分类号 TP3 密级

UDC 编号

中国科学院研究生院
博士学位论文

树到串统计翻译模型研究

刘洋

指导教师 林守勋 研究员

中国科学院计算技术研究所

申请学位级别 工学博士 学科专业名称 计算机应用技术

论文提交日期 2007 年 4 月 论文答辩日期 2007 年 6 月

培养单位 中国科学院计算技术研究所

学位授予单位 中国科学院研究生院

答辩委员会主席

声　　明

我声明本论文是我本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，本论文中不包含其他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

作者签名：　　日期：

论文版权使用授权书

本人授权中国科学院计算技术研究所可以保留并向国家有关部门或机构送交本论文的复印件和电子文档，允许本论文被查阅和借阅，可以将本论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编本论文。

（保密论文在解密后适用本授权书。）

作者签名：　　导师签名：　　日期：

摘要

近十年来，统计机器翻译取得了很大的成功。基于短语的翻译模型超越了最初的基于词的翻译模型，在近期的机器翻译评测中屡次取得领先成绩，成为统计机器翻译的主流技术。然而，基于短语的模型的一大缺点是难以处理短语间的重排序。因此，能将句法信息引入翻译的基于句法的翻译模型成为当前的研究热点。

本文重点研究了统计机器翻译中的两个关键问题：词语对齐和翻译模型。

词语对齐对统计机器翻译而言至关重要，因为经过词语对齐的语料是极有价值的翻译知识源。本文为词语对齐提出对数线性模型框架。在此框架下，所有的知识源被视作依赖于源语言句子、目标语言句子以及可能的其他变量的特征函数。对数线性模型使统计对齐模型易于扩展，方便加入更多的语言学信息，从而能同时处理与具体语言相关和不相关的语言现象。本文讨论了框架的形式化定义、特征函数、最小错误率训练、搜索算法以及 n-best 列表生成等问题。我们在三个词语对齐评测的数据集（包含五个语言对）上对词语对齐的对数线性模型进行评价。实验表明，对数线性模型超过了绝大多数参加评测的系统。

翻译模型设计是统计机器翻译的核心问题。本文提出三个基于句法的树到串翻译模型：

1. 嵌入句法树的基于短语的翻译模型，简称模型 1。此模型在传统的基于短语的模型的基础上以隐变量的方式嵌入句法树，从而可以利用句法信息指导短语的切分、重排序和翻译。模型 1 只使用句法双语短语，搜索空间比传统模型小。我们的主要创新点是提出了树节点重排序，实现了利用句法信息指导短语重排序。

2. 基于树到串对齐模板的翻译模型，简称模型 2。此模型在模型 1 的基础上提出了树到串对齐模板。树到串对齐模板描述了源语言句法树和目标语言串之间的对齐关系。它既能生成终结符又能生成非终结符，既能执行局部的重排序又能执行全局的重排序。

3. 融入森林到串规则的树到串翻译模型，简称模型 3。此模型对模型 2 进行扩充，引入森林到串翻译规则，通过描述森林和串之间的对齐关系来捕获非句法短语，使表达能力得到进一步提升。为了将森林到串翻译规则融入到树到串翻译模型中，我们引入辅助规则来提供泛化层。

我们将这三个基于句法的翻译模型与国际学术界最常用的基于短语的翻译系统 Pharaoh 做对比。在 2005 年 NIST 汉译英机器翻译评测测试集上，模型 1 的翻译性能接近基准系统，模型 2 和模型 3 均明显超过了基准系统。

关键词：统计翻译模型；词语对齐；树节点重排序；树到串对齐模板；森林到串翻译规则

Research on Tree-to-String Statistical Translation Models

Yang Liu (Computer Applied Technology)

Directed by Shouxun Lin

Statistical machine translation (SMT) has shown considerable success over the past decade. Phrase-based translation models, which go beyond the original word-based models, have been suggested to be the state of the art by recent empirical evaluations. However, one major problem with phrase-based models is their incapability of robust phrase-level reordering. As a result, syntax-based models that incorporate syntax into translation are drawing increasing interests from SMT researchers.

In this thesis, we put emphasis on two key subfields of statistical machine translation: word alignment and translation model.

Word alignment plays a crucial role in statistical machine translation as word-aligned data have been proven to be an excellent source of translation-related knowledge. This thesis describes a framework for word alignment based on log-linear models. Within this framework, all knowledge sources are treated as feature functions, which depend on the source language sentence, the target language sentence, and possible additional variables. Log-linear models allow statistical alignment models to be easily extended by incorporating linguistic information so as to handle both language-independent and language-dependent phenomena. We consider the formalism of the framework, feature functions, minimum error rate training, search algorithm, and *n*-best list generation. We evaluate the framework on three word alignment shared tasks for five language pairs with various divergences and resources. Experiments show that our log-linear models outperform most of the participating systems for all the three shared tasks.

The design of translation model is the central problem in statistical machine translation. In this thesis, we present three syntax-informed tree-to-string translation models with varying expressive power:

1. Tree-embedded phrase-based translation model, i.e. Model 1. This model is phrase-based, but with parse tree incorporated as a hidden variable. Consequently, syntactic information can be utilized to facilitate phrase segmentation, phrase reordering, and phrase translation. By making use of only syntactic bilingual phrases, the search space is dramatically reduced in contrast to conventional models. Our major contribution is the proposition of Tree Node Reordering (TNR) to enable linguistically motivated reordering.

2. Translation model based on tree-to-string alignment template, i.e. Model 2. Departing from phrases, we propose tree-to-string alignment template (TAT) on the basis of Model 1. A TAT describes the alignment between a source tree and a target string. It is capable of generating both terminals and nonterminals and performing both local and global reorderings.
3. Tree-to-string translation model augmented with forest-to-string rules, i.e. Model 3. This model evolves from Model 2 by including forest-to-string translation rules so as to gain more expressive power. A forest-to-string rule is capable of capturing non-syntactic phrase pairs by describing the correspondence between multiple parse trees and a string. To integrate these rules into tree-to-string models, auxiliary rules are introduced to provide a generalization level.

We compared our three syntax-based translation models with Pharaoh, a widely-used freely available phrase-based system. On the 2005 NIST Chinese-to-English machine translation evaluation test set, Model 1 achieves comparable result with the Pharaoh while both Model 2 and Model 3 outperform the baseline system significantly.

Keywords: statistical translation model; word alignment; tree node reordering; tree-to-string alignment template; forest-to-string translation rule

目 录

第一章 引言

1.1 机器翻译

机器翻译是指利用计算机将一种自然语言翻译成另一种自然语言。

1949年，美国洛克菲勒基金会自然科学部门的负责人Warren Weaver发表了一份以《翻译》为题的备忘录，正式提出了机器翻译问题。五十多年来，机器翻译的研究大大加深了人们对于语言、知识和智能等问题的了解，促进了相关学科的发展。虽然机器翻译的现状离人们的期望和市场的需求都还有相当大的距离，研究人员对机器翻译研究的热情依然很高。一方面，由于科学技术的发展日新月异，各民族之间的文化交流越来越频繁，语言障碍问题相对说来也越来越严重，机器翻译的需求和应用前景十分巨大。另一方面，仅从学术角度来看，机器翻译也是一个非常有意义的研究课题，其复杂性、挑战性和高难度对研究人员而言充满了魅力。对全自动高质量机器翻译的不懈追求，正是计算语言学研究的终极目标之一和不竭动力的源泉。

在 20 世纪 90 年代以前，机器翻译的主流方法一直是基于规则的方法，也称为传统的机器翻译方法。建造一个实用的基于规则的机器翻译系统，往往需要建立各类知识库，描述源语言和目标语言的词法、句法以及语义知识，甚至还需描述和语言知识无关的世界知识。然而，这些知识库的描述和建立是极其困难的。首先，知识库必须由许多训练有素的专家创建和维护。更糟糕的是，随着知识库的规模不断扩大，如何保证新引入的知识不与旧知识相互矛盾也成为难题。因此，知识获取成为传统的机器翻译方法的瓶颈。

在 20 世纪 80 年代中后期，一些研究人员提出基于语料库的机器翻译方法。与传统方法不同的是，基于语料库的方法不对语言进行深层次的分析，而是大规模收集互为译文的双语语料并基于这些语料进行翻译。基于语料库的方法有两个分枝：一种称为基于实例的机器翻译方法，认为可以通过在双语语料库中查找最为相似的翻译实例的方法来获得语言的翻译；另一种称为基于统计的机器翻译方法，即统计机器翻译，主张对翻译过程建立数学模型，利用双语语料库估计模型参数，进而根据模型及经过估计的参数执行翻译。

统计机器翻译是当前机器翻译乃至自然语言处理领域的研究热点。从近几年国际机器翻译评测的成绩来看，统计机器翻译系统的翻译水平已经明显超过基于规则和基于实例的机器翻译系统，成为机器翻译的主流技术。

1.2 统计机器翻译

一个源语言句子通常有多种翻译方式。在统计机器翻译中，对于一个源语言句子 $f_1^J = f_1,...,f_j,...,f_J$，任何目标语言句子 $e_1^I = e_1,...,e_i,...,e_I$ 都被视作是其可能的译文，只不过可能性有大有小。对每一个源语言和目标语言句子对 $\left(f_1^J, e_1^I\right)$，翻译概率 $\Pr\left(e_1^I \mid f_1^J\right)$ 表示 f_1^J 被翻译成 e_1^I 的可能性。 因此，翻译问题就转化为已知源语言句子 f_1^J 求翻译概率最大的目标语言句子 $\hat{e}_1^I$：

$$\hat{e}_1^I = \arg\max_{e_1^I} \Pr\left(e_1^I \mid f_1^J\right) \qquad \text{公式 1.1}$$

假如翻译概率 $\Pr\left(e_1^I \mid f_1^J\right)$ 是可知的，那么机器翻译的问题自然迎刃而解。不幸的是，虽然自然语言的词汇量是有限的，但是能生成的句子的数量却几乎是无限的，枚举所有源语言和目标语言句子对并赋概率是不现实的。

由于直接估计 $\Pr\left(e_1^I \mid f_1^J\right)$ 是不可能的，统计机器翻译的研究人员通常对翻译过程建立数学模型，将 $\Pr\left(e_1^I \mid f_1^J\right)$ 分解成若干个概率的乘积，将问题转化为估计这些概率。这个数学模型通常被称为翻译模型（translation model）。估计这些分解后的概率通常被称为参数估计（parameter estimation）或训练（training）。在统计机器翻译中，研究人员通常使用双语语料库来为翻译模型做参数估计。已知翻译模型和经过估计的模型参数，就可以执行翻译了，这在统计机器翻译中通常称为搜索（search）或解码（decoding），用数学方式来表述即公式 1.1。

因此，统计机器翻译的本质是对翻译过程建立数学模型，在真实世界数据上自动学习模型参数，从而利用这些参数执行翻译。统计机器翻译的三个基本问题是：

1. 建模，即为翻译过程建立数学模型。
2. 训练，即根据训练数据自动学习模型参数。
3. 搜索，即利用学习到的模型参数执行翻译。

我们认为建立翻译模型是统计机器翻译中的核心问题。建模直接决定了训练和搜索算法的设计，从根本上决定了系统的翻译性能。因此，在本节我们将以翻译模型为主线介绍统计机器翻译十几年来的发展历程。

1.2.1 基于词的翻译模型

统计机器翻译的起源可追溯到上世纪 50 年代，由于遭到一些语言学家的强烈批判，这种方法很快就被放弃了。一些人认为，统计方法在 50 年代遭到放弃

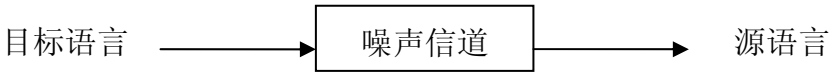

图 1.1：噪声信道

的真正原因是缺乏高性能计算机和联机语料，因为统计方法需要大量的计算和大规模的语料。现在，计算机的运算速度和存储器容量有了大幅度的提高，早期大型计算机才能胜任的工作现在甚至个人计算机就能够完成。同时，大量的联机语料也可以很容易地获取。在这样的背景下，IBM 公司在 20 世纪 90 年代提出了最早的统计机器翻译模型，开创了统计机器翻译的新时代。

[Brown 1993]将翻译视作一个噪声信道问题。如图 1.1 所示，目标语言由于经过了一个噪声信道而发生了扭曲变形，从而在信道的另一端呈现为源语言。翻译问题实际上就是如何根据观察得到的源语言恢复最可能的目标语言的问题。

根据贝叶斯公式，翻译概率 $\Pr\left(e_1^I \mid f_1^J\right)$ 可以写为：

$$\Pr\left(e_1^I \mid f_1^J\right)=\frac{\Pr\left(e_1^I\right)\Pr\left(f_1^J \mid e_1^I\right)}{\Pr\left(f_1^J\right)} \qquad \text{公式 1.2}$$

相应地，搜索公式为：

$$\begin{aligned}\hat{e}_1^I &= \arg\max_{e_1^I} \Pr\left(e_1^I \mid f_1^J\right) \\ &= \arg\max_{e_1^I} \frac{\Pr\left(e_1^I\right)\Pr\left(f_1^J \mid e_1^I\right)}{\Pr\left(f_1^J\right)} \\ &= \arg\max_{e_1^I} \Pr\left(e_1^I\right)\Pr\left(f_1^J \mid e_1^I\right)\end{aligned} \qquad \text{公式 1.3}$$

其中，$\Pr\left(e_1^I\right)$ 称为语言模型，$\Pr\left(f_1^J \mid e_1^I\right)$ 被[Brown 1993]称为翻译模型。由于语言模型在语音识别领域已经得到充分的研究，[Brown 1993]将重点放在翻译模型的设计上，他们提出五个翻译模型，通常被称为 IBM 模型。由于 IBM 模型的基本翻译单位是词语，因此也称为基于词的翻译模型。

基于词的翻译模型被提出后，吸引了许多研究人员的兴趣。在此，我们分建

模、训练和搜索三个方面介绍基于词的翻译模型在 20 世纪 90 年代的研究进展。

在建模方面，理论改进并不大。由于基于词的模型的一大缺陷在于没有考虑上下文信息，[Berger 1996]根据最大熵原理提出了依赖上下文的词模型。

在训练方面，由于[Brown 1993]已经给出十分完备的数学描述，之后没有引人注目的理论改进，主要方法还是 EM 算法。由于基于词的模型十分复杂，IBM 公司也没有公布源代码，研究人员想重新实现 IBM 模型的参数训练十分困难。1999 年夏季，美国约翰霍普金斯大学（Johns Hopkins University，简称 JHU）召开第一届统计机器翻译研讨班，旨在开发一个统计机器翻译工具包并使之能向所有研究人员开放。这个统计机器翻译工具包被称为 EGYPT，其中最重要的一个工具叫 GIZA，它的主要功能是实现 IBM 翻译模型的参数训练。当时，GIZA 只实现了前 3 个模型。研讨班结束后，GIZA 的主要开发者 Franz J. Och 继续研究，开发出 GIZA++，实现了 IBM 全部 5 个翻译模型以及 HMM 模型的参数训练。GIZA++现在已经成为统计机器翻译领域最常用的工具。需要指出，当前的统计机器翻译研究人员使用 GIZA++的主要目的是为了得到双语语料库的词语对齐，而对 IBM 模型参数本身已较少使用。

在搜索方面，由于[Brown 1993]并没有讨论这个问题，研究人员在这方面的理论探索也最多。比较突出的工作有：[Wang 1997]为 IBM 模型 2 提出栈搜索算法，[Knight 1999]讨论了基于词的翻译模型的搜索复杂度，[Germann 2001]提出了快速算法和最优算法。其中，[Germann 2001]获得了 ACL 2001 的最佳论文奖，根据此论文及后续工作开发的解码器 ISI Rewrite Decoder 是最著名的、可公开免费使用的基于词的统计机器翻译系统。

基于词的翻译模型是最早的翻译模型，其数学描述十分严密。但是，该模型的缺点也十分明显：

1. 翻译的基本单位是词，没有考虑上下文信息。
2. 重排序（reordering）功能很弱，不论是在局部（词）还是在全局（短语、子句）。
3. 模型的复杂度过高，给参数训练和解码带来极大的难度。事实上，最复杂的 IBM 模型 5 很少被使用。

几乎也就是在 2001 年，学术界对于基于词的翻译模型的研究进入了终止期，而基于短语的翻译模型则成为新的研究热点。

1.2.2 基于短语的翻译模型

基于短语的翻译模型的基本翻译单位是短语。这里的短语并不是语言学意义

上的短语，而只是连续的词串。基于短语的翻译模型有三个基本问题：

1. 短语划分，即将一个句子划分成短语。
2. 短语重排序，即对目标语言短语的顺序进行重新排列。
3. 短语翻译，即将源语言短语翻译成目标语言短语。

[Wang 1998]提出的基于结构的翻译模型是最早的基于短语的模型。该模型首先对源语言和目标语言的短语进行粗对齐（rough alignment），然后对短语内部的单词进行细对齐（detailed alignment）。粗对齐模型类似于 IBM 模型 2，细对齐模型类似于 IBM 模型 4。他们同样采用 EM 算法进行参数估计。基于结构的翻译模型深受 IBM 模型的影响，无论建模方式和参数估计都有类似之处，特别是对短语的切分和对齐也建立了子模型。因此，此模型相当复杂，参数估计和搜索的复杂度都很高。

[Och 1999]提出了对齐模板，实际上就是将短语中的词用词类来代替以获得泛化能力。这是特殊的短语翻译方法，除 Och 以外的研究人员几乎都对短语采用直接翻译方法，即直接将双语短语中的目标语言短语作为源语言短语的译文。与[Wang 1998]相比，基于对齐模板的模型对短语划分和短语重排序做了很大的简化：

1. 采用唯一的短语划分。
2. 短语重排序只依赖于前一个短语的位置。

基于对齐模板的模型不再将词语对齐作为隐变量，该模型本身无法对双语语料库做词语对齐，因此所使用的训练数据必须是经过词语对齐的双语语料库。[Och 1999]对参数估计的方式做出大胆的改进，摒弃了传统的基于 EM 的估计方法，而采用更加简单的基于相对频度的估计方法，从而极大地降低了参数估计的复杂度。这是一个重大变化，此后的许多翻译模型都采用类似的方式，都不具备生成词语对齐的能力，而是直接使用经过词语对齐的双语语料库进行参数估计。这也是为什么 GIZA++目前被如此广泛使用的主要原因。

[Och 1999]采用了柱搜索（beam search）算法，而不是传统的 A*算法和栈算法。柱搜索算法的优点在于能够利用各种剪枝策略来平衡效率和准确度。此后的许多统计机器翻译系统都采用这种算法。

2002 年，统计机器翻译领域发生了两件大事。

第一件大事是[Och 2002a]将对数线性模型引入统计机器翻译。对数线性模型对 $\Pr\left(e_1^I \mid f_1^J\right)$ 直接建模，将各种知识源视作特征函数：

$$\begin{aligned}\Pr\left(e_1^I \mid f_1^J\right) &= p_{\lambda_1^M}\left(e_1^I \mid f_1^J\right) \\ &= \frac{\exp\left[\sum_{m=1}^{M}\lambda_m h_m\left(e_1^I, f_1^J\right)\right]}{\sum_{\tilde{e}_1^I}\exp\left[\sum_{m=1}^{M}\lambda_m h_m\left(\tilde{e}_1^I, f_1^J\right)\right]}\end{aligned} \quad \text{公式 1.4}$$

相应地，搜索公式为：

$$\begin{aligned}\hat{e}_1^I &= \arg\max_{e_1^I}\left\{\Pr\left(e_1^I \mid f_1^J\right)\right\} \\ &= \arg\max_{e_1^I}\left\{\sum_{m=1}^{M}\lambda_m h_m\left(e_1^I, f_1^J\right)\right\}\end{aligned} \quad \text{公式 1.5}$$

对数线性模型可以包含噪声信道模型，其主要优点在于可以很容易地整合各种知识源并自动调节知识源之间的权重[Och 2003a]。Och 的这篇 ACL 2002 最佳论文对统计机器翻译的影响极大，现在的统计机器翻译系统几乎全部是采用对数线性模型框架。

第二件大事是 NIST 机器翻译评测开始举办。NIST 机器翻译评测是由 DAPRA(Defense Advanced Research Projects Agency)资助、NIST（National Institute of Standards and Technology）组织的影响力最大的国际性机器翻译评测。NIST 评测既采用人工评价，也采用自动评价。NIST 的正式自动评价指标是[Papineni 2002]提出的 BLEU。NIST 机器翻译评测吸引了全世界许多大学、公司和科研机构参加，对机器翻译的发展起到了难以估量的推动作用，主要表现在：

1. 评测推动研究。参评单位都很看重评测成绩，都希望通过不断改进自己的系统来获得更好的成绩，从而推动了整个机器翻译研究的发展。
2. 为各种机器翻译方法提供了公平的比较平台。无论是基于规则的方法、基于实例的方法还是基于统计的方法，都可以进行公平的对比。
3. NIST 测试数据成为实验标准。研究人员在撰写学术论文时大多采用 NIST 评测数据，这样就很容易与其他工作进行比较，从而更容易获得学术界的认可。研究人员使用 NIST 测试集和自动评测工具可以十分方便快捷地得知算法改进的效果，极大缩短了系统开发周期。

[Marcu 2002]提出了一种基于短语的联合概率模型，但影响并不大。

[Koehn 2003]在[Och 1999]的基础上舍弃对齐模板回归短语，并且提出词汇化权重（lexical weighting），丰富了基于短语的翻译模型的参数估计方式。由 Philipp Koehn 开发的 Pharaoh 是影响最大的、可公开免费使用的基于短语的统计机器翻译系统。

至此，基于短语的翻译模型基本定型。对于建模的三个基本问题，短语划分和短语翻译都得到比较好的解决，而短语重排序则成为瓶颈。因此，研究人员在

短语重排序问题上深入做了许多工作。[Och 2002b]提出对跳转长度进行惩罚。[Zens 2004]研究了 IBM 限制和 ITG 限制。[Tillmann 2005]将双语短语定义为块，抽取左、中、右三种方位（orientation）。[Xiong 2006]提出基于最大熵的短语重排序模型。[Al-Onaizan 2006]提出 outbound、inbound 和 pairwise 三种扭曲模型。

由于这些工作都没有利用句法信息，一些研究人员开始尝试将句法信息引入到基于短语的翻译模型中。[Xia 2004]和[Collins 2005]都利用句法信息对源语言句子做前处理，然后用基于短语的系统翻译。[Och 2004a]则采用 Reranking 的方法利用句法信息对基于短语的系统的翻译结果做后处理。他们都尝试利用句法信息，但都作为改进基于短语的系统的辅助手段，也都没有获得显著的效果。

[Och 2004a]是对 2003 年第二届 JHU 统计机器翻译研讨班的研究工作的总结。在这届研讨班上，Och 组织研究人员在基于对齐模板的翻译系统上尝试了大量的特征，但收效甚微。Och 宣称句法信息对统计机器翻译的作用不大，引发了旷日持久的争论。

[Quirk 2006a]对基于短语的模型的优缺点做了较好的总结。

优点：

1. 擅长翻译习惯用语等非组合性短语。
2. 擅长局部重排序。
3. 包含一定的上下文信息。

缺点：

1. 只允许完全子串匹配。
2. 不允许非连续短语。
3. 不擅长全局重排序。
4. 概率估计存在缺陷。
5. 对短语划分的假设存在问题。

从 2002 年到 2005 年，由 Franz J. Och 开发的基于对齐模板的统计机器翻译系统连续在 NIST 评测中独占鳌头，而其他基于短语的系统也都排在前列，标志着基于短语的翻译模型已成为统计机器翻译的主流。

然而在 2005 年，蛰伏多年的基于句法的翻译模型开始出现转机，成为当前统计机器翻译研究的热点。

1.2.3 基于句法的翻译模型

[Wu 1997]提出反向转录语法（Inversion Transduction Grammar，简称 ITG），将翻译过程视作利用同步语法对源语言和目标语言句子做双语句法分析（bilingual parsing）。这是第一个将同步语法引入统计机器翻译的工作。

基于类似的思想，[Alshawi 2000]将翻译视作利用中心转录机（head transducer）同步生成源语言和目标语言的依存树的过程。

严格地讲，同步语法不是翻译模型，因为它实际上是对双语生成建模，而不是对翻译过程建模。[Yamada 2001]提出了第一个真正意义上的基于句法的树到串的翻译模型。该模型是一个噪声信道模型，输入是一棵句法树，输出是一个句子。在翻译过程中，他们对句法树定义了几种变换操作。需要强调的是，他们首次在开发基于句法的系统时利用了短语。

[Melamed 2003]提出多文本语法（Multitext Grammar，简称为 MTG）进行翻译。多文本语法比同步上下文无关语法更通用，能够处理更复杂的情况。在 2005 年夏季召开的第三届 JHU 统计机器翻译研讨班上，以 Melamed 为首的研究人员试图将 MTG 实用化，他们推出了工具包 GenPar。但是，到目前为止，这个工作还没有在实际效果上取得引人注目的进展。

这些早期的工作都没有取得良好的实际效果，至少在机器翻译评测中的成绩与基于短语的系统相差甚远。一个重要的原因是从大规模训练数据中学习同步语法或者树转换规则十分困难。[Ding 2005]认为有几个原因：

1. 同步语法和树转换规则处理非同构性（non-isomorphism）能力受到限制。在某种层次上，源语言和目标语言句子的同步推导必须存在。
2. 无论是从大规模数据训练同步语法和树转换规则，还是利用它们执行搜索，计算复杂度都非常高。
3. 训练数据的噪音使问题更严重。在这一点上，基于句法的模型受到的影响要比基于短语的模型大的多。

[Ding 2005]利用概率化同步依存插入语法（Probabilistic Synchronous Dependency Insertion Grammar，简称 PSDIG）和非同构性随机树到树转录机（stochastic tree-to-tree transducer）构建了一个基于句法的统计机器翻译系统。他们利用句法分析器对双语做短语结构分析，再转换成依存树，然后从中抽取语法和转录机。尽管这个系统在实验效果上超过了 IBM 模型 4，但是没有进一步和主流的基于短语的模型做比较。

[Quirk 2005]同样倾向于使用依存分析而不是短语结构分析，他们提出了一种基于依存分析的树到树的翻译模型。但是，他们并没有对双语都做句法分析，而只是对源语言句子做句法分析，然后通过映射得到目标语言句子的依存树。这个工作的一大特点在于强调源语言分析的重要性。之前的工作，如[Yamada 2001]，受传统的噪声信道模型的影响，更侧重于对目标语言进行分析。他们的系统在实验效果上超过了基于短语的系统 Pharaoh，这是很大的突破。

[Chiang 2005]提出层次化（hierarchical）基于短语的翻译模型。尽管 Chiang

称之为基于短语的模型，由于这个模型实际上是同步上下文无关语法，我们仍将其归于基于句法的模型。这篇论文被评为 ACL 2005 最佳论文，我们认为最重要的原因在于它在语法获取上找到了十分恰当的方法。如前所述，将同步语法引入统计机器翻译的主要难点在于语法获取的复杂度过高，因为获取算法大多采用类似于 EM 的迭代算法，在大规模双语语料库上获取同步语法是不现实的。[Chiang 2005]把训练数据定位为经过词语对齐的双语语料库，而不是没有加任何标注的双语语料库，从而可以利用极大似然估计方法而不是传统的 EM 算法进行参数估计。这样做的好处在于极大地降低了参数估计的复杂度。需要强调的是，这并不是 Chiang 的首创，而是基于短语的系统已普遍使用的训练技术。Chiang 的工作的优点在于：

1. 与传统的基于短语的模型相比，层次化短语不仅兼容了所有短语，还具备泛化功能，重排序功能也更强。
2. 与传统的基于句法的模型相比，训练复杂度要低得多，受数据噪音的影响更小。传统的模型或者使用复杂度高的迭代算法，或者需要对语料库作句法分析。而目前的句法分析技术仍不成熟，在处理真实文本时不仅速度慢，而且准确率较低。

Chiang 的系统 Hiero 在 2005 年和 2006 年的 NIST 评测中都取得了极佳的成绩，超过了许多基于短语的系统，成为目前最好的基于句法的系统之一。

在统计机器翻译领域，南加州大学信息科学研究所一直占据领导地位。[Germann 2001]为基于词的翻译模型的搜索算法做出了重要贡献，基于短语翻译模型的代表人物 Franz J. Och、Daniel Marcu 和 Philipp Koehn 均在南加州大学工作过，而[Yamada 2001]更被认为是第一个基于句法的翻译模型。我们下面将介绍南加州大学近年来在基于句法的翻译模型上的研究进展。[Galley 2004]详细介绍了如何从经过词语对齐和目标语言端句法分析的双语语料库上抽取树到串转换规则。他们将重点放在规则抽取算法上，没有开发实际的翻译系统。[Knight 2005]系统全面地介绍了概率化树转录机（probabilistic tree transducer），反映了南加州大学将概率化树转录机应用于统计机器翻译的理论探索。[Galley 2006]在[Galley 2004]的基础上做出改进，丰富了规则抽取算法和概率估计算法。这次，他们开发了实际的系统并和 Och 的基于对齐模板的系统作对比，显示了一定的潜力。他们的系统还十分简单，甚至没有采用对数线性模型。[Marcu 2006]采用了[Galley 2006]的规则抽取算法，在建模上略有变化，采用了对数线性模型框架并使用了十几个特征函数，在实验效果上超越了 Och 的基于对齐模板的系统。[Marcu 2006]的另一个贡献在于提出了让他们的系统兼容非句法双语短语（non-syntactic bilingual phrase）的解决方案。

[Liu 2006]提出了基于树到串对齐模板的翻译模型。树到串对齐模板是一个三

元组，描述了源语言句法树和目标语言串之间的对齐关系。此模型与[Quirk 2005]一样侧重源语言分析，但是采用短语结构分析。从另一个角度来看，一个树到串对齐模板实际上就是一个树到串翻译规则，可以解释源语言句法树和目标语言串是如何同步生成的。因此，这个工作在基本思想上和[Galley 2006]是相同的，只不过[Liu 2006]侧重源语言分析，[Galley 2006]侧重于目标语言分析，而这两个工作是分别独立提出的。

我们可以将这些工作分为两类。一类是利用同步形式语法，其基本思想是同步生成两种语言，以[Chiang 2005]为代表。众所周知，形式语法是形式化表达自然语言的有效手段，因此将同步形式语法引入机器翻译是顺理成章的。形式语法可以生成或者识别语言，并且描述语言中的结构，但能做到这些的并不只有形式语法，状态转移网络也可以。因此，另一类是利用转录机，其基本思想是将一种语言转换为另一种语言，以[Yamada 2001]为代表。近期的工作已将这两种思想结合起来。例如，[Galley 2006; Marcu 2006; Liu 2006]所使用的树到串翻译规则既可以看成是将树转换成串的规则，又可以看成是解释树和串同步生成的规则。

[Chiang 2005]将基于句法的方法分为两类：形式化基于句法（formally syntax-based）和语言学基于句法（linguistically syntax-based）。形式化基于句法的方法借用了形式化语法的结构，但并不利用语言学知识，而语言学基于句法的方法则需利用语言学知识。例如，[Chiang 2005]是形式化基于句法的方法，而[Liu 2006]则是语言学基于句法的方法。

至此，我们已经介绍了基于句法的翻译模型的主要工作。下面，我们来讨论基于句法的翻译模型的两个关键问题：

1. 句法信息的利用。简单地说，句法信息的利用是指在训练和搜索中是否使用了在树库上训练得到的句法分析器。例如，[Chiang 2005]没有利用句法信息，[Liu 2006]利用了句法信息。利用句法信息有利有弊：
    a) 利在于能够真正获得句法信息的指导，这正是基于句法的翻译模型的主要目标。
    b) 弊在于这些句法信息既难获得又不太可靠。一方面，句法分析的复杂度比较高，尤其是在处理长句子时。这给大规模训练带来了极大的困难。另一方面，目前句法分析的准确度还难以满足实用要求。句法分析器的训练数据往往来源自宾州树库，不仅规模较小，而且领域狭窄，在分析领域广泛的真实文本时准确度势必会降低。
2. 短语兼容性。利用句法信息的模型往往只能使用句法双语短语（syntactic bilingual phrase），而不能使用非句法双语短语。这会在相当大的程度上降低系统的翻译性能。基于句法的模型应当能够兼容所有的短语，这样才能既充分保留基于短语的模型的优点，又能发挥句法信息的指导作用。在这方面，[Chiang 2005]做得最好，实现了对双语短语的完全兼容。

在 2005 年的NIST机器翻译评测中，Chiang的系统代表马里兰大学参加汉译英任务，取得了第 3 名的好成绩。此后，Chiang来到南加州大学工作。在 2006 年的NIST机器翻译评测中，南加州大学将Och的系统 [1]、Marcu的系统和Chiang的系统合并起来，在汉译英任务的一个子项上超过了Google的系统。这是个重要事件，标志着基于句法的模型已经开始撼动基于短语的模型的统治地位。实际上，无论是Chiang的系统还是Marcu的系统都已经在公平的实验环境下超过Och的系统。只不过在NIST机器翻译评测中，Och可以借助Google的海量数据训练庞大的语言模型（这是其他参加评测的单位所无法企及的），从而使这种撼动显得不太明显。我们的基于树到串对齐模板的系统在 2006 年的NIST机器翻译评测汉译英任务的两个子项上分别取得了第 5 名和第 8 名的好成绩。

从实际效果来看，目前最成功的基于句法的模型是Chiang的层次化基于短语的翻译模型、Galley和Marcu的串到树 [2]模型和我们的树到串对齐模板模型。

1.3 论文组织

本文选择词语对齐和基于句法的翻译模型作为研究对象，是因为：

1. 词语对齐对统计机器翻译至关重要。统计机器翻译的基本思想是从真实文本中自动获取翻译知识，然后再运用这些知识进行翻译。直接从未标注的双语语料库中自动获取翻译知识是极其困难的，因此统计机器翻译系统的训练语料库通常需要经过标注。词语对齐正是最重要的一种标注，它指出了源语言和目标语言词语之间的对应关系。基于词的模型、基于短语的模型和基于句法的模型的训练数据均需要词语对齐标注，甚至连基于实例的方法也需要词语对齐。目前的主流方法是 IBM 模型，作为生成模型，它有许多难以克服的缺点，如难以扩充模型以容纳更多有用的信息。
2. 基于句法的翻译模型是当前统计机器翻译的研究热点。目前，基于词的模型已经过时，而基于短语的模型的发展空间也已经十分有限。基于句法的模型能够利用句法信息指导翻译，在表达能力上要比基于词的模型和基于短语的模型强。虽然早在上世纪九十年代研究人员就开始研究基于句法的翻译模型，但进展一直十分缓慢，主要表现在训练和搜索复杂度过高，而且翻译性能较低。

