

课程内容

NLP基础技术	NLP应用技术
<ul style="list-style-type: none">• 正则表达式与自动机• 语言模型• 词性标注• 句法剖析• 统计剖析• 向量语义• 词汇语义• 词义排歧	<ul style="list-style-type: none">• 信息抽取• 问答• 信息检索• 摘要• 机器翻译

By Ying-xin Wu

WEEK 1

- 正则表达式 (Regular Expression)
- 有限状态自动机 (Finite State Automata)
- 分词 (Word Segmentation)

正则表达式

表示：

析取 a. [X1X2…Xn]

任意 i, Xi

(1) [wW]ow: w|W o: w|W o w (2)[A-Z] 里面取一个

b. X1|…|Xn =[X1X2…Xn]

否定 [^X1X2…Xn] 否定 Xi, 任意 i

(1) [^Ss]: S, s 都不要

(2) 要是不在[]最前面或者在[]外面，都只表示它自身

布尔 ab? b 可以要也可以不要

连续的都要 - [a-zA-Z]

>=0 个重复 * wo*w: wo[...]ow

>=1 个重复 + wo+w

全部匹配 . beg.n

固定位置匹配 ^X X一定要在句首； Y\$ Y一定要在句尾

注意这些特殊符号自己用的时候要用 \ 转义，比如* 才表示.*

常见的匹配错误

False positives 匹配了但是不对: precision

False negatives 对但是没匹配到: recall

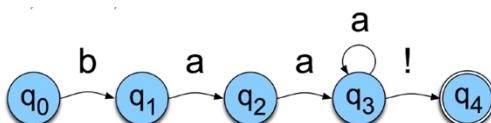
例子: the → (avoid FN) 考虑开头首字母 [Tt] he → (avoid FP) [^a-zA-Z][Tt] he [^a-zA-Z]

FSA 有限状态自动机 (Finite-State Automata)

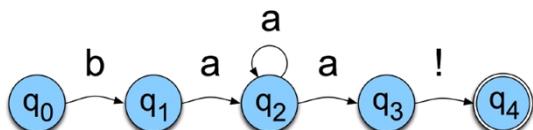
正则语言是特殊的 形式语言: 一组由有限的符号组成的字符串

[baa⁺] 可以有多种表示

确定性



不确定性



Q 状态: 看有几个圈; 开始状态无入度, 结束状态有两个环

$Q \times \Sigma \rightarrow Q$ 转移表: 看有几个箭头

Σ . 字母表: 看转移过程中用到的字母集合

Generators: 按 FSA 规则产生语言

Acceptors: 通过看是否达到结束状态来判断任意 string 符不符合形式语言

好处是这样可以用有限的字母表以及状态转移表述无限的序列, 从而进行匹配。

识别: 输入是一个 tap, 按照 FSA 的规则, 按 tap 走, 直到 tap 走完。

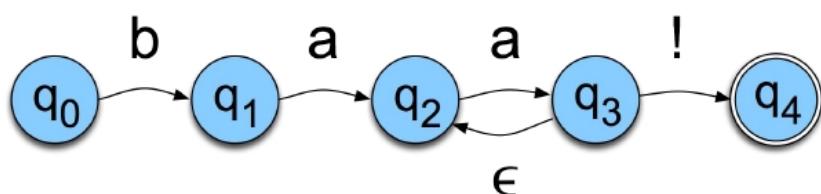
确定性: 在($tap[i], qj$), $qj \rightarrow tap[i+1] \rightarrow qk$, qk 是确定的(只有一条路可走)

普适性: 改变转移表, 即可对应所有的 FSAs

搜索: 搜索状态为($tap[i], qj$)

成功: $tap[-1]$ 对应终止的 q

失败: 其它



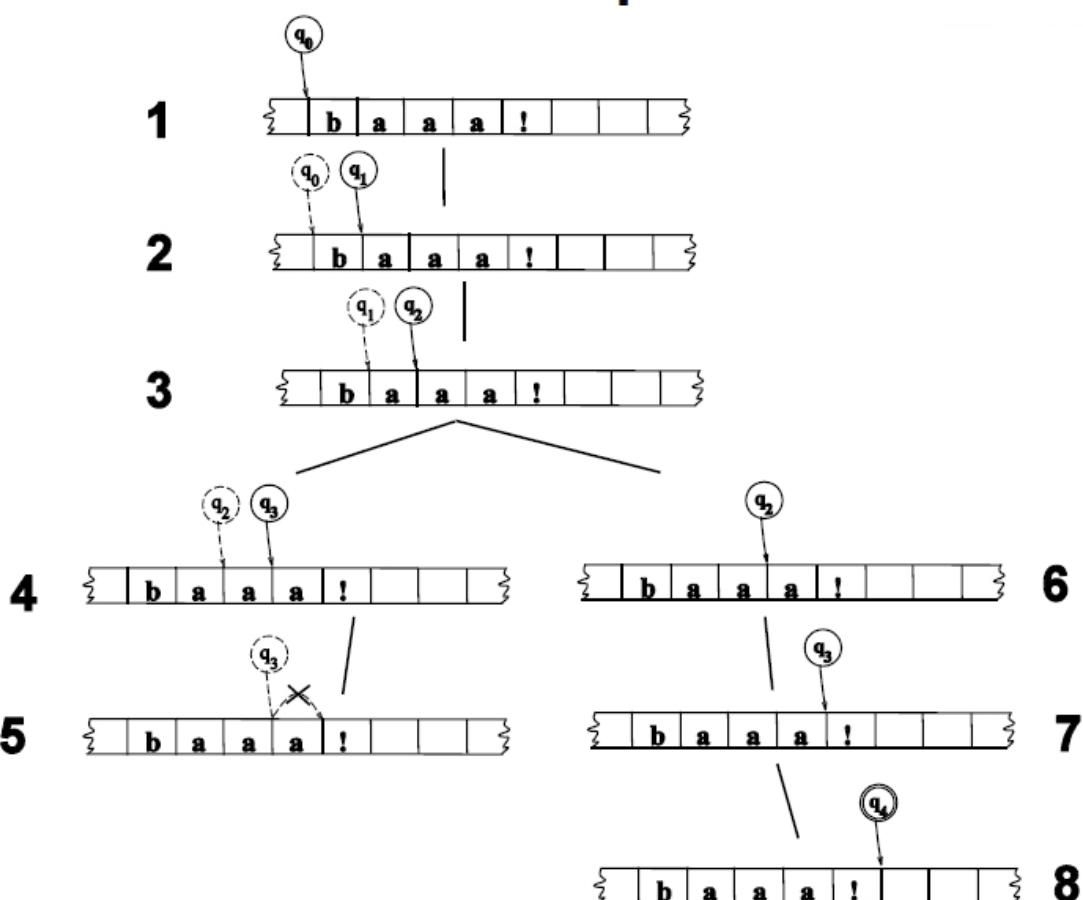
不确定性 FSA:

PPT week1 p53 用 ϵ 转移，在识别过程中，这些转换不会检查或推进 tap?

