



机器翻译原理与方法

第六讲 基于短语的机器翻译方法

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

- 对数线性模型
- 最小错误率训练
- 基于对数线性模型的词语对齐
- 基于短语的翻译模型
- 短语模型的解码算法
- 开源机器翻译系统简介
- 总结

统计机器翻译的对数线性模型(1)

- Och 于 ACL2002 提出，思想来源于 Papineni 提出的基于特征的自然语言理解方法，该论文获得 ACL2002 的最佳论文称号
- 不使用信源信道思想，而是采用多特征思想
- 信源信道模型是一种生成模型，而对数线性模型是一种判别模型
- 是一个比信源信道模型更具一般性的模型，信源信道模型是其一个特例
- 原始论文的提法是“最大熵”模型，现在通常使用“对数线性（Log-Linear）模型”这个概念。“对数线性模型”的含义比“最大熵模型”更宽泛，而且现在这个模型通常都不再使用最大熵的方法进行参数训练，因此“对数线性”模型的提法更为准确。
- 与 NLP 中通常使用的最大熵方法的区别：使用连续量（实数）作为特征，而不是使用离散的布尔量（只取0和1值）作为特征

统计机器翻译的对数线性模型 (2)

假设 e 、 f 是机器翻译的目标语言和源语言句子， $h_1(e, f), \dots, h_M(e, f)$ 分别是 e 、 f 上的 M 个特征， $\lambda_1, \dots, \lambda_M$ 是与这些特征分别对应的 M 个参数，那么直接翻译概率可以用以下公式模拟：

$$\begin{aligned} P(e|f) &= p_{\lambda_1 \dots \lambda_M}(e|f) \\ &= \frac{\exp \sum_{m=1}^M \lambda_m h_m(e, f)}{\sum_{e'} \exp \sum_{m=1}^M \lambda_m h_m(e', f)} \end{aligned}$$

统计机器翻译的对数线性模型 (3)

对于给定的 f , 其最佳译文 e 可以用以下公式表示:

$$\begin{aligned}\hat{e} &= \operatorname{argmax}_e P(e|f) \\ &= \operatorname{argmax}_e \sum_{m=1}^M \lambda_m h_m(e, f)\end{aligned}$$

对数线性模型 vs. 噪声信道模型

- 取以下特征和参数时，对数线性模型等价于噪声信道模型：
 - 仅使用两个特征
 - $h_1(e, f) = \log p(e)$
 - $h_2(e, f) = \log p(f|e)$
 - $\lambda_1 = \lambda_2 = 1$

对数线性模型：Och 的实验 (1)

- 方案

- 首先将信源信道模型中的翻译模型换成反向的翻译模型，简化了搜索算法，但翻译系统的性能并没有下降；
- 调整参数 λ_1 和 λ_2 ，系统性能有了较大提高；
- 再依次引入其他一些特征，系统性能又有了更大的提高。

对数线性模型：Och 的实验 (2)

- 其他特征

- 句子长度特征 (WP)：对于产生的每一个目标语言单词进行惩罚；
- 附加的语言模型特征 (CLM)：一个基于类的语言模型特征；
- 词典特征 (MX)：计算给定的输入输出句子中有多少词典中存在的共现词对。

对数线性模型：Och 的实验 (3)

- 实验结果

	objective criteria [%]					subjective criteria [%]	
	SER	WER	PER	mWER	BLEU	SSER	IER
Baseline($\lambda_m = 1$)	86.9	42.8	33.0	37.7	43.9	35.9	39.0
ME	81.7	40.2	28.7	34.6	49.7	32.5	34.8
ME+WP	80.5	38.6	26.9	32.4	54.1	29.9	32.2
ME+WP+CLM	78.1	38.3	26.9	32.1	55.0	29.1	30.9
ME+WP+CLM+MX	77.8	38.4	26.8	31.9	55.2	28.8	30.9

对数线性模型的优点

- 噪声模型只有在理想的情况下才能达到最优，对于简化的语言模型和翻译模型，取不同的参数值实际效果更好；
- 对数线性模型大大扩充了统计机器翻译的思路；
- 特征的选择更加灵活，可以引入任何可能有用的特征。

内容提要

- 对数线性模型
- 最小错误率训练
- 基于对数线性模型的词语对齐
- 基于短语的翻译模型
- 短语模型的解码算法
- 开源机器翻译系统简介
- 总结

对数线性模型的参数训练

- 目的是得到各个特征的参数 $\lambda_1 \dots \lambda_n$
- 常见的训练算法
 - GIS（最大熵模型的训练算法）
 - 感知机
 - 最小错误率（MER）：直接以评测指标（如 BLEU）最好为训练目标
 - 最大互信息（MMI）：把导致总体 BLEU 值最高的译文定义为好的译文，其他译文定义为不好的译文，进行判别式训练
 - 单纯形算法
- 目前通常使用最小错误率训练算法或单纯形算法

最小错误率参数训练算法

- Franz Josef Och, "Minimum Error Rate Training for Statistical Machine Translation". In "ACL 2003: Proc. of the 41st Annual Meeting of the Association for Computational Linguistics", Japan, Sapporo, July 2003.
- Ashish Venugopal and Stephan Vogel, "Considerations in MCE and MMI training for statistical machine translation", Proceedings of the Tenth Conference of the European Association for Machine Translation (EAMT-05), Budapest, Hungary, May 2005
- 开源工具: <http://www.cs.cmu.edu/~ashishv/mer.html>
(针对 BLEU 的 MER 算法, 用 matlab 编写)

最小错误率参数训练算法

假设搜索得到一个汉语句子 f 的英文翻译 e ，这个汉语句子的参考译文是 r ，我们用函数 $E(r, e)$ 表示错误个数，同时假设错误个数对于多个句子是可以累加的，即：

$$E(r, e) = E(r_1^S, e_1^S) = \sum_{s=1}^S E(r_s, e_s)$$

最小错误率参数训练算法

在机器翻译训练中，我们通常利用一个开发集来调试对数线性模型的参数。在开发集中，一般的对于一个汉语句子 f_s ，我们根据解码算法会得到一个 **N-best** 的候选译文的集合 C_s ，那么，我们的目标就是优化参数，使得这 S 个句子的候选集合中错误最少的译文被选择出来：

$$\hat{\lambda}_1^M = \operatorname{argmin} \left\{ \sum_{s=1}^S E(r_s, \hat{e}(f_s, \lambda_1^M)) \right\}$$
$$\hat{e}(f_s, \lambda_1^M) = \operatorname{argmax}_{e \in C_s} \left\{ \sum_{m=1}^M \lambda_m h_m(e, f_s) \right\}$$

针对 BLEU 的最小错误率参数训练 (1)

- 我们将错误率定义为 BLEU 值的负值 (一般取 $N=4$, $w_n=1/N$):

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

其中 p_n 是 n-gram 的准确率, BP 是长度惩罚因子, 一般的

- 注意这里 BLEU 值是对整个开发集来计算的, 不具有可加性。

因此相应的最小错误率参数应表示为:

$$\hat{\lambda}_1^M = \operatorname{argmax} \{ BLEU(r_{1\dots s}, \hat{e}(f_{1\dots s}, \lambda_1^M)) \}$$

$$\hat{e}(f_s, \lambda_1^M) = \operatorname{argmax}_{e \in C_s} \left\{ \sum_{m=1}^M \lambda_m h_m(e, f_s) \right\}$$

针对 BLEU 的最小错误率参数训练(2)
















- 我们的目标是通过调整对数线性模型的权重，使得系统在开发集上总体 **BLEU** 值最高
 - 首先，利用初始参数，对开发集中的每个句子进行翻译，得到该句子的 **n-best** 候选译文
 - 依次调整每个参数，参数调整后，每个句子的 **n-best** 候选译文的评分（特征值的线性加权和）将发生变化，根据评分重新排列顺序，可以得到新的 **1-best** 候选译文，由这些新 **1-best** 候选结果组成的总体翻译结果的 **BLEU** 值也发生变化。
 - 我们的目标就是调整参数，使得这个 **1-best** 候选译文组成的翻译结果的 **BLEU** 值最高
















针对 BLEU 的最小错误率参数训练 (3)
















- 在算法中，每次调整参数后都要做两件事：
 - 重新选择某个句子的最优候选译文；
 - 根据每个句子的最优候选译文，重新计算总体 BLEU 值。
- 为了便于计算，我们为每个候选译文建立包含以下信息的数据结构，这样调整译文时只要重新对每一阶 n-gram 重新求和即可得到新的 BLEU 值：

$H_1(e,f) \dots H_m(e,f)$	n1_match	n1_sys	...	n4_match	n4_sys	length
所有特征	unigram 匹配数	unigram 总数		4-gram 匹配数	4-gram 总数	译文长度

针对 BLEU 的最小错误率参数训练 (4)

f1			
f2			
f3			
f4			
f5			

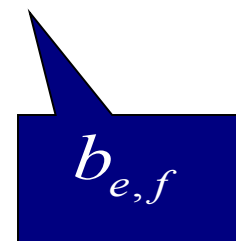
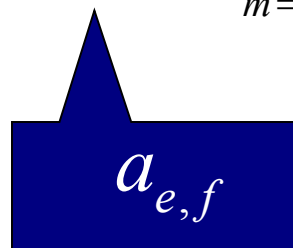
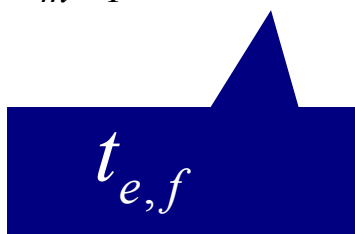
f1			
f2			
f3			
f4			
f5			

f1			
f2			
f3			
f4			
f5			

针对 BLEU 的最小错误率参数训练 (5)

- 在进行参数训练时，先调节某一维的参数，将其他的参数看作常数。
- 一个候选译文的得分表示如下：

$$e \Leftrightarrow \sum_{m=1}^M \lambda_m h_m(e, f) = \lambda_d h_d(e, f) + \sum_{m=1, m \neq d}^M \lambda_m h_m(e, f)$$



针对 BLEU 的最小错误率参数训练 (6)

令：

$$b_{e,f} = \sum_{m=1, m \neq d}^M \lambda_m h_m(e, f)$$

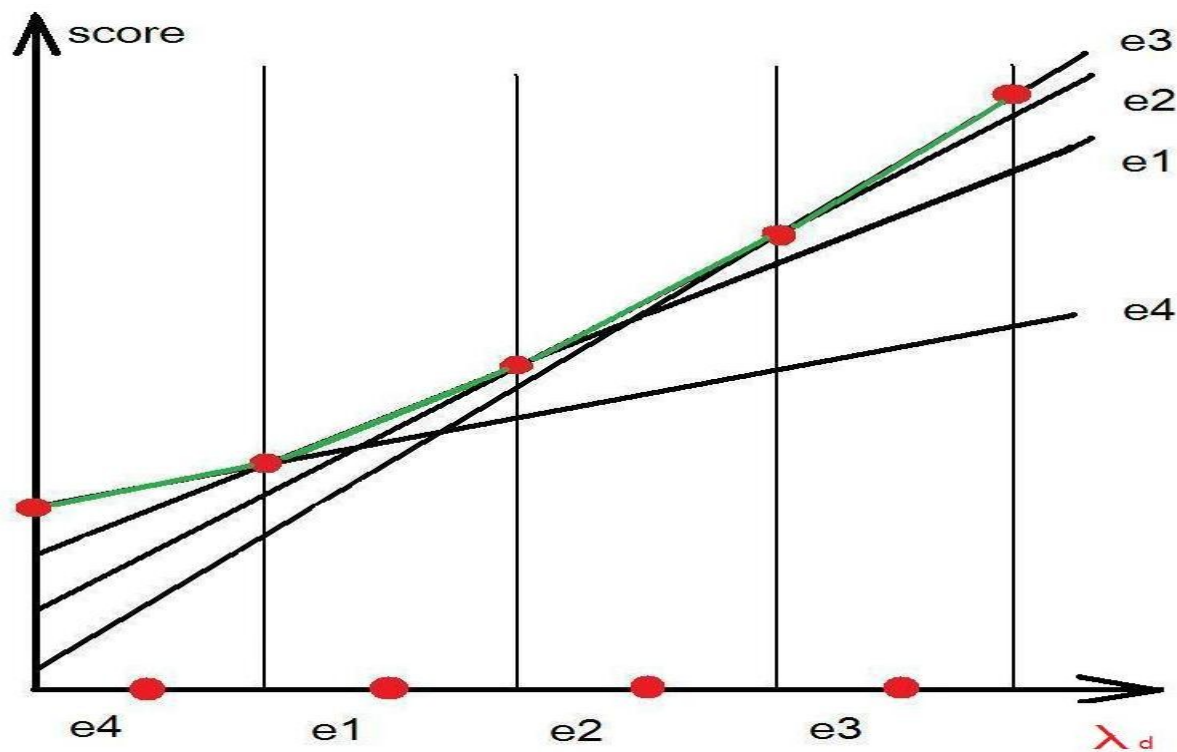
$$a_{e,f} = h_d(e, f)$$

$$t_{e,f} = \sum_{m=1}^M \lambda_m h_m(e, f)$$

$$t_{e,f} = a_{e,f} \lambda_d + b_{e,f}$$

每个句子对应一条直线
其中，待调整的 lambda
对应的特征值是斜率

针对 BLEU 的最小错误率参数训练 (7)



固定其他维的参数 λ ，只调节 λ_d ，各个候选译文分数的变化情况。最优候选译文总是在一些临界点发生变化。只要计算这些临界点的分数即可。

针对 BLEU 的最小错误率参数训练 (8)

- 训练集中每一个句子 f ，其 N -best 都对应一簇直线，根据 log-linear 模型，当 $\lambda = d$ 固定时，我们选择分数最高的译文作为最终译文，对应于在 d 点的分数最高的那条直线 l 。当我们取另外一个 $\lambda = d'$ 时，如果分数最高的那条直线 l' 与 l 不是同一条直线，意味着，BLEU 值会发生变化。而这两条直线必然有交点，这个交点称为临界点。
- 只有在临界点两侧 BLEU 才会发生变化！

针对 BLEU 的最小错误率参数训练 (9)

- 对于每个句子存在若干个临界点，对于整个测试集来说，我们可以将所有句子的临界点合并后排序，这样得到的相邻两个临界点之间的 λ 值变化不会导致任何一个句子的译文选择发生变化，这样我们只要在每个区间上任选一个值（通常取中点）作为代表，就可以找到最好的那个 λ 值的区间。
- 我们采用贪婪算法，每次调整一个 λ 值，反复迭代，使得总体 BLEU 值逐步提高（错误率降低）

针对 BLEU 的最小错误率参数训练 (10)

1. iter=1; lasterror=0; newerror=1; epsilon=0.001;
2. While (iter<Maxiter and abs(lasterror-newerror)> epsilon)
3. for d=1 to M //for each dimension
4. for s=1 to S //for each sentences in training corpus
5. Compute critical value and delta error
6. endfor
7. merge all critical value and sort them
8. compute error within each pair of non-identical boundaries
9. Select λ as the midpoint of the interval corresponding
 to the lowest error
10. end for
11. iter++;
12. lasterror=newerror;
13. newerror = lowest error
14. end while

针对 BLEU 的最小错误率参数训练(11)