在第二章，我们介绍了词语对齐的对数线性模型。对数线性模型的优点在于易于整合各种知识源并自动调节知识源之间的权重。我们讨论了模型的形式化定义、特征函数设计、参数训练和搜索等问题。这一章在我们的 ACL 2005 论文[Liu

[1] Och 在 2002 年博士毕业后去南加州大学工作，之后去 Google 工作。因此，南加州大学有 Och 的基于对齐模板的系统。Och 后来在 Google 对这个系统进行进一步的改进。

[2] 南加州大学受噪声信道模型的影响，翻译规则在源语言端是串，在目标语言端是树，因此可称为串到树模型。与之相反，我们的翻译规则在源语言端是树，在目标语言端是串，因此是树到串模型。

2005]的基础上做出了许多重要扩充，主要表现在：

1. 搜索算法取消了增益阈值，因而在逻辑上更合理，在表达上更简洁。
2. 采用了最小错误率训练自动调节特征权重。
3. 采用了许多新的特征函数。
4. 在三个词语对齐评测数据集上与参加评测的系统做对比。

第三至五章分别介绍了三个基于句法的翻译模型：嵌入句法树的基于短语的翻译模型、基于树到串对齐模板的翻译模型和融入森林到串规则的树到串翻译模型。

第三章介绍了嵌入句法树的基于短语的翻译模型，简称模型 1。该模型在本质上是基于短语的，只不过以隐变量的方式嵌入句法树。在句法树的约束下，模型 1 只能使用句法双语短语。为了实现能够利用句法信息指导短语重排序，我们提出树节点重排序。任何句法短语划分都可以用一个树节点重排序序列来描述。

第四章介绍了基于树到串对齐模板的翻译模型，简称模型 2。该模型的核心是树到串对齐模板，它描述了源语言树和目标语言串之间的对应关系。在模型 2 中，只需要使用树到串对齐模板就可以完成翻译，双语短语并不是必需的，也不需要像[Quirk 2005]那样设计专门的重排序模型。因此，模型结构非常简单，训练和解码的复杂度都相对较低。本章对我们的 COLING/ACL 2006 论文[Liu 2006]进行了更详细深入的阐释，特别是训练算法和搜索算法，同时介绍了利用双语短语和语言模型提高译文流利度的技术。

第五章介绍了融入森林到串规则的树到串翻译模型，简称模型 3。模型 2 使用的是树到串对齐模板，实际上就是树到串翻译规则。这种规则只能表达句法双语短语并进行泛化，无法捕获非句法双语短语并进行泛化。为此，我们提出森林到串翻译规则，它描述了多个源语言句法树和一个目标语言串的对应关系。模型 3 是对模型 2 的扩充，它虽然在本质上是树到串翻译模型，却融入了森林到串规则来捕获非句法双语短语并进行泛化，表达能力得到增强。为了将森林到串规则融入到树到串模型中，我们引入辅助规则。辅助规则并不是从训练语料库中自动学习到的，而是在解码过程中动态构造的。我们介绍模型 3 的论文已经被 ACL 2007 录用。

在第六章，我们将这三个模型与基于短语的模型进行了对比。这四个模型使用完全一样的训练集、开发集、测试集和相关工具，以保证对比实验的公平性。实验结果表明，模型 1 的翻译性能与基准系统接近，模型 2 和模型 3 均明显超过了基准系统。我们同时还介绍了模型 2 在大规模数据上的实验结果。

第七章给出了总结和今后工作的一些设想。

第二章 词语对齐的对数线性模型

2.1 引言

词语对齐的目标在于指明平行文本中词语之间的对应关系，最早是作为统计机器翻译模型的中间产物而被提出[Brown 1993]。

词语对齐对自然语言处理的许多应用都起着关键的作用。在统计机器翻译中，经过词语对齐的语料是重要的翻译资源。统计翻译模型的参数估计大多依赖于经过词语对齐的语料，不论是基于短语的模型[Marcu 2002; Koehn 2003; Och 2004b]，还是基于句法的模型[Chiang 2005; Quirk 2005; Galley 2006; Marcu 2006; Liu 2006]。[Och 2000]指出词语对齐直接影响了机器翻译输出译文的质量。除了机器翻译，词语对齐还被应用于以下领域：

1. 机器辅助翻译[Shemtov 1993]
2. 文本生成[Smadja 1996]
3. 双语词典构造[Melamed 2000]
4. 词义消歧[Diab 2000]

研究人员提出各种各样的方法在平行文本中计算词语对齐，[Och 2003b]将这些方法分为两类：统计（statistical）方法和启发式（heuristic）方法。统计方法往往试图通过建立模型来描述平行文本之间的关系，模型参数可以从训练语料库中自动学习[Brown 1993; Vogel 1996]。这些模型将词语对齐作为一个隐变量，通过 EM 算法最大化句对的概率，从而获得每个句对的词语对齐。启发式方法通过根据语言对设计各种各样的相似度函数来计算词语对齐[Smadja 1996; Ker 1997; Melamed 2000]。统计方法和启发式方法的主要区别在于统计方法是基于概率模型而启发式方法则依赖于相似度函数。研究表明，统计对齐模型要优于简单的 Dice 系数方法[Och 2003b]。

虽然 IBM 模型已经被证明能够对大规模双语语料库执行准确度较高的词语对齐，这种生成（generative）模型有很多局限性：

1. IBM 模型对子模型做了极强的独立性假设，加入新的子模型十分困难。例如，除了观察到一个源语言词被连到一个目标语言词，我们还希望能为这个词对设计词法的特征，如词性标记、前缀、后缀等。这些特征能够更好地处理数据稀疏问题。然而，加入这些新的特征必须对模型做较大的改动，使模型变得更复杂。
2. IBM 模型是与具体语言无关的，虽然理论上能够处理任何语言对，但是无法充分利用具体语言的特性来改进词语对齐的效果。

3. 较复杂的 IBM 模型，如模型 4，采用一些启发式策略来简化搜索算法，从而不能搜索到最优的词语对齐。
4. 虽然 IBM 模型在理论上是无监督学习，但实际上许多参数需要在开发集上优化，这些参数的设置对于词语对齐的质量有很大的影响。

最近，一些研究人员将判别（discriminative）方法引入到词语对齐，取得了很大的进展。

[Liu 2005]提出词语对齐的对数线性模型，使用 IBM 模型 3、词性信息和双语词典作为特征，在汉语/英语数据上超过了 IBM 模型。

[Moore 2005]为词语对齐提出一种判别框架，对特征做线性组合，并利用感知机算法来优化特征权重。Moore 将搜索词语对齐分为两阶段，首先用一个较简单的模型对语料做第一遍对齐，再在这个经过对齐的语料上学习较复杂模型的参数，然后再执行第二遍搜索。在英语/法语数据上，该判别模型超过了 IBM 模型 4。

[Taskar 2005]同样也提出了一个词语对齐的判别模型，同样也对特征做线性组合。与[Moore 2005]不同的是，他们对源语言和目标语言词对设计特征，而不是针对源语言和目标语言句对。该模型使用了许多特征，在英语/法语数据上也超过了 IBM 模型 4。不过，这个模型只能生成一对一的词语对齐。

[Blunsom 2006]将条件随机场引入词语对齐，对多对一的对齐建模，在英语/法语和英语/罗马尼亚语的数据集上取得了很好的效果。

这些判别模型都在小规模数据上超过了 IBM 模型。虽然在结构上有差异，这些模型的共同特点是为词语对齐设计了若干特征，并且利用开发集自动训练特征权重。[Moore 2006]指出，对于词语对齐的判别模型而言，模型结构和特征设计至关重要，判别训练手段相对而言显得次要。

在本章，我们为词语对齐提出基于对数线性模型的框架。在此框架下，所有的知识源被视作依赖于源语言句子、目标语言句子以及可能的其他变量的特征函数。对数线性模型使统计对齐模型易于扩展，方便加入更多的语言学信息，从而能同时处理与具体语言相关和不相关的语言现象。我们讨论了框架的形式化定义、特征函数、最小错误率训练、搜索算法以及 n-best 列表生成等问题。我们首先将对数线性模型和 IBM 模型进行了比较，然后在三个词语对齐评测的数据集（包含五个语言对）上对词语对齐的对数线性模型进行评价。

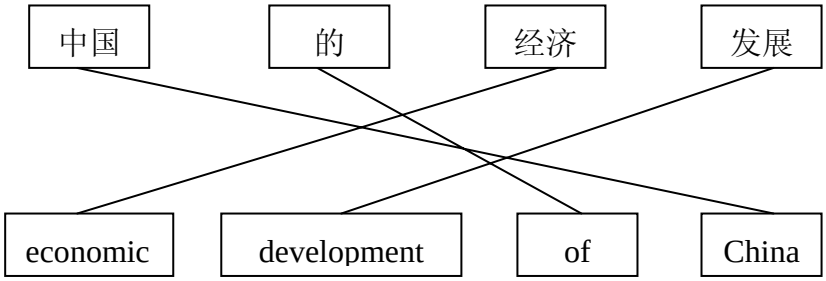

图 2.1：词语对齐例子

2.2 词语对齐的对数线性模型

下面我们将给出词语对齐的正式定义。已知源语言句子 $\mathrm{f}=f_1^J=f_1,...,f_j,...,f_J$ 和目标语言句子 $\mathrm{e}=e_1^I=e_1,...e_i,...,e_I$，如果 f_j 和 e_i 互为翻译（或者部分翻译），我们定义 $l=(j,i)$ 是一个连线。词语对齐 a 被定义为词语位置的笛卡尔集的子集：

$$\mathrm{a}\subseteq\{(j,i):j=1,...,J;i=1,...,I\} \qquad \text{公式 2.1}$$

例如，图 2.1 中的词语对齐可以表示为 $\{(1,4),(2,3),(3,1),(4,2)\}$。

我们定义对齐问题为已知源语言句子 f 和目标语言句子 e，求使 $\Pr(\mathrm{a}|\mathrm{f},\mathrm{e})$ 取得最大值的对齐 $\hat{\mathrm{a}}$：

$$\hat{\mathrm{a}}=\arg\max_{\mathrm{a}}\Pr(\mathrm{a}|\mathrm{f},\mathrm{e}) \qquad \text{公式 2.2}$$

我们直接对 $\Pr(\mathrm{a}|\mathrm{f},\mathrm{e})$ 建模，而最大熵是非常合适的框架[Berger 1996]。[Papineni 1997]将这种方法应用于自然语言理解，并且由[Och 2002a]成功地应用于统计机器翻译中。在此框架下，我们可以设计一组特征函数 $h_m(\mathrm{a},\mathrm{f},\mathrm{e})$，其中 $m=1,...,M$。对于每个特征函数，存在相应的模型参数 λ_m，其中 $m=1,...,M$。因此，词语对齐的对数线性模型可表示为：

$$\Pr(\mathrm{a}|\mathrm{f},\mathrm{e})=\frac{\exp\left[\sum_{m=1}^{M}\lambda_m h_m(\mathrm{a},\mathrm{f},\mathrm{e})\right]}{\sum_{\mathrm{a}'}\exp\left[\sum_{m=1}^{M}\lambda_m h_m(\mathrm{a},\mathrm{f},\mathrm{e})\right]} \qquad \text{公式 2.3}$$

因此，我们获得以下决策规则：

$$\hat{a} = \arg\max_{a} \left\{ \sum_{m=1}^{M} \lambda_m h_m (a, f, e) \right\} \quad \text{公式 2.4}$$

一般而言，源语言句子 f 和目标语言句子 e 是词语对齐的两个基本知识源，而那些可以确定词汇之间关联的语言学知识，往往会被传统的词语对齐方法所忽略。一些语言学工具，如词性标记器、句法分析器、命名实体识别器，已经越来越成熟并且可用于越来越多的自然语言。利用这些语言学信息来提高词语对齐是很有必要的，而且对数线性模型非常适合把这些知识利用特征函数的形式整合到模型中来。

为了加入一个新的，有别于源语言和目标语言的依赖关系，我们可以在公式 2.3 的基础上添加一个新变量 v：

$$\Pr(a \mid f, e, v) = \frac{\exp\left[\sum_{m=1}^{M} \lambda_m h_m (a, f, e, v) \right]}{\sum_{a'} \exp\left[\sum_{m=1}^{M} \lambda_m h_m (a, f, e, v) \right]} \quad \text{公式 2.5}$$

相应的决策规则为：

$$\hat{a} = \arg\max_{a} \left\{ \sum_{m=1}^{M} \lambda_m h_m (a, f, e, v) \right\} \quad \text{公式 2.6}$$

需要注意的是，我们的对数线性模型和[Och 2003b]提出的模型 6 是不同的，后者将词语对齐定义为：已知源语言句子 f，求使 $\Pr(e, a \mid f)$ 概率最大的对齐 $\hat{a}$。

2.3 特征函数

由于自然语言的多样性，词语对齐问题还远未达到充分解决的地步。比如，习惯表达、随意翻译以及内容词或功能词省略等语言现象给词语对齐带来很大的困难。当两种语言在词语顺序上差异很大时，词语对齐尤其困难。

我们认为词语对齐模型应当既能够处理与具体语言不相关的语言现象，也能够处理与具体语言相关的现象。因此，我们将对数线性模型的特征函数分为两类：非语言相关的（language-independent）和语言相关的（language-dependent）。

2.3.1 非语言相关的特征

IBM 模型

[Brown 1993]为翻译过程建立了一系列统计模型。IBM 翻译模型试图对翻译概率 $\Pr\left(f_1^J \mid e_1^I\right)$ 进行建模，以描述目标语言句子 e_1^I 和源语言句子 f_1^J 之间的关系。在统计对齐模型 $\Pr\left(f_1^J, a_1^I \mid e_1^I\right)$ 中，词语对齐 $\mathrm{a}=a_1^J$ 作为隐变量引入，描述了源语言词位置 j 到目标语言词位置 $i=a_j$ 的映射关系。翻译模型和对齐模型之间的关系可由下面的公式得到：

$$\Pr\left(f_1^J \mid e_1^I\right)=\sum_{a_1^J}\Pr\left(f_1^J, a_1^J \mid e_1^I\right) \qquad \text{公式 2.7}$$

虽然 IBM 模型被认为在逻辑上比启发式方法更有条理，它们也有两个缺点。第一，IBM 模型限制每个源语言词 f_j 只能连向一个目标语言词 e_{a_j}。更普遍的方法应当是建立一个对齐模型，使得源语言和目标语言词之间可以任意连线。第二，IBM 模型是与具体语言无关的，这样就无法处理一些与具体语言相关的语言现象。

在本文，我们使用 IBM 模型 1 至 4 的对数值作为特征。例如，以 IBM 模型 1 作为特征可表示为：

$$\begin{aligned} h_{m_1}(\mathrm{a},\mathrm{f},\mathrm{e}) &= \log\left(\Pr\left(f_1^J, a_1^J \mid e_1^I\right)\right) \\ &= \log\left(\frac{1}{(l+1)^m}\prod_{j=1}^{m} t\left(f_j \mid e_{a_j}\right)\right) \\ &= -m\log(l+1)+\sum_{j=1}^{m}\log\left(t\left(f_j \mid e_{a_j}\right)\right) \end{aligned} \qquad \text{公式 2.8}$$

我们将不同翻译方向的IBM模型视作不同的特征：将英语作为源语言、法语作为目标语言或者将法语作为源语言、英语作为目标语言[3]。

词性标记转换模型

除了源语言和目标语言句子，我们采用的第一个语言学信息是词性标记（Part-of-Speech）。[Toutanova 2002]使用词性标记来提高基于 HMM 模型的对齐质量。他们为两种语言引入词性标记的附加词汇概率。

[3] 假设是英语/法语平行文本。

在 IBM 模型和 HMM 模型中，如果想要容纳新的信息，必须设计一个扩充的模型使之能够利用前面的模型参数。而对数线性模型却可以很容易地容纳新信息。

我们使用词性标记转换模型作为特征。这个特征从外部数据（held-out data）通过简单计数学习词性标记转换概率，然后将学习到的概率分布应用到评价词语对齐中去。概率估计方法如下：

$$p(fT \mid eT) = \frac{N_A(fT, eT)}{N(eT)} \qquad \text{公式 2.9}$$

其中，$N_A(fT, eT)$是指词性标记 fT 连向词性标记eT 的次数，$N(eT)$是词性标记eT 出现的次数。

我们定义 $\mathrm{eT} = eT_1^I = eT_1, ..., eT_i, ..., eT_I$ 和 $\mathrm{fT} = fT_1^J = fT_1, ..., fT_j, ..., fT_J$ 分别是句子 e 和 f 的词性标记序列，则词性标记转换模型定义如下：

$$\Pr(\mathrm{fT}|\mathrm{a},\mathrm{eT}) = \prod_{l \in \mathrm{a}} t\left(fT_{l(j)} \mid eT_{l(i)}\right) \qquad \text{公式 2.10}$$

其中，l 是 a 中的一个元素，换言之，l 是一条连线。$l(i)$ 是 l 中的目标语言词位置，$l(j)$是 l 中的源语言词位置。

因此，特征函数可以设计为：

$$h_{pos}(\mathrm{a},\mathrm{f},\mathrm{e}) = \log\left(\prod_{l \in \mathrm{a}} t\left(fT_{l(j)} \mid eT_{l(i)}\right)\right) \qquad \text{公式 2.11}$$

需要说明的是，公式 2.11 中的特征函数左侧的完整形式应该是 $h_{pos}(\mathrm{a},\mathrm{f},\mathrm{e}, fT_1^J, eT_1^I)$。为了表示上的简便，我们忽略了 fT_1^J 和 eT_1^I 这两个依赖关系。

类似地，我们将不同翻译方向的词性标记转换模型区分为不同的特征：将英语作为源语言、法语作为目标语言或者将法语作为源语言、英语作为目标语言。

双语词典

双语词典也可以作为附加的知识源。给定词语对齐，我们可以统计双语词典中有多少个词条在对齐中共现。因此，双语短语的权重就可以获得。我们采用双语词典作为特征的原因在于双语词典应该比自动获得的词典更可靠，同时也应当

获得较大的权重。

我们定义双语词典是词条的集合：$\mathrm{D}=\{f,e,conf\}$。其中，f 是源语言词，e 是目标语言词，$conf$ 是一个正实数（通常是 1.0）。$conf$ 是由词典编纂者设定，用来表示该词条有效性的程度。因此，使用双语词典的特征为：

$$h_{dict}(\mathrm{a},\mathrm{f},\mathrm{e})=\sum_{l\in \mathrm{a}} occur\left(f_{l(j)},e_{l(i)},\mathrm{D}\right) \qquad \text{公式 2.12}$$

其中，

$$occur(e,f,\mathrm{D})=\begin{cases} conf & \text{if } (\mathrm{e},\mathrm{f}) \text{occurs in D} \\ 0 & \text{else} \end{cases} \qquad \text{公式 2.13}$$

连线计数

我们使用连线计数作为特征，用来控制连线的数量：

$$h_{lc}(\mathrm{a},\mathrm{f},\mathrm{e})=|\mathrm{a}| \qquad \text{公式 2.14}$$

例如，在图 2.1 所示的词语对齐中，连线计数是 4。

交叉计数

我们使用交叉计数作为特征，用来控制词序调整情况：

$$h_{cc}(\mathrm{a},\mathrm{f},\mathrm{e})=\sum_{l_1\in \mathrm{a}}\sum_{l_2\in \mathrm{a}} cross(l_1,l_2) \qquad \text{公式 2.15}$$

其中，

$$cross(l_1,l_2)=\begin{cases} 1 & (l_1(i)-l_2(i))\times(l_1(j)-l_2(j))<0 \\ 0 & (l_1(i)-l_2(i))\times(l_1(j)-l_2(j))\geq 0 \end{cases} \qquad \text{公式 2.16}$$

例如，在图 2.1 所示的词语对齐中，交叉计数是 5。

2.3.2 语言相关的特征

词根还原的 IBM 模型

许多拉丁语系的语言都有丰富的词后缀。在训练语料库规模较小的情况下，这些后缀会造成严重的数据稀疏问题。因此，可以考虑词根还原来克服这个问题。在此，我们借鉴了[Fraser 2005]使用的方法，也就是对每个词语选取前 4 个字母。这是个非常简单的方法，事实上，如果能够获得合适的工具，对各种语言做真正

的词根还原并非难事。

例如，下面的英语句子

I was playing basketball when my friends arrived

经过词根还原后的形式是

I was play bask when my frie arri

因此，可以对拉丁语系的训练语料库进行词根还原，然后使用 GIZA++训练 IBM 模型参数。使用词根还原的 IBM 模型在对测试集进行对齐时，同样也只读入前 4 个字母。这样一来，词根还原的 IBM 模型与原始的 IBM 模型相互独立，可以同时使用。

因此，词根还原的 IBM 模型 1 特征可以表示为：

$$h_{\bar{m}_1}(\mathrm{a},\mathrm{f},\mathrm{e}) = h_{m_1}(\mathrm{a},\bar{\mathrm{f}},\bar{\mathrm{e}}) \qquad \text{公式 2.17}$$

其中，$\bar{m}_1$ 是指词根还原的 IBM 模型 1，$\bar{\mathrm{f}}$ 和 $\bar{\mathrm{e}}$ 分别指词根还原的源语言句子和目标语言句子。其他的词根还原的 IBM 模型特征也可以类似地定义。

完全匹配

很多语言都是同源的，虽然后来的发展方向不一样，但是还是会有不少词语在两种语言是完全一样的。例如，日语中借用了许多汉字，而英语和法语的一些单词是一样的。如果源语言词 f 和目标语言词 e 完全相同，我们称之为完全匹配，其特征函数可以表示为：

$$h_{em}(\mathrm{a},\mathrm{f},\mathrm{e}) = \sum_{l\in \mathrm{a}} \delta\left(f_{l(j)}, e_{l(i)}\right) \qquad \text{公式 2.18}$$

2.4 训练

本节介绍如何自动获得词语对齐的对数线性模型的特征权重，主要采用两种方法：通用迭代算法和最小错误率训练。

2.4.1 通用迭代算法

通用迭代算法（Generalized Iterative Scaling，简称 GIS）是[Darroch 1972]提出的，大致可以概括为以下几个步骤：

1. 假定初始模型为等概率的均匀分布。
2. 用当前模型来估算各种特征在训练数据中的分布，从而生成新的模型。
3. 重复步骤 2 直至收敛。

根据公式 2.3，我们使用通用迭代算法来训练对数线性模型的模型参数 λ_1^M 。经过适当的转换，通用迭代算法可以用来处理实数值特征。我们采用Franz J. Och开发的YASMET[4]来执行训练。

公式 2.3 中的重正化（renormalization）需要大量的候选对齐集合。如果源语言句子 f 包含 J 个词，目标语言句子 e 包含 I 个词，那么总共能够产生的词语对齐数目是 $2^{I\times J}$ [Brown 1993]。当句子长度很长时，枚举所有可能的词语对齐是不现实的。因此，我们用较大数量的高概率对齐集合来逼近所有可能的对齐集合，这样的对齐集合也称之为 n-best 列表。

我们在开发集上训练模型参数。开发集包含数百个人工对齐的双语句对。使用 n-best 列表逼近可能会导致使用通用迭代算法训练的参数在测试集上产生质量较差的对齐，甚至是在开发集上也质量较差。这是因为在训练过程中，模型参数变化很大，并且可能会包含训练中没有考虑的对齐。为了避免这个问题，我们依照[Och 2002a]的方法来迭代训练模型参数，每次迭代都会合并 n-best 列表，直至 n-best 列表不再变化为止。然而，这种训练方法是基于极大似然准则（maximum likelihood criterion）的，与最终未知双语文本的对齐质量关联很小。因此，当迭代结束时，我们会得到一组模型参数，我们选择在开发集上产生最好对齐的模型参数。

2.4.2 最小错误率训练

最小错误率训练（minimum error rate training）是由[Och 2003a]提出的，被广泛应用于统计机器翻译中。我们将这种方法引入到词语对齐中来。

已知词语对齐 a 和参考对齐 r，假定我们可以通过函数 $E(\mathrm{r},\mathrm{a})$ 来估计词语对齐 a 的错误。一组词语对齐 a_1^S 的错误可由单个词语对齐的错误累加起来：

$$E\left(\mathrm{r}_1^S,\mathrm{a}_1^S\right)=\sum_{s=1}^{S}E\left(\mathrm{r}_s,\mathrm{a}_s\right) \qquad \text{公式 2.19}$$

已知一个双语语料库 $<\mathrm{f}_1^S,\mathrm{e}_1^S>$ 和参考对齐集合 r_1^S，并且每个句对 $<\mathrm{f}_s,\mathrm{e}_s>$ 有

[4] 可在 http://www.fjoch.com/YASMET.html 下载。

K 个候选对齐 $\mathrm{C}_s=\{\mathrm{a}_{s,1},...,\mathrm{a}_{s,K}\}$，最小错误率训练的目的在于获取令错误率最小的模型参数：

$$\begin{aligned}\hat{\lambda}_1^M &= \arg\min_{\lambda_1^M}\left\{\sum_{s=1}^{S}E\left(\mathrm{r}_s,\hat{\mathrm{a}}\left(\mathrm{f}_s,\mathrm{e}_s;\lambda_1^M\right)\right)\right\}\\ &= \arg\min_{\lambda_1^M}\left\{\sum_{s=1}^{S}\sum_{k=1}^{K}E\left(\mathrm{r}_s,\mathrm{a}_{s,k}\right)\delta\left(\hat{\mathrm{a}}\left(\mathrm{f}_s,\mathrm{e}_s;\lambda_1^M\right),\mathrm{a}_{s,k}\right)\right\}\end{aligned} \qquad \text{公式 2.20}$$

其中，

$$\hat{\mathrm{a}}\left(\mathrm{f}_s,\mathrm{e}_s;\lambda_1^M\right)=\arg\max_{\mathrm{a}\in\mathrm{C}_s}\left\{\sum_{m=1}^{M}\lambda_m h_m\left(\mathrm{a}\mid\mathrm{f}_s,\mathrm{e}_s\right)\right\} \qquad \text{公式 2.21}$$

Powells 算法及基于格（grid-based）的线性优化方法[Press 2002]是处理这种无平滑错误计数优化问题的标准算法。这种方法在 K 维参数空间中随机选择一个初始点，通过固定 $K-1$ 维参数对单维进行优化的方法在参数空间中找到评价值更高的点。[Och 2003a]提出一种算法能够显著减少评价的次数。该算法的基本思想是只考虑那些会产生不同错误计数的分界点。

由于基本思想是一样的，只是具体问题略为不同，我们直接将[Och 2003a]所描述的算法应用到词语对齐的对数线性模型。

2.5 搜索

我们采用贪心算法从对齐空间中搜索概率最高的对齐。空间中的一个状态是一个部分对齐。在当前状态下增加一条连线被称之为迁移。开始状态是空对齐，源语言和目标语言的所有词都连向空。终止状态是添加任何连线都无法使概率进一步增长的状态。搜索的过程就是从开始状态开始，不断地添加连线，直至概率不再增长为止。

已知一个对齐 a，其概率为：

$$\Pr\left(\mathrm{a}\mid\mathrm{f},\mathrm{e}\right)=\frac{\exp\left[\sum_{m=1}^{M}\lambda_m h_m\left(\mathrm{a},\mathrm{f},\mathrm{e}\right)\right]}{\sum_{\mathrm{a}'}\exp\left[\sum_{m=1}^{M}\lambda_m h_m\left(\mathrm{a},\mathrm{f},\mathrm{e}\right)\right]}$$

新增一条连线 l 后的对齐为 $\mathrm{a}\cup l$，其概率为：

$$\Pr\left(\mathrm{a}\cup l\mid\mathrm{f},\mathrm{e}\right)=\frac{\exp\left[\sum_{m=1}^{M}\lambda_m h_m\left(\mathrm{a}\cup l,\mathrm{f},\mathrm{e}\right)\right]}{\sum_{\mathrm{a}'}\exp\left[\sum_{m=1}^{M}\lambda_m h_m\left(\mathrm{a},\mathrm{f},\mathrm{e}\right)\right]}$$

输入：源语言句子 f，目标语言句子 e，其他依赖关系
1. 初始化词语对齐：a=ϕ。 2. 对每个不属于 a 的连线 $l=(j,i)$ 计算增益 $gain(\mathrm{a},l)$。 3. 如果对于任意的连线 l，均有 $gain(\mathrm{a},l)\le 0$，算法终止。 4. 向 a 中添加增益 $gain(\mathrm{a},l)$ 最大的连线 $\hat{l}$。 5. 转到 2。
输出：词语对齐 a

图 2.2：搜索算法

如果要求概率是增长的，必须有：

$$\frac{\Pr(\mathrm{a}\cup l\,|\,\mathrm{f},\mathrm{e})}{\Pr(\mathrm{a}\,|\,\mathrm{f},\mathrm{e})}=\frac{\exp\left[\sum_{m=1}^{M}\lambda_m h_m(\mathrm{a}\cup l,\mathrm{f},\mathrm{e})\right]}{\exp\left[\sum_{m=1}^{M}\lambda_m h_m(\mathrm{a},\mathrm{f},\mathrm{e})\right]}$$

$$=\exp\left[\sum_{m=1}^{M}\lambda_m\left[h_m(\mathrm{a}\cup l,\mathrm{f},\mathrm{e})-h_m(\mathrm{a},\mathrm{f},\mathrm{e})\right]\right]>1$$

因此，我们通过计算增益而不是概率来提高效率。增益的定义如下：

$$gain(\mathrm{a},l)=\sum_{m=1}^{M}\lambda_m\left[h_m(\mathrm{a}\cup l,\mathrm{f},\mathrm{e})-h_m(\mathrm{a},\mathrm{f},\mathrm{e})\right] \qquad \text{公式 2.22}$$

其中，$l=(j,i)$ 是添加到 a 的一条连线。

需要指出的是，2.3 节所描述的各种特征函数特意被设计成适合做特征值的减法。

对于词语对齐的对数线性模型，搜索算法如图 2.2 所示。我们收集搜索过程中生成的所有候选对齐作为参数训练所需的 n-best 列表。

2.6 实验

评价在自然语言处理领域日益重要。我们认为，词语对齐评价务必要考虑到以下因素：

1. 语言对的差异程度

   语言对的差异(divergence)程度决定了词语对齐的难易程度。比如，汉英对齐明显要难于法英对齐。考察多种差异程度的语言对，有利于全面衡量一种词语对齐方法的优劣。

2. 资源的丰富程度

丰富的资源会给词语对齐提供足够的学习样本，如果资源匮乏则会给词语对齐带来困难。一个好的词语对齐方法应该在资源匮乏的情况下也表现良好。

3. 学习的方式

在提供开发集和测试集的情况下，基于有监督的学习方式的词语对齐方法明显比基于无监督的学习方式的方法更有优势。这样的比较不一定能够反映在大规模数据上的对齐性能差异。因此，在大规模数据上利用其他的评价方式间接地评价词语对齐质量是有必要的。

在本节，我们将对词语对齐的对数线性模型进行系统全面的评价。2.6.1 节将对数线性模型和 IBM 模型在汉语/英语数据集上进行了对比。2.6.2 节则在三个词语对齐评测的数据集（包含五个语言对）上对词语对齐的对数线性模型进行评价。

实验结果表明，对数线性模型不仅超过了传统的 IBM 模型，而且超过了三个词语对齐评测的绝大多数参评系统。

2.6.1 与 IBM 模型比较

本节将给出在汉英平行语料库上的实验结果。在这个实验中，我们使用三个特征：IBM 模型 3、词性标记转换模型和双语词典。我们使用了训练集、双语词典、开发集和测试集。表 2.1 给出了它们的一些统计数据。

表 2.1：训练集、双语词典、开发集和测试集的统计数据。

		英语	汉语
训练集	句子数	108,925	
	词语数	3,784,106	3,862,637
	词汇量	49,962	55,698
双语词典	词条数	415,753	
	词汇量	206,616	203,497
开发集	句子数	435	
	词语数	11,462	14,252
	词汇量	26.35	32.76
测试集	句子数	500	
	词语数	13,891	15,291
	词汇量	27.78	30.58

开发集和测试集中的汉语句子采用 ICTCLAS[Zhang 2003]进行分词和标注。

我们自己开发了一个简单的 tokenizer 处理英语句子，然后用 Eric Brill 开发的基于规则的标记器[Brill 1995]做词性标记。我们对 935 个句对进行人工对齐，从中挑选 500 句对作为测试集，其余 435 句作为开发集，用来优化模型参数。在这个实验中，我们采用的是 2.4.1 节描述的通用迭代算法来训练模型参数。

给定人工标注的词语对齐，我们采用准确率（precision）、召回率（recall）和[Och 2003b]提出的对齐错误率(alignment error rate，简称 AER)作为评价标准：

$$precision = \frac{|A \cap P|}{|A|} \qquad \text{公式 2.23}$$

$$recall = \frac{|A \cap S|}{|S|} \qquad \text{公式 2.24}$$

$$AER = 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|} \qquad \text{公式 2.25}$$

其中，A 是词语对齐系统输出的连线集合，S 是人工标注者标为“确定”的连线集合，P 是人工标注者标为“可能”的连线集合，S 是 P 的子集。在实验中，我们只采用一种标记类型，因此 $S = P$ 。

我们使用 GIZA++[Och 2003b]训练 IBM 模型。训练方案是 $1^5H^53^5$，即模型 1 训练 5 次，HMM 模型训练 5 次，模型 3 训练 5 次。除了改变模型的迭代次数，我们使用 GIZA++的默认配置。之后，我们使用了[Och 2003b]提出的三种 IBM 模型的平衡化（symmetrization）方法：相交（intersection）、合并（union）和优化法（refined method）。

给定 GIZA++的输出参数，我们将其用于对数线性模型的特征 IBM 模型 3。换言之，除了词性标记转换概率表和双语词典，我们的对数线性模型和 GIZA++使用完全相同的参数。

