提纲

南京大学 NANJING UNIVERSITY

- **自然语言处理**
  - 语言模型
  - 文法
  - 句法分析
  - 扩展文法
- **自然语言处理中的深度学习**
  - 词嵌入
  - 循环神经网络
  - 序列到序列模型
  - Transformer架构
  - 预训练和迁移学习

提纲

南京大学 NANJING UNIVERSITY

- **自然语言处理**
  - 语言模型
  - 文法
  - 句法分析
  - 扩展文法
- **自然语言处理中的深度学习**
  - 词嵌入
  - 循环神经网络
  - 序列到序列模型
  - Transformer架构
  - 预训练和迁移学习

语言模型

- 不同的人在不同的时间对于语言的判断会有所差别
- 自然语言存在歧义性和模糊性
- 自然语言没有正式定义从符号到对象的映射

- **语言模型**：描述任意字符串可能性的概率分布

- **词袋模型**
- 朴素贝叶斯方法在单词字符串上的应用
- 生成模型，描述了句子生成的过程

给定一个由单词$w_1, w_2, \ldots w_N$组成的句子，根据朴素贝叶斯公式有

$$\mathbf{P}(Class \mid w_{1:N}) = \alpha \mathbf{P}(Class) \prod_j \mathbf{P}(w_j \mid Class).$$

- 每个类别都有一个装满单词的袋子
- 要生成一段文本，首先选择其中一个袋子并丢掉其他袋子
- 从袋子中重复的随机抽出一个单词并放回
- 重复上述操作直到出现句末指示符（如句号）

语言模型

南京大学 NANJING UNIVERSITY

n元单词模型

- 让单词依赖于它所在句子中之前的所有单词

$$P(w_{1:N}) = \prod_{j=1}^{N} P(w_j \mid w_{1:j-1}).$$

- 捕捉了单词间所有可能的交互，但并不实用
  - 当词汇表中有100 000个单词，句子长度为40时，模型需要估计10^{200} 个参数
  - 使用马尔可夫链模型做一个折中考虑
  - **n元模型（n-gram model）**：长度为n的符号序列称为n元。当n分别为1、2和3时，我们将其分别称作 一元（unigram，即1-gram）、二元（bigram，即2-gram）和三元（trigram，即3- gram）
  - 在n元模型中，每个单词出现的概率仅依赖于前面的n−1个单词

$$P(w_{1:N}) = \prod_{j=1}^{N} P(w_j \mid w_{j-n+1:j-1}).$$

语言模型

南京大学 NANJING UNIVERSITY

其他n元模型

- 字符级模型（character-level model）
  - 每个字符的概率由之前的n−1个字符决定
  - 有助于处理未知单词
  - 有助于处理倾 向于将单词放在一起的语言
  - 适用于语言识别（language identification）任务

- 跳字（skip-gram）模型
  - 考虑那些距离相近的单词，但是跳过它们之间的一个（或多个）单词

n元模型的平滑

- 可能需要处理包含未知单词或未登录词（out-of-vocabulary）的文本。 虽然未知单词或未登录词从未在训练语料库中出现过
- 对未知单词进行建模的一种方法是修改训练语料库，用特殊符号替换不常见单词， 一般替换为<UNK>

- 一些低概率的n元出现在训练语料库中，而其他同样低概率的n元却从未出现
- 对所有相似的n元进行平滑（smoothing）处理，为从未出现在训练语料库中的n元保留模型的一部分概率，以减小模型的方差
  - 使用拉普拉斯平滑估计罕见事件发生的概率
  - 另一种选择是回退模型（backoff model），
    - 首先进行n元计数统计
    - 如果某些序列的统计值很低（或为0），就回退到n−1元.

- **线性插值平滑（linear interpolation smoothing）**是一种通过线性插值将三元模型、二元模型和一元模型组合起来的回退模型

$$\hat{P}(c_i|c_{i-2:i-1}) = \lambda_3 P(c_i|c_{i-2:i-1}) + \lambda_2 P(c_i|c_{i-1}) + \lambda_1 P(c_i),$$

其中 $\lambda_3 + \lambda_2 + \lambda_1 = 1$

语言模型

南京大学 NANJING UNIVERSITY

单词表示

- n元模型缺乏泛化能力，因为它是一个原子模型
- 每个单词都是一个原子且与其他单词不同，没有内部结构
- **词典（dictionary）：**结构化单词模型.
- ***Wordnet*** :
  - 一种机器可读格式的开源手工编写词典
  - 帮助区分名词和动词
  - 获得基本类别

语言模型

南京大学 NANJING UNIVERSITY

词性标注

- 对单词进行分类的一种基本方法，也称为词汇范畴或词汇标注
- 词性允许语言模型捕捉一般模式
- 许多其他自然语言处理任务（如问答或翻译）中非常有用的第一步

- **隐马尔可夫模型**是一种常见的词性标注模型
- 隐马尔可夫模型是一种生成模型，认为产生语言的方法是，从一种状态开始，如 IN，介词状态，然后做两个选择:
  - 应该生成哪个单词
  - 接下来应该生成哪种状态

From	the	start	,	it	took	a	person	with	great	qualities	to	succeed
IN	DT	NN	,	PRP	VBD	DT	NN	IN	JJ	NNS	TO	VB

- 隐马尔可夫模型的一个缺点是我们对语言的所有了解都必须用转移模型和传感器模型来表达。

- **逻辑斯谛回归**
  - 逻辑斯谛回归模型中的权重对应于每个特征对每个类别的预测能力;
  - 权重值可以通过梯度下降法学习

语言模型

南京大学 NANJING UNIVERSITY

词性标注

标签	单词	描述	标签	单词	描述
CC	and	并列连词	PRP$	your	所有格代词
CD	three	基数词	RB	quickly	副词
DT	the	限定词	RBR	quicker	副词，比较级
EX	there	存在 there	RBS	quickest	副词，最高级
FW	per se	外来词	RP	off	小品词
IN	of	介词	SYM	+	符号
JJ	purple	形容词	TO	to	to
JJR	better	形容词，比较级	UH	eureka	感叹词
JJS	best	形容词，最高级	VB	talk	动词，原形
LS	1	列表项标记	VBD	talked	动词，过去式
MD	should	情态动词	VBG	talking	动词，动名词
NN	kitten	名词，单数或不可数	VBN	talked	动词，过去分词
NNS	kittens	名词，复数	VBP	talk	动词，非第三人称单数现在时
NNP	Ali	专有名词，单数	VBZ	talks	动词，第三人称单数现在时
NNPS	Fords	专有名词，复数	WDT	which	wh 限定词
PDT	all	前位限定词	WP	who	wh 代词
POS	's	所有格标记	WP$	whose	wh 所有格代词
PRP	you	人称代词	WRB	where	wh 副词
$	$	美元符号	#	#	英镑符号
“	‘	左引号	”	’	右引号
(	[	左括号	)	]	右括号
,	,	逗号	.	!	句子结束
:	;	句中标点			

Penn Treebank语料库的词性标签（每个标签都有一个示例单词）（Marcus *et al.*, 1993）

词性标注

- 一组词性标注特征可能包括：动词 (VBP)，过去时动词 (VBD)，名词 (NN)，现在时动词 (VBP)

$w_{i-1} =$ "I"	$w_{i+1} =$ "for"
$w_{i-1} =$ "you"	$c_{i-1} =$ IN
w_i ends with "ous"	w_i contains a hyphen
w_i ends with "ly"	w_i contains a digit
w_i starts with "un"	w_i is all uppercase
$w_{i-2} =$ "to" and $c_{i-1} =$ VB	w_{i-2} has attribute PRESENT
$w_{i-1} =$ "I" and $w_{i+1} =$ "to"	w_{i-2} has attribute PAST

语言模型

南京大学 NANJING UNIVERSITY

语言模型的比较

- 为了了解不同的n元模型的效果，我们对本书中的单词建立了一元模型（即词袋）、二元模型、三元模型和四元模型，然后从4个模型中随机采样单词序列。

- $n = 1$: *logical are as are confusion a may right tries agent goal the was*
- $n = 2$: *systems are very similar computational approach would be represented*
- $n = 3$: *planning and scheduling are integrated the success of naive Bayes model is*
- $n = 4$: *taking advantage of the structure of Bayesian networks and developed various languages for writing "templates" with logical variables, from which large networks could be constructed automatically for each problem instance*

文法

- 文法（**grammar**）是一组规则，定义了合法短语的树结构
- 语言 （**language**）是遵循这些规则的句子集
- 句法范畴（**syntactic category**），诸如名词短语或动词短语，有助于对句子中每个位置上可能出现的单词进行约束
- 短语结构（**phrase structure**）为句子的含义或语义提供了框架
- 概率上下文无关文法 (*PCFG*)
  - 概率文法为每个字符串分配一个概率
  - “上下文无关”意味着任一规则都可以在任何上下文中使用
- *PCFG* 文法应用于*Wumpus* 世界
  - 为一小段英语文本定义一 个PCFG文法，该文法适用于探索wumpus世界的智能体之间的通信
  - 文法规则

$$
\begin{array}{lcll}
Adjs & \rightarrow & Adjective & [0.80] \\
 & | & Adjective\ Adjs & [0.20]
\end{array}
$$