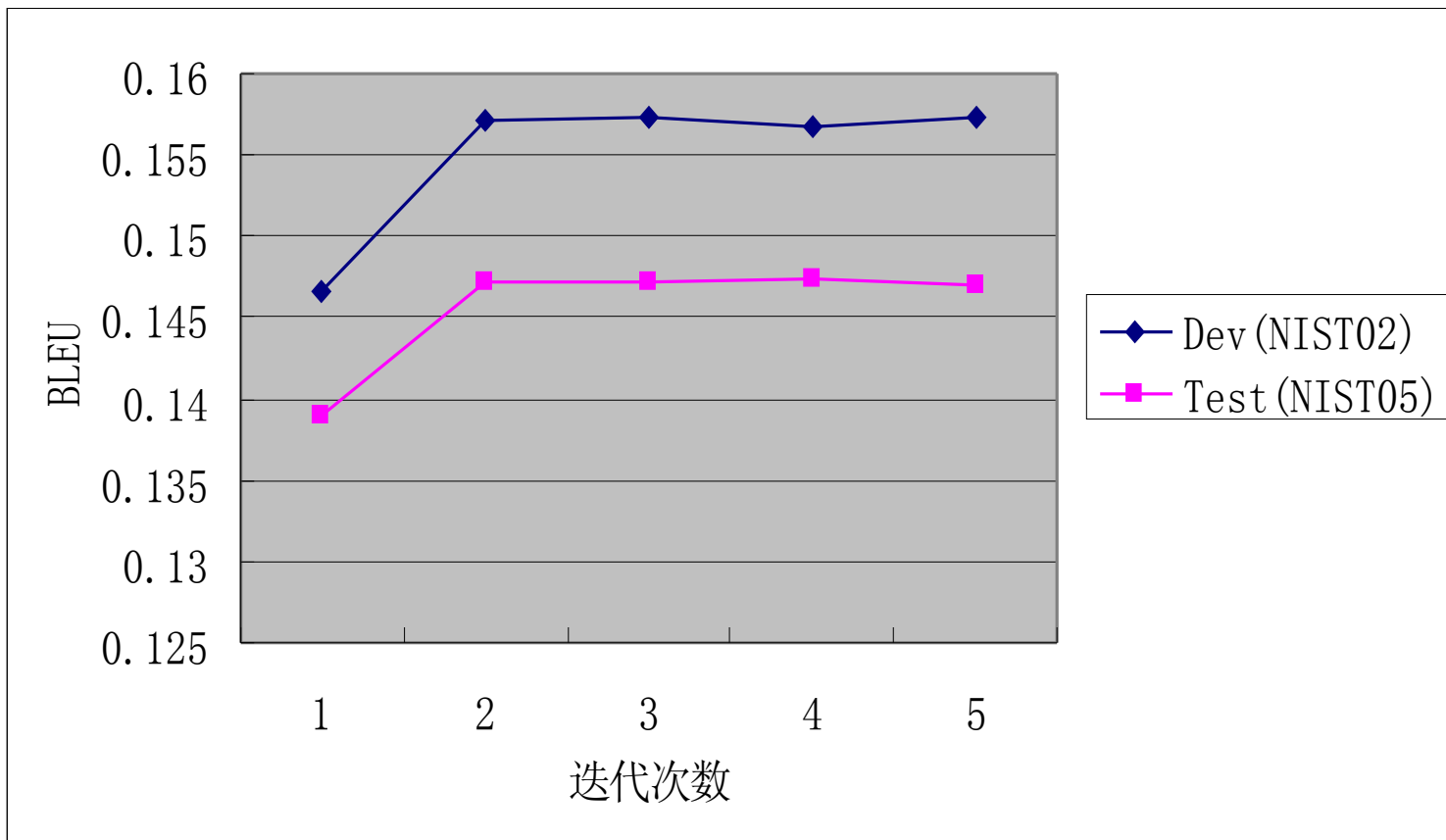
N-best 的合并以及重新解码:

1. 初始化 λ , 进行解码, 产生 N-best 特征文件 f_1
2. 最小错误率训练得到新的 λ
3. 利用新的 λ 进行解码, 产生 N-best 特征文件 f_2
4. 合并 f_1 、 f_2 , 生成新的特征文件 f_1'
5. 如果 f_1' 与 f_1 相等, 即没有新的 N-best 产生, 结束, 否则转6
6. $f_1=f_1'$, 重新进行参数训练得到新的 λ , 转3

实验

- 实验系统采用简单的基于短语的方法，利用 **log-linear** 模型将语言模型 (LM)，翻译模型 (TM)，以及句子长度模型 (SL) 统一起来。
- **NIST02** 年的测试集用作开发集，**NIST05** 年作为测试集
- 从训练语料库中抽取出约4492万条双语短语，利用 **NIST02** 年和05年测试文件选出126万条双语短语，语言模型：利用 **CMU** 工具在 **BNC** 语料库上训练得到的2-gram，除此之外，没有用到其他任何资源。

实验（续）



最小错误率的优缺点

- 直接针对评测方法训练参数，非常有效，这同时也是遭到人们批判的一点（局限于评测方法）
- 与初值有关，收敛到局部极值，有时初值不好，得到的参数效果并不好
- 多次迭代，速度较慢。每当训练集或者解码器有所变化，就要重新训练，非常繁琐

对数线性模型的特征 (1)

- 无论在噪声信道模型还是在対数线性模型中，语言模型和翻译模型都是两个最主要的特征
- 对于语言模型，目前主流的做法都还是采用 n 元语法，还没有发现哪些方法能够超过这种简单的模型
- 对于翻译模型，研究者进行了大量的尝试
 - 最早期的 IBM Model 1-5 是基于词的翻译模型
 - 目前最成熟和稳定的模型是基于短语的翻译模型
 - 基于句法的翻译模型近年来也取得了较大进展
- 在对数线性模型中，多个翻译模型和语言模型可以同时使用

对数线性模型的特征 (2)

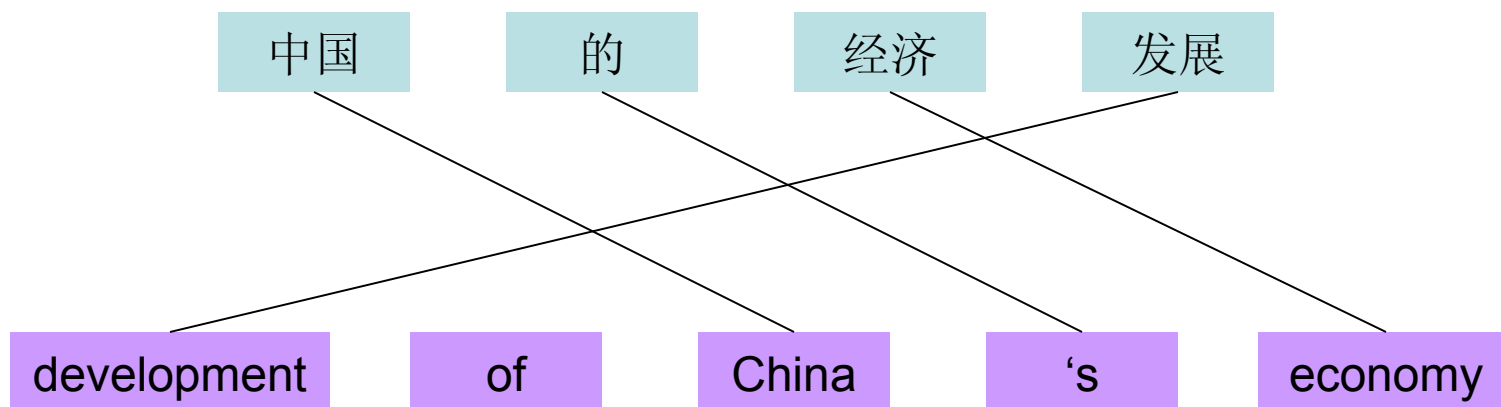
- 其他特征
 - 词典特征
 - 长度特征：句子单词数。这个特征可以一定程度上避免由于使用语言模型导致的过于偏向短句子的倾向
 -

内容提要

- 对数线性模型
- 最小错误率训练
- 基于对数线性模型的词语对齐
- 基于短语的翻译模型
- 短语模型的解码算法
- 开源机器翻译系统简介
- 总结

词语对齐

- 词语对齐的目标是确定双语句中词语之间的对应关系。



词语对齐的对数线性模型

- 模型公式

$$P(a|f, e) = \frac{\exp \sum_{m=1}^M \lambda_m h_m(a, f, e)}{\sum_{a'} \exp \sum_{m=1}^M \lambda_m h_m(a', f, e)}$$

特征权重 特征函数

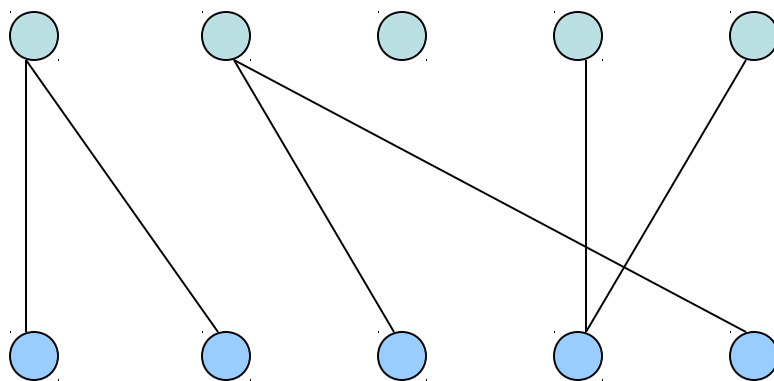
归一化因子

- 搜索公式

$$\hat{a} = \operatorname{argmax}_a \sum_{m=1}^M \lambda_m h_m(a, f, e)$$

特征举例：连线计数

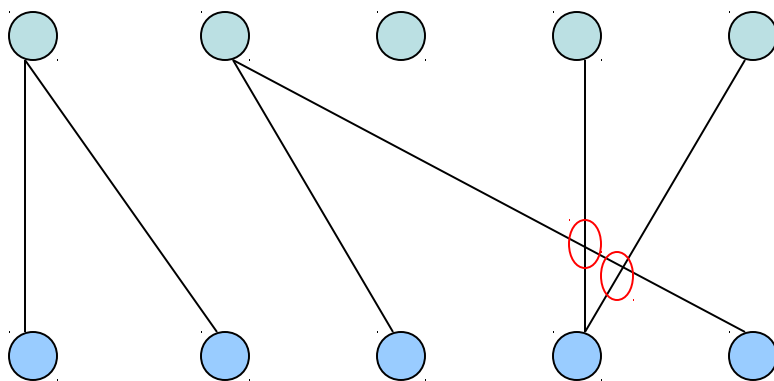
- 连线计数统计对齐中的连线数量。



连线计数是6

特征举例：交叉计数

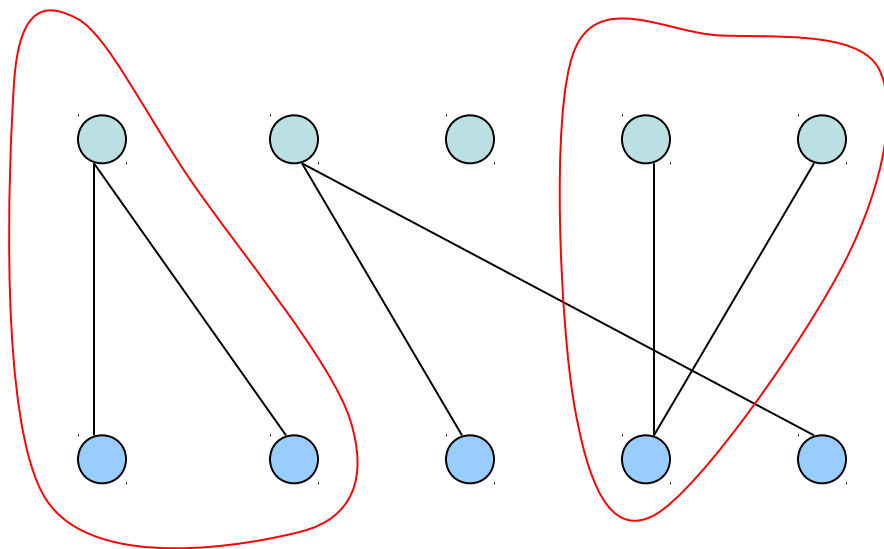
- 交叉计数统计对齐中连线交叉的数量。



交叉计数是2

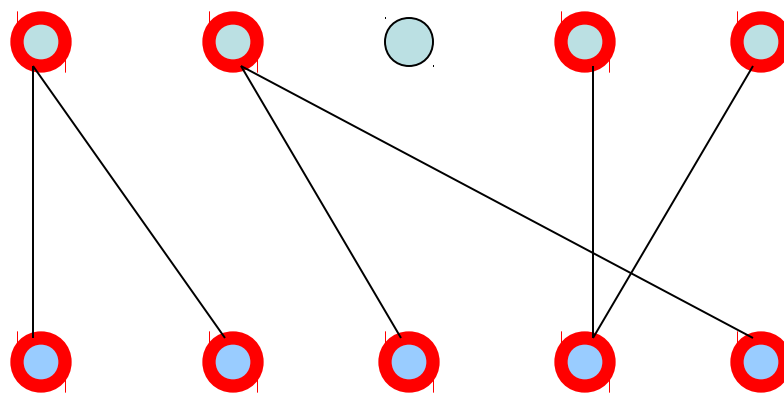
特征举例：邻居计数

- 邻居计数统计对齐中相邻连线对的数量。



特征举例：连接词语计数

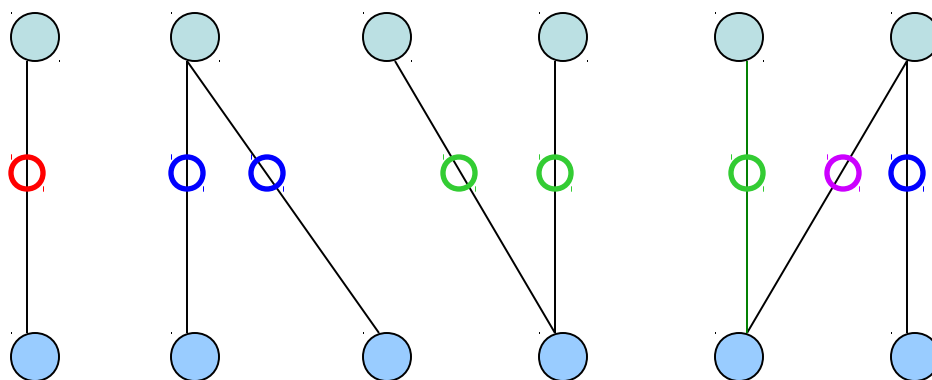
- 连接词语计数统计对齐中有连线的词语的数量。



连接词语计数是9

特征举例：连线类型计数

- 连线类型计数统计对齐中一对一、一对多、多对一和多对多连线的数量。



一对一	1
一对多	3
多对一	3
多对多	1

实验结果：对齐质量

Table 9
Comparison with participating systems on three shared tasks.

Shared Task	Task	Participating Systems	Vigne
HLT-NAACL 2003	Romanian-English, non-null, limited	28.9-52.7	23.8
	Romanian-English, null, limited	37.4-59.8	27.1
	English-French, non-null, limited	8.5-29.4	4.2
	English, null, limited	18.5-51.7	4.7
ACL 2005	English-Inuktitut, limited	9.5-71.3	9.0
	Romanian-English, limited	26.6-44.5	24.8
	English-Hindi, limited	51.4	45.1
HTRDP 2005	Chinese-English, unlimited	23.5-49.2	14.5

实验结果：翻译质量

	Moses	Hiero	Lynx
IBM Model 4 C→E	0.2465	0.2570	0.2484
IBM Model 4 E→C	0.2055	0.2348	0.2157
IBM Model 4 intersection	0.2013	0.2319	0.2116
IBM Model 4 union	0.2430	0.2408	0.2511
IBM Model 4 refined method	0.2421	0.2404	0.2417
IBM Model 4 grow-diag-final	0.2495	0.2579	0.2432
Cross-EM HMM	0.2357	0.2491	0.2479
F-measure ($\alpha = 0.1$) tuned	0.2387	0.2530	0.2603
F-measure ($\alpha = 0.3$) tuned	0.2491	0.2675	0.2614
F-measure ($\alpha = 0.5$) tuned	0.2567	0.2659	0.2427
F-measure ($\alpha = 0.7$) tuned	0.2367	0.2538	0.2470
F-measure ($\alpha = 0.9$) tuned	0.2185	0.2473	0.2390

Table 4: Effect on translation quality.