表 2.2 给出了我们的对数线性模型和 IBM 模型 3 的结果。其中，第 3 至 7 行是 IBM 模型 3 的结果，第 8 至 12 行是对数线性模型的结果。第 9 行“+Model C->E”的意思是对数线性模型采用两个特征：Model 3 E->C 和 Model 3 C->E，依此类推。

表 2.2：IBM 模型 3 和对数线性模型的 AER 值比较

		训练语料库规模				
		1K	5K	9K	39K	109K
IBM 模型	Model 3 E->C	0.4497	0.4081	0.4009	0.3791	0.3745
	Model 3 C->E	0.4688	0.4261	0.4221	0.3856	0.3469
	Intersection	0.4588	0.4106	0.4044	0.3823	0.3687
	Union	0.4596	0.4210	0.4157	0.3824	0.3703
	Refined Method	0.4154	0.3586	0.3499	0.3153	0.3068
对数线性模型	Model 3 E->C	0.4490	0.3987	0.3834	0.3639	0.3533
	+ Model 3 C->E	0.3970	0.3317	0.3217	0.2949	0.2850
	+POS E->C	0.3828	0.3182	0.3082	0.2838	0.2739
	+POS C->E	0.3795	0.3160	0.3032	0.2821	0.2726
	+Dict	0.3650	0.3092	0.2982	0.2738	0.2685

从表 2.2 可以看出，我们的对数线性模型在所有的训练语料库规模上都比 IBM 模型取得更低的 AER 值。单独考虑 Model 3 E->C，即以英语为源语言、汉语为目标语言的模型 3。我们的贪心搜索算法比 GIZA++所采用的爬山算法（hill climbing algorithm）取得更好的结果。

表 2.3 给出了我们的对数线性模型和 IBM 模型 5 的结果。训练方案是 $1^5H^53^54^55^5$。对数线性模型同样使用 GIZA++的输出参数。

对比表 2.2 和表 2.3，我们发现对数线性模型使用训练方案$1^5H^53^54^55^5$的输出参数的对齐质量要略高于使用训练方案$1^5H^53^5$，这归功于附加的模型 4 和模型 5 的训练。

对数线性模型采用了词性标记信息和双语词典，而 IBM 模型没有采用。然而，如果把对数线性组合(Model 3 E->C + Model 3 C->E)视作一种平衡化的方法，它依然比交叉、合并和优化法要好。

表 2.3：IBM 模型 5 和对数线性模型的 AER 值比较

		训练语料库规模				
		1K	5K	9K	39K	109K
IBM 模型	Model 5 E->C	0.4384	0.3934	0.3853	0.3573	0.3429
	Model 5 C->E	0.4564	0.4067	0.3900	0.3423	0.3239
	Intersection	0.4432	0.3916	0.3798	0.3466	0.3267
	Union	0.4499	0.4051	0.3923	0.3516	0.3375
	Refined Method	0.4106	0.3446	0.3262	0.2878	0.2748
对数线性模型	Model 3 E->C	0.4372	0.3873	0.3724	0.3456	0.3334
	+ Model 3 C->E	0.3920	0.3269	0.3167	0.2842	0.2727
	+POS E->C	0.3807	0.3122	0.3039	0.2732	0.2667
	+POS C->E	0.3731	0.3091	0.3017	0.2722	0.2657
	+Dict	0.3612	0.3046	0.2943	0.2658	0.2625

图 2.4 给出了特征数量和训练语料库规模对于搜索效率的影响。可以看出，特征越多，训练语料库规模越大，搜索时间越长。

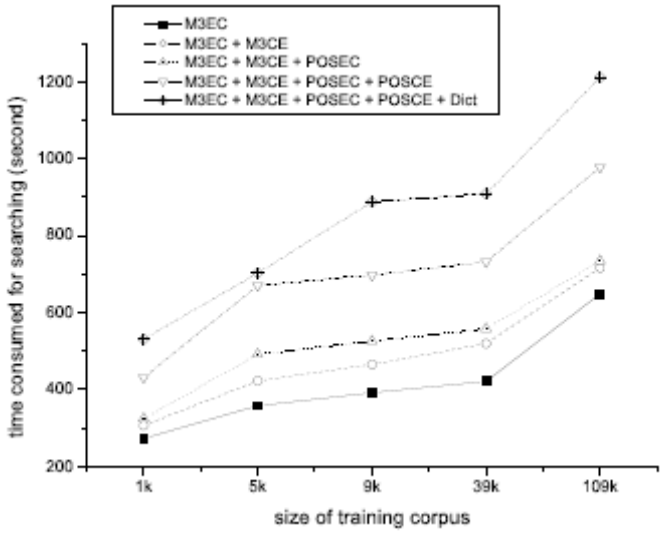

图 2.4：特征数量和训练语料库规模对于搜索效率的影响

表 2.4 给出了我们在开发集上训练得到的模型参数。我们注意到加入新的特

征会影响到其它特征的模型参数。

表 2.4：模型参数。λ_1：Model 3 E->C (MEC)；λ_2：Model 3 C->E (MCE)；λ_3：POS E->C (PEC)；λ_4：POS C->E (PCE)；λ_5：Dict。模型参数被正规化使得 $\sum_{m=1}^{5}\lambda_m=1$。

	MEC	+MCE	+PEC	+PCE	+Dict
λ_1	1.000	0.466	0.291	0.202	0.151
λ_2	-	0.534	0.312	0.212	0.167
λ_3	-	-	0.397	0.270	0.257
λ_4	-	-	-	0.316	0.306
λ_5	-	-	-	-	0.119

2.6.2 在三个评测数据集上的结果

在本节，我们在三个公开的评测数据集上做实验，与当时参加评测的系统做对比。在这三个词语对齐评测上，我们使用 2.3 节描述的所有特征函数，并采用 2.4.2 节的最小错误率训练来自动学习模型参数。

三个词语对齐评测简介

HLT-NAACL 2003 有个 Workshop 名为“Building and Using Parallel Texts: Data Driven Machine Translation and Beyond”，其中有个词语对齐的 Shared Task。这个评测有两个子任务，一个是 Romanian-English，一个是 English-French。资源使用分为受限和不受限两种。表 2.5 给出了 HLT-NAACL 2003 Workshop 评测数据情况。

表 2.5：HLT-NAACL 2003 Workshop 评测数据情况

任务	句对数量		
	训练集	开发集	测试集
Romanian-English	48,481	17	248
English-French	1,130,104	37	447

共有 7 个队伍参加 Shared Task，总共提交了 27 个 结果，其中 13 个属于 Romanian-English 子任务，14 个属于 English-French 子任务。

为了方便做对比，我们所有的实验均是在资源受限的条件下做的。

ACL 2005 同样有个 Workshop 名为“Building and Using Parallel Text”，其中有个词语对齐的 Shared Task。这个评测有三个子任务，一个是 English-Inuktitut，

表 2.6：ACL 2005 Workshop 评测数据情况

任务	句对数量		
	训练集	开发集	测试集
English-Inuktitut	340,526	25	75
Romanian-English	48,481	248	200
English-Hindi	3,441	25	90

一个是 Romanian-English，一个是 English-Hindi。本次 shared Task 的重点放在处理资源匮乏的语言上。资源使用分为受限和不受限两种。评测数据情况见表 2.6。

共有 7 个队伍参加 Romanian-English 子任务，4 个队伍参加 English-Inuktitut，两个队伍参加 English-Hindi 子任务，总共提交了 27 个 结果。

为了方便做对比，我们所有的实验均是在资源受限的条件下做的。

2005 年 863 词语对齐评测只有一个子任务：Chinese-English。资源不受限，国内有两家单位报名参加。这是国内首次举办的词语对齐评测。评测数据情况见表 2.7

表 2.7：HTRDP 2005 评测数据情况

任务	句对数量		
	训练集	开发集	测试集
Chinese-English	-	502	505

IBM 模型特征的效果

表 2.8 给出了对数线性模型使用 IBM 模型特征在所有评测任务上的 AER 值。

表 2.8：IBM 模型特征的效果

特征	1_RE	2_EF	3_EI	4_RE	5_EH	6_CE
m1_s2t	0.4159	0.2108	0.4498	0.4337	0.5290	0.2914
m1_t2s	0.4161	0.1970	0.3437	0.4285	0.5330	0.2992
m1+	0.3749	0.1581	0.2222	0.3700	0.4812	0.2369
m2_s2t	0.3706	0.1248	0.4433	0.3752	0.5323	0.2308
m2_t2s	0.3730	0.1206	0.3052	0.3635	0.5420	0.2655
m2+	0.3495	0.1046	0.2017	0.3256	0.4778	0.2159
m3_s2t	0.3716	0.1184	0.4734	0.3917	0.5472	0.2182
m3_t2s	0.3735	0.1254	0.3002	0.3750	0.5634	0.2483
m3+	0.3514	0.1127	0.2127	0.3338	0.5057	0.2185
m4_s2t	0.3507	0.0881	0.4521	0.3932	0.5565	0.2049
m4_t2s	0.3549	0.0955	0.2999	0.3773	0.5652	0.2292
m4+	0.3016	0.0728	0.1808	0.3113	0.5298	0.1970

其中，1_RE 和 2_EF 分别表示 HLT-NAACL 2003 Workshop 的 Romanian-English 和 English-French 这两个任务，3_EI、4_RE 和 5_EH 分别表示 ACL 2005 Workshop 的 English-Inuktitut、Romanian-English 和 English-Hindi 这三个任务，6_CE 表示 HTRDP 2005 的 Chinese-English 任务。

m1_s2t 表示以 IBM 模型 1 为特征，翻译方向是源语言到目标语言。m1_t2s 表示以 IBM 模型 1 为特征，翻译方向是目标语言到源语言。m1+是指对数线性模型同时使用 m1_s2t 和 m1_t2s 为特征，其特征权重比例是 1:1。我们称之为简单的对数线性组合。依此类推。

从表中可以看出，对于所有的 IBM 模型和所有的语言对，简单的对数线性组合都能取得比单独使用两个特征更好的效果。

通常认为，IBM 模型越往后对齐效果是越好的，这一点可以从[Och 2003b]中看出。然而，在我们的实验中，IBM 模型 3 却普遍比 IBM 模型 2 差一些。原因可能在于，为了实验方便，我们实际上使用 GIZA++训练到模型 4，然后再提取参数，计算 IBM 模型 2 和 IBM 模型 3 的特征值。因此，这两个特征共同使用的参数（如翻译概率表）没有差别，使模型 3 的对齐效果不一定比模型 2 好。

另一个有趣的现象是，在 English-Hindi 任务上，模型 1 和 2 的效果居然比模型 3 和 4 好。众所周知，模型 1 和 2 与模型 3、4 和 5 是基于不同的对翻译过程的理解的。一般认为后者是优于前者的，但我们发现 English-Hindi 这个语言对也许是个特例。

词根还原的 IBM 模型

表 2.9：词根还原的 IBM 模型的效果

特征	原始	词根还原
m1_s2t	0.4337	0.4040
m1_t2s	0.4285	0.3944
m1+	0.3700	0.3565
m2_s2t	0.3752	0.3301
m2_t2s	0.3635	0.3166
m2+	0.3256	0.3030
m3_s2t	0.3917	0.3391
m3_t2s	0.3750	0.3212
m3+	0.3338	0.3113
m4_s2t	0.3932	0.3324
m4_t2s	0.3773	0.3164
m4+	0.3113	0.2932

我们使用 ACL 2005 Workshop 中的 Romanian-English 任务来考察词根还原的 IBM 模型的效果。表 2.9 给出了原始的 IBM 模型和词根还原后的 IBM 模型的对比情况。可以看出，使用词根还原可以明显降低 AER 值。

连线计数特征的效果

我们使用 HTRDP 2005 的 Chinese-English 任务来考察连线计数的效果。我们使用三个特征：m4_s2t、m4_t2s 和连线计数。设定 m4_s2t 和 m4_t2s 的特征权重均为 1，手工设定连线计数的特征权重，实验结果如表 2.10 所示。一般而言，连线计数特征权重越大，越鼓励更多的连线，从而召回率升高，准确率降低；反之则准确率升高、召回率降低。AER 值综合考虑了准确率和召回率，连线计数特征权重太大或者太小都会降低 AER 值。在表 2.10，连线计数特征权重设为-2.0 时取得了最低的 AER。

表 2.10：连线计数特征的影响

连线计数	AER
-4.0	0.1973
-3.0	0.1965
-2.0	0.1955
-1.0	0.1961
0	0.1970
1.0	0.1981

完全匹配特征的效果

表 2.11：完全匹配特征的效果

特征	开发集	测试集
m2_s2t+m2_t2s+m4_s2t+m4_t2s+lc+cc	0.0771	0.0681
m2_s2t+m2_t2s+m4_s2t+m4_t2s+lc+cc+em	0.0691	0.0633

表 2.11 给出了完全匹配特征在 HLT-NAACL 2003 Workshop 的 English-French 任务上的效果。其中，lc 是指连线计数，cc 是指交叉计数，em 是指完全匹配。从结果可以看出，添加完全匹配特征在 AER 上能降低约 0.005。

评测成绩汇总

表 2.12：对数线性模型在三个词语对齐评测上的 AER 值。

任务	子任务	参评系统结果	我们的结果	排名
HLT/NAACL 2003 workshop on Parallel Texts	Romanian-English, non-null	0.2886-0.5267	0.2660	1/10
	Romanian-English, null	0.3741-0.5979	0.3234	1/10
	English-French, non-null	0.0853-0.2938	0.0633	1/9
	English-French, null	0.1850-0.5171	0.0633	1/9
ACL 2005 workshop on Parallel Texts	English-Inuktitut	0.0946-0.7127	0.1784	4/11
	Romanian-English	0.2655-0.4449	0.2614	1/34
	English-Hindi	0.5142	0.4764	1/2
HTRDP 2005	Chinese-English	0.2348-0.4918	0.1815	1/3

表 2.12 总结了我们的对数线性模型在这三个词语对齐评测上的实验结果。

可以看到，对数线性模型超过了绝大多数参评系统。我们唯一没超过参评系统的子任务是 English-Inuktitut，取得最好成绩的参评单位专门针对纽因特语的具体特点做了许多处理，我们目前的系统还没有设计相应的特征。

实验结果表明，由于对数线性模型能够根据语言对的特性灵活采用适当的特征函数，无论是差异性较大的语言对（如 English-Inuktitut）还是差异性较小的语言对（如 English-French），无论是资源较丰富的语言对（如 Chinese-English）还是资源较匮乏的语言对（如 English-Hindi），使用对数线性模型都可以得到比较好的效果。

2.7 结论

在本章，我们为词语对齐提出基于对数线性模型的框架。在此框架下，所有的知识源被视作依赖于源语言句子、目标语言句子以及可能的其他变量的特征函数。对数线性模型使统计对齐模型易于扩展，方便加入更多的语言学信息，从而能同时处理与具体语言相关和不相关的语言现象。我们讨论了框架的形式化定义、特征函数、最小错误率训练、搜索算法以及 n-best 列表生成等问题。我们首先将对数线性模型和 IBM 模型进行了比较，然后在三个词语对齐评测的数据集（包含五个语言对）上对词语对齐的对数线性模型进行评价。

我们对实验结果做一个简单的总结：

1. 对数线性模型明显超过了 IBM 模型。
2. 简单的对数线性组合要优于相交、合并和优化法等平衡化方法。
3. 在处理 English-Hindi 语料时，IBM 模型 1 和 2 要优于模型 3 和 4。这是比较反常的现象，说明模型的性能与具体语言有一定的关系。
4. 在三个词语对齐评测上，对数线性模型超过了绝大多数参评系统。

然而，词语对齐的对数线性模型目前存在以下问题：

1. 偏向生成一对一对齐。在我们的词语对齐对数线性模型中，IBM 模型是主要特征。由于 IBM 模型只允许一对多对齐，当两个不同翻译方向的 IBM 模型同时被激活时，会偏向于生成一对一对齐。更普遍的情况应当是使得源语言和目标语言词之间可以任意连线。对数线性模型是否允许多对多对齐要看所使用的特征是否允许多对多对齐。事实上，这也是词语对齐目前面临的一大难题，即如何为多对多对齐建立模型。
2. 难以在大规模语料上使用。对数线性模型在处理大规模语料时要面临三个困难：
   a) 速度。所采用的特征越多，搜索速度越慢。这是无法避免的。
   b) 特征权重确定。对数线性模型必须在开发集上调节好模型参数，而处理真实数据是不存在开发集的。一个解决方案是随机从语料库中选出一个子集做人工标注，作为开发集使用。
   c) AER 值越低并不意味着 BLEU 值越高。对大规模语料进行词语对齐

> 标注的主要目的是为了训练统计机器翻译模型参数。词语对齐质量越高，系统输出译文的质量就越高。从目前的研究成果来看，AER 值越低并不一定意味着 BLEU 值越高，通常认为召回率更重要些。

事实上，这是词语对齐的判别方法所普遍面临的挑战。这也是为什么尽管判别方法在词语对齐评测中明显超过了 IBM 模型，却无法在大规模数据处理上取代 GIZA++的原因。

第三章 嵌入句法树的基于短语的翻译模型

3.1 引言

基于短语的方法是目前统计机器翻译的主流方法。我们首先给出一个较常见的基于短语的翻译模型的形式化定义[5]，进而讨论基于短语的翻译模型的几个基本问题。

3.1.1 基于短语的模型

给定源语言句子 $f_1^J = f_1,...,f_j,...,f_J$ 和目标语言句子 $e_1^I = e_1,...,e_i,...,e_I$，首先假设存在一个隐变量 B 将源语言句子和目标语言句子同时划分为 K 个短语：

$$f_1^J = \tilde{f}_1^K, \tilde{f}_k = f_{j_{k-1}+1},...,f_{j_k} \quad \text{公式 3.1}$$

$$e_1^I = \tilde{e}_1^K, \tilde{e}_k = e_{i_{k-1}+1},...,e_{i_k} \quad \text{公式 3.2}$$

其推导方式为

$$\begin{aligned} \Pr(e_1^I \mid f_1^J) &= \sum_B \Pr(e_1^I, B \mid f_1^J) \\ &= \sum_B \Pr(B \mid f_1^J) \Pr(e_1^I \mid B, f_1^J) \\ &= \sum_B \Pr(B \mid f_1^J) \Pr(\tilde{e}_1^K \mid \tilde{f}_1^K) \end{aligned} \quad \text{公式 3.3}$$

为了减少符号表示，我们忽略了对隐变量 B 的显式依赖关系，认为 $\Pr(e_1^I \mid B, f_1^J) = \Pr(\tilde{e}_1^K \mid \tilde{f}_1^K)$。我们将 $\Pr(B \mid f_1^J)$ 称为短语划分模型。

之后，我们引入隐变量 $\tilde{a}_1^K$ 来表示源语言短语序列 $\tilde{f}_1^K$ 和目标语言短语序列 $\tilde{e}_1^K$ 之间的对应关系：

$$\begin{aligned} \Pr(\tilde{e}_1^K \mid \tilde{f}_1^K) &= \sum_{\tilde{a}_1^K} \Pr(\tilde{e}_1^K, \tilde{a}_1^K \mid \tilde{f}_1^K) \\ &= \sum_{\tilde{a}_1^K} \Pr(\tilde{a}_1^K \mid \tilde{f}_1^K) \Pr(\tilde{e}_1^K \mid \tilde{a}_1^K, \tilde{f}_1^K) \\ &= \sum_{\tilde{a}_1^K} \Pr(\tilde{a}_1^K \mid \tilde{f}_1^K) \prod_{k=1}^{K} \Pr(\tilde{e}_k \mid \tilde{f}_{\tilde{a}_k}) \end{aligned} \quad \text{公式 3.4}$$

[5] 我们采用[Och, 2002]中对基于短语的翻译模型的形式化定义，但对 $\Pr(e_1^I \mid f_1^J)$ 建模，而不是 $\Pr(f_1^J \mid e_1^I)$。

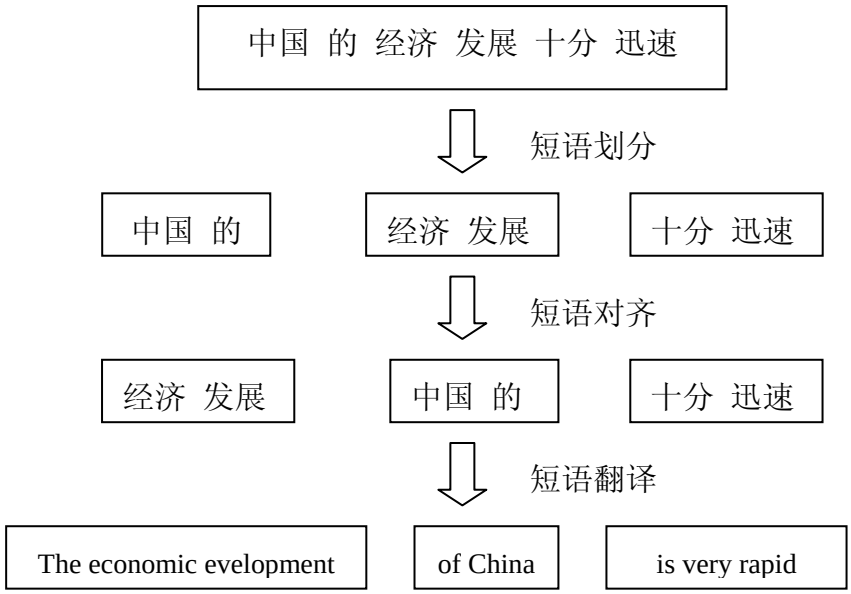

图 3.1：短语划分、短语对齐和短语翻译。

我们将 $\Pr\left(\tilde{a}_1^K \mid \tilde{f}_1^K\right)$ 称为短语对齐模型，将 $\Pr\left(\tilde{e}_k \mid \tilde{f}_{a_k}\right)$ 称为短语翻译模型。

因此，基于短语的翻译模型主要可分为三个子模型：

1. 短语划分模型 $\Pr\left(B \mid f_1^J\right)$
2. 短语对齐模型 $\Pr\left(\tilde{a}_1^K \mid \tilde{f}_1^K\right)$
3. 短语翻译模型 $\Pr\left(\tilde{e}_k \mid \tilde{f}_{a_k}\right)$

下面，我们以图 3.1 为例来说明基于短语的模型的翻译过程。已知汉语句子“中国 的 经济 发展 十分 迅速”，短语划分模型将这个句子分成 3 个汉语短语“中国 的 | 经济 发展 | 十分 迅速”。短语对齐模型设定第 1 个英语短语对应第 2 个汉语短语，第 2 个英语短语对应第 1 个汉语短语，第 3 个英语短语对应着第 3 个汉语短语。短语翻译模型将第 2 个汉语短语翻译成“The economic development”，将第 1 个汉语短语翻译成“of China”，将第 3 个汉语短语翻译成“is very rapid”，从而得到整个汉语句子的译文“The economic development of China is very rapid”。

直观上看，我们也能发现短语对齐是最困难的部分。

3.1.2 短语划分

通常，基于短语的翻译模型将任意连续的词串都视作为短语。因此，对于一

表 3.1：短语划分例子。

编号	短语划分
1	中国 的 经济 发展
2	中国 | 的 经济 发展
3	中国 的 | 经济 发展
4	中国 的 经济 | 发展
5	中国 | 的 | 经济 发展
6	中国 | 的 经济 | 发展
7	中国 的 | 经济 | 发展
8	中国 | 的 | 经济 | 发展

个包含 n 个词的句子，所有可能的短语划分的数量是 2^{n-1}。例如，表 3.1 列出了汉语句子“中国 的 经济 发展”所有可能的短语划分。

目前，最常见的假设是所有短语划分的概率是相等的[Och 2002b]。

很显然，词串“的 经济”并非语言学意义上的短语。在本章，我们将短语分为两类：句法短语和非句法短语。所谓句法短语，是指能被句法树的某棵子树覆盖的词串。反之则是非句法短语。

图 3.2 给出了汉语句子“中国 的 经济 发展”的句法分析树。在前面的定义下，“中国 的”和“经济 发展”是句法短语，而“中国 的 经济”和“的 经济”是非句法短语。

[Koehn 2003]探讨了句法双语短语在机器翻译中的作用。Koehn 等人首先对双语语料库作词语对齐，然后对源语言和目标语言都进行句法分析。他们认为，当且仅当一个双语短语满足对齐一致性并且源语言短语和目标语言短语都分别被句法子树覆盖时，它才能被称为是句法双语短语。他们将这些句法双语短语用于翻译中，翻译质量要远比使用全部双语短语差。因此，[Koehn 2003]认为只使用句法双语短语并不能带来翻译质量的提高。

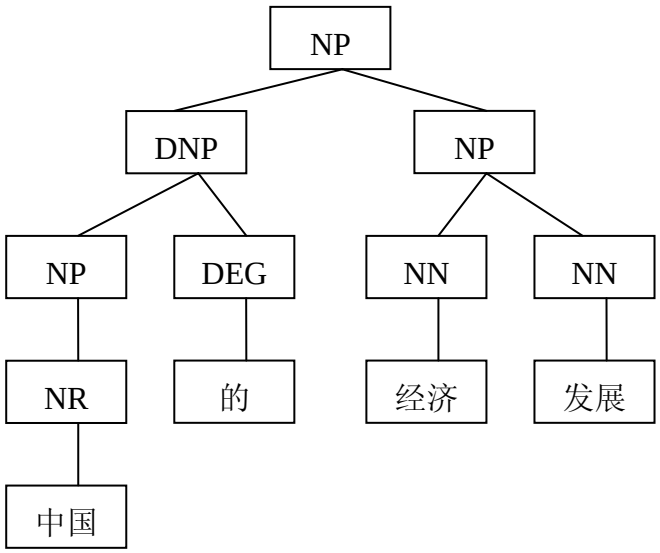

图 3.2：句法树例子。

3.1.3 短语对齐

短语对齐通常被称为短语重排序（reordering），即调整目标语言短语的顺序。短语重排序在基于短语的翻译模型的三个子模型中是最关键、最困难的，是基于短语的模型的研究热点。

研究人员在短语重排序问题上做了许多工作。[Och 2002]提出对跳转长度进行惩罚。[Zens 2004]研究了 IBM 限制和 ITG 限制。[Tillmann 2005]将双语短语定义为块，抽取左、中、右三种方位（orientation）。[Xiong 2006]提出基于最大熵的短语重排序模型，[Al-Onaizan 2006]提出 outbound、inbound 和 pairwise 三种扭曲模型。

由于这些工作都没有利用句法信息，一些研究人员开始尝试将句法信息引入到基于短语的翻译模型中。[Xia 2004]和[Collins 2005]都利用句法信息对源语言句子做前处理，然后用基于短语的系统翻译。他们都只将句法信息作为改进基于短语的系统的辅助手段，也没有获得显著的效果。

3.1.4 短语翻译

就目前为止，短语翻译的方法主要有两种。一种是直接翻译法，另一种是对齐模板。绝大多数基于短语的系统都是采用直接翻译法，即直接将双语短语中的目标语言短语作为译文。Franz Och 的对齐模板利用词类来代替词语以期获得泛化功能。然而，双语聚类很难取得理想的效果。在 2006 年 NIST 机器翻译评测中，Och 也采用了直接翻译法而不再使用对齐模板。

3.2 模型

在本节，我们将介绍嵌入句法树的基于短语的翻译模型，也称之为模型 1。

3.2.1 形式化定义

我们首先引入源语言句子的句法树$T\left(f_1^J\right)$作为隐变量：

$$\begin{aligned}\Pr\left(e_1^I \mid f_1^J\right) &= \sum_{T\left(f_1^J\right)} \Pr\left(e_1^I, T\left(f_1^J\right) \mid f_1^J\right) \\ &= \sum_{T\left(f_1^J\right)} \Pr\left(T\left(f_1^J\right) \mid f_1^J\right) \Pr\left(e_1^I \mid T\left(f_1^J\right), f_1^J\right)\end{aligned} \quad \text{公式 3.5}$$

我们将$\Pr\left(T\left(f_1^J\right) \mid f_1^J\right)$称为句法分析模型。

然后，假设存在一个隐变量B将源语言句子和目标语言句子同时划分为K个短语：

$$\begin{aligned}\Pr\left(e_1^I \mid T\left(f_1^J\right), f_1^J\right) &= \sum_B \Pr\left(e_1^I, B \mid T\left(f_1^J\right), f_1^J\right) \\ &= \sum_B \Pr\left(B \mid T\left(f_1^J\right), f_1^J\right) \Pr\left(e_1^I \mid B, T\left(f_1^J\right), f_1^J\right) \\ &= \sum_B \Pr\left(B \mid T\left(f_1^J\right), f_1^J\right) \Pr\left(\tilde{e}_1^K \mid T\left(f_1^J\right), \tilde{f}_1^K\right)\end{aligned} \quad \text{公式 3.6}$$

我们将$\Pr\left(B \mid T\left(f_1^J\right), f_1^J\right)$成为短语划分模型。

之后，我们引入隐变量$\tilde{a}_1^K$来表示源语言短语序列$\tilde{f}_1^K$和目标语言短语序列$\tilde{e}_1^K$之间的对应关系：

$$\begin{aligned}\Pr\left(\tilde{e}_1^K \mid T\left(f_1^J\right), \tilde{f}_1^K\right) &= \sum_{\tilde{a}_1^K} \Pr\left(\tilde{e}_1^K, \tilde{a}_1^K \mid T\left(f_1^J\right), \tilde{f}_1^K\right) \\ &= \sum_{\tilde{a}_1^K} \Pr\left(\tilde{a}_1^K \mid T\left(f_1^J\right), \tilde{f}_1^K\right) \Pr\left(\tilde{e}_1^K \mid \tilde{a}_1^K, T\left(f_1^J\right), \tilde{f}_1^K\right) \\ &= \sum_{\tilde{a}_1^K} \Pr\left(\tilde{a}_1^K \mid T\left(f_1^J\right), \tilde{f}_1^K\right) \prod_{k=1}^K \Pr\left(\tilde{e}_k \mid T\left(f_1^J\right), \tilde{f}_{\tilde{a}_k}\right)\end{aligned} \quad \text{公式 3.7}$$

我们将$\Pr\left(\tilde{a}_1^K \mid T\left(f_1^J\right), \tilde{f}_1^K\right)$称为短语对齐模型，将$\Pr\left(\tilde{e}_k \mid T\left(f_1^J\right), \tilde{f}_{\tilde{a}_k}\right)$称为短语翻译模型。

因此，嵌入句法树的基于短语的翻译模型可以分为四个子模型：

1. 句法分析模型$\Pr\left(T\left(f_1^J\right) \mid f_1^J\right)$

2. 短语划分模型 $\Pr\left(B|T\left(f_1^J\right),f_1^J\right)$

3. 短语对齐模型 $\Pr\left(\tilde{a}_1^K|T\left(f_1^J\right),\tilde{f}_1^K\right)$

4. 短语翻译模型 $\Pr\left(\tilde{e}_k|T\left(f_1^J\right),\tilde{f}_{\tilde{a}_k}\right)$

由于句法分析模型属于句法分析的任务，在此我们不予讨论。而短语翻译我们也采用直接翻译法，也不予讨论。

我们的短语划分模型 $\Pr\left(B|T\left(f_1^J\right),f_1^J\right)$ 与传统的短语划分模型相比，最大的不同之处在于嵌入了一颗句法树。这样，我们就可以利用句法信息来区分句法短语和非句法短语。

我们将在下一节重点讨论如何利用句法信息实现短语重排序。

3.2.2 树节点重排序

与传统的短语对齐模型 $\Pr\left(\tilde{a}_1^K|\tilde{f}_1^K\right)$ 相比，我们的短语对齐模型 $\Pr\left(\tilde{a}_1^K|T\left(f_1^J\right),\tilde{f}_1^K\right)$ 嵌入了一棵句法树，因此可以利用句法信息指导短语重排序。

我们提出树节点重排序（Tree Node Reordering，简称 TNR）。TNR 是一个二元组 $\tau=(T,R)$，其中，T 是一个只包含非终结符的源语言句法树，R 是一个整数向量，其大小等于 T 的叶子数，描述了叶子节点位置的重排序。

图 3.3 给出了一个 TNR 的图形化表示。它的含义是第 1 个叶子节点 NP 所覆盖的源语言词串将被翻译成第 3 个目标语言词串，第 2 个叶子节点 DEG 所覆盖的源语言词串将被翻译成第 2 个目标语言词串，第 3 个叶子节点 NP 所覆盖的源语言词串将被翻译成第 1 个目标语言词串。该 TNR 的形式化表示为：

$$\left("\left(NP\left(DNP\left(NP\right)\left(DEG\right)\right)\left(NP\right)\right)",[3\ 2\ 1]\right)$$

理论上，一个 TNR 就可以描述整个句子的短语重排序情况。然而，当句子较长时，句法树结构也会比较复杂，仅用一个 TNR 来描述整个句子的短语重排序情况是不现实的。因为不仅参数估计的复杂度会极高，搜索时这样一个 TNR 也很难完全匹配上。因此，我们通常采用多个 TNR 来描述，即：

$$\Pr\left(\tilde{a}_1^K|T\left(f_1^J\right),\tilde{f}_1^K\right)=\prod_{n=1}^{N}\Pr\left(\tau_n\right) \qquad \text{公式 3.8}$$