文法

$S \rightarrow$	*NP VP*	[0.90] I+feel a breeze
|	*S Conj S*	[0.10] I feel a breeze+and+It stinks
$NP \rightarrow$	*Pronoun*	[0.25] I
|	*Name*	[0.10] Ali
|	*Noun*	[0.10] pits
|	*Article Noun*	[0.25] the+wumpus
|	*Article Adjs Noun*	[0.05] the+smelly dead+wumpus
|	*Digit Digit*	[0.05] 3 4
|	*NP PP*	[0.10] the wumpus+in 1 3
|	*NP RelClause*	[0.05] the wumpus+that is smelly
|	*NP Conj NP*	[0.05] the wumpus+and+I
$VP \rightarrow$	*Verb*	[0.40] stinks
|	*VP NP*	[0.35] feel+a breeze
|	*VP Adjective*	[0.05] smells+dead
|	*VP PP*	[0.10] is+in 1 3
|	*VP Adverb*	[0.10] go+ahead
$Adjs \rightarrow$	*Adjective*	[0.80] smelly
|	*Adjective Adjs*	[0.20] smelly+dead
$PP \rightarrow$	*Prep NP*	[1.00] to+the east
$RelClause \rightarrow$	*RelPro VP*	[1.00] that+is smelly

E_0的文法，每个规则都有示例短语。句法范畴包括句子（*S*）、名词短语（*NP*）、动词短语（*VP*）、形容词列表（*Adjs*）、介词短语（*PP*）和关系从句（*RelClause*）

文法

Noun	→	**stench** [0.05] | **breeze** [0.10] | **wumpus** [0.15] | **pits** [0.05] | ...
Verb	→	**is** [0.10] | **feel** [0.10] | **smells** [0.10] | **stinks** [0.05] | ...
Adjective	→	**right** [0.10] | **dead** [0.05] | **smelly** [0.02] | **breezy** [0.02] ...
Adverb	→	**here** [0.05] | **ahead** [0.05] | **nearby** [0.02] | ...
Pronoun	→	**me** [0.10] | **you** [0.03] | **I** [0.10] | **it** [0.10] | ...
RelPro	→	**that** [0.40] | **which** [0.15] | **who** [0.20] | **whom** [0.02] | ...
Name	→	**Ali** [0.01] | **Bo** [0.01] | **Boston** [0.01] | ...
Article	→	**the** [0.40] | **a** [0.30] | **an** [0.10] | **every** [0.05] | ...
Prep	→	**to** [0.20] | **in** [0.10] | **on** [0.05] | **near** [0.10] | ...
Conj	→	**and** [0.50] | **or** [0.10] | **but** [0.20] | **yet** [0.02] | ...
Digit	→	**0** [0.20] | **1** [0.20] | **2** [0.20] | **3** [0.20] | **4** [0.20] | ...

E_0的词典。*RelPro*是关系代词（relative pronoun）的缩写，*Prep*是介词（preposition）的缩写，*Conj*是连词（conjunction）的缩写。每一类的概率之和都为1

句法分析

- 句法分析（parsing）是根据文法规则分析一串单词以获得其短语结构的过程
- 可以看作对有效句法分析树的搜索，树的叶节点是字符串中的单词
- 可以从S符号开始自顶向下搜索，也可以从单词开始自底向上搜索
- 低效的：如果算法一开始的猜测是错误的，那么它不得不一直回溯到第一个单词，然后在另一种解释下重新分析整个句子
- **动态规划**：每次分析子字符串时，存储分析结果，以免之后重复分析
- 图表句法分析器（chart parser）：将结果记录在一种被称为图表（chart）的数据结构中

句法分析

项目列表	规则
S	
NP VP	*S* → *NP VP*
NP VP Adjective	*VP* → *VP Adjective*
NP Verb Adjective	*VP* → *Verb*
NP Verb **dead**	*Adjective* → **dead**
NP **is dead**	*Verb* → **is**
Article Noun **is dead**	*NP* → *Article Noun*
Article **wumpus is dead**	*Noun* → **wumpus**
the wumpus is dead	*Article* → **the**

根据E_0文法，对字符串“The wumpus is dead”进行句法分析。对于自顶向下的句法分析，我们从*S*开始，在每一步中，将一个非终结符*X*与形式为（$X \rightarrow Y...$）的规则匹配并将项目列表中的*X*替换为*Y...*。例如，用序列*NP VP*替换*S*。对于自底向上的句法分析，我们从“the wumpus is dead”开始，在每一步中，将一串记号，如（*Y...*），与规则（$X \rightarrow Y...$）相匹配，并将记号替换为*X*。例如，用*Article*替换“the”或用*NP*替换*Article*

CYK句法分析算法

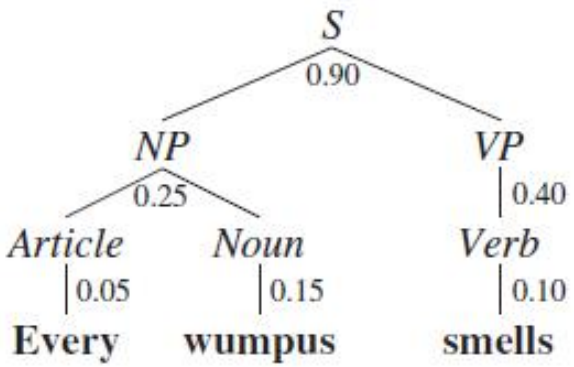
```
function CYK-PARSE(words, grammar) returns 一个句法分析树表
  inputs: words，单词列表
          grammar，LEXICALRULES和GRAMMARRULES结构
  T ← 一个表      // T[X, i, k]是涵盖words_{i:k}的最可能的X树
  P ← 一个表，初始时全为0   // P[X, i, k]表示树T[X, i, k]的概率
  // 为每个单词插入词汇范畴
  for i = 1 to LEN(words) do
    for each (X, p) in grammar.LEXICALRULES(words_i) do
      P[X, i, i] ← p
      T[X, i, i] ← TREE(X, words_i)
  // 通过Y_{i:j} + Z_{j+1:k}构建X_{i:k}，最短区间优先
  for each (i, j, k) in SUBSPANS(LEN(words)) do
    for each (X, Y, Z, p) in grammar.GRAMMARRULES do
      PYZ ← P[Y, i, j] × P[Z, j+1, k] × p
      if PYZ > P[X, i, k] do
        P[X, i, k] ← PYZ
        T[X, i, k] ← TREE(X, T[Y, i, j], T[Z, j + 1, k])
  return T

function SUBSPANS(N) yields (i, j, k)元组
  for length = 2 to N do
    for i = 1 to N + 1 − length do
      k ← i + length − 1
      for j = i to k − 1 do
        yield (i, j, k)
```

句法分析

南京大学 NANJING UNIVERSITY

A*搜索算法

- 不需要搜索整个状态空间
- 可以保证算法找到的第一个分析就是可能性最大的分析 (假设是一个可容许的启发式).
- 通常比CYK算法快，但（取决于文法的细节）仍然比O(n)慢.



根据E_0文法得到的句子 “Every wumpus smells”的句法分析树。树的每个内部节点均标有概率。

句法分析

南京大学 NANJING UNIVERSITY

束搜索算法

- 只考虑 b个最可能的选择
- 不能保证找到概率最高的分析
- 尽管如此，可以在O(n)时间内运行，并且在大多数情况下仍能找到最佳的句法分析
- b = 1的束搜索句法分析器称为**确定性句法分析器**
- **移位减少句法分析（shift-reduce parsing）**
  - 逐字遍历整个句子
  - 在每一点上选择将单词移到成分栈上
  - 或根据文法规则减少栈最顶端的成分

扩展文法

南京大学 NANJING UNIVERSITY

- 代词范畴
  - “I”：可以作为句子的主语，单数
  - “us”：不可以作为句子的主语，复数
  - 像代词这样的范畴，已经用诸如“主格、第一人称单数”之类的特征进一步扩展，称为子范畴（subcategory）

- **词汇化PCFG**（lexicalized PCFG）：一种扩展文法，它允许我们根据短语中单词的属性而不仅仅是句法范畴来分配概率

$VP(v) \rightarrow Verb(v)\ NP(n)$	$[P_1(v,n)]$
$VP(v) \rightarrow Verb(v)$	$[P_2(v)]$
$NP(n) \rightarrow Article(a)\ Adjs(j)\ Noun(n)$	$[P_3(n,a)]$
$NP(n) \rightarrow NP(n)\ Conjunction(c)\ NP(m)$	$[P_4(n,c,m)]$
$Verb(\mathbf{ate}) \rightarrow \mathbf{ate}$	[0.002]
$Noun(\mathbf{banana}) \rightarrow \mathbf{banana}$	[0.0007]

