



南京大學
NANJING UNIVERSITY



自然语言处理

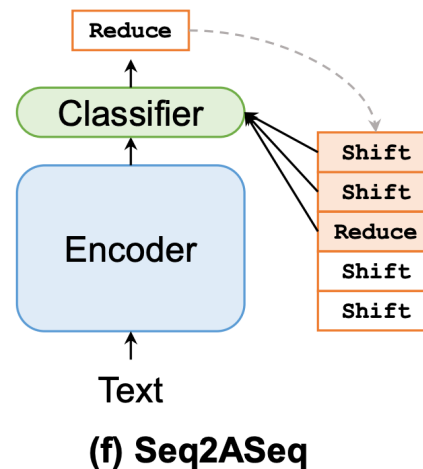
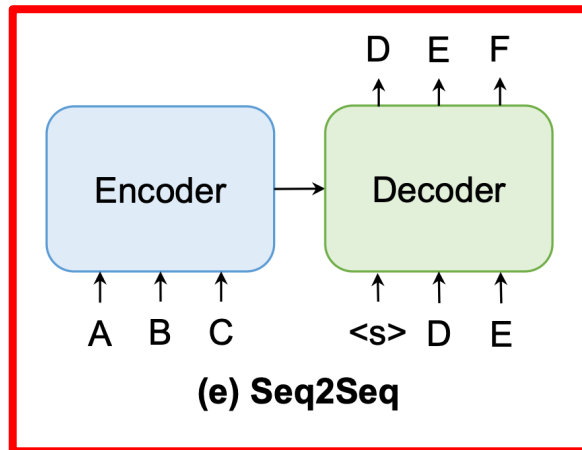
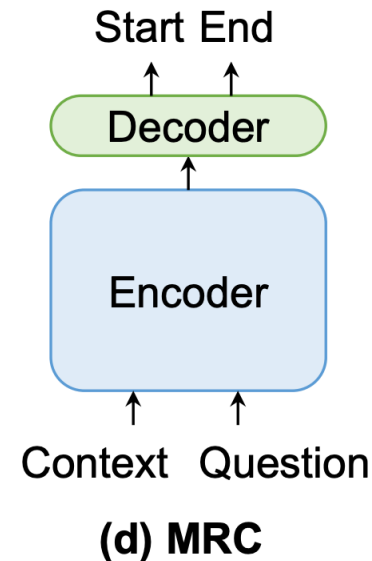
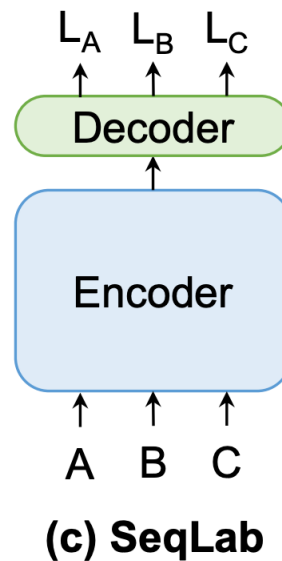
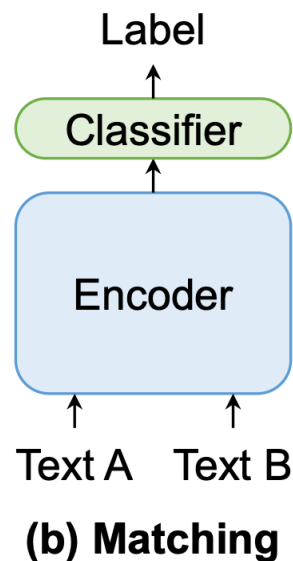
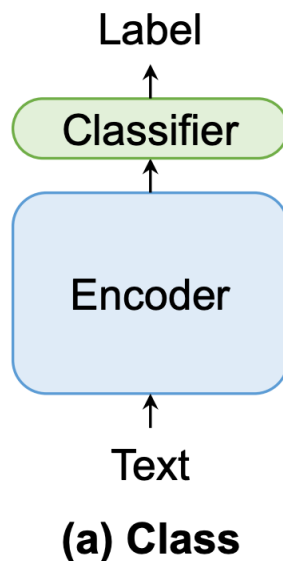
机器翻译

吴震

南京大学人工智能学院
南京大学自然语言处理研究组

2023年6月

自然语言处理中典型的任务形式



- 背景介绍
- 基于规则的机器翻译
- 基于实例的机器翻译
- 统计机器翻译
- 神经机器翻译
- 机器翻译评估



01



背景介绍

INTRODUCTION

- 机器翻译 (Machine Translation) 是一个将源语言的句子 x 翻译成目标语言句子 y (译文) 的任务。

源语言

我来自中国



$$\operatorname{argmax}_y P(y|x)$$

目标语言

I come from China

- 1947 , Warren Weaver提出机器翻译概念 (人工智能概念尚未提出) ;
- 1954 , 第一个公开展示的俄英翻译系统 , 走向热潮 ;
- 1966 , 美国科学院发布ALPAC报告 , 走入低谷 ;
- 1970s , 实用机器翻译系统TAUM-METEO , 将英文天气翻译为法文 , 重燃希望 ;
- 1990s-2000s , 统计机器翻译 , 逐渐火热 ;
- 2004 , Google发布多语言在线翻译引擎 , 走向应用 ;
- 2014-至今 , 神经机器翻译 , 全面繁荣。

1954 MT system video:

<https://voiceinthemachine.com/2012/06/18/can-machines-really-think/>

理性主义 VS 经验主义

- 理性主义：以生成语言学为基础，依靠人类先验知识
 - 基于规则的机器翻译
- 经验主义：以数据驱动为基础，从数据中学习经验和知识
 - 基于实例的机器翻译
 - 统计机器翻译
 - 神经机器翻译



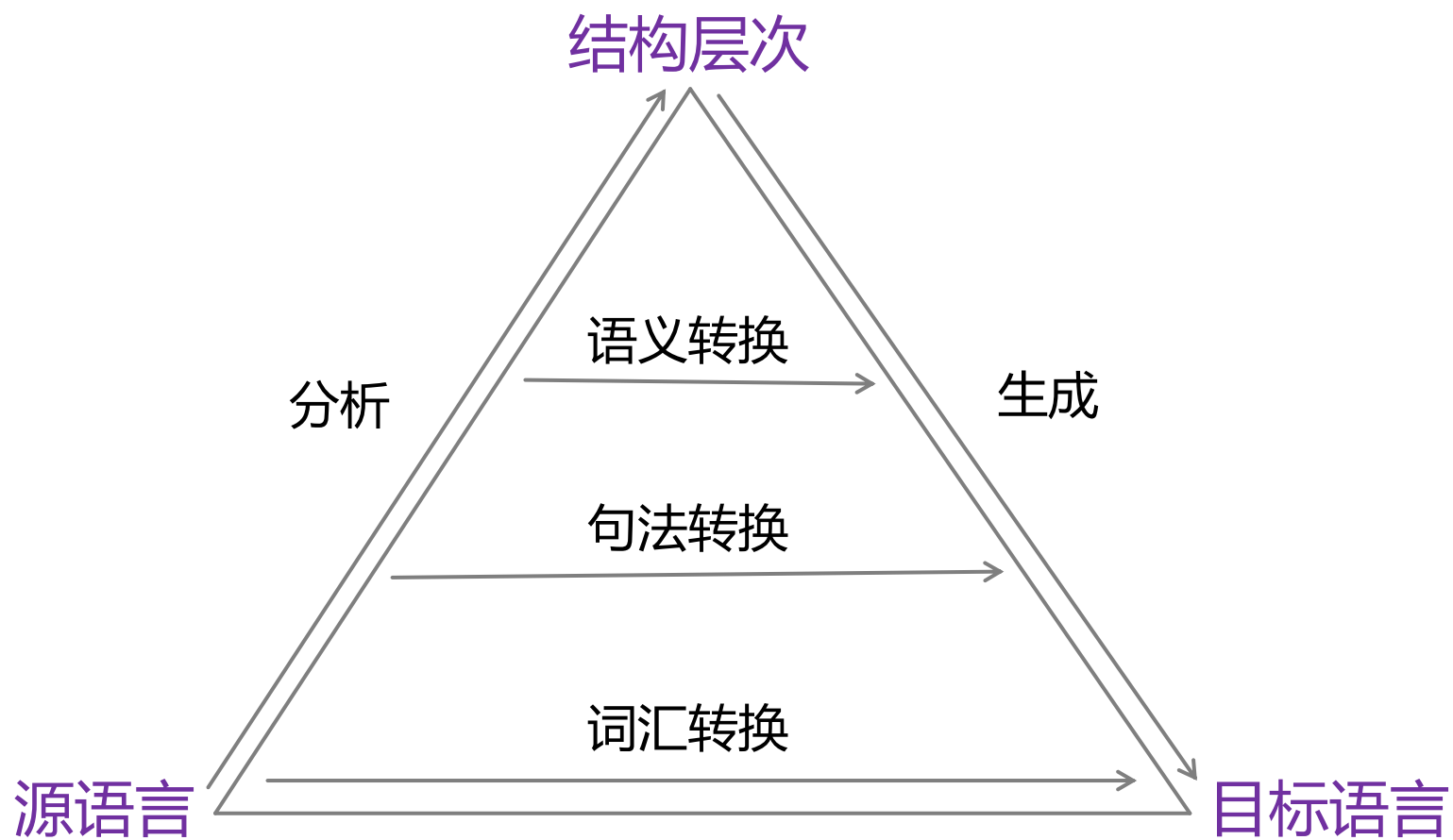
02



基于规则的机器翻译

RULE-BASED MACHINE TRANSLATION

- 分析
 - 将源语言句子解析成一种深层的结构表示
- 转换
 - 将源语言句子的深层结构表示转换成目标语言的深层结构表示
- 生成
 - 根据目标语言的深层结构表示生成对应的目标语言句子



- 词汇转换规则

- 他 → He , 在 → in , 北京 → Beijing , 工作 → works

源语言

他

在

北京

工作

目标语言



He



in



Beijing



works



- 分析规则

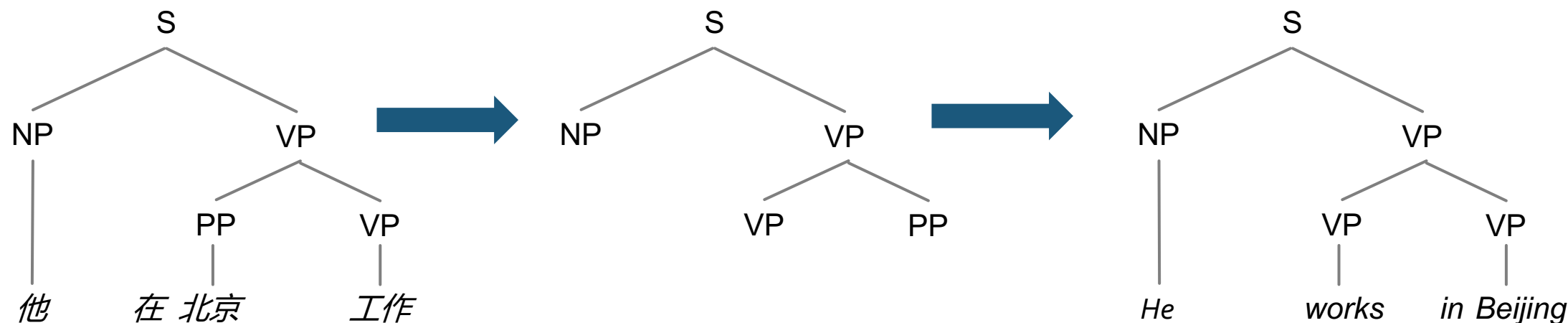
- NP \rightarrow 他, PP \rightarrow 在北京, VP \rightarrow 工作, VP \rightarrow PP VP, S \rightarrow NP VP

- 句法转换规则

- S (NP VP) \rightarrow S (NP VP), VP (PP VP) \rightarrow VP (VP PP)

- 词汇转换规则

- 他 \rightarrow He, 在 \rightarrow in, 北京 \rightarrow Beijing, 工作 \rightarrow works



- 南京大学日汉规则翻译系统
 - 国家七五科技攻关计划、863计划
 - 基于短语结构文法和格语法的日语句法、语义分析与转换，一体化的汉语生成技术等
 - 6万余词基本词典，10万余词领域词典，10万余人名地名词典等；
 - 句法规则1000余条，动词格框架1800余条，通用格框架60余条，转换/生成规则大约有800余条，40万句对翻译记忆体

国家七五科技攻关重大成果奖

江苏省科技进步三等奖

南京大学日汉机器翻译系统:

<http://nlp.nju.edu.cn/homepage/introduction.html>

- 规则质量依赖于语言学家的知识和经验，获取成本高
- 大规模规则系统维护难度大
- 规则之间容易发生冲突



"Anytime a linguist leaves the group the recognition rate goes up."