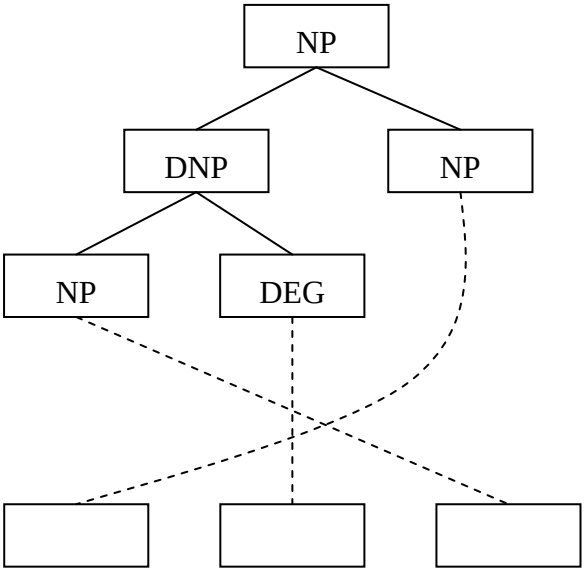

图 3.3：TNR 的图形化表示

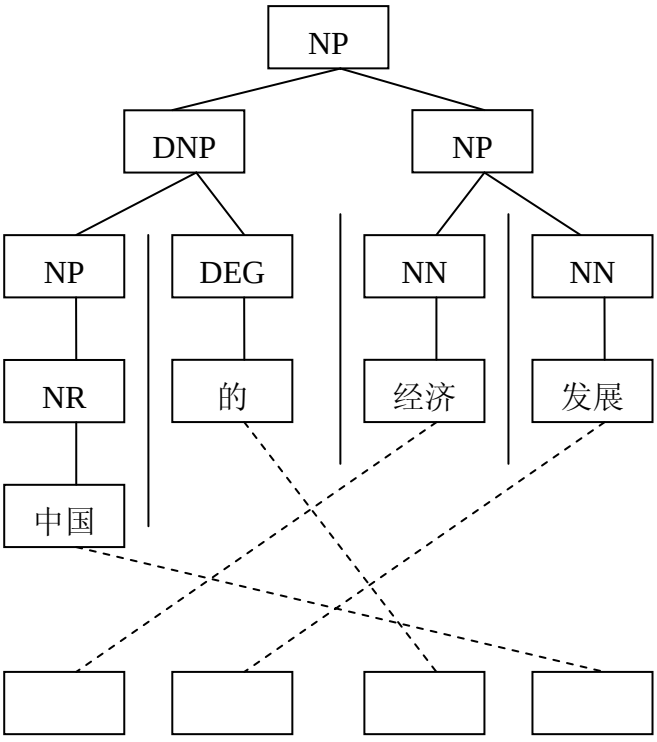

图 3.4：短语重排序例子。

例如，在图 3.4 中，汉语句子被切分成四个短语：“中国 | 的 | 经济 | 发展”，短语对齐是$a_1=3,a_2=4,a_3=2,a_4=1$。该短语对齐可以使用表 3.2 所示的 TNR 序列来表示。

表 3.2：TNR 序列

编号	TNR
1	$\left("\left(NP\left(DNP\right)\left(NP\right)\right)",[2\ 1]\right)$
2	$\left("\left(DNP\left(NP\left(NR\right)\right)\left(DEG\right)\right)",[2\ 1]\right)$
3	$\left("\left(NP\left(NN\right)\left(NN\right)\right)",[1\ 2]\right)$

理论上，只要全部采用句法短语，所有的短语对齐都能够被 TNR 序列所表示。

3.2.3 对数线性模型特征设计

对数线性模型目前已经成为统计机器翻译领域的标准模型框架[Och 2002a]。我们同样也采用这种框架：

$$\begin{aligned}\Pr\left(e_1^I \mid f_1^J\right) &= p_{\lambda_1^M}\left(e_1^I \mid f_1^J\right) \\ &= \frac{\exp\left[\sum_{m=1}^{M}\lambda_m h_m\left(e_1^I, f_1^J\right)\right]}{\sum_{\tilde{e}_1^I}\exp\left[\sum_{m=1}^{M}\lambda_m h_m\left(\tilde{e}_1^I, f_1^J\right)\right]}\end{aligned} \qquad \text{公式 3.9}$$

相应地，搜索公式为：

$$\begin{aligned}\hat{e}_1^I &= \arg\max_{e_1^I}\left\{\Pr\left(e_1^I \mid f_1^J\right)\right\} \\ &= \arg\max_{e_1^I}\left\{\sum_{m=1}^{M}\lambda_m h_m\left(e_1^I, f_1^J\right)\right\}\end{aligned} \qquad \text{公式 3.10}$$

针对嵌入句法树的基于短语的翻译模型，我们设计特征函数如下：

1. 源语言到目标语言短语翻译概率：$h_{pfe}\left(f_1^J, e_1^I\right)=\sum_{k=1}^{K}\log\left(p\left(\tilde{e}_k \mid \tilde{f}_{\tilde{a}_k}\right)\right)$

2. 目标语言到源语言短语翻译概率：$h_{pef}\left(f_1^J, e_1^I\right)=\sum_{k=1}^{K}\log\left(p\left(\tilde{f}_{\tilde{a}_k} \mid \tilde{e}_k\right)\right)$

3. 源语言到目标语言词汇化权重：$h_{lfe}\left(f_1^J,e_1^I\right)=\sum_{k=1}^{K}\log\left(lex\left(\tilde{e}_k\mid\tilde{f}_{\tilde{a}_k}\right)\right)$

4. 目标语言到源语言词汇化权重：$h_{lef}\left(f_1^J,e_1^I\right)=\sum_{k=1}^{K}\log\left(lex\left(\tilde{f}_{\tilde{a}_k}\mid\tilde{e}_k\right)\right)$

5. TNR 概率：$h_{tnr}\left(f_1^J,e_1^I\right)=\sum_{n=1}^{N}\log\left(p\left(\tau_n\right)\right)$

6. 短语数量：$h_{pc}\left(f_1^J,e_1^I\right)=K$

7. 词语数量：$h_{wc}\left(f_1^J,e_1^I\right)=I$

8. TNR 数量：$h_{tc}\left(f_1^J,e_1^I\right)=N$

9. 三元语言模型：$h_{lm}\left(f_1^J,e_1^I\right)=\sum_{i=1}^{I}\log\left(p\left(e_i\mid e_{i-2},e_{i-1}\right)\right)$

3.3 训练

3.3.1 抽取算法

本节介绍如何从真实数据中抽取 TNR。

TNR 抽取算法的输入是一个经过词语对齐和源语言句法分析的双语句对$\left(T\left(f_1^J\right),e_1^I,A\right)$，输出是 TNR 频度表 Υ。

抽取算法首先后序遍历源语言句法树，对每个树节点进行编号。我们设计了一个数据结构 TNR 栈向量，来存储每个树节点抽出的 TNR。为了抽取 TNR，首先必须确定样本，我们称之为（树，串，对齐）三元组，简称为三元组。每个树节点对应着唯一的三元组。确定三元组后，我们还必须检查对齐一致性。如果不满足对齐一致性，则不能抽取 TNR。对于叶子节点的三元组，TNR 的抽取是非常简单的，只保留节点的标记，重排序设为 1，这样的 TNR 被称为最小 TNR。如果三元组中的树的高度大于 1，则需先计算基准 TNR，然后再根据孩子节点已经抽取的 TNR，组合构造该节点的 TNR。最后，将 TNR 从栈向量中导出为频度表 Υ。图 3.5 给出了 TNR 抽取算法。

下面，我们将详细介绍抽取算法的几个重要概念和函数：

1. （树，串，对齐）三元组的确定
2. 对齐一致性
3. 计算基准 TNR
4. 组合构造 TNR

输入：源语言句法树 $T\left(f_1^J\right)$，目标语言句子 e_1^I，词语对齐 A

[1] $\Upsilon := \phi$

[2] 后序遍历源语言句法树 $T\left(f_1^J\right)$，对每个节点进行编号。

[3] 初始化 TNR 栈向量 *tnrStackVec*

[4] for $id := 1$ to $nodeCount\left(T\left(f_1^J\right)\right)$

[5] 确定与编号 *id* 对应的（树，串，对齐）三元组：

$$\left(\tilde{T},\tilde{S},\tilde{A}\right) = identify\left(id, T\left(f_1^J\right), e_1^I, A\right)$$

[6] 如果三元组不满足对齐一致性，则继续。

[7] 如果与编号 *id* 对应的是叶子节点，则构造 TNR 并压入栈中：

$$tnrStackVec[id] \leftarrow \left("\left(leafLabel\right)", [1]\right)$$

[8] 否则

[9] 计算基准 TNR：$\tau_0 = getBaseTNR\left(\tilde{T},\tilde{S},\tilde{A}\right)$

[10] 根据基准 TNR 和孩子节点已经抽取的 TNR 构造该节点的 TNR：

$$tnrStackVec\left[id\right] := build\left(\tau_0, tnrStackVec\right)$$

[11] end for

[12] 将 TNR 从栈向量中导出到频度表：$\Upsilon := \text{export}\left(tnrStackVec\right)$

输出：TNR 频度表 Υ

图 3.5：TNR 抽取算法

3.3.2 确定三元组

已知编号，很容易找到相应的树节点，以及相应的子树 $\tilde{T}$。而串 $\tilde{S}$ 和对齐 $\tilde{A}$ 的确定则需要通过句对之间的对齐 A 来确定。

例如，在图 3.6 的训练样本中，已知编号 3，可以确定子树 $\tilde{T}$ 为 $\left(NP\left(NR\ 布什\right)\left(NN\ 总统\right)\right)$，源语言的词位置范围是[1，2]，通过句对之间的对齐 A 可以确定对应的串 $\tilde{S}$ 为“President Bush”，对齐 $\tilde{A}$ 为 $\left\{1:2, 2:1\right\}$。

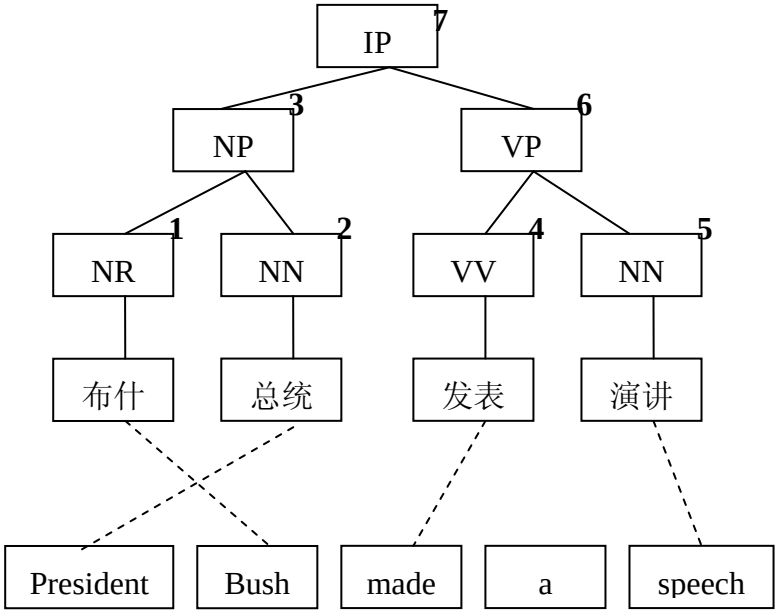

图 3.6：训练样本

我们要求串 $\tilde{S}$ 的第一个词和最后一个词必须对应到某个源语言词。因此，与编号 4 对应的串 $\tilde{S}$ 是“made”而不是“made a”。这样做的好处是每个编号都对应着唯一的三元组。

表 3.3：训练样本的所有的三元组

编号	树	串	对齐
1	(NR 布什)	Bush	1:1
2	(NN 总统)	President	1:1
3	(NP (NR 布什) (NN 总统))	President Bush	1:2 2:1
4	(VV 发表)	made	1:1
5	(NN 演讲)	speech	1:1
6	(VP (VV 发表) (NN 演讲))	made a speech	1:1 2:3
7	(IP (NP (NR 布什) (NN 总统)) (VP (VV 发表) (NN 演讲)))	President Bush made a speech	1:2 2:1 3:3 4:5

表 3.3 列出了图 3.6 的训练样本的所有的三元组。

输入：三元组 $(\tilde{T}, \tilde{S}, \tilde{A})$

[1] 对 $\tilde{T}$ 的根节点的每个直接孩子节点，计算对应的目标语言的跨度 $span_i$ [2] 如果存在两个孩子的跨度有重叠，则退出： $\exists i, j : \left(span_i.begin - span_j.end\right) \times \left(span_i.end - span_j.begin\right) \le 0$ [3] 只保留 $\tilde{T}$ 的根节点及其直接孩子，得到树 T [4] 对跨度进行排序，得到孩子节点的重排序 R [5] 构造基准 TNR： $\tau_0 = (T, R)$
输出：基准 TNR τ_0

图 3.7：计算基准 TNR

3.3.3 对齐一致性

所谓对齐一致性是指子树 $\tilde{T} = T\left(f_j^{j+m}\right)$ 和串 $\tilde{S} = e_i^{i+n}$ 必须与词语对齐 A 保持一致：

$$\begin{aligned} &\forall (i', j') \in A : j \le j' \le j+m \leftrightarrow i \le i' \le i+n \\ &\wedge \exists (i', j') \in A : j \le j' \le j+m \wedge i \le i' \le i+n \end{aligned} \quad \text{公式 3.11}$$

也就是说，子树的所有叶子节点 $\tilde{T} = T\left(f_j^{j+m}\right)$ 都必须连向串 $\tilde{S} = e_i^{i+n}$ 的目标语言词，而串的所有目标语言词都必须连向子树的所有叶子节点，并且子树和串之间至少有一条连线。

上述的三元组确定方法能够保证子树的所有叶子节点 $\tilde{T} = T\left(f_j^{j+m}\right)$ 都连向串 $\tilde{S} = e_i^{i+n}$ 的目标语言词，但是并不能保证串的所有目标语言词都连向子树的所有叶子节点。因此，必须进行对齐一致性的检查。

表 3.3 列出的三元组均满足对齐一致性。

3.3.4 计算基准 TNR

如果三元组中的树的层高只有 1，那么 TNR 是很容易获取的：只保留根节点的标记，重排序设定为 1。我们将这种 TNR 称为最小 TNR。对于层高大于 1 的三元组，我们也计算最小 TNR。虽然最小 TNR 实际上是没有意义的重排序，但对于抽取算法是必要的。在导出到 TNR 频度表时，这种 TNR 将被摒弃。对于层

高大于 1 的三元组，必须先计算基准 TNR，然后再组合孩子节点的 TNR 得到父亲节点的 TNR。

图 3.7 给出了计算基准 TNR 的算法。每个基准 TNR 的树层高均为 2，通过第二层节点的跨度来获得重排序信息。所谓跨度，是指目标语言串的起止范围。以标号 6 对应的三元组为例，第二层节点 VV 对应的跨度是[1,1]，节点 NN 对应的跨度是[3,3]，对跨度进行排序后节点的顺序依然是[VV，NN]，因此 $R=[1,2]$。需要注意，未被对齐的目标语言词（如图 3.6 中的“a”）对计算重排序被忽略了。如果孩子节点的跨度有重叠，则无法计算基准 TNR。

表 3.4：几个三元组的基准 TNR

编号	基准 TNR
3	$\left("\left(NP\left(NR\right)\left(NN\right)\right)",[2\ 1]\right)$
6	$\left("\left(VP\left(VV\right)\left(NN\right)\right)",[1\ 2]\right)$
7	$\left("\left(IP\left(NP\right)\left(VP\right)\right)",[1\ 2]\right)$

表 3.4 列出了表 3.3 中几个三元组的基准 TNR。

3.3.4 组合构造 TNR

已知基准 TNR 和孩子节点已经抽出的 TNR，就可以组合构造出父亲节点的所有 TNR。

例如，编号为 7 的节点的基准 TNR 是$\left("\left(IP\left(NP\right)\left(VP\right)\right)",[1\ 2]\right)$，它的两个孩子节点的编号分别是 3 和 6。

编号为 3 的节点已经抽出以下 TNR：

$\left("\left(NP\right)",[1]\right)$

$\left("\left(NP\left(NR\right)\left(NN\right)\right)",[2\ 1]\right)$

编号为 6 的节点已经抽出以下 TNR：

$\left("\left(VP\right)",[1]\right)$

$\left("\left(VP\left(VV\right)\left(NN\right)\right)",[1\ 2]\right)$

在基准 TNR$\left("\left(IP\left(NP\right)\left(VP\right)\right)",[1\ 2]\right)$的基础上，将这 4 个 TNR 进行组合，就可以得到父亲节点的 4 个 TNR。再加上最小 TNR，总共是 5 个 TNR。

$\left("\left(IP\left(NP\right)\left(VP\right)\right)",[1\ 2]\right)$

$\left("\left(IP\left(NP\left(NR\right)\left(NN\right)\right)\left(VP\right)\right)",[2\ 1\ 3]\right)$

$\left("\left(IP\left(NP\right)\left(VP\left(VV\right)\left(NN\right)\right)\right)",[1\ 2\ 3]\right)$

$\left("\left(IP\left(NP\left(NR\right)\left(NN\right)\right)\left(VP\left(VV\right)\left(NN\right)\right)\right)",[2\ 1\ 3\ 4]\right)$

表 3.5：TNR 栈向量的内容

编号	TNR 栈
1	(“(NR)”, [1])
2	(“(NN)”, [1])
3	(“(NP)”, [1]) (“(NP (NR) (NN))”, [2 1])
4	(“(VV)”, [1])
5	(“(NN)”, [1])
6	(“(VP)”, [1]) (“(VP (VV) (NN))”, [1 2])
7	(“(IP)”, [1]) (“(IP (NP) (VP))”, [1 2]) (“(IP (NP (NR) (NN)) (VP))”, [2 1 3]) (“(IP (NP) (VP (VV) (NN)))”, [1 2 3]) (“(IP (NP (NR) (NN)) (VP (VV) (NN)))”, [2 1 3 4])

表 3.5 给出了最终 TNR 栈向量的内容。最后的步骤是将 TNR 从栈向量中导出到 TNR 频度表中，在这个过程中，最小 TNR 将被摒弃。

表 3.6：与图 3.6 中的训练样本对应的 TNR 表。

TNR	频度	概率
(“(NP (NR) (NN))”, [2 1])	1	1.0
(“(VP (VV) (NN))”, [1 2])	1	1.0
(“(IP (NP) (VP))”, [1 2])	1	1.0
(“(IP (NP (NR) (NN)) (VP))”, [2 1 3])	1	1.0
(“(IP (NP) (VP (VV) (NN)))”, [1 2 3])	1	1.0
(“(IP (NP (NR) (NN)) (VP (VV) (NN)))”, [2 1 3 4])	1	1.0

输入：源语言句法树 $T\left(f_1^J\right)$

[1] 后序遍历源语言句法树 $T\left(f_1^J\right)$，并对每个节点设定编号。

[2] 初始化候选栈向量 *candStackVec*

[3] for $id := 1$ to $nodeCount\left(T\left(f_1^J\right)\right)$

[4] 确定与 *id* 对应的子树 T' 及源语言短语 $\tilde{f}$

[5] 如果能在双语短语表中查到 $\tilde{f}$ 的译文

[6] 对每个译文构造候选翻译，并压入到 *candStackVec*[*id*]中

[7] 否则

[8] 如果 T' 是叶子节点

[9] 构造默认翻译，并压入到 *candStackVec*[*id*]中

[10] 否则

[11] 针对子树 T'，在 TNR 表中搜索可用的 TNR

[12] 如果在 TNR 表中找不到可用的 TNR，构造默认 TNR

[13] 对每个可用的 TNR 构造候选翻译，并压到 *candStackVec*[*id*]中

[14] end for

[15] 在 *candStackVec*[*nodeCount*]中找到概率最大的翻译，并提取最佳译文 $\hat{e}_1^I$

输出：译文 $\hat{e}_1^I$

图 3.8：搜索算法

3.3.5 概率估计

TNR 的概率计算方法如下：

$$p(\tau)=\frac{count(\tau)}{\sum_{\tau'} count(\tau')\times\delta\left(T(\tau),T(\tau')\right)} \qquad \text{公式 3.12}$$

其中，$count(\tau)$是指 TNR 在语料库中出现的次数，$T(\tau)$指的是 TNR 中的树。

表 3.6 列出了从图 3.6 的训练样本中最终得到的 TNR 表。由于只有一个样本，TNR 均只出现一次，概率也均为 1。

3.4 搜索

3.4.1 搜索算法

与通常的基于短语的解码器不同的是，我们的解码器的输入是源语言句法树，而不是源语言句子。

我们采用自底向上的柱搜索（beam search）算法。后序遍历输入的源语言句法树，对每个节点所对应的源语言短语搜索候选翻译。当处理完根节点后，就得到整个句子的翻译。

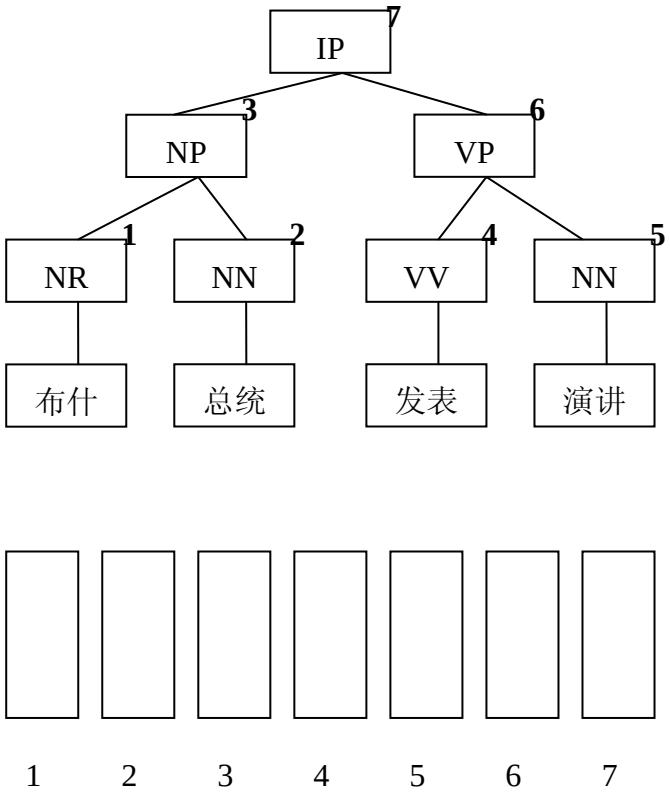

图 3.9：候选栈向量。

候选翻译包含以下信息：

1. 译文
2. TNR 序列
3. 英汉 MLE 概率累积特征值
4. 英汉词汇权重累积特征值
5. 汉英 MLE 概率累积特征值
6. 汉英词汇权重累积特征值
7. 短语数量累积特征值
8. 语言模型累积特征值
9. 词语数量累积特征值
10. TNR 概率累积特征值
11. TNR 频度累积特征值
12. TNR 数量累积特征值
13. 分值
14. 被重合并项

记录累积特征值的目的在于方便进行最小错误率训练[Och 2003a]。

图 3.8 给出了搜索算法。每个子树对应的候选翻译都被存放在栈中，这些栈被组织成向量的形式，被称为候选栈向量。如图 3.9 所示，7 个节点对应着 7 棵子树，所以有 7 个候选栈。

接下来，我们将重点讨论以下几个问题：

1. 默认翻译和 TNR
2. TNR 的可用性
3. 根据 TNR 构造候选翻译
4. 剪枝策略

3.4.2 默认翻译和 TNR

在搜索过程中，无法找到可用的双语短语和 TNR 是十分常见的事。为了保证搜索能够顺利进行，我们提出默认翻译和默认 TNR。

如果源语言句法树的某个叶子节点（也就是某个源语言词）无法在双语短语表中找到译文，则将源语言词本身作为译文，我们称之为默认翻译。例如，如果在双语短语表中找不到“滥觞”这个词的候选翻译，则将“滥觞”本身作为默认

翻译。

如果针对某个句法子树，无法在 TNR 表中找到可用（我们将在下节介绍可用性）的 TNR，则按照以下方法构造默认 TNR：树是该句法子树的根节点及其直接孩子，重排序是顺序。例如，假设对图 3.1 中的句法树无法找到可用的 TNR，则构造默认 TNR

$$\left("\left(NP\left(DNP\right)\left(NP\right)\right)",[1\ 2]\right)$$

3.4.3 TNR 的可用性

对于一个句法树T，一个 TNR 是可用的当且仅当：

1. $T\left(\tau\right)\subseteq T$，即 TNR 的树是该句法树的子图。
2. $root\left(T\left(\tau\right)\right)=root\left(T\right)$，即 TNR 的树的根节点和该句法树是相同的。

例如，针对图 3.2 中的句法树，以下 TNR 是可用的：

$$\left("\left(NP\left(DNP\right)\left(NP\right)\right)",[2\ 1]\right)$$

$$\left("\left(NP\left(DNP\right)\left(NP\left(NN\right)\left(NN\right)\right)\right)",[3\ 1\ 2]\right)$$

而以下 TNR 是不可用的：

$$\left("\left(NP\left(DNP\right)\left(NP\left(NN\right)\right)\right)",[1\ 2]\right)$$

$$\left("\left(IP\left(DNP\right)\left(NP\right)\right)",[1\ 2]\right)$$

给定一个句法树T，为了在 TNR 表中搜索所有可用的 TNR。我们首先枚举出句法树T的所有同根子树，然后再去 TNR 表中查找。

3.4.4 根据 TNR 构造候选翻译

已知一个 TNR，它的树的每个叶子节点都有相应的翻译，通过重排序信息可以组成新的翻译。

以图 3.9 中的句法树为例，假设当处理到根节点时，我们找到一个可用的 TNR：

$$\left("\left(IP\left(NP\left(NR\right)\left(NN\right)\right)\left(VP\right)\right)",[2\ 1\ 3]\right)$$

该 TNR 的树有三个叶子节点，分别对应着编号 1、2 和 6。假设栈 1 中已经

存储了 2 个翻译：“Bush”和“bush”。栈 2 中存储了 2 个翻译：“President”和“president”。栈 3 中存储了 2 个翻译：“made a speech”和“makes a talk”。根据重排序信息[2 1 3]，可以为根节点组合成 8 个译文：

1. President Bush made a speech
2. President bush made a speech
3. President Bush makes a talk
4. President bush makes a talk
5. president Bush made a speech
6. president bush made a speech
7. president Bush makes a talk
8. president bush makes a talk

由于搜索算法是自底向上执行的，处理任何一个节点时，它的孩子节点都已经被处理过了。因此，利用 TNR 就很容易通过组合孩子节点的翻译来拼成该节点的翻译。

可以看出，TNR 的作用就是对短语进行重排序，翻译的基本单元仍然是短语。因此，模型 1 从本质上说是基于短语的模型，只不过利用句法信息来指导短语重排序。

3.4.5 剪枝策略

我们引入各种剪枝策略，希望能减小搜索空间，提高搜索速度。

对于一个源语言短语，可以通过以下两种方式限制对应目标语言短语的数量：

1. 观察剪枝（observation pruning）。保留前 a 个概率最大的目标语言短语。
2. 阈值剪枝（threshold pruning）。假设目标语言短语中的最大概率是 $\hat{p}$，保留概率 $p \geq \hat{p} \times \alpha$ 的目标语言短语。

对于 TNR 表的剪枝也采用观察剪枝和阈值剪枝，但略有不同，即只对树相同的 TNR 进行剪枝。例如，已知句法树 T，我们找到 3 个树均为 T_1 的可用的 TNR，3 个树均为 T_2 的可用的 TNR。如果设定 $a = 2$ 进行观察剪枝，则树为 T_1 的 TNR 保

留 2 个，树为T_2的 TNR 也保留 2 个。因此，总共保留了 4 个 TNR，而不是 2 个。

对于候选栈，可以通过以下两种方式限制：

1. 柱状图剪枝（histogram pruning）。保留前b个概率最大的候选翻译。
2. 阈值剪枝。假设候选中的最大概率是$\hat{p}$，保留概率$p \geq \hat{p} \times \beta$的候选翻译。

3.5 讨论

在本章，我们介绍了嵌入句法树的基于短语的翻译模型。这个模型只使用句法双语短语，利用树节点重排序（TNR）来执行短语重排序。我们重点讨论了如何从经过词语对齐和源语言句法分析的双语语料库上自动抽取 TNR 并估计概率。我们还介绍了自底向上的柱搜索算法及剪枝策略。

在句法双语短语的定义上，我们与前人略有不同。[Koehn 2003]是在参数训练时确定句法双语短语的。他们对双语语料库做词语对齐，然后对源语言和目标语言都进行句法分析。他们认为，当且仅当一个双语短语满足对齐一致性并且两个短语都分别被句法子树覆盖时，它才能被称为是句法双语短语。我们则是在搜索过程中确定句法双语短语。在参数估计时，我们采用传统的短语抽取算法获得所有的双语短语。在搜索时，我们对输入的源语言句子做句法分析，认为当一个双语短语的源语言短语被句法子树覆盖时，它就被称为是句法双语短语，并不关心目标语言短语的情况。

嵌入句法树的基于短语的翻译模型的主要贡献在于，它是第一个在建模上利用句法信息指导短语重排序的基于短语的模型。虽然[Xia 2004]和[Collins 2005]都利用句法信息对短语进行重排序，他们都采用了前处理的手段，并没有建立真正的数学模型。

然而，有两个因素制约着嵌入句法树的基于短语的翻译模型的性能。

1. 只能使用句法双语短语。非句法短语对于机器翻译也是十分有用的，只使用句法双语短语会制约嵌入句法树的基于短语的翻译模型的性能。
2. 句法分析错误。无论是在分析训练语料还是测试语料，句法分析错误都难以避免。训练语料的分析错误会导致学习到错误的 TNR。测试语料的分析错误会导致正确的短语重排序永远无法被找到，因为短语树结构实际上决定了短语重排序的搜索空间。

第四章 基于树到串对齐模板的翻译模型

4.1 引言

基于短语的翻译模型[Marcu 2002; Koehn 2003; Och 2004b]对短语的翻译而不是单个词建模，超过了最初的 IBM 翻译模型[Brown 1993]，在近年来的机器翻译评测中连续取得领先成绩，被认为是当前统计机器翻译的主流方法。

在基于短语的模型中，短语通常是连续的词串而不是句法成分，擅长捕获局部重排序和翻译在训练语料库中出现频度高的词串。然而，基于短语的模型的一个重要缺陷在于难以处理短语间的重排序。通常，短语重排序是通过惩罚词位置偏移来实现的[Koehn 2004; Och 2004b]，很少或没有直接利用句法信息。

近年来，统计机器翻译的研究人员在基于句法的模型研究上取得了较大的进展。[Wu 1997]提出了反向转录语法（Inversion Transduction Grammar），将翻译过程视作利用同步语法对源语言和目标语言作双语句法分析。[Alshawi 2000]将平行依存分析树的每一步生成表示为中心转录机（head transducer）。[Melamed 2004]提出多文本语法（Multitext Grammar）进行翻译。[Graehl 2004]讨论了通用树到串和树到树转录机的训练和解码算法。[Chiang 2005]提出了层次化基于短语的翻译模型，实际上是使用了同步上下文无关语法。[Ding 2005]根据概率化同步依存插入语法提出一个基于句法的模型。这些模型虽然在具体细节上有所不同，但都利用同步语法或者基于树的转换规则同时对源语言和目标语言建模。

另外一类方法只利用了目标语言端的句法信息，将机器翻译视作句法分析问题。[Yamada 2001]对目标语言做句法分析，从而学习一系列将目标语言树转换成源语言串的操作的概率。

与之相反，[Quirk 2005]则更注重源语言分析。他们利用源语言依存分析器、目标语言词语切分器和无监督词语对齐器从平行语料库中学习 treelet 的翻译。

在本章，我们提出基于树到串对齐模板的统计翻译模型。树到串对齐模板描述了源语言句法树和目标语言串之间的对齐关系，既能生成终结符又能生成非终结符，既能执行局部重排序又能执行全局重排序。这个模型是语言学基于句法的（linguistically syntax-based），因为树到串对齐模板是从经过词语对齐和源语言句法分析的平行文本中自动获取的。为了翻译一个源语言句子，我们首先使用句法分析器得到句法树，然后再利用树到串对齐模板将该句法树转换成目标语言串。

我们的模型的一大优点能够自动获取树到串对齐模板，从而捕获语言学驱动（linguistically motivated）的局部（词）重排序和全局（短语、子句）重排序。此外，基于树到串对齐模板的统计翻译模型的训练复杂度要远低于树到树的模型。与[Galley 2004]类似，树到串对齐模板实际上也是转换规则，最大的区别在于我们是对源语言而不是目标语言建模。因此，在解码时，我们的任务是搜索概率最大的目标语言串，而[Galley 2004]则搜索概率最大的目标语言树。

4.2 模型

在本节，我们先介绍树到串对齐模板的定义，然后给出基于树到串对齐模板的翻译模型的形式化定义，最后讨论对数线性模型的特征设计问题。

4.2.1 树到串对齐模板

树到串对齐模板（简称为树模板，英文名 Tree-to-string Alignment Template，简称 TAT）z 是一个三元组$\langle \tilde{T}, \tilde{S}, \tilde{A} \rangle$，描述了源语言句法树$\tilde{T} = T\left(f_1^{J'}\right)$和目标语言串$\tilde{S} = e_1^{I'}$之间的对齐关系$\tilde{A}$。

在这里，我们用$T(\bullet)$来表示一棵句法树。为了避免引入更多的符号，我们用$T(z)$来表示树到串对齐模板 z 中的树。类似地，$S(z)$表示树到串对齐模板 z 中的串。

源语言串 $f_1^{J'}$ 是源语言句法树$T\left(f_1^{J'}\right)$的叶子节点序列，既可能包含终结符（词）也可能包含非终结符（词性标记或短语结构类）。目标语言串$e_1^{I'}$ 也同样既可能包含终结符（词）也可能包含非终结符（占位符）。对齐$\tilde{A}$被定义为源语言和目标语言符号位置的笛卡尔集的子集：

$$\tilde{A} \subseteq \{(j,i): j = 1,...,J'; i = 1,...,I'\} \qquad \text{公式 4.1}$$

我们将树到串对齐模板按照词汇化程度分为三类：

1. 词汇化：所有的源语言和目标语言符号均是终结符
2. 部分词汇化：源语言和目标语言符号既包含非终结符也包含终结符
3. 非词汇化：所有的源语言和目标语言符号均是非终结符