内容提要

- 对数线性模型
- 最小错误率训练
- 基于对数线性模型的词语对齐
- 基于短语的翻译模型
- 短语模型的解码算法
- 开源机器翻译系统简介
- 总结

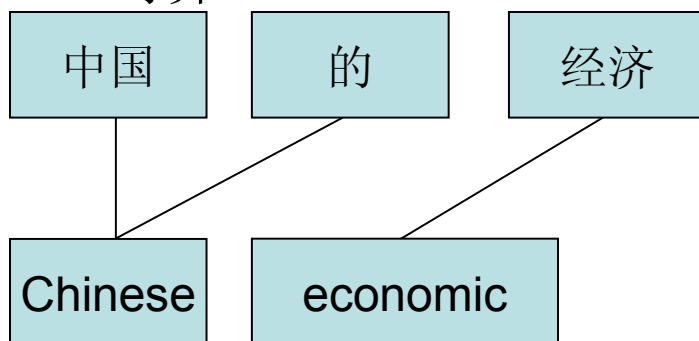
基于短语的翻译模型

- 基于词的翻译模型存在的问题
- 基于短语的统计机器翻译
- 短语的自动抽取算法

基于词的翻译模型存在的问题(1)

- 以词作为翻译的最小单位，从语料库中自动学习单词的翻译，对于一个词翻译到多个词的情况，都分解成一个词到一个词的翻译概率，很不合理

比如下面例子，将“中国 的” - “Chinese”的概率分解成“中国” - “Chinese”的概率和“的” - “Chinese”的概率，显然不合理，因为“的”只有在前面出现“中国”的时候才可能跟“Chinese”对齐



基于词的翻译模型存在的问题(2)

- **IBM** 模型只刻画了词到词的翻译概率，词翻译的时候没有考虑上下文，难以刻画一些固定搭配、习惯用法的翻译

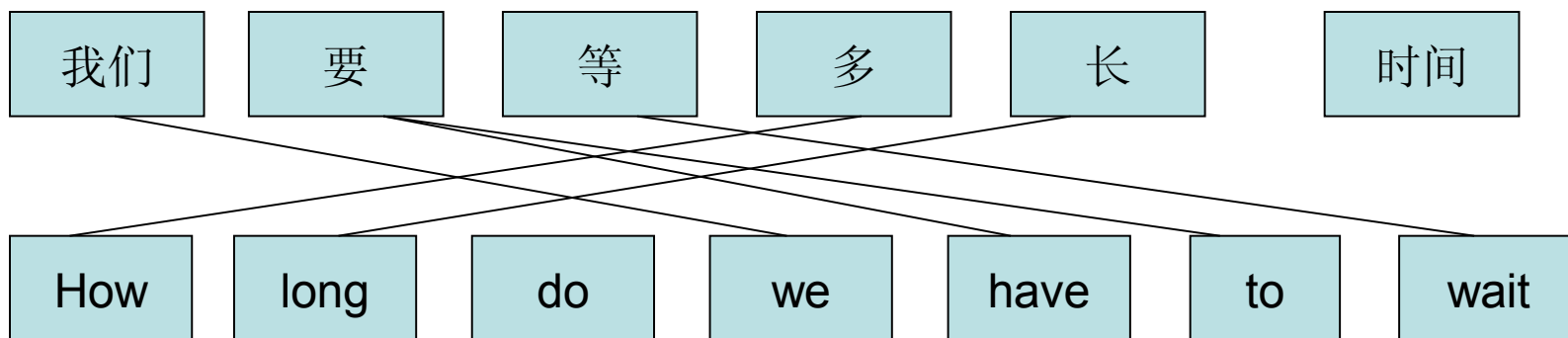
看 书 see book | watch book | **read book**

看 电视 **watch TV** | look TV

基于词的翻译模型存在的问题(3)

- 词序调整的复杂性

IBM 模型中词语调序模型过于简单，很难刻画复杂的词序调整规律



基于短语的翻译模型 (1)

- 基本思想

- 把训练语料库中所有对齐的短语及其翻译概率存储起来，作为一部带概率的短语词典
- 这里所说的短语是任意连续的词串，不一定是一个独立的语言单位
- 翻译的时候将输入的句子与短语词典进行匹配，选择最好的短语划分，将得到的短语译文重新排序，得到最优的译文

- 问题：

- 短语如何抽取？
- 短语概率如何计算？

基于短语的翻译模型 (2)

- 假设 \mathbf{f} 和 \mathbf{e} 之间存在一个短语对齐 B ，而且这个短语对齐是一一对应的，那么：

$$P(f_1^J | e_1^I) = \sum_B P(f_1^J, B | e_1^I) = \sum_B P(B | e_1^I) P(f_1^J | B, e_1^I)$$

- 假设短语划分的概率 $\Pr(B | e_1^I)$ 为均匀分布：

$$P(B | e_1^I) = \sum \alpha(e_1^I)$$

- 于是：

$$P(f_1^J | e_1^I) = \alpha(e_1^I) \sum_B P(f_1^J | B, e_1^I)$$

基于短语的翻译模型 (3)

- 假设短语的翻译是互相独立的，并且各种短语顺序调整的概率完全相同，那么：

$$P(f_1^J | e_1^J) = \alpha(e_1^J) \sum_B \prod_k p(\tilde{f}_k | \tilde{e}_k)$$

这里 \tilde{f}_k 和 \tilde{e}_k 是在对齐 B 下源语言和目标语言的短语，而 $p(\tilde{f}_k | \tilde{e}_k)$ 可以通过对短语对齐的语料库统计得到：

$$p(\tilde{f}_k | \tilde{e}_k) = \frac{N(\tilde{f}_k, \tilde{e}_k)}{N(\tilde{e}_k)}$$

基于短语的翻译模型 (4)

- 实际上，目前在计算短语翻译概率的时候，通常并不去真正生成一个短语对齐的语料库，而是直接从词语对齐的语料库上，去产生所有可能的短语对齐
- 所以，需要先利用 **IBM Model** 进行词语对齐，但：
 - **IBM Model** 只能产生单向一对多的对齐
 - 为了产生更合理的对齐，需要实现多对多对齐，通常的做法是：
 - 先用 **IBM Model** 对两个方向分别进行一对多对齐
 - 将两个对齐进行某种合并（交集、并集、部分并集），这个操作称为“平衡化”
- 根据词语对齐的结果抽取短语并计算概率

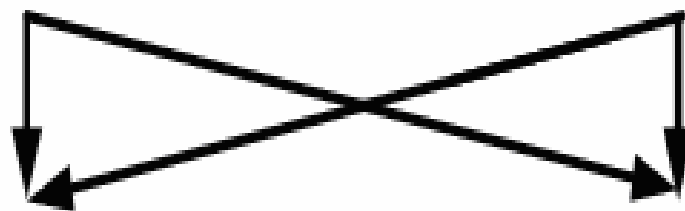
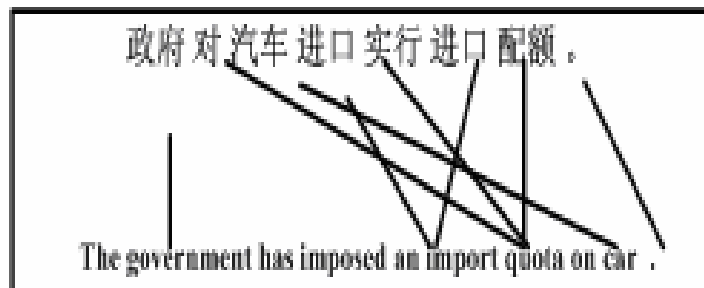
利用 IBM 模型进行双向对齐

政府对汽车进口实行进口配额。 <-> The government has imposed an import quota on car .

中文-英文的词语对齐结果

GIZA++双向
训练

英文-中文的词语对齐结果



两个方向的对齐结果取交集

Intersect

两个方向的对齐结果取并集

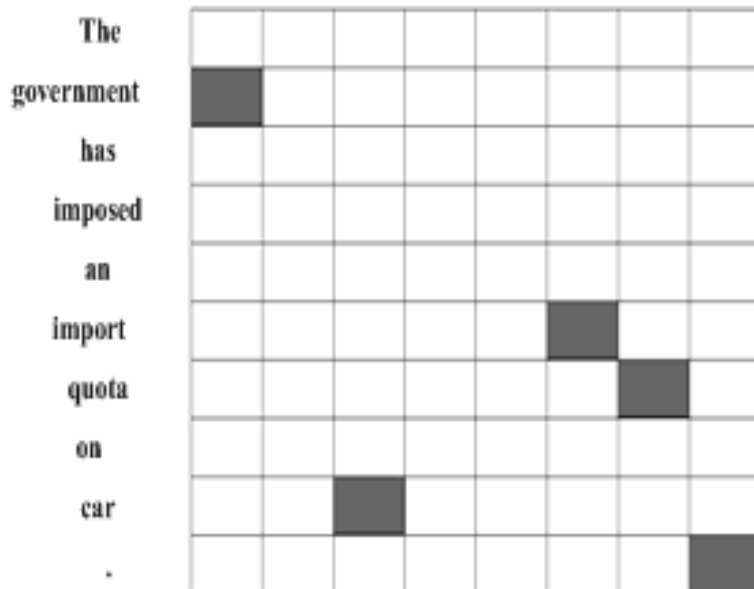
Union

对双向对齐结果取交集和并集

两个方向的对齐结果取交集

Intersect

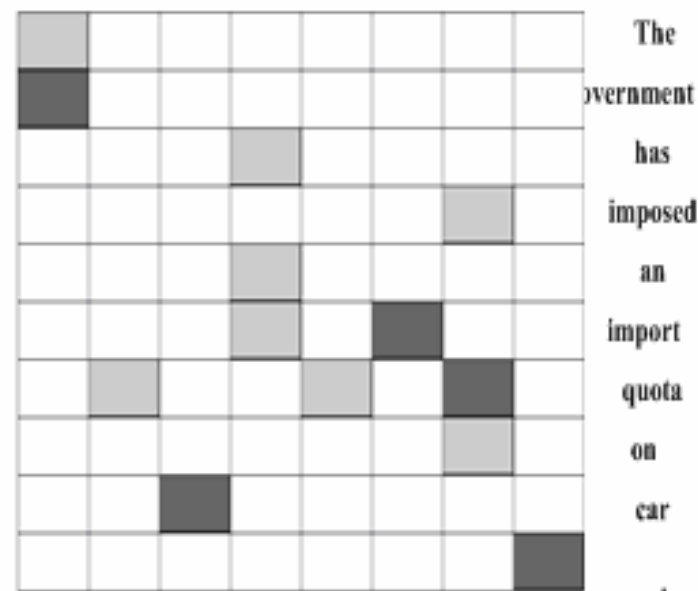
政府对汽车进口实行进口配额。



两个方向的对齐结果取并集

Union

政府对汽车进口实行进口配额。



以交集为中心点，扩展四周属于并集的点

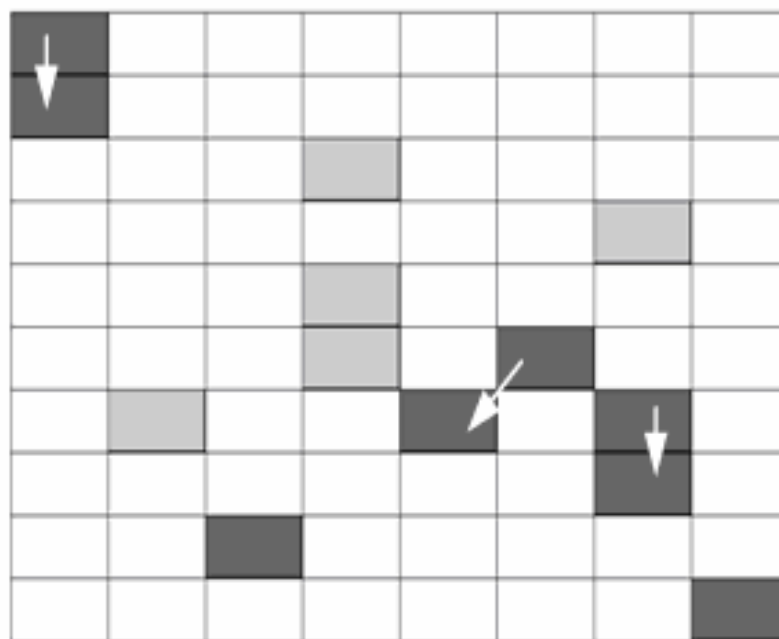
Heuristic

对 Grow 和 Diag 进行扩展

以交集为中心点，扩展四周属于并集的点

Heuristic

政府对汽车进口实行进口配额。



The
government
has
imposed
an
import
quota
on
car
.

以两个方向对齐结果的交集为中心点，检查其上下左右(grow)及对角(diag)相邻的8个点，若在并集中，则作为扩展的对齐点加入对齐序列中。

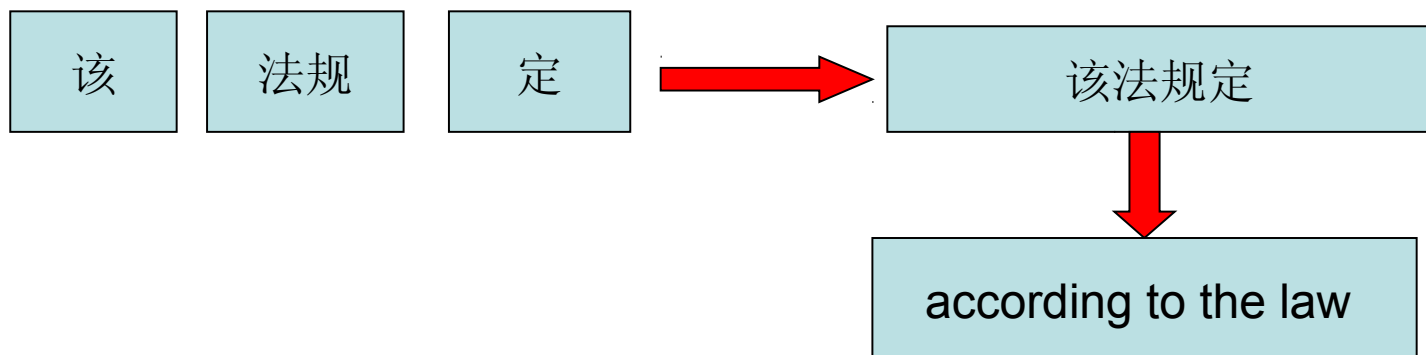
对 Final 进行扩展

基于短语建立翻译模型的优点

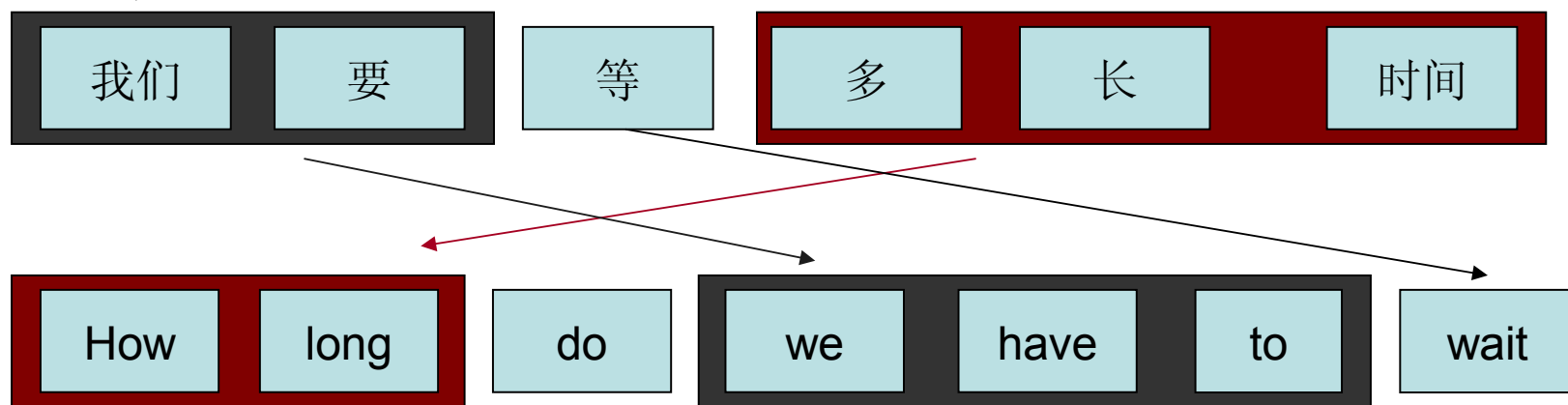
- 以短语作为翻译的最小单位，一般的，将短语定义为连续的词语串，从语料库中学习短语的翻译
- 固定搭配、习惯用法
 - 看书 read book
 - 看电视 watch TV
 - 毫无疑问 out of question
 - 不可能 out of **the** question

基于短语建立翻译模型的优点

- 分词错误



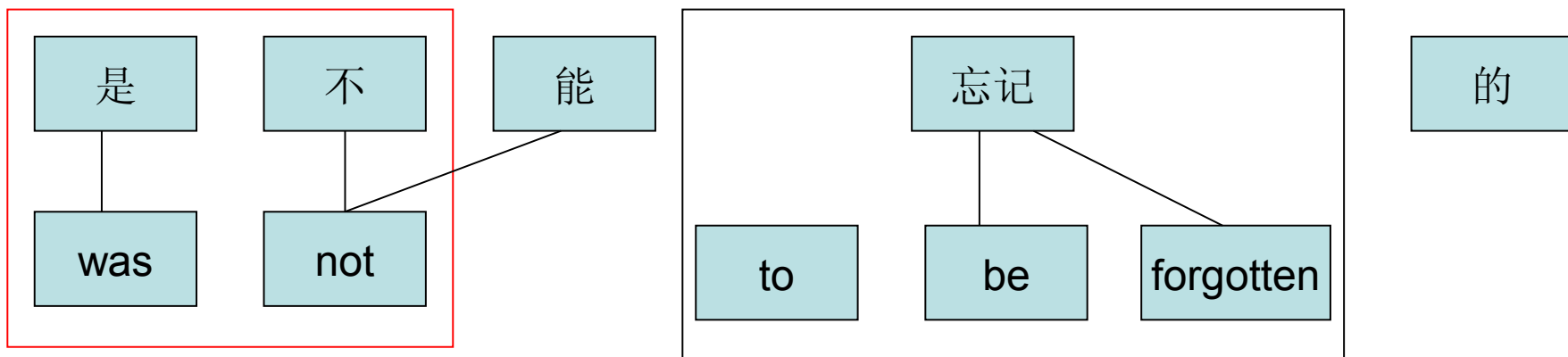
- 词序调整



基于词语对齐的短语自动抽取

- 如何定义一个双语短语 (f_j^{j+m}, e_i^{i+n}) :
 1. 短语内的单词在原来句子中位置上必须连续
 2. 双语短语必须与对齐矩阵相容，即根据源语言句子和目标语言句子的对齐矩阵，源语言短语中的词语 $f_{j'} (j \leq j' \leq j+m)$ 或者对齐到 NULL，或者对应的目标语言词语 $e_{i'}$ 必须在短语 e_i^{i+n} 中，反之亦然。