可以被等价地转换为确定性的 FSA，表明 D-FSA 和 ND-FSA 一样强

识别：

- (1) 把 ND 变成 D 然后用 D-FSA 的算法
- (2) 直接用 ND
一样按 tap 走，不确定的地方就作为根节点 DFS 搜索下去



分词

要记的：

parts of speech: 词类

词元 lemma: 包括词干 词性 词义

cat N. animal

词形 wordform: cats

词型 type: 词汇中的一个元素 = V

词例 token: 词型一样的 instance 数目 = N

type: they /lay/ back/ on /the/ San Francisco/ grass /and /looked /at /the /stars/ and /their/ 14-1(the)-1(and) = 12

token : they /lay /back/ on/ the /San Francisco /grass /and /looked/ at /the/ stars/ and /their 14

$$|V| > O(N^{\frac{1}{2}})$$

步骤：

把非字母数字的符号都变成换行
按照字母表排序
整合大小写的 Case 并且计算每种 Type 的数目

这样操作会有各种问题

中文的最大匹配分词算法

给定一个词典，往后找最长的且能和词典中匹配到的文字，循环

英文不一定会能 work

The table down there	the table down there
	theta bled own there

分句(in week2):

Problem:

- (1) 不能直接按标点： P.H.D jw.ustc.edu.cn
- (2) 有附着的词素： I'am
- (3) Multi-token words: New York
- (4) !,? 相对来说较无歧义

解决方案：用 hand-written rules 或者机器学习来对一个 . 判断是不是句尾。

WEEK 2

□ Morphology (形态学)

□ Edit Distance (编辑距离)

□ Language Model (语言模型)

形态学：词怎么造出来

要记的：

词目(lemmas)
词素：词语构建的最小单元
词干(stem): 词的主要部分
词缀：依附在词干上的部分

形态学包括屈折(Inflectional)和派生(Derivational)

屈折：不改变词类: walk→walking

派生：会改变词类: computer→computerization

词干还原(Stemming):

Porter Stemmer

思路：只关心词干，还原后可能不合法，但对 IR 影响不大

方法：按一定的规则，直接把后缀扔掉

例子：

- Computerization

- ization -> -ize computerize
- ize -> ε computer

编辑距离

将两个对齐的字符串从一个变成另一个所需的带权操作次数

操作：插入、删除、代替

搜索求解：

状态：字符串 X 目前的构成 X'

初状态：X； 末状态：Y

路径损失：操作的累计

DFS 求解(不带重复状态)----减小时间复杂度

不需要记录字符串，而只记录操作过程---减小空间复杂度

动态规划求解：

$$D[i, j] = \text{Dis}(X[1, \dots, i], Y[1, \dots, j])$$

Base conditions:

$$D(i, 0) = i \quad D(0, j) = j$$

Termination:

$$D(N, M) \text{ is distance}$$

Recurrence Relation:

For each $i = 1..M$

For each $j = 1..N$

$$D(i, j) = \min \begin{cases} D(i-1, j) + 1 & \text{deletion} \\ D(i, j-1) + 1 & \text{insertion} \\ D(i-1, j-1) + 2; & \begin{cases} \text{if } X(i) \neq Y(j) & \text{substitution} \\ 0; & \text{if } X(i) = Y(j) \end{cases} \end{cases}$$

$$\text{ptr}(i, j) = \begin{cases} \text{LEFT} & \text{insertion} \\ \text{DOWN} & \text{deletion} \\ \text{DIAG} & \text{substitution} \end{cases}$$

距离矩阵：

N	9										
O	8										
I	7										
T	6										
N	5										
E	4										
T	3										
N	2										
I	1										
#	0	1	2	3	4	5	6	7	8	9	
	#	E	X	E	C	U	T	I	O	N	

$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$

n	9	↓ 8	↙ ↘ 9	↙ ↘ 10	↙ ↘ 11	↙ ↘ 12	↓ 11	↓ 10	↓ 9	↙ 8	
o	8	↓ 7	↙ ↘ 8	↙ ↘ 9	↙ ↘ 10	↙ ↘ 11	↓ 10	↓ 9	↙ 8	← 9	
i	7	↓ 6	↙ ↘ 7	↙ ↘ 8	↙ ↘ 9	↙ ↘ 10	↓ 9	↙ 8	← 9	← 10	
t	6	↓ 5	↙ ↘ 6	↙ ↘ 7	↙ ↘ 8	↙ ↘ 9	↙ 8	← 9	← 10	← 11	
n	5	↓ 4	↙ ↘ 5	↙ ↘ 6	↙ ↘ 7	↙ ↘ 8	↙ ↘ 9	↙ ↘ 10	↙ ↘ 11	↙ ↘ 10	
e	4	↙ 3	← 4	↙ ↘ 5	← 6	← 7	← 8	↙ ↘ 9	↙ ↘ 10	↓ 9	
t	3	↙ ↘ 4	↙ ↘ 5	↙ ↘ 6	↙ ↘ 7	↙ ↘ 8	↙ 7	← 8	↙ ↘ 9	↓ 8	
n	2	↙ ↘ 3	↙ ↘ 4	↙ ↘ 5	↙ ↘ 6	↙ ↘ 7	↙ ↘ 8	↓ 7	↙ ↘ 8	↙ 7	
i	1	↙ ↘ 2	↙ ↘ 3	↙ ↘ 4	↙ ↘ 5	↙ ↘ 6	↙ ↘ 7	↙ 6	← 7	← 8	
#	0	1	2	3	4	5	6	7	8	9	
	#	e	x	e	c	u	t	i	o	n	

回溯：

intention → execution

(i=5,j=5) 由 8→6: substitution

intention → intention

(i=4,j=4) 由 6→5: left 表明 insert

intention → intention

(i=2,j=3) 由 5→3: substitution

intention → intention

(i=1,j=2) 由 3→1: substitution

intention → intention

(i=0,j=1) 由 1→0: down 表明 delete

intention → intention

每一条从 (0,0) 到 (m, n) 的非递减路径都代表一个两个序列之间的对齐方案

复杂度：时间 O(mn); 空间 O(mn); 回溯 O(n+m)

加权：给不同的操作不同的“惩罚”

Initialization:

$$\begin{aligned} D(0,0) &= 0 \\ D(i,0) &= D(i-1,0) + \text{del}[x(i)]; \quad 1 < i \leq N \\ D(0,j) &= D(0,j-1) + \text{ins}[y(j)]; \quad 1 < j \leq M \end{aligned}$$

Recurrence Relation:

$$D(i,j) = \min \begin{cases} D(i-1,j) + \text{del}[x(i)] \\ D(i,j-1) + \text{ins}[y(j)] \\ D(i-1,j-1) + \text{sub}[x(i), y(j)] \end{cases}$$

Termination:

$D(N,M)$ is distance

语言模型 : $P(W_n|W_1, \dots, W_{n-1})$

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\dots P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^n P(w_k|w_1^{k-1}) \end{aligned}$$

$w_i^j = w_i, \dots, w_j$,

$$\begin{aligned} P(\text{you} | \text{the river is so wide that}) \\ = \\ \text{Count}(\text{the river is so wide that you}) \end{aligned}$$

用频率估计概率: $\frac{\text{Count}(\text{the river is so wide that})}{\text{Count}(\text{the river is so wide that})}$

Markov Assumption:

$$P(w_n | w_1^{n-1}) \approx P(w_n | w_{n-N+1}^{n-1}) \quad P(w_n | w_1^{n-1}) \approx P(w_n | w_{n-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

Open/ Closed vocabulary tasks: 事先不知道/知道所有词汇

Out of Vocabulary = OOV words

PPT week2 P63?

Instead: create an unknown word token <UNK>

- Training of <UNK> probabilities
 - Create a fixed lexicon L of size V
 - At text normalization phase, any training word not in L changed to <UNK>
 - Now we train its probabilities like a normal word
- At decoding time
 - If text input: Use UNK probabilities for any word not in training

外在测评: 模型 A 与模型 B 的 word error rate 做外在评测

内在测评 (Perplexity):

$$\text{PP}(W) = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}}$$

最小化困惑度和最大化概率是一样的

问题:

① 过拟合，测试集和训练集相差很大

② 出现概率=0：

平滑：

$$(1) \text{ Laplace} \quad \text{平} \quad \text{滑}, \quad P_{\text{Laplace}}(w_i) = \frac{c_i + 1}{N + V}$$

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{C(w_{n-1}) + V}$$

$$\text{等价于} \quad c^*(w_{n-1}w_n) = \frac{[C(w_{n-1}w_n) + 1] \times C(w_{n-1})}{C(w_{n-1}) + V}$$

(2) Good-Turing 平滑法， N_x 为出现 x 次的词的个数

$$c_x^* = (c_x + 1) \frac{N_{x+1}}{N_x}$$

回退： $P(z|xy)$ 因为 $c(xyz)=0$ 算不出来，回退到 $p(z|y)$ 或者 $p(z)$

内插：将不同阶输出结果线性插值，权重可以和前面的词相关；采用搜索算法找到最优权重（比如 EM 算法）

$$\begin{aligned} \hat{P}(w_n|w_{n-2}w_{n-1}) &= \lambda_1(w_{n-2}^{n-1})P(w_n|w_{n-2}w_{n-1}) \\ &\quad + \lambda_2(w_{n-2}^{n-1})P(w_n|w_{n-1}) \\ &\quad + \lambda_3(w_{n-2}^{n-1})P(w_n) \end{aligned}$$

