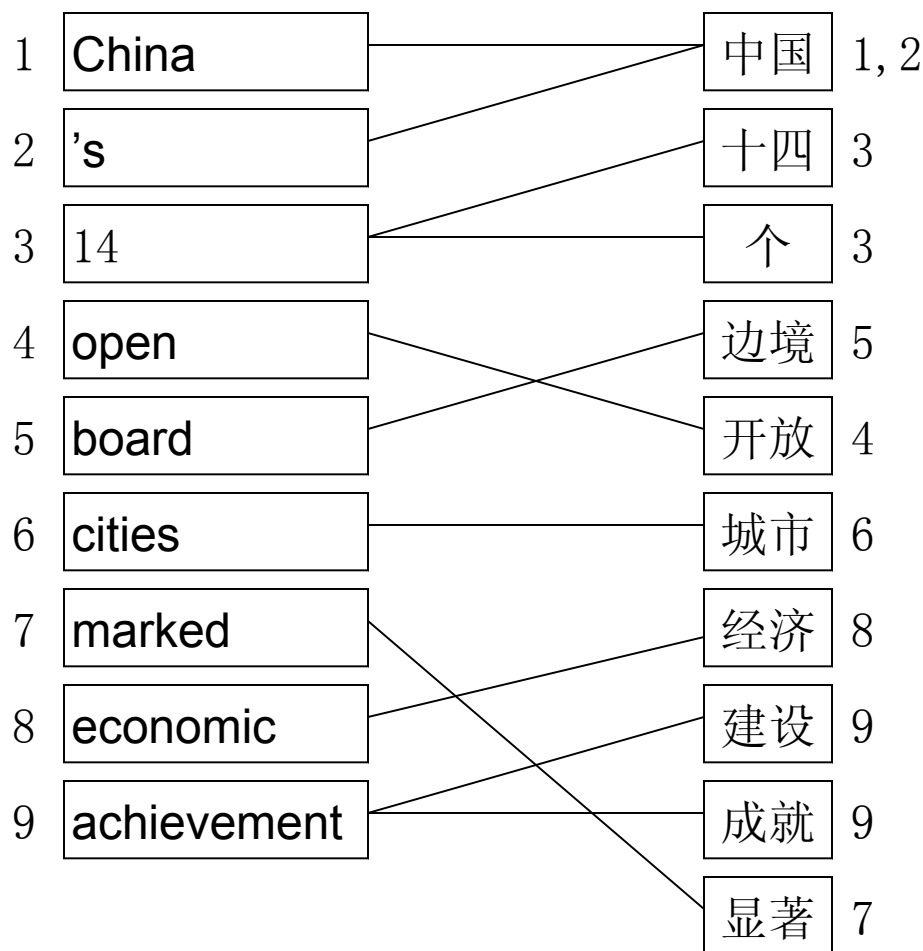
- 图形表示

- ✓ 连线

- ✓ 矩阵（见下页）

- 数字表示

- ✓ 给每个目标语言单词标记其所有对应的源语言单词



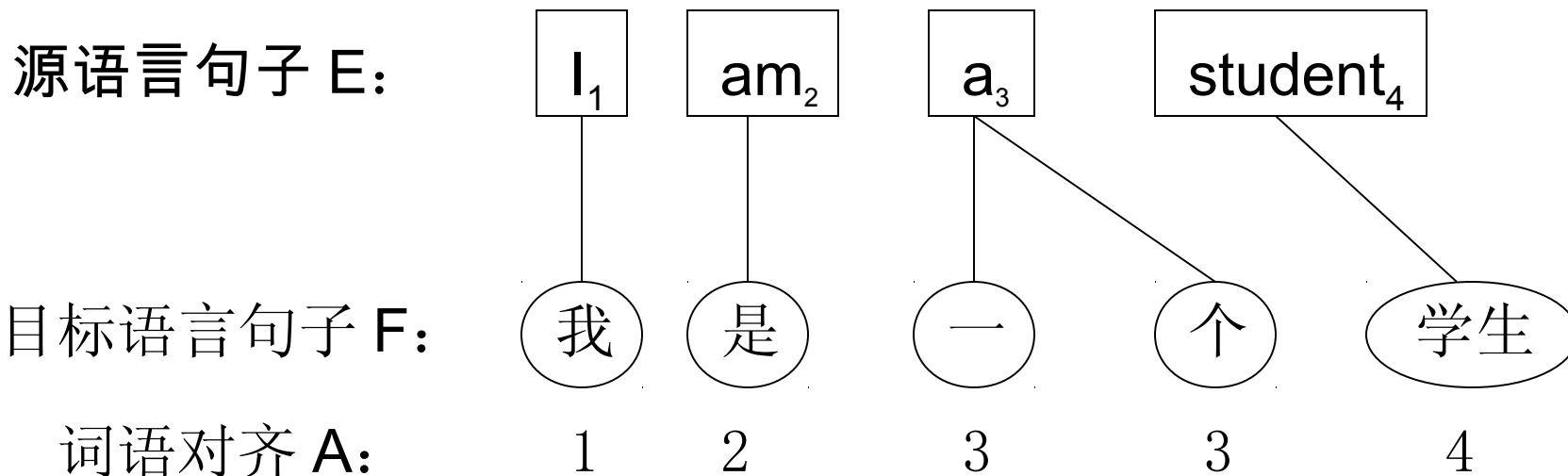
词语对齐的表示 (2)

achievement										
economic										
marked										
cities										
board										
open										
14										
's										
China										
	中国	十四	个	边境	开放	城市	经济	建设	成就	显著

IBM Model

- 对 $P(F,A|E)$ 的估计
- IBM Model 1 仅考虑词对词的互译概率
- IBM Model 2 加入了词的位置变化的概率
- IBM Model 3 加入了一个词翻译成多个词的概率
- IBM Model 4
- IBM Model 5

IBM Model 1 & 2 推导方式 (1)



IBM 模型1&2的推导过程:

1. 猜测目标语言句子长度;
2. 从左至右, 对于每个目标语言单词:
 - 首先猜测该单词由哪一个源语言单词翻译而来;
 - 再猜测该单词应该翻译成什么目标语言词。

IBM Model 1 & 2 推导方式 (2)

假设翻译的目标语言句子为: $F = f_1^m = f_1 f_2 \dots f_m$

假设翻译的源语言句子为: $E = e_1^l = e_1 e_2 \dots e_l$

假设词语对齐表示为:

$$A = a_1^m = a_1 a_2 \dots a_m, \forall i \in \{1, \dots, m\}, a_i \in \{0, \dots, l\}$$

那么词语对齐的概率可以表示为:

$$P(F, A|E) = P(m|E) \prod_{j=1}^m P(a_j | a_1^{j-1}, f_1^{j-1}, m, E) P(f_j | a_1^j, f_1^{j-1}, m, E)$$

注意: 在 **IBM Model** 中, 词语对齐只考虑了源语言到目标语言的单向一对多形式, 不考虑多对一和多对多的形式。

IBM Model 1 的推导 (1)

假设所有翻译长度都是等概率的: $P(m|E)=\epsilon$

假设词语对齐只与源语言长度有关, 与其他因素无关:

$$P(a_j|a_1^{j-1}, f_1^{j-1}, m, E) = \frac{1}{l+1}$$

假设目标词语的选择只与其对应的源语言词语有关, 与其他因素无关:

$$P(f_j|a_1^j, f_1^{j-1}, m, E) = t(f_j|e_{a_j})$$

IBM Model 1 的推导(2)

那么对齐概率可以表示为：

$$P(F, A|E) = \frac{\epsilon}{(l+1)^m} \prod_{j=1}^m t(f_j | e_{a_j})$$

对所有可能的对齐求和，那么翻译概率就可以表示为：

$$P(F|E) = \sum_A P(F, A|E) = \frac{\epsilon}{(l+1)^m} \sum_{a_1=1}^l \dots \sum_{a_m=1}^l \prod_{j=1}^m t(f_j | e_{a_j})$$

这就是 IBM Model 1 的**翻译模型公式**。

也就是说，给定参数 $t(f|e)$ ，我们就可以计算出句子 **E** 翻译成句子 **F** 的概率。

IBM Model 1 的参数求解(1)

- 在 IBM Model 1 中, ε 是个常数, 无关紧要, 起重要作用的就是单词翻译概率分布:

$$t(f|e)$$

- 这个单词翻译概率分布表现为一个翻译概率表, 这个表给出了每一个源语言单词翻译成任何一个目标语言单词的概率, 并且这个概率满足归一性约束条件:

$$\sum_f t(f|e)=1$$

IBM Model 1 的参数求解 (2)

- 根据最大似然估计，我们希望得到一组概率分布，使得我们的训练语料库出现的概率最大。
- 也就是说，给定训练语料库 E 和 F ，我们要求解一个概率分布 $t(f|e)$ ，使得翻译概率 $P(F|E)$ 最大。
- 这是一个受约束的极值问题，约束条件即是 $t(f|e)$ 的归一性条件。
- 为了求解这个问题，我们需要引入拉格朗日乘子，构造一个辅助函数，将上述受约束的极值问题转换成一个不受约束的极值问题。

IBM Model 1 的参数求解 (3)

- 引入拉格朗日乘子 λ_e ，构造辅助函数如下：

$$h(t, \lambda) \equiv \frac{\epsilon}{(l+1)^m} \sum_{a_1=1}^l \dots \sum_{a_m=1}^l \prod_{j=1}^m t(f_j | e_{a_j}) - \sum_e \lambda_e \left(\sum_f t(f | e) - 1 \right)$$

- 将上述函数对 $t(f | e)$ 求导得到：

$$\frac{\partial h(t, \lambda)}{\partial t(f | e)} = \frac{\epsilon}{(l+1)^m} \sum_{a_1=1}^l \dots \sum_{a_m=1}^l \delta(f, f_j) \delta(e, e_{a_j}) \frac{\prod_{k=1}^m t(f_k | e_{a_k})}{t(f | e)} - \lambda_e$$

IBM Model 1 的参数求解 (4)

- 令上式为0，我们得到：

$$t(f|e) = \lambda_e^{-1} \frac{\epsilon}{(l+1)^m} \sum_{a_1=1}^l \dots \sum_{a_m=1}^l \delta(f, f_j) \delta(e, e_{a_j}) \prod_{k=1}^m t(f_k | e_{a_k})$$

- 我们看到，这个公式的左边和右边都出现了 $t(f|e)$
- 我们无法直接用这个公式从给定的语料库 ($F|E$) 中计算出 $t(f|e)$
- 我们可以将这个公式看成是一个**迭代公式**，给定一个初值 $t(f|e)$ ，利用这个公式反复迭代，最后可以收敛到一个稳定的 $t(f|e)$ 值，这就是 **EM 算法**。

IBM Model 1 的参数求解 (5)

- 上述迭代公式代入 IBM Model 1 的翻译模型公式, 我们得到:

$$t(f|e) = \lambda_e^{-1} \sum_A P(F, A|E) \underbrace{\sum_{j=1}^m \delta(f, f_j) \delta(e, e_{a_j})}_{\text{对齐 } A \text{ 中 } e \text{ 连接到 } f \text{ 的次数}}$$

- 定义在 E 和 F 的所有可能的对齐 A 下 e 和 f 连接数的均值为:

$$c(f|e; F, E) \equiv \sum_A P(A|F, E) \sum_{j=1}^m \delta(f, f_j) \delta(e, e_{a_j})$$

IBM Model 1 的参数求解 (6)

- 我们有:

$$\begin{aligned} c(f|e; F, E) &= \sum_A \frac{P(F, A|E)}{P(F|E)} \sum_{j=1}^m \delta(f, f_j) \delta(e, e_{a_j}) \\ &= \frac{\sum_A P(F, A|E) \sum_{j=1}^m \delta(f, f_j) \delta(e, e_{a_j})}{P(F|E)} \end{aligned}$$