- Frederick Jelinek, 1988

Frederick Jelinek (1932-2010)

Researcher in Information Theory, Speech Recognition, and Natural Language Processing

Professor at Cornell 1962-1974

Head of Continuous Speech Recognition group, IBM T.J. Watson 1972-1993

Head of Center for Language and Speech Processing, JHU 1993-2010



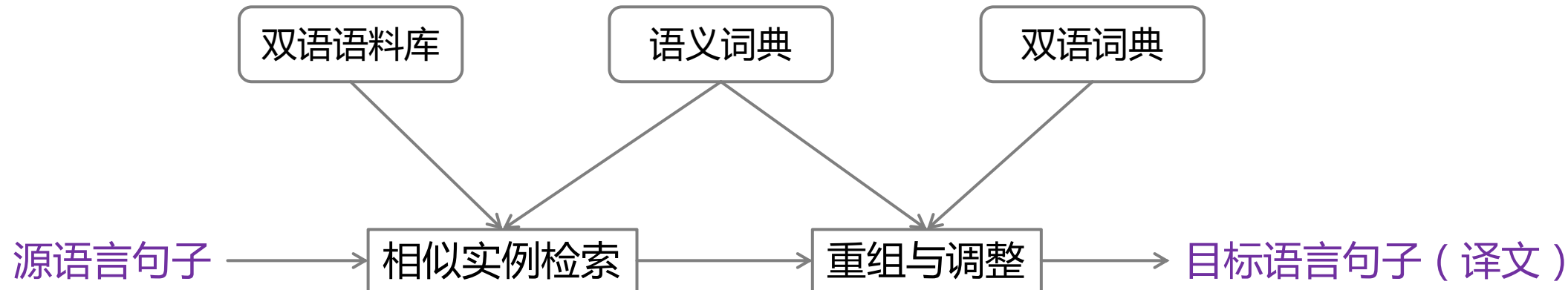
03



基于实例的机器翻译

ANALOGY-BASED MACHINE TRANSLATION

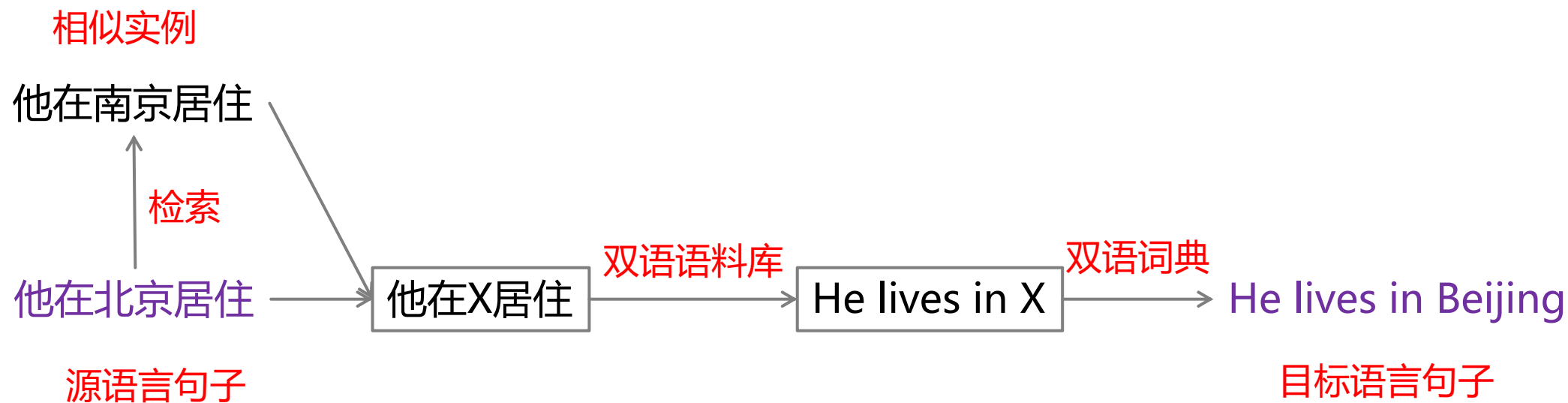
- 从双语语料库中学习翻译实例 [Nagao, 1984]
 - 利用类比思想（ analogy ） ，避免复杂的结构分析
 - 从语料库中查找与待翻译句子相近的实例 ，通过逐词替换进行翻译



翻译样例

- 双语语料库

中文	英文
他喜欢北京	He likes Beijing
他在南京居住	He lives in Nanjing
.....



- 检索的实例粒度一般为句子，无法支持较长文本的翻译
- 实例相似度较低时翻译欠佳



04



统计机器翻译

STATISTICAL MACHINE TRANSLATION

- 将源语言的句子x翻译成目标语言句子y（译文）

源语言

我来自中国



目标语言

I come from China

$$\operatorname{argmax}_y P(y|x)$$

噪声信道模型 (NOISY CHANNEL)

- 描述源语言的句子x是如何由目标语言句子y生成的 (生成式模型)
 - 原始信号y经过“噪声”扰动，变成了观测信号x
 - 根据观测信号x，推测出原始信号y

贝叶斯公式

$$\begin{aligned}\operatorname{argmax}_y P(y|x) &= \operatorname{argmax}_y \frac{P(y)P(x|y)}{P(x)} \\ &= \operatorname{argmax}_y P(y)P(x|y)\end{aligned}$$

↑ ↑

源模型 信道模型

语言模型 翻译模型

流利度 忠实度

- 1. 如何估计语言模型的概率 $P(y)$?
- 2. 如何估计翻译模型概率 $P(x|y)$?
- 3. 如何快速解码目标语言 y , 使得 $P(y)P(x|y)$ 最大?

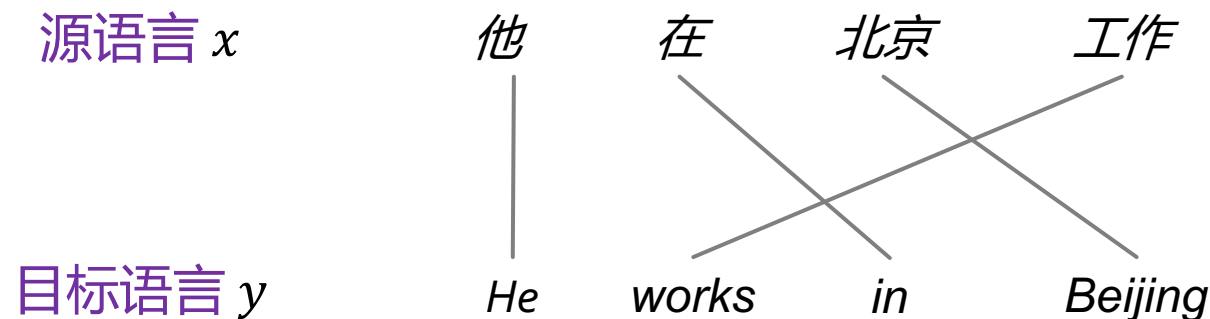
$$\operatorname{argmax}_y P(y|x) = \operatorname{argmax}_y P(y)P(x|y)$$

- 对于由 m 个单词构成的目标语言句子 $y = y_1 \dots y_m$, 其语言模型概率 $P(y)$ 为 :

$$P(y) = P(y_1)P(y_2|y_1) \dots P(y_m|y_1 \dots y_{m-1})$$

可用统计语言模型建模 (ngram模型)

- IBM Model : 引入对齐概念来帮助计算翻译模型



- 假设源语言句子 x 和目标语言句子 y 之间的一种对齐方式为 $a = a_1 \dots a_n$, 其中 $a_n \in [0, 1, \dots, m]$, n 和 m 分别为源语言句子和目标语言句子长度 , 则 :

$$P(x|y) = \sum_a P(x, a|y)$$

$$= P(n|y) \prod_{j=1}^n (P(a_j | a_1 \dots a_{j-1}, x_1 \dots x_{j-1}, y, n) \times P(x_j | a_1 \dots a_{j-1}, x_1 \dots x_{j-1}, y, n))$$

长度生成模型 词语对齐模型 词对翻译模型

- IBM Model1假设：

- $P(n|y) = c$ ，即长度生成概率是一个常量

- $a_j \sim \text{uniform}(0, 1, \dots, m)$ ，即词语对齐模型服从均匀分布，词语对齐概率

$$P(a_j | a_1 \dots a_{j-1}, x_1 \dots x_{j-1}, y, n) = \frac{1}{m+1}$$

- $x_j \sim \text{Categorical}(\theta_{y_{a_j}})$ ，即词对翻译概率服从类别分布，词对翻译概率

$$P(x_j | a_1 \dots a_{j-1}, x_1 \dots x_{j-1}, y, n) = P(x_j | y_{a_j})$$

- 计算 $P(x, a|y)$

$$P(x, a|y) = \frac{c}{(m+1)^n} \prod_{j=1}^n p(x_j|y_{a_j})$$

- 计算 $P(x|y)$

$$P(x|y) = \sum_a P(x, a|y) = \frac{c}{(m+1)^n} \prod_{j=1}^n \sum_{i=0}^m P(x_j|y_i)$$

IBM Model : <https://zhuanlan.zhihu.com/p/154262144>

EM算法 : <https://zhuanlan.zhihu.com/p/78311644>

● Viterbi 算法（动态规划算法）

假设给定隐式马尔可夫模型（HMM）状态空间 S ，共有 k 个状态，初始状态 i 的概率为 π_i ，从状态 i 到状态 j 的转移概率为 $a_{i,j}$ 。令观察到的输出为 y_1, \dots, y_T 。产生观察结果的最有可能的状态序列 x_1, \dots, x_T 由递推关系给出：[2]

$$V_{1,k} = P(y_1 | k) \cdot \pi_k$$

$$V_{t,k} = \max_{x \in S} (P(y_t | k) \cdot a_{x,k} \cdot V_{t-1,x})$$

此处 $V_{t,k}$ 是前 t 个最终状态为 k 的观测结果最有可能对应的状态序列的概率。通过保存向后指针记住在第二个等式中用到的状态 x 可以获得维特比路径。声明一个函数 $\text{Ptr}(k, t)$ ，它返回若 $t > 1$ 时计算 $V_{t,k}$ 用到的 x 值 或若 $t = 1$ 时的 k 。这样：

$$x_T = \arg \max_{x \in S} (V_{T,x})$$

$$x_{t-1} = \text{Ptr}(x_t, t)$$

这里我们使用了 $\arg \max$ （英语：arg max）的标准定义 算法复杂度为 $O(T \times |S|^2)$

Viterbi算法：https://en.wikipedia.org/wiki/Viterbi_algorithm



05

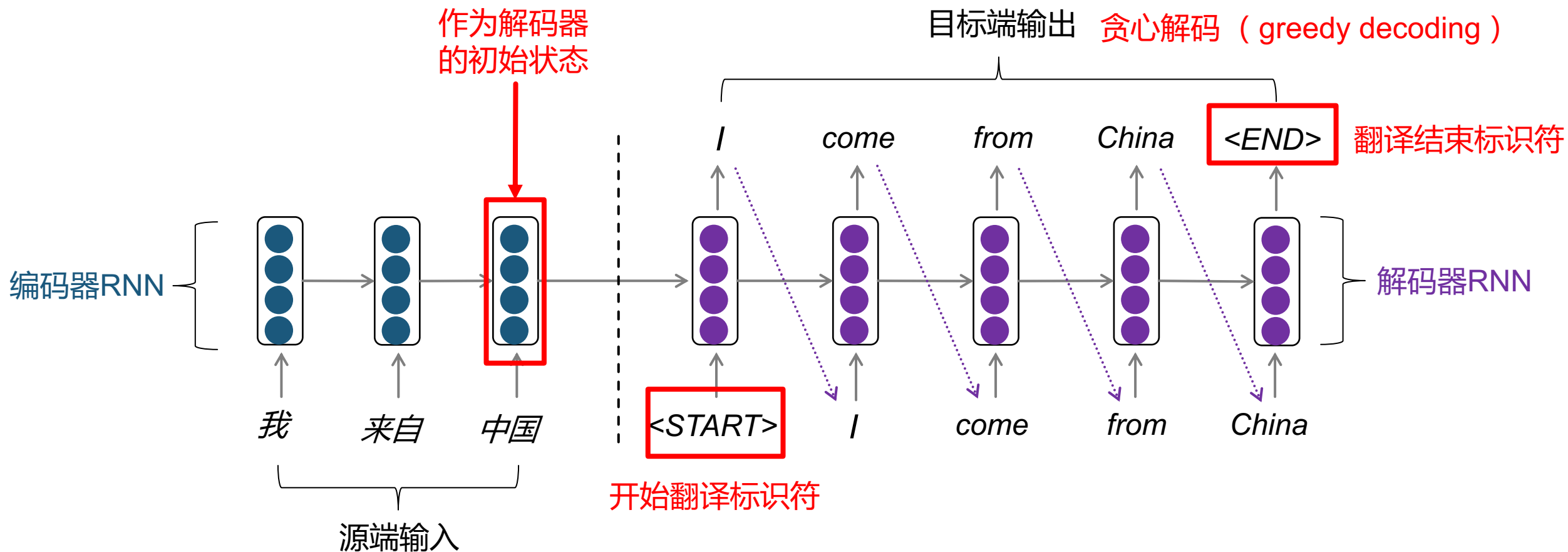


神经机器翻译

NEURAL MACHINE TRANSLATION

- 神经机器翻译（Neural Machine Translation）：用端到端（end-to-end）的神经网络来建模机器翻译任务。
- 编码器-解码器框架（Encoder-decoder）
 - Seq2Seq
 - 编码器（encoder）：用来编码源语言的输入
 - 解码器（decoder）：用来生成目标语言的输出