Week2 PPT p89 没看懂

用 $\exp\{\Sigma \log p_i\}$ 来算 $p_1..p_n$ ：避免下溢+加快速度

N-gram 优点：容易构建、训练量大；平滑使能泛化到新的数据

缺点：test set 和 training set 相似时效果才好；只捕捉到短语境

神经网络：one-hot 编码，可以做非线性预测，BP+SGD

词嵌入向量可以表征语义相似度，但训练消耗大

WEEK 3

□Part of speech tagging (词类标注)

- Parts of speech (词类)
- Tag sets (词类标记集)
- Rule-based tagging (self-study)
- Statistical tagging
 - Simple most-frequent-tag baseline
- HMM (隐马尔科夫模型) tagging

词类

Noun 名词 , verb 动词 , adjective 形容词 , adverb 副词 , preposition 介词 , pronoun 代词 , conjunction 连接词 , determiner 限定词 , etc

案例:

			WORD	tag
N	noun	<i>chair, bandwidth, pacing</i>		
V	verb	<i>study, debate, munch</i>	the	DET
ADJ	adjective	<i>purple, tall, ridiculous</i>	koala	N
ADV	adverb	<i>unfortunately, slowly</i>	put	V
P	preposition	<i>of, by, to</i>	the	DET
PRO	pronoun	<i>I, me, mine</i>	keys	N
DET	determiner	<i>the, a, that, those</i>	on	P
			the	DET
			table	N

一个词可以有多个 Tags

- The **back**door = JJ (adj)
- On my **back** = NN
- Win the voters **back** = RB (adverb)
- Promised to **back** the bill = VB (base verb)

POS tagging: 给出现在某个句子中的特定词贴标签

(1) Rule-based:

选最常见的类、查表、正则匹配

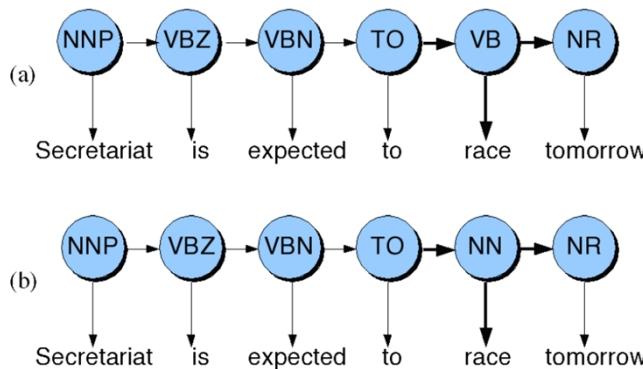
(2) HMM:

Tags for sentence:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n) = \underset{t_1^n}{\operatorname{argmax}} P(w_1^n | t_1^n)P(t_1^n) \approx \underset{t_1^n}{\operatorname{argmax}} \prod_{i=1}^n P(w_i | t_i)P(t_i | t_{i-1})$$

乘积中的每一项用 MLE 计算

Tag for word:



$$P(VB|TO)P(NR|VB)P(race|VB) = .00000027$$

$$P(NN|TO)P(NR|NN)P(race|NN) = .00000000032$$

马尔科夫链: 状态+初始概率+转移概率+马尔科夫性

HMM 定义: 状态+初始概率+转移概率 A+观测值+输出概率 B+ 马尔科夫性.
为生成模型。

基本问题:

(1) (评估或计算得分问题) 如何计算给定观察序列出现的概率?

FW 算法:

$$\alpha_t(i) = P(\mathbf{o}_1 \mathbf{o}_2 \dots \mathbf{o}_t, q_t = S_i | \lambda)$$

Inductively solve for $\alpha_t(i)$ as:

1. Initialization

$$\alpha_1(i) = \pi_i b_i(\mathbf{o}_1), \quad 1 \leq i \leq N$$

2. Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(\mathbf{o}_{t+1}), \quad 1 \leq t \leq T-1, 1 \leq j \leq N$$

3. Termination

$$P(\mathbf{O} | \lambda) = \sum_{i=1}^N P(\mathbf{o}_1 \mathbf{o}_2 \dots \mathbf{o}_T, q_T = S_i | \lambda) = \sum_{i=1}^N \alpha_T(i)$$

Computation: $N^2 T$ versus $2TN^T$; $N = 5, T = 100 \Rightarrow 2500$ versus 10^{72}

BW 算法:

Consider the backward variable, $\beta_t(i)$, defined as the probability of the partial observation sequence from $t+1$ to the end, given state S_i at time t , and the model, i.e.,

$$\beta_t(i) = P(\mathbf{o}_{t+1} \mathbf{o}_{t+2} \dots \mathbf{o}_T | q_t = S_i, \lambda)$$

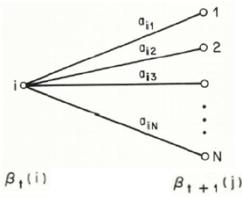
Inductive Solution

1. Initialization

$$\beta_T(i) = 1, \quad 1 \leq i \leq N$$

2. Induction

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j), \quad t = T-1, T-2, \dots, 1, 1 \leq i \leq N$$



$N^2 T$ calculations, same as in forward case

(2) (解码问题) 给定观察序列, 如何计算最优的隐状态序列?

$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} P[q_1 q_2 \dots q_{t-1}, q_t = i, \mathbf{o}_1 \mathbf{o}_2 \dots \mathbf{o}_t | \lambda]$$

Step 1 - Initialization

$$\delta_1(i) = \pi_i b_i(\mathbf{o}_1), \quad 1 \leq i \leq N$$

$$\psi_1(i) = 0, \quad 1 \leq i \leq N$$

Step 2 - Recursion

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(\mathbf{o}_t), \quad 2 \leq t \leq T, 1 \leq j \leq N$$

$$\psi_t(j) = \operatorname{argmax}_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}], \quad 2 \leq t \leq T, 1 \leq j \leq N$$

Step 3 - Termination

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)]$$

$$q_T^* = \operatorname{argmax}_{1 \leq i \leq N} [\delta_T(i)]$$

Step 4 - Path (State Sequence) Backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T-1, T-2, \dots, 1$$

Calculation $\approx N^2 T$ operations ($*, +$)

为了加速也可以进行对数化。

$$\begin{aligned}\tilde{\pi}_i &= \log(\pi_i) & 1 \leq i \leq N \\ \tilde{b}_i(o_t) &= \log[b_i(o_t)] & 1 \leq i \leq N, 1 \leq t \leq T \\ \tilde{a}_{ij} &= \log[a_{ij}] & 1 \leq i, j \leq N\end{aligned}$$

(3) (训练问题) 如何调整模型参数来最大化某特定观察序列的概率?

无全局最优解、可以用梯度类或 EM 算法求解

Re-estimation:

$$\lambda^{(0)} \xrightarrow{E[\text{Model Events}]} \lambda^{(1)} \xrightarrow{E[\text{Model Events}]} \lambda^{(2)} \dots$$

Define $\xi_t(i, j)$, the probability of being in state S_i at time t , and state S_j at time $t+1$, given the model and the observation sequence, i.e.,

$$\xi_t(i, j) = P[q_t = S_i, q_{t+1} = S_j | \mathcal{O}, \lambda]$$

$$\gamma_t(i) = \sum_{j=1}^N \xi_t(i, j) \quad \begin{array}{l} \sum_{t=1}^{T-1} \gamma_t(i) = \text{Expected number of transitions from } S_i \\ \sum_{t=1}^{T-1} \xi_t(i, j) = \text{Expected number of transitions from } S_i \text{ to } S_j \end{array}$$

更新参数:

$$\bar{\pi}_i = \text{Expected number of times in state } S_i \text{ at } t=1 \\ = \gamma_1(i)$$

$$\bar{a}_{ij} = \frac{\text{Expected number of transitions from state } S_i \text{ to state } S_j}{\text{Expected number of transitions from state } S_i}$$

$$= \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\bar{b}_j(k) = \frac{\text{Expected number of times in state } j \text{ with symbol } v_k}{\text{Expected number of times in state } j}$$

$$= \frac{\sum_{\substack{t=1 \text{ s.t. } O_t=v_k}}^T \gamma_t(j)}{\sum_{t=1}^T \gamma_t(j)}$$