- 将 $c(f|e; F, E)$ 代入迭代公式, 并将 $P(F|E)$ 并入参数 λ_e , 我们得到新的迭代公式:

$$t(f|e) = \lambda_e^{-1} c(f|e; F, E)$$

IBM Model 1 的参数求解 (7)

- 这个新的迭代公式可以理解为：
 - 一旦我们得到了一组参数 $t(f|e)$ ，我们就可以计算所有的词语对齐的概率 $P(F,A|E)$ ；
 - 有了每个词语对齐的概率 $P(F,A|E)$ ，我们就可以计算新的 $t(f|e)$ 的值，就是所有的出现词语链接 (e,f) 的词语对齐概率之和，并对 e 进行归一化。
- 这个迭代算法就是一个经典的 EM 算法。

IBM Model 1 的参数求解(8)

- 通常，训练语料库 ($\mathbf{F}|\mathbf{E}$) 是由一系列句子对组成的：

$$(\mathbf{F}^{(1)}, \mathbf{E}^{(1)}), (\mathbf{F}^{(2)}, \mathbf{E}^{(2)}), \dots, (\mathbf{F}^{(s)}, \mathbf{E}^{(s)})$$

- 因此实际计算时我们采用以下公式：

$$t(f|e) = \lambda_e^{-1} \sum_s c(f|e; \mathbf{F}^{(s)}, \mathbf{E}^{(s)})$$

- 这里 λ_e 仅仅起到一个归一化因子的作用。

IBM Model 1 的 EM 训练示例 (0 A)

我们用一个简单的例子来演示 EM 训练的过程

- 假设有两个句子对: ($a\ b|x\ y$) 和 ($a\ y$)
- 先假设所有词语翻译概率平均分布 $P(f|e)$:

$P(a x)$	1/2	$P(a y)$	1/2
$P(b x)$	1/2	$P(b y)$	1/2

我们这里为方便起见, 对 IBM Model 1 做了简化:

- 只考虑词语一对一的情况, 不考虑词语一对多或者对齐到空的情况;
- 对齐概率计算的时候, 忽略了词语长度和词语对齐概率, 仅考虑词语翻译概率。

IBM Model 1 的 EM 训练示例 (0 B)

- E 步骤

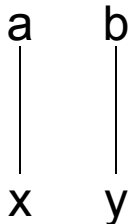


$$P(A|F, E)$$

- M 步骤

$$c(f|e; F, E) \equiv \sum_A P(A|F, E) \sum_{j=1}^m \delta(f, f_j) \delta(e, e_{a_j})$$

$$t(f|e) = \lambda_e^{-1} \sum_s c(f|e; F^{(s)}, E^{(s)})$$

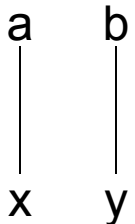


IBM Model 1 的 EM 训练示例 (1 E)

	对所有可能的对齐 计算 $P(F, A E)$	对 $P(F, A E)$ 归一化 得到 $P(A F, E)$
	$P(F, A E) = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$	$P(A F, E) = \frac{1/4}{2/4} = \frac{1}{2}$
	$P(F, A E) = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$	$P(A F, E) = \frac{1/4}{2/4} = \frac{1}{2}$
	$P(F, A E) = \frac{1}{2}$	$P(A F, E) = \frac{1/2}{1/2} = 1$

IBM Model 1 的 EM 训练示例 (1 M)

计算 $c(f e)$	重新计算 $P(f e)$
$c(a x) = \frac{1}{2}$	$P(a x) = \frac{1}{2} / (\frac{1}{2} + \frac{1}{2}) = \frac{1}{2}$
$c(b x) = \frac{1}{2}$	$P(b x) = \frac{1}{2} / (\frac{1}{2} + \frac{1}{2}) = \frac{1}{2}$
$c(a y) = \frac{1}{2} + 1 = \frac{3}{2}$	$P(a y) = \frac{3}{2} / (\frac{3}{2} + \frac{1}{2}) = \frac{3}{4}$
$c(b y) = \frac{1}{2}$	$P(b y) = \frac{1}{2} / (\frac{3}{2} + \frac{1}{2}) = \frac{1}{4}$

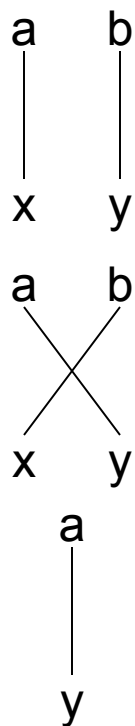
IBM Model 1 的 EM 训练示例 (2 E)

	对所有可能的对齐 计算 $P(F,A E)$	对 $P(F,A E)$ 归一化 得到 $P(A F,E)$
	$P(F,A E) = \frac{1}{2} \times \frac{1}{4} = \frac{1}{8}$	$P(A F,E) = \frac{1/8}{4/8} = \frac{1}{4}$
	$P(F,A E) = \frac{1}{2} \times \frac{3}{4} = \frac{3}{8}$	$P(A F,E) = \frac{3/8}{4/8} = \frac{3}{4}$
	$P(F,A E) = \frac{3}{4}$	$P(A F,E) = \frac{3/4}{3/4} = 1$

IBM Model 1 的 EM 训练示例 (2 M)

计算 $c(f e)$	重新计算 $P(f e)$
$c(a x) = \frac{1}{4}$	$P(a x) = \frac{1}{4} / (\frac{1}{4} + \frac{3}{4}) = \frac{1}{4}$
$c(b x) = \frac{3}{4}$	$P(b x) = \frac{3}{4} / (\frac{1}{4} + \frac{3}{4}) = \frac{3}{4}$
$c(a y) = \frac{3}{4} + 1 = \frac{7}{4}$	$P(a y) = \frac{7}{4} / (\frac{7}{4} + \frac{1}{4}) = \frac{7}{8}$
$c(b y) = \frac{1}{4}$	$P(b y) = \frac{1}{4} / (\frac{7}{4} + \frac{1}{4}) = \frac{1}{8}$

IBM Model 1 的 EM 训练示例 (n)



$$P(A|F, E) = 0.00 \dots 1$$

$$P(a|x) = 0.00 \dots 1$$

$$P(b|x) = 0.99 \dots 9$$

$$P(A|F, E) = 0.99 \dots 9$$

$$P(a|y) = 0.99 \dots 9$$

$$P(A|F, E) = 1$$

$$P(b|y) = 0.00 \dots 1$$

IBM Model 1 的化简(1)

- 前面 IBM Model 1 的翻译模型公式为:

$$P(F|E) = \sum_A P(F, A|E) = \frac{\epsilon}{(l+1)^m} \sum_{a_1=1}^l \dots \sum_{a_m=1}^l \prod_{j=1}^m t(f_j|e_{a_j})$$

- 其复杂度太高: $(l+1)^m$
- 这个公式实际上可以进一步简化。

因为:

$$\sum_{a_1=1}^l \dots \sum_{a_m=1}^l \prod_{j=1}^m t(f_j|e_{a_j}) = \prod_{j=1}^m \sum_{i=1}^l t(f_j|e_i)$$

IBM Model 1 的化简 (2)

- 所以翻译模型公式就可以简化为：

$$P(F|E) = \sum_A P(F, A|E) = \frac{\epsilon}{(l+1)^m} \prod_{j=1}^m \sum_{i=1}^l t(f_j|e_i)$$

- 其复杂度减少为： $l+m$
- 而 $c(f|e; F, E)$ 也可以简化为：

$$c(f|e; F, E) = \frac{t(f|e)}{t(f|e_0) + \dots + t(f|e_l)} \sum_{j=1}^m \delta(f, f_j) \sum_{i=1}^l \delta(e, e_i)$$

IBM Model 2 的推导 (1)

假设词语对齐只与源语言长度、目标语言的长度和两个词的位置有关，与其他因素无关：

$$P(a_j | a_1^{j-1}, f_1^{j-1}, m, E) = a(a_j | j, m, l)$$

归一化条件为：

$$\sum_{i=0}^l a(i | j, m, l) = 1$$

IBM Model 2 的推导 (2)

经过推导我们可以得到：

$$P(F|E) = \epsilon \sum_{a_1=0}^l \dots \sum_{a_m=0}^l \prod_{j=1}^m t(f_j | e_{a_j}) a(a_j | j, m, l)$$

经过化简我们可以得到 IBM Model 2 翻译模型：

$$P(F|E) = \epsilon \prod_{j=1}^m \sum_{i=0}^l t(f_j | e_i) a(i | j, m, l)$$

IBM Model 2 的参数求解 (1)

同样通过引入拉格朗日乘子推导可以得到:

$$t(f|e) = \lambda_e^{-1} c(f|e; F, E)$$

$$a(i|j, m, l) = \mu_{jml}^{-1} c(i|j, m, l; F, E)$$

$$c(f|e; F, E) = \sum_{j=1}^m \sum_{i=0}^l \frac{t(f|e) a(i|j, m, l) \delta(f, f_j) \delta(e, e_j)}{t(f|e_0) a(0|j, m, l) + \dots + t(f|e_l) a(l|j, m, l)}$$

$$c(i|j, m, l; F, E) = \frac{t(f_j|e_i) a(i|j, m, l)}{t(f_j|e_0) a(0|j, m, l) + \dots + t(f_j|e_l) a(l|j, m, l)}$$

IBM Model 2 的参数求解 (2)

- 考虑到训练语料库 ($\mathbf{F}|\mathbf{E}$) 是由一系列句子对组成的:

$$(F^{(1)}, E^{(1)}), (F^{(2)}, E^{(2)}), \dots, (F^{(s)}, E^{(s)})$$

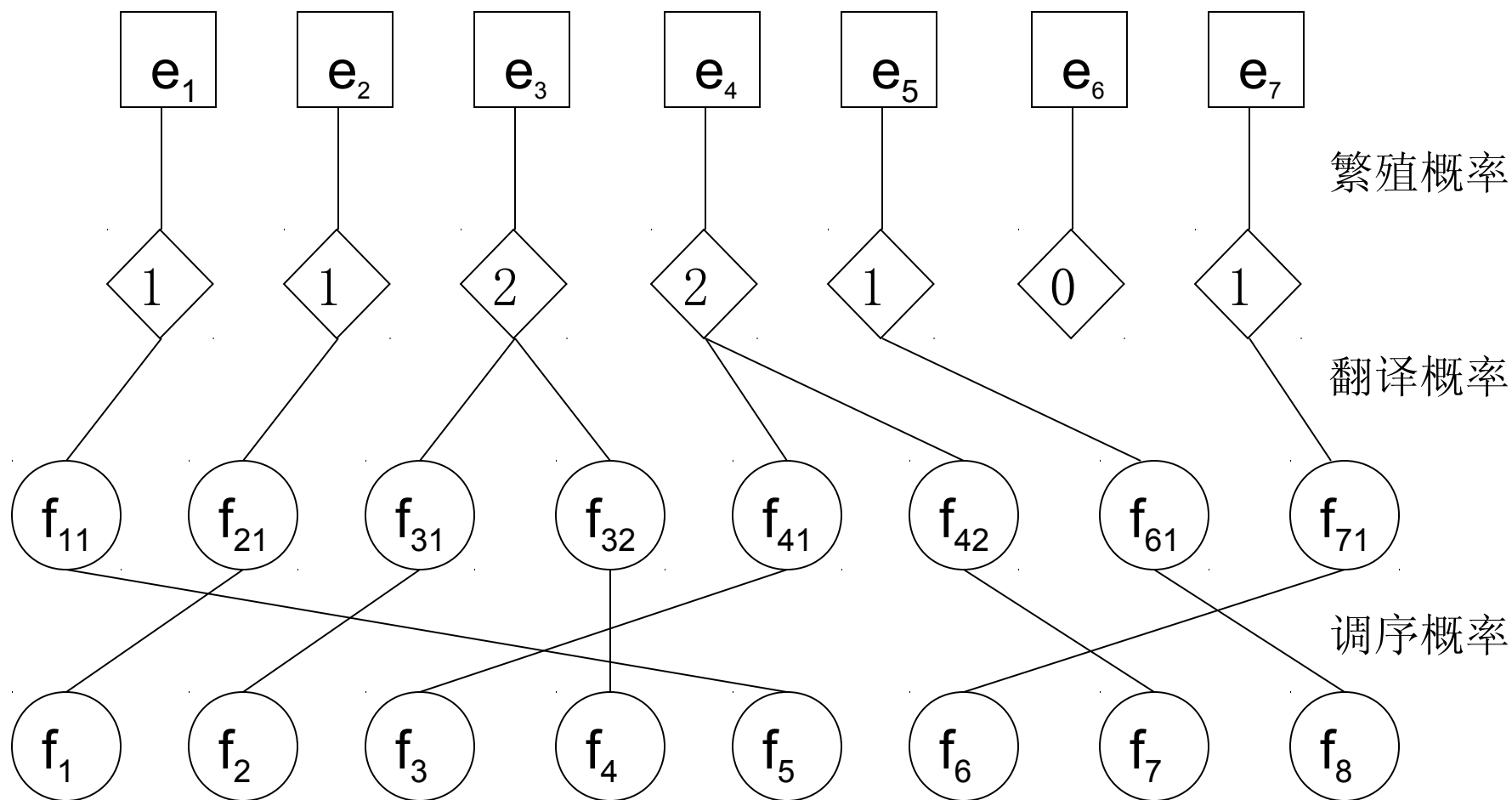
- 因此实际计算时我们采用以下公式:

$$t(f|e) = \lambda_e^{-1} \sum_s c(f|e; F^{(s)}, E^{(s)})$$

$$a(i|j, m, l) = \mu_{jml}^{-1} \sum_s c(i|j, m, l; F^{(s)}, E^{(s)})$$

- 这里 λ_e 和 μ_{jml} 仅仅起到归一化因子的作用。

IBM Model 3 & 4 & 5 推导方式 (1)



IBM Model 3 & 4 & 5 推导方式 (2)

1. 首先根据源语言词语的繁殖概率，确定每个源语言词翻译成多少个目标语言词；
2. 根据每个源语言词语的目标语言词数，将每个源语言词复制若干次；
3. 将复制后得到的每个源语言词，根据翻译概率，翻译成一个目标语言词；
4. 根据调序概率，将翻译得到的目标语言词重新调整顺序，得到目标语言句子。

IBM Model 3 的推导

- 对于句子中每一个英语单词 e ，选择一个产出率 ϕ ，其概率为 $n(\phi|e)$ ；
- 对于所有单词的产出率求和得到 **m-prime**；
- 按照下面的方式构造一个新的英语单词串：删除产出率为0的单词，复制产出率为1的单词，复制两遍产出率为2的单词，依此类推；
- 在这 **m-prime** 个单词的每一个后面，决定是否插入一个空单词 **NULL**，插入和不插入的概率分别为 p_1 和 p_0 ；
- ϕ_0 为插入的空单词 **NULL** 的个数。
- 设 m 为目前的总单词数： $m\text{-prime} + \phi_0$ ；
- 根据概率表 $t(f|e)$ ，将每一个单词 e 替换为外文单词 f ；
- 对于不是由空单词 **NULL** 产生的每一个外语单词，根据概率表 $d(j|i,l,m)$ ，赋予一个位置。这里 j 是法语单词在法语串中的位置， i 是产生当前这个法语单词的对应英语单词在英语句子中的位置， l 是英语串的长度， m 是法语串的长度；
- 如果任何一个目标语言位置被多重登录（含有一个以上单词），则返回失败；
- 给空单词 **NULL** 产生的单词赋予一个目标语言位置。这些位置必须是空位置（没有被占用）。任何一个赋值都被认为是等概率的，概率值为 $1/\phi_0$ 。
- 最后，读出法语串，其概率为上述每一步概率的乘积。

IBM 模型的参数训练(1): EM 算法

- EM 参数训练算法是经典的无指导学习的算法:
 1. 给定初始参数;
 2. E 步骤: 用已有的参数计算每一个句子对的所有可能的对齐的概率;
 3. M 步骤: 用得到的所有对齐的概率重新计算参数;
 4. 重复执行 E 步骤和 M 步骤, 直到收敛。
- 由于 EM 算法的 E 步骤需要穷尽所有可能的对齐, 通常这会带来极大的计算量, 除非我们可以对计算公式进行化简 (就像前面 IBM Model 1 所做的那样), 否则这种计算量通常是不可承受的。

IBM 模型的参数训练(2): Viterbi 训练

- **Viterbi 参数训练算法:**
 1. 给定初始参数;
 2. 用已有的参数求概率最大 (**Viterbi**) 的词语对齐;
 3. 用得到的概率最大的词语对齐重新计算参数;
 4. 回到第二步, 直到收敛为止。
- 在对参数计算公式无法化简的情况下, 采用 **Viterbi** 参数训练算法是一种可行的做法, 这种算法通常可以迅速收敛到一个可以接受的结果。

IBM 模型的参数训练 (3)

- IBM Model 1
 - 任何初始值均可达到全局最优
- IBM Model 2~5:
 - 存在大量局部最优，任意给定的初值很容易导致局部最优，而无法到达全局最优的结果
 - IBM 的训练策略：
 - 依次训练 IBM Model 1-5
 - 对于与上一级模型相同的参数初始值，直接取上一个模型训练的结果；
 - 对于新增加的参数，取任意初始值。

IBM 模型的参数训练 (4)

- 由于 IBM Model 1 和 2 存在简化的迭代公式，实际上在 EM 算法迭代是并不用真的去计算所有的对齐，而是可以利用迭代公式直接计算下一次的参数；
- 由于 IBM Model 3、4、5 的翻译模型公式无法化简，理论上应该进行 EM 迭代。由于实际上由于计算所有词语对齐的代价太大，通常采用 Viterbi 训练，每次 E 步骤只生成最好的一个或者若干个对齐。

内容提要

- 为翻译建立概率模型
- IBM 的信源信道模型
- 语言模型 – – n 元语法模型
- 翻译模型 – – IBM 模型1-5
- 词语对齐算法
- 解码算法
- Candide 系统
- Egypt 工具包与 Giza++
- 机器翻译自动评价

词语对齐算法

- 基于 IBM 模型的柱搜索（Beam Search）
词语对齐方法

内容提要

- 为翻译建立概率模型
- IBM 的信源信道模型
- 语言模型 – – n 元语法模型
- 翻译模型 – – IBM 模型1-5
- 词语对齐算法
- 解码算法
- Candide 系统
- Egypt 工具包与 Giza++
- 机器翻译自动评价

统计机器翻译的解码

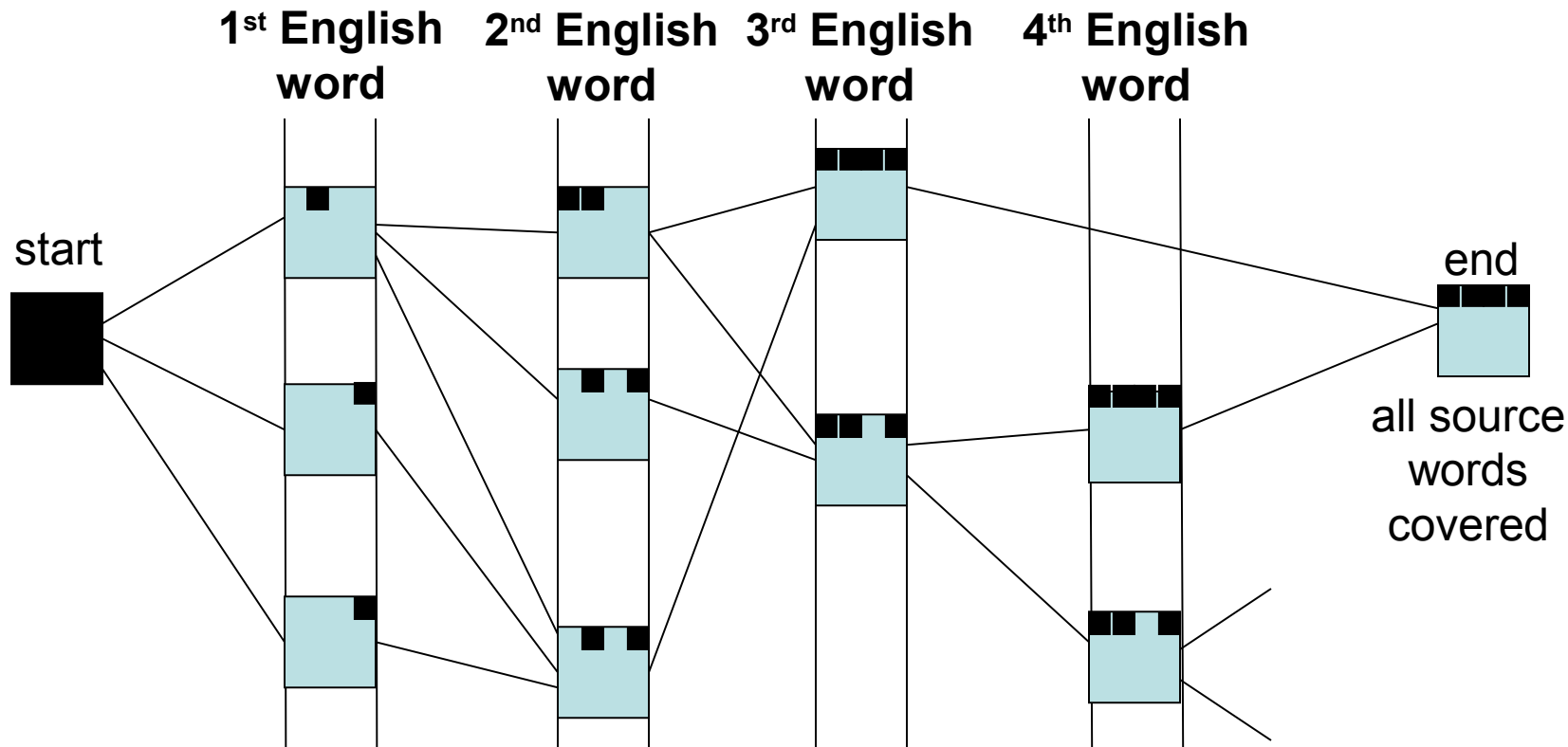
- 给定 F ，求 E ，使得 $P(E)*P(F|E)$ 最大
- 解码问题实际上是一个搜索问题，搜索空间巨大，不能保证总能找到全局最优，但通常一些局部最优也是可以接受的
- 如果考虑所有的词语对齐可能性，那么这个问题是一个 **NP 完全问题** [Knight 99]
- 经典的算法：
 - 单调解码（不调整词序）
 - 非单调解码
 - 贪婪算法
 - 堆栈搜索

单调解码

贪婪解码

堆栈搜索解码算法 (1)

[Brown et al US Patent #5,477,451]



Each partial translation hypothesis contains:

- Last English word chosen + source words covered by it
- Next-to-last English word chosen
- Entire coverage vector (so far) of source sentence ■ ■ ■
- Language model and translation model scores (so far)