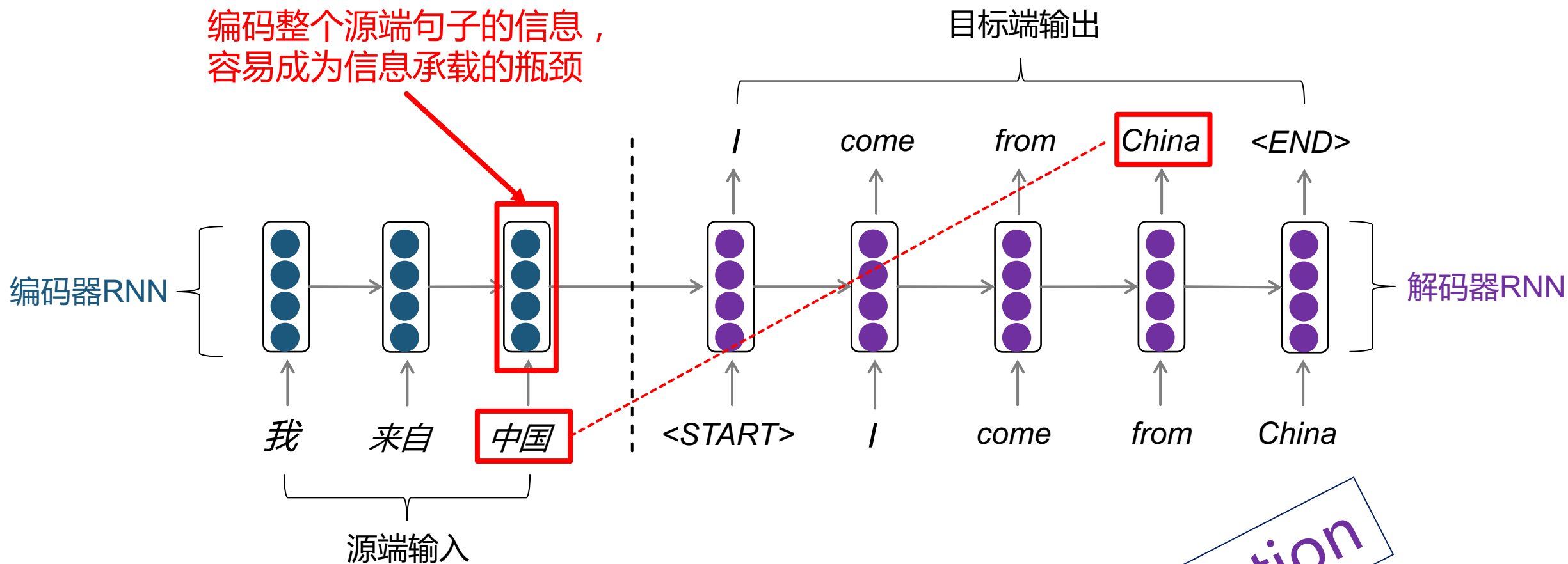
编码器-解码器框架



编码器RNN编码了源端句子的上下文信息

解码器RNN是一个给定输入编码条件下的语言模型

编码器-解码器框架的问题

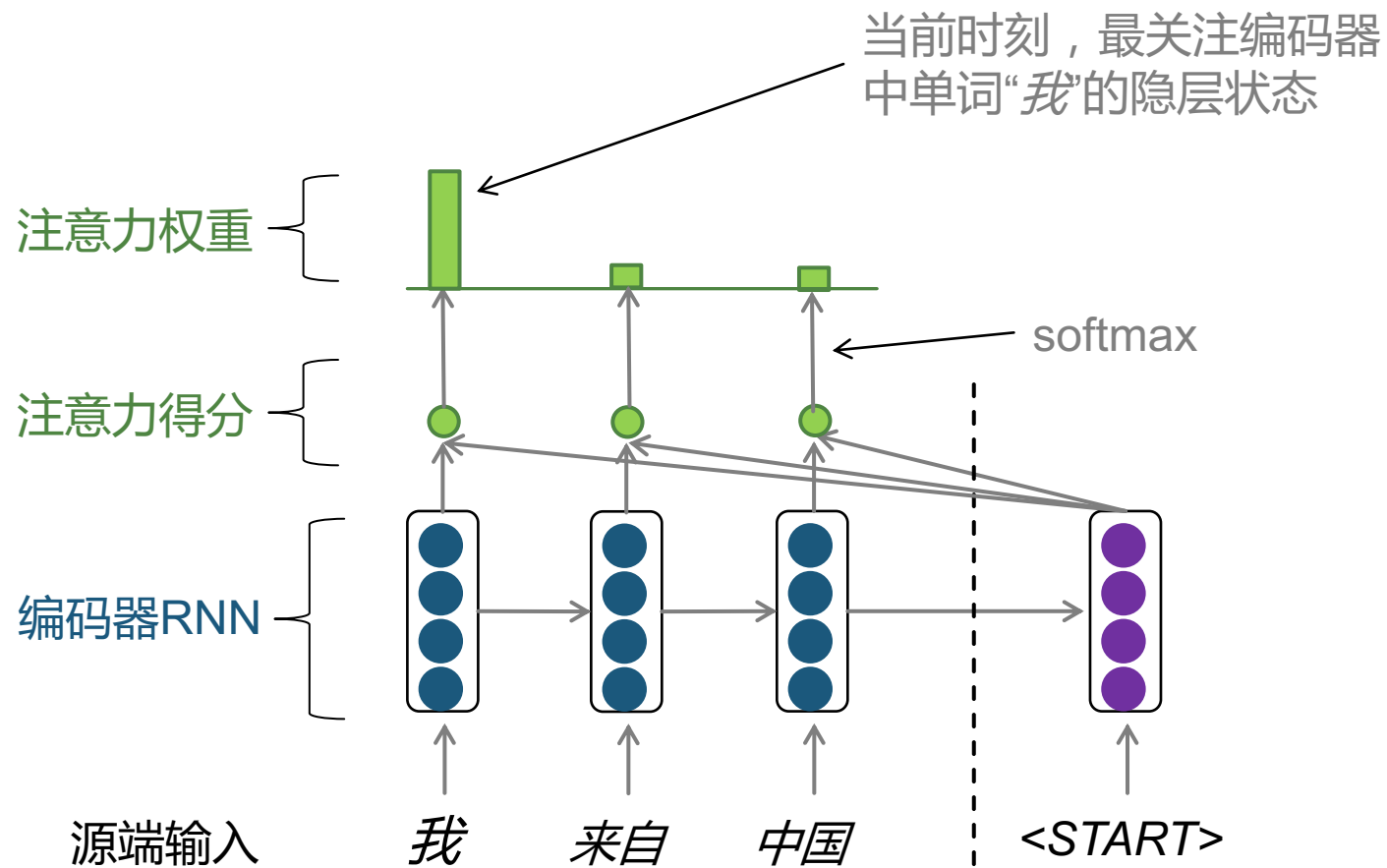


翻译过程不具有可解释性

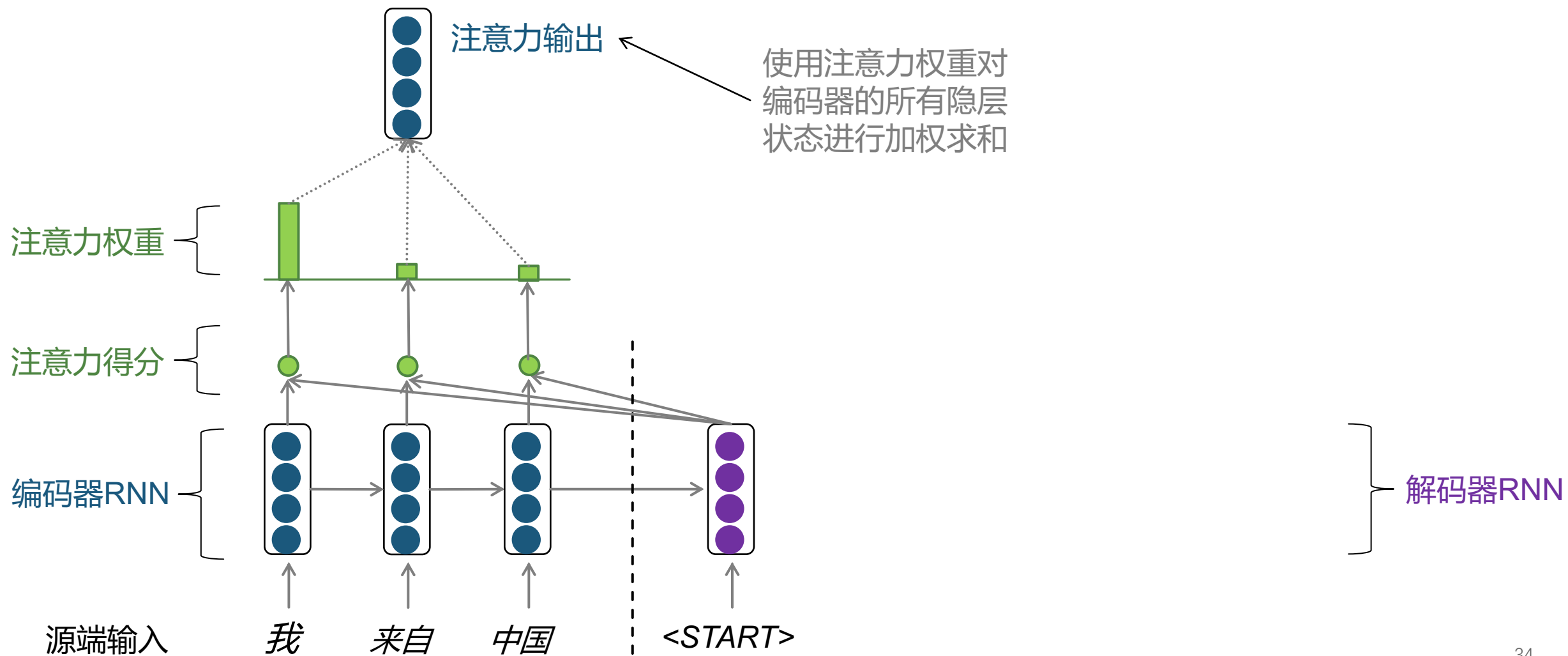
目标端解码某个特定单词时，应
该重点关注源端相关的单词

Attention

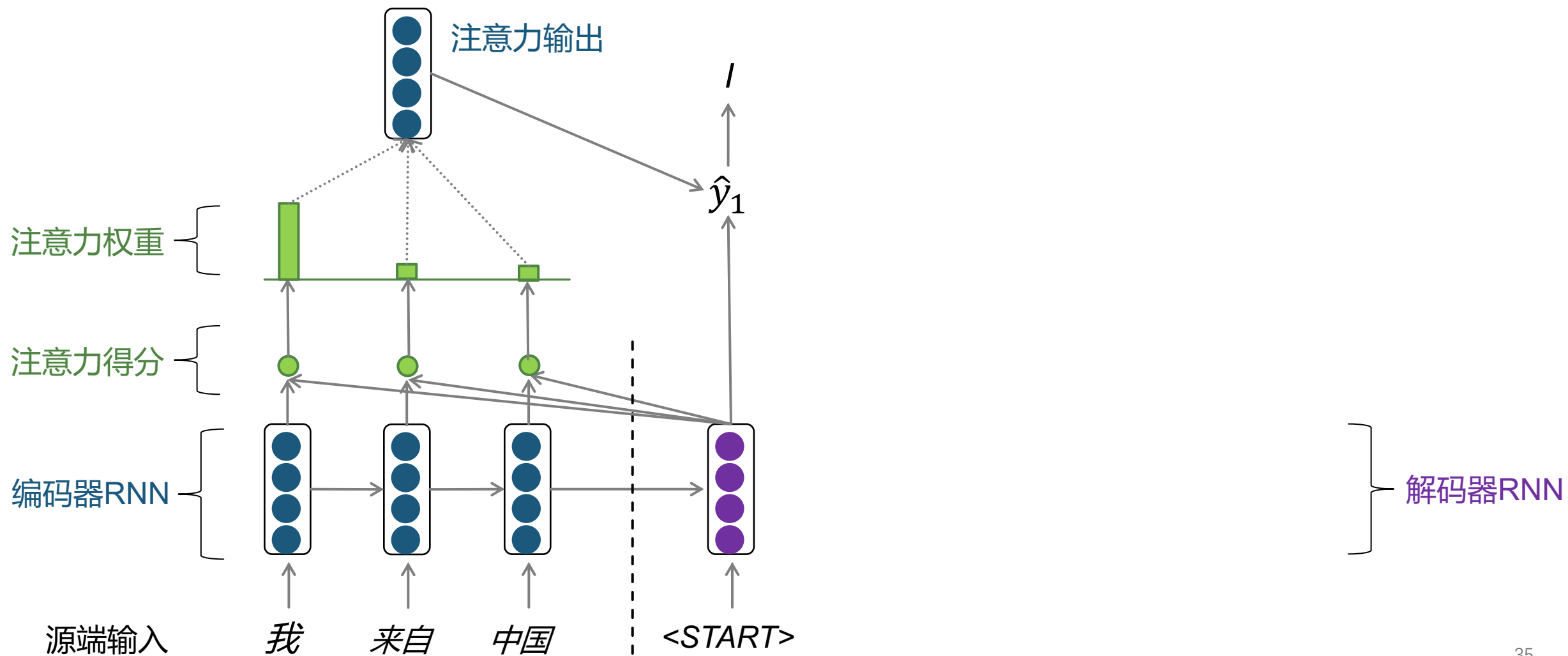
基于注意力机制的编码器-解码器框架



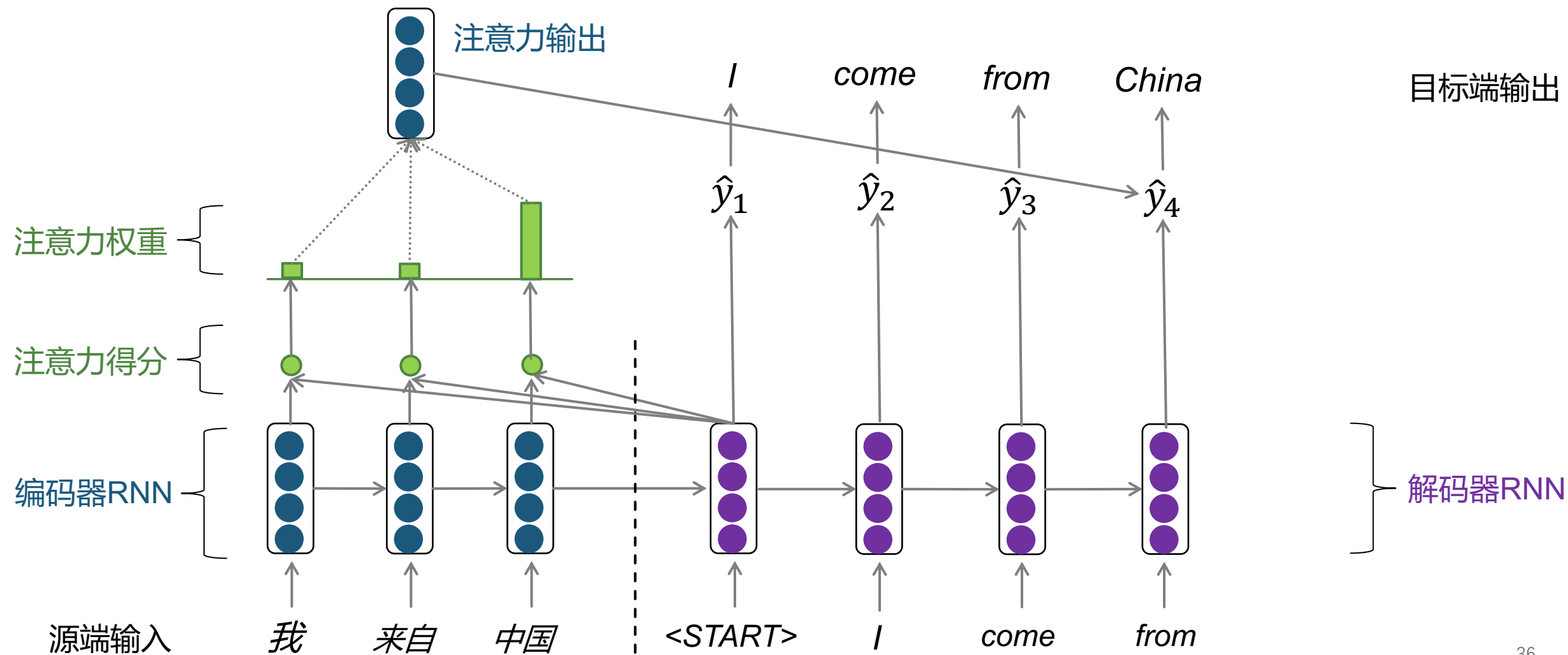
基于注意力机制的编码器-解码器框架



基于注意力机制的编码器-解码器框架

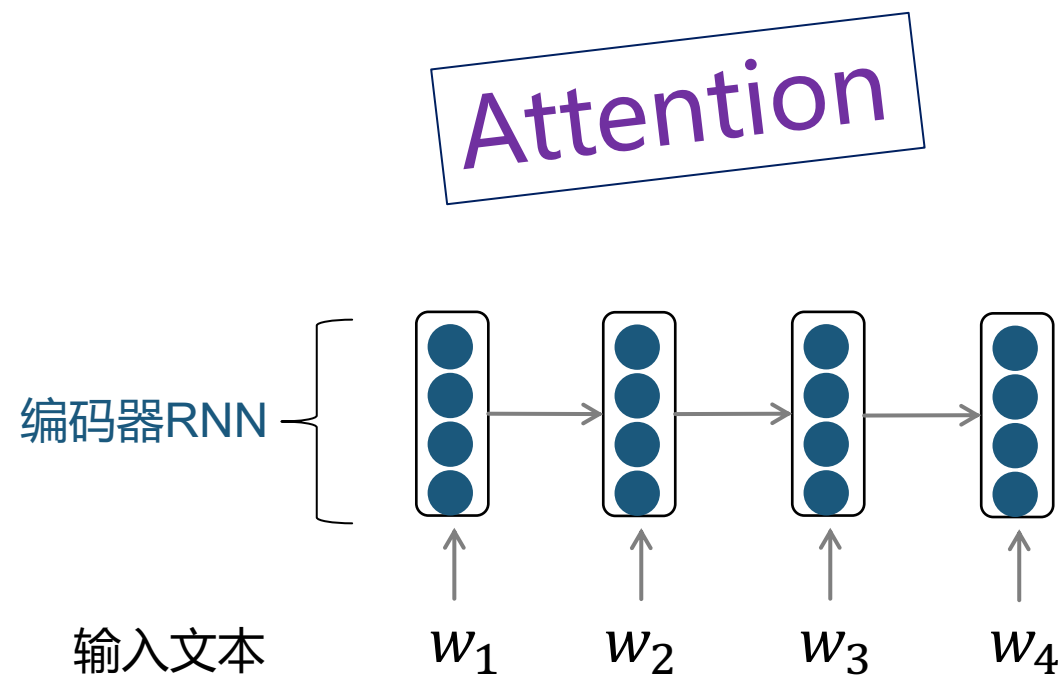


基于注意力机制的编码器-解码器框架



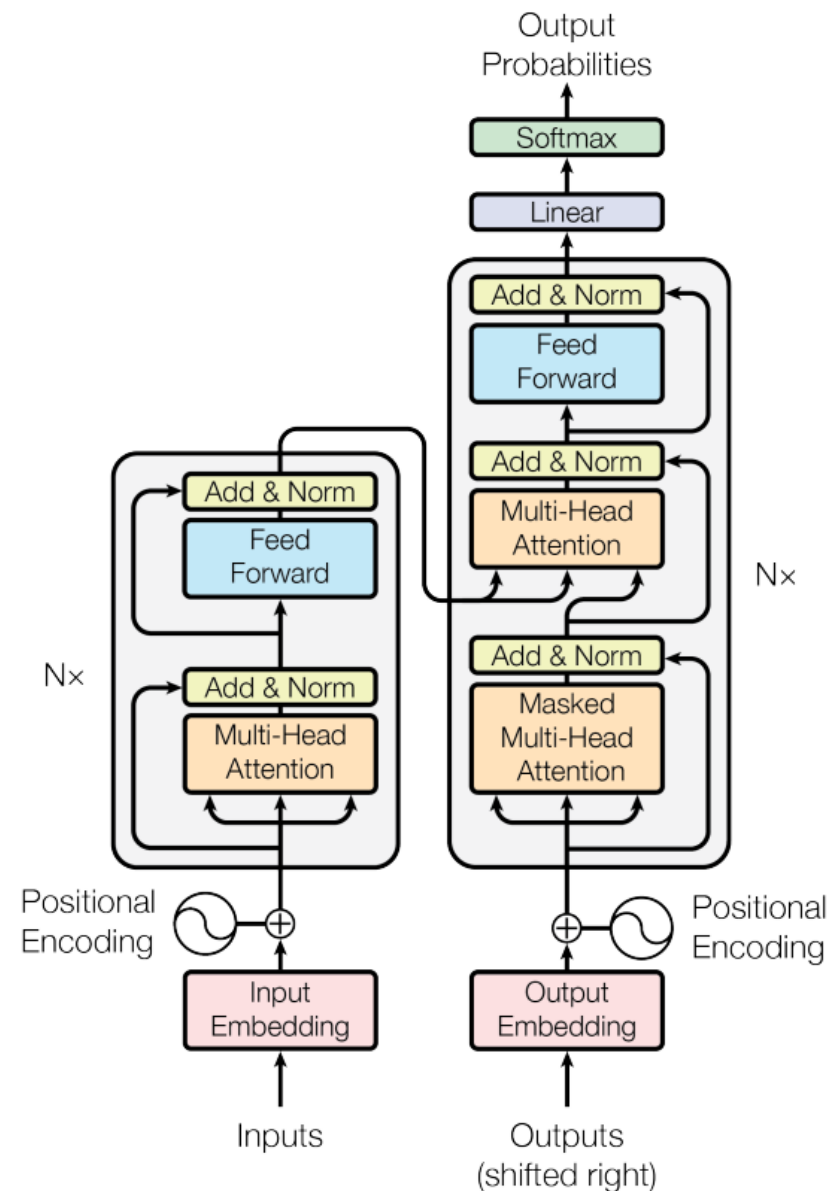
基于RNN的编码器-解码器问题

- 有限的信息交互距离
 - RNN能捕捉局部信息，但无法很好地解决长距离依赖关系（long-distance dependency）
 - 不能很好地建模序列中的非线性结构关系
- 无法并行
 - RNN的隐层状态具有序列依赖性
 - 时间消耗随序列长度的增加而增加



ATTENTION IS ALL YOU NEED

- Transformer [Vaswani et al., 2017]
 - 完全基于attention机制
 - 建模输入序列的全局依赖关系、并行计算
 - 一个基于attention机制的编码器
 - 一个基于attention机制的解码器