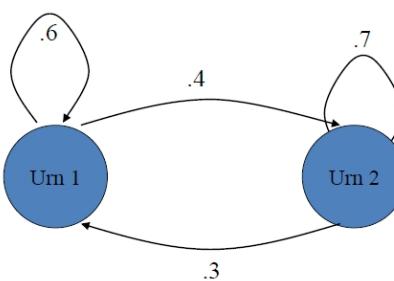
实际操作: 对于不认识的词语可以利用形态学知识给其贴标签; -able=JJ

例子:

Pi: Urn 1: 0.9; Urn 2: 0.1

A	Urn 1	Urn 2
Urn 1	0.6	0.4
Urn 2	0.3	0.7

B	Urn 1	Urn 2
Red	0.7	0.4
Blue	0.3	0.6



计算要注意啊

Forward: $P(BBR | \text{model}) = .0792$

\sum

1 1 1	$(0.9*0.3)*(0.6*0.3)*(0.6*0.7) = 0.0204$
1 1 2	$(0.9*0.3)*(0.6*0.3)*(0.4*0.4) = 0.0077$
1 2 1	$(0.9*0.3)*(0.4*0.6)*(0.3*0.7) = 0.0136$
1 2 2	$(0.9*0.3)*(0.4*0.6)*(0.7*0.4) = 0.0181$
2 1 1	$(0.1*0.6)*(0.3*0.7)*(0.6*0.7) = 0.0052$
2 1 2	$(0.1*0.6)*(0.3*0.7)*(0.4*0.4) = 0.0020$
2 2 1	$(0.1*0.6)*(0.7*0.6)*(0.3*0.7) = 0.0052$
2 2 2	$(0.1*0.6)*(0.7*0.6)*(0.7*0.4) = 0.0070$

FW 的值就是都加起来

若参数不准确，则 Re-estimate。如要更新 $1 \rightarrow 2$ 的概率和 $1 \rightarrow 1$ 的概率，则根据上面的概率值求 $1 \rightarrow 2$ 与 $1 \rightarrow 1$ 的加权频率(两个一起作 Normalize)，作为更新后的概率

$$1 \rightarrow 2 \quad (2 * .0204) + (1 * .0077) + (1 * .0052) = .0537$$

$$1 \rightarrow 1 \quad (.0077 * 1) + (.0136 * 1) + (.0181 * 1) + (.0020 * 1) = .0414$$

The $1 \rightarrow 2$ transition probability is
 $.0414 / (.0414 + .0537) = 0.435$

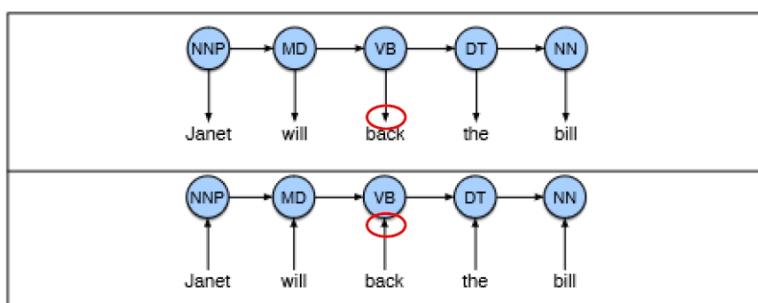
The $1 \rightarrow 1$ transition probability is
 $.0537 / (.0414 + .0537) = 0.565$

So in re-estimation the $1 \rightarrow 2$ transition went from .4 to .435 and the $1 \rightarrow 1$ transition went from .6 to .565

MEMM(判别模型):

$$p(c|x) = \frac{1}{Z} \exp\left(\sum_i w_i f_i\right)$$

一般采用布尔特征，如句子长度是否大于 5，是否有数字与 HMM 的区别：



测评：人类分析准确率大概 96-97% 作为黄金标准。

混淆矩阵：横向是 Actual Class, 纵向是 Predict

	IN	JJ	NN	NNP	RB	VBD	VBN
IN	-	.2			.7		
JJ	.2	-	3.3	2.1	1.7	.2	2.7
NN		8.7	-				.2
NNP	.2	3.3	4.1	-	.2		
RB	2.2	2.0	.5		-		
VBD		.3	.5			-	4.4
VBN		2.8			2.6		-

中文更具歧义性，不认识的词语更多，神经网络效果最好。

WEEK 4

□ Syntax (句法) and Grammer (语法)

– CFG, Dependency Grammer, TreeBanks

□ Parsing (剖析)

– CKY Parsing Algorithm

□ Partial Parsing

– Rule-Based / Machine Learning-Based

Syntax and Grammer(句法和语法)

常识：

Sentences 句子

Noun phrases 名词短语

Verb phrases 动词短语

Prepositional phrases 介词短语

句子具有完整性；子句里包含一个动词；短语比较随意，不一定有主体和动词

Sentence-Types

Declaratives 陈述式: A plane left

$S \rightarrow NP VP$

Imperatives 命令式: Leave!

$S \rightarrow VP$

Yes-No Questions: Did the plane leave?

$S \rightarrow Aux NP VP$

WH Questions: When did the plane leave?

$S \rightarrow WH Aux NP VP$

$S \rightarrow NP VP$	$Det \rightarrow that this a$
$S \rightarrow Aux NP VP$	$Noun \rightarrow book flight meal money$
$S \rightarrow VP$	$Verb \rightarrow book include prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I she me$
$NP \rightarrow Proper-Noun$	$Proper-Noun \rightarrow Houston TWA$
$NP \rightarrow Det Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from to on near through$
$Nominal \rightarrow Nominal Noun$	
$Nominal \rightarrow Nominal PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb NP$	
$VP \rightarrow Verb NP PP$	
$VP \rightarrow Verb PP$	
$VP \rightarrow VP PP$	
$PP \rightarrow Preposition NP$	

Syntax(句法): 词语怎么被组织在一起的

Context-Free Grammer: 规则(产生式), 具有顺序性(ordering)和组成性(constituency)

规则: $A \rightarrow B C$

Context: 只要看到 A 就能写出 B, C; 看到 B C 就能推断出 A: A 是独立存在的。

<https://zhuanlan.zhihu.com/p/56011904>

产生式左侧的符号 (语句、主语、谓语和宾语) 称为 非终结符 (nonterminal) , 代表可以继续扩展或产生的符号, 也就是说, 当在某条产生式的右边遇到了此非终结符的时候, 总是可以用本产生式的右边来替换这个非终结符。

而 “我、你、他、吃、饭、菜” 这些符号是 X 语言中的词, 无法再产生新的符号了, 称为 终结符 (terminal) 。终结符只能出现在产生式的右边, 非终结符则左边和右边都可以出现。

上述产生式中有一个特别的非终结符: “语句”, X 语言中的所有句子都以它为起点产生, 这个符号被称为 起始符号 (start symbol)。

一个上下文无关语法 G 就是由一个终结符集合 T, 一个非终结符集合 N (N 和 T 不相交), 一个产生式集合 P, 以及一个起始符号 S ($S \in N$) 组成。由语法 G 推导 (产生) 出来的所有的句子的集合称为 G 语言。因此一个语法可以代表一个句子集合, 也就是一个语言。

案例:

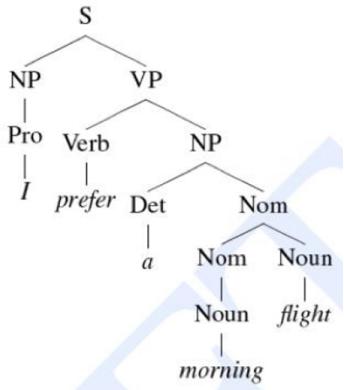
```

S -> NP VP
NP -> Det NOMINAL
NOMINAL -> Noun
VP -> Verb