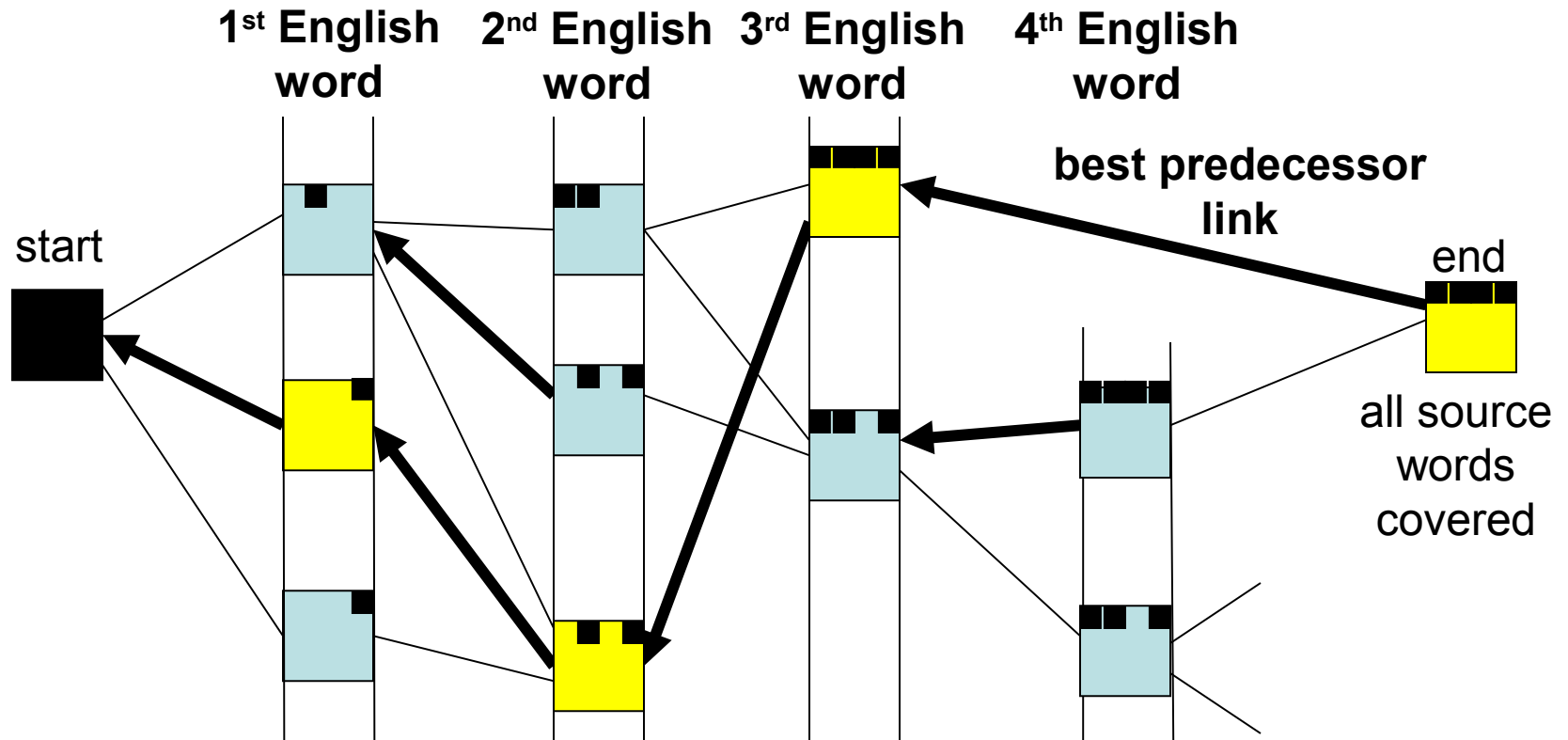
[Jelinek 69;
Och, Ueffing, and Ney, 01]

器翻译方法

59

堆栈搜索解码算法 (2)

[Brown et al US Patent #5,477,451]



Each partial translation hypothesis contains:

- Last English word chosen + source words covered by it
- Next-to-last English word chosen
- Entire coverage vector (so far) of source sentence ■ ■ ■
- Language model and translation model scores (so far)

[Jelinek 69;
Och, Ueffing, and Ney, 01]

器翻译方法

60

堆栈搜索解码算法 - 例子(1/13)

- 待翻译句子:
- 翻译概率表:

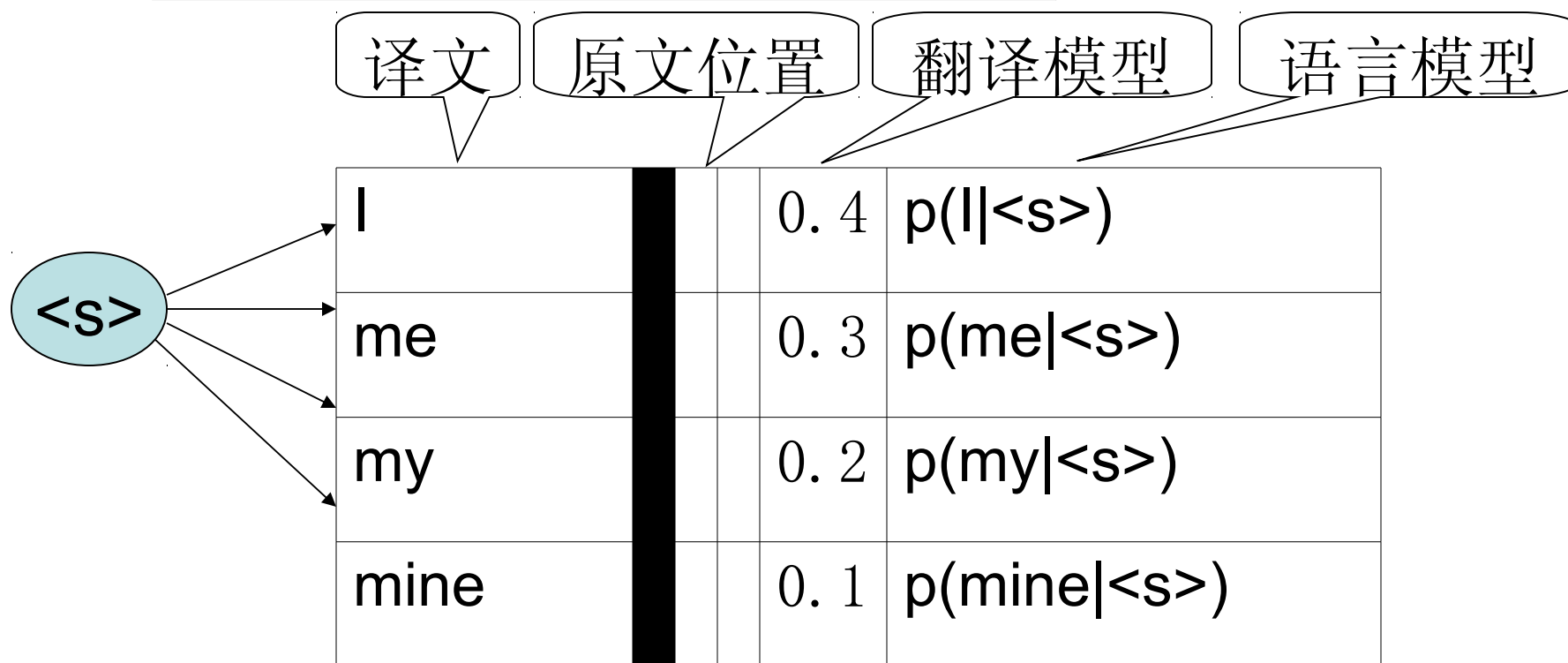
我	的	书
---	---	---

我	I	0.4	me	0.3	my	0.2	mine	0.1
的	of	0.5	↪			0.5		
书	book	0.5	the book			0.5		

- 不考虑扭曲概率 (IBM 模型1)
- 语言模型 (略) Beam Width=3
- 假设 the book 是一个词

堆栈搜索解码算法 - 例子(2/13)

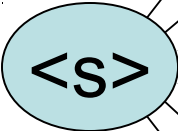
第一个译文词由“我”翻译过来，
得到四个翻译假设(hypothesis)



译文	原文位置	翻译模型	语言模型
I		0.4	$p(I <s>)$
me		0.3	$p(me <s>)$
my		0.2	$p(my <s>)$
mine		0.1	$p(mine <s>)$

堆栈搜索解码算法 - 例子 (3/13)

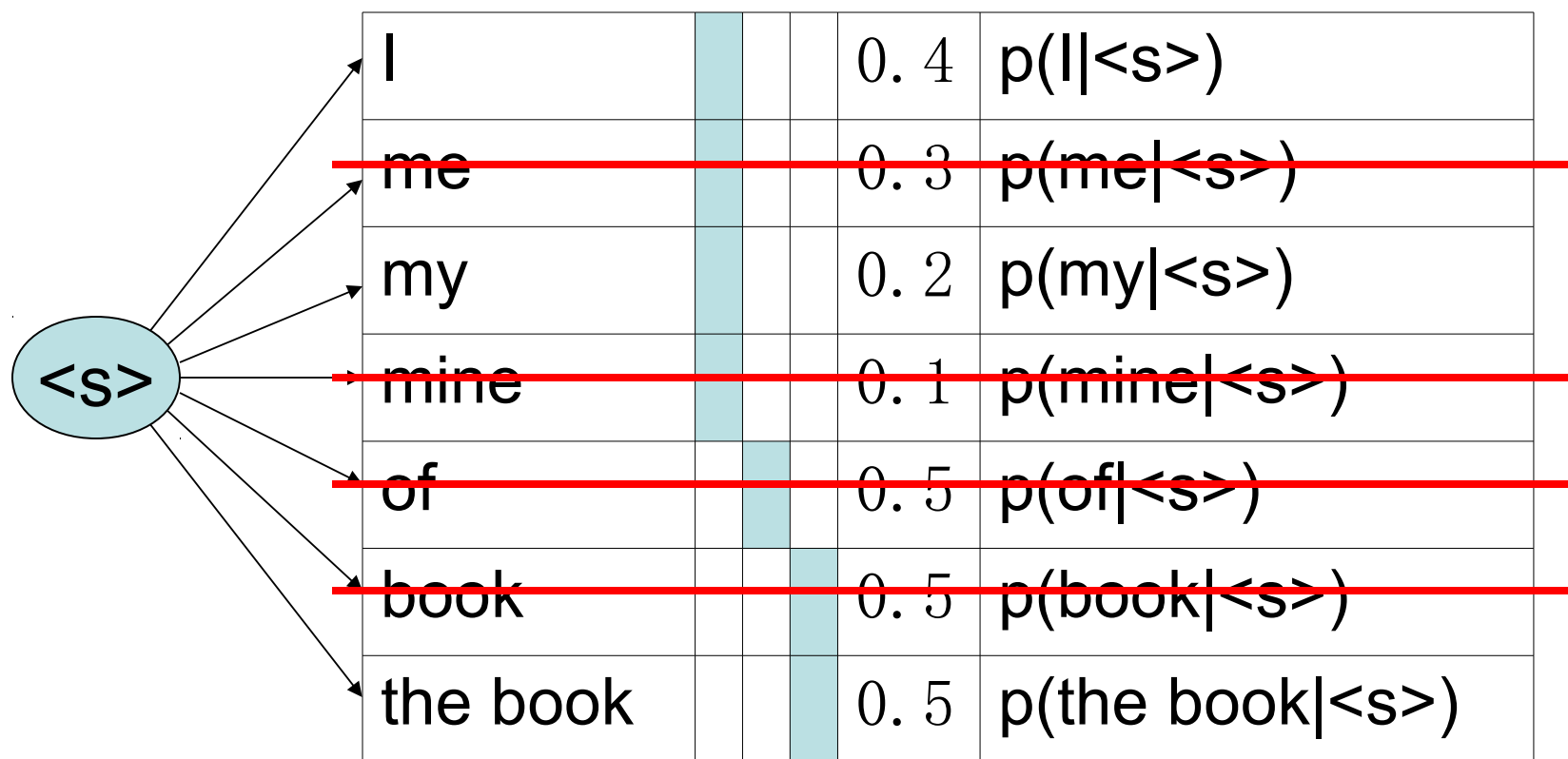
加上其他翻译假设 (hypothesis)