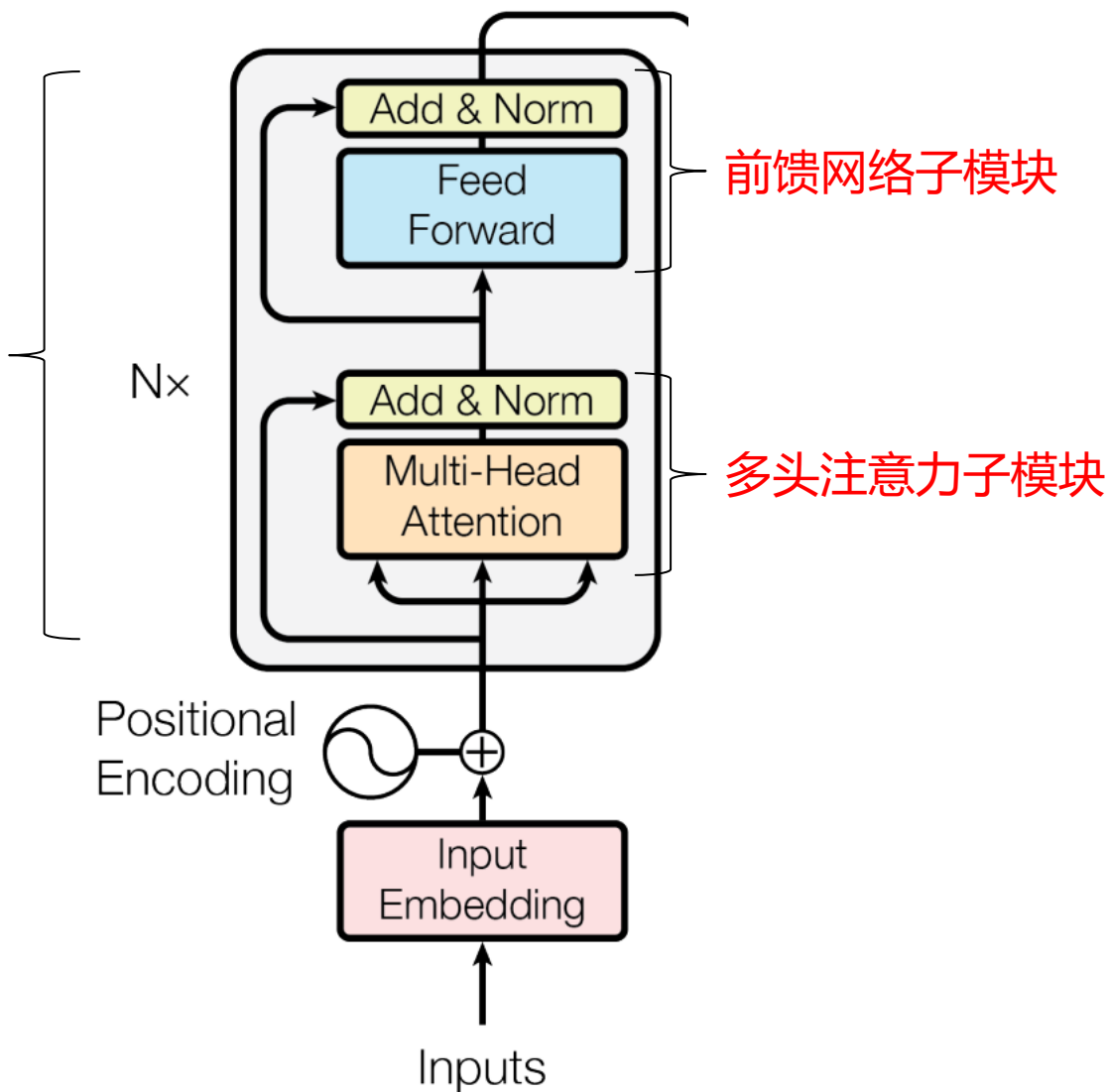
TRANSFORMER编码器



- 编码器

- 输入编码+位置编码
- 多头注意力机制
- 残差连接&层正则
- 前馈神经网络
- 残差连接&层正则

N个相同的
层堆叠而成



TRANSFORMER解码器

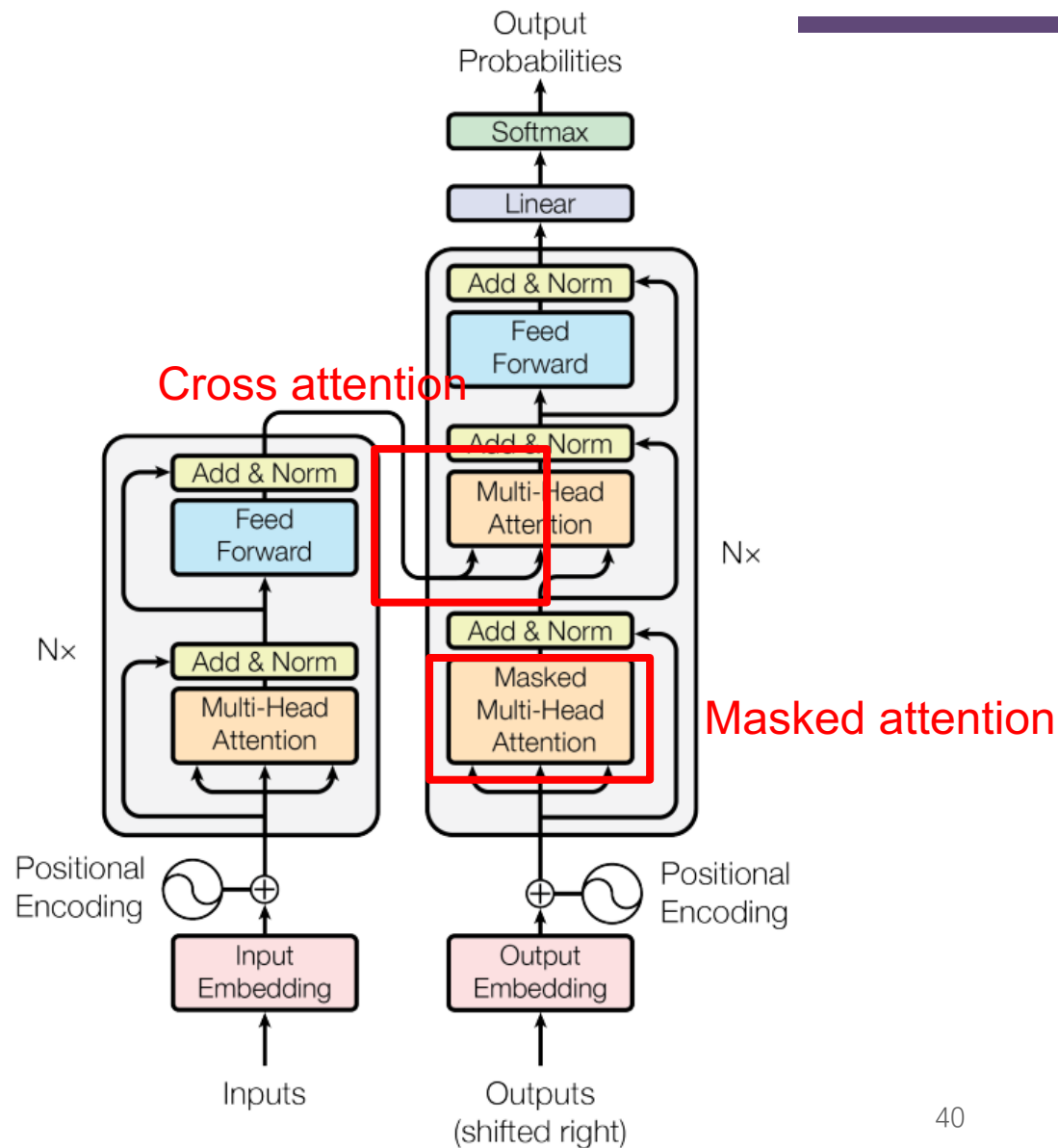


- Cross Attention
 - 解码时需要关注源端信息
- Masked Attention
 - 解码时（训练）不应该看到未来的信息

关注的词（负无穷不关注）

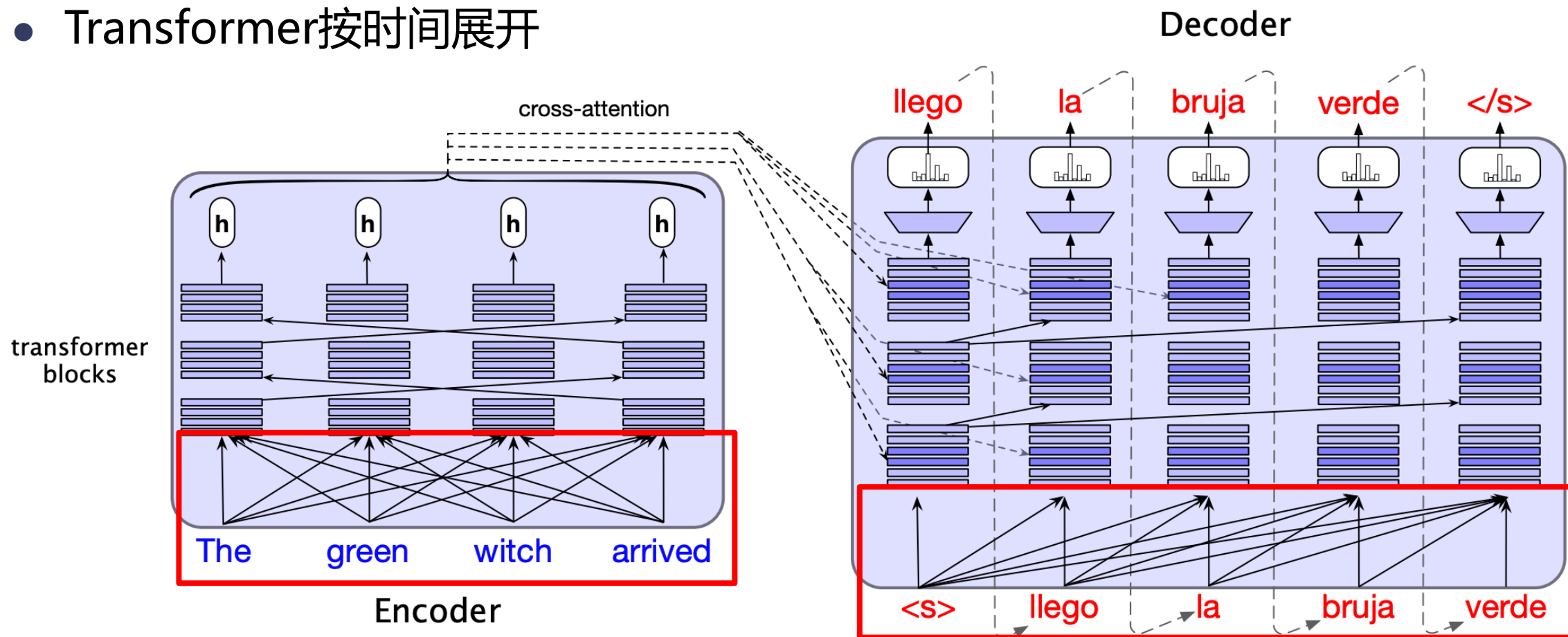
解码端输入的词

	[START]	The	chef	who
[START]	$-\infty$	$-\infty$	$-\infty$	$-\infty$
The		$-\infty$	$-\infty$	$-\infty$
chef			$-\infty$	$-\infty$
who				$-\infty$