基于词语对齐的短语自动抽取



不相容

相容

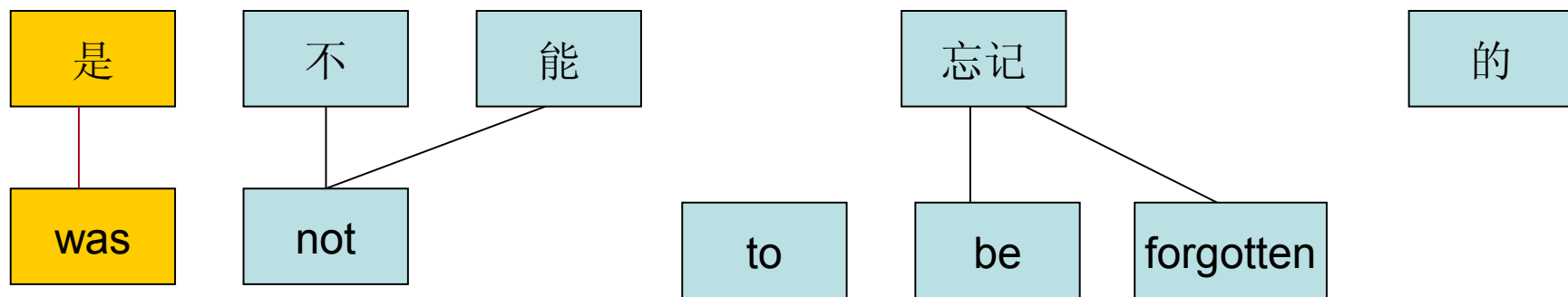
短语自动抽取算法

```
1 INPUT:  $e_1^I, f_1^J, A, \text{stoplist}_f, \text{stoplist}_e$ 
2 FOR i1=1 to I DO
3   FOR i2=i1 to I DO
4     FOR i=i1 to i2 DO
5       IF  $\forall j: A(i, j) = 0 \wedge e_i \notin \text{stoplist}_e$  THEN
6         goto 3;
7        $TP = \{j \mid \exists i: i_1 \leq i \leq i_2 \wedge A(i, j)\}$ ;
8       j1 = min(TP);
9       j2 = max(TP);
10      FOR j=j1 to j2 DO
11        IF  $\forall i: A(i, j) = 0 \wedge f_j \notin \text{stoplist}_f$  THEN
12          goto 3;
13        IF quasi-consecutive(TP) THEN
14           $SP = \{i \mid \exists j: j_1 \leq j \leq j_2 \wedge A(i, j)\}$ 
15          IF  $SP \subseteq \{i_1, i_{1+1}, \dots, i_2\}$  THEN
16            lene = i2 - i1 + 1;
17            lenf = j2 - j1 + 1;
18            IF lene*FERT < lenf || lenf*FERT < lene THEN
19              goto 3;
20             $BP = BP \cup \{(e_{i_1}^{i_2}, f_{j_1}^{j_2})\}$ 
21            WHILE  $j_1 > 0 \wedge \forall i: A(i, j_1) = 0 \wedge f_{j_1} \in \text{stoplist}_f$  DO
22              j'' = j2;
23              WHILE  $j'' \leq J \wedge \forall i: A(i, j'') = 0 \wedge f_{j''} \in \text{stoplist}_f$  DO
24                lenf = j'' - j1 + 1;
25                IF lene*FERT > lenf && lenf*FERT > lene THEN
26                   $BP = BP \cup \{(e_{i_1}^{i_2}, f_{j_1}^{j''})\}$ 
27                  j'' = j'' + 1;
28              j1 = j1 - 1;
29 output BP
```

图 3.3 改进后的短语抽取算法

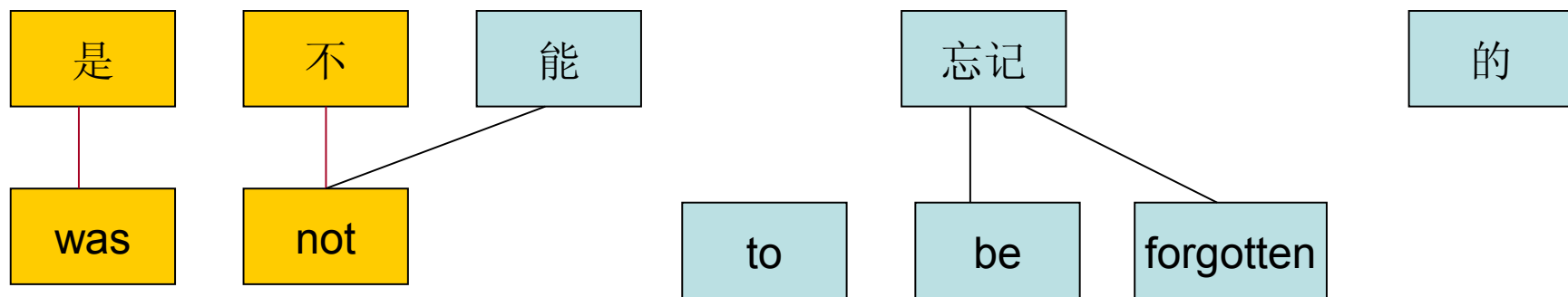
短语自动抽取算法运行示例 (1)

- 列举源语言所有可能的短语，根据对齐检查相容性



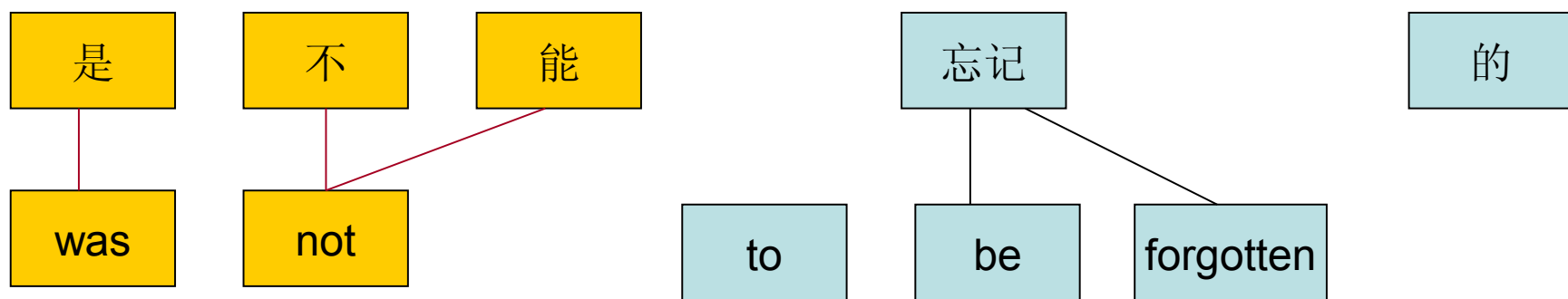
(是, was)

短语自动抽取算法运行示例 (2)



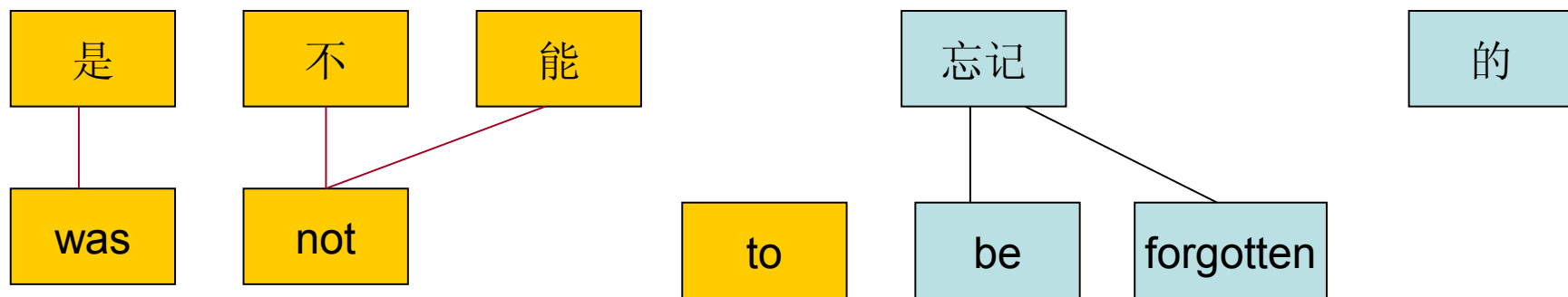
不相容

短语自动抽取算法运行示例 (3)



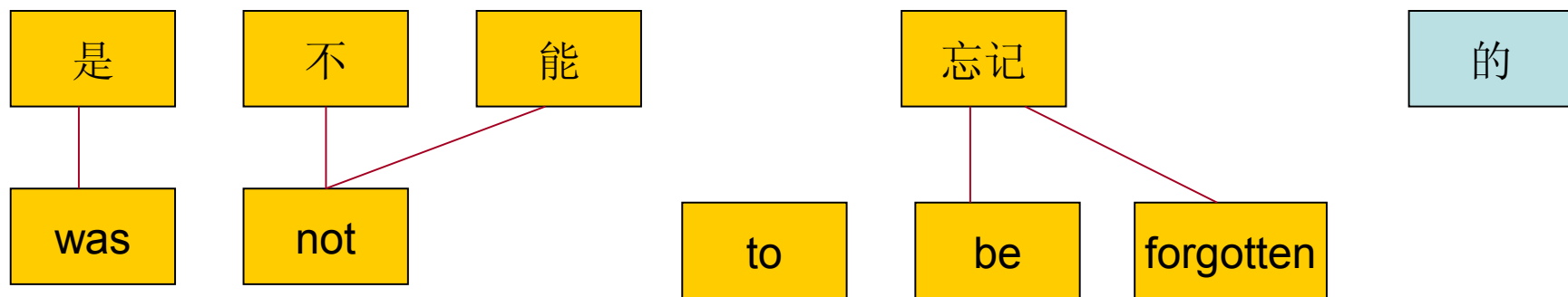
(是不能, was not)

短语自动抽取算法运行示例 (4)



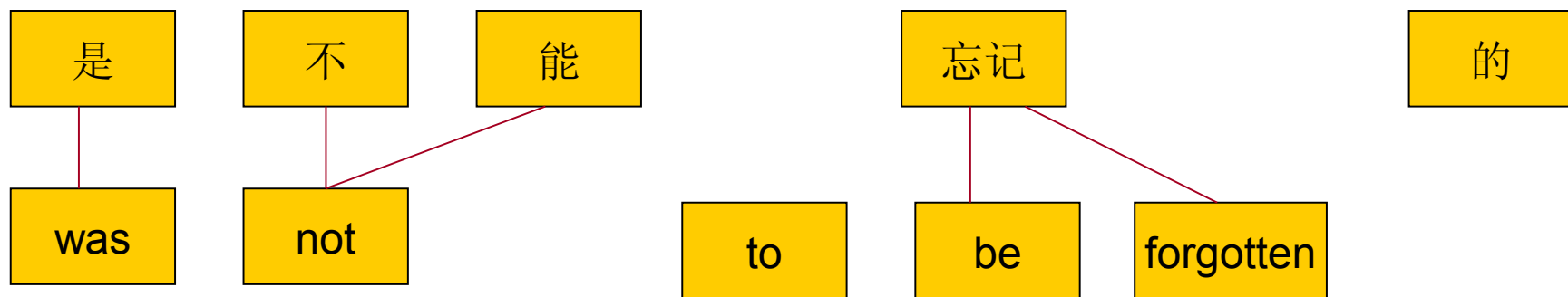
(是不能, was not to)

短语自动抽取算法运行示例 (5)



(是不能忘记, was not to be forgotten)

短语自动抽取算法运行示例 (6)

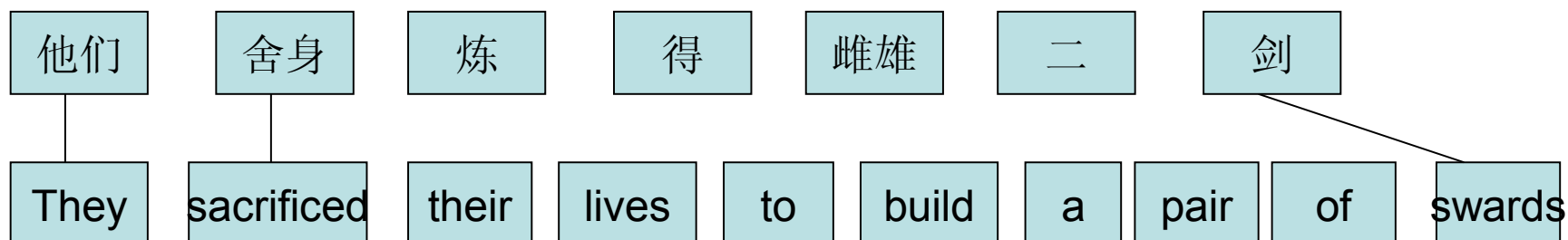


(是不能忘记的, was not to be forgotten)

短语表

source phrase	target phrase
是	was
是 不 能	was not
是 不 能	was not to
是 不 能 忘 记	was not to be forgotten
是 不 能 忘 记 的	was not to be forgotten
不 能	not
不 能	not to
不 能 忘 记	not to be forgotten
不 能 忘 记 的	not to be forgotten
忘 记	be forgotten
忘 记	to be forgotten
忘 记 的	be forgotten
忘 记 的	to be forgotten

如何减少垃圾短语 (1)



- 舍身 sacrificed
- 舍身 sacrificed their
- 舍身 sacrificed their lives
- 舍身 sacrificed their lives to
- 舍身 sacrificed their lives to build
- 舍身 sacrificed their lives to build a
- 舍身 sacrificed their lives to build a pair
- 舍身 sacrificed their lives to build a pair of

如何减少垃圾短语 (2)

- 设置单词的最大翻译个数 **FERT**
一个单词最多可以翻译为 **FERT** 个单词，设源短语有 n 个词，则目标短语最多有 $n * \text{FERT}$ 个词。

当 **FERT = 5**

舍身 sacrificed
舍身 sacrificed their
舍身 sacrificed their lives
舍身 sacrificed their lives to
舍身 sacrificed their lives to build

如何减少垃圾短语 (3)

- 设置停用词表 **stoplist**, 只有当对齐到空的单词在 **stoplist** 中时, 才进行扩展。

stoplist_e		stoplist_f	
单词	对齐到空的次数	单词	对齐到空的次数
the	5705141	的	2977887
of	3444587	,	465930
to	1748495	了	399438
in	1167247	在	270613
,	1116469	第	237133
a	666736	中	222386
for	630741	所	211090
that	589094	》	172683
and	44 0447	对	152805
on	415784	是	129218

表 3.2 从 300 万句对上得到的 stoplist

如何减少垃圾短语(4)

使用 **stoplist** 后抽到的短语:

他们	They
他们舍身	They sacrificed
舍身	sacrificed
剑	swords
剑	of swords

使用 **stoplist** 过滤短语表会导致翻译性能有一定程度下降。

双语短语的概率计算 (1)

- 利用共现次数计算概率

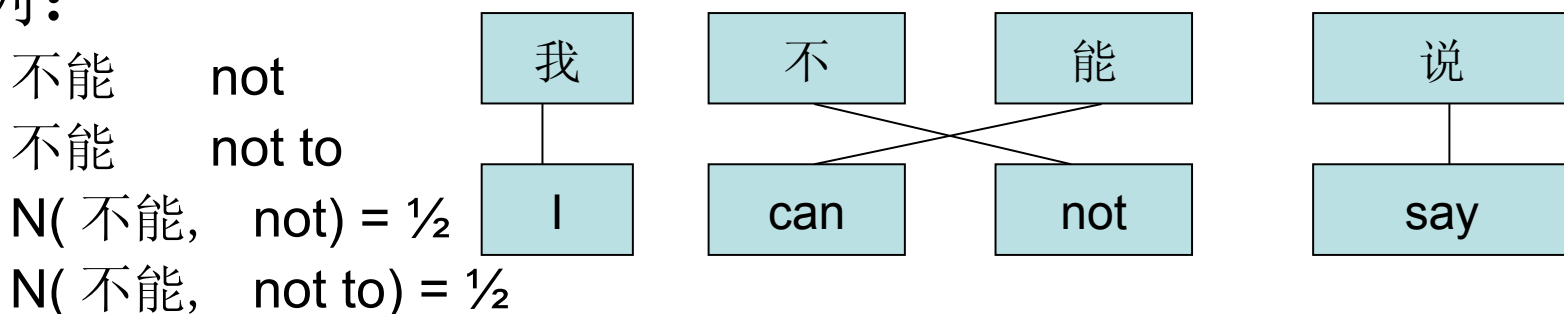
$$p(\tilde{f}|\tilde{e}) = \frac{N(\tilde{f}, \tilde{e})}{\sum_{f'} N(\tilde{f}', \tilde{e})}$$