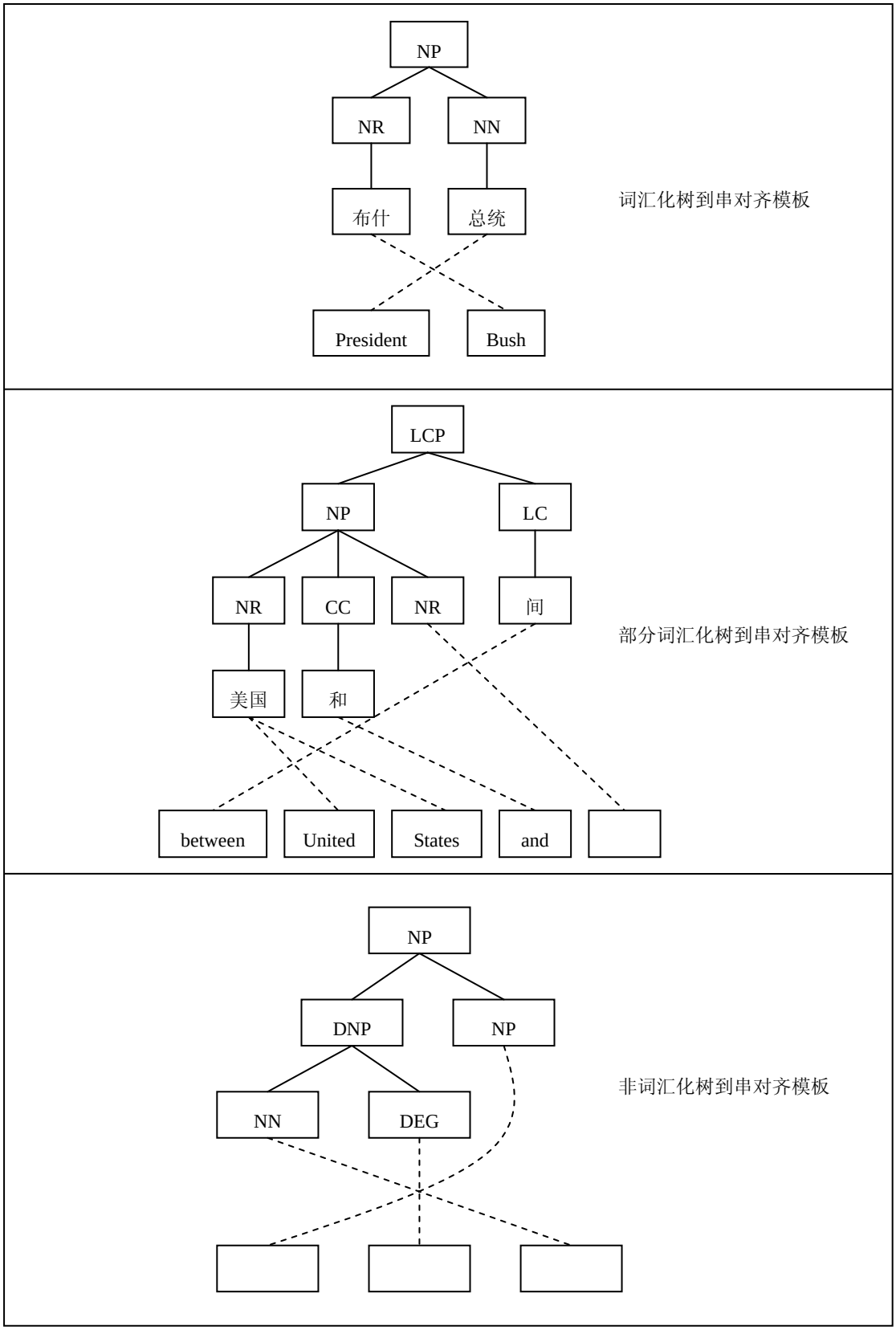

图 4.1：词汇化、部分词汇化和非词汇化树到串对齐模板。

图 4.1 分别给出了词汇化、部分词汇化和非词汇化树到串对齐模板。

在图形化表示中，目标语言非终结符用空白格来表示；在文本表示中，目标语言非终结符用 X 来表示，有时会加上编号来区别不同的非终结符。因此，图 4.1 中的三个树到串对齐模板也可以表示为：

$$\left\langle "\left(\mathrm{NP}\left(\mathrm{NR}\ 布什\right)\left(\mathrm{NN}\ 总统\right)\right)","\mathrm{President\ Bush}",\left\{(1,2),(2,1)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{LCP}\left(\mathrm{NP}\left(\mathrm{NR}\ 美国\right)\left(\mathrm{CC}\ 和\right)\left(\mathrm{NR}\right)\right)\left(\mathrm{LC}\ 间\right)\right)","\mathrm{between\ United\ States\ and}\ X ",\left\{(1,2),(1,3),(2,4),(3,5),(4,1)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{NP}\left(\mathrm{DNP}\left(\mathrm{NN}\right)\left(\mathrm{DEG}\right)\right)\left(\mathrm{NP}\right)\right)","X_1\ X_2\ X_3 ",\left\{(1,3),(2,2),(3,1)\right\}\right\rangle$$

4.2.2 形式化定义

本节我们将介绍利用树到串对齐模板建立翻译模型。

我们首先引入源语言句子的句法树 $T\left(f_1^J\right)$ 作为隐变量：

$$\begin{aligned}\Pr\left(e_1^I \mid f_1^J\right) &= \sum_{T\left(f_1^J\right)} \Pr\left(e_1^I, T\left(f_1^J\right) \mid f_1^J\right) \\ &= \sum_{T\left(f_1^J\right)} \Pr\left(T\left(f_1^J\right) \mid f_1^J\right) \Pr\left(e_1^I \mid T\left(f_1^J\right), f_1^J\right)\end{aligned} \quad \text{公式 4.2}$$

然后，我们引入隐变量 D 将源语言句法树 $T\left(f_1^J\right)$ 拆分成包含 K 棵子树的序列 $\tilde{T}_1^K$，这些子树按照其根节点在先序遍历中的顺序排序。我们假设每棵源语言子树 $\tilde{T}_k$ 都能生成一个目标语言串 $\tilde{S}_k$。因此，子树序列 $\tilde{T}_1^K$ 能够生成串序列 $\tilde{S}_1^K$，序列化合并 $\tilde{S}_1^K$ 就可以得到目标语言句子 e_1^I。

$$\begin{aligned}\Pr\left(e_1^I \mid T\left(f_1^J\right), f_1^J\right) &= \sum_D \Pr\left(e_1^I, D \mid T\left(f_1^J\right), f_1^J\right) \\ &= \sum_D \Pr\left(D \mid T\left(f_1^J\right), f_1^J\right) \Pr\left(e_1^I \mid D, T\left(f_1^J\right), f_1^J\right) \\ &= \sum_D \Pr\left(D \mid T\left(f_1^J\right), f_1^J\right) \Pr\left(\tilde{S}_1^K \mid \tilde{T}_1^K\right) \\ &= \sum_D \Pr\left(D \mid T\left(f_1^J\right), f_1^J\right) \prod_{k=1}^{K} \Pr\left(\tilde{S}_k \mid \tilde{T}_k\right)\end{aligned} \quad \text{公式 4.3}$$

我们假设 $\Pr\left(e_1^I \mid D, T\left(f_1^J\right), f_1^J\right) \equiv \Pr\left(\tilde{S}_1^K \mid \tilde{T}_1^K\right)$，因为目标语言句子 e_1^I 实际上是

由串序列 $\tilde{S}_1^K$ 推导生成的。为了减少符号表示，我们忽略了对隐变量 D 的显式依赖。

为了进一步分解 $\Pr(\tilde{S}|\tilde{T})$，我们引入隐变量树到串对齐模板 z：

$$\begin{aligned}\Pr(\tilde{S}|\tilde{T}) &= \sum_z \Pr(\tilde{S}, z|\tilde{T}) \\ &= \sum_z \Pr(z|\tilde{T}) \Pr(\tilde{S}|z, \tilde{T})\end{aligned} \quad \text{公式 4.4}$$

因此，基于树到串对齐模板的翻译模型包含四个子模型：

1. 句法分析模型：$\Pr(T(f_1^J)|f_1^J)$
2. 树拆分模型：$\Pr(D|T(f_1^J), f_1^J)$
3. 树模板选择模型：$\Pr(z|\tilde{T})$
4. 树模板使用模型：$\Pr(\tilde{S}|z, \tilde{T})$

图 4.2 显示了如何利用树到串对齐模板进行翻译。首先对源语言句子进行句法分析，得到句法树。然后，句法树被拆分成五个子树：

(NP(DNP(NP)(DEG 的))(NP))

(NP(NR 中国))

(NP(NN)(NN))

(NN 经济)

(NN 发展)

需要注意的是，这五棵子树的排列顺序是由它们的根节点在先序遍历中的顺序决定的。

对于每棵子树，我们都利用一个树到串对齐模板生成一个串。最后，系列化合并这些串就能得到最终的译文。系列化合并实际上是一个推导：

$$\begin{aligned}X_1 &\Rightarrow X_2 \text{ of } X_3 \\ &\Rightarrow X_2 \text{ of China} \\ &\Rightarrow X_3\ X_4 \text{ of China} \\ &\Rightarrow \text{economic } X_4 \text{ of China} \\ &\Rightarrow \text{economic development of China}\end{aligned}$$

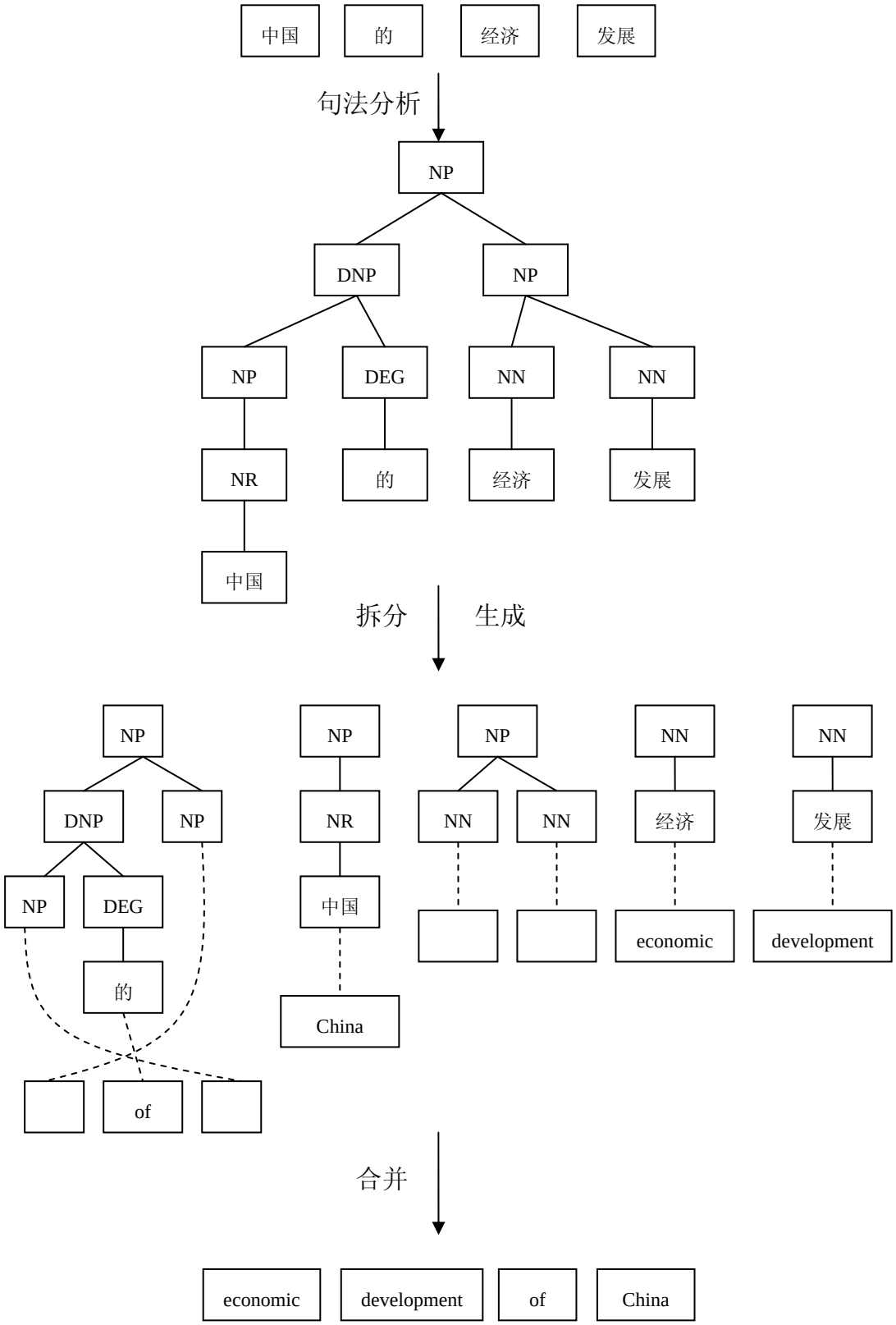

图 4.2：翻译过程的图形化表示

4.2.3 对数线性模型特征设计

我们将基于树到串对齐模板的翻译模型建立在对数线性模型框架上[Och 2002a]。因此，所有的知识源都被描述为依赖于源语言句子 f_1^J、目标语言句子 e_1^I 和可能的隐变量的特征函数。我们忽略了隐变量 $T\left(f_1^J\right)$，因为我们通常只使用句法分析器输出的最好结果。假设树拆分模型是等概率分布的，我们也忽略了隐变量 D。对于树模板使用模型，我们假定 $\Pr\left(\tilde{S} \mid z, \tilde{T}\right) = \delta\left(T(z), \tilde{T}\right)$。因此，我们真正所采用的是受限的模型，因为句法分析模型、树拆分模型和树模板使用模型都被简化了。

对数线性模型如下：

$$\begin{aligned} &\Pr\left(e_1^I, z_1^K \mid f_1^J\right) = p_{\lambda_1^M}\left(e_1^I, z_1^K \mid f_1^J\right) \\ &= \frac{\exp\left[\sum_{m=1}^{M} \lambda_m h_m\left(e_1^I, f_1^J, z_1^K\right)\right]}{\sum_{\tilde{e}_1^I} \exp\left[\sum_{m=1}^{M} \lambda_m h_m\left(\tilde{e}_1^I, f_1^J, z'^K_1\right)\right]} \end{aligned} \qquad \text{公式 4.5}$$

相应地，搜索公式为：

$$\begin{aligned} \hat{e}_1^I &= \arg\max_{e_1^I, z_1^K}\left\{\Pr\left(e_1^I, z_1^K \mid f_1^J\right)\right\} \\ &= \arg\max_{e_1^I, z_1^K}\left\{\sum_{m=1}^{M} \lambda_m h_m\left(e_1^I, f_1^J, z_1^K\right)\right\} \end{aligned} \qquad \text{公式 4.6}$$

为了方便与基于短语的系统 Pharaoh 做对比，我们设计了类似的特征函数：

1. 目标语言到源语言概率：

$$h_{pef}\left(e_1^I, f_1^J, z_1^K\right) = \sum_{k=1}^{K} \log\left(\frac{N(z) \cdot \delta\left(T(z), \tilde{T}_k\right)}{N\left(S(z)\right)}\right)$$

2. 源语言到目标语言概率：

$$h_{pfe}\left(e_1^I, f_1^J, z_1^K\right) = \sum_{k=1}^{K} \log\left(\frac{N(z) \cdot \delta\left(T(z), \tilde{T}_k\right)}{N\left(T(z)\right)}\right)$$

3. 目标语言到源语言词汇化权重：

$$h_{lef}\left(e_1^I, f_1^J, z_1^K\right) = \sum_{k=1}^{K} \log\left(lex\left(T(z) \mid S(z)\right) \cdot \delta\left(T(z), \tilde{T}_k\right)\right)$$

4. 源语言到目标语言词汇化权重：

$$h_{lfe}\left(e_1^I, f_1^J, z_1^K\right)=\sum_{k=1}^{K}\log\left(lex\left(S\left(z\right)|T\left(z\right)\right)\cdot\delta\left(T\left(z\right),\tilde{T}_k\right)\right)$$

5. 树模板数量：

$$h_{tc}\left(e_1^I, f_1^J, z_1^K\right)=K$$

6. 三元语言模型：

$$h_{lm}\left(e_1^I, f_1^J, z_1^K\right)=\sum_{i=1}^{I}\log\left(p\left(e_i|e_{i-2},e_{i-1}\right)\right)$$

7. 词语数量：

$$h_{wc}\left(e_1^I, f_1^J, z_1^K\right)=I$$

在计算词汇化权重[Koehn 2003]时，我们只考虑终结符。如果树模板中没有终结符，我们将特征值设为 1。我们用$lex\left(\cdot\right)$表示词汇化权重，K表示搜索过程中使用的树模板的数量，I表示最终译文的长度，也就是词语的数量。

4.3 训练

我们在第三章提到了树节点重排序（TNR），它实际上等价于非词汇化的树到串对齐模板（TAT）。因此，TAT 的抽取算法是非常类似于 TNR 的。我们不再赘述共同的地方，将重点放在两个区别较大的问题：计算基准 TAT 和组合构造 TAT。

4.3.1 抽取算法

本节介绍如何从真实数据中抽取 TAT。

TAT 抽取算法的输入是一个经过词语对齐和源语言句法分析的双语句对$\left(T\left(f_1^J\right),e_1^I,A\right)$，输出是 TAT 频度表$\Upsilon$。

抽取算法首先后序遍历源语言句法树，对每个树节点进行编号。我们设计了一个数据结构 TAT 栈向量来存储每个树节点抽出的 TAT。为了抽取 TAT，首先必须确定样本，我们称之为（树，串，对齐）三元组，简称为三元组。每个树节点对应着唯一的三元组。确定三元组后，我们还必须检查对齐一致性。如果不

输入：源语言句法树 $T\left(f_1^J\right)$，目标语言句子 e_1^I，词语对齐 A

[1] $\Upsilon := \phi$

[2] 后序遍历源语言句法树 $T\left(f_1^J\right)$，对每个节点进行编号。

[3] 初始化 TAT 栈向量 *tatStackVec*

[4] for $id := 1$ to $nodeCount\left(T\left(f_1^J\right)\right)$

[5] 确定与编号 *id* 对应的（树，串，对齐）三元组：

$$\left(\tilde{T},\tilde{S},\tilde{A}\right) = identify\left(id, T\left(f_1^J\right), e_1^I, A\right)$$

[6] 如果三元组不满足对齐一致性，则继续（continue）。

[7] 如果与编号 *id* 对应的是叶子节点，则构造 TNR 并压入栈中。

[8] 否则

[9] 计算基准 TAT： $z_0 = getBaseTAT\left(\tilde{T},\tilde{S},\tilde{A}\right)$

[10] 根据基准 TAT 和孩子节点已经抽取的 TAT 构造该节点的所有 TAT：

$$tatStackVec[id] := build\left(z_0, tatStackVec\right)$$

[11] end for

[12] 将 TAT 从栈向量中导出到频度表： $\Upsilon := \text{export}\left(tatStackVec\right)$

输出：TAT 频度表 Υ

图 4.3：TAT 抽取算法

满足对齐一致性，则不能抽取 TAT。对于叶子节点的三元组，TAT 的抽取是非常简单的。如果三元组中的树的高度大于 1，则需先计算基准 TAT，然后再根据孩子节点已经抽取的 TAT，组合构造该节点的 TAT。最后，将 TAT 从栈向量中导出为频度表 Υ。图 4.3 给出了 TAT 抽取算法。

三元组的确定和对齐一致性问题我们已经在第三章介绍过了，这里不再赘述。为了方便对比嵌入句法树的基于短语的模型和基于树到串对齐模板的模型的异同，我们还将沿用第三章所使用的训练样本。为了便于阅读，我们将第三章的训练样本和三元组拷贝到这里，见图 4.4 和表 4.1。

下面，我们将重点讨论两个问题：计算基准 TAT 和构造组合 TAT。

表 4.1：训练样本的所有的三元组

编号	树	串	对齐
1	(NR 布什)	Bush	1:1
2	(NN 总统)	President	1:1
3	(NP (NR 布什) (NN 总统))	President Bush	1:2 2:1
4	(VV 发表)	made	1:1
5	(NN 演讲)	speech	1:1
6	(VP (VV 发表) (NN 演讲))	made a speech	1:1 2:3
7	(IP (NP (NR 布什) (NN 总统)) (VP (VV 发表) (NN 演讲)))	President Bush made a speech	1:2 2:1 3:3 4:5

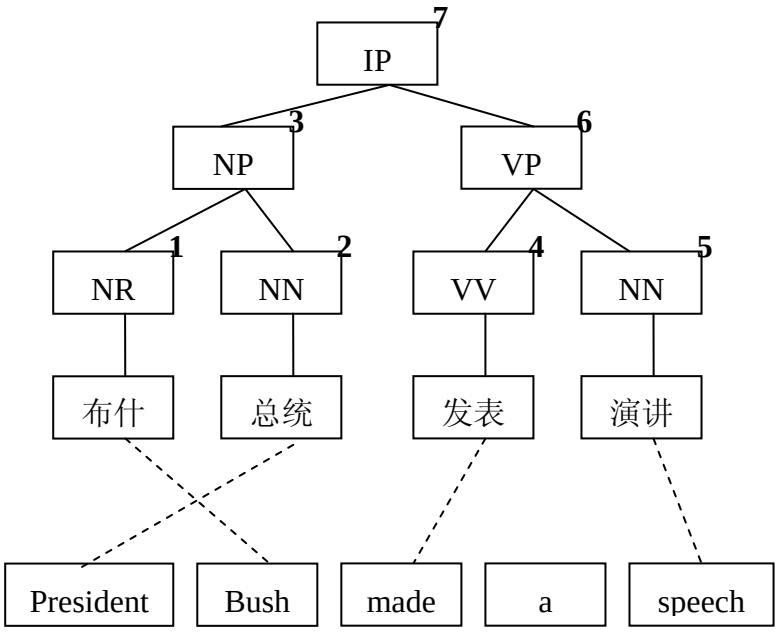

图 4.4：训练样本

4.3.2 计算基准 TAT

我们首先介绍如何在树层高只有 1 的三元组上抽取 TAT。在这种情况下，可以得到两个 TAT。一个是三元组本身，一个是对三元组进行泛化得到的。例如，以表 4.1 中编号为 1 的三元组而言，我们可以得到两个 TAT：

$$\left\langle "(\text{NR 布什})","\text{Bush}",\{(1,1)\}\right\rangle$$

$$\left\langle "(\text{NR})","X",\{(1,1)\}\right\rangle$$

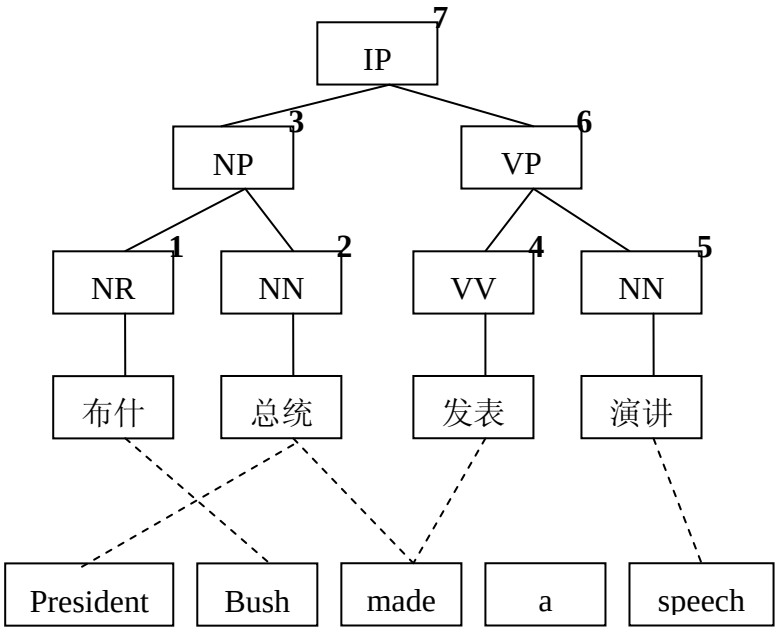

图 4.5：存在对齐不一致情况的训练样本

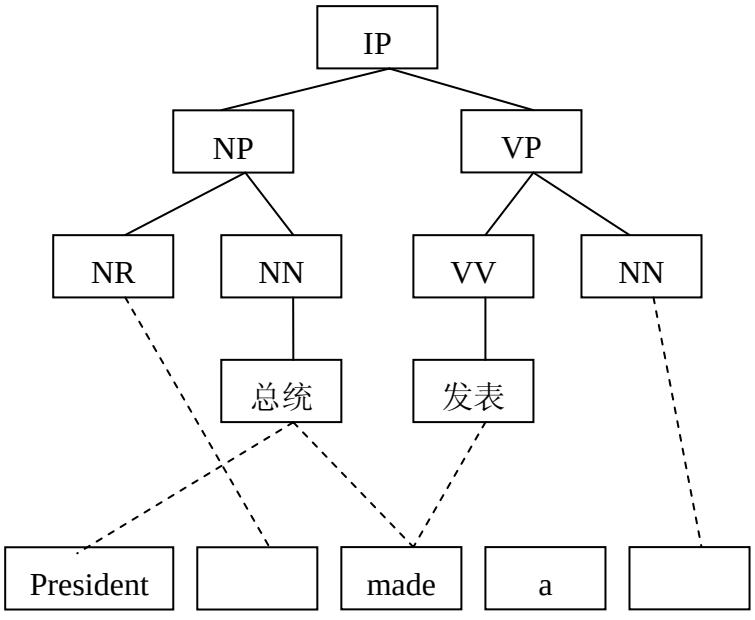

图 4.6：允许内部对齐不一致情况的树模板

输入：三元组$\left(\tilde{T},\tilde{S},\tilde{A}\right)$

[1] 先序遍历$\tilde{T}$，确定每个节点对应的目标语言词的跨度和位置。

[2] 初始化基准源语言树：$\tilde{T}_0 = T$

[3] 初始化基准目标语言串：$\tilde{S}_0 = \tilde{S}$

[4] 先序遍历$\tilde{T}_0$，对于每个节点：

[5] 如果该节点存在对应的目标语言词并且满足对齐一致性

[6] 删除该节点所有的孩子节点，并将该节点的词语设为空串

[7] 将$\tilde{S}_0$中与该节点对应的目标语言词合并成一个非终结符X

[8] 先序遍历$\tilde{T}_0$，对于每个节点：

[9] 如果是叶子节点

[10] 如果该节点的词语为空串

[11] 将对应的目标语言词的跨度存入m_1

[12] 否则

[13] 如果该节点存在对应的目标语言词

[14] 将对应的目标语言词的位置存入m_2

[15] 初始化向量v_1、v_2和v_3。v_1存放源语言叶子节点序号，v_2存放对应目标语言跨度，v_3用来指示v_2存放的目标语言跨度是从m_1或m_2得到的。

[16] 对于m_1中的每个元素，直接存入向量v_1、v_2和v_3。

[17] 对于m_2中的每个元素，对每个目标语言词位置单独构造目标语言跨度，然后存入向量v_1、v_2和v_3。

[18] 对向量v_1、v_2和v_3进行排序，构造基准对齐$\tilde{A}_0$

[19] 构造基准树模板：$z_0 = \left\langle \tilde{T}_0, \tilde{S}_0, \tilde{A}_0 \right\rangle$

输出：基准树模板z_0

图 4.7：计算基准 TAT 算法。

我们将后者称为最小 TAT。与最小 TNR 相似，最小 TAT 虽然没有实际意义，但是在抽取过程中是必需的，在导出时会被摒弃。

为什么计算基准 TAT 和计算基准 TNR 有重大的区别？关键在于处理对齐不一致情况的能力不同。

图 4.5 给出了一个存在对齐不一致情况的训练样本。当在处理到编号 7 的三元组（即整个训练样本）时，无法抽出 TNR，这是因为计算基准 TNR 时不允许孩子的跨度有重叠。换句话说，从三元组泛化得到 TNR 的前提条件是不允许内部出现对齐不一致现象。

对于抽取 TAT 来说则没有这个限制。抽取 TAT 的基本原则是：保留不能泛化的部分，泛化能泛化的部分。图 4.6 给出了一个允许内部对齐不一致的树模板。

图 4.7 给出了计算基准 TAT 算法。为了让读者有更直观的印象，下面将举例子来说明基准 TAT 是如何得到的。

以图 4.5 的训练样本为三元组，算法首先确定每个节点对应的目标语言词的位置和跨度，见表 4.2。

表 4.2：训练样本每个节点对应的目标语言词的位置和跨度。

节点编号	跨度	位置
1	[2,2]	2
2	[1,3]	1,3
3	[1,3]	1,2,3
4	[3,3]	3
5	[5,5]	5
6	[3,5]	3,5
7	[1,5]	1,2,3,5

然后，先序遍历源语言句法树。如果一个节点存在对应的目标语言词并且满足对齐一致性，则删除该节点的孩子节点并将该节点的词语设为空串，然后再将对应的目标语言词合并成一个非终结符。在图 4.5 中，只有节点 1 和 5 满足这个条件。“存在对应的目标语言词并且满足对齐一致性”实际上就是能够进行泛化的条件。这样，就可以得到基准源语言树 $\tilde{T}_0$

$$\left(\mathrm{IP}\left(\mathrm{NP}\left(\mathrm{NR}\right)\left(\mathrm{NN}\ 总统\right)\right)\left(\mathrm{VP}\left(\mathrm{VV}\ 发表\right)\left(\mathrm{NN}\right)\right)\right)$$

表 4.3：叶子节点对应目标语言词的跨度和位置

编号	跨度	位置
1	[2,2]	-
2	-	1,3
4	-	3
5	[5,5]	-

```
int spanSum=0;
for (int i = 0; i < v1.size(); i++)
{
    baseAlignment.push(make_pair(v1[i], v2[i].first - spanSum));

    if (v3[i])
    {
        spanSum += v2[i].second – v2[i].first;
    }
}
```

图 4.8：根据向量v_1、v_2和v_3得到基准对齐的 C++代码。

和基准目标语言串$\tilde{S}_0$

$$\text{President } X_1 \text{ made a } X_2$$

而基准对齐$\tilde{A}_0$的计算就要复杂得多。首先先序遍历基准源语言树$\tilde{T}_0$，对每个叶子节点记录对应目标语言词的跨度和位置，分别存入数据结构m_1和m_2。

我们对能泛化的节点保留跨度，对不能泛化的节点保留位置。之后，将这些信息存入向量v_1、v_2和v_3中，v_1存放源语言叶子节点序号，v_2存放对应目标语言跨度，v_3用来指示v_2存放的目标语言跨度是从m_1或m_2得到的。然后再对向量v_1、v_2和v_3进行排序。

表 4.4：排序后的向量v_1、v_2和v_3

v_1	v_2	v_3
2	[1,1]	0
1	[2,2]	1
2	[3,3]	0
4	[3,3]	0
5	[5,5]	1

表 4.4 给出了排序后的向量v_1、v_2和v_3。在v_3中，1 表示对应v_2中的跨度是由m_1中的跨度直接得到的，0 表示是从m_2中的位置得到的。

已知排序后的向量v_1、v_2和v_3，就可以利用图 4.8 所示的 C++代码生成基准对齐$\tilde{A}_0$：$\{(1,2),(2,1),(2,3),(3,3),(5,5)\}$。最终的基准树模板见图 4.6。

4.3.3 组合构造 TAT

与组合构造 TNR 类似，组合构造 TAT 也需要利用基准 TAT 和孩子节点已经抽出的 TAT。与之不同的是，组合构造时不但需要修改源语言树和对齐信息，还需修改目标语言串。

以图 4.4 中的训练样本为例，假设节点 1 抽出 2 个 TAT：

$$\langle "(\text{NR})","X ",\{(1,1)\}\rangle$$

$$\langle "(\text{NR 布什})","\text{Bush}",\{(1,1)\}\rangle$$

节点 2 抽出 2 个 TAT：

$$\langle "(\text{NN})","X ",\{(1,1)\}\rangle$$

$$\langle "(\text{NN 总统})","\text{President}",\{(1,1)\}\rangle$$

节点 3 的基准 TAT 是：

$$\langle "(\text{NP}(\text{NR})(\text{NN}))","X_1X_2 ",\{(1,2),(2,1)\}\rangle$$

那么可以组合得到节点 3 的 4 个 TAT：

$$\langle "(\text{NP}(\text{NR})(\text{NN}))","X_1X_2 ",\{(1,2),(2,1)\}\rangle$$

$$\langle "(\text{NP}(\text{NR 布什})(\text{NN}))","X \text{ Bush}",\{(1,2),(2,1)\}\rangle$$

$$\langle "(\text{NP}(\text{NR})(\text{NN 总统}))","\text{President } X",\{(1,2),(2,1)\}\rangle$$

$$\langle "(\text{NP}(\text{NR 布什})(\text{NN 总统}))","\text{President Bush}",\{(1,2),(2,1)\}\rangle$$

4.3.4 限制条件

如果限制树模板中的树高度最大为 2，从图 4.4 的训练样本可以抽出以下 13 个TAT[6]：

$$\langle "(\text{NR 布什})","\text{Bush}",\{(1,1)\}\rangle$$

$$\langle "(\text{NN 总统})","\text{President}",\{(1,1)\}\rangle$$

$$\langle "(\text{NP}(\text{NR})(\text{NN}))","X_1X_2",\{(1,2),(2,1)\}\rangle$$

$$\langle "(\text{NP}(\text{NR 布什})(\text{NN}))","X \text{ Bush}",\{(1,2),(2,1)\}\rangle$$

$$\langle "(\text{NP}(\text{NR})(\text{NN 总统}))","\text{President } X",\{(1,2),(2,1)\}\rangle$$

$$\langle "(\text{NP}(\text{NR 布什})(\text{NN 总统}))","\text{President Bush}",\{(1,2),(2,1)\}\rangle$$

$$\langle "(\text{VV 发表})","\text{made}",\{(1,1)\}\rangle$$

$$\langle "(\text{NN 演讲})","\text{speech}",\{(1,1)\}\rangle$$

$$\langle "(\text{VP}(\text{VV})(\text{NN}))","X_1 \text{ a } X_2",\{(1,1),(2,3)\}\rangle$$

$$\langle "(\text{VP}(\text{VV 发表})(\text{NN}))","\text{made a } X",\{(1,1),(2,3)\}\rangle$$

$$\langle "(\text{VP}(\text{VV})(\text{NN 演讲}))","X \text{ a speech}",\{(1,1),(2,3)\}\rangle$$

$$\langle "(\text{VP}(\text{VV 发表})(\text{NN 演讲}))","\text{made a speech}",\{(1,1),(2,3)\}\rangle$$

$$\langle "(\text{IP}(\text{NP})(\text{VP}))","X_1X_2",\{(1,1),(2,2)\}\rangle$$

由此可以看出，树模板的数量通过不断组合而急剧增长。通常，我们能在真实数据上抽出数量庞大的树模板，使训练和解码都非常慢。为了降低抽取的树模板的数量，我们提出两个限制条件：