- 符号$VP(v)$表示范畴为VP且头为v的短语
- 这里 $P_1(v, n)$ 表示一个头为v的VP 加上一个头为n的 NP 形成一个VP的概率

扩展文法

- 将这些事实完全编码在概率条目中, 例如: 对所有动词v，$P_1(v, she)$ 都是一个非常小的值

$S(v) \rightarrow NP(Sbj, pn, n)\ VP(pn, v)\ [P_5(n, v)]$

该规则表示，当一个NP后跟一个VP时，它们可以形成一个S，但前提是NP是主格 （Sbj），并且NP和VP的人称和单复数（pn）相同。

$$
\begin{aligned}
S(v) &\rightarrow NP(Sbj,pn,n)\ VP(pn,v) \mid \ldots \\
NP(c,pn,n) &\rightarrow Pronoun(c,pn,n) \mid Noun(c,pn,n) \mid \ldots \\
VP(pn,v) &\rightarrow Verb(pn,v)\ NP(Obj,pn,n) \mid \ldots \\
PP(head) &\rightarrow Prep(head)\ NP(Obj,pn,h) \\
Pronoun(Sbj,1S,\mathbf{I}) &\rightarrow \mathbf{I} \\
Pronoun(Sbj,1P,\mathbf{we}) &\rightarrow \mathbf{we} \\
Pronoun(Obj,1S,\mathbf{me}) &\rightarrow \mathbf{me} \\
Pronoun(Obj,3P,\mathbf{them}) &\rightarrow \mathbf{them} \\
Verb(3S,\mathbf{see}) &\rightarrow \mathbf{see}
\end{aligned}
$$

保有格一致性、主语-动词一致性和短语头单词的扩展文法的一部分

扩展文法

南京大学 NANJING UNIVERSITY

语义解释

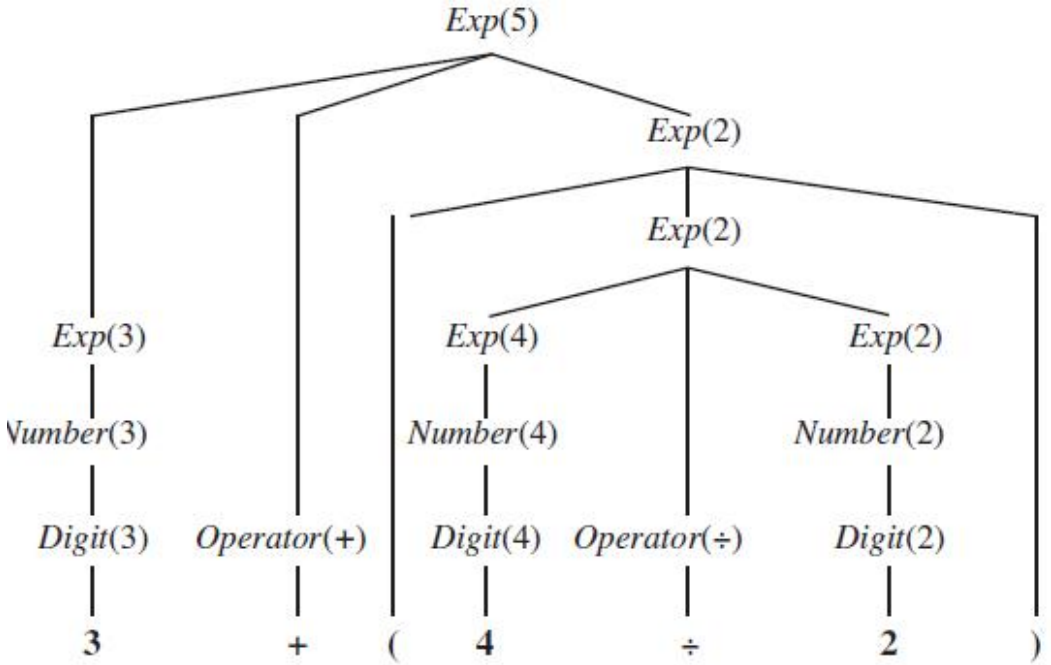
$$
\begin{aligned}
&Exp(op(x_1,x_2)) \rightarrow Exp(x_1)\ Operator(op)\ Exp(x_2)\\
&Exp(x) \rightarrow (\ Exp(x)\ )\\
&Exp(x) \rightarrow Number(x)\\
&Number(x) \rightarrow Digit(x)\\
&Number(10\times x_1+x_2) \rightarrow Number(x_1)\ Digit(x_2)\\
&Operator(+) \rightarrow +\\
&Operator(-) \rightarrow -\\
&Operator(\times) \rightarrow \times\\
&Operator(\div) \rightarrow \div\\
&Digit(0) \rightarrow \mathbf{0}\\
&Digit(1) \rightarrow \mathbf{1}\\
&\ldots
\end{aligned}
$$

算术表达式的文法，作了语义扩展。每个变量x_i表示一个成分的语义。这些文法规则遵循组合语义（compositional semantics）原则——短语的语义是其子短语语义的函数

扩展文法

南京大学 NANJING UNIVERSITY

语义解释



字符串“3 + (4÷2)”的带语义解释的句法分析树

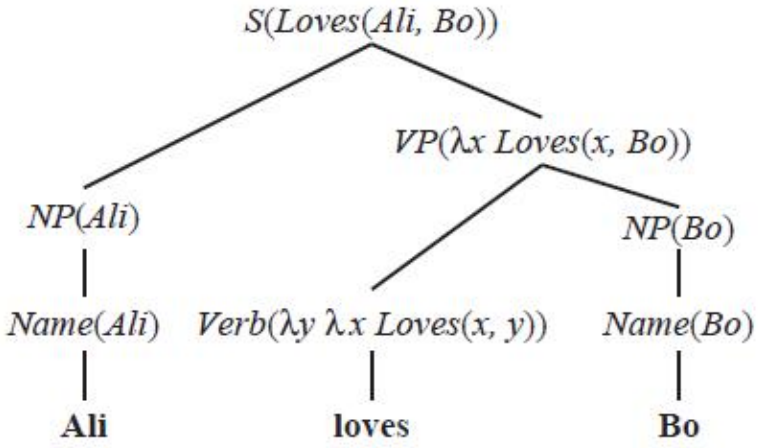
扩展文法

语义解释

$S(pred(n)) \rightarrow NP(n)\ VP(pred)$
$VP(pred(n)) \rightarrow Verb(pred)\ NP(n)$
$NP(n) \rightarrow Name(n)$

$Name(Ali) \rightarrow$ **Ali**
$Name(Bo) \rightarrow$ **Bo**
$Verb(\lambda y\ \lambda x\ Loves(x,y)) \rightarrow$ **loves**

(a)



(b)

（a）一种文法，可以推导出“Ali loves Bo”（以及其他3个句子）的句法分析树和语义解释。每个范畴都用一个表示语义的参数进行扩展。（b）字符串“Ali loves Bo”的带有语义解释的句法分析树

扩展文法

南京大学 NANJING UNIVERSITY

学习语义文法

- 给定大量下述样例对以及每个新领域的少量人工标注知识，系统就会生成合理的词汇条目

句子: What states border Texas?
逻辑形式: $\lambda x.state(x) \wedge \lambda x.borders(x, Texas)$

- 收集问题-答案对样例要更容易:

问题: What states border Texas?
回答: Louisiana, Arkansas, Oklahoma, New Mexico.
问题: How many times would Rhode Island fit into California?
回答: 135

真实自然语言的复杂性

量词限定（quantification）：

- 标准方法：文法不定义实际的逻辑语义语句，而是定义一种准逻辑形式（quasi-logical form），可以通过分析过程之外的算法将该形式转化为逻辑语句。
- 有选择量词限定范围的优先规则

语用学（pragmatics）：

- 确定索引词（indexical）的含义，索引词是直接指代当前情景的短语
- 例如，在句子“I am in Boston today”中，“I”和“today”都是 索引词。单词“I”可以用Speaker表示，在不同时间指代不同对象
- 解释**说话者的意图**
- 说话者的话语被认为是一种言语行为，需要由听者辨认它是一种什么类型的行为，如提问、陈述、承诺、警 告、命令等

真实自然语言的复杂性

南京大学 NANJING UNIVERSITY

长距离依存关系（long-distance dependencies）

时间和时态（time and tense）：

- 英语使用动词时态（过去时、现在时和将来时）表示某个事件的相对时间
- 表示事件时间的一个不错选择是事件演算符号

Ali loves Bo: $E_1 \in Loves(Ali, Bo) \wedge During(Now, Extent(E_1))$
Ali loved Bo: $E_2 \in Loves(Ali, Bo) \wedge After(Now, Extent(E_2))$.

$Verb(\lambda y\ \lambda x\ e \in Loves(x, y) \wedge During(Now, e)) \rightarrow \textbf{loves}$
$Verb(\lambda y\ \lambda x\ e \in Loves(x, y) \wedge After(Now, e)) \rightarrow \textbf{loved}$.

真实自然语言的复杂性

歧义（ambiguity）：

- 倾向于将歧义视为沟通失败
- 当听者意识到话语中的歧义时，意味着这句话是不清楚或令人困惑的
- 词汇歧义（lexical ambiguity）是指一个单词有不止一种含义
  - “back” 可以是副词 (go back),
  - 形容词 (back door),
  - 名词 (the back of the room),
  - 动词 (back a candidate), or
  - 专有名词 (a river in Nunavut, Canada).