其中 $N(\tilde{f}, \tilde{e})$ 表示短语 (\tilde{f}, \tilde{e}) 在语料库中出现的次数

如果 \tilde{e} 的一次出现对应有 N 个可能的翻译，那么每一个翻译对 $N(\tilde{f}, \tilde{e})$ 的贡献是 $1/N$

双语短语的概率计算 (2)

- 前例:



如果语料库中另外有一句话:

$$N(\text{不能}, \text{can not}) = 1$$

则

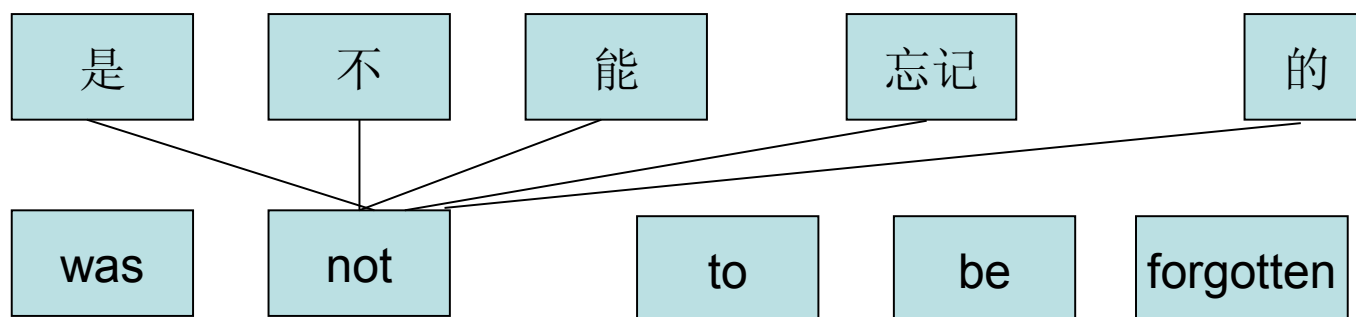
$$p(\text{not to}|\text{不能}) = \frac{1/2}{(1/2+1/2+1)} = 1/4$$

$$p(\text{not}|\text{不能}) = \frac{1/2}{(1/2+1/2+1)} = 1/4$$

$$p(\text{can not}|\text{不能}) = \frac{1}{(1/2+1/2+1)} = 1/2$$

双语短语的概率计算 (3)

- 仅利用共现次数计算概率信息不全面
较短的短语（如单词）出现次数多，概率不集中，较长的短语概率比较大



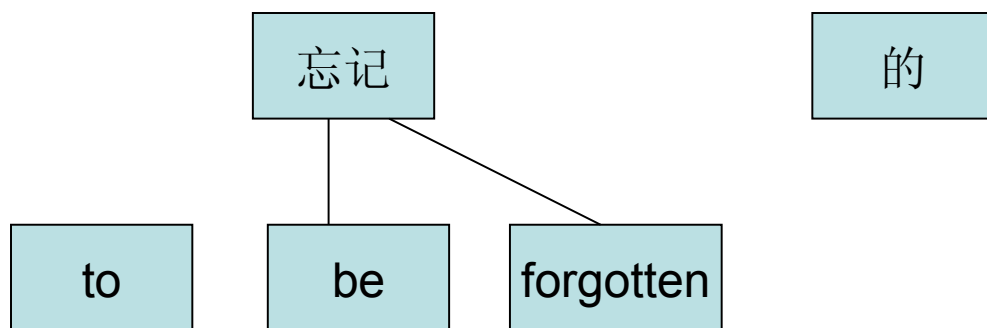
$$p(\text{not}|\text{是不能忘记的})=1$$

对齐错误使得概率计算不准确，影响解码

双语短语的概率计算 (4)

引入短语的词汇化翻译概率，利用 IBM Model 1 训练得到的词语翻译表计算双向的词语翻译概率

$$\text{lex}(\tilde{f}|\tilde{e}, a) = \prod_{j=1}^n \frac{1}{|\{i|(j, i) \in a\}|} \sum_{\forall (i, j) \in a} p(f_j|e_i)$$



$$\text{lex}(\text{忘记的}|to\ be\ forgotten) = \frac{1}{2} \times (p(\text{忘记}|be) + p(\text{忘记}|forgotten)) \\ \times p(\text{的}|NULL)$$

双语短语的概率计算 (5)

- 通常都使用双向的短语翻译概率和双向的词汇化翻译概率，一个四个概率作为特征，与其他特征一起，利用最小错误率算法调整特征参数

短语翻译概率表

$f \parallel e \parallel p(f|e) \text{ lex}(f|e) p(e|f) \text{ lex}(e|f) \text{ length}$

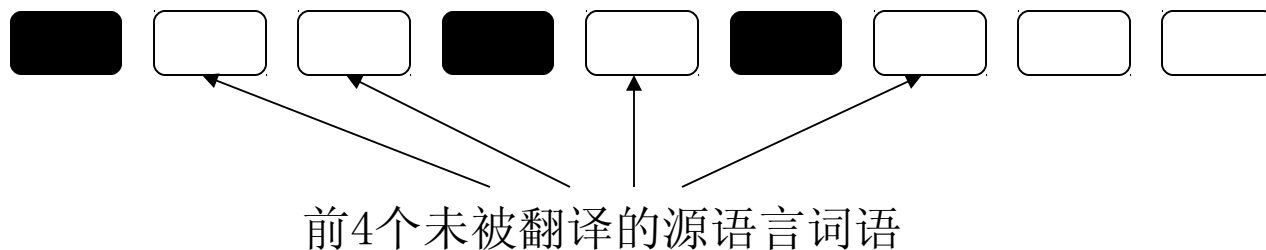
没有达成共识 ||| no consensus was reached ||| 1 0.00210153 1 8.87474e-05 2.718
没有达成共识。 ||| no consensus was reached . ||| 1 0.0017517 1 8.83361e-05 2.718
没有得到澄清 ||| clarified ||| 1 0.000592593 1 0.036396 2.718
没有得到南方的响应 ||| no response ||| 0.5 1.49065e-06 1 0.00921419 2.718
没有得到证实 ||| no evidence ||| 0.5 0.000178961 1 0.0021538 2.718
没有兑现 ||| has sent ||| 0.2 6.64599e-05 1 0.00346412 2.718
没有发生变化 ||| had not changed ||| 0.5 0.000141333 1 8.84361e-05 2.718
没有发现明显 ||| is no obvious ||| 1 0.00114645 1 0.000308419 2.718
没有犯罪 ||| no criminal ||| 1 0.0613205 1 0.0251376 2.718
没有犯罪纪录 ||| no criminal record ||| 1 0.0196688 1 0.0123866 2.718
没有放弃 ||| has not given up its ||| 1 0.000278368 1 8.34878e-06 2.718
没有改变 ||| There is no change ||| 0.5 0.0148622 0.333333 1.50262e-05 2.718
没有改变 ||| has not changed ||| 0.5 0.00505152 0.333333 0.00145408 2.718
没有改变 ||| is no change ||| 1 0.0283586 0.333333 0.00201351 2.718
没有改变。 ||| is no change . ||| 1 0.0236378 1 0.00200418 2.718
没有改变， ||| has not changed , and ||| 1 0.000628986 1 8.72846e-05 2.718
没有改变， 如果 ||| has not changed , and if ||| 1 0.000308107 1 5.71048e-05 2.718
没有工作 ||| without work ||| 1 0.0559111 1 0.0130721 2.718
没有工作许可证 ||| without work permits ||| 1 0.00559111 1 0.000344003 2.718
没有归还 ||| not repaid till now ||| 1 0.00498227 1 6.22208e-05 2.718
没有和平 ||| without a peaceful ||| 1 0.0398149 1 7.62298e-06 2.718

短语的调序

- 理论上，短语在目标端可以任意排序
- 也就是说，对于任何一个中间结果进行扩展时，如果当前还有 m 个短语未被翻译，那么下一个被翻译的短语可以是这 m 个短语中的任何一个
- 这样的话， n 个短语的句子对应的译文短语顺序可能性有 $n!$ 种，这么大的搜索空间是不可能被接受的

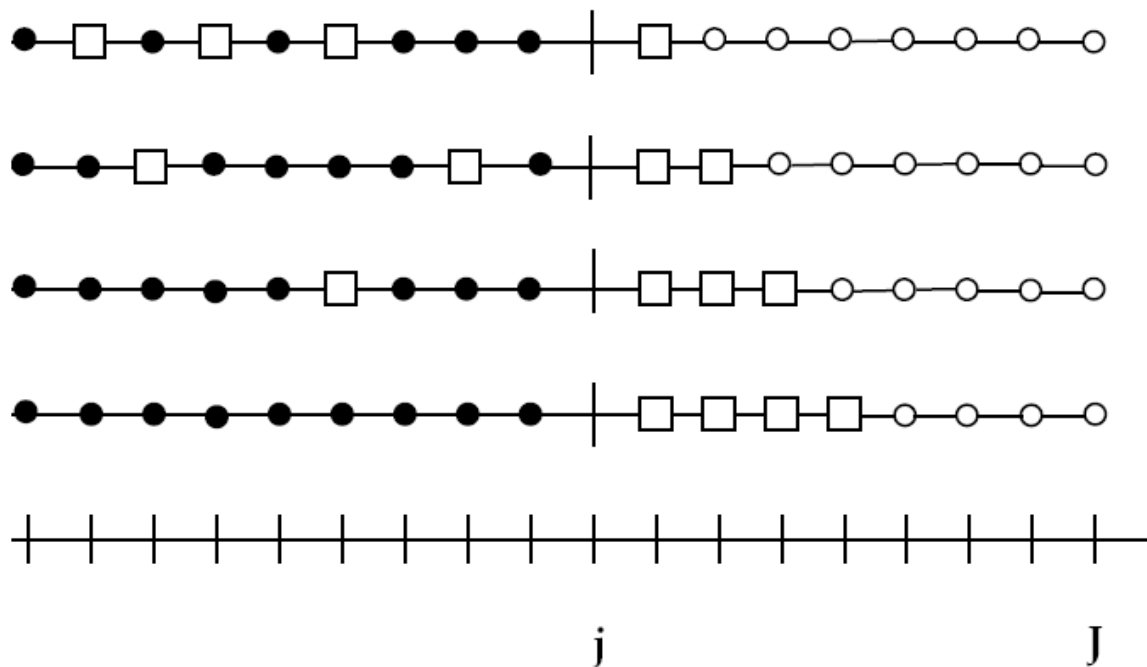
短语调序的 IBM 约束

- 为了缩小搜索的空间，通常在从左向右的解码算法中，我们采用某种约束来限定选择下一个要翻译的短语的范围。
- **IBM 约束**：从当前未被翻译的 m 个词中，选择最左边的 k 个未被翻译的词作为下一个要被翻译的候选短语的起点



短语调序的 IBM 约束

- uncovered position
- covered position
- uncovered position for extension



MOSES 采用的调序约束

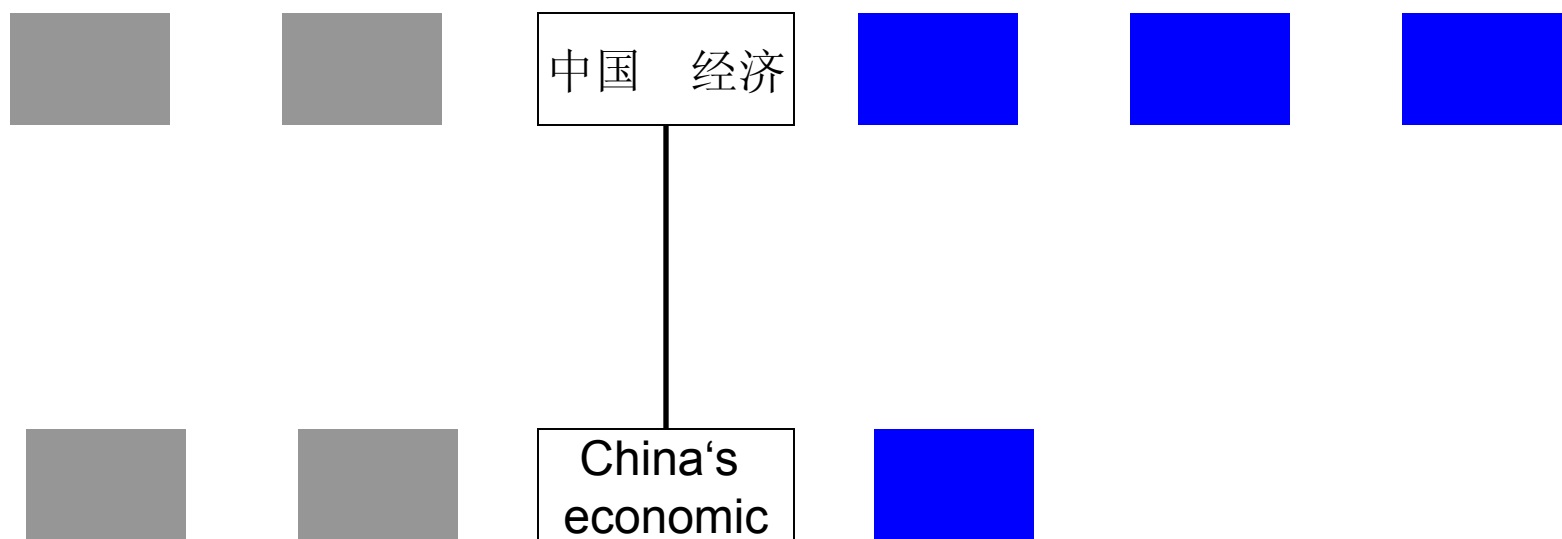
- 当前源语言词左右一个给定范围的窗口
- 问题：可能导致最后某些词无法被选择到

短语调序的调序模型

- 调序模型用于刻画短语调序对翻译概率的影响
- **IBM** 模型的模型2以后都有调序模型，但都比较粗糙，效果不好
- 基于短语的机器翻译方法中最常用的一种调序模型称为词汇化的调序模型，有时也简称为 **MSD** 模型。

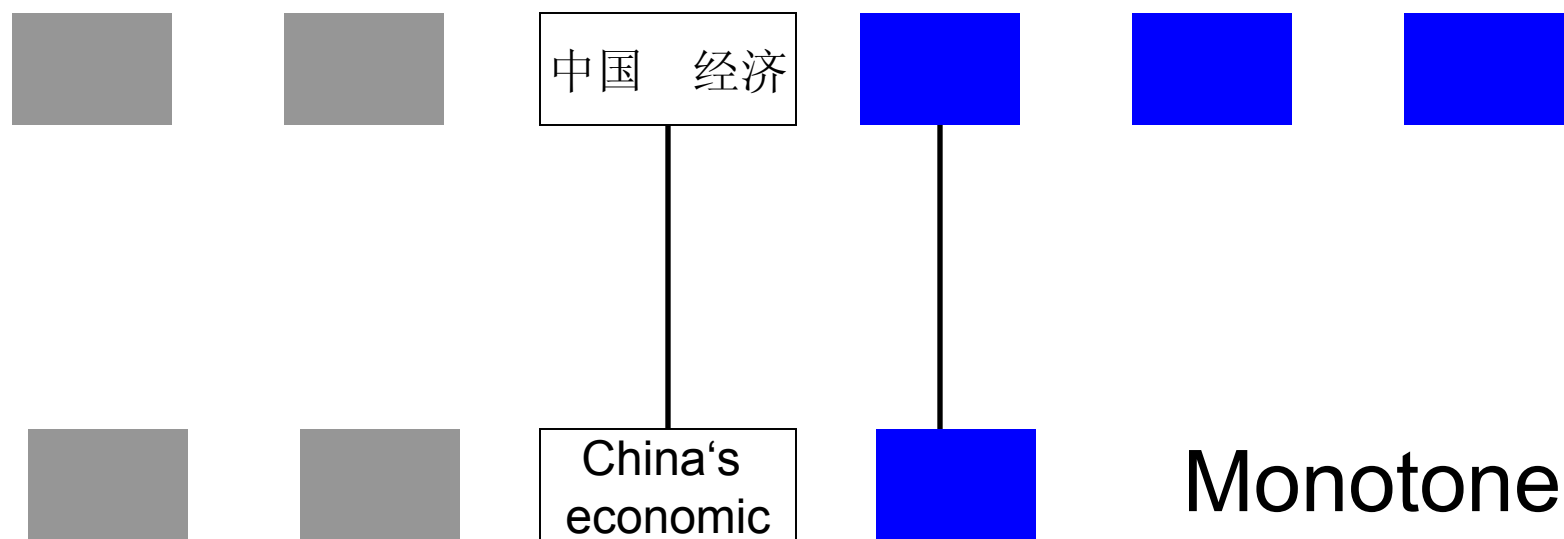
词汇化的调序模型

- **MSD**: 对每一对双语短语, 在训练语料库中统计目标端相邻的下一个短语在源语言端是 **Monotone**、**Swap** 和 **Discontinuous** 的概率