```

Generativity(生成性): 可以生成、判断 in-Language strings, 产生树结构

Derivations(推导): 是否有一系列的规则能被应用到这个字符串中



Recusion(递归): 左边的非终结符又出现在右边的规则

用 CFG 解析文法的问题:

(1) 一致关系

(名词代词和动词单复数形式应该对应, 可以细化语法产生式, 但泛化能力不够强)

(2) 次范畴化

谓语和后边的“参数”类型应该满足一些语义上的约束:

对谓语 (基本是动词) 进行次范畴化, 约束框架称为次范畴化框架, 增加语法规则。

Overgenerate problem: 规则是对的, 语义上不对

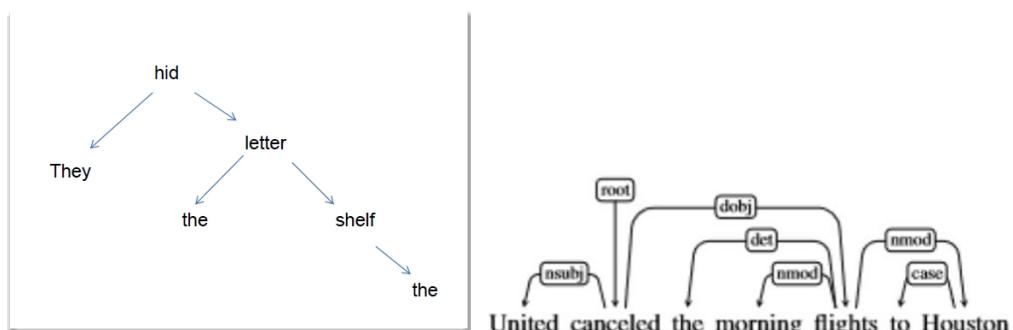
(3) 移位

倒装等特殊句式使动词和宾语的位置发生变化, 难以使用 CFG。

(Dependency Grammars)依存语法:

记录词之间的论旨角色(thematic roles: links), 一般是二元的; 没有非终结符。

(Dependency Parsing)依赖解析: link 可以有标签(只关心词之间的关系)



语法库的来源:

(1) 手工构造

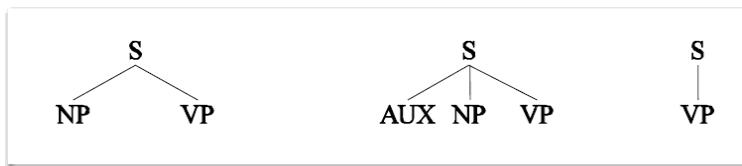
(2) TreeBanks, 根据 POS 过的句子自动生成语法, 自动解析, 手工修正。

```
((S
  (NP-SBJ (DT That)
    (JJ cold) (, ,)
    (JJ empty) (NN sky) )
  (VP (VBD was)
    (ADJP-PRD (JJ full)
      (PP (IN of)
        (NP (NN fire)
          (CC and)
          (NN light) ))))
  (. .) )))
```

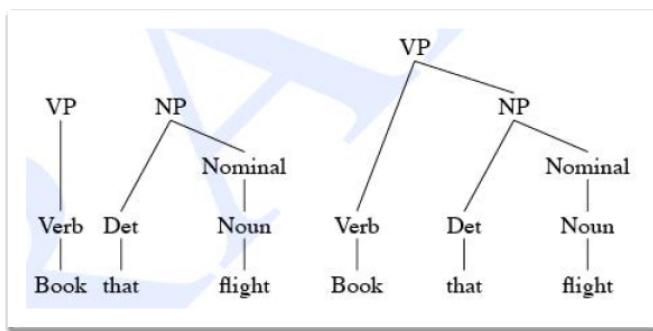
Parsing(解析): String + Grammar assign → **Correct** Parse Tree

Correct: (a) S 为根 (b) 覆盖并仅覆盖输入的所有元素

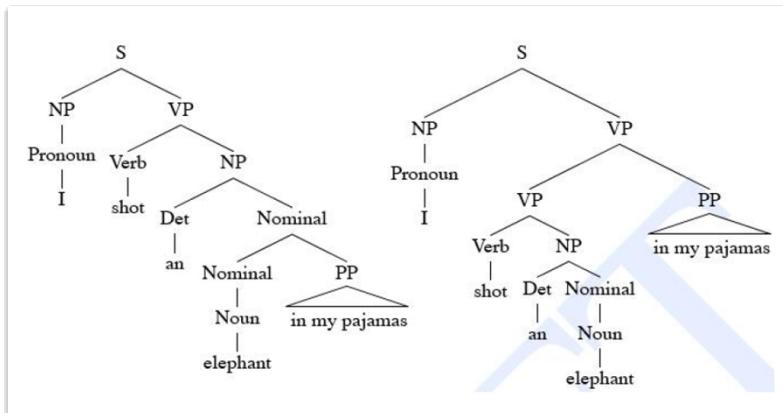
Top-Down(自顶向下解析): 满足(a), 但可能没有覆盖输入的 words



Bottom-Up(自底向上解析): 满足(b), 但可能不是要的 sentence



歧义性(Ambiguity):



动态规划解析(Dynamic Programming Parsing):

不做重复工作, 多项式时间求解, **有效存储二义句子?**

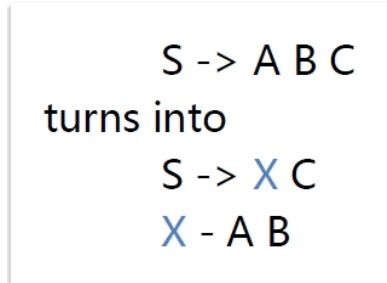
时间复杂度: CKY Bottom-Up O(n^3); Earley Top-Down O (n^3)

CKY Parsing:

(1) Chomsky 范式(CNF): $A \rightarrow BC$ 或 $A \rightarrow w$;

B、C 非终结符, w 是终结符。

不是 Chomsky 范式的文法可以重写成 CNF, 并且 Binarization。



要求 X 没有出现在其它地方。

(2) 算法:

索引 0, 1, ..., n 分别代表第一个单词前, 第二个前, ..., 第 n 个单词后;

计算 $\text{table}[i, j]$ 时穷举从 $i+1$ 到 $j-1$ 这些可能分隔, 并且检测其是否是有效的 Chomsky 范式, $\text{table}[i, j]$ 中填写可能的非终结符, 整个句子的终结符即为 $\text{table}[0, n]$ 。

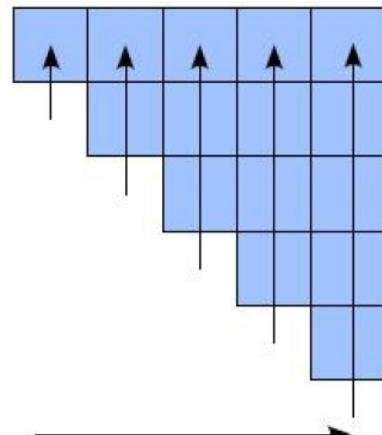
时间复杂度: $O(n^3)$

```
function CKY-PARSE(words, grammar) returns table
  for j ← from 1 to LENGTH(words) do   Iterate over columns
    table[j − 1, j] ← {A | A → wj ∈ grammar}
    for i ← from j − 2 down to 0 do   Iterate over rows (from bottom up)
      for k ← i + 1 to j − 1 do   Iterate over all possible splits
        table[i, j] ← table[i, j] ∪
          {A | A → BC ∈ grammar,
           B ∈ table[i, k],
           C ∈ table[k, j] }
```