I				0.4	$p(I <s>)$
me				0.3	$p(me <s>)$
my				0.2	$p(my <s>)$
mine				0.1	$p(mine <s>)$
of				0.5	$p(of <s>)$
book				0.5	$p(book <s>)$
the book				0.5	$p(the\ book <s>)$

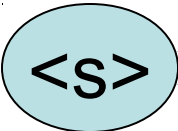
堆栈搜索解码算法 - 例子(4/13)

剪枝 (prune)



堆栈搜索解码算法 - 例子 (5/13)

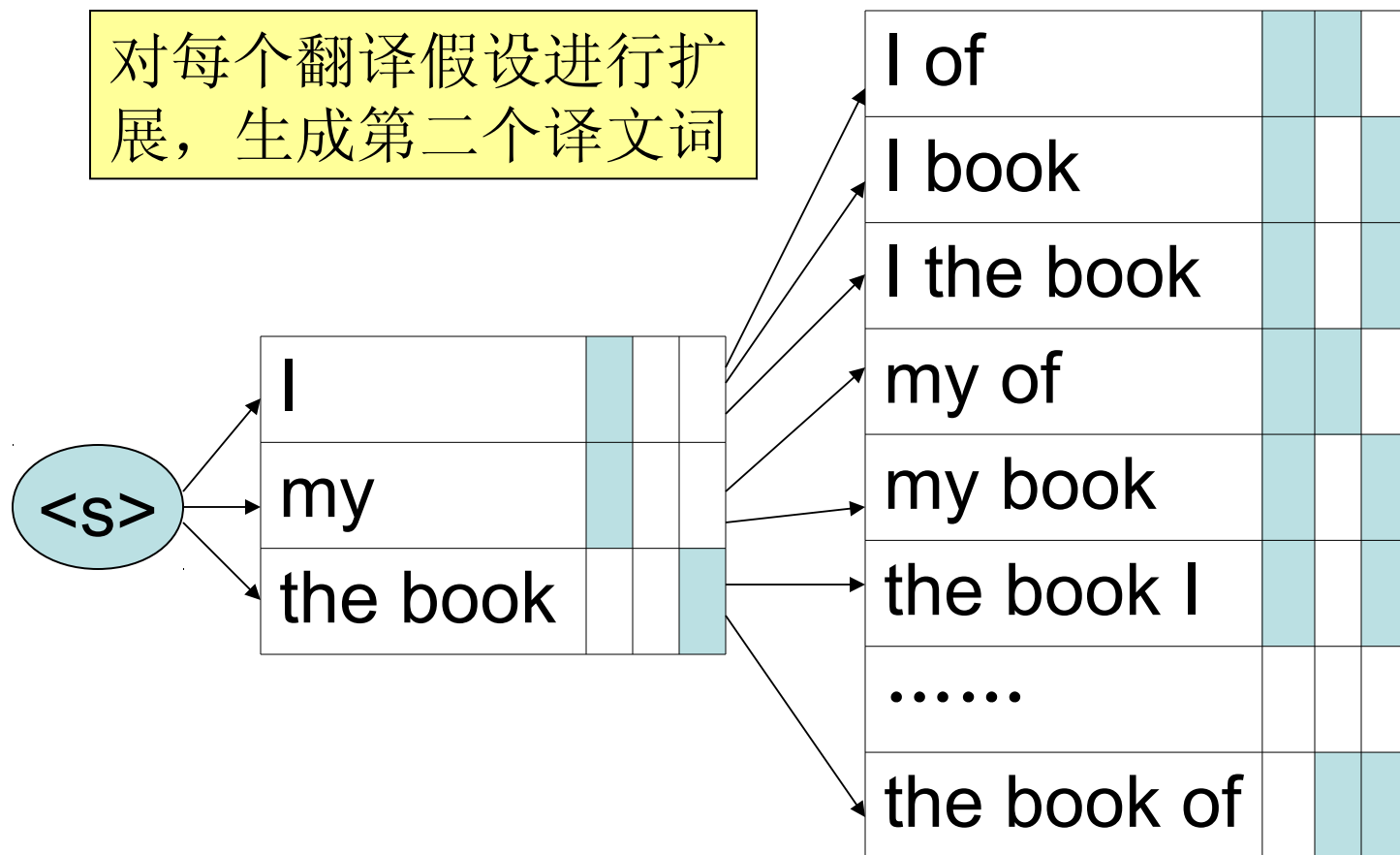
剪枝 (prune) 后的结果



I				0.4	$p(I <s>)$
my				0.2	$p(my <s>)$
the book				0.5	$p(the\ book <s>)$

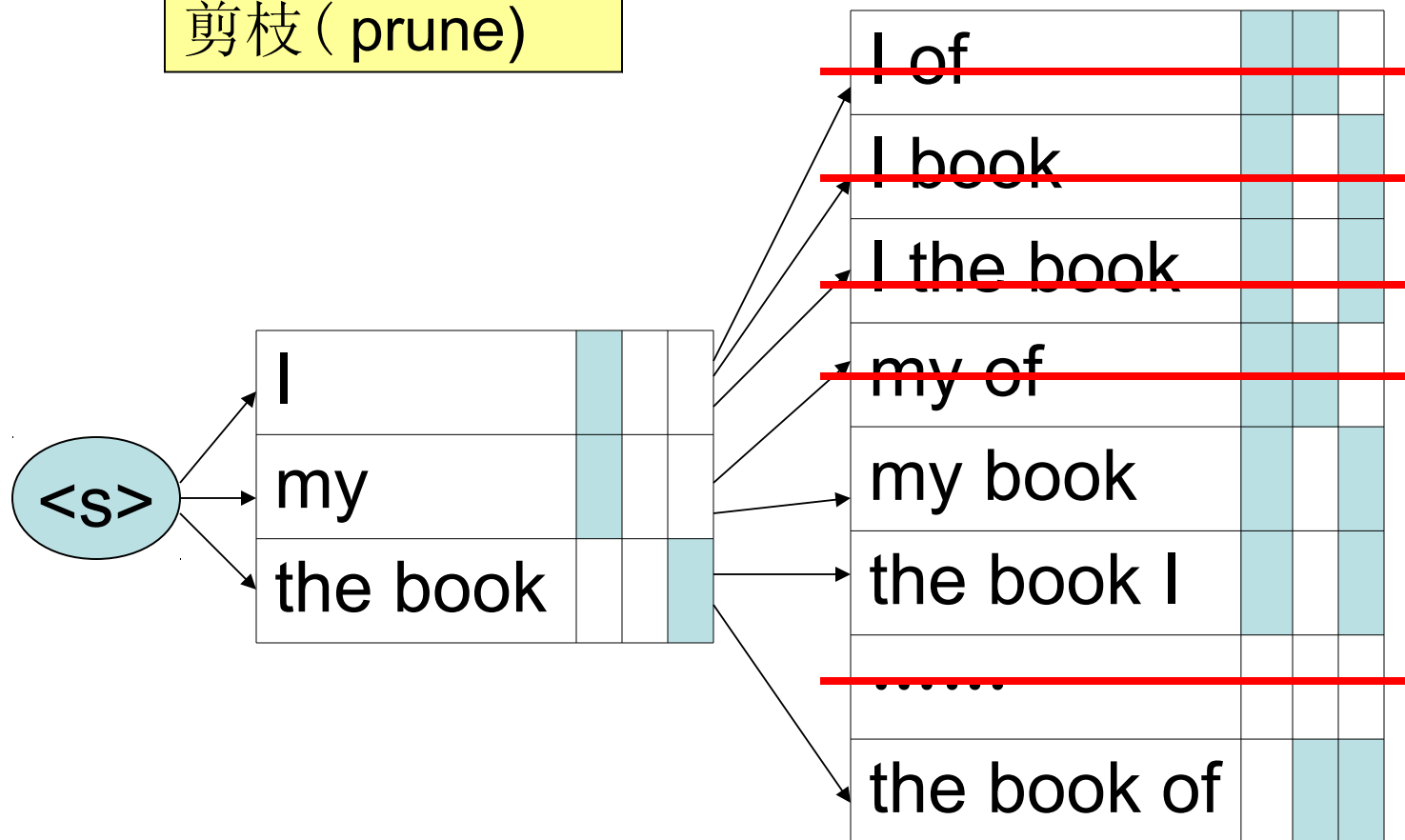
堆栈搜索解码算法 - 例子(6/13)

对每个翻译假设进行扩展，生成第二个译文词



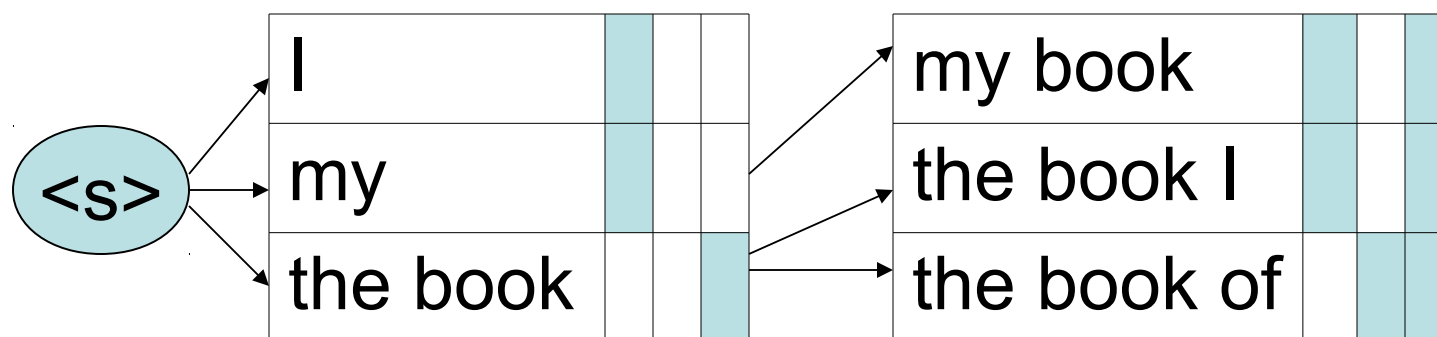
堆栈搜索解码算法 - 例子(7/13)

剪枝 (prune)



堆栈搜索解码算法 - 例子(8/13)