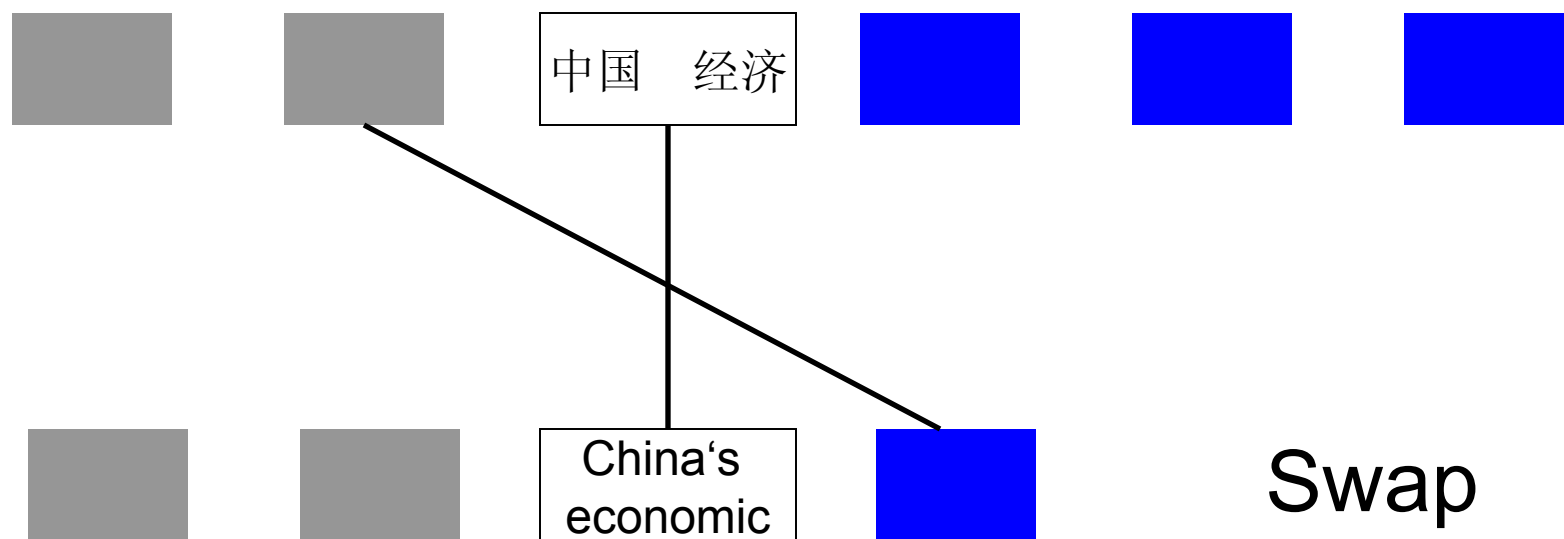
词汇化的调序模型

- **MSD**: 对每一对双语短语, 在训练语料库中统计目标端相邻的下一个短语在源语言端是 **Monotone**、**Swap** 和 **Discontinuous** 的概率



词汇化的调序模型

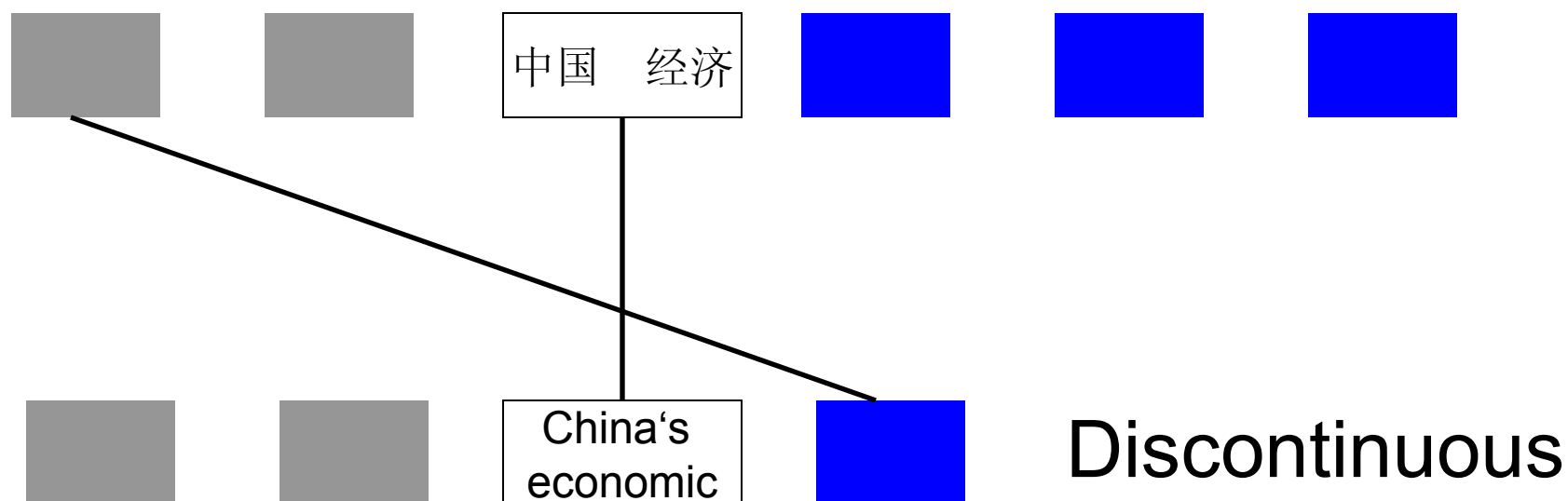
- **MSD**: 对每一对双语短语, 在训练语料库中统计目标端相邻的下一个短语在源语言端是 **Monotone**、**Swap** 和 **Discontinuous** 的概率



Swap

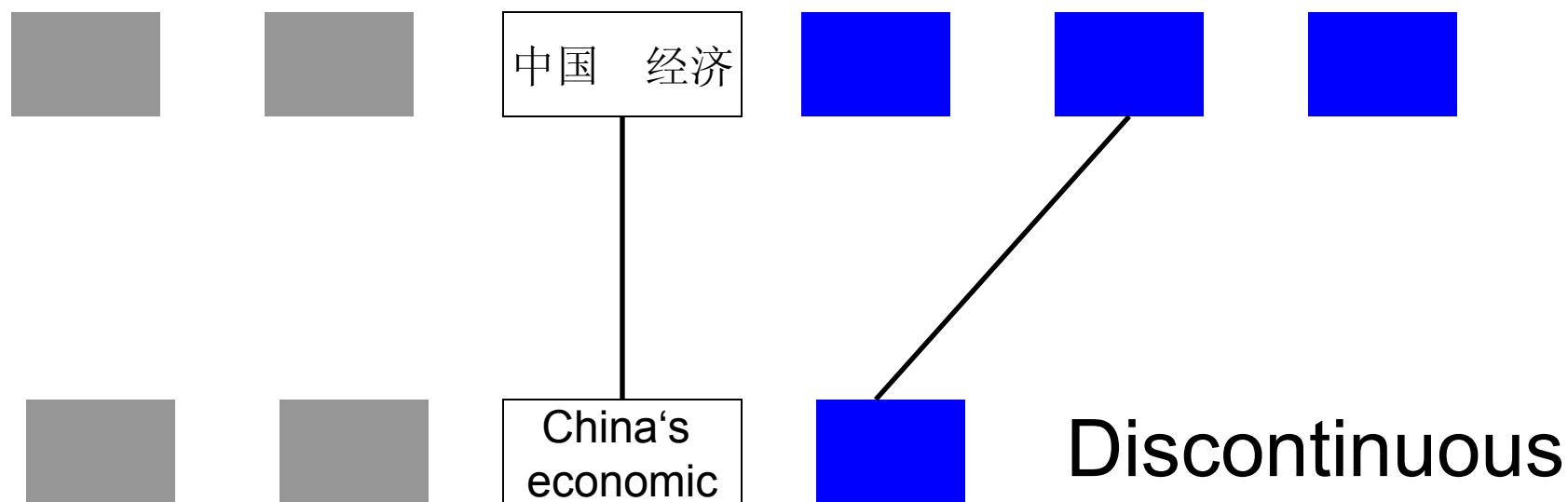
词汇化的调序模型

- **MSD**: 对每一对双语短语, 在训练语料库中统计目标端相邻的下一个短语在源语言端是 **Monotone**、**Swap** 和 **Discontinuous** 的概率



词汇化的调序模型

- **MSD**: 对每一对双语短语, 在训练语料库中统计目标端相邻的下一个短语在源语言端是 **Monotone**、**Swap** 和 **Discontinuous** 的概率



词汇化的调序模型

- **MSD**: 对每一对双语短语, 在训练语料库中统计目标端相邻的下一个短语在源语言端是 **Monotone**、**Swap** 和 **Discontinuous** 的概率

Source	Target	Monotone	Swap	Discontinuous
中国 经济	China's Economic	0.5	0.1	0.4
.....
.....

内容提要

- 对数线性模型
- 最小错误率训练
- 基于对数线性模型的词语对齐
- 基于短语的翻译模型
- 短语模型的解码算法
- 开源机器翻译系统简介
- 总结

解码算法

- 与基于词的模型大体类似，采用自左向右的堆栈搜索算法（**Beam Search**）
- 引入 **Future Cost**（类似于 **A*** 搜索）
- 过程：
 - 查找候选短语
 - 生成 **Future Cost**
 - **Beam Search**
 - 生成 **N-Best List**

查找候选短语 (1)

- 对一个汉语句子，穷举所有可能的短语
- 从 **Bilingual Phrase Table** 中选择所有与之匹配的双语短语（**Translation Options**）

查找候选短语 (2)

```
for(p1=0; p1<ChineseWord.size(); p1++)  
{  
    for(p2=p1; p2<sp.ChineseWord.size(); p2++)  
    {  
        if((p2-p1)>MAX_PHRASE_LEN)  
            break;  
        string phrase = ChineseWord[p1, p2];  
        Search options for phrase from BP table  
    }  
}
```

查找候选短语 (3)

- 为了减小搜索空间，加快搜索速度，对每一个汉语短语，根据以下公式只选取最“好”的 N 个英语翻译（ P 值最高）：

$$P = \sum_i \lambda_i \times \log(p_i(\tilde{e}, \tilde{f}))$$

$p_i(\tilde{e}, \tilde{f})$ 表示短语翻译概率特征

查找候选短语 (4)

- 中国 经济 发展 十分 迅速

<src phrase=" 中国" beg="0" end="0">

<trans lex="China">-0.282683 | -0.242107 | -0.995656 | -0.391329 | -1 | </trans>

<trans lex="China 's">-0.331812 | -0.69002 | -1.76154 | -1.87886 | -2 | </trans>

</src>

<src phrase=" 中国经济" beg="0" end="1">

<trans lex="China 's economic">-0.344841 | -0.916413 | -0.904456 | -2.43623 | -3 | </trans>

<trans lex="China 's economy">-0.241162 | -1.04799 | -1.33977 | -2.98078 | -3 | </trans>

</src>

<src phrase=" 中国经济发展" beg="0" end="2">

<trans lex="China 's economic development">0 | -1.34465 | -0.606135 | -2.8362 | -4 | </trans>

<trans lex="China 's economic development .">0 | -1.34465 | -1.70475 | -6.01209 | -5 | </trans>

</src>

<src phrase=" 经济" beg="1" end="1">

<trans lex="economic">-0.121522 | -0.226393 | -0.943174 | -0.557362 | -1 | </trans>

<trans lex="economy">-0.0626278 | -0.357969 | -1.45799 | -1.10192 | -1 | </trans>

</src>

.....

计算 Future Cost (1)

- 目的：计算“未来搜索”的代价，便于搜索状态的比较
- 方法：动态规划

根据候选短语计算一个句子任意连续位置的最大翻译概率，存入 FutureCostTable 中

计算 Future Cost (2)

for each ChinesePhrase in Translation Option

Compute Max Translation Probability:

$$TP(f_{start}^{\tilde{end}}) = \max \sum_i \lambda_i \log(p_i(\tilde{e}, \tilde{f}))$$

其中 $p_i(\tilde{e}, \tilde{f})$ 是对数线性模型的特征，

包括翻译概率、语言模型概率、英语短语数量等等

FutureCost(start,end) = TP;

for len = 1 to Wordlen

for i=0 to Wordlen-len

for j=i to i+len

double p= FutureCost(i,j) + FutureCost(j+1, i+len)

if (p > FutureCost(i, i+len))

FutureCost(i, i+len) = p;

计算 Future Cost (3) Sample

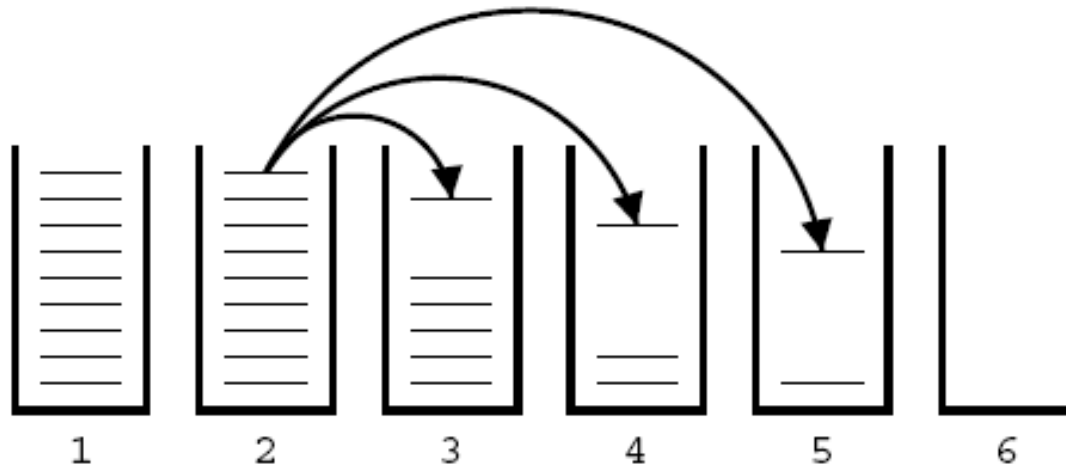
中国0	经济1	发展2	十分3	迅速4
-1.9118	-1.8484	-2.2069	-2.1182	-2.2024
-4.60194				
-4.78698				
	-2.51778			

$$FC(0,1) = FC(0,0) + FC(1,1) = -3.7602$$

取代根据原有短语“中国经济”计算出来的 $FC(0,1) = -4.60194$

Beam Search (1)

- 将概率、英语翻译等信息存储到 Hypothesis 中
- 将 Hypothesis 按照覆盖的源语言单词个数存储到相应的 Stack 中，也就是说，覆盖源语言单词个数相同的 Hypothesis 放到同一个 Stack 中