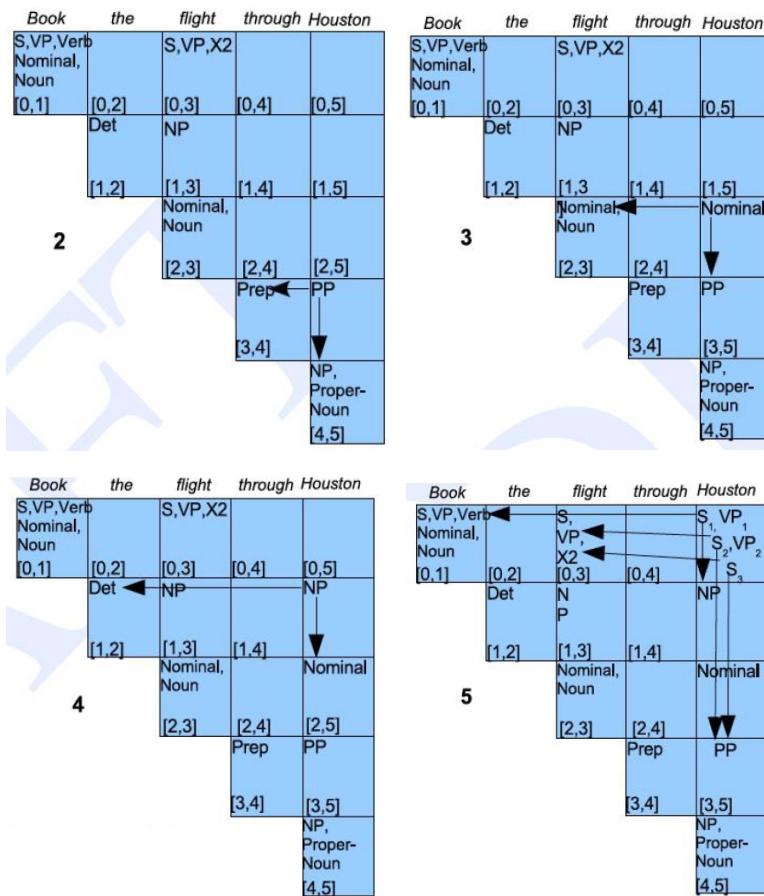
Book	the	flight	through	Houston
S,VP,Verb Nominal, Noun [0,1]		S,VP,X2 [0,2]		S, VP [0,5]
	Det [1,2]	NP [1,3]		NP [1,5]
		Nominal, Noun [2,3]	[1,4]	Nominal [2,5]
			Prep [3,4]	PP [3,5]
				NP, Proper- Noun [4,5]

i

j



(3) 例子:



缺陷: 可能产生一些无关的成分 week4 PPT p81 没懂啥意思

无法解决 Ambiguity 的问题, 即存在在 Table[0,n] 有多个 S 结构

- They both efficiently store the sub-parts that are shared between multiple parses.
- And they obviously avoid re-deriving those sub-parts.
- But neither can tell us which one is right.

Partial Parsing(部分解析):只处理大的组块 (chunk) 在哪;

Full: [NP The morning flight] [PP from] [NP Denver] [VP has arrived.]

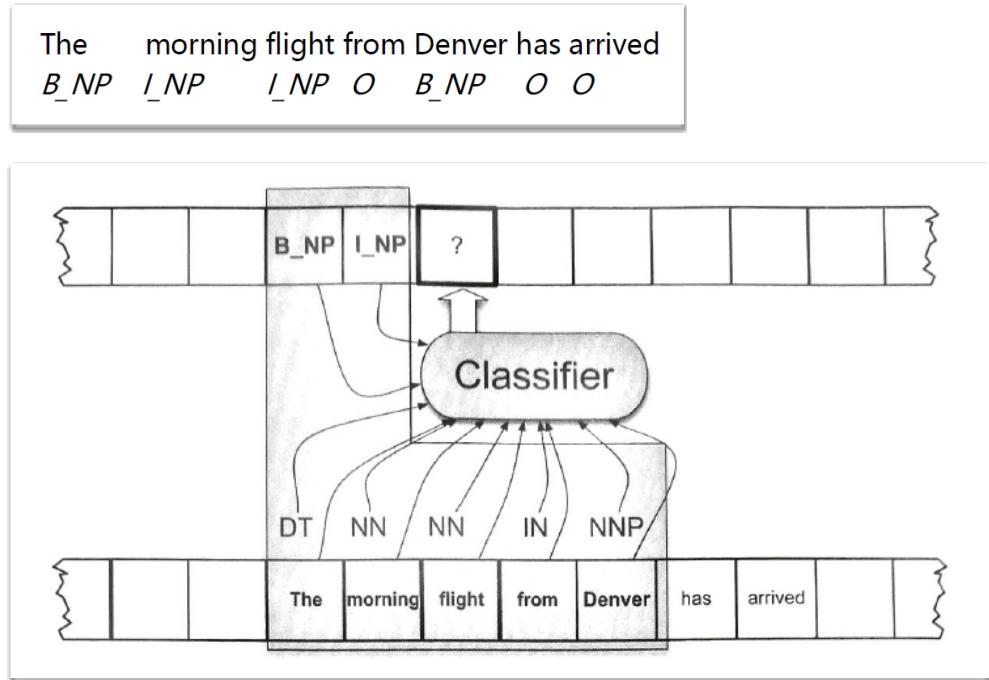
Partial: [NP The morning flight] from [NP Denver] has arrived.

(2) 基于规则: 去掉规则集中的递归元素 (变成正则文法);

- Phase 1: Part of speech tags
- Phase 2: Base syntactic phrases
- Phase 3: Larger verb and noun groups
- Phase 4: Sentential level rules

week 4 PPT p89-92 没太懂

(3) ML-Based Chunking: 一个序列分类任务，其把词分为三类“开始 B，内部 I，外部 O”，外部表示其不在任何组块中，内部表示其在某组块中 (I_NP 在 NP 中)，开始表示其为某块的开始 (B_NP)



WEEK 5

□ Probabilistic CFGs

□ Structural Dependencies

□ Parser Evaluation

(PCFG)Probabilistic CFGs:

给语法规则赋概率，用来解决二义性，同时模仿人类解析语言的过程

VP -> Verb	.55
VP -> Verb NP	.40
VP -> Verb NP NP	.05
– Read this as P(Specific rule LHS)	
• $P(VP \rightarrow \text{Verb} VP) = .55$	

$$P(T, S) = \prod_{node \in T} P(rule(n))$$

Sentence 的概率：非二义的情况下就是该语法树的概率，二义的情况为 trees 概率和。

$$P(\alpha \rightarrow \beta | \alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\sum_{\gamma} \text{Count}(\alpha \rightarrow \gamma)} = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)}$$

from tree bank

若没有 tree bank, 用含概率的 grammar 解析一个大的语料库, 一开始随机化 rule 的概率, 之后采用 re-estimate。

If we don't have a treebank, but we do have a grammar can we get reasonable probabilities?

Yes. Use a prob parser to parse a large corpus and then get the counts as above.

- In the unambiguous case we're fine
- In ambiguous cases, weight the counts of the rules by the probabilities of the trees they occur in.
- Where do those probabilities come from?
- Make them up. And then re-estimate them.

Prob CKY:

一次应用规则的概率： $P(A \rightarrow BC) * P(B) * P(C)$, DP 时 $P(B)$ 和 $P(C)$ 都已经计算出来了；如果有种 B 和 C (来源于不同的分点, 但是都合法) 则取最大概率的那种。

```
function PROBABILISTIC-CKY(words,grammar) returns most probable parse
    and its probability
    for j ← from 1 to LENGTH(words) do
        for all { A | A → words[j] ∈ grammar }
            table[j-1,j,A] ← P(A → words[j])
        for i ← from j-2 downto 0 do
            for k ← i+1 to j-1 do
                for all { A | A → BC ∈ grammar,
                           and table[i,k,B] > 0 and table[k,j,C] > 0 }
                    if (table[i,j,A] < P(A → BC) × table[i,k,B] × table[k,j,C]) then
                        table[i,j,A] ← P(A → BC) × table[i,k,B] × table[k,j,C]
                        back[i,j,A] ← {k,B,C}
    return BUILD-TREE(back[1, LENGTH(words), S]), table[1, LENGTH(words), S]
```