[6]如果允许树的最大高度为 3，那么仅在根节点就可以抽取 16 个树模板，总共可抽取 28 个树模板。

1. 树模板中树的高度不能超过 h
2. 树模板中树节点的孩子数量不能超过 c

需要指出的是，树模板的数量受词语对齐的影响很大。对齐不一致的情况越多，能抽出的树模板就越少。例如，即使不对树模板中树的高度做限制，从图 4.5 中的训练样本也仅能抽出 6 个 TAT：

$$\left\langle "(\text{NR 布什})","\text{Bush}",\{(1,1)\}\right\rangle$$

$$\left\langle "(\text{NN 演讲})","\text{speech}",\{(1,1)\}\right\rangle$$

$$\left\langle "(\text{IP}(\text{NP}(\text{NR})(\text{NN 总统}))(\text{VP}(\text{VV 发表})(\text{NN})))","\text{President } X_1 \text{ made a } X_2",\{(1,3),(2,1),(2,3),(3,3),(5,5)\}\right\rangle$$

$$\left\langle "(\text{IP}(\text{NP}(\text{NR 布什})(\text{NN 总统}))(\text{VP}(\text{VV 发表})(\text{NN})))","\text{President Bush made a } X",\{(1,3),(2,1),(2,3),(3,3),(5,5)\}\right\rangle$$

$$\left\langle "(\text{IP}(\text{NP}(\text{NR})(\text{NN 总统}))(\text{VP}(\text{VV 发表})(\text{NN 演讲})))","\text{President } X \text{ made a speech}",\{(1,3),(2,1),(2,3),(3,3),(5,5)\}\right\rangle$$

$$\left\langle "(\text{IP}(\text{NP}(\text{NR 布什})(\text{NN 总统}))(\text{VP}(\text{VV 发表})(\text{NN 演讲})))","\text{President Bush made a speech}",\{(1,3),(2,1),(2,3),(3,3),(5,5)\}\right\rangle$$

4.4 搜索

4.4.1 搜索算法

我们采用自底向上的柱搜索（beam search）算法。后序遍历输入的源语言句法树，对每个节点所对应的源语言短语搜索推导。当处理完根节点后，就得到整个句子的翻译。

任何一个翻译都是由推导生成的，而每个推导就是一个树模板序列。推导包含以下信息：

1. 译文
2. 树模板序列
3. 英汉 MLE 概率累积特征值
4. 英汉词汇权重累积特征值
5. 汉英 MLE 概率累积特征值
6. 汉英词汇权重累积特征值
7. 树模板数量累积特征值

8. 语言模型累积特征值
9. 词语数量累积特征值
10. 分值
11. 被重合并项

在嵌入句法树的基于短语的翻译模型中，双语短语是主体，TNR 只起到短语重排序的作用。而在基于树到串对齐模板的翻译模型中，唯一使用的是树模板。树模板既能生成非终结符又能生成终结符，既能执行局部重排序又能执行全局重排序。

在后序遍历源语言句法树时，对每个句法子树计算推导。其方法是首先搜索可用的树模板，如果树模板覆盖整个句法子树，则直接将该树模板作为推导，否则利用该树模板和孩子节点的树模板组合构造推导。

与 TNR 类似，对于一个句法树T，一个 TAT 是可用的当且仅当：

3. $T(z) \subseteq T$，即 TAT 的树是该句法树的子图。
4. $root(T(z)) = root(T)$，即 TAT 的树的根节点和该句法树是相同的。

图 4.9 给出了搜索算法。每个子树对应的推导都被存放在栈中，这些栈被组织成向量的形式，被称为推导栈向量。

在搜索过程中，无法找到可用的树模板是不可避免的。为了保证搜索能够顺利进行，我们提出默认树模板。默认树模板的构造方法是：

1. 如果句法树T只包含一个节点，则默认树模板的树等于T，串是句法树叶子节点的词语，对齐是一对一。
2. 如果句法树包含多个节点，则默认模板的树等于T的根节点及其直接孩子节点，串由非终结符组成，数量等于树的叶子节点数，对齐是顺序对齐。

例如，与句法树(NR 中国)对应的默认模板是：

$$\langle "(\text{NR 中国})","中国",\{(1,1)\}\rangle$$

与句法树$(\text{NP}(\text{NR 中国})(\text{NN 经济}))$对应的默认模板是：

$$\langle "(\text{NP}(\text{NR})(\text{NN}))","X_1X_2",\{(1,1),(2,2)\}\rangle$$

利用树模板组合构造推导的基本原理是在可用树模板的基础上利用未覆盖子树的推导组合构造推导。

输入：源语言句法树 $T\left(f_1^J\right)$

[1] 后序遍历源语言句法树 $T\left(f_1^J\right)$，并对每个节点设定编号。

[2] 初始化推导栈向量 $deriStackVec$

[3] for $id := 1$ to $nodeCount\left(T\left(f_1^J\right)\right)$

[4] 确定与 id 对应的子树 T'

[5] 搜索可用的树模板

[6] 如果没有可用的树模板，构造默认树模板。

[7] 对每个可用的树模板 z

[8] 如果 $T\left(z\right)=T'$

[9] 直接根据树模板 z 构造推导，并压入相应栈中

[10] 如果 $T\left(z\right)\subset T'$

[11] 根据树模板 z 和孩子节点的树模板组合构造推导，并压入相应栈中

[12] end for

[13] 在 $deriStackVec[nodeCount]$ 中找到概率最大的候选翻译，并提取最佳译文 $\hat{e}_1^I$

输出：译文 $\hat{e}_1^I$

图 4.9：搜索算法。

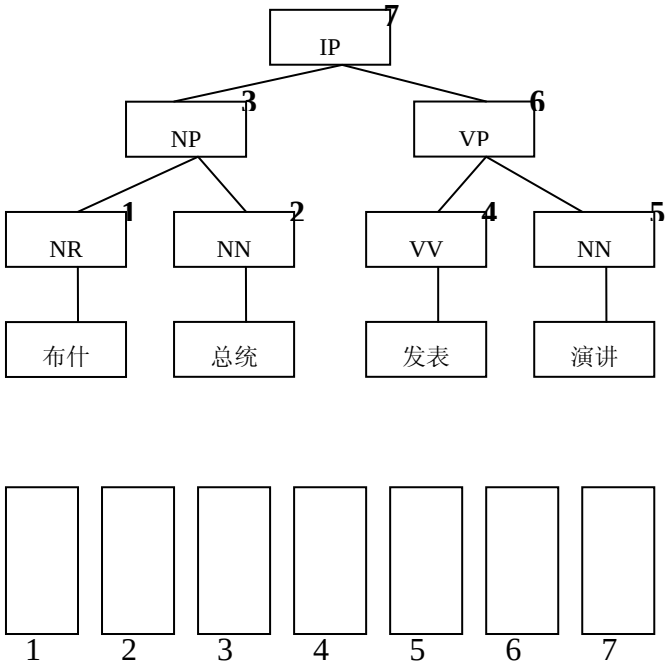

图 4.10：推导栈向量。

以图 4.10 为例，当处理到编号为 7 的根节点时，假设我们找到一个可用的树模板如下：

$$\left\langle "\left(\mathrm{IP}\left(\mathrm{NP}\left(\mathrm{NR}\right)\left(\mathrm{NN}\ 总统\right)\right)\left(\mathrm{VP}\right)\right)","\mathrm{President}\ X_1\ X_2","\left\{(1,2),(2,1),(3,3)\right\}\right\rangle$$

该树模板没有覆盖的子树的根节点分别是 1 和 6。

假设推导栈 1 存储了一个推导：

$$\left\langle "\left(\mathrm{NR}\ 布什\right)","\mathrm{Bush}","\left\{(1,1)\right\}\right\rangle$$

推导栈 6 存储了一个推导：

$$\left\langle "\left(\mathrm{VP}\left(\mathrm{VV}\right)\left(\mathrm{NN}\right)\right)","X_1\ \mathrm{a}\ X_2","\left\{(1,1),(2,3)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{VV}\ 发表\right)","\mathrm{made}","\left\{(1,1)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{NN}\ 演讲\right)","\mathrm{speech}","\left\{(1,1)\right\}\right\rangle$$

那么，我们就可以得到编号为 7 的子树的推导：

$$\left\langle "\left(\mathrm{IP}\left(\mathrm{NP}\left(\mathrm{NR}\right)\left(\mathrm{NN}\ 总统\right)\right)\left(\mathrm{VP}\right)\right)","\mathrm{President}\ X_1\ X_2","\left\{(1,2),(2,1),(3,3)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{NR}\ 布什\right)","\mathrm{Bush}","\left\{(1,1)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{VP}\left(\mathrm{VV}\right)\left(\mathrm{NN}\right)\right)","X_1\ \mathrm{a}\ X_2","\left\{(1,1),(2,3)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{VV}\ 发表\right)","\mathrm{made}","\left\{(1,1)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{NN}\ 演讲\right)","\mathrm{speech}","\left\{(1,1)\right\}\right\rangle$$

如果栈 1 和栈 6 分别有多个推导，那么就需要组合生成节点 7 的推导。

至于剪枝策略，3.4.5 节已经讨论，在此不在赘述。

4.4.2 利用双语短语

本节介绍如何将双语短语引入到基于树到串对齐模板的解码器设计中。我们首先讨论利用双语短语的必要性，然后介绍利用双语短语的两种方式：视作词汇

化树模板和后处理提高流利度。

必要性

我们认为，基于树到串对齐模板的翻译模型面临以下三个难题：

1. 树到串对齐模板只能表示句法短语。
2. 句法分析准确度低。
3. 句法分析速度慢。

词汇化的树到串对齐模板能够表示句法短语，只不过在源语言端存在一棵句法树。但是，由于在抽取树到串对齐模板要求源语言短语之上必须存在子树，非句法双语短语就不可能被树到串对齐模板表示。例如，在图 4.4 的训练样本中，双语短语$\langle$"布什 总统 发表","President Bush made"$\rangle$不可能被树模板表示，因为“布什 总统 发表”之上不存在一棵子树。损失这些非句法短语会降低翻译性能。

通常，句法分析器都是使用宾州树库进行训练。由于宾州树库的规模和领域有限，句法分析器在处理大量的、领域广泛的真实文本时势必会大大降低准确度。

此外，相对于词语对齐而言，句法分析的速度要慢得多，处理上百万句对的双语语料库需要很长时间。因此，树模板的获取代价较高。

另一方面，通常的双语短语既包含句法短语也包含非句法短语，参数训练时只需要词语对齐的双语语料库，更容易获取。

因此，将双语短语加入到基于树到串对齐模板的解码器中有利于提高系统性能。

视作词汇化树模板

第一种利用方法是将双语短语视作特殊的词汇化树模板。给定一个句法子树$T\left(f_{j_1}^{j_2}\right)$并假设$f_{j_1}^{j_2}$只包含终结符，对于一个双语短语$\left\langle \tilde{f},\tilde{e},\tilde{A}\right\rangle$，如果双语短语的源语言短语和句法子树的叶子序列相等，即$\tilde{f}=f_{j_1}^{j_2}$，则可以构造这样一个树模板：

$$z=\left\langle T\left(f_{j_1}^{j_2}\right),\tilde{e},\tilde{A}\right\rangle$$

这样一来，概率计算会出现问题，因为树模板和双语短语的参数训练是相互独立的，它们的概率分布之间没有联系。但我们认为，在对数线性模型框架下，

利用最小错误率训练可以自动调节特征权重，概率计算的问题影响不大。如果根据双语短语构造的树模板和真正抽取出来的树模板相同，只是概率不同，我们优先选择数值高的概率。

例如，对于句法子树$\left(\text{NP}\left(\text{NR 中国}\right)\left(\text{NN 经济}\right)\right)$，我们找到一个双语短语：

中国 经济 ||| China ‘s economic ||| 1:1 2:3 ||| 0.3 0.5 0.2 0.7

那么，可以构造出一个树模板：

(NP (NR 中国) (NN 经济)) ||| China ‘s economic ||| 1:1 2:3 ||| 0.3 0.5 0.2 0.7

我们还在树模板表中找到一个可用的树模板：

(NP (NR 中国) (NN 经济)) ||| China ‘s economic ||| 1:1 2:3 ||| 0.4 0.3 0.6 0.5

经过优先选择数值高的概率，我们得到最终的树模板：

(NP (NR 中国) (NN 经济)) ||| China ‘s economic ||| 1:1 2:3 ||| 0.4 0.5 0.6 0.7

一定存在这样的情况：根据双语短语构造一个树模板，在树模板表中找不到与之相等的树模板。这是因为抽取双语短语只有对齐一致性的限制，而抽取树模板除了要求对齐一致性，还要求源语言短语之上必须存在子树。因此，词汇化树模板的数量要比双语短语的数量少得多。

这种方法的优点在于增大了词汇化树模板的数量，能够提升系统的翻译性能，缺点是只能利用句法双语短语，仍然无法利用非句法双语短语。

后处理提高流利度

如前所述，搜索算法是按照后序遍历的顺序翻译各个子树。在翻译过程中，不会考虑翻译“布什 总统 发表”这样的非句法短语，这样做的好处是减小了搜索空间。但是，即使将双语短语视作词汇化树模板，也无法利用全部的双语短语。比如，我们在双语短语表中能找到下面的双语短语却无法利用。

$\left\langle \text{"布什 总统 发表","President Bush made"} \right\rangle$

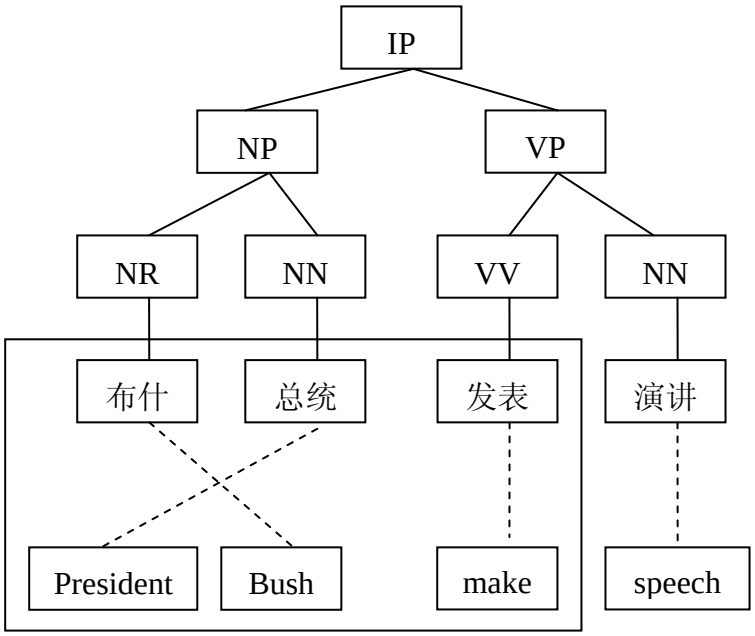

图 4.11：翻译结果

为此，我们提出利用双语短语做后处理提高译文流利度。其基本思想是：当搜索结束后，根据推导生成源语言句法树和译文之间的词语对齐。考察那些在搜索过程中被忽略的非句法短语对[7]（必须保持对齐一致性），在双语短语表中搜索可用的候选翻译。利用语言模型概率来比较候选翻译和原译文，如果候选翻译的语言模型值高于原译文，则用候选翻译替换原译文。如果存在多个优于原译文的候选翻译，则根据短语概率和语言模型概率来选择最佳的替换方案。

例如，图 4.11 给出了一个源语言句法树和译文之间的词语对齐，我们可以找到一个非句法短语对，在图中用方框圈起来。假设我们在双语短语表中找到三个候选翻译：

President Bush made

President Bush made a

President Bush makes

假设这三个候选翻译的语言模型值均高于原译文“President Bush make”，而且短语概率和语言模型概率最高的候选翻译是“President Bush made a”，那么我们就可以将译文修改成“President Bush made a speech”。

[7]之所以只考虑非句法短语对是因为在搜索过程中句法短语已经被考虑了。

这种方法的优点在于所有的双语短语都可以被利用，而且后处理的速度极快，缺点是只依赖于语言模型概率决定是否替换，没有考虑翻译概率。

4.5 讨论

在本章，我们介绍了基于树到串对齐模板的翻译模型。树到串对齐模板描述了源语言句法树和目标语言串之间的对齐关系，既能生成非终结符又能生成终结符，既能执行局部重排序又能执行全局重排序。这个模型是语言学基于句法的（linguistically syntax-based），因为树到串对齐模板是从经过词语对齐和源语言句法分析的平行文本中自动获取的。为了翻译一个源语言句子，我们首先使用句法分析器得到句法树，然后再利用树到串对齐模板将该句法树转换成目标语言串。

基于树到串对齐模板的翻译模型是在嵌入句法树的基于短语的翻译模型的基础上发展而来的。树节点重排序等价于非词汇化树到串对齐模板，即只有调序功能，没有生成终结符的功能。

树到串对齐模板实际上是树到串转换规则，等价于[Galley 2006]使用的 xRS 规则。它与[Chiang 2005]提出的层次化双语短语或同步上下文无关语法也有类似之处，都是允许短语中带变量。不同的是，层次化双语短语只有一种非终结符（占位符），树到串对齐模板只是在目标语言端只有一种非终结符（占位符），源语言端则存在句法树并以词性标记和短语结构类作为非终结符。

基于树到串对齐模板的翻译模型的主要缺点在于不能兼容非句法短语。为此，我们提出将双语短语引入到解码器的设计中。一种方法是根据双语短语构造词汇化树模板，另一种是后处理提高译文流利度。但这两种方法均是辅助手段，并没有从建模上根本解决短语兼容性问题。

第五章 融入森林到串规则的树到串翻译模型

5.1 引言

近两年来，语言学基于句法（linguistically syntax-based）的翻译模型[Quirk 2005; Galley 2006; Marcu 2006; Liu 2006]取得了迅速的发展。这些模型从经过语言学标注的平行文本中学习树到串翻译规则，并在2006年NIST机器翻译评测中取得与最好的基于短语的系统[Och 2004b]十分接近的成绩[8]。其中，[Galley 2006]和[Marcu 2006]将重点放在目标语言分析上，而[Quirk 2005]和[Liu 2006]则证明对源语言的句法建模同样大有裨益。

然而，语言学基于句法的翻译模型的一大缺点是树到串规则无法句法化（syntactify）非句法双语短语，因为树到串规则要求必须存在一棵句法子树覆盖双语短语。在这里，我们将双语短语分为两类：句法双语短语和非句法双语短语。所谓“句法”是指双语短语被某棵句法子树覆盖，否则称为“非句法”。[Marcu 2006]指出，他们在汉英平行语料库上抽取的双语短语中有28%是非句法双语短语。

我们认为让基于句法的模型能够利用所有的双语短语（既包含句法双语短语也包含非句法双语短语）是十分重要的。一方面，双语短语被证明是一种十分简单有效的机器翻译机制，擅长翻译在训练语料库中出现频度高的词串和捕获局部重排序，能够直接包含上下文信息。[Chiang 2005]在将句法引入翻译的同时保留了双语短语的优点，明显超过了传统的基于短语的模型。另一方面，语言学基于句法的翻译模型只利用句法短语会使翻译性能受到限制。[Quirk 2006]指出语言学基于句法的翻译模型的性能直接受到句法分析质量的影响。由于训练语料库的规模和领域有限，句法分析器在处理领域广泛的大量真实文本时准确度势必会大大降低，从而影响语言学基于句法的模型的性能。

一些研究人员针对这个问题提出了解决方案。[Marcu 2006]对每一个非句法双语短语构建一个特殊的 xRS 规则。该规则以一个虚节点作为根节点，将多棵树归于一棵树，从而能够覆盖非句法短语。为了解释这个特殊的 xRS 规则如何和其他 xRS 规则一起构成推导，[Marcu 2006]还构建一个兄弟（sibling）规则。虚节点的命名方式反映了使用特殊 xRS 规则及其兄弟规则的方式。然而，[Marcu 2006]在创建兄弟规则时破坏了对齐一致性。此外，虚节点的命名方式过于简单，难于处理更复杂的情况。

[Liu 2006]将双语短语视作特殊的词汇化树到串对齐模板。如果一个双语短语的源语言部分能够被输入句法子树覆盖，那么该双语短语就能够在解码中使

[8] 参见 http://www.nist.gov/speech/tests/mt/mt06eval_official_results.html。

用。虽然这个方法增大了词汇化树到串对齐模板的数量，只有句法双语短语能够被基于树到串对齐模板的模型使用。此外，树模板和双语短语的概率分布是独立估计的，直接合并可能会存在问题。

在本章，我们提出森林到串翻译规则，它描述了多棵树和一个串之间的对应关系。森林到串翻译规则不仅能够捕获非句法双语短语，而且还具备泛化能力。为了将森林到串翻译规则融入到树到串翻译模型中，我们提出辅助规则来提供泛化层。由于没有虚节点和命名机制，森林到串翻译规则的融入十分灵活，只依赖于森林的根节点序列。森林到串翻译规则和辅助规则为树到串翻译模型提供了更普遍的推导方式，同时保留了树到串翻译规则的优点。

5.2 模型

我们将树到串翻译规则定义为一个三元组 $r=\langle \tilde{T},\tilde{S},\tilde{A}\rangle$，描述了源语言句法树 $\tilde{T}=T\left(f_1^{J'}\right)$ 和目标语言串 $\tilde{S}=e_1^{I'}$ 之间的对齐关系 $\tilde{A}$。源语言串 $f_1^{J'}$ 是源语言句法树 $T\left(f_1^{J'}\right)$ 的叶子节点序列，既可能包含终结符（词）也可能包含非终结符（词性标记或短语结构类）。目标语言串 $e_1^{I'}$ 也同样既可能包含终结符（词）也可能包含非终结符（占位符）。对齐 $\tilde{A}$ 被定义为源语言和目标语言符号位置的笛卡尔集的子集：

$$\tilde{A}\subseteq\left\{(j,i):j=1,...,J';i=1,...,I'\right\}$$

实际上，第四章介绍的树到串对齐模板就是树到串翻译规则。

推导 $\theta=r_1\circ r_2\circ...\circ r_n$ 是规则的序列，可以解释一个源语言句法树 $T=T\left(f_1^J\right)$、一个目标语言串 $S=e_1^I$ 和两者之间的词语对齐 A 是如何同步生成的。

例如，表 5.1 给出了一个只包含树到串规则的推导，解释了图 5.1 中的源语言句法树、目标语言串及词语对齐是如何同步生成的。

如前所述，树到串规则无法句法化非句法双语短语，因为树到串规则要求必须存在一棵句法子树覆盖双语短语。例如，对于图 5.1 中的双语短语

$$\langle\text{"枪手 被","The gunman was"}\rangle$$

我们无法抽取一个等价的树到串翻译规则，因为不存在句法子树能够覆盖源语言短语“枪手 被”。

表 5.1：只包含树到串规则的推导

编号	树	串	对齐
(1)	(IP (NP) (VP) (PU))	$X_1\ X_2\ X_3$	1:1 2:2 3:3
(2)	(NP (NN 枪手))	The gunman	1:1 2:2
(3)	(VP (SB 被) (VP (NP (NN)) (VV 击毙)))	was killed by X	1:1 2:4 3:2
(4)	(NN 警方)	police	1:1
(5)	(PU 。)	.	1:1

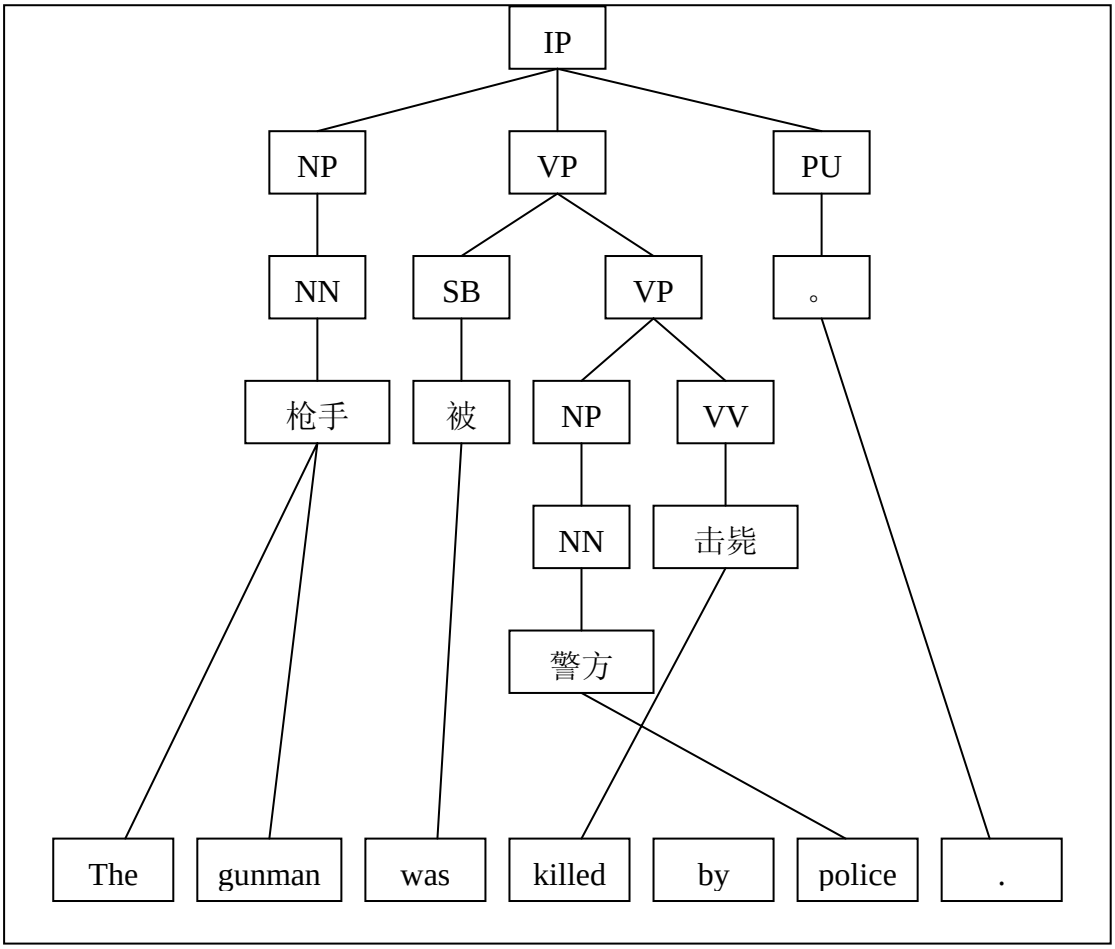

图 5.1：源语言句法树、目标语言串及词语对齐。

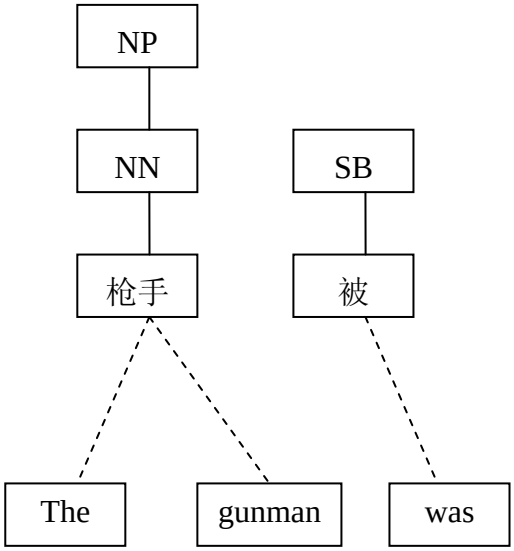

图 5.2：森林到串翻译规则。

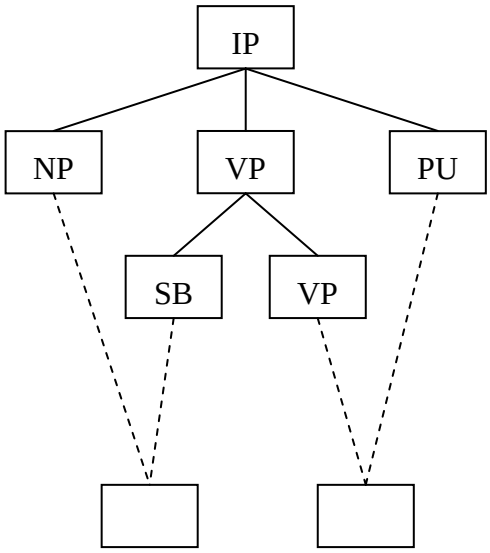

图 5.3：辅助规则。

为了解决这个问题，我们提出森林到串翻译规则。一个森林到串翻译规则是一个三元组 $r=\langle \tilde{F},\tilde{S},\tilde{A}\rangle$，描述了包含 K 棵源语言句法树的森林 $\tilde{F}=\tilde{T}_1^K$ 和目标语言串 $\tilde{S}=e_1^{I'}$ 之间的对齐关系 $\tilde{A}$。源语言串 $f_1^{J'}$ 是森林 $\tilde{F}$ 的叶子节点序列。森林到串翻译规则与树到串规则一样具备泛化能力，即源语言串 $f_1^{J'}$ 既可能包含终结符（词）也可能包含非终结符（词性标记或短语结构类）。目标语言串 $e_1^{I'}$ 也同样既可能包含终结符（词）也可能包含非终结符（占位符）。

因此，我们可以为上面的双语短语设计一个等价的森林到串翻译规则，如图 5.2 所示。

我们将森林到串对齐模板按照词汇化程度分为三类：

1. 词汇化：所有的源语言和目标语言符号均是终结符
2. 部分词汇化：源语言和目标语言符号既包含非终结符也包含终结符
3. 非词汇化：所有的源语言和目标语言符号均是非终结符

表 5.2：包含树到串规则、森林到串规则和辅助规则的推导。

编号	树/森林	串	对齐
(1)	(IP (NP) (VP (SB) (VP)) (PU))	$X_1\ X_2$	1:1 2:1 3:2 4:2
(2)	(NP (NN 枪手)) (SB 被)	The gunman was	1:1 1:2 2:3
(3)	(VP (NP) (VV 击毙)) (PU 。)	killed by X .	1:3 2:1 3:4
(4)	(NP (NN 警方))	police	1:1

为了将森林到串翻译规则融入到树到串翻译模型之中，我们引入辅助规则。一个辅助规则是一个特殊的非词汇化的树到串规则，允许多个源语言非终结符连向一个目标语言非终结符。这是特殊的表示方式，用来表明森林到串翻译规则的融入条件。图 5.3 给出了一个辅助规则，它允许融入两个森林到串规则：一个以“NP”和“SB”为根节点序列，一个以“VP”和“PU”为根节点序列。因此，森林到串翻译规则的融入十分灵活，只依赖于森林的根节点序列。

表 5.2 给出了一个包含树到串规则、森林到串规则和辅助规则的推导，同样也解释了图 5.1 中的源语言句法树、目标语言串及词语对齐是如何同步生成的。其中，编号为 1 的是辅助规则，编号为 2 和 3 的是森林到串规则，编号为 4 的是树到串规则。

类似于[Marcu 2006]，我们将三元组$\langle T,S,A\rangle$的概率定义为所有与三元组一致的推导的概率之和，而单个推导的概率等于推导中每个规则的概率乘积：

$$\Pr(T,S,A)=\sum_{\theta_i\in\Theta,c(\Theta)=\langle T,S,A\rangle}\prod_{r_j\in\theta_i}p(r_j) \qquad \text{公式 5.1}$$

5.3 训练

我们分别在第三章介绍了树节点排序和树到串对齐模板的抽取算法，森林到串规则的抽取算法在基本思想上与前两种是一致的，都是自底向上的策略和组合构造规则。因此，我们省略前面已经提及的概念和算法，将重点放在森林到串规则抽取算法的新特性上。

5.3.1 抽取算法

森林到串规则抽取算法的输入是一个经过词语对齐和源语言句法分析的双语句对$\left(T\left(f_1^J\right),e_1^I,A\right)$，输出是规则频度表$R$。该规则频度表既包含树到串规则，也包含森林到串规则。

抽取算法依然采用自底向上的策略，不过不再采用后续遍历的方式，而是采用类似于 CKY 算法的形式考察每个源语言跨度。对于每个源语言跨度，我们首先确定（树，串，对齐）或（森林，串，对齐）三元组，同时检查对齐一致性。给定（树，串，对齐）三元组，抽取树到串规则的方法与第四章相同。给定（森林，串，对齐）三元组，我们同样计算一个基准森林到串规则，然后再根据已经抽取的规则组合构造森林到串规则。

在确定三元组上，新的规则抽取算法和第四章的算法有所不同。

在图 4.3 描述的 TAT 抽取算法中，我们是后序遍历句法树，按照节点逐个进行处理。已知节点，其对应的子树是固定的，从而对应的三元组也是唯一的。

在图 5.4 描述的规则抽取算法中，我们采用类似 CKY 算法的形式，按照源语言跨度逐个进行处理。每个源语言跨度可能对应着多个的句法树或森林，因此对应的三元组不是唯一的。

输入：源语言句法树 $T\left(f_1^J\right)$，目标语言句子 e_1^I，词语对齐 A

[1] $R := \phi$

[2] for $u := 0$ to J -1 do

[3] for $v := 1$ to $J - u$ do

[4] 确定与跨度 $(v, v+u)$ 对应的三元组集合 Γ

[5] 对于每个三元组 $t = \langle T', S', A' \rangle \in \Gamma$

[6] 如果三元组不满足对齐一致性，则继续

[7] 如果 $u = 0 \wedge node(T') = 1$

[8] 将 t 加入到 R 中

[9] 将 $\langle root(T'), "X", \{(1,1)\} \rangle$ 加入到 R 中

[10] 否则

[11] 计算三元组的基准规则

[12] 根据基准规则和孩子的规则组合构造新规则，并加到 R 中

$R := R \cup build(s, R)$

[13] end for

[14] end for

输出：规则频度表 R

图 5.4：规则抽取算法

以图 5.1 为例，源语言跨度（1,1）可能对应着两棵句法树：

(NN 枪手)

(NP (NN 枪手))

源语言跨度(1,2)可能对应着两个句法森林：

(NN 枪手) (SB 被)

(NP (NN 枪手)) (SB 被)

为了减少森林的数量，我们限定森林中每棵树的根节点在完整树（即 $T\left(f_1^J\right)$）中必须有兄弟。因此，源语言跨度(1,2)只能对应一个森林：

(NP (NN 枪手)) (SB 被)