Beam Search (2) — Hypothesis

Point father;// 指向父亲栈及栈内结点

vector<int> CoveredWord;// 目前覆盖的汉语词

int EndPosOfLastPhrase;// 最近一次翻译的汉语短语的最后单词的位置
// 用于计算扭曲概率

string LastTwoEnglishWord;// 后两个英语词

string CurEnglishTranslation;// 英语翻译

Feature feat;// 特征

double CurProb;// 当前概率

double EstimateProb;// 总的估计概率（当前概率+未来概率）

vector<AddArc> AdditionalArcs;// 存放合并的状态

Bear

中国0



Init_Hyp

中科院计算所

INSTITUTE OF COMPUTING TECHNOLOGY

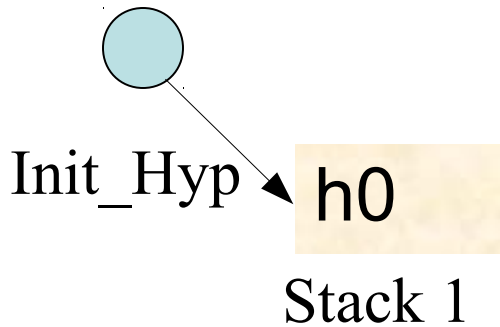
```
<src phrase=" 中国" beg="0" end="0">
<trans lex="China">-0.282683 | -0.242107 | -0.995656 | -0.391329 | -1 | </trans>
<trans lex="China 's">-0.331812 | -0.69002 | -1.76154 | -1.87886 | -2 | </trans>
</src>
<src phrase=" 中国经济" beg="0" end="1">
<trans lex="China 's economic">-0.344841 | -0.916413 | -0.904456 | -2.43623 | -3 | </trans>
<trans lex="China 's economy">-0.241162 | -1.04799 | -1.33977 | -2.98078 | -3 | </trans>
</src>
<src phrase=" 中国经济发展" beg="0" end="2">
<trans lex="China 's economic development">0 | -1.34465 | -0.606135 | -2.8362 | -4 | </trans>
<trans lex="China 's economic development .">0 | -1.34465 | -1.70475 | -6.01209 | -5 | </trans>
</src>
<src phrase=" 经济" beg="1" end="1">
<trans lex="economic">-0.121522 | -0.226393 | -0.943174 | -0.557362 | -1 | </trans>
<trans lex="economy">-0.0626278 | -0.357969 | -1.45799 | -1.10192 | -1 | </trans>
</src>
<src phrase=" 经济发展" beg="1" end="2">
<trans lex="economic development">-0.26748 | -0.654627 | -0.638339 | -0.957337 | -2 |
</trans>
<trans lex="economy development">0 | -0.786202 | -3.16407 | -1.50189 | -2 | </trans>
</src>
<src phrase=" 发展" beg="2" end="2">
<trans lex="development">-0.424533 | -0.428234 | -0.954138 | -0.399975 | -1 | </trans>
<trans lex="development of">-0.450021 | -0.428234 | -2.18543 | -2.34386 | -2 | </trans>
</src>
<src phrase=" 十分" beg="3" end="3">
<trans lex="extremely">-0.364644 | -0.336472 | -0.71295 | -0.704198 | -1 | </trans>
<trans lex="the extremely">0 | -0.336472 | -3.23868 | -2.2988 | -2 | </trans>
</src>
<src phrase=" 迅速" beg="4" end="4">
<trans lex="rapid">-1.38629 | -0.993253 | -1.96944 | -1.64866 | -1 | </trans>
<trans lex="rapidly">-0.287682 | -0.430783 | -0.790786 | -0.693147 | -1 | </trans> </src>
```

Beam Search (3.2) — *Sample*

中国0

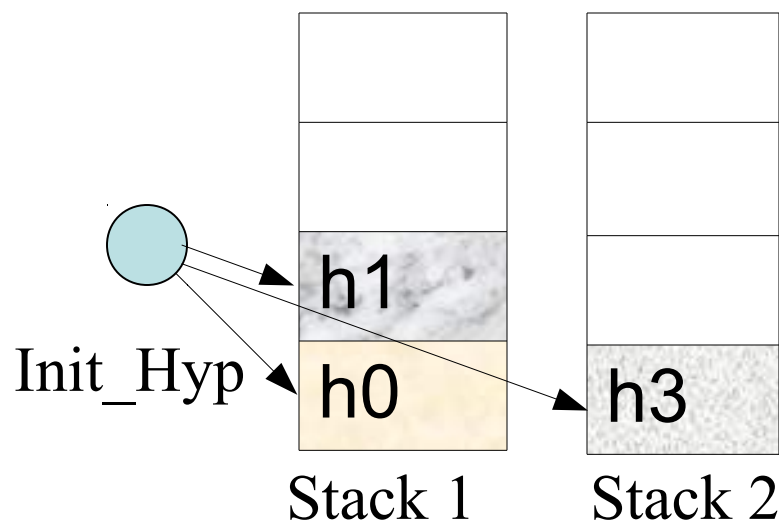
经济

```
Father = -1;  
CoveredWord = 10000;  
EndPosOfLastPhrase = 0;  
LastTwoEnglishWord = China 's;  
CurEnglishTranslation = China 's ;  
  
Feat = -0.331812 | -0.69002 | -1.76154 |  
-1.87886 | -2 | lm(China 's) | d(Mono|<s>);  
CurProb =  $\sum \lambda_i \times \log p_i(\tilde{e}, \tilde{f})$   
EstimateProb = CurProb + FC(1,4);  
  
vector<AddArc> AdditionalArcs;
```



Beam Search (3.3) —— *Sample*

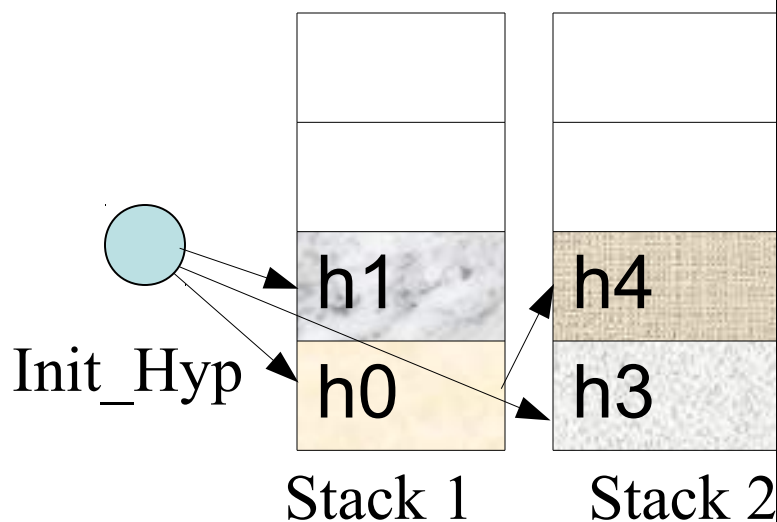
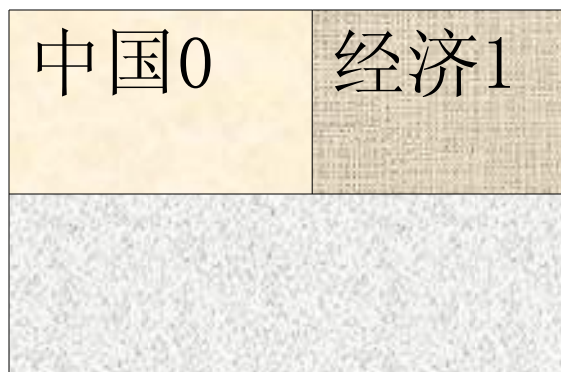
中国0	经济1	发展2	十分3	迅速4



扩展的条件:

1. 待扩展的短语存在于短语表中
2. 待扩展的短语是句子中还未翻译的部分

Beam Search (3.4) — *Sample*

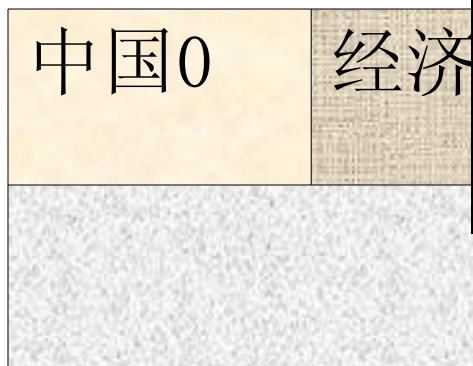


Father = h0;
 CoveredWord = 11000;
 EndPosOfLastPhrase = 1;
 LastTwoEnglishWord = 's economic
 CurEnglishTranslation = economic
 Feat = -0.121522 | -0.226393 | -0.943174
 | -0.557362 | -1 | $\text{lm}(\text{economic}|\text{China 's})$ |
 $\text{d}(\text{Mono}|\text{中国})$

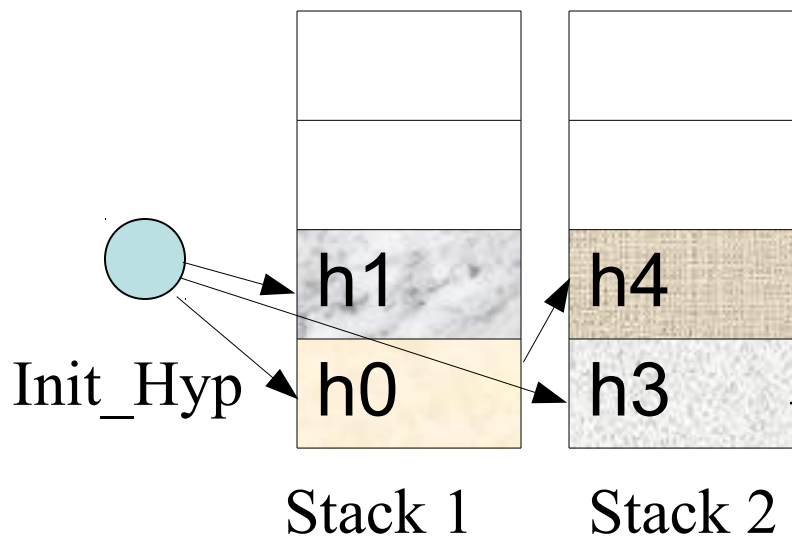
CurProb = Father.CurProb + $\sum \lambda_i \times \log p_i(\tilde{e}, \tilde{f})$
 EstimateProb = CurProb + FC(2,4);

vector<AddArc> AdditionalArcs;

Beam Search



CoveredWord = 11000;
 EndPosOfLastPhrase = 1;
 LastTwoEnglishWord = 's economic



CoveredWord = 11000;
 EndPosOfLastPhrase = 1;
 LastTwoEnglishWord = 's economic

Beam Search (4.1) — *Pruning*

- Recombination – risk free

两个 Hypothesis 满足下面的条件就可以进行 Recombine:

1. 覆盖的汉语词位置相同
2. 最后2个英语词相同 (LM 是3-gram)
3. 最近一次翻译的汉语短语的最后一个词的位置相同

Beam Search (4.2) — *Pruning*

- Histogram pruning

每个栈保留前 N 个最好的 Hypothesis

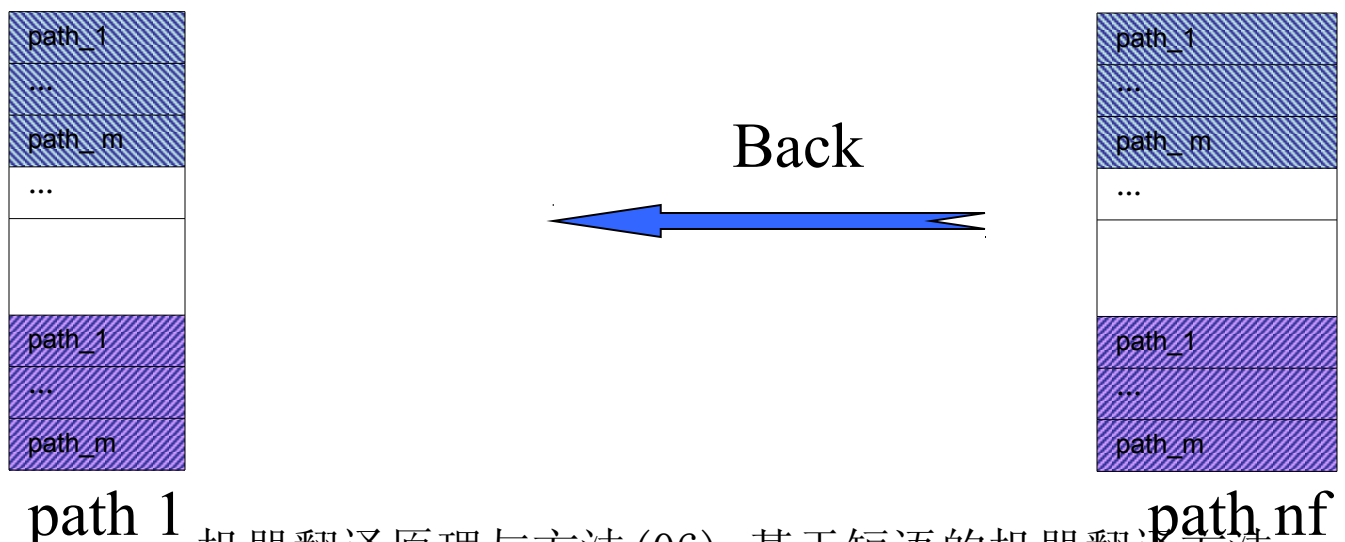
Beam Search (5) — *Algorithm*

```
initialize hypothesisStack[0 .. nf];
create initial hypothesis hyp_init;
add to stack hypothesisStack[0];
for i=0 to nf-1:
  for each hyp in hypothesisStack[i]:
    for each new_hyp that can be derived from hyp:
      nf[new_hyp] = number of foreign words covered by new_hyp;
      add new_hyp to hypothesisStack[nf[new_hyp]];
      prune hypothesisStack[nf[new_hyp]];
```

生成 N-best list (1)

- 从最后一个栈开始，从后向前回溯产生路径
- 每一个假设，都保持 m-best list

生成 N-best list (2) — *Sample*



生成 N-best list (3) — *Algorithm*

```
Initialize PathStack[1, ..., nf];
for i=0 to hp_stack[nf].size()
    select m_best hp from hp_stack[nf][i];
    for each hp in m_best
        PathStack[hp.father].push_back(hp_path);

for i=PathStack.size()-1 to 0
    for each path in PathStack[i]
        hp_father = path.father,
        if(hp_father = -1)
            {
                find a path, and push it to the N-best list;
            }
        create new_path;
        PathStack[hp_father].push_back(new_path);
```

内容提要

- 对数线性模型
- 最小错误率训练
- 基于对数线性模型的词语对齐
- 基于短语的翻译模型
- 短语模型的解码算法
- 开源机器翻译系统简介
- 总结

基于短语的统计机器翻译工具

- 法老 Pharaoh
 - 最早的著名的基于短语的统计机器翻译工具，由 Philip Kohnn 开发
 - 性能远高于基于词的系统
 - 部分开放源代码（核心解码器没有开源）
- 摩西 Moses
 - 最新的开源统计机器翻译工具，完全开源
 - 由国际上很多单位共同开发，性能和效率都很高
 - 基于短语模型和 **Factored Translation Model**
 - 加入了层次短语模型

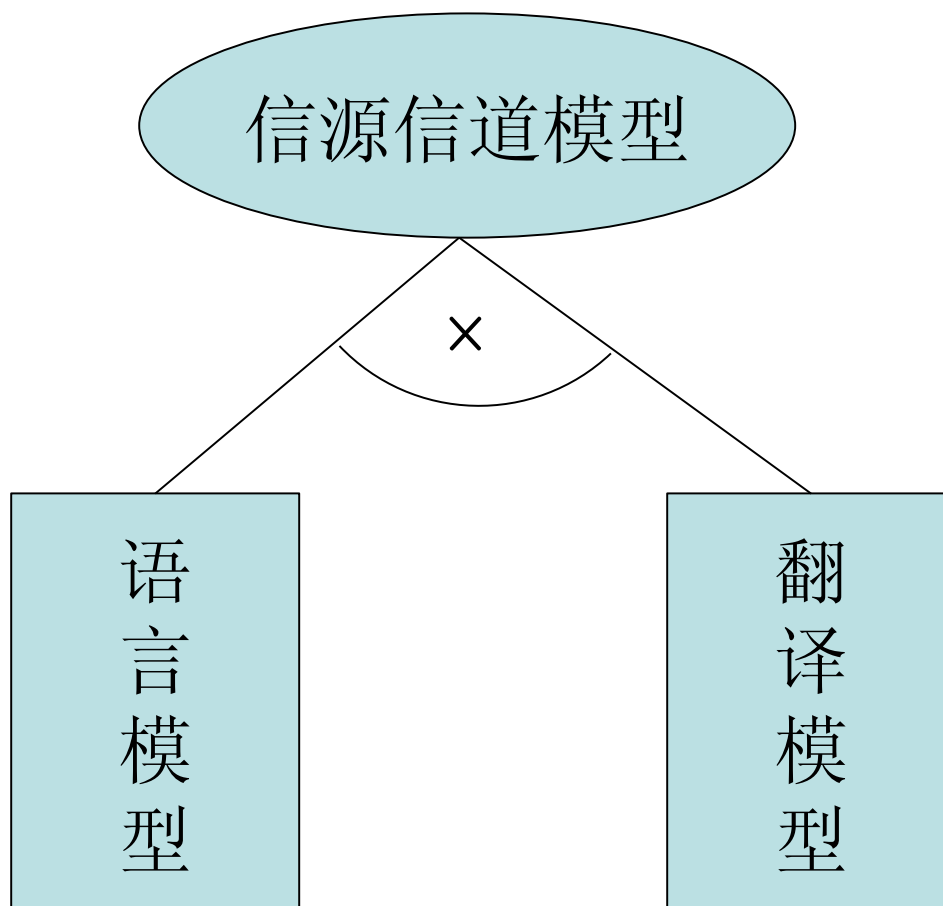
基于短语的统计机器翻译工具

- Joshua
 - 约翰霍普金斯大学李志飞开发
 - Java 语言
 - 支持层次短语模型
- 丝路
 - 由中科院计算所等国内五家单位共同开发
 - 短语模型
 - 完善的中文文档，易于使用
- NiuTrans
 - 东北大学 NLP 研究组开发