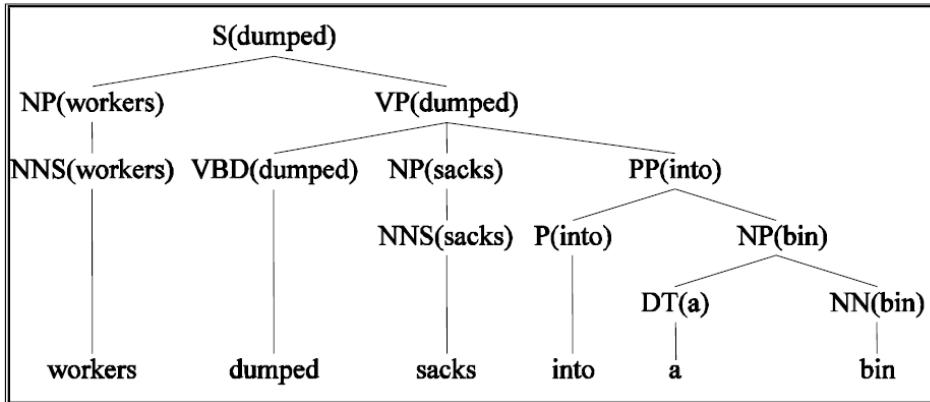
问题：没有考虑到推导环境的上下文（只基于很小的上下文得到一个概率），导致最优推导和 TreeBank 中的实际推导存在很大差异。

Structural Dependencies:

(1) **父节点标注：** 将非终结符一分为 n, 每个都标上其父节点的信息 ($NP \rightarrow NP_{subject}$), 但是增加了语法复杂性, 并且需要更多数据;

(2) **中心词** ($NP \rightarrow$ 名词, $VP \rightarrow$ 动词, $PP \rightarrow$ 介词) 增加 lexical dependencies

语法符号: $VP(\text{中心词})$



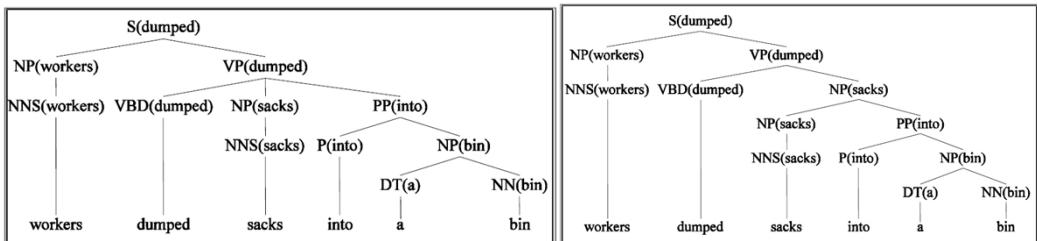
原本: $P(r|VP) = \text{Count}(\text{这个规则用在 } VP \text{ 上}) / \text{Count}(VP \text{ 出现次数})$

现在: $P(r|VP^{\text{dumped}}) = \text{Count}(\text{这个规则用在 dump 上}) / \text{Count}(VP \text{ 中 dump 作为中心词出现次数})$

这其实是一种次范畴化: 捕获 VPhead(动词)和它们所伴随的 VP 规则之间的关联性。;

优先级: 次范畴化是在描述 VP heads 和语法之间的选择亲和性

还应加入 VP heads 和 VP 其他孩子的 heads 之间的亲和性



看 dump 是 head, PP(into)是他的兄弟 v.s sack 是头和 PP(into)是他的兄弟的次数

问题: 句子太少, 规则不变

Parser 评估:

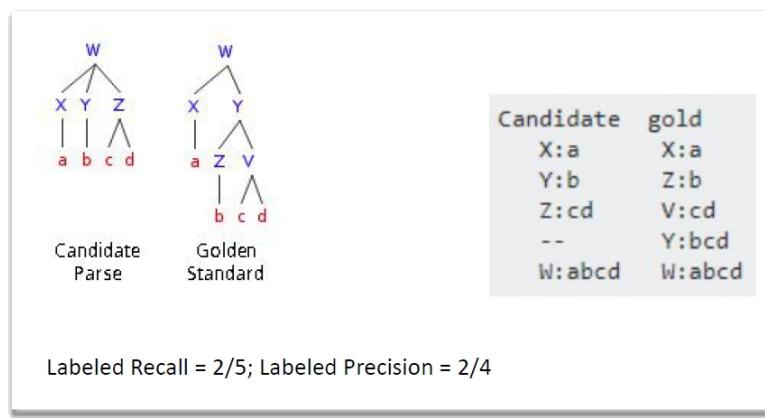
(1) 构成评估 (Constituent-level Evaluation)

a. 覆盖率 (Recall)

正确 (即有相同节点标签和正确分点的节点) 的节点数 / Treebank 中的相应 constituent 节点数

b. 精准度 (Precision)

正确的节点数 / 解析结果中的节点数



(2) 交叉括号 (Cross Brackets)

Treebank has ((X Y) Z) and candidate has (X (Y Z)) ;
作为客观的函数来最小化这种情形。

Parseval 缺点：

会更偏向“安全的，浅的解析”；
部分错误可能不停向上传播，导致很多交叉括号的情况；
将所有节点一视同仁，而不是更关注核心的语义关系

Dependency Parsing Metrics:

Head Attachment Score; Label Precision; Labeled Attachment Score

WEEK 6

□Vector Semantics (语义学)

四种向量语义模型：

- (1) 稀疏向量表示
 - a. 以互信息 (Mutual-information) 为权重的单词关联矩阵
- (2) 稠密向量表示
 - a. 奇异值分解和潜在语义分析 (Latent Semantic Analysis)
 - b. 各种神经网络模型 (Skip-grams, CBOW)
 - c. Brown clusters

词项-文档矩阵 (Term-Document Matrix)

$tf(t,d)$ ：词 t 在 d 文档中出现的次数

两列相近则两个文档相近；两个行相近则两个词相近。

词-词矩阵 (word-word) : 词-上下文 (word-context) 矩阵

a_{ij} 表示在每次单词 i 的 ± 7 (或者别的数) 的范围内出现单词 j 的次数之和
矩阵会比较稀疏，并且窗口大小和目标有关

(窗口越小语法信息越少，窗口越大语义信息越多)

两个单词间的两种互相关 (co-occurrence) :

- (1) 一阶互相关 (syntagmatic 组合)，这些词基本都靠在一起，比如 wrote 和 book
- (2) 二阶互相关 (paradigmatic 聚合)，这些词的邻居相近，比如 wrote 和 said

Raw Counts 的问题：

直接用原始词频考察词-词相关性会有很大偏差

"the" 和 "of" 一起出现的频率非常高，但是词义不一定最贴近

PPMI (点间互信息, Positive Pointwise Mutual Information) : 修正 Raw Counts 的问题

$$PPMI(word_1, word_2) = \max\left(\log_2 \frac{P(word_1, word_2)}{P(word_1)P(word_2)}, 0\right)$$

f_{ij} 定义为 w_i 出现在以 c_j 为上下文的中心词中的频数，

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^W \sum_{j=1}^C f_{ij}}$$

$$p_{ik} = \frac{\sum_{j=1}^C f_{ij}}{\sum_{i=1}^W \sum_{j=1}^C f_{ij}}$$

$$p_{kj} = \frac{\sum_{i=1}^W f_{ij}}{\sum_{i=1}^W \sum_{j=1}^C f_{ij}}$$

$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{ik} p_{kj}}$$

$$ppmi_{ij} = \begin{cases} pmi_{ij} & \text{if } pmi_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

aardvark	computer	data	pinch	result	sugar
apricot	0	0	1	0	1
pineapple	0	0	1	0	1
digital	2	1	0	1	0
information	1	6	0	4	0

Example:

		Count(w,context)					I
		computer	data	pinch	result	sugar	
apricot		0	0	1	0	1	
pineapple		0	0	1	0	1	
digital		2	1	0	1	0	
information		1	6	0	4	0	
		$\sum_{j=1}^C f_{ij}$		$\sum_{i=1}^W f_{ij}$			
p(w=information,c=data)	=	6/19 = .32		$p(w_i) = \frac{\sum_{j=1}^C f_{ij}}{N}$			
p(w=information)	=	11/19 = .58		$p(c_j) = \frac{\sum_{i=1}^W f_{ij}}{N}$			
p(c=data)	=	7/19 = .37					
		p(w,context)					p(w)
		computer	data	pinch	result	sugar	
apricot		0.00	0.00	0.05	0.00	0.05	0.11
pineapple		0.00	0.00	0.05	0.00	0.05	0.11
digital		0.11	0.05	0.00	0.05	0.00	0.21
information		0.05	0.32	0.00	0.21	0.00	0.58
		p(context)	0.16	0.37	0.11	0.26	0.11
		p(w,context)					p(w)
		computer	data	pinch	result	sugar	
apricot		0.00	0.00	0.05	0.00	0.05	0.11
pineapple		0.00	0.00	0.05	0.00	0.05	0.11
digital		0.11	0.05	0.00	0.05	0.00	0.21
information		0.05	0.32	0.00	0.21	0.00	0.58
		p(context)	0.16	0.37	0.11	0.26	0.11