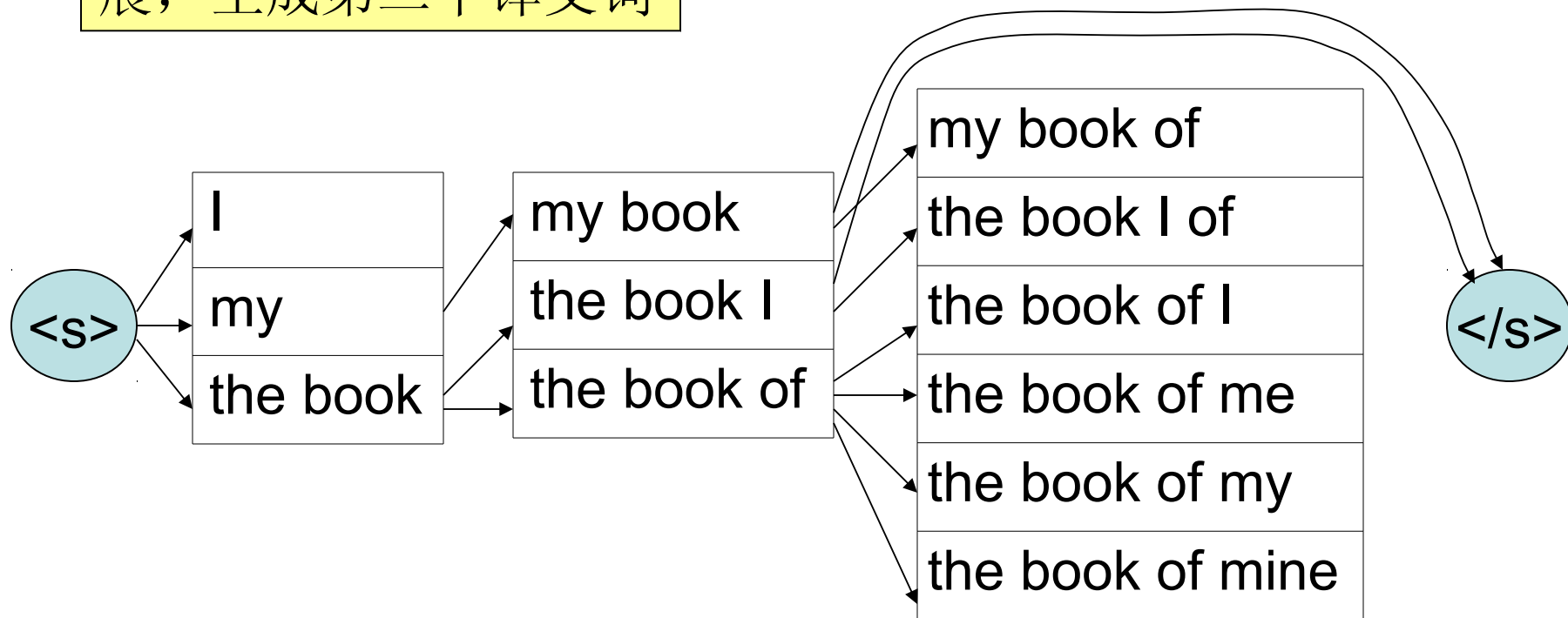
剪枝 (prune) 后的结果



堆栈搜索解码算法 - 例子 (9/13)

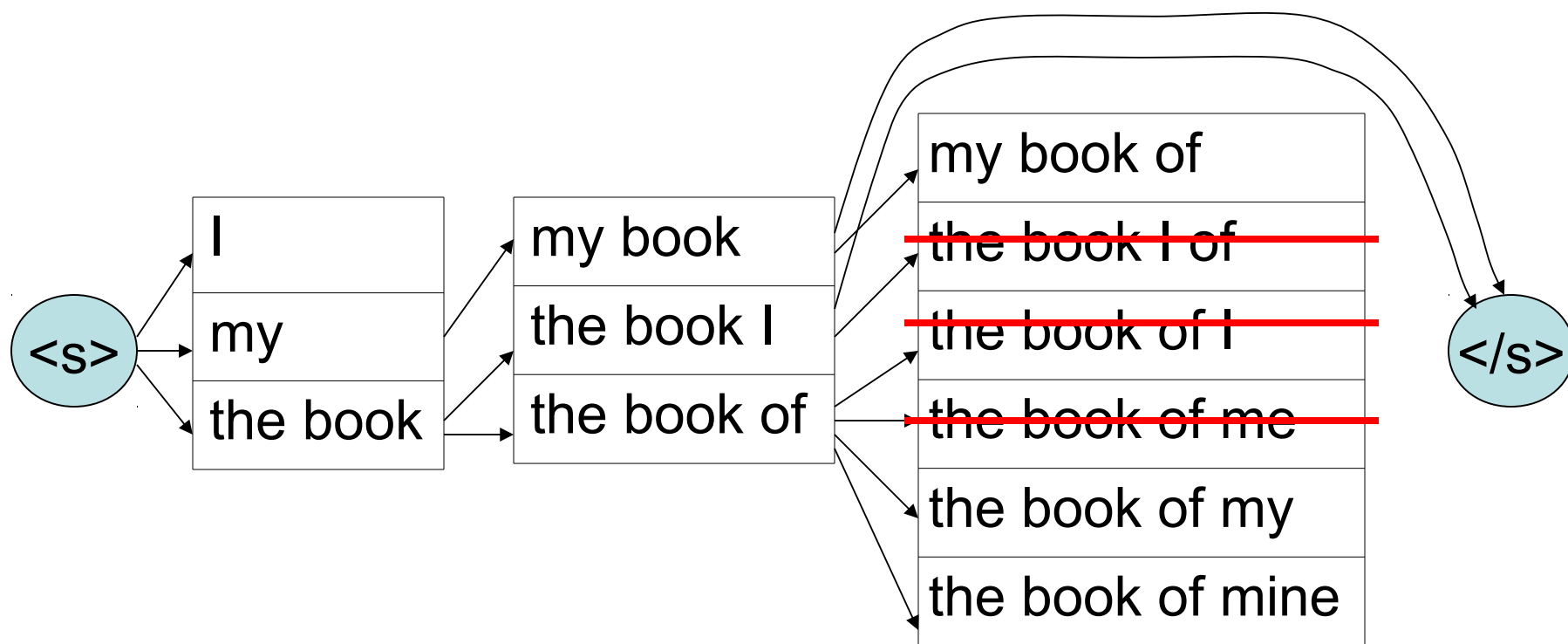
对每个翻译假设进行扩展，生成第二个译文词

如果剩下的词都可以翻译到空，
该假设可以不再扩展



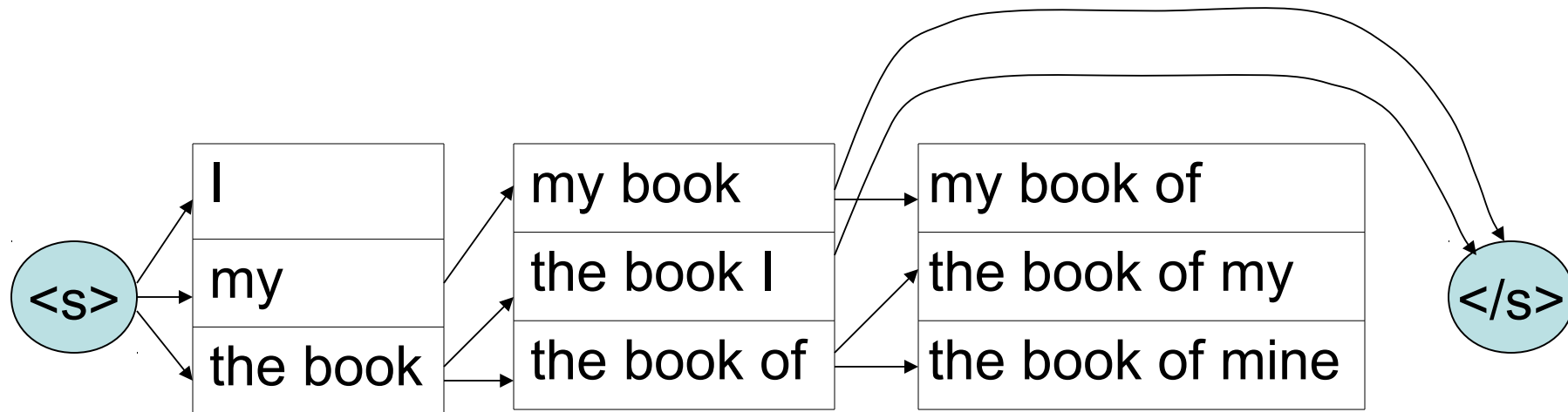
堆栈搜索解码算法 - 例子(10/13)

剪枝 (prune)



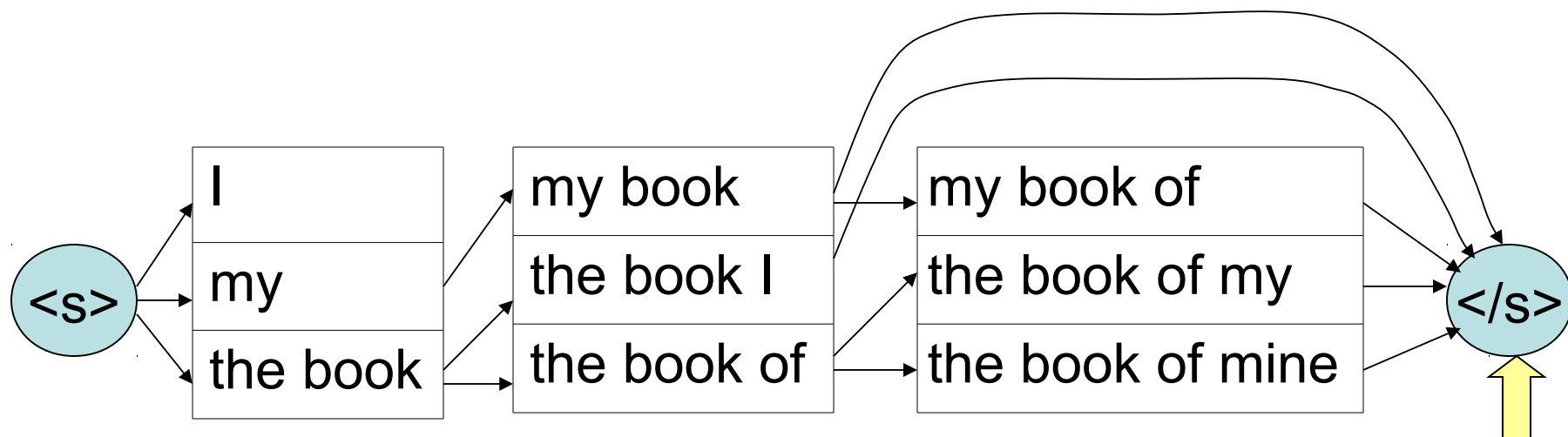
堆栈搜索解码算法 - 例子(11/13)

剪枝 (prune) 后的结果



堆栈搜索解码算法 - 例子(12/13)

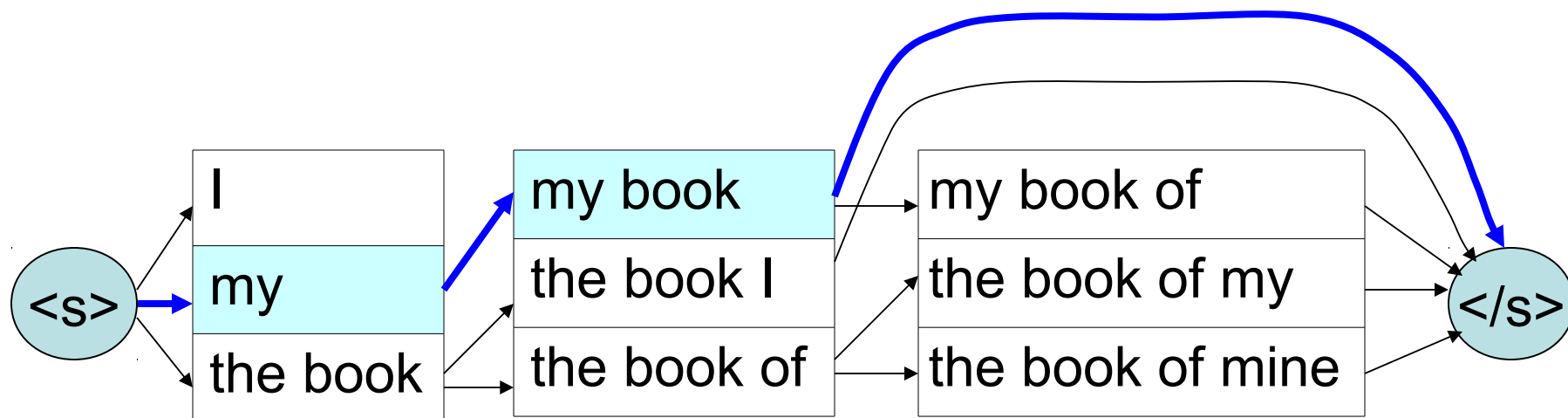
所有源文词都已经翻译，不再对假设进行扩展



在这里加上句尾标记</ s>，重新计算完整句子的语言模型分数，并选择最优译文。

堆栈搜索解码算法 - 例子(13/13)