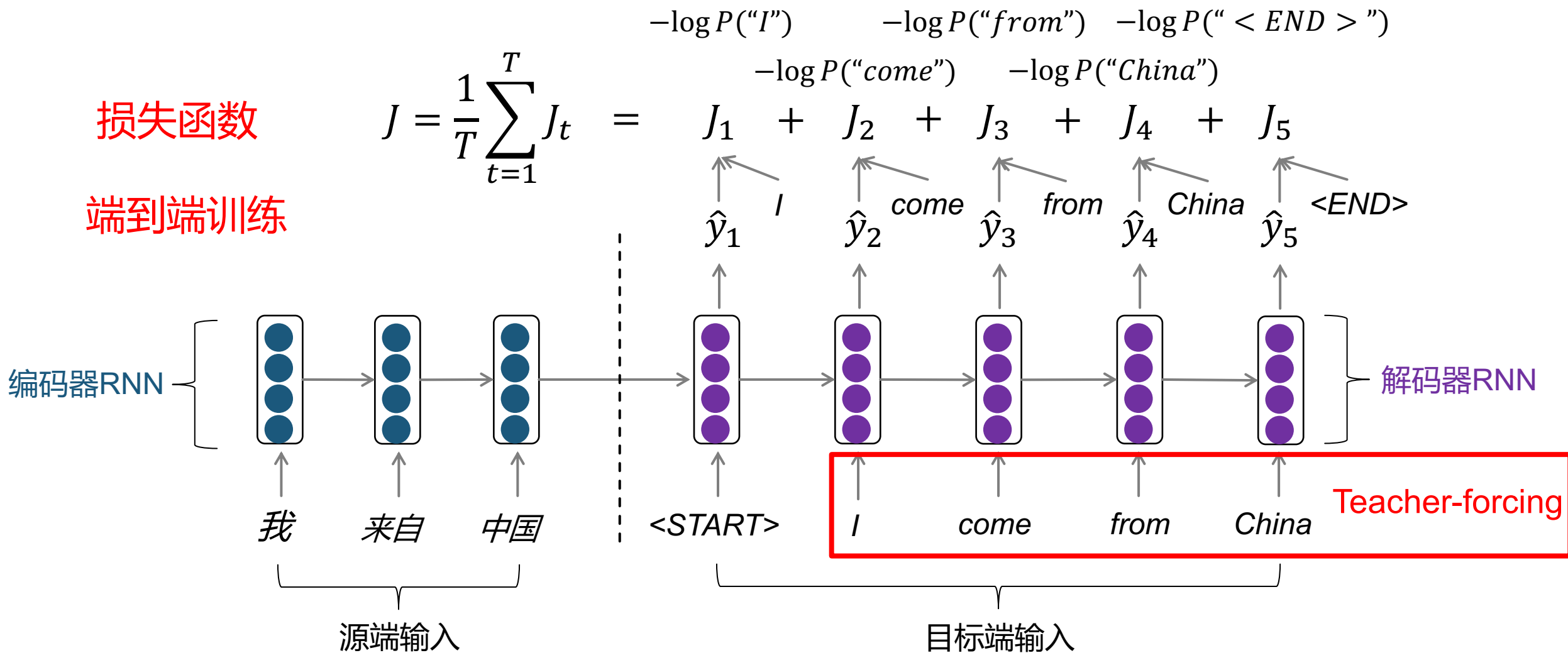
TRANSFORMER翻译示例

- Transformer按时间展开



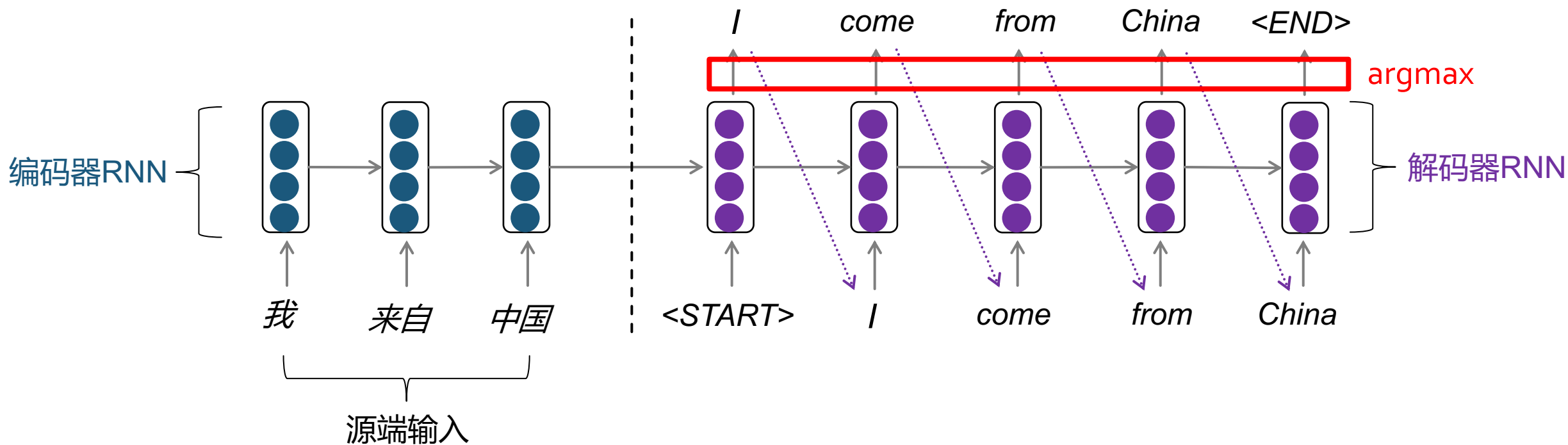
损失函数

端到端训练

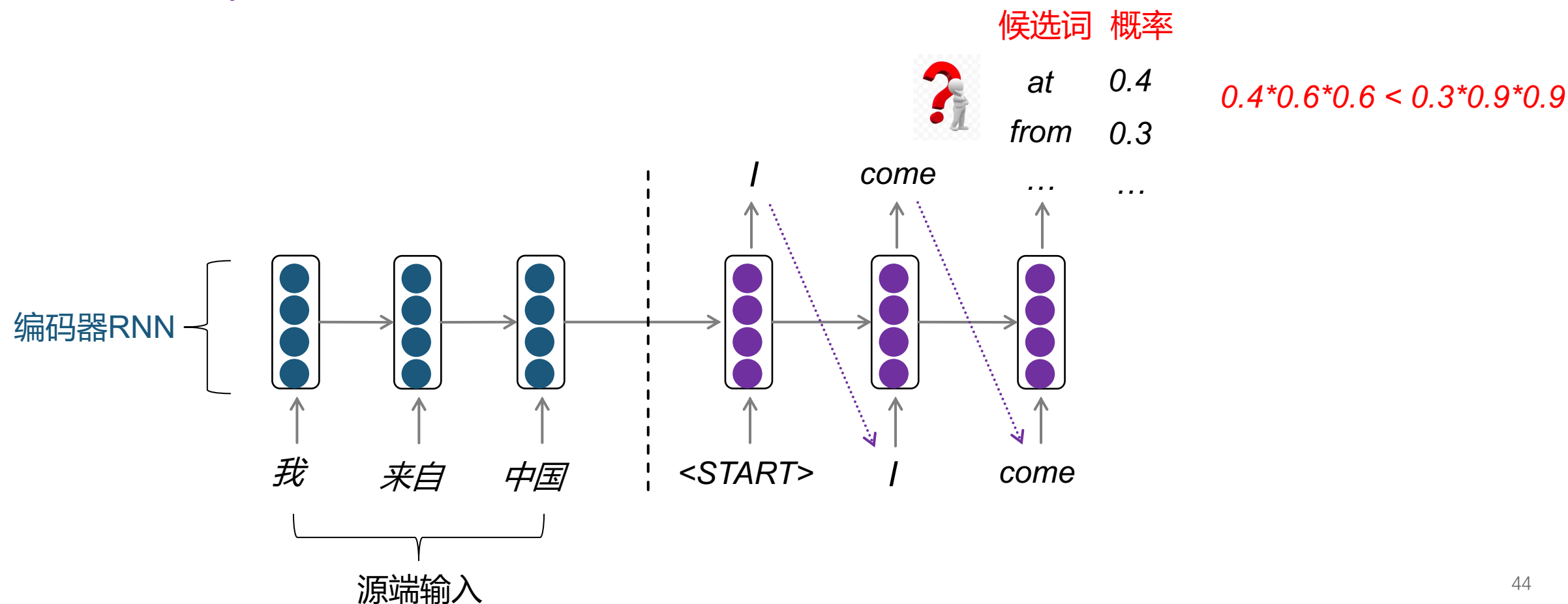


贪心解码 (GREEDY SEARCH)

- 解码时，每个时刻从解码器取出概率最大的词 (argmax) 作为预测结果



- 某一时刻的**错误翻译**会影响后续所有翻译，因此不一定能解码出**全局最佳**（**译文概率最大**）的译文



- 翻译目标：对于给定的源语言 x ，解码出对应的目标语言 $y = \operatorname{argmax}_y P(y|x)$

$$\begin{aligned} P(y|x) &= P(y_1|x)P(y_2|y_1, x)P(y_3|y_1, y_2, x) \dots P(y_T|y_1, \dots, y_{T-1}, x) \\ &= \prod_{t=1}^T P(y_t|y_1, \dots, y_{t-1}, x) \end{aligned}$$

- 解码时枚举所有的翻译结果 y
 - 复杂度 $O(V^T)$ ， V 为目标语言的词表大小， T 为生成译文长度
 - 极其耗时

柱搜索解码 (BEAM SEARCH)

- 在贪心搜索基础上扩大搜索空间，从而更有可能解码出全局最优的译文
- 核心思想：在t时刻，保留k个概率最大的翻译结果（k为beam size）
 - 相比于枚举，极大提高搜索效率
 - 并不能保证一定解码出最优的译文
- 打分函数：

$$score(y_1, \dots, y_t) = \log P(y_1, \dots, y_t | x) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$