句法歧义指一个 短语有多种分析，句法歧义会导致**语义歧义**

真实自然语言的复杂性

歧义消解（disambiguation）是还原一句话最可能的含义的过程。

正确消歧需要结合以下4个模型:

- **世界模型**：一个命题在世界上发生的可能性
- **心理模型**：说话者形成向听者传达某个事实的意图的可能性
- **语言模型**：在说话者已有传达某个特定事实的意图的情 况下，选择某个特定单词串的可能性
- **声学模型**：对语音交流来说，在说话者已经选择某个特 定单词串的情况下，生成特定声音序列的可能性。

自然语言任务

- **语音识别**是将语音转换为文本的任务。
  - 可以对生成的文本执行进一步的任务（如问答）
  - 挑战：即使个别单词有错误，也要做出适当的响应.
- **文本-语音合成（text-to-speech ）**则是与语音识别相反的过程，将文本转换为声音。另一个发展领域是合成不同的声音，从普通男性或女性的声音开始，接着可以合成地方方言，甚至模仿名人的声音。
- **机器翻译**将文本从一种语言转换到另一种语言。
- **信息提取**是通过浏览文本并查找文本中特定类别的对象及其关系来获取知识的过程。
- **信息检索**的任务是查找与给定查询相关且重要的文档。
- **问答**与信息检索不同，它的查询其实是一个问题。

提纲

南京大学 NANJING UNIVERSITY

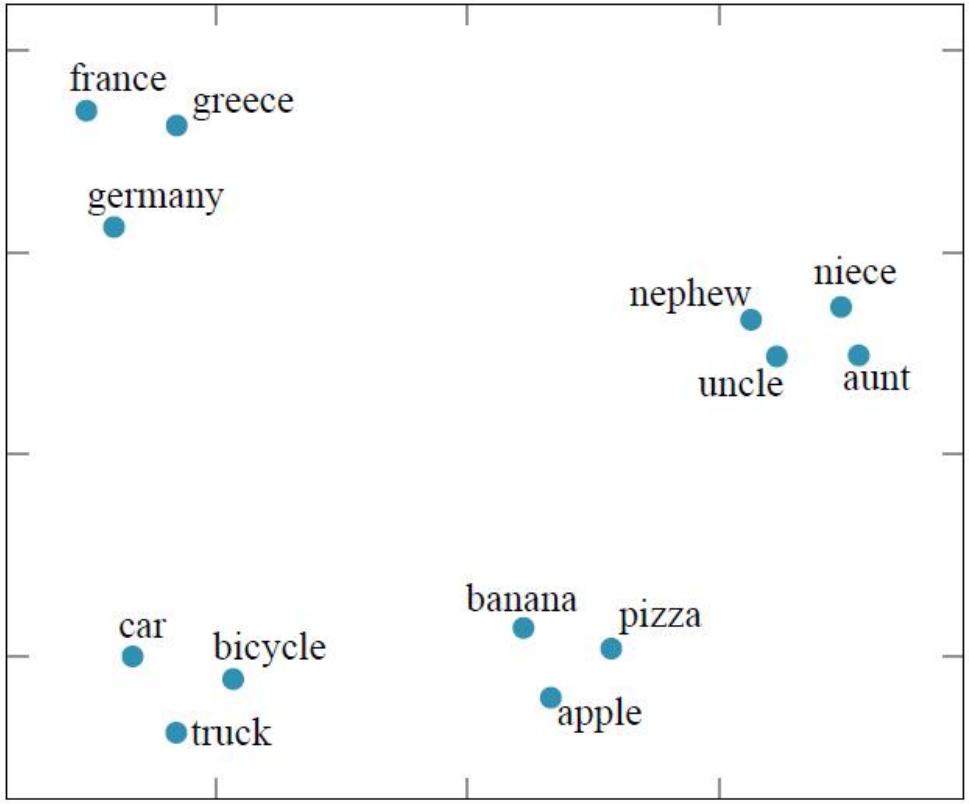
- 自然语言处理
  - 语言模型
  - 文法
  - 句法分析
  - 扩展文法
- **自然语言处理中的深度学习**
  - 词嵌入
  - NLP中的循环神经网络
  - 序列到序列模型
  - Transformer架构
  - 预训练和迁移学习

词嵌入

- 希望得到一种不需要手工特征工程的单词表示，但是能在相关单词之间进行泛化

- **词嵌入（*word embedding*）**：表示单词的低维向量
  - 从数据中自动学习
  - 语义相似的单词会具有相似的向量值
  - 事实证明对于下游语言任务（如问答、翻译或摘要）词嵌入是一种很好的表示
  - 可以使用任意一种通用预训练向量执行某个特定的自然语言处理任务

词嵌入



在60亿字的文本上训练得到的GloVe算法计算出的词嵌入向量。本图这种可视化将100维的词向量投影到二维空间上。相似的单词在图中彼此接近。

词嵌入

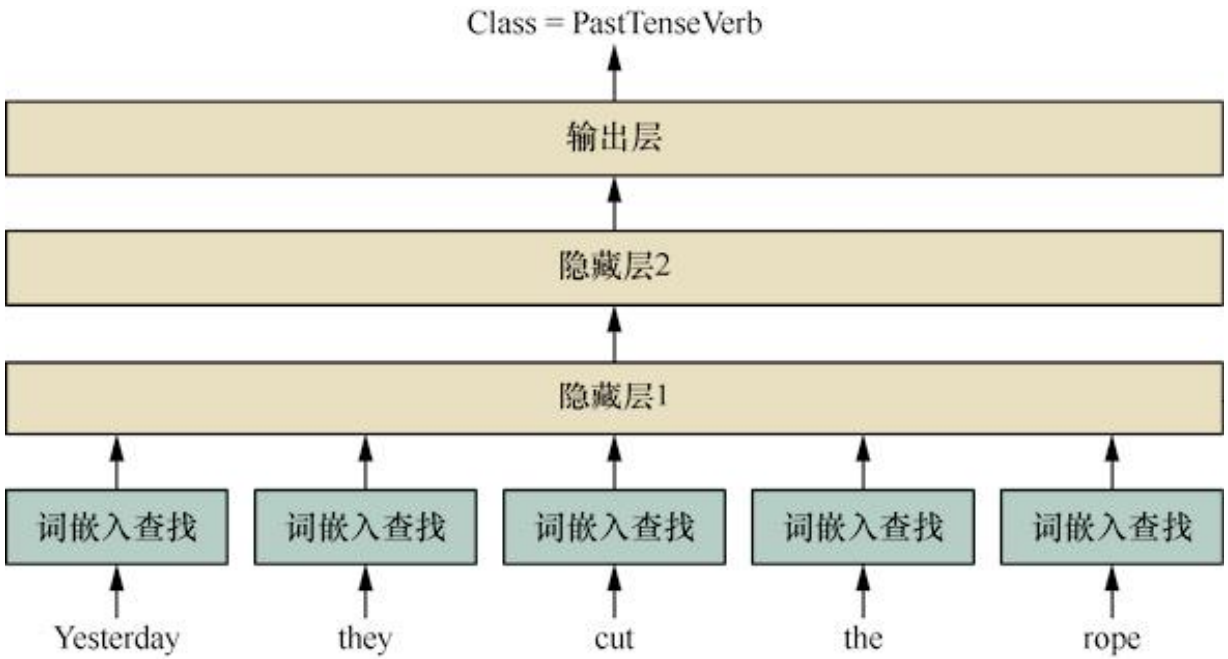
A	*B*	*C*	$D = C + (B - A)$	关系
Athens	Greece	Oslo	Norway	*Capital*
Astana	Kazakhstan	Harare	Zimbabwe	*Capital*
Angola	kwanza	Iran	rial	*Currency*
copper	Cu	gold	Au	*Atomic Symbol*
Microsoft	Windows	Google	Android	*Operating System*
New York	New York Times	Baltimore	Baltimore Sun	*Newspaper*
Berlusconi	Silvio	Obama	Barack	*Firstname*
Switzerland	Swiss	Cambodia	Cambodian	*Nationality*
Einstein	scientist	Picasso	painter	*Occupation*
brother	sister	grandson	granddaughter	*Family Relation*
Chicago	Illinois	Stockton	California	*State*
possibly	impossibly	ethical	unethical	*Negative*
mouse	mice	dollar	dollars	*Plural*
easy	easiest	lucky	luckiest	*Superlative*
walking	walked	swimming	swam	*Past tense*

词嵌入模型有时可以通过向量算法回答以下问题："**A**之于**B**就像**C**之于[什么]？"。向量算法指给定单词的词嵌入向量$\boldsymbol{A}$、$\boldsymbol{B}$和$\boldsymbol{C}$，计算向量$\boldsymbol{D} = \boldsymbol{C} + (\boldsymbol{B} - \boldsymbol{A})$，然后查找最接近$\boldsymbol{D}$的单词。（$\boldsymbol{D}$列中的答案由模型自动计算，"关系"列中的描述是手动添加的。）

词性标注

- 给定带有词性标注的句子语料库，我们将同时学习词嵌入和词性标注器的参数。
- 具体过程如下：
  - 选择用于标注每个单词词性的预测窗口的宽度w（奇数个单词）；
  - 创建词汇表，包含所有在训练数据中出现5次以上的唯一单词记号。词汇表中的单词总数记为v；
  - 以任意顺序（可能是字母序）对词汇表进行排序；
  - 选择每个词嵌入向量的大小d；
  - 创建一个新的v×d权重矩阵，称为E。这就是词嵌入矩阵。E的第i行是词汇表 中第i个词的词嵌入。对E进行随机初始化（或使用预训练向量）；
  - 建立一个输出词性标注的神经网络；
  - 将一个包含w个单词的序列编码为一个输入向量。只需要查找每个单词的词嵌入，然后连接成一个向量；
  - 使用梯度下降法训练权重矩阵E和其他权重矩阵$\mathbf{W}_1$, $\mathbf{W}_2$ 和 $\mathbf{W}_{out}$