最优译文



内容提要

- 为翻译建立概率模型
- IBM 的信源信道模型
- 语言模型 – – n 元语法模型
- 翻译模型 – – IBM 模型1-5
- 词语对齐算法
- 解码算法
- **Candide 系统**
- **Egypt 工具包与 Giza++**
- 机器翻译自动评价

IBM 公司的 Candide 系统(1)

- 基于统计的机器翻译方法
- 分析—转换—生成
 - 中间表示是线性的
 - 分析和生成都是可逆的
- 分析（预处理）：
 1. 短语切分
 2. 专名与数词检测
 3. 大小写与拼写校正
 4. 形态分析
 5. 语言的归一化

IBM 公司的 Candide 系统(2)

- 转换（解码）：基于统计的机器翻译
- 解码分为两个阶段：
 - 第一阶段：使用粗糙模型的堆栈搜索
 - 输出140个评分最高的译文
 - 语言模型：三元语法
 - 翻译模型：EM Trained IBM Model 5
 - 第二阶段：使用精细模型的扰动搜索
 - 对第一阶段的输出结果先扩充，再重新评分
 - 语言模型：链语法
 - 翻译模型：最大熵翻译模型（选择译文词）

IBM 公司的 Candide 系统 (3)

- ARPA 的测试结果 :

	Fluency		Adequacy		Time Ratio	
	1992	1993	1992	1993	1992	1993
Systran	. 466	. 540	. 686	. 743		
Candide	. 511	. 580	. 575	. 670		
Transman	. 819	. 838	. 837	. 850	. 688	. 625
Manual		. 833		. 840		

内容提要

- 为翻译建立概率模型
- IBM 的信源信道模型
- 语言模型 – – n 元语法模型
- 翻译模型 – – IBM 模型1-5
- 词语对齐算法
- 解码算法
- Candide 系统
- Egypt 工具包与 Giza++
- 机器翻译自动评价

JHU 的1999年夏季研讨班

- 由来
 - IBM 的实验引起了广泛的兴趣
 - IBM 的实验很难重复：工作量太大
- 目的
 - 构造一个统计机器翻译工具（EGYPT）并使它对于研究者来说是可用的（免费传播）；
 - 在研讨班上用这个工具集构造一个捷克语—英语的机器翻译系统；
 - 进行基准评价：主观和客观；
 - 通过使用形态和句法转录机改进基准测试的结果；
 - 在研讨班最后，在一天之内构造一个新语对的翻译器。
- JHU 夏季研讨班大大促进了统计机器翻译的研究

EGYPT 工具包

- EGYPT 的模块
 1. **GIZA**: 这个模块用于从双语语料库中抽取统计知识（参数训练）
 2. **Decoder**: 解码器，用于执行具体的翻译过程（在信源信道模型中，“翻译”就是“解码”）
 3. **Cairo**: 整个翻译系统的可视化界面，用于管理所有的参数、查看双语语料库对齐的过程和翻译模型的解码过程
 4. **Whittle**: 语料库预处理工具
- EGYPT 可在网上免费下载，成为 SMT 的基准

EGYPT 工具包的性能

“当解码器的原形系统在研讨班上完成时，我们很高兴并惊异于其速度和性能。1990年代早期在 IBM 公司举行的 DARPA 机器翻译评价时，我们曾经预计只有很短（10个词左右）的句子才可以用统计方法进行解码，即使那样，每个句子的解码时间也可能是几个小时。在早期 IBM 的工作过去将近10年后，摩尔定律、更好的编译器以及更加充足的内存和硬盘空间帮助我们构造了一个能够在几秒钟之内对25个单词的句子进行解码的系统。为了确保成功，我们在搜索中使用了相当严格的阈值和约束，如下所述。但是，解码器相当有效这个事实为这个方向未来的工作预示了很好的前景，并肯定了 IBM 的工作的初衷，即强调概率模型比效率更重要。”

——引自 JHU 统计机器翻译研讨班的技术报告

内容提要

- 为翻译建立概率模型
- IBM 的信源信道模型
- 语言模型 – – n 元语法模型
- 翻译模型 – – IBM 模型1-5
- 词语对齐算法
- 解码算法
- Candide 系统
- Egypt 工具包与 Giza++
- 机器翻译自动评价

机器翻译的评价

- 常见的人工评价指标
 - 忠实度和流利度
 - 可理解率
 -
- 自动评价的重要意义
 - 反复使用无需成本
 - 为通过频繁的实验提高系统性能提供了基本的保证

基于测试点的机器翻译自动评价

- 北大俞士汶于1990年代初期提出
- 模仿人类的标准化考试的方法，对每个题目（源文句子）设置若干个测试点
- 每个测试点只考察一个问题（比如汉语分词、词语译文选择等）
- 判断测试点是否被正确翻译，完全通过字符串匹配，每个测试点可以有多种候选的正确答案
- 是国际上最早出现的机器翻译自动评价方法之一
- 缺点是题库的构造成本很高，需要对机器翻译有相当了解的专家

基于编辑距离的机器翻译自动评价

- 编辑距离: **Edit distance**, 又称 **Levenshtein Distance**, 用于计算两个字符串之间的距离
- 编辑距离的含义, 是指通过插入、删除、替换等编辑操作, 将一个字符串变成另外一个字符串时, 所需要的编辑操作的次数
- 常见的基于编辑距离的评价指标:
 - WER, PER, mWER, mPER
- 缺点: 对词序问题没有好的处理方法

基于 N 元语法的机器翻译自动评价

- 基本思想
 - 对于每个源文句子，由多位翻译人员提供人工翻译的结果
 - 将机器翻译的结果与这多个人工翻译的结果进行比较，越相似的句子，评价越高
 - 这种比较按照一元语法、二元语法、三元语法、……分别进行，然后进行评价
- 常见的评价指标
 - **BLEU**: 各层语法的结果进行几何平均
 - **NIST**: 各层语法的结果进行算术平均，同时考虑信息增益

机器翻译自动评价：例子

考虑例子：

Candidate 1: It is a guide to action which ensures that the military always obeys the command of the party

Candidate 2: It is to insure the troops forever hearing the activity guidebook that party direct

Reference 1: It is a guide to action that ensures that the military will forever heed party commands

Reference 2: It is the guiding principle which guarantees the military forces always being under the command of the party

Reference 3: It is the practical guide for the army to heed the directions of the party

候选译文更好？

简单计数方法

- 计算候选译文中 **n-gram** 在参考译文中出现的次数，除以该候选译文 **n-gram** 总数，也就是 **n-gram** 的正确率。
- 举例来说，对于 **unigram**，也就是计算候选译文中的所有词语中，在参考译文中出现的词语数，除以候选译文的词语总数，就是 **unigram** 的正确率。

改进的计数方法 (1)

考虑例子：

- Candidate: *the the the the the the the*
- Reference 1: *the cat is on the mat.*
- Reference 2: *there is a cat on the mat.*

在这个例子中，如果采用简单计数方法，候选译文的 **unigram** 正确率将是100%，显然是不合理的

改进的计数方法：对于同一个词的多次出现，其匹配次数最多只能等于在同一个参考译文中该词出现最多的次数

跟进改进的计数方法，这个例子中 **the** 在参考译文的同一个句子中最多出现两次，因此候选译文的 **unigram** 正确率只有2/7。

改进的计数方法 (2)

再看前面的例子：

Candidate 1: It is a guide to action which ensures that the military always obeys the command of the party

Candidate 2: It is to insure the troops forever hearing the activity guidebook that party direct

Reference 1: It is a guide to action that ensures that the military will forever heed party commands

Reference 2: It is the guiding principle which guarantees the military forces always being under the command of the party

Reference 3: It is the practical guide for the army to heed the directions of the party

■Candidate 1 的 Bigram 正确率达到了10/17

■而 Candidate 2 的 Bigram 正确率只有 1/13.

与忠实度和流利度的关系

- 修改后的 **n-gram** 正确率可以认为在一定程度上反映了译文质量的两个方面：
 - 一元语法 **unigram** 正确率主要体现了词语是否翻译正确，这反映了译文忠实度
 - 二元语法 **bigram** 以上的 **ngram** 正确率主要体现了词语的排列顺序是否正确，这反映了译文的流利度

N-Gram 正确率的计算公式

$$P_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n-gram \in C} Count_{clip}(n-gram)}{\sum_{C' \in \{Candidates\}} \sum_{n-gram' \in C'} Count(n-gram')}$$

$\{Candidates\}$ 是所有测试句子的机器翻译结果

$Count(n-gram)$ 是 $n-gram$ 在某个候选译文中的出现次数

$Count_{clip}(n-gram)$ 是 $n-gram$ 在某个候选译文中出现的次数，按照同一个参考译文中出现最多次数剪切后的次数

新问题：召回率问题（1）

考虑这个例子：

Candidate1: of the

Reference1: It is a guide to action that ensures that the military will forever heed party commands

Reference2: It is the guiding principle which guarantees the military forces always being under the command of the party

Reference3: It is the practical guide for the army to heed the directions of the party

- 按照上面公式计算的 bigram 正确率将是100%，不合理
- 是召回率太低引起的吗？

新问题：召回率问题 (2)

再考虑一个例子：

Candidate1: I always invariably perpetually do.

Candidate2: I always do.

Reference1: I always do.

Reference2: I invariably do.

Reference3: I perpetually do.

- Candidate 1 的召回率高于 Candidate 2，但实际上译文质量不如 Candidate 2
- 解决办法：不考虑召回率，而是简单地对太短的译文进行惩罚