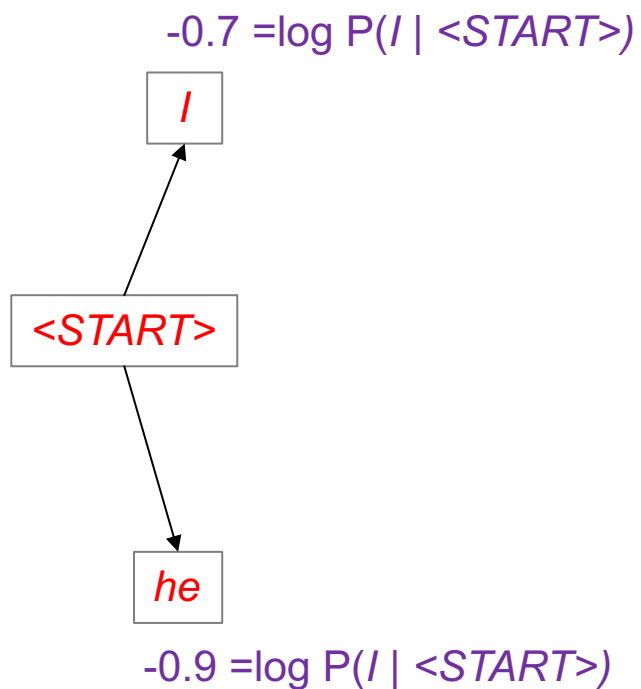
越大越好

柱搜索解码样例



Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$



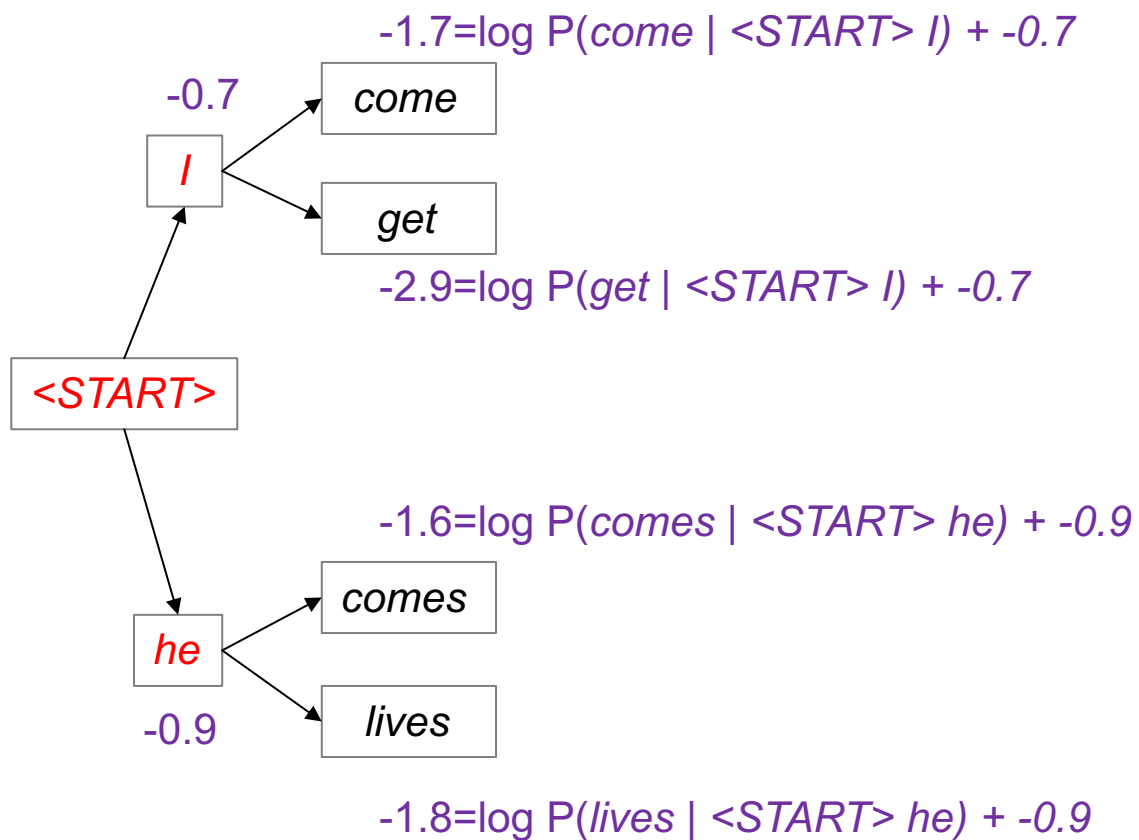
计算打分保留打分最高的2个词

柱搜索解码样例



Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$

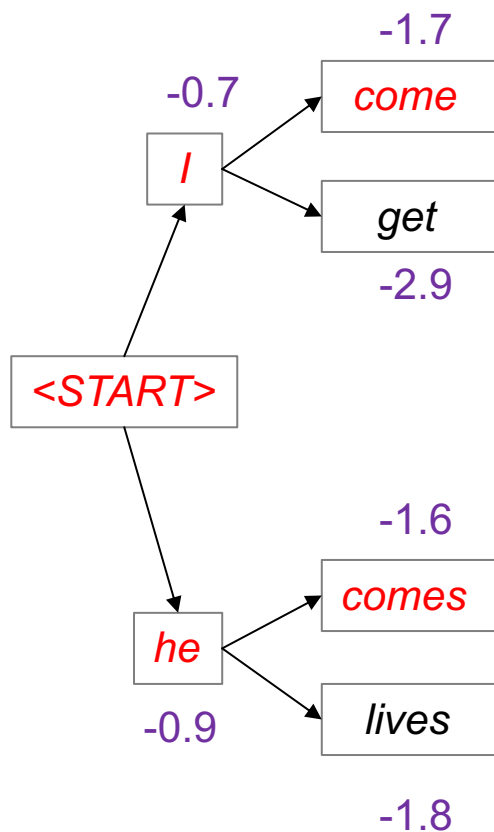


柱搜索解码样例



Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$

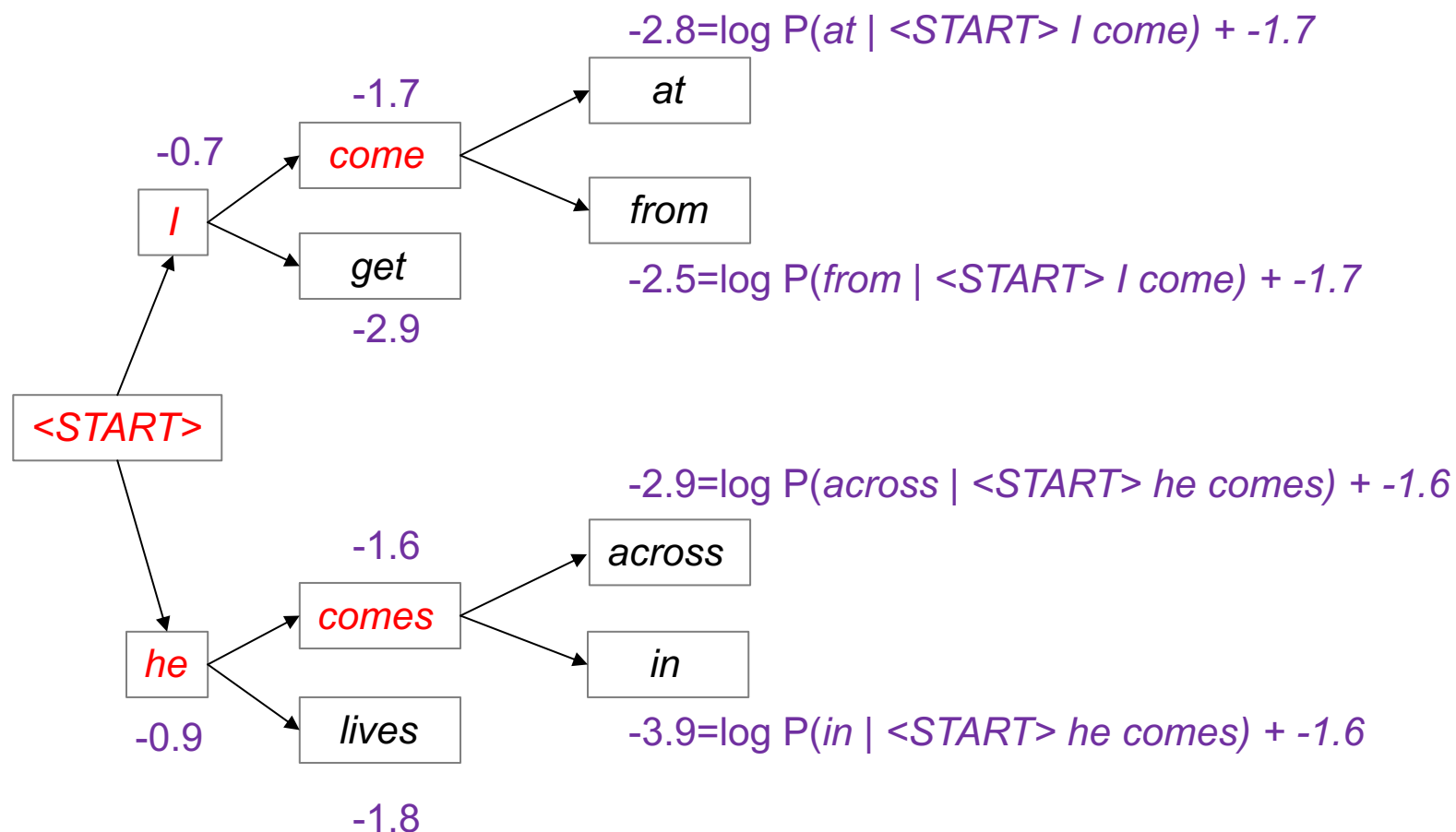


柱搜索解码样例



Beam size=2

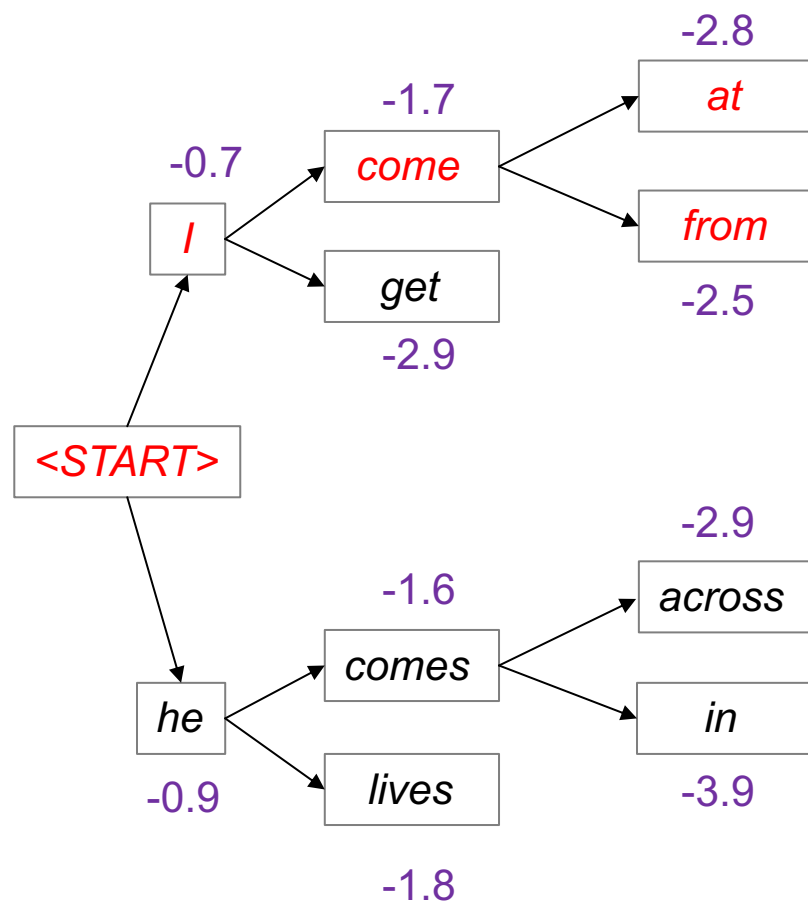
$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$