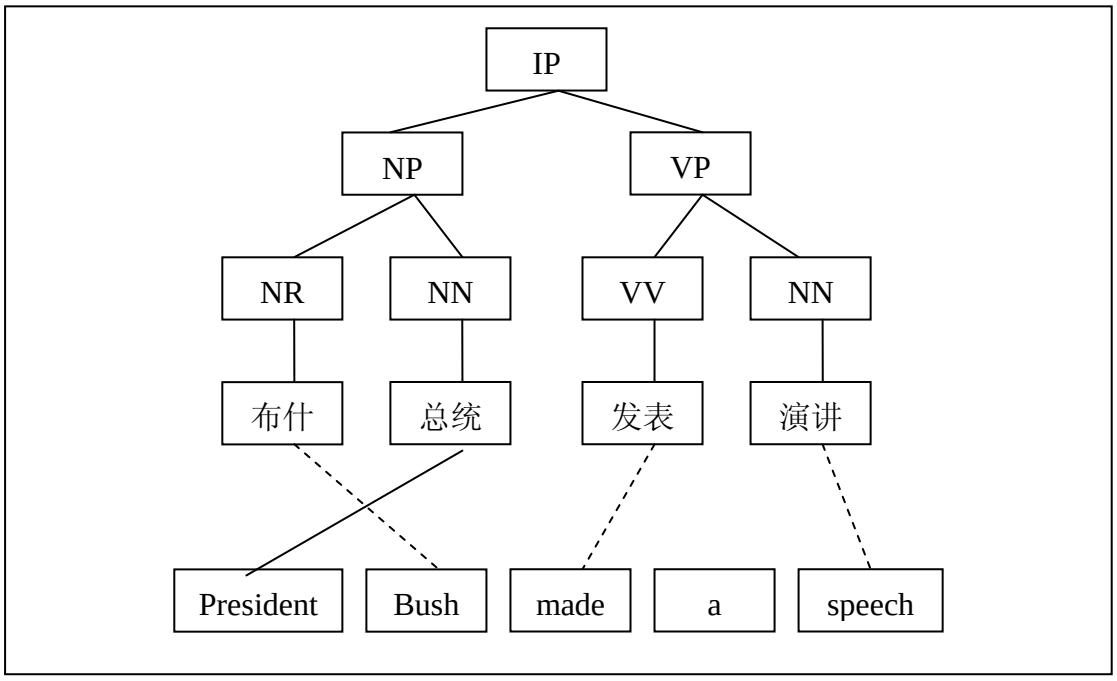

图 5.5：训练样本

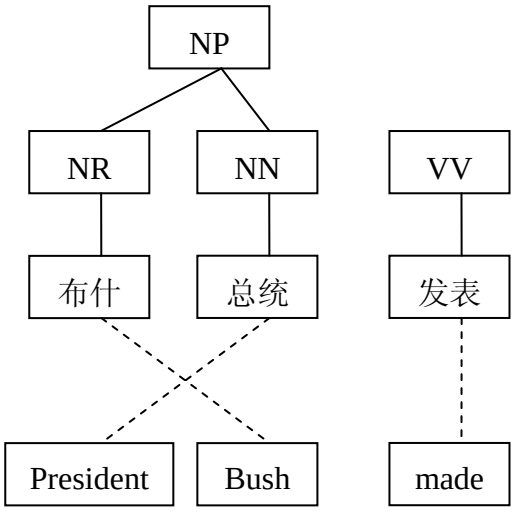

图 5.6：（森林，串，对齐）三元组

已知一个三元组t，基准规则s是满足以下条件的规则：

1. $s \in R(t)$，即基准规则是从三元组t上抽取的规则
2. $node(T(s)) \geq 2$，即基准规则至少包含两个节点
3. $\forall r \in R(t) \wedge node(T(r)) \geq 2, T(s) \subseteq T(r)$，即对所有从三元组$t$上抽取的、至少包含两个节点的规则，基准规则的树或者森林都是它们的树或者森林的子图。

计算基准规则的算法和图 4.7 描述的算法十分接近，我们不予详细讨论。

以图 5.5 中的训练样本为例，对于源语言跨度(1,3)，我们可以确定如图 5.6 所示的（森林，串，对齐）三元组。这个三元组的基准规则是：

$$("(\text{NP})(\text{VV})","X_1\ X_2",\{(1,1),(2,2)\})$$

由于算法是自底向上运行的，我们可以假设跨度(1,2)上已经抽取了 5 个规则：

$$("(\text{NP})","X",\{(1,1)\})$$

$$("(\text{NP}(\text{NR})(\text{NN}))","X_1\ X_2",\{(1,2),(2,1)\})$$

$$("(\text{NP}(\text{NR 布什})(\text{NN}))","X\ \text{Bush}",\{(1,2),(2,1)\})$$

$$("(\text{NP}(\text{NR})(\text{NN 总统}))","\text{President}\ X",\{(1,2),(2,1)\})$$

$$("(\text{NP}(\text{NR 布什})(\text{NN 总统}))","\text{President Bush}",\{(1,2),(2,1)\})$$

假设在跨度(3,3)上已经抽取了 2 个规则：

$$("(\text{VV})","X",\{(1,1)\})$$

$$("(\text{VV 发表})","\text{made}",\{(1,1)\})$$

因此，我们可以组合得到该三元组的 10 个规则：

$$("(\text{NP})(\text{VV})","X_1\ X_2",\{(1,1),(2,2)\})$$

$$("(\text{NP}(\text{NR})(\text{NN}))(\text{VV})","X_1\ X_2\ X_3",\{(1,2),(2,1),(3,3)\})$$

$$("(\text{NP}(\text{NR 布什})(\text{NN}))(\text{VV})","X_1\ \text{Bush}\ X_2",\{(1,2),(2,1),(3,3)\})$$

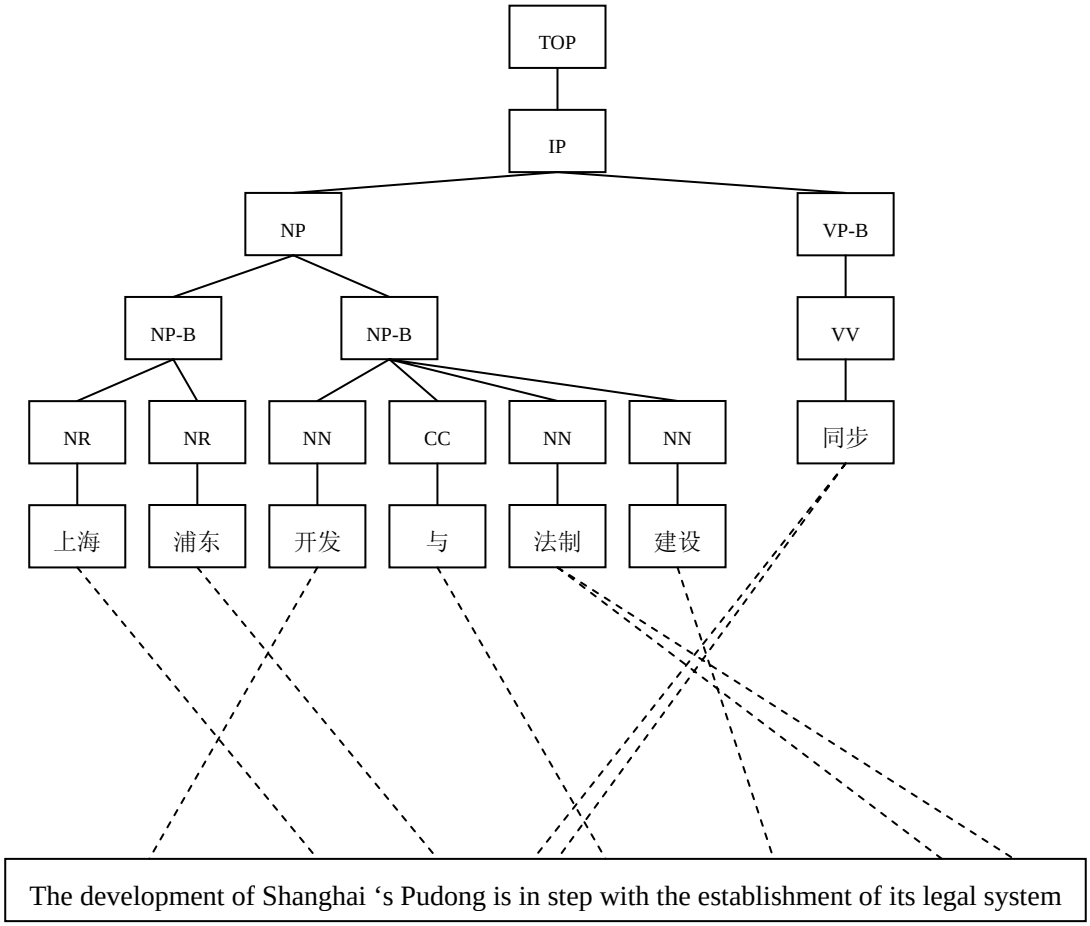

图 5.7：真实训练样本。

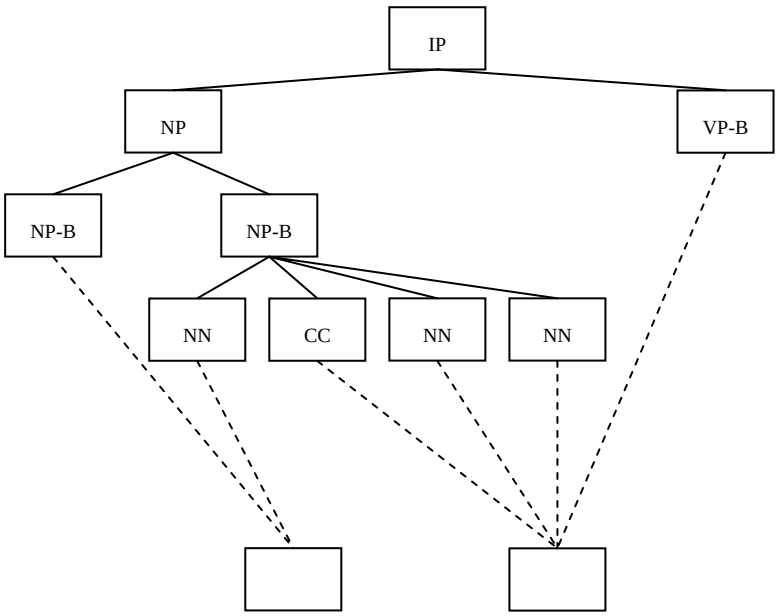

图 5.8：从真实训练样本抽出的辅助规则。

$$\left("\left(\text{NP}\left(\text{NR}\right)\left(\text{NN 总统}\right)\right)\left(\text{VV}\right)","\text{President } X_1\ X_2","\left\{\left(1,2\right),\left(2,1\right),\left(3,3\right)\right\}\right)$$

$$\left("\left(\text{NP}\left(\text{NR 布什}\right)\left(\text{NN 总统}\right)\right)\left(\text{VV}\right)","\text{President Bush } X","\left\{\left(1,2\right),\left(2,1\right),\left(3,3\right)\right\}\right)$$

$$\left("\left(\text{NP}\right)\left(\text{VV 发表}\right)","X\ \text{made}","\left\{\left(1,1\right),\left(2,2\right)\right\}\right)$$

$$\left("\left(\text{NP}\left(\text{NR}\right)\left(\text{NN}\right)\right)\left(\text{VV 发表}\right)","X_1\ X_2\ \text{made}","\left\{\left(1,2\right),\left(2,1\right),\left(3,3\right)\right\}\right)$$

$$\left("\left(\text{NP}\left(\text{NR 布什}\right)\left(\text{NN}\right)\right)\left(\text{VV 发表}\right)","X\ \text{Bush made}","\left\{\left(1,2\right),\left(2,1\right),\left(3,3\right)\right\}\right)$$

$$\left("\left(\text{NP}\left(\text{NR}\right)\left(\text{NN 总统}\right)\right)\left(\text{VV 发表}\right)","\text{President } X\ \text{made}","\left\{\left(1,2\right),\left(2,1\right),\left(3,3\right)\right\}\right)$$

$$\left("\left(\text{NP}\left(\text{NR 布什}\right)\left(\text{NN 总统}\right)\right)\left(\text{VV 发表}\right)","\text{President Bush made}","\left\{\left(1,2\right),\left(2,1\right),\left(3,3\right)\right\}\right)$$

5.3.2 辅助规则获取

虽然有可能从训练语料库中直接学习辅助规则，但实际上是不现实的。为了抽取辅助规则，必须向根节点回溯查找能覆盖整个森林的节点，同时还需保证对齐一致性。这样在处理真实训练语料库时往往会生成极为复杂的辅助模板，导致严重的数据稀疏问题。

图 5.7 给出了一个真实训练样本。我们可以从中对非句法双语短语（“上海 浦东 开发”，“development of Shanghai ‘s Pudong”）构建森林到串翻译规则。为了抽出相应的辅助模板，我们向根节点回溯，找到第一个能够覆盖这个森林的节点“NP”，即“IP”的左孩子。但是，与“NP”对应的三元组不满足对齐一致性，因为“同步”连向“in step”。为了保持对齐一致性，我们继续回溯到“IP”，从而构造出图 5.8 所示的辅助规则。这样一个复杂的辅助规则，不但在训练语料库中出现的频度低，在解码时也很难使用到。

自动抽取的辅助规则之所以复杂，是因为必须保证对齐一致性。[Marcu 2006] 为了降低他们提出的兄弟规则的复杂度，不得不破坏对齐一致性。我们认为这种做法不合适。

因此，我们不在训练语料库中自动学习辅助模板，而是在解码时动态构造。

输入：源语言句法树 $T\left(f_1^J\right)$

[1] for $u := 0$ to J -1
[2] for $v := 1$ to J - u
[3] 对 $[v, v+u]$ 对应的每个树或森林 T'
[4] 如果 T' 是树
[5] 对每个可用的树到串规则 r
[6] 对每个根据 r 和 $matrix$ 中的推导构造出的推导 θ
[7] 将推导 θ 添加到 $matrix\left[v, v+u, root\left(T'\right)\right]$ 中
[8] 搜索子跨度分割 $D\left[v, v+u\right]$
[9] 对于 $D\left[v, v+u\right]$ 中的每个子跨度分割 d
[10] 如果子跨度分割 d 包含至少一个森林跨度
[11] 构造辅助规则 r_a
[12] 对每个根据 r 和 $matrix$ 中的推导构造出的推导 θ
[13] 将推导 θ 添加到 $matrix\left[v, v+u, root\left(T'\right)\right]$ 中
[14] 否则
[15] 对每个可用的森林到串规则 r
[16] 对每个根据 r 和 $matrix$ 中的推导构造出的推导 θ
[17] 将推导 θ 添加到 $matrix\left[v, v+u, root\left(T'\right)\right]$ 中
[18] 搜索子跨度分割 $D\left[v, v+u\right]$
[19] end for
[20] end for
[21] 在 $matrix\left[1, J, root\left(T\right)\right]$ 中找到概率最大的推导 $\hat{\theta}$，并提取最佳译文 $\hat{S} = e\left(\hat{\theta}\right)$

输出：译文 $\hat{S}$

图 5.9：搜索算法

5.4 搜索

给定一个源语言句子$T\left(f_1^J\right)$，我们的解码器搜索能生成$T\left(f_1^J\right)$的概率最大的推导，将该推导的目标语言串作为最佳译文：

$$\begin{aligned}\hat{S} &= \arg\max_{S,A} \Pr(T,S,A) \\ &= \arg\max_{S,A} \sum_{\theta_i \in \Theta, c(\Theta)=\langle T,S,A\rangle} \prod_{r_j \in \theta_i} p(r_j) \\ &\approx \arg\max_{S,A,\theta} \prod_{r_j \in \theta, c(\theta)=\langle T,S,A\rangle} p(r_j)\end{aligned} \qquad \text{公式 5.2}$$

我们采用自底向上的柱搜索（beam search）算法。采用类似于 CKY 算法的形式考察每个源语言跨度。对于每个源语言跨度，对相应的树或者森林搜索推导。当处理完整个句子的跨度后，就得到整个句子的翻译。

任何一个翻译都是由推导生成的，而每个推导就是一个规则序列，可能包含树到串规则、森林到串规则和辅助规则。推导包含以下信息：

1. 译文
2. 规则序列
3. 英汉 MLE 概率累积特征值
4. 英汉词汇权重累积特征值
5. 汉英 MLE 概率累积特征值
6. 汉英词汇权重累积特征值
7. 规则数量累积特征值
8. 语言模型累积特征值
9. 词语数量累积特征值
10. 分值
11. 被重合并项

图 5.9 给出了搜索算法。我们利用一个数组 *matrix* 来组织推导，数组中的元素 $matrix[j_1, j_2, X]$ 存放的是推导集合，$[j_1, j_2, X]$ 表示源语言跨度为 (j_1, j_2) 根节点为 X 的树或者森林。我们用空串“”来表示森林的虚节点。

对于一个输入树或者森林T'，一个翻译规则r是可用的当且仅当：

1. $T(r)\subseteq T'$，即翻译规则r的树或者森林是T'的子图。
2. $root(T(r))=root(T')$，即翻译规则r的树或者森林的根节点序列与T'的根节点序列相同。

例如，以下三个规则对于输入树“(NP (NR 中国) (NN 经济))”是可用的：

$$\langle "(\mathrm{NP}(\mathrm{NR})(\mathrm{NN}))","X_1\ X_2",\{(1,2),(2,1)\}\rangle$$

$$\langle "(\mathrm{NP}(\mathrm{NR}\ 中国)(\mathrm{NN}))","\mathrm{China}\ X",\{(1,1),(2,2)\}\rangle$$

$$\langle "(\mathrm{NP}(\mathrm{NR})(\mathrm{NN}\ 经济))","X\ \mathrm{economy}",\{(1,1),(2,2)\}\rangle$$

类似的，森林到串规则

$$\langle "(\mathrm{NP}(\mathrm{NR})(\mathrm{NN}))(\mathrm{VP})","X_1\ X_2\ X_3",\{(1,2),(2,1),(3,3)\}\rangle$$

对于输入森林“$(\mathrm{NP}(\mathrm{NR}\ 布什)(\mathrm{NN}\ 总统))(\mathrm{VP}(\mathrm{VV}\ 发表)(\mathrm{NN}\ 演讲))$”是可用的。

接下来，我们来介绍如何利用可用规则为一棵输入树或森林搜索推导。

如果规则的树或者森林与输入完全相同，即$T(r)=T'$，该规则r构成一个推导。这种情况多发生在处理叶子节点时。

如果规则的树或者森林是输入的子图，即$T(r)\subset T'$，则需要利用子跨度已经搜索的推导组合构造新的推导。

假设我们现在要翻译图 5.1 中的源语言句法树，并且为$[1,5,"\mathrm{IP}"]$对应的句法树找到一个可用的规则：

$$\langle "(\mathrm{IP}(\mathrm{NP})(\mathrm{VP})(\mathrm{PU}))","X_1\ X_2\ X_3",\{(1,1),(2,2),(3,3)\}\rangle$$

由于搜索算法是自底向上运行的，该规则没有覆盖的部分已经被翻译了。

对$[1,1,"\mathrm{NP}"]$对应的句法树，假设我们可以在数组*matrix*中找到一个推导：

$$\langle "(NP(NN枪手))","\mathrm{The\ gunman}",\{(1,1),(2,2),(3,3)\}\rangle$$

对$[2,4,"VP"]$对应的句法树，我们在数组*matrix*中找到一个推导：

$$\langle "(VP(SB\ 被)(VP(NP(NN))(VV\ 击毙)))","was\ killed\ by\ X",\{(1,1),(2,4),(3,2)\}\rangle$$

$$\langle "(NN\ 警察)","police",\{1,1\}\rangle$$

对$[5,5,"PU"]$对应的句法树，我们在数组*matrix*中找到一个推导：

$$\langle "(PU\ 。)",".",\{1,1\}\rangle$$

因此，我们就能为$[1,5,"IP"]$对应的句法树构建一个推导，如表 5.1 所示。

在前面我们已经提到，辅助规则是特殊的非词汇化树到串规则，我们将在搜索时动态构造，而不是从真实训练数据中学习。为了给一个跨度构造辅助规则，我们需要首先确定它的子跨度分割。

表 5.3：子跨度分割及对应的辅助规则。

子跨度分割	辅助规则		
[1,1] [2,2] [3,5]	(IP(NP)(VP(SB)(VP))(PU))	$X_1\ X_2\ X_3$	1:1 2:2 3:3 4:3
[1,2] [3,4] [5,5]	(IP(NP)(VP(SB)(VP))(PU))	$X_1\ X_2\ X_3$	1:1 2:1 3:2 4:3
[1,3] [4,5]	(IP(NP)(VP(SB)(VP(NP)(VV)))(PU))	$X_1\ X_2$	1:1 2:1 3:1 4:2 5:2
[1,1] [2,5]	(IP(NP)(VP)(PU))	$X_1\ X_2$	1:1 2:2 3:2

一个跨度序列$c_1,c_2,...,c_n$是跨度c的子跨度分割，当且仅当：

1. $c_1.begin = c.begin$
2. $c_n.end = c.end$
3. $c_j.end + 1 = c_{j+1}.begin, 1 \le j < n$

给定一个子跨度划分，很容易为一个跨度对应的句法树构建辅助规则。对于每一个子跨度，我们需要向上搜索第一个能够覆盖它的节点。该节点的所有后代都被删除，从而得到辅助规则的树。辅助规则的串只包含非终结符，非终结符的数量等于子跨度的数量。我们假设辅助规则中树和串之间的对齐是顺序的。

表 5.3 给出了图 5.1 中句法树的一些子跨度分割及对应的辅助规则。为了简

输入：跨度$[j_1, j_2]$，推导数组 *matrix*，子跨度分割数组 D

[1] 如果 $j_1 = j_2$

[2] 初始化 $\hat{p} := 0$

[3] 对 $matrix[j_1, j_2, \cdot]$ 中的每个推导 θ

[4] $\hat{p} := \max\left(p(\theta), \hat{p}\right)$

[5] 将 $\{[j_1, j_2]\} : \hat{p}$ 添加到 $D[j_1, j_2]$ 中

[6] 否则

[7] 如果 $[j_1, j_2]$ 是一个森林跨度

[8] 初始化 $\hat{p} := 0$

[9] 对 $matrix[j_1, j_2, \cdot]$ 中的每个推导 θ

[10] $\hat{p} := \max\left(p(\theta), \hat{p}\right)$

[11] 将 $\{[j_1, j_2]\} : \hat{p}$ 添加到 $D[j_1, j_2]$ 中

[12] for $j := j_1$ to j_2-1

[13] 对于 $D[j_1, j]$ 中的每个子跨度分割 d_1

[14] 对于 $D[j+1, j_2]$ 中的每个子跨度分割 d_2

[15] 构建一个新子跨度分割：$d := d_1 \oplus d_2$

输出：子跨度分割 $D[j_1, j_2]$

图 5.10：子跨度分割搜索算法

化表示，我们省略了节点标记。

对于一个长度为 n 的跨度，总共有 2^{n-1} 个子跨度分割。我们只考虑那些至少包含一个森林子跨度的分割，因为树到串规则能够处理那些只包含树跨度的分割。

图 5.10 给出了我们所采用的子跨度分割算法。我们用 $matrix[j_1, j_2, \cdot]$ 来表示跨度为 (j_1, j_2) 的所有对应的树或森林。子跨度分割及相关的概率被存储在数组 D 中。我们对于两个子跨度分割定义了合并操作符 $\oplus$：两个子跨度分割的子跨度被连接起来，它们的概率被累加。

我们依然采用对数线性模型框架，除了第四章描述的特征函数，我们还单独设计了一个特征累加辅助规则中树的节点数。我们认为，在搜索过程中，平衡传统的树到串规则和新引入的森林到串规则和辅助规则的使用是非常重要的，因此用这个新加的特征来控制规则选择的偏向性。

5.5 讨论

在本章，我们介绍了融入森林到串规则的树到串翻译模型。第四章提出的树到串对齐模板实际上是树到串翻译规则，它要求源语言端必须是一棵句法树。正是因为这个限制，树到串规则只能表达句法双语短语并进行泛化，无法表达和泛化非句法短语。研究表明，同时使用句法双语短语和非句法双语短语对于基于句法的模型极为重要。为此，我们提出森林到串翻译规则来描述多棵句法树和一个串之间的对应关系，从而能够表达和泛化全部的双语短语。为了将森林到串规则融入到树到串模型，我们提出辅助规则来提供泛化层。辅助规则并不是从训练语料库中自动学习的，而是在搜索时动态构造的。模型 3 在保留模型 2 的全部优点的同时增加了表达能力，使翻译性能得到进一步提升，但是训练和搜索的复杂度都要比基于树到串对齐模板的模型大得多。

从理论上说，使用树到串规则和森林规则就可以完全不再使用双语短语。但是，为了控制抽取的规则的数量，我们加了许多限制，词汇化规则的实际数量还是会大大少于双语短语的数量。

在构造辅助规则时，我们默认采用顺序对齐。一个更合理的方案是认为所有可能的对齐的概率是均匀分布的，但代价是扩大了搜索空间。

如果将森林到串规则改成在目标语言端是多棵句法树，在源语言端是串，理

论上应该也可以用在串到树翻译模型[Galley 2006; Marcu 2006]中。抽取算法不需要改动，但是搜索算法会有较大的区别，特别是辅助规则也许无法动态构造。如何将森林到串规则应用到串到树模型将是很有意义的研究题目。

第六章 对比实验

6.1 实验设置

6.1.1 基准系统

在本章，我们将对以下三个翻译模型的翻译性能进行评估：

1. 模型 1，即嵌入句法树的基于短语的翻译模型；
2. 模型 2，即基于树到串对齐模板的翻译模型；
3. 模型 3，即融入森林到串规则的树到串翻译模型。

为了和前人的工作有可比性，我们选择目前学术界广泛使用的、可免费获取的基于短语的统计机器翻译系统Pharaoh[Koehn, 2004][9]作为基准系统。Pharaoh采用了对数线性模型框架：

$$p(e|f) = p_{\phi}(f|e)^{\lambda_{\phi}} \times p_{LM}(e)^{\lambda_{LM}} \times p_{D}(e,f)^{\lambda_{D}} \times \omega^{length(e)\lambda_{w(e)}} \quad \text{公式 6.1}$$

Pharaoh 对双语短语采用多种方式进行评分（scoring）：

1. 短语翻译概率 $\phi(e|f)$
2. 词汇化权重 $lex(e|f)$
3. 短语翻译概率 $\phi(f|e)$
4. 词汇化权重 $lex(f|e)$
5. 短语惩罚（通常是 $\exp(1)=2.718$）

在短语重排序上，Pharaoh 采用了惩罚位置偏移的方法。其计算方法是：

$$p_{D} = \exp\left(-\sum_{i} d_{i}\right) \quad \text{公式 6.2}$$

其中，对于每个目标语言短语，d 的计算方法如下：前一个被翻译的短语的最后一个词的位置加上 1，然后再减去新翻译的短语的第一个词的位置，最后取绝对值。

[9] 可在 Philipp Koehn 的个人主页 http://www.iccs.inf.ed.ac.uk/~pkoehn/下载，包含完整的训练工具和解码器。

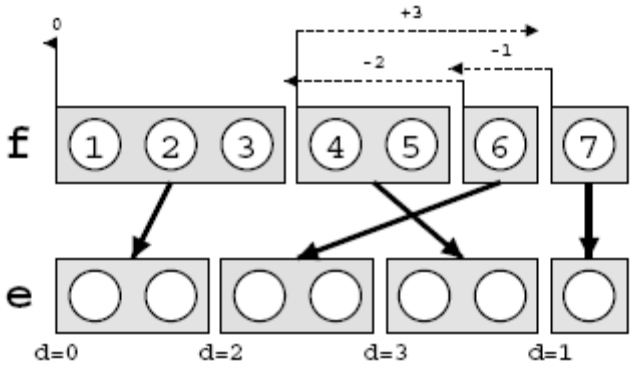

图 6.1：Pharaoh 的重排序模型。

图 6.1 解释了如何计算 d。以第 2 个目标短语为例，前一个被翻译的源语言短语（即第 1 个）的最后一个词的位置是 3，新翻译的源语言短语（即第 3 个）的第一个词的位置是 6，因此 $d = abs(3+1-6) = 2$。

这种方法的基本思想是限制大幅度的位置偏移，并不考虑短语或者词本身。在后面，我们会用第三章提到的 TNR 和这个重排序模型做比较。

6.1.2 数据和工具

在本实验中，我们以汉语为源语言，以英语为目标语言。

我们所使用的汉英双语语料库主要来自语言学数据联盟（Linguistic Data Consortium）[10]，具体由表 6.1 列出的语料库组成[11]。

表 6.1：训练语料库来源

编号	名称
LDC2002E18	Xinhua Chinese-English Parallel News Text Version 1.0 beta2
LDC2004T07	Multiple Translation Chinese Part3
LDC2005T06	Chinese News Translation Text Part1
LDC2003E07	Chinese Treebank English Parallel Corpus

表 6.2：训练语料库统计信息

	汉语	英语
句子	31,149	
词语	843,256	949,583

[10] 即 LDC，网址是 http://www.ldc.upenn.edu/

[11] LDC2002E18 只用了前 1.5 万句。此外，还有不少句子句法分析失败。因此，真正使用的句对数只有 31149。

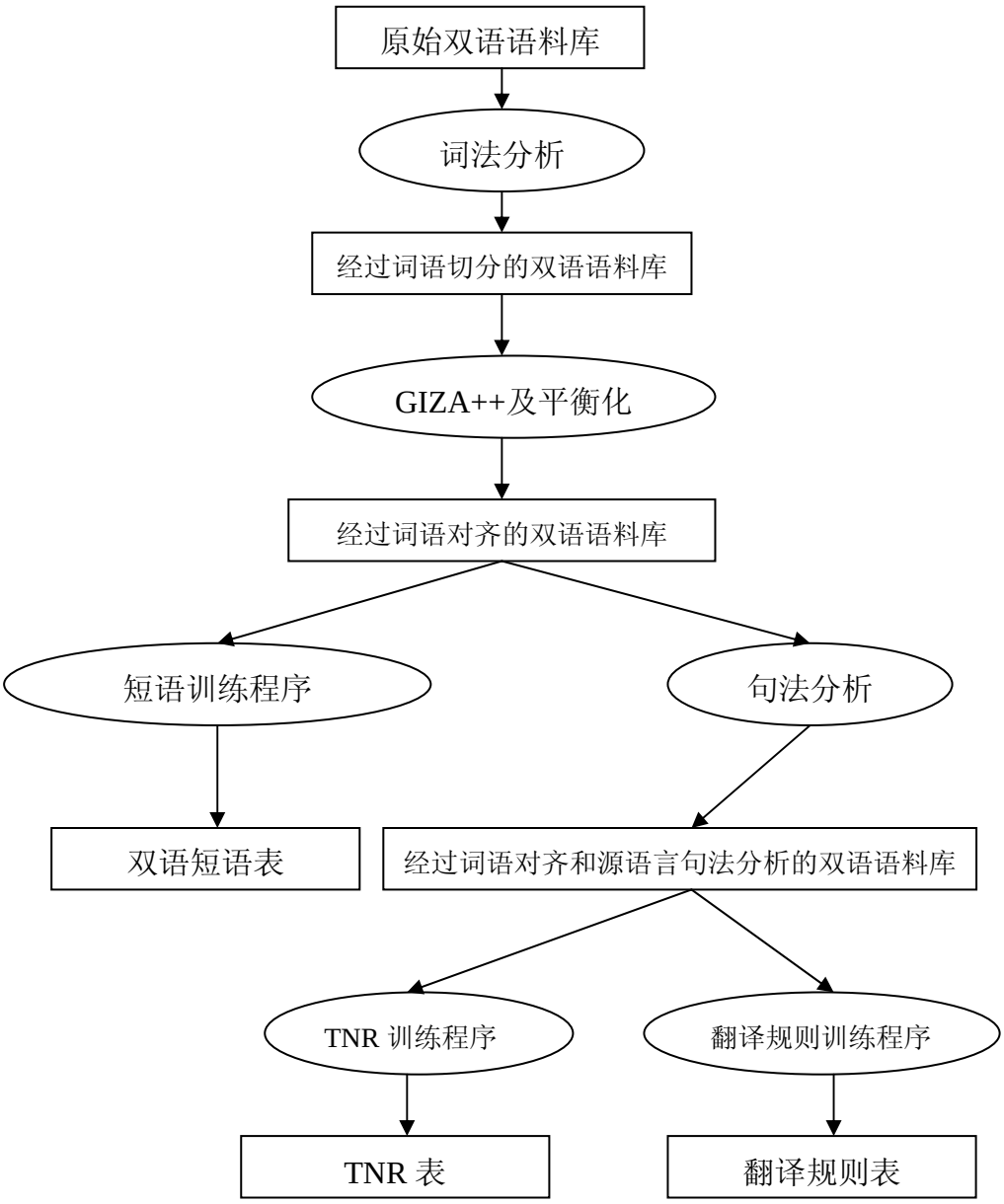

图 6.2：参数估计流程图

表 6.2 给出了训练语料库的统计信息：包含 31,149 句对，843,256 个汉语词，949,583 个英语词。

我们从 2002 年 NIST 机器翻译评测汉译英测试集中挑选了 571 句较短的句子作为开发集，用来自动调节对数线性模型的特征权重。测试集选用的是 2005 年 NIST 机器翻译评测汉译英测试集，共包含 1,082 个句子。

我们使用 SRI Language Modeling Toolkit[Stolcke 2002]在训练语料库的 31，149 英语句子上训练一个三元语言模型，平滑方法采用的是 Kneser-Ney

smoothing[Chen 1998]。

评测工具采用的是NIST官方网站发布的mteval-v11.pl[12]，以自动评价指标BLEU[Papineni 2002]来衡量翻译质量。在本章，所有的BLEU值都是以大小写敏感的方式计算的。

关于最小错误率训练[Och 2003a]，我们采用[Venugopal 2005]开发的optimizeV5IBMBLEU.m[13]。原始程序是用Matlab编写的，我们用C++进行了重新实现。

6.1.3 参数估计

流程

图 6.2 给出了参数估计的流程图。

我们首先对原始双语语料库做词法分析。我们使用中国科学院计算技术研究所汉语词法分析系统 ICTCLAS[Zhang 2003]对汉语句子进行词语切分，自行开发了一个简单的工具对英语句子进行 tokenization。