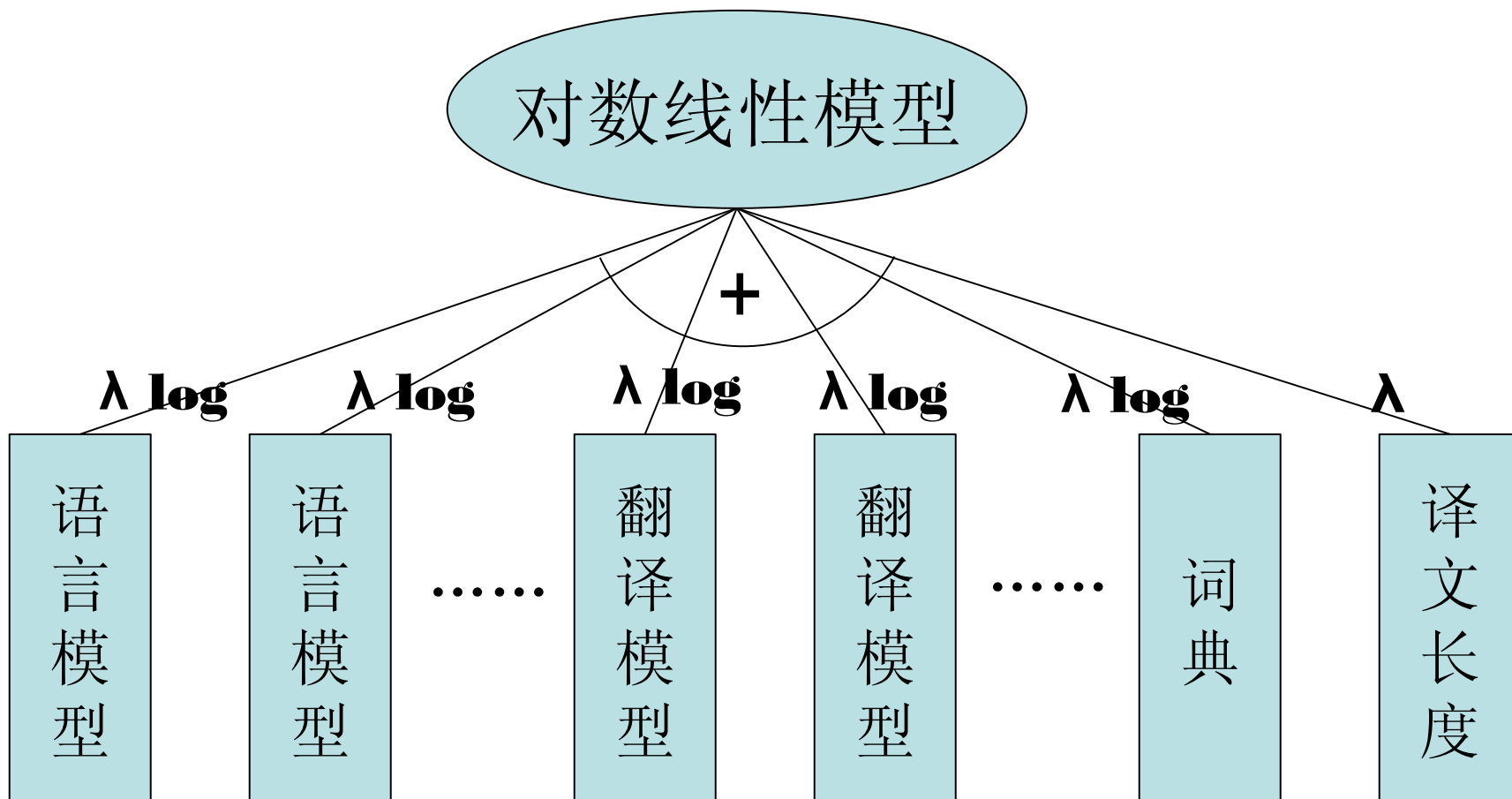
内容提要

- 对数线性模型
- 最小错误率训练
- 基于对数线性模型的词语对齐
- 基于短语的翻译模型
- 短语模型的解码算法
- 开源机器翻译系统简介
- 总结

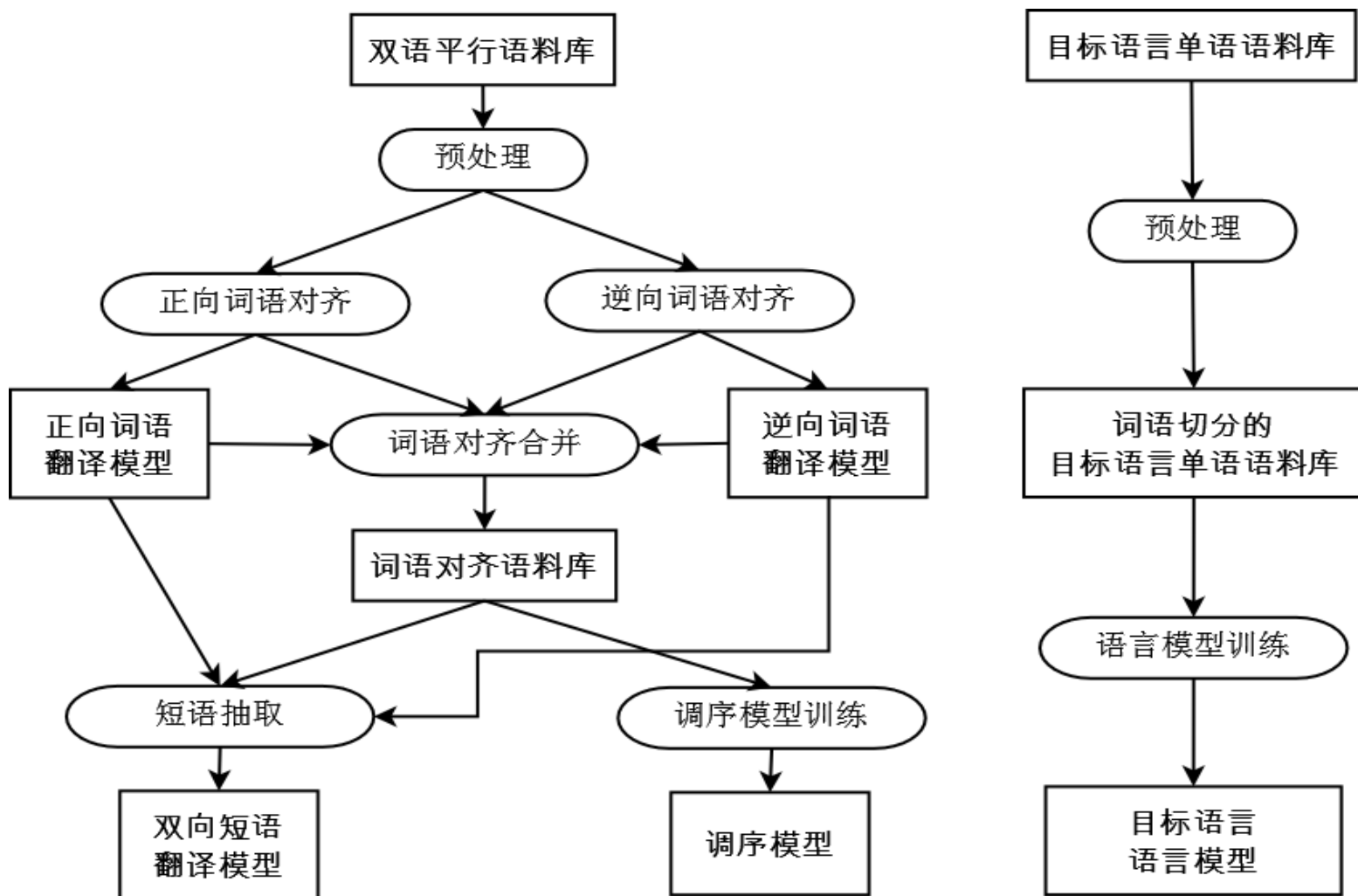
信源信道模型



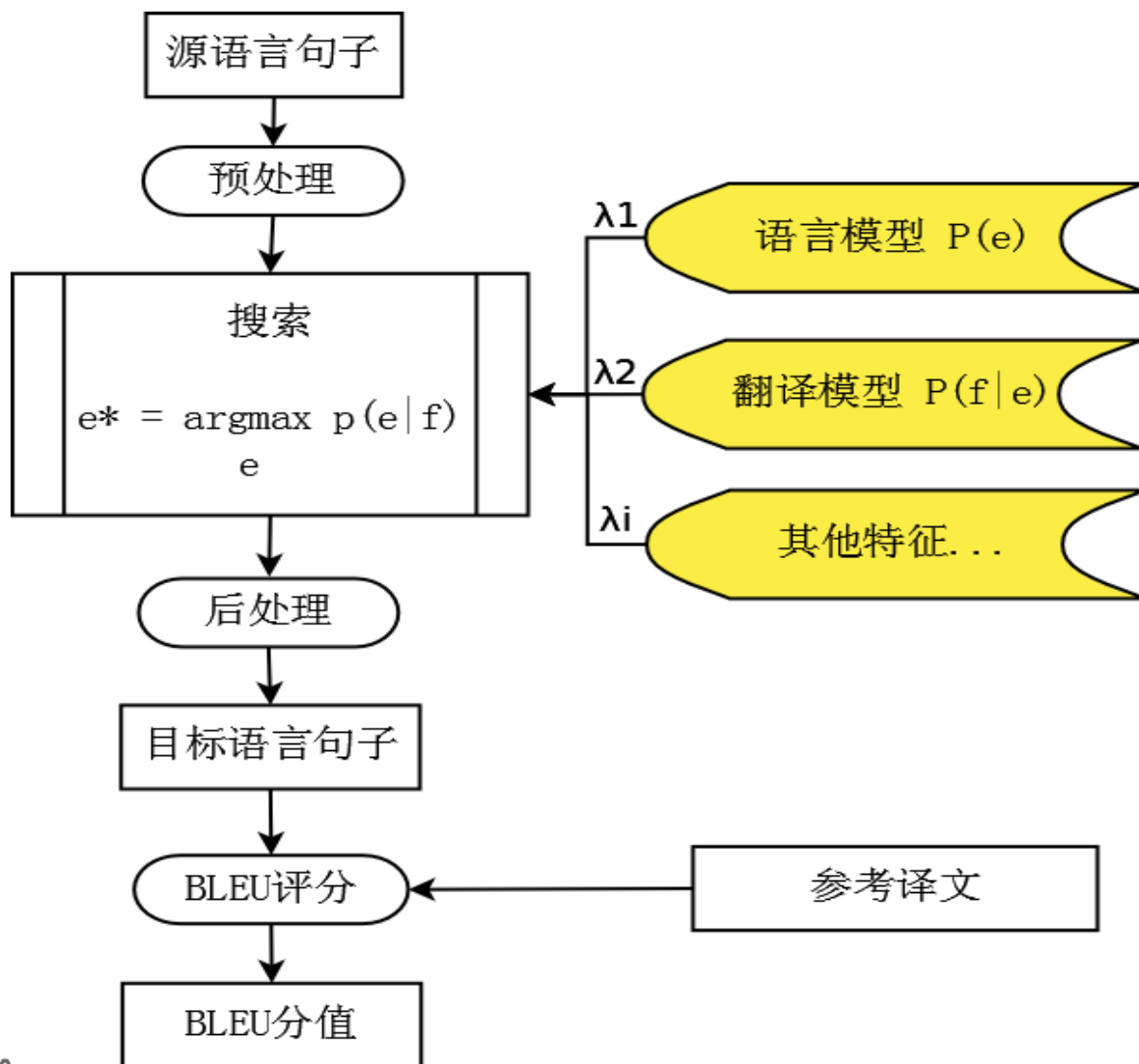
对数线性模型



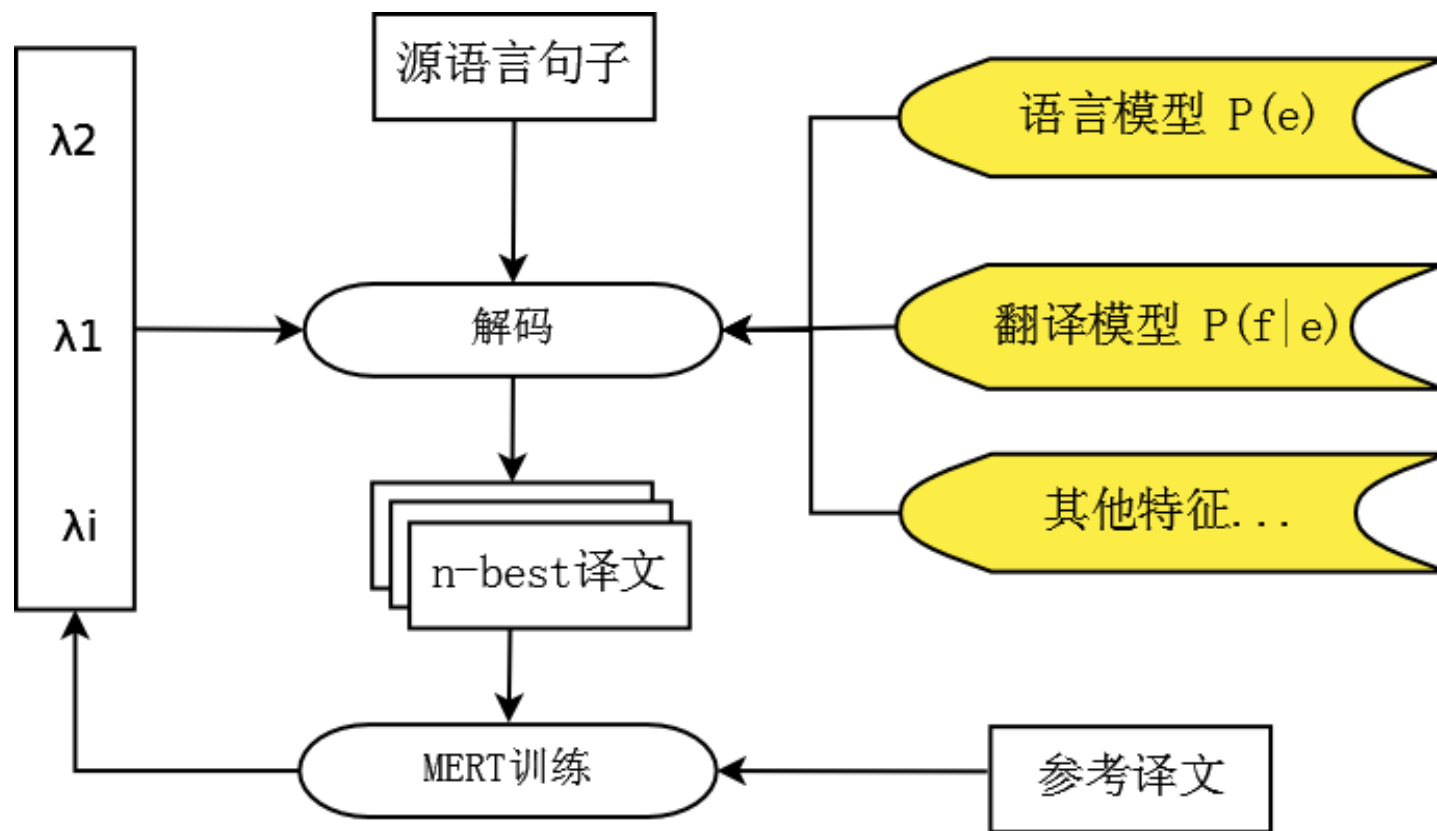
基于短语的统计机器翻译训练



基于短语的统计机器翻译解码



基于短语的统计机器翻译调参



基于短语的翻译模型小结

- 基于短语的翻译模型是目前最成功的翻译模型
- 形式上类似于基于实例的翻译方法，区别在于引入了统计模型，性能上远远超过了传统的基于实例的方法
- 讨论：实验证明，基于短语的翻译模型的性能与词语对齐错误率的关联并不明显，也就是说，降低词语对齐的评价错误率，并不能使得基于短语模型的统计机器翻译系统性能明显提高。这是什么原因呢？