词嵌入

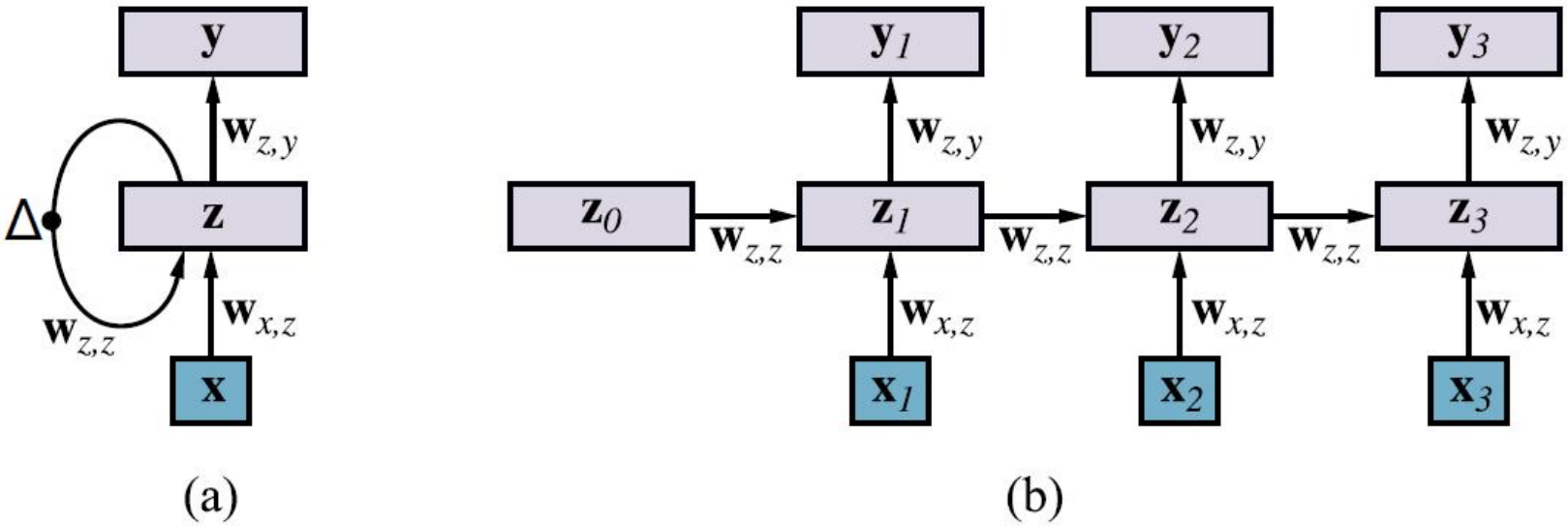


前馈词性标注模型。模型接受一个包含5个单词的窗口作为输入，然后预测中间单词（此处为“cut”）的词性。该模型能够说明单词的位置，因为5个输入词嵌入中的每一个都与第一个隐藏层的不同部分相乘。在训练过程中同时学习词嵌入及三层网络的参数值

NLP中循环神经网络

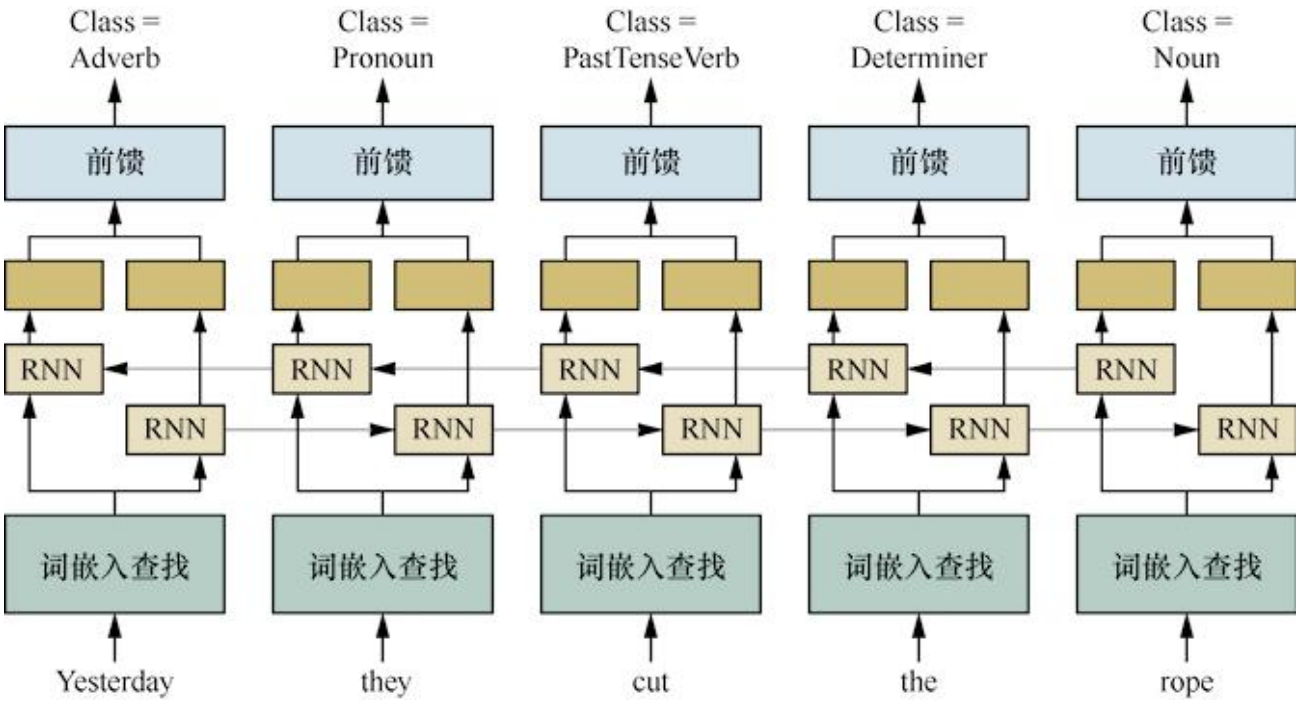
- 前馈网络还存在不对称的问题并且模型的参数太多
- 在一个RNN语言模型中，每个输入单词都被编码为一个词嵌入向量, $\mathbf{x}_i$. 隐藏层 $\mathbf{z}_t$作为输入从一个时间步传递到下一个时间步
- 输出 $\mathbf{y}_t$ 为句子中下一个单词可能值的softmax概率分布
- 权重矩阵$w_{,z,z}$, $w_{,x,z}$ $w_{,z,y}$的参数数量是固定的，与单词数量无关
- RNN架构还解决了不对称问题，因为每个单词位置的权重都相同
- 训练RNN来解决上述分类问题的方法与语言模型相同。
  - 训练数据需要标签，例如词性标注或指代指示
- 为了捕捉右侧的上下文信息，我们可以使用双向RNN，它将 一个单独的从右到左模型连结到一个从左到右模型上。

NLP中循环神经网络



（a）RNN示意图，其中隐藏层z具有循环连接，符号Δ表示延迟。每个输入x是句子中下一个单词的词嵌入向量。每个输出y是该时间步的输出。（b）同一网络在3个时间步上展开以创建前馈网络。注意，权重在所有时间步中是共享的

NLP中循环神经网络



用于词性标注的双向RNN网络

NLP中循环神经网络

南京大学 NANJING UNIVERSITY

自然语言处理任务中的LSTM模型

长短期记忆（LSTM）模型是一种带有门控单元的RNN

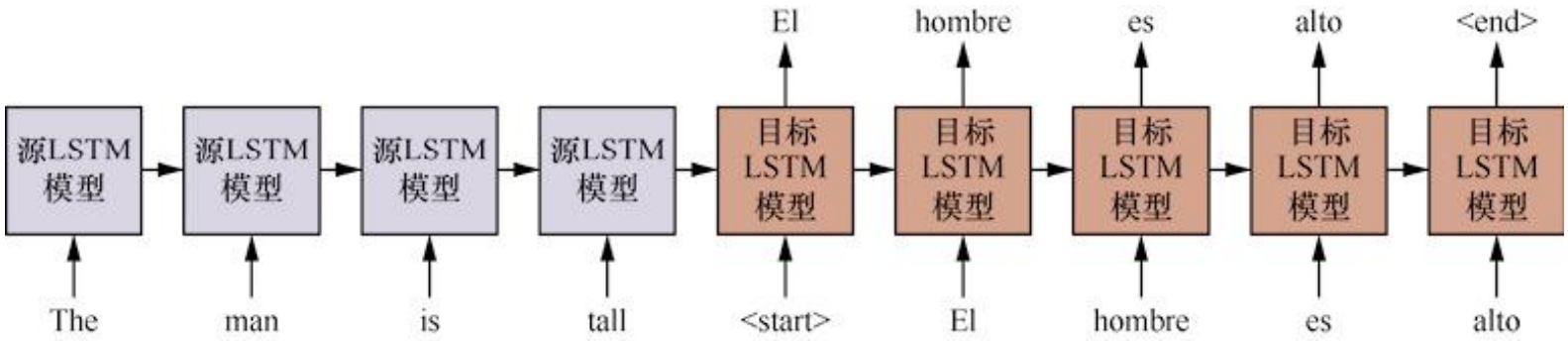
- 不会遇到无法完美地从一个时间步复制消息到下一个时间步的问题
- 可以选择记住输入的某些部分，并将其复制到下一个时间步，而忘记其他部分