之后，利用 GIZA++和平衡化生成词语对齐，再从经过词语对齐的双语语料库上抽取双语短语表，这两部分都是 Pharaoh 的训练工具包完成的。

我们采用由熊德意开发的句法分析器[Xiong 2005]对训练语料库的汉语句子做句法分析。这个句法分析器以汉语宾州树库 1.0 版的 1-270 篇作为训练集，取得了 79.4%的 F1 值。已知经过词语对齐和源语言句法分析的双语语料库，就可以自动获取 TNR、树到串对齐模板和森林到串翻译规则。

双语短语表

我们在训练语料库上总共抽取了 1,358,900 个双语短语，可用于 524,887 个汉语短语。根据开发集和测试集对这些短语进行过滤，得到 251,173 个双语短语，可用于 22,077 个汉语短语，其中包含 158,822 个句法双语短语，可用于 7,294 个汉语短语。关于双语短语表的详细情况见表 6.3。

表 6.3：双语短语表统计信息

		条目数量	可用数量	平均候选数量
过滤前	总共	1,358,900	524,887	2.59
过滤后	句法双语短语	158,822	7,294	21.77
	非句法双语短语	92,351	14,783	6.25
	总共	251,173	22,077	11.38

[12] 可在 http://www.nist.gov/speech/tests/mt/resources/scoring.htm 下载。
[13] 可在 http://www.cs.cmu.edu/~ashishv/mer.html 下载。

表 6.4：双语短语表例子

编号	汉语短语	英语短语	是否被过滤	是否句法短语
1	布什	Bush	否	是
2	中国 经济	China economy	是	-
3	总统 发表	President made	否	否
4	发表 演讲	made a speech	否	是
5	发表 演讲	give a talk	否	是
6	迅速	Rapid	是	-

下面，我们以一个例子来解释这些统计信息的意义。假设我们从训练语料库上得到如表 6.4 所示的双语短语表。为了表示上的简单，我们省略了概率和对齐信息。

假设测试集只有一句话“布什 总统 发表 演讲”，其句法分析树见图 3.6。那么，编号为 2 和 6 的双语短语都无法在测试集上被使用，它们将被过滤掉以减少双语短语表的规模，这样在解码时对内存的需求也会降低。过滤后，有 4 个双语短语被保留了，其中有 3 个句法短语，1 个非句法短语。编号为 3 的双语短语之所以是非句法短语，是因为测试集上没有句法子树能够覆盖它。

所谓条目数量是指双语短语的数量，可用数量是指可翻译的汉语短语的数量，平均候选数量等于条目数量除以可用数量。例如，在表 6.4 中，条目数量为 6，可用数量是 5，平均候选数量是 1.2。

在表 6.3 中，过滤前短语表的平均候选数量是 2.59，因为大量的双语短语都是只有一个候选翻译，只有极少量常见词才会有很多的候选翻译，例如标点符号。经过过滤后，短语的平均候选数量上升至 11.38，说明大多数只有一个候选翻译的短语都不能在测试集上使用。而句法短语的平均候选数量高达 21.77，说明句法短语在训练语料库中出现的频度高，候选翻译的数量也比较多。

一个值得关注的现象是，虽然非句法短语在条目数量上低于句法短语，但是在可用数量是句法短语的两倍。如果不使用这些非句法短语，势必会降低翻译质量，我们会从后面的实验结果中得到验证。

TNR 表

表 6.5：TNR 表统计信息

树高	条目数量	可用数量	平均候选数量
2	1,941	1,305	1.49
3	17,519	13,665	1.28
4	44,517	40,260	1.11
5	64,018	62,079	1.03

表 6.5 给出了 TNR 表的统计信息。可以看出，随着 TNR 的树高的增加，抽取的 TNR 数量也不断增加。一个值得注意的现象是，随着树高的增长，平均候选数量不断下降。这是合理的，因为树高越大，在训练语料库中出现的频度就越小，出现不同重排序情况的可能性就越低。但是，即使树高为 2，平均候选数量也仅为 1.49，说明给定一个句法子树，大多数情况下都是只有一种重排序在训练样本中出现。

翻译规则表

我们使用图 5.4 描述的算法同时抽取树到串规则（简称树规则）和森林到串规则（森林规则）。图 6.6 给出了翻译规则表的统计信息。

我们采用以下三个限制条件来控制抽取的翻译规则的数量：

1. 规则中树的高度不大于 3
2. 规则中树节点的孩子数量不大于 5
3. 规则中树叶子节点的数量不大于 7

即便如此，在训练语料库上抽取的翻译规则的数量仍然十分庞大，在包含约三万句对的训练语料库上抽出接近四百万树规则和森林规则，在解码时完全使用这些规则会造成极大的内存需求。因此，我们利用开发集和测试集对规则表进行过滤，方法是抽出开发集和测试集上所有满足上面三个限制条件的句法树和句法森林，如果从训练集中获取的规则的树或者森林不在其中，则被过滤掉。

我们在开发集和测试集的句法树上总共抽取了 247,841 个句法子树和 19,101,983 个句法森林，然后据此对翻译规则表进行过滤。总共有 10.3%的规则被保留下来，其中 27.9%的树规则被保留，8.5%的森林规则被保留。在经过过滤后的规则表中，词汇化规则是树规则的主体，部分词汇化规则是森林规则的主体。

需要指出的是，词汇化、部分词汇化和非词汇化的可用数量加起来并不等于总共的可用数量。以下面两个树规则为例：

表 6.6：翻译规则表统计信息

			条目数量	可用数量	平均候选数
过滤前	树规则	词汇化	138,851	66,607	2.08
		部分词汇化	220,235	176,644	1.25
		非词汇化	5,486	4,213	1.30
		总共	364,572	245,870	1.48
	森林规则	词汇化	268,653	236,490	1.14
		部分词汇化	3,176,504	2,969,879	1.07
		非词汇化	49,084	43,598	1.13
		总共	3,494,241	3,239,090	1.08
	总共		3,858,813	3,484,960	1.11
过滤后	树规则	词汇化	56,983	10,010	5.69
		部分词汇化	41,027	15,867	2.59
		非词汇化	3,529	2,335	1.51
		总共	101,539	26,863	3.78
	森林规则	词汇化	16,609	6,869	2.42
		部分词汇化	254,346	134,568	1.89
		非词汇化	25,051	19,935	1.26
		总共	296,006	152,151	1.95
	总共		397,545	179,014	2.22

$$\left\langle "\left(\mathrm{NP}\left(\mathrm{NR}\right)\left(\mathrm{NN}\right)\right)","X_1X_2","\left\{(1,2),(2,1)\right\}\right\rangle$$

$$\left\langle "\left(\mathrm{NP}\left(\mathrm{NR}\right)\left(\mathrm{NN}\right)\right)","X_1 \text{ and } X_2","\left\{(1,1),(2,3)\right\}\right\rangle$$

第一个规则是非词汇化规则，第二个是部分词汇化规则。因此，非词汇化规则的可用数量是 1，部分词汇化规则的可用数量是 1，而总体的可用数量也是 1。因此从总体上看，这两个规则都只能用于“(NP (NR) (NN))”这棵树。

6.1.4 后处理

我们对于所有系统的翻译结果采用两种后处理：一是将未登录词从译文中删除，二是将句首的英语词的首字母大写。

采用后处理大概能使 BLEU 值提高 1 个百分点。

6.2 对比实验结果

表 6.7：对比实验结果

系统	规则	BLEU4
Pharaoh	BP	0.2182 ± 0.0089
	SBP	0.2033 ± 0.0087
Model 1	TNR + SBP	0.2123 ± 0.0085
Model 2	SBP	0.1912 ± 0.0085
	TR	0.2302 ± 0.0089
	TR + SBP	0.2346 ± 0.0088
Model 3	BP	0.2059 ± 0.0083
	TR + FR + AR	0.2402 ± 0.0087

在表 6.7 中，我们给出了对比实验的结果。我们设置剪枝参数如下（见 3.4.5 节）：$a=20$，$\alpha=0$，$b=100$，$\beta=0$。

其中，“Model 1”是嵌入句法树的基于短语的翻译模型，“Model 2”是基于树到串对齐模板的翻译模型，“Model 3”是引入森林到串规则的树到串翻译模型。“BP”是指双语短语，“SBP”是指句法双语短语，“TR”是指树到串翻译规则（即树到串对齐模板），“FR”是森林到串翻译规则，“AR”是辅助规则。我们采用[Zhang 2004]的方法[14]计算置信区间。

从表中可以看出，Pharaoh 只使用句法双语短语要比使用全部双语短语降低 1.5 个百分点，说明只使用句法双语短语对于基于短语的系统而言确实会降低翻译性能。

Model 1 能够只能使用句法双语短语，利用 TNR 实现短语间的重排序。在只使用句法双语短语的情况下，Model 1 要优于 Pharaoh，说明利用 TNR 进行重排序的方法要优于惩罚位置偏移的方法，原因是前者得到了句法信息的指导。

Model 2 可以使用句法短语和默认规则实现单调搜索，结果仅为 0.1912。Model 2 只使用树到串翻译规则，BLEU 值可以达到 0.2302，比 Pharaoh 高出 1.3 个百分点。如果把句法短语视作特殊的词汇化规则，可以带来微弱的提高。这种方法的局限性我们已在第 4 章讨论过，即只能增大词汇化树规则的数量，无法利用非句法双语短语。

[14] 可在 http://projectile.is.cs.cmu.edu/下载。

Model 3 可以使用全部短语和默认规则实现单调搜索，结果比 Model 2 使用句法短语高 1.48 个百分点，再次证明了利用非句法短语的重要性。如果 Model 3 采用树到串规则、森林到串规则和辅助规则，BLEU 值可以达到 0.2402，比 Pharaoh 使用全部短语高 2.3 个百分点，比 Model 2 只使用树规则高 1 个百分点，这些差别在统计意义上都是显著的。为了得到这个最好的结果，解码器在搜索过程中使用了 26,082 个树到串翻译规则，9,219 个默认规则，5,432 个森林到串规则和 2,919 个辅助规则。这些数据表明，虽然加入森林到串规则和辅助规则确实能带来翻译质量的提高，树到串规则在树到串翻译模型的搜索中仍然占据主导地位。

表 6.8：不同词汇化程度的森林规则的影响

森林到串规则集	BLEU
无	0.2225 ± 0.0085
词汇化	0.2297 ± 0.0081
部分词汇化	0.2279 ± 0.0083
非词汇化	0.2270 ± 0.0087
全部	0.2312 ± 0.0082

表 6.8 考察了森林到串规则的词汇程度的影响。我们设置剪枝参数如下：$a=3$，$\alpha=0$，$b=10$，$\beta=0$。其中，“无”表示只使用树到串规则，“词汇化”表示同时使用树到串规则和词汇化森林到串规则，“部分词汇化”表示同时使用树到串规则和部分词汇化森林到串规则，“非词汇化”表示同时使用树到串规则和非词汇化森林到串规则，“全部”表示使用树到串规则和全部森林到串规则。我们发现词汇化森林规则起到的作用最明显。

因此，我们可以得到几个结论：

1. 无论是对于基于短语的翻译模型，还是对于基于句法的翻译模型，使用非句法双语短语都是十分必要的。换句话说，只使用句法短语势必会降低翻译性能。
2. 利用句法信息进行短语重排序要明显优于惩罚位置偏移的方法。
3. 基于树到串对齐模板的翻译模型在翻译性能上要优于 Pharaoh。
4. 森林到串翻译规则和辅助规则的引入不仅能使树到串翻译模型使用非句法短语，而且还具备泛化能力，能提高树到串翻译模型的性能。

6.3 在大规模数据上的结果

我们根据模型 2 开发的基于树到串对齐模板的统计机器翻译系统 Lynx，作为

主系统代表中国科学院计算技术研究所参加 2006 年 NIST 机器翻译评测。该系统采用了本文第四章描述的所有技术，除了使用树到串对齐模板，还将双语短语视作词汇化模板，并且利用双语短语做后处理提高译文流利度。

表 6.9：2006 年 NIST 评测训练语料库来源

编号	名称
LDC2003E14	FBIS Metalanguage Texts
LDC2004T08	Hong Kong Parallel Text
LDC2002E18	Xinhua Chinese-English Parallel News Text Version 1.0 beta2
LDC2004T07	Multiple Translation Chinese Part3
LDC2005T06	Chinese News Translation Text Part1
LDC2003E07	Chinese Treebank English Parallel Corpus
LDC2005E47	Chinese English News Magazine Parallel Text

表 6.9 列出了训练语料库的来源。该双语语料库共 254 万句对，包含 6 千 8 百万汉语词和 7 千 4 百万英语词。

我们使用了全部 254 万句对来抽取双语短语。针对开发集和测试集过滤后，双语短语的数量是 8,792,651，可用于 95,687 个汉语词串。

我们使用了 254 万句对中的 80 万来抽取树到串对齐模板。在抽取过程中，树的最大高度限制为 3，节点的最大孩子数限制为 5。针对开发集和测试集过滤后，树模板的数量是 938,633，可用于 79,494 个汉语树。

同时，我们用基于规则的方法对人名、日期和数字进行翻译。当对一个汉语词串找不到可用的树模板和双语短语时，我们才使用基于规则的翻译。在评测中，我们使用了 2,614 个翻译，可用于 1,410 个汉语词串。

我们使用 SRI 语言模型工具训练了两个四元语言模型。一个是双语语料库的英语部分，包含约 7 千 4 百万英语词。另一个是 Gigaword 语料的新华部分，包含约 1 亿 8 千 1 百万英语词。这两个语言模型被作为独立的特征使用。

我们的系统Lynx在NIST子集上取得了第 5 名，在GALE子集上取得了第 8 名[15]。我们所使用的双语语料库相对较小，语言模型的规模也不大，远不能和排名靠前的单位相比。在这样的条件下能取得这样的好成绩，证明基于树到串对齐模板的翻译模型在理论上确实具备一定的先进性。

[15] NIST 官方公布的成绩见 http://www.nist.gov/speech/tests/mt/mt06eval_official_results.html。

第七章 结论

机器翻译研究如何利用计算机把一种自然语言的文本翻译成另外一种自然语言的文本，是一个具有巨大应用价值的研究课题。近十年来，统计机器翻译取得了很大的成功，在多次国际性机器翻译评测中取得领先成绩，成为目前机器翻译的主流技术。

本文重点研究了统计机器翻译中的两个关键问题：词语对齐和翻译模型。

词语对齐对统计机器翻译而言至关重要，因为经过词语对齐的语料是极有价值的翻译知识源。目前词语对齐的主流方法是 IBM 模型，作为生成模型，它具有许多难以克服的缺陷，如难以扩充模型以容纳新的信息。

本文提出的词语对齐的对数线性模型是第一个将判别方法引入词语对齐的工作。在词语对齐的对数线性模型中，所有的知识源被视作依赖于源语言句子、目标语言句子以及可能的其他变量的特征函数。对数线性模型使统计对齐模型易于扩展，方便加入更多的语言学信息，从而能同时处理与具体语言相关和不相关的语言现象。本文讨论了框架的形式化定义、特征函数设计、参数训练、搜索算法以及 n-best 列表生成等问题。我们在三个词语对齐评测的数据集（包含五个语言对）上对词语对齐的对数线性模型进行评价。实验表明，对数线性模型超过了绝大多数参加评测的系统。

翻译模型设计是统计机器翻译的核心问题，不仅直接决定了训练和搜索算法的设计，而且从根本上决定了模型的翻译性能。目前，统计机器翻译模型可分为三类：基于词的模型、基于短语的模型和基于句法的模型。基于词的模型已经过时，基于短语的模型的发展空间已经不大，而基于句法的模型则成为目前的研究热点。

本文提出了三个基于句法的树到串翻译模型：嵌入句法树的基于短语的翻译模型、基于树到串对齐模板的翻译模型和融入森林到串规则的树到串翻译模型。

嵌入句法树的基于短语的翻译模型（简称模型 1）以隐变量的方式在源语言端嵌入句法树，可以利用句法信息指导短语重排序。在句法树的约束下，模型 1 只能够使用句法双语短语。我们提出树节点重排序来描述短语重排序。理论上，任何句法短语划分都可以被一个树节点重排序序列描述。与[Xia 2004]和[Collins 2005]不同，模型 1 首次真正实现了从建模上利用句法信息指导短语重排序，而不是作为预处理手段。

基于树到串对齐模板的翻译模型（简称模型 2）是对模型 1 的发展。在模型 1 中，真正执行翻译的是句法双语短语，树节点重排序只起到描述短语重排序的作用。模型 2 将树节点重排序升华为树到串对齐模板。树到串对齐模板描述了源语言句法树和目标语言串之间的对应关系，源语言端和目标语言端都既包含终结符也包含非终结符。因此，只使用树到串对齐模板就能够既执行翻译又执行重排序，不必像[Quirk 2005]那样设计专门的重排序模型。在重排序方面，树到串对齐模板既能执行局部重排序又能执行全局重排序，而且同时能实现词汇化、部分词汇化和非词汇化重排序。模型 1 使用的树节点重排序实际上等价于非词汇化树到串对齐模板。模型 2 在结构上非常简单，相对于传统的基于句法的模型训练和搜索的复杂度大大降低，同时保持很高的翻译性能。

融入森林到串规则的树到串翻译模型（简称模型 3）对模型 2 进行进一步的扩充。树到串对齐模板实际上是树到串翻译规则，它要求源语言端必须是一棵句法树。正是因为这个限制，树到串规则只能表达句法双语短语并进行泛化，无法表达和泛化非句法短语。研究表明，同时使用句法双语短语和非句法双语短语对于基于句法的模型极为重要。为此，我们提出森林到串翻译规则来描述多棵句法树和一个串之间的对应关系，从而能够表达和泛化全部的双语短语。为了将森林到串规则融入到树到串模型，我们提出辅助规则来提供泛化层。辅助规则并不是从训练语料库中自动学习的，而是在搜索时动态构造的。模型 3 在保留模型 2 的全部优点的同时增加了表达能力，使翻译性能得到进一步提升。

这三个翻译模型一脉相承，其基本思想都是对源语言端进行句法分析，根据源语言句法树搜索目标语言串。模型 1 在本质上仍然是基于短语的，使用句法双语短语进行翻译，只是利用树节点重排序执行短语重排序。这个模型的缺点在于无法利用非句法双语短语，这极大地影响了翻译性能。此外，树节点重排序的功能有所限制，只能执行非词汇化重排序。模型 2 所采用的树到串对齐模板大大提升了模型的表达能力，不但能执行翻译，而且重排序能力也比树节点重排序强。但是，模型 2 仍然无法表达和泛化非句法双语短语。模型 3 采用的森林到串规则和辅助规则很好地解决了这个问题。模型 3 完全兼容模型 2，表达能力在三个模型中是最强的，但代价是训练和搜索的复杂度也最高。

根据实验结果，我们可以得到以下结论：

1. 模型 1 的翻译性能接近于国际学术界最常用的基于短语的翻译系统 Pharaoh。
2. 模型 2 的翻译性能明显超过 Pharaoh，并在 2006 年 NIST 机器翻译评测汉译英任务的两个子项上分别取得第 5 名和第 8 名的好成绩，达到世界先进水平。
3. 模型 3 的翻译性能在小规模数据上明显超过模型 2，在大规模数据上的

效果还有待验证。

我们下一步的工作包括：

1. 研究支持多对多对应关系的词语对齐模型。IBM 模型的一大缺陷在于只支持一对多对应关系，多对多对齐只能通过平衡化的手段生成。支持多对多对应关系的词语对齐模型是非常有意义的研究题目。
2. 在大规模数据上考察模型 3 的翻译性能。我们将使用更大的训练语料库和语言模型，在 NIST 评测测试集上考察模型 3 的翻译性能。

参考文献

[Al-Onaizan 2006] Yaser Al-Onaizan and Kishore Papineni. 2006. Distortion models for statistical machine translation. In *Proceedings of COLING/ACL 2006*, pages 529-536, Sydney, Australia, July.

[Alshawi 2000] Hiyan Alshawi, Srinivas Bangalore, and Shona Douglas. 2000. Learning dependency translation models as collections of finite-state head transducers. *Computational Linguistics*, 26:45-60.

[Berger 1996] Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39-72.

[Blunsom 2006] Phil Blunsom and Trevor Cohn. 2006. Discriminative word alignment with conditional random fields. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of Association for Computational Linguistics*, pages 65-72, Sydney, Australia, July.

[Brill 1995] Eric Brill. 1995. Transformation-based-error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics, 21(4),* September.

[Brown 1993] Peter F. Brown, Stephan A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter Estimation. *Computational Linguistics,* 19(2):263-311.

[Chen 1998] Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. *Technical Report TR-10-98*, Harvard University Center for Research in Computing Technology.

[Chiang 2005] David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL 2005*, pages 263-270, Ann Arbor, Michigan, June.

[Collins 2005] Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause reconstructuring for statistical machine translation. In *Proceedings of ACL 2005*, pages 531-540, Ann Arbor, Michigan, June.

[Darroch 1972] J. N. Darroch and D. Ratcliff. 1972. Generalized iterative scaling for

log-linear models. *Annals of Mathematical Statistics*, 43:1470-1480.

[Diab 2000] Mona Diab. 2000. An unsupervised method for multilingual word sense tagging using parallel corpora: A preliminary investigation. In *Proceedings of ACL 2000 Workshop on Word Senses and Multilinguality*, pages 1-9, Hong Kong, October.

[Ding 2005] Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of ACL 2005, pages 541-548*, Ann Arbor, Michigan, June.

[Fraser 2005] Alexander Fraser and Daniel Marcu. 2005. ISI's Participation in the Romanian-English Alignment Task. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 91-94, Ann Arbor, June.

[Galley 2004] Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of HLT/NAACL 2004*, pages 273-280, Boston, USA, May.

[Galley 2006] Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang; and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of COLING/ACL 2006*, pages 961-968, Sydney, Australia, July.

[Germann 2001] Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast decoding and optimal decoding for machine translation. In *Proceedings of ACL 2001*, pages 228-235, Toulouse, France, July.

[Graehl 2004] Jonathan Graehl and Kevin Knight. 2004. Training tree transducers. In *Proceedings of NAACL-HLT 2004*, pages 105-112.

[He 2006] Zhongjun He, Yang Liu, Deyi Xiong, Hongxu Hou, and Qun Liu. 2006. ICT System Description for the 2006 TC-STAR Run#2 SLT Evaluation. In *Proceedings of TC-STAR Workshop on Speech-to-Speech Translation*, pages 63-68, Barcelona, Spain, June.

[Ker 1997] Sue J. Ker and Jason S. Chang. 1997. A class-based approach to word alignment. *Computational Linguistics*, 23(2):313-343.

[Knight 1999] Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics*, Squibs and Discussion, 25(4):607-615.

[Knight 2005] Kevin Knight and Jonathan Graehl. 2005. An overview of probabilistic tree transducers for natural language processing. In *Proceedings of CICLing 2005*, pages 1-25. Springer Verlag.

[Koehn 2003] Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT/NAACL 2003*, pages 127-133.

[Koehn 2004] Phillip Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of the Sixth Conference of the Association for Machine Translation in the Americas*, pages 115-124.

[Liu 2005] Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *Proceedings of ACL 2005*, pages 459-466, Ann Arbor, USA, June.

[Liu 2006] Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of COLING/ACL 2006*, pages 609-616, Sydney, Australia, July.

[Marcu 2002] Daniel Marcu and William Wong. 2002. A phrase-based, jointed probability model for statistical machine translation. In *Proceedings of EMNLP 2002*, pages 133-139.

[Marcu 2006] Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical machine translation with syntactified target language phrases. In *Proceedings of EMNLP 2006*, pages 44-52.

[Melamed 2000] I. Dan Melamed. 2000. Models of translation equivalence among words. *Computational Linguistics*, 26(2):221-249.

[Melamed 2003] I. Dan Melamed. 2003. Multitext grammars and synchronous parsers. In Proceedings of HLT/NAACL 2003, pages 79-86.

[Moore 2005] Robert Moore. 2005. A discriminative framework for bilingual word alignment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 81-88, Vancouver, British Columbia, Canada, October.

[Moore 2006] Robert C. Moore. Wen-tau Yih, and Andreas Bode. 2006. Improved discriminative word alignment. In *Proceedings of COLING/ACL 2006*, pages 513-520, Sydney, Australia, July.

[Och 1999] Franz J. Och, Christoph Tillman, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of EMNLP 1999*, pages 20-28, College Park, Maryland, June.

[Och 2000] Franz J. Och and Hermann Ney. 2000. A comparison of alignment models for statistical machine translation. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 1086-1090, Saarbrucken, Germany, July.

[Och 2002a] Franz J. Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL 2002*, pages 295-302, Philadelphia, PA, July.

[Och 2002b] Franz J. Och. 2002. Statistical machine translation: From single-word models to alignment templates. *Ph.D. thesis*, Computer Science Department, RWTH Aachen, Germany, October.

[Och 2003a] Franz J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL 2003*, pages 160-167, Sapporo, Japan.

[Och 2003b] Franz J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1)19-51.

[Och 2004a] Franz J. Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *Proceedings of HLT/NAACL 2004*, Boston, MA, May.

[Och 2004b] Franz J. Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417-449.

[Papineni 1997] Kishore A. Papineni, Salim Roukos, and Todd Ward. 1997. Feature-based language understanding. In *European Conference on Speech Communication and Technology*, pages 1435-1438, Rhodes, Greece, September.

[Papineni 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL 2002*, pages 311-318, Philadelphia, PA, July.

[Press 2002] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P.

Flannery. 2002. Numerical Recipes in C++. *Cambridge University Press*, Cambridge, UK.

[Shemtov 1993] H. Shemtov. 1993. Text alignment in a tool for translating revised documents. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, pages 449-453, Utrecht, Netherlands.

[Smadja 1996] F. Smadja, K. R. MaKeown, and V. Hatzivassiloglou. 1996. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 22(1):1-38.

[Stolcke 2002] Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*, volume 2, pages 901-904.

[Quirk 2005] Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of ACL 2005*, pages 271-279, Ann Arbor, Michigan, June.

[Quirk 2006a] Chris Quirk and Arul Menezes. 2006. Do we need phrases? Challenging the conventional wisdom in statistical machine translation. In *Proceedings of HLT/NAACL 2006*, pages 9-16, New York, USA, June.

[Quirk 2006b] Chris Quirk and Simon Corston-Oliver. 2006. The impact of parse quality on syntactically-informed statistical machine translation. In *Proceedings of EMNLP 2006*, pages 62-69, Sydney, Australia, July.

[Taskar 2005] Ben Taskar, Simon Lacoste-Julien, and Dan Klein. 2005. A discriminative matching approach to word alignment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 73-80, Vancouver, British Columbia, Canada, October.

[Tillmann 2005] Christoph Tillmann and Tong Zhang. 2005. A localized prediction model for statistical machine translation. In *Proceedings of ACL 2005*, pages 557-564, Ann Arbor, Michigan, June.

[Venugopal 2005] Ashish Venugopal and Stephan Vogel. 2005. Considereations in maximum mutual information and minimum classification error training for statistical machine translation. In *Proceedings of the Tenth Conference of the European Association for Machine Translation (EMAT-05).*

[Vogel 1996] Stephan Vogel and Hermann Ney. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 836-841, Copenhagen, Denmark, August.

[Wang 1997] Ye-Yi Wang and Alex Waibel. 1997. Decoding algorithm in statistical machine translation. In *Proceedings of ACL 1997*, pages 366-372, Madrid, Spain,

[Wang 1998] Ye-Yi Wang and Alex Waibel. 1998. Modeling with structures in statistical machine translation. In *Proceedings of COLING/ACL 1998*, pages 1357-1363, Montreal, Quebec, Canada.

[Wu 1996] Dekai Wu. 1996. A polynomial-time algorithm for statistical machine translation. In *Proceedings of ACL 1996*, pages 152-158, Santa Cruz, California, June.

[Wu 1997] Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23:377-404.

[Xia 2004] Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of COLING 2004*, pages 508-514, Geneva, Switzerland, August.

[Xiong 2005] Deyi Xiong, Shuanglong Li, Qun Liu, Shouxun Lin, and Yueliang Qian. 2005. Parsing the Penn Chinese Treebank with semantic knowledge. In *Proceedings of IJCNLP 2005*, pages 70-81.

[Xiong 2006] Deyi Xiong, Qun liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of COLING/ACL 2006*, pages 521-528, Sydney, Australia, July.

[Yamada 2001] Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of ACL 2001*, pages 523-530.

[Zens 2004] Richard Zens, Hermann Ney, Taro Watanabe, and Eiichiro Sumita. 2004. Reordering constraints for phrase-based statistical machine translation. In *Proceedings of COLING 2004*, pages 205-211, Geneva, Switzerland, August.

[Zhang 2003] Huaping Zhang, Hongkui Yu, Deyi Xiong, and Qun Liu. 2003. Hhmm-based Chinese lexical analyzer ICTCLAS. In *Proceedings of the*

second SigHan workshop affiliated with 41st ACL, pages 184-187, Sapporo, Japan.

[Zhang 2004] Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting bleu/nist scores how much improvement do we need to have a better system? In *Proceedings of Fourth International Conference on Language Resources and Evaluation*, pages 2051-2054.

致　谢

历经三年的紧张艰苦的研究工作，在前人研究成果的基础上和大家的热情帮助下，我得以完成这篇论文。在此，我特别感谢所有曾经帮助过我、为我的研究工作提出建议和指导的人们。

首先，我衷心感谢我的导师林守勋研究员和刘群研究员几年来对我的关心和指导。他们渊博的学识、严谨的治学态度、孜孜不倦忘我工作的精神和灵活的思路都使我受益匪浅。

感谢钱跃良主任和吕雅娟老师。他们一直关心着我所做的研究工作，并提出了许多建设性的意见。他们的治学态度和进行科研工作的方法不但在博士研究工作中给我以启迪，也将对我将来的工作和学习起到重大的作用。

感谢曾经在自然语言处理组学习的几位师兄和师姐：邹纲、邓丹和俞鸿魁。他们在我刚进入计算所的时候给予了极大的帮助。感谢刘宏、骆卫华、侯宏旭、熊德意、王向东、何中军、崔世起、杜守栓、李金国、黄瑾、米海涛、傅雷、黄利科、李梅茵、谢峰、黄赟、叶莎妮、高扬、赵丹、刘乐中、李双龙、牧仁高娃和赵明早，与他们的共事使我学到了很多，我为在这样一个团结向上的集体中工作学习而骄傲。感谢远在美国的张浩师兄，2005 年他代我宣讲 ACL 论文，2006 年在悉尼参加 COLING/ACL 2006 时对我照顾有加。

感谢刘卫玲、雷俊、刘玉东和李娜等几位秘书，她们对我的学习和工作给予了极大的帮助。感谢石锦彩老师几年来不遗余力地支持我们的系统测试，她高度认真负责的工作态度令我深怀敬意。感谢宋守礼、张晓辉和周世佳等几位老师在生活和学习上给予我的关心和帮助。

感谢所有关心和帮助过我的老师、同学和朋友。

最后感谢我的家人，是他们无尽的爱让我有勇气面对任何困难。

作者简历

姓名：刘洋　　性别：男　　出生日期：1979.6.21　　籍贯：湖北武汉

2002.9 – 2007.7　　中科院计算所计算机应用专业硕博连读生
1998.9 -- 2002.7　　武汉大学计算机系计算机应用专业本科生

【攻读博士学位期间发表的论文】

[1] Yang Liu, Yun Huang, Qun Liu, and Shouxun Lin. 2007. Forest-to-String Statistical Translation Rules. To appear in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, Prague, Czech, June.

[2] Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-String Alignment Template for Statistical Machine Translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL 2006)*, pages 609-616, Sydney, Australia, July.

[3] Zhongjun He, Yang Liu, Hongxu Hou, and Qun Liu. 2006. ICT System Description for the 2006 TC-STAR Run#2 SLT Evaluation. In *Proceedings of TC-STAR Workshop on Speech-to-Speech Translation*, pages 63-68, Barcelona, Spain, June.

[4] Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear Models for Word Alignment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 459-466, Ann Arbor, USA, June.

[5] 刘洋，刘群，林守勋。机器翻译评测中的模糊匹配。中文信息学报，第 19 卷，第 3 期，第 45 至第 53 页，2005 年。

[6] 刘洋，刘群。Fuzzy Matching in Machine Translation Evaluation。全国计算语言学第二届学生研讨会，第 267 至第 274 页，2004 年 8 月，北京。

[7] 邹纲，刘洋，刘群，孟遥，于浩，西野文人，亢世勇。基于 Internet 的新词语检测。中文信息学报，第 18 卷，第 6 期，第 1 至第 9 页，2004 年。

[8] Yueliang Qian, Shouxun Lin, Yongdong Zhong, Yang Liu, Hong Liu, and Qun Liu. 2004. An Introduction to Corpora Resources of 863 Program for Chinese Language Processing and Human-Machine Interaction. In *Proceedings of the 4th Workshop on Asian Language Resources (ALR-04) affiliated to the 1st International Joint Conference on Natural Language Processing (IJCNLP 2004)*, Hainan, China, March.

【攻读博士学位期间参加的科研项目】

[1] 规则可控的统计机器翻译方法研究（计算所知识创新课题，2006-2007）

[2] 基于短语结构转换模板的统计机器翻译方法研究（国家自然科学基金，2006-2009）

[3] 机器翻译新方法的研究（863 项目，2005-2006）

[4] 中文平台评价体系研究与基础数据库建设（863 重点项目，2004－2005）

[5] 中文信息处理与人机交互技术的测评系统和体系（863 项目，2003－2005）

[6] 基于 Internet 的新词语检测与分析（富士通研究开发中心委托开发项目，2003－2005）

[7] 奥运多语言信息服务系统资源建设和核心技术评测（北京市科委数字奥运十大专项课题，2003－2004）

[8] 中文平台总体技术研究与基础数据库建设（863 重点项目，2000－2003）

【攻读博士学位期间的获奖情况】

[1] 所长优秀奖，2006 年，中国科学院计算技术研究所

[2] 三好学生，2006 年，中国科学院计算技术研究所

[3] Meritorious Asian NLP Paper Award，2006 年，COLING/ACL 2006

[4] The Don and Betty Walker International Student Fund，2006 年，美国计算语言学协会

[5] The AFNLP-Nagao Conference Participation Award，2006 年，亚洲自然语言处理联盟

[6] The Don and Betty Walker International Student Fund，2005 年，美国计算语言学协会

[7] Outstanding Authors Prize of SONY Research Award，2005 年，SONY 公司

[8] 优秀论文奖，2004 年，全国计算语言学第二届学生研讨会