BLEU Metric

$$BLEU = BP \times \exp\left(\sum_{n=1}^N w_n \log p_n\right)$$

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1 - \frac{r}{c})} & \text{if } c \leq r \end{cases}$$

$$\log BLEU = \min\left(1 - \frac{r}{c}, 0\right) + \sum_{n=1}^N w_n \log p_n$$

$$N = 4, W_n = \frac{1}{N}$$

BLEU: An Example

此投影片引自微软亚洲研究院周明博士的一个报告，特此声明并感谢！

• **Candidate 1: the book is on the desk**

• **Reference 1: there is a book on the desk**

• **Reference 2: the book is on the table**

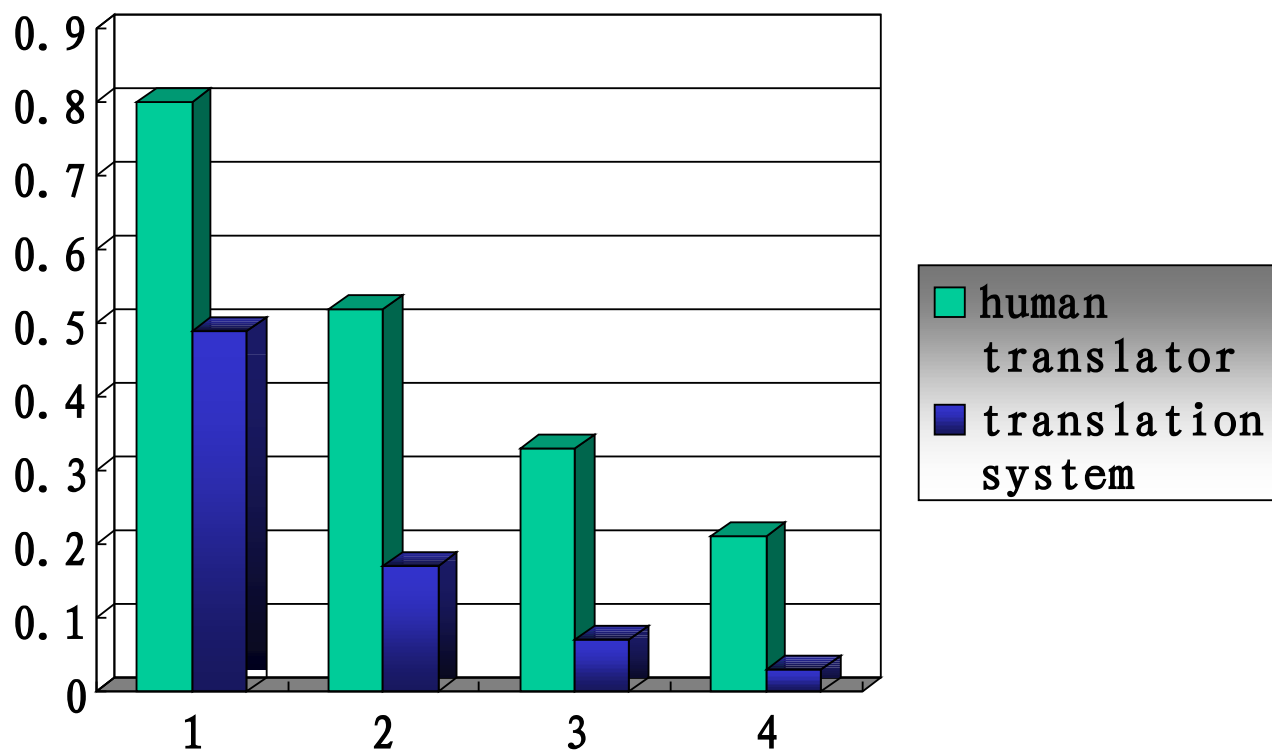
unigram:	bigram:	trigram:
$Count_{clip}(the)=2$	$Count_{clip}(the,book)=1$	$Count_{clip}(the, book,is)=1$
$Count_{clip}(book)=1$	$Count_{clip}(book,is)=1$	$Count_{clip}(book,is,on)=1$
$Count_{clip}(is)=1$	$Count_{clip}(is,on)=1$	$Count_{clip}(is,on,the)=1$
$Count_{clip}(on)=1$	$Count_{clip}(on, the)=1$	$Count_{clip}(on,the,desk)=1$
$Count_{clip}(desk)=1$	$Count_{clip}(the, desk)=1$	
$\Sigma Count(unigram)=6$	$\Sigma Count(bigram)=5$	$\Sigma Count(trigram)=4$
$p_1=1$	$p_2=1$	$p_3=1$

$$\left. \begin{matrix} c=6 \\ r=6 \end{matrix} \right\} = e^{1-\frac{r}{c}} = e^0 = 1 = BP$$

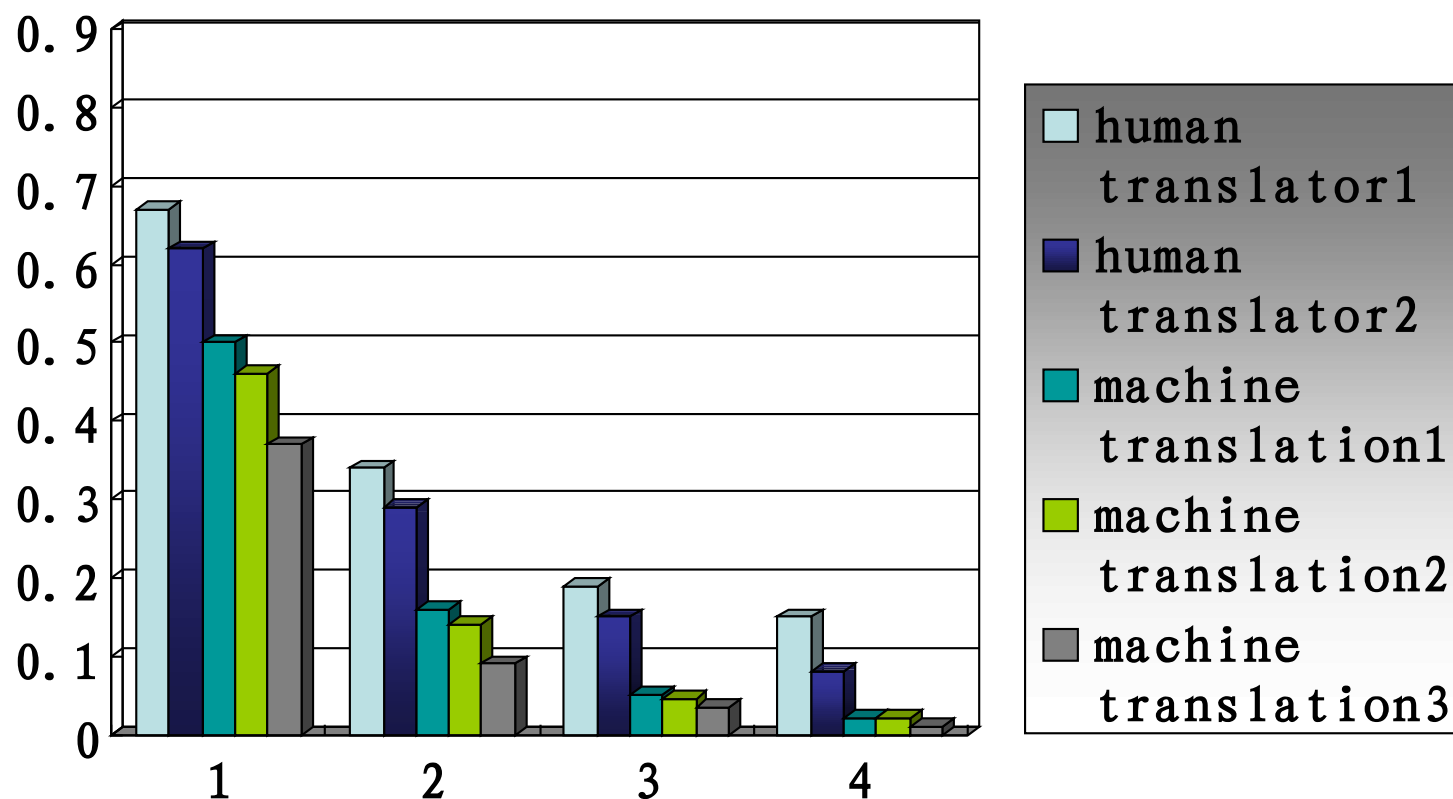
$$BLEU = BP \times \exp\left(\sum_{n=1}^N w_n \log p_n\right)$$

$$= \exp\left(\frac{1}{3}(\log 1 + \log 1 + \log 1)\right) = \sqrt[3]{1 * 1 * 1} = 1$$

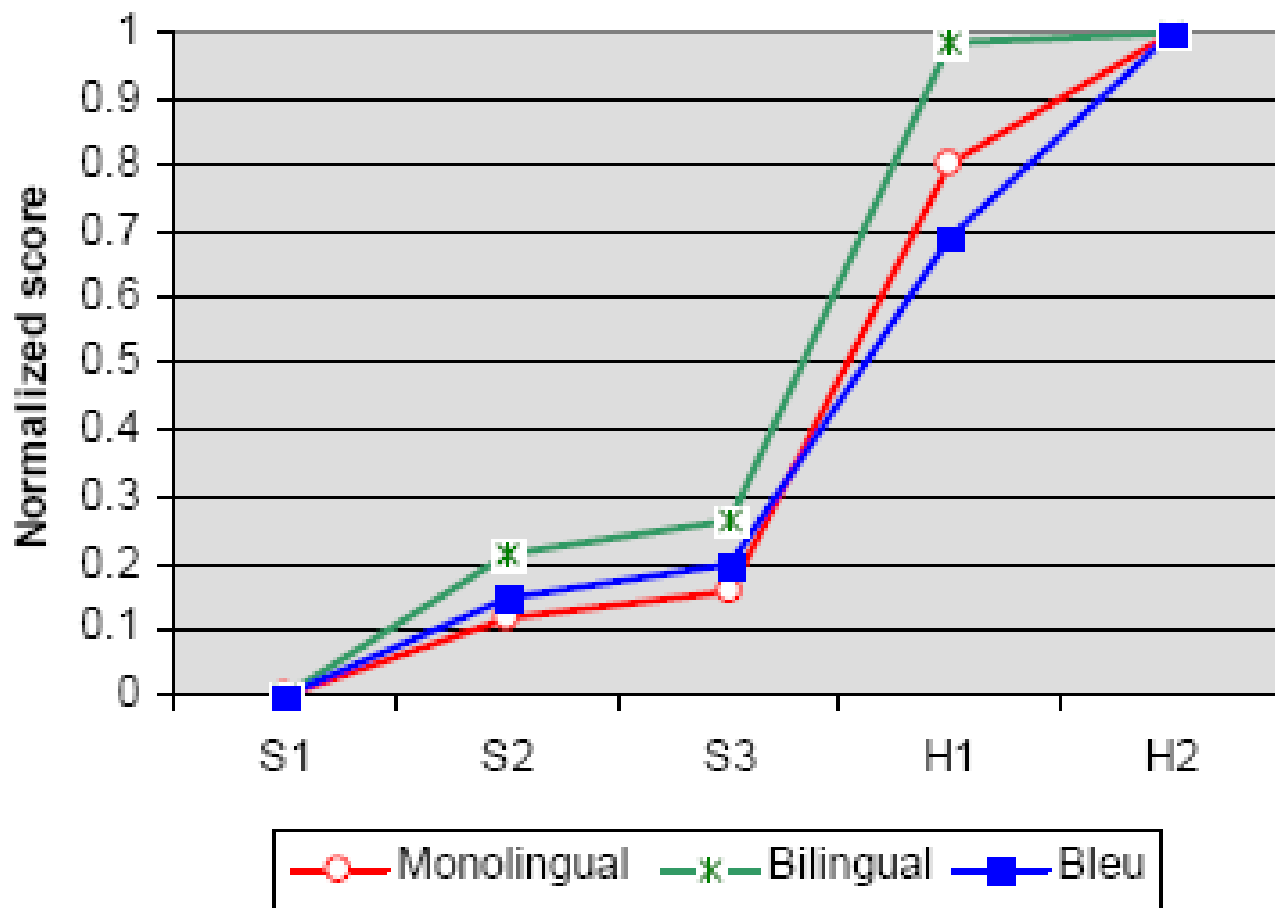
实验：人工译文和机器译文的比较



实验：多个人工译文和机器译文的比较



BLEU 与人工评测的一致性



N 元语法方法的优缺点

- 优点
 - 构造成本比较低，只需要普通翻译人员即可
 - 在译文质量普遍不高的情况下，与人工评价的结果具有较好的拟合度
- 缺点
 - 在译文质量较好的情况下，与人工评价的结果拟合度较差，因而这种方法不宜用于评价人工翻译的结果
- 就目前而言，用于进行机器翻译的系统评价还是适合的，而且起到了非常好的作用

讨论

- 在实际的双语语料库上运行 **Giza++** 进行句子对齐，并获得统计翻译模型参数
- 利用北京大学的切分标注语料库和 **SRI** 语言模型工具实现一个汉语语言模型
- 利用上述翻译模型和语言模型尝试编写一个基于词的解码算法，可以采用贪心算法或者堆栈搜索算法