序列到序列模型

南京大学 NANJING UNIVERSITY

- 机器翻译的目标是将一个句子从源语言翻译到目标语言
- **序列到序列模型**使用两个RNN，一个用于源语言，另一个用于目标语言
- 在源语句上运行源RNN，然后使用源RNN的最终隐藏状态作为目标RNN的初始隐藏状态
- 每个目标词都隐式地取决于整个源句和前面的目标词
- 最常用于机器翻译，但也可用于许多其他任务，例如从图像中自动生成文本描述，长文本改写为含义相同的短文本
- 主要的缺点:
  - 邻近上下文偏差
  - 固定上下文大小限制
  - 缓慢的顺序处理

序列到序列模型



基本的序列到序列模型。每个块代表一个LSTM模型时间步。（简单起见，图中没有显示嵌入层和输出层。）我们向网络中连续输入源句“The man is tall”的单词，后跟<start>标记，表示网络要开始生成目标句。源句末尾的最终隐藏状态被用作目标句的初始隐藏状态。然后，将时刻t的目标句单词作为时刻t + 1的输入，直到网络生成<end>标记，表示句子完成

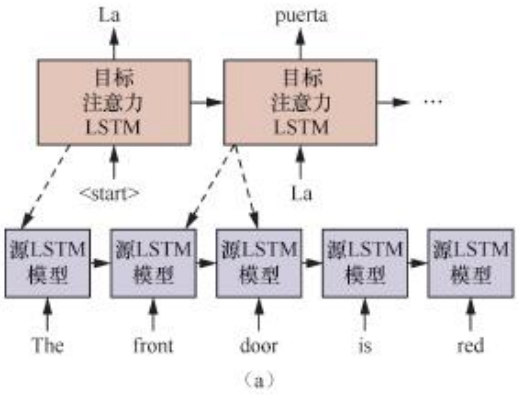
序列到序列模型

南京大学 NANJING UNIVERSITY

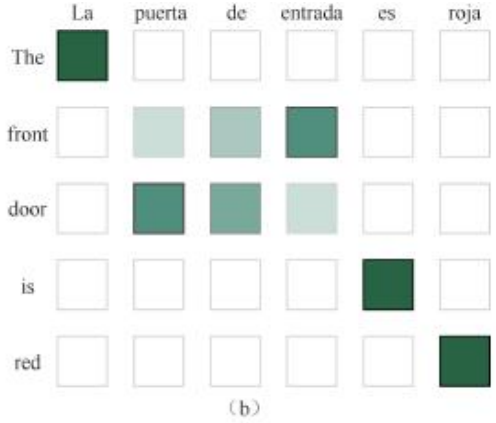
注意力

- 当目标RNN一次生成一个目标单词时，与每个目标单词实际相关的很可能只有源句中的一小部分；对于每个单词，目标RNN必须注意源句的不同部分。
- 注意力序列到序列模型的目标RNN可以写为：$\boldsymbol{h}_i = RNN(\mathbf{h}_{i-1}, [\mathbf{x}_i; \mathbf{c}_i])$
  其中$\mathbf{c}_i$定义为：

$$r_{ij} = \mathbf{h}_{i-1} \cdot \mathbf{s}_j$$
$$a_{ij} = e^{r_{ij}} / (\sum_k e^{r_{ik}})$$
$$\mathbf{c}_i = \sum_j a_{ij} \cdot \mathbf{s}_j$$



（a）



（b）

序列到序列模型

南京大学 NANJING UNIVERSITY

解码

- 目标是生成相应的目标句
- 可以一次生成一个目标词，然后在下一个时间步将其反馈回网络

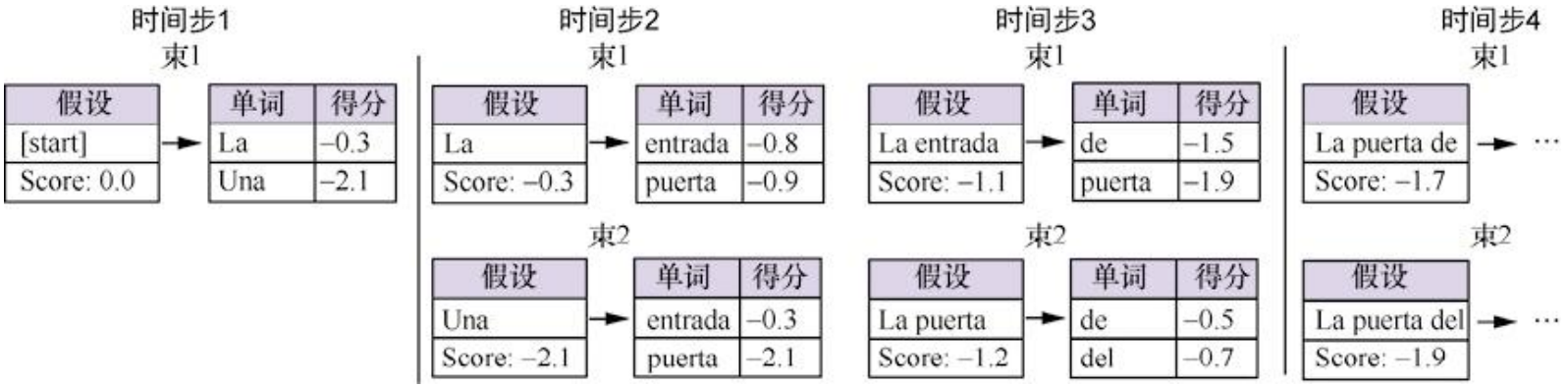
贪心解码

- 最简单的解码形式是在每个时间步都选择概率最高的单词，然后将这个单词作为输 入反馈到下一个时间步
- 可能无法最大化整个目标序列的概率
- 更好的方法是使用搜索算法来搜索一个最优解码

束搜索

- 在机器翻译解码中，束搜索通常在 每个阶段保留最可能的k个假设，对于每个假设，在下一步都继续保留最可能的k个单词,
- 然后从得到的k^2个新假设中再选出最好的k个
- 当束中的所有假设都生成了特殊的<end>记号时，算法输出得分最高的假设
- 目前最高水平的神经网络机器翻译模型使用的束大小为4～8，而较 老的统计机器翻译模型使用的束大小至少为100

序列到序列模型



束大小b = 2的束搜索。每个单词的得分是目标RNN中softmax生成的对数概率，每个假设的得分是单词得分的总和。在时间步3中，得分最高的假设“La entrada”只能生成低概率的延续，因此它“从束上脱落”

Transformer架构

南京大学 NANJING UNIVERSITY

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

NeurIPS 2017

Attention is **all you need**

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

… to attend to **all** positions in the decoder up to and including that position. **We need** to prevent … **We** implement this inside of scaled dot-product **attention** by masking out (setting to $-\infty$) …

☆ Save 99 Cite Cited by 100238 Related articles All 62 versions »

Transformer架构

南京大学 NANJING UNIVERSITY

自注意力

- 在序列到序列模型中，注意力是从目标RNN到源RNN
- 自注意力让每个隐藏状态序列同样关注自身
  - 从源到源
  - 从目标到目标
  - 可以额外捕获每个序列中的长距离（以及邻近）上下文

- 使用3个不同的权重矩阵将输入投影到3种不同的表示中
  - **查询向量** $\mathbf{q}_i = \boldsymbol{W}_q\mathbf{x}_i$是注意力来自（from）的对象
  - **键向量** $\mathbf{k}_i = \mathbf{W}_k\mathbf{x}_i$是注意力去到（to）的对象
  - **值向量** $\mathbf{v}_i = \mathbf{W}_v\mathbf{x}_i$是正在生成的上下文

$$
\begin{aligned}
r_{ij} &= (\mathbf{q}_i \cdot \mathbf{k}_j)/\sqrt{d} \\
a_{ij} &= e^{r_{ij}}/(\sum_k e^{r_{ik}}) \\
\mathbf{c}_i &= \sum_j a_{ij} \cdot \mathbf{v}_j ,
\end{aligned}
$$

,

Transformer架构

南京大学 NANJING UNIVERSITY

自注意力

- 自注意力机制是非对称的, r_{ij} 不同于 r_{ji}.
- 加入的比例因子$\sqrt{d}$可以提高数值稳定性
- 一个句子中所有单词的编码可以 同时计算
- 有时会丢失重要的信息，因为它基本上是整个句子的平均

多头注意力（multiheaded attention）

- 我们把句子复制成m个相同的部分，然后把注意力模型应用于每一部分
- 每一部分都有自己的一组权重
- 最后将结果连接在一起。通过连接而不是求和，更容易凸显出重要的子模块