柱搜索解码样例

Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$

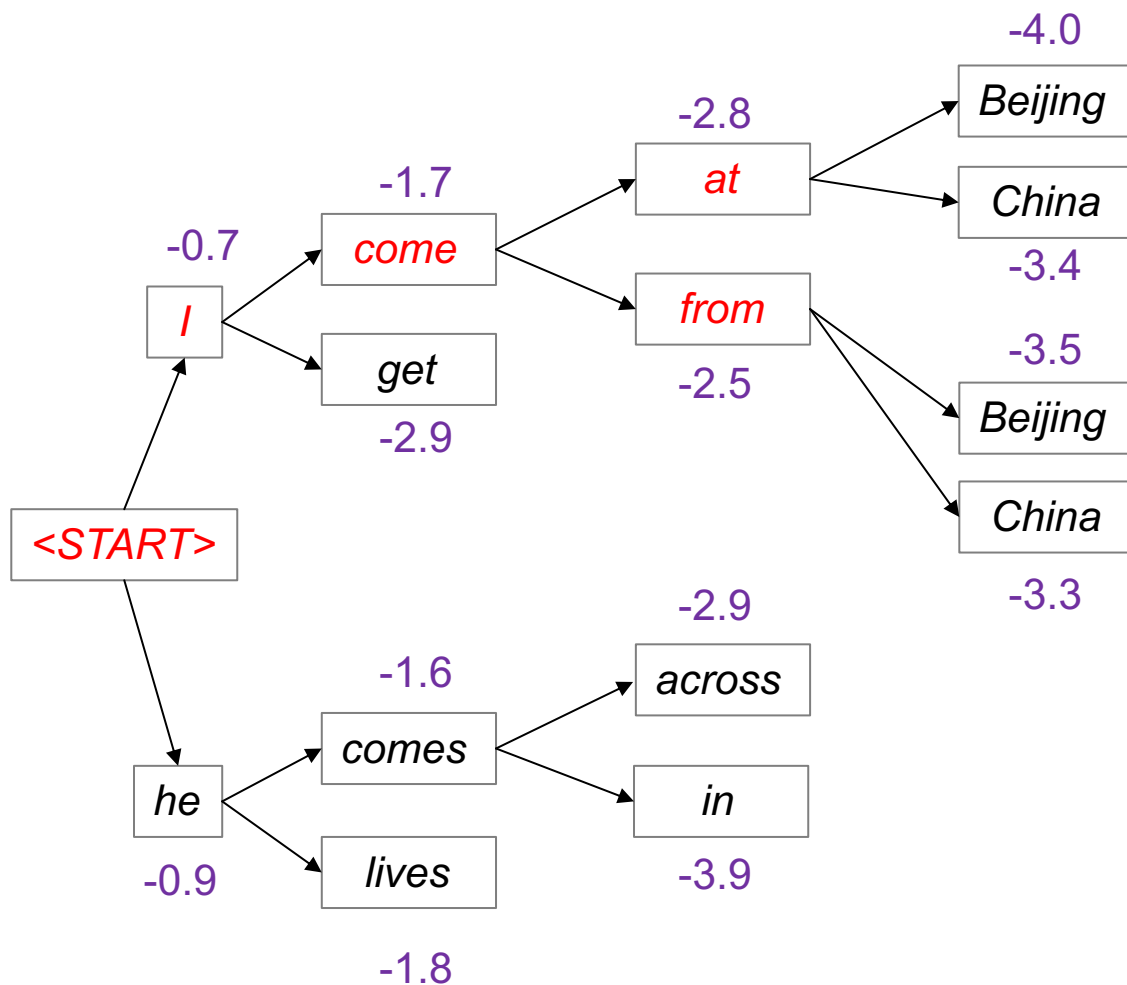


柱搜索解码样例



Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$

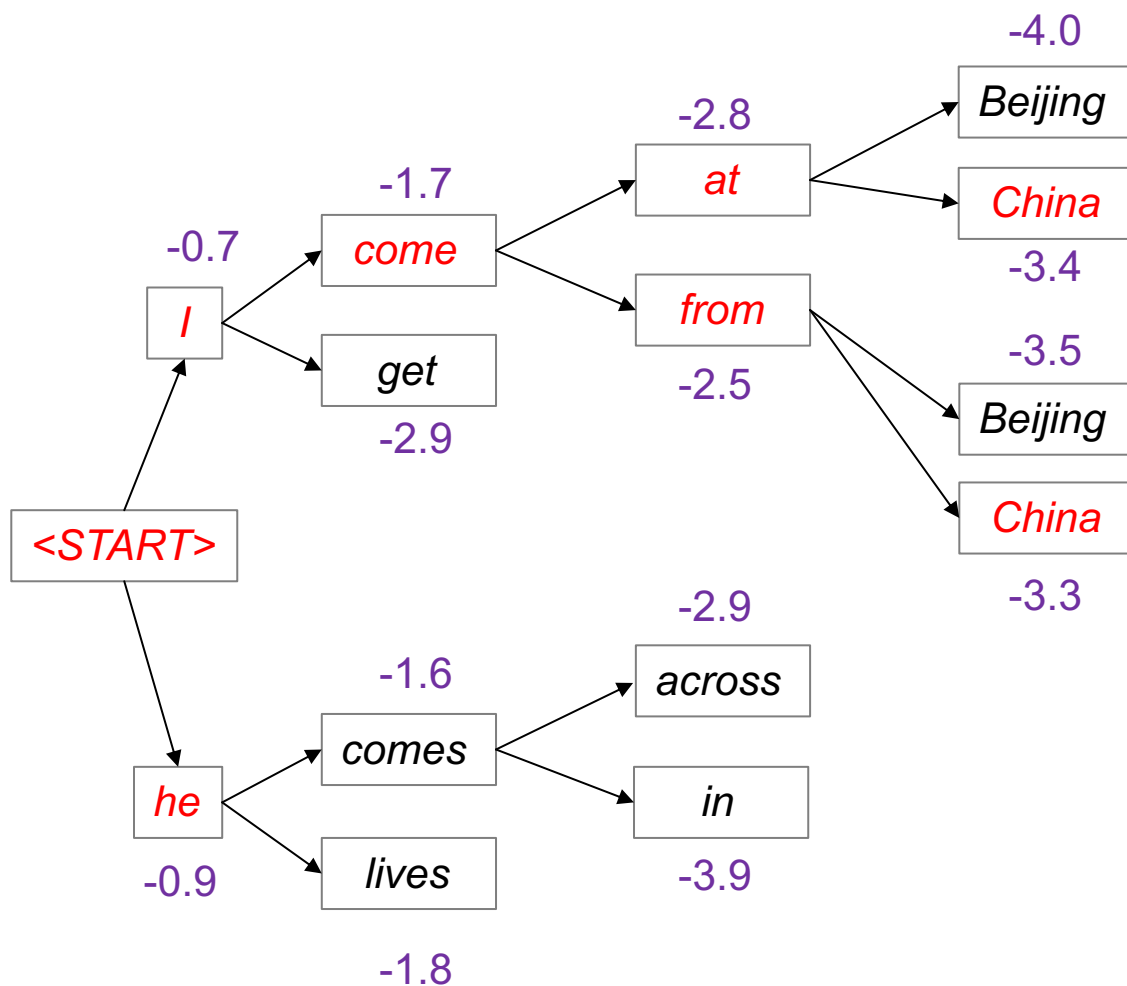


柱搜索解码样例



Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$

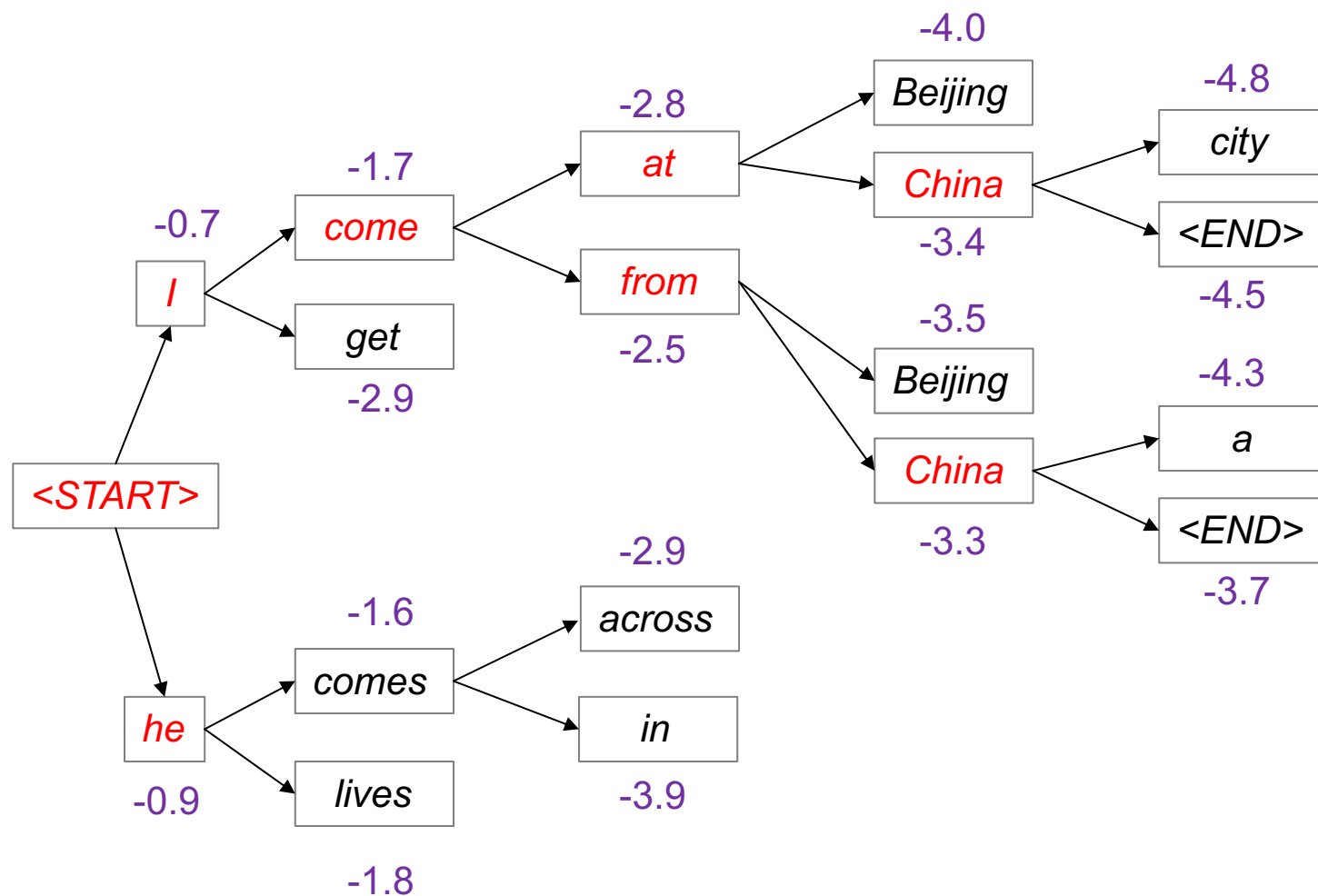


柱搜索解码样例



Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$

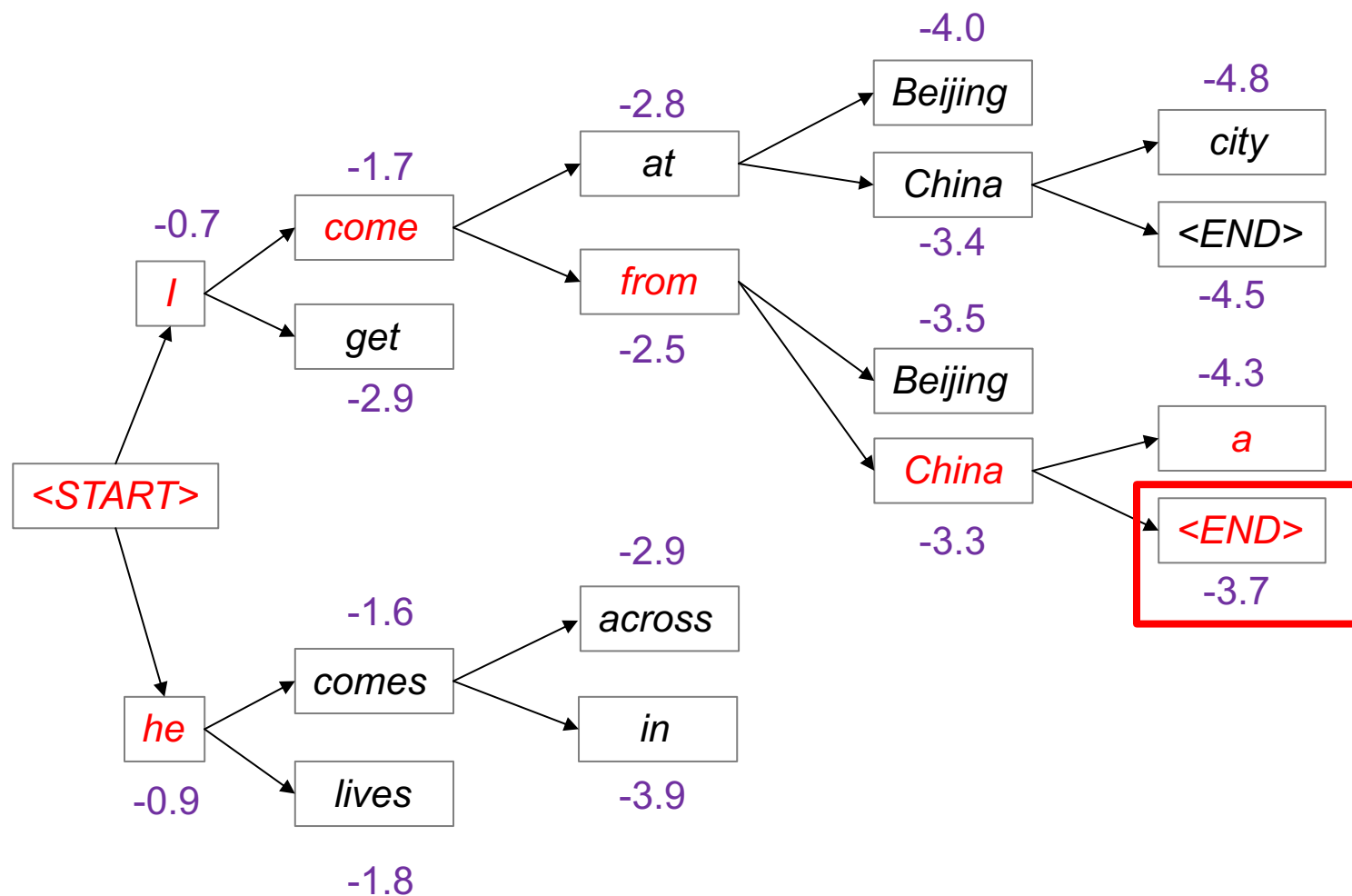


柱搜索解码样例



Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$

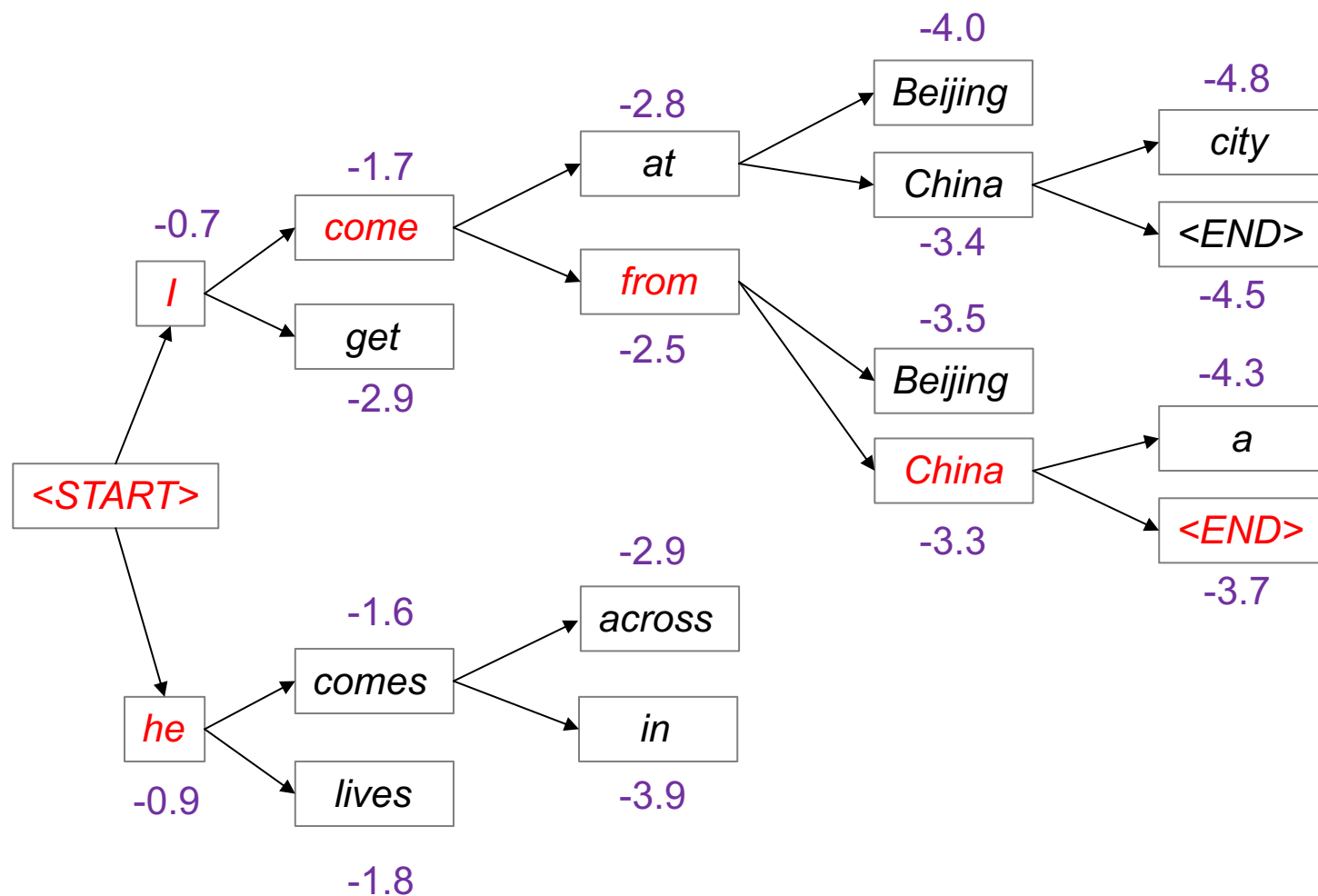


柱搜索解码样例



Beam size=2

$$\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P(y_i | y_1, \dots, y_{i-1}, x)$$



Algorithm 1 Standard beam search⁴

Input: \mathbf{x} : source sentence

k : maximum beam size

n_{max} : maximum hypothesis length \leftarrow 译文最大长度

$\text{score}(\cdot, \cdot)$: scoring function

1: $B_0 \leftarrow \{\langle 0, \text{BOS} \rangle\}$ \leftarrow 译文打分和对应的译文

2: **for** $t \in \{1, \dots, n_{max} - 1\}$:

3: $B \leftarrow \emptyset$

4: **for** $\langle s, \mathbf{y} \rangle \in B_{i-1}$:

5: **if** $\mathbf{y}.\text{last}() = \text{EOS}$:

6: $B.\text{add}(\langle s, \mathbf{y} \rangle) \leftarrow$ 某条译文翻译结束，将译文保存

7: **continue**

8: **for** $y \in \mathcal{V}$:

9: $s \leftarrow \text{score}(\mathbf{x}, \mathbf{y} \circ y)$

10: $B.\text{add}(\langle s, \mathbf{y} \circ y \rangle) \leftarrow$ 计算翻译中的译文打分并暂存

11: $B_i \leftarrow B.\text{top}(k) \leftarrow$ 保留打分最高的k个译文

12: **return** $B.\text{max}()$



06



机器翻译质量评估

MACHINE TRANSLATION EVALUATION

- 评估准则（打分制）
 - 忠实度：译文对原文信息和语义的保留程度（“信”）
 - 流利度：衡量译文是否流畅通顺（“达”）
- 缺点
 - 主观偏差
 - 成本昂贵
 - 效率低

- 核心思想
 - 通过比较机器翻译的译文和参考译文（人工翻译结果）之间的相似程度来衡量翻译结果的好坏
 - 机器译文越接近人工翻译结果，其译文的质量就越好
- BLEU
 - 一种衡量机器翻译质量的自动评估指标
 - 统计机器翻译译文与参考译文中 n 元文法匹配的数目占系统译文中所有 n 元文法总数的比例，即 n 元文法的精确率

- 符号定义

- y^* : 源语言句子 x 对应的机器翻译译文
- (y_1, \dots, y_M) : M 个人工参考译文
- $count_{match}(ngram)$: 某 $ngram$ 片段在 y^* 和 (y_1, \dots, y_M) 中共同出现的最大次数 (将系统译文和参考译文逐个对比并统计共现次数, 将共现最大值作为结果)
- $count(ngram)$: $ngram$ 片段在 y^* 中出现的次数

- 频数统计

源语言句子：I love the Great Wall and
the Imperial Palace

机器翻译译文：我 爱 长城 又 爱 爱 爱 故宫