，

Transformer架构

南京大学 NANJING UNIVERSITY

从自注意力到Transformer

- 每个Transformer层由几个子层组成

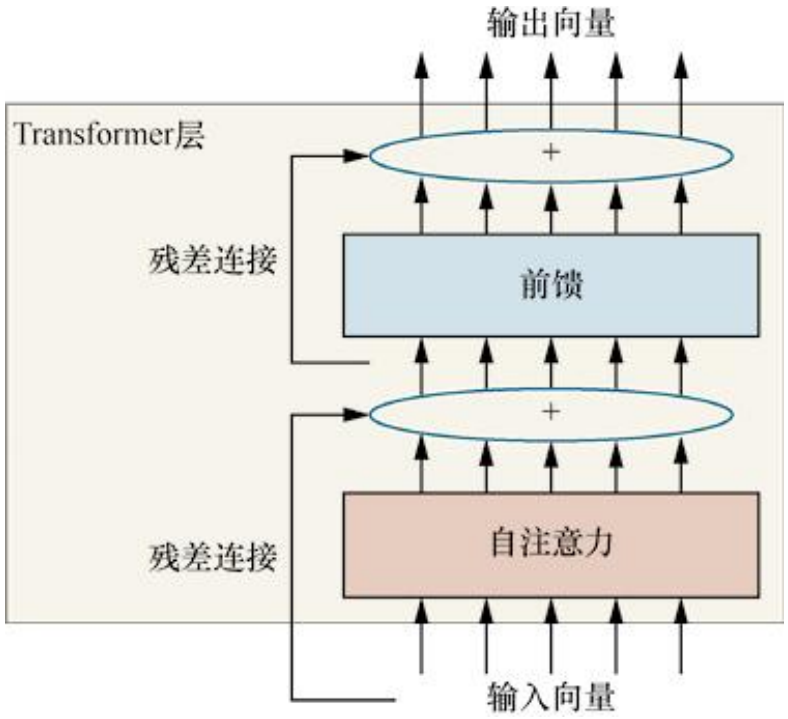
对于每个Transformer 层:

- 首先应用自注意力。
- 注意力模块的输出通过前馈层，每个位置上分别应用相同的前馈权重矩阵，之后应用非线性激活函数（通常为ReLU）。
- 为了解决梯度消失问题，Transformer层中还添加了两个残差连接。
- Transformer架构并没有显式地捕捉序列中单词的顺序，因为上下文仅通过自注意力建模，而自注意力与单词顺序无关。
- 使用**位置嵌入（positional embedding）**捕捉单词顺序
  - 如果输入序列的最大长度为n，那么我们将学习n个新的嵌入向量，每个单词位置对应一个向量；
  - 第一个Transformer层的输入是位置t的词嵌入加上对应位置t的位置嵌入。
- 编码器和解码器几乎是相同的，除解码器使用的自注意力之外，在解码器中，每个单词只能注意它前面的单词，因为文本是从左到右生成的。

Transformer架构

南京大学 NANJING UNIVERSITY

从自注意力到Transformer

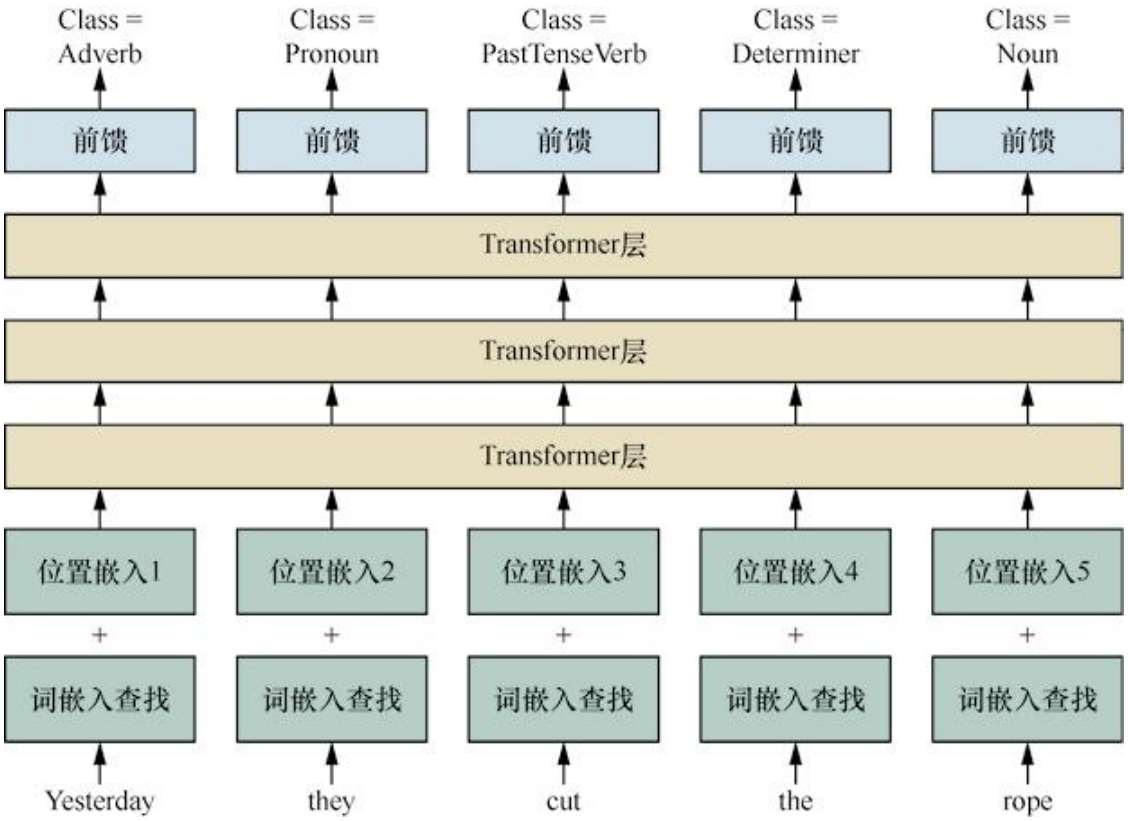


单层Transformer由自注意力、前馈网络和残差连接组成

Transformer架构

南京大学 NANJING UNIVERSITY

从自注意力到Transformer



使用Transformer架构进行词性标注

预训练词嵌入

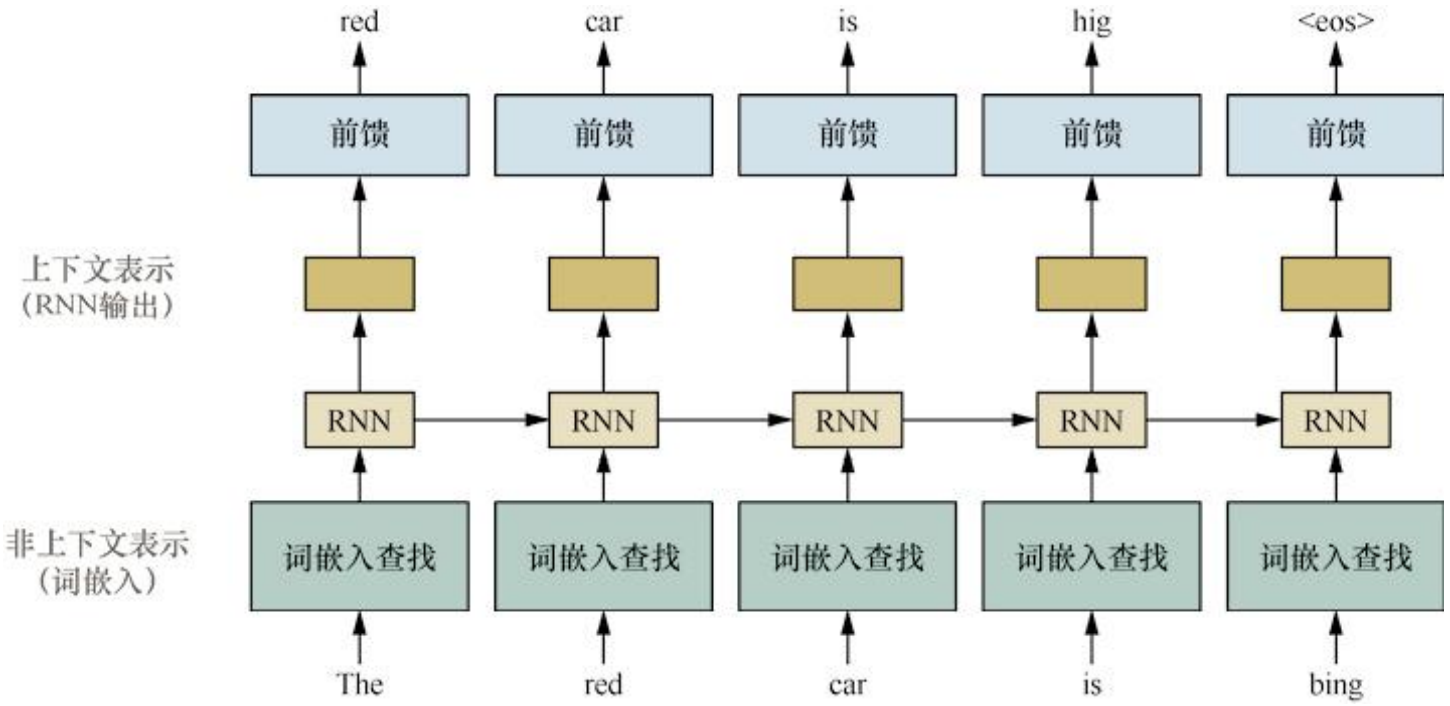
GloVe (全局向量) 模型

- 通过将两个单词与其他单词进行比较，充分地捕捉到两个单词之间的关系
- 首先统 计每个单词在另一个单词的窗口中出现的次数
- 选择窗口大小 （例如5个单词），并将X_{ij}定义为单词*i*和*j*在一个窗口内同时出现的次数, X_i为单词*i*与其他任何单词同时出现的次数
- 令 $P_{ij} = X_{ij}/X_i$为单词*j*在单词*i*的上下文中出现的概率，$\mathbf{E}_i$为单词*i*的嵌入

$$\boldsymbol{E}_i \cdot \boldsymbol{E}_j' = \log(\alpha P_{ij})$$

预训练和迁移学习

预训练上下文表示



使用从左到右的语言模型训练上下文表示

预训练和迁移学习

南京大学 NANJING UNIVERSITY

掩码语言模型

- 预测是从左到右的。但有时句子中后面的上下文也有助于阐明前面的单词

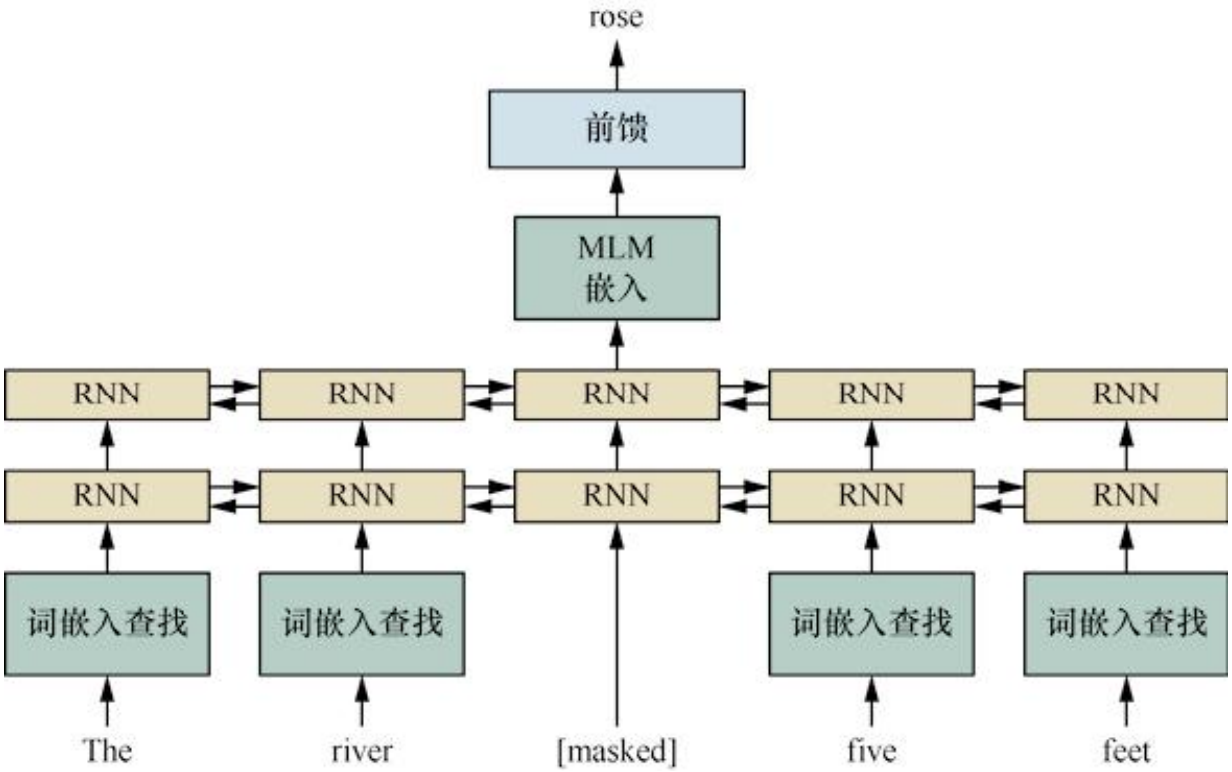
- **掩码语言模型**的训练方法是掩码（隐藏）输入中的单个单词，然后要求模型预测被掩码的单词
- 对于这一任务，我们可以在被掩码的句子上使用一个深度双向RNN或Transformer模型
- 使用对应被掩码的记号的最后的隐藏向量来预测被掩码的单词
- 在训练过程中，一个句子可以在不同单词被掩码的情况下 多次使用
  - 不需要标记数据；句子本身为被掩码的单词提供了标签。

- 如果在大规模文本语料库上训练这个模型，它会生成预训练的表示，这些表示在各 种自然语言处理任务（机器翻译、问答、摘要、语法判断等）中都表现良好。

预训练和迁移学习

南京大学 NANJING UNIVERSITY

掩码语言模型



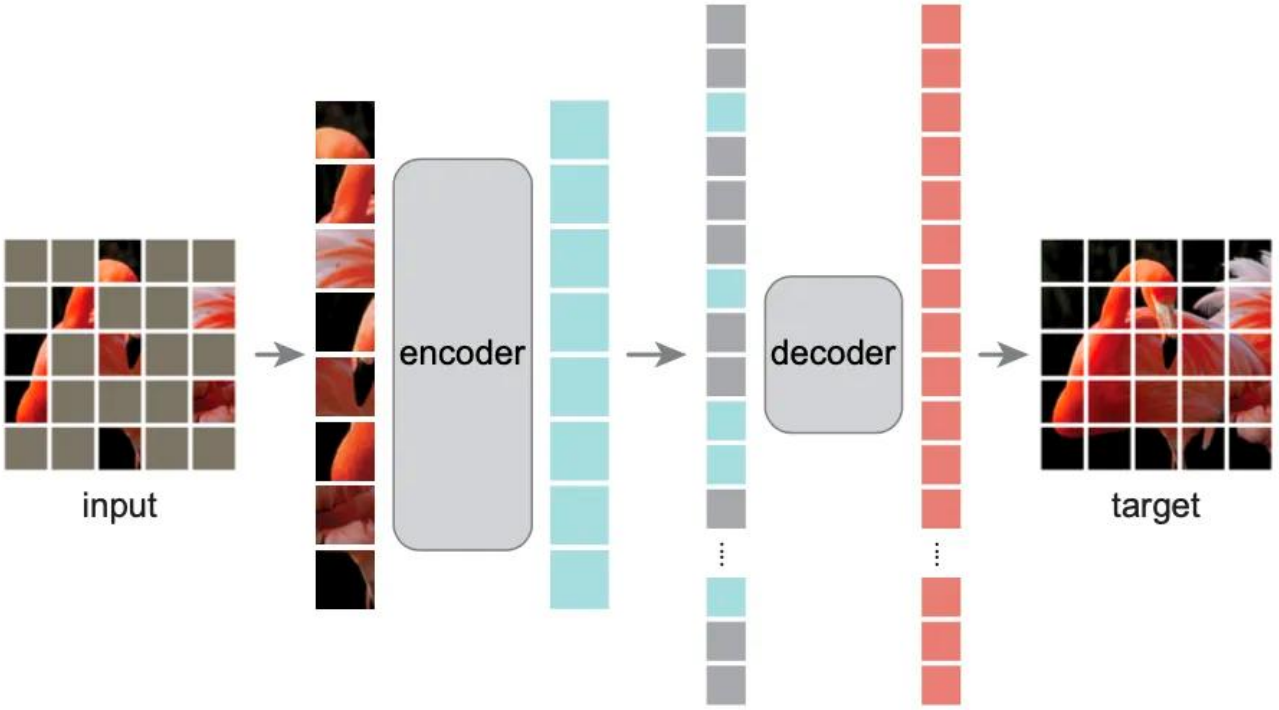
掩码语言建模：通过掩码（隐藏）输入单词并只预测那些被掩码的单词

预训练一个双向模型，如一个多层RNN

预训练和迁移学习

南京大学 NANJING UNIVERSITY

掩码图像模型（Masked Autoencoders）



Masked Autoencoders Are Scalable Vision Learners (K. He et al., 2021)

小结

- 使用词嵌入的词的连续表示比离散的原子表示更加健壮，并且可以使用未标注的文本数据进行预训练。

- 通过在隐藏状态向量中保留相关信息，循环神经网络可以有效地对局部和远距离上下文建模。

- 序列到序列模型可用于机器翻译和文本生成。

- Transformer模型使用自注意力机制，可以对远距离上下文和局部上下文进行建模。它们可以有效地利用硬件矩阵乘法。

- 包含预训练的上下文词嵌入的迁移学习允许从非常大的未标注语料库中开发模型，并应用于一系列任务。在目标领域进行微调后，通过预测缺失词预训练的模型可以处理该领域的任务，如问答和文本蕴含。