参考译文1：我 爱 长城 和 故宫

参考译文2：我 爱 长城 也 爱 故宫

$$\text{count}_{\text{match}}(\text{我}) = 1$$

$$\text{count}(\text{我}) = 1$$

$$\text{count}_{\text{match}}(\text{爱}) = 2$$

$$\text{count}(\text{爱}) = 4$$

$$\text{count}_{\text{match}}(\text{长城}) = 1$$

$$\text{count}(\text{长城}) = 1$$

$$\text{count}_{\text{match}}(\text{又}) = 0$$

$$\text{count}(\text{又}) = 1$$

$$\text{count}_{\text{match}}(\text{故宫}) = 1$$

$$\text{count}(\text{故宫}) = 1$$

- 计算所有 $ngram$ 的精确率 p_n
 - n 为 $ngram$ 长度，最大值一般取4（ n 增大时， $ngram$ 共现次数呈指数级下降）

$$p_n = \frac{\sum_{y^*} \sum_{ngram \in y^*} count_{match}(ngram)}{\sum_{y^*} \sum_{ngram \in y^*} count(ngram)}$$

- 当 n 最大值取4时， p_1 、 p_2 、 p_3 、 p_4 的加权精确率为：

$$\prod_{n=1}^4 p_n^{w_n} \quad \text{通常 } w_n = \frac{1}{4}$$

- 精确率 p_n 的问题
 - p_n 的计算公式中分母为机器翻译译文，译文越短， p_n 的值倾向更大
 - 具有偏向性，对于漏翻的词不敏感

机器翻译译文：我 爱 长城 也  $p_1 = 1$, $p_2 = 1$, $p_3 = 1$, $p_4 = 1$

- 引入长度惩罚因子 BP ，对过短的译文进行惩罚
 - c ：测试语料中每个源语言句子对应的系统译文 y^* 的长度
 - r ：测试语料中每个源语言句子对应的多个参考译文 (y_1, \dots, y_n) 中最短译文或者与 y^* 长度最接近的参考译文的长度

$$BP = \begin{cases} 1 & , c > r \\ e^{1-\frac{r}{c}} & , c \leq r \end{cases}$$

- 综合 $ngram$ 匹配精确率和长度惩罚因子，BLEU评分公式为：

$$BLEU = BP \times \prod_{n=1}^4 p_n^{w_n} = BP \times \exp\left(\sum_{n=1}^4 w_n \log p_n\right)$$

- 优点：
 - 自动评估
 - 能够从ngram角度评估翻译质量
- 缺点
 - 无法从语义层面度量翻译质量

机器翻译挑战



南京大学
NANJING UNIVERSITY



- Nagao M. A framework of a mechanical translation between Japanese and English by analogy principle[M]. na, 1984.
- Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. arXiv preprint arXiv:1409.0473, 2014.
- Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.
- He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.
- Ba J L, Kiros J R, Hinton G E. Layer normalization[J]. arXiv preprint arXiv:1607.06450, 2016.
- Ilic S, Marrese-Taylor E, Balazs J, et al. Deep contextualized word representations for detecting sarcasm and irony[C]. Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. 2018: 2-7.
- Kenton J D M W C, Toutanova L K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[C]. Proceedings of NAACL-HLT. 2019: 4171-4186.



南京大學
NANJING UNIVERSITY

Q&A