$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{ik} p_{kj}}$$

$$pmi(\text{information}, \text{data}) = \log_2 (.32 / (.37 * .58)) = .57$$

PPMI(w,context)					
	computer	data	pinch	result	sugar
apricot	-	-	2.25	-	2.25
pineapple	-	-	2.25	-	2.25
digital	1.66	0.00	-	0.00	-
information	0.00	0.57	-	0.47	-

Problem: PMI 会向出现较少的事件倾斜

Solution:

- (1) 给稀有事件更高概率: α 为一个预先给定的值 (如 0.75), 此方法可以平滑较大和较小的概率

Raise the context probabilities to $\alpha = 0.75$:

$$\text{PPMI}_\alpha(w, c) = \max(\log_2 \frac{P(w, c)}{P(w)P_\alpha(c)}, 0)$$

$$P_\alpha(c) = \frac{\text{count}(c)^\alpha}{\sum_c \text{count}(c)^\alpha}$$

This helps because $P_\alpha(c) > P(c)$ for rare c

Consider two events, $P(a) = .99$ and $P(b) = .01$

$$P_\alpha(a) = \frac{.99^{.75}}{.99^{.75} + .01^{.75}} = .97 \quad P_\alpha(b) = \frac{.01^{.75}}{.01^{.75} + .01^{.75}} = .03$$

(2) 用加+1(+2)平滑法 (和上面有相似的效果)

Add-2 Smoothed Count(w,context)					
	computer	data	pinch	result	sugar
apricot	2	2	3	2	3
pineapple	2	2	3	2	3
digital	4	3	2	3	2
information	3	8	2	6	2

	p(w,context) [add-2]					p(w)
	computer	data	pinch	result	sugar	
apricot	0.03	0.03	0.05	0.03	0.05	0.20
pineapple	0.03	0.03	0.05	0.03	0.05	0.20
digital	0.07	0.05	0.03	0.05	0.03	0.24
information	0.05	0.14	0.03	0.10	0.03	0.36
p(context)	0.19	0.25	0.17	0.22	0.17	

PPMI(w,context)					
	computer	data	pinch	result	sugar
apricot	-	-	2.25	-	2.25
pineapple	-	-	2.25	-	2.25
digital	1.66	0.00	-	0.00	-
information	0.00	0.57	-	0.47	-

PPMI(w,context) [add-2]					
	computer	data	pinch	result	sugar
apricot	0.00	0.00	0.56	0.00	0.56
pineapple	0.00	0.00	0.56	0.00	0.56
digital	0.62	0.00	0.00	0.00	0.00
information	0.00	0.58	0.00	0.37	0.00

相似性度量：

内积：Vector length 越大 (隐含着越 frequent), 则 dot prod 越大

向量夹角来归一化内积：

$$\cos(\vec{V}, \vec{W}) = \frac{\vec{V} \cdot \vec{W}}{|\vec{V}| |\vec{W}|} = \frac{\vec{V}}{|\vec{V}|} \cdot \frac{\vec{W}}{|\vec{W}|} = \frac{\sum_{i=1}^N V_i W_i}{\sqrt{\sum_{i=1}^N V_i^2} \sqrt{\sum_{i=1}^N W_i^2}}$$

V_i 和 W_i 分别是 V 和 W 在 context i 的 PPMI 值。PPMI 非负, cos 值在 0-1 之间。

其他方法：

$\text{sim}_{\text{cosine}}(\vec{v}, \vec{w})$	$= \frac{\vec{v} \cdot \vec{w}}{ \vec{v} \vec{w} } = \frac{\sum_{i=1}^N v_i \times w_i}{\sqrt{\sum_{i=1}^N v_i^2} \sqrt{\sum_{i=1}^N w_i^2}}$
$\text{sim}_{\text{Jaccard}}(\vec{v}, \vec{w})$	$= \frac{\sum_{i=1}^N \min(v_i, w_i)}{\sum_{i=1}^N \max(v_i, w_i)}$
$\text{sim}_{\text{Dice}}(\vec{v}, \vec{w})$	$= \frac{2 \times \sum_{i=1}^N \min(v_i, w_i)}{\sum_{i=1}^N (v_i + w_i)}$
$\text{sim}_{\text{JS}}(\vec{v} \vec{w})$	$= D(\vec{v} \frac{\vec{v} + \vec{w}}{2}) + D(\vec{w} \frac{\vec{v} + \vec{w}}{2})$

其它定义上下文的方法：

- (1) 通过语法环境, 比如一个名词可以被那些形容词修饰, 或者做哪些动词的宾语;
一个向量用 Relation*|V| 个关系进行修饰, 比如“subject-of, absorb”;
- (2) 另一种方法是将上下文定义成
“counts of words that occur in one of R Dependencies”, 而不是直接用滑动窗口,
比 如 $M(\text{"cell"}, \text{"absorb"}) = \text{count}(\text{subj}(\text{cell}, \text{absorb})) + \text{count}(\text{obj}(\text{cell}, \text{absorb})) + \text{count}(\text{pobj}(\text{cell}, \text{absorb}))$ etc.

不太明白

PMI applied to dependency relations



Hindle, Don. 1990. Noun Classification from Predicate-Argument Structure. ACL

Object of “drink”	Count	PMI
tea	2	11.8
liquid	2	10.5
wine	2	9.3
anything	3	5.2
it	3	1.3

- “Drink it” more common than “drink wine”
- But “wine” is a better “drinkable” thing than “it” ? ?

PPMI 的替代指标:

tf-idf (term-frequency and inverse document frequency)

df_i = 有 word i 的文档总数; N 是文档的总数; tf 可以不计算($=1$).

$$w_{ij} = \text{word } i \text{ in document } j$$

$$w_{ij} = tf_{ij} idf_i$$

$$idf_i = \log\left(\frac{N}{df_i}\right)$$

Similarity Metric:

Extrinsic Evaluation: Question Answering; Spell Checking; Essay grading

Intrinsic Evaluation: 算法与人类的评分比较, TOEFL 词汇考试

稠密向量:

PPMI 方法构造的是稀疏向量

稠密向量更容易学习 weight,

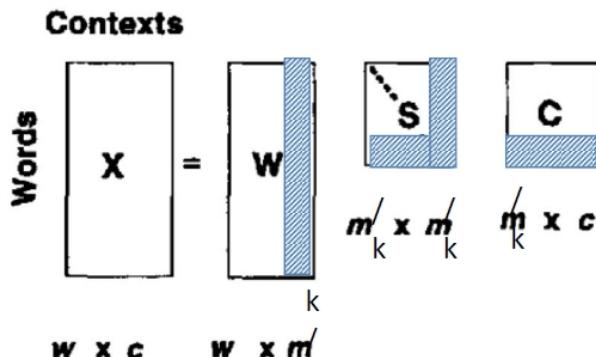
比记录 count 的泛化性能更好,

容易捕获“同义”信息(因为非零维度对应)

构造稠密向量的方法:

(1) **SVD:** 去噪、泛化、易于训练、捕获“同义”信息

压缩后的词项-文档矩阵 W 可以作为一种潜在语义分析



维度: 一般用 300 维, cell 乘以 $\log(tf)*idf$ 或 $\log(tf)*entropy$, 即 local * global

其他: PCA, 成分分析

PPMI Word-Word 矩阵的 SVD 可以生成词嵌入向量

(2) **Skip-grams, CBOW:**

在猜测词的过程中产生词向量, 通过训练一个神经网络来猜测临近词的意思

优势: 比 SVD 快很多倍, 可以调 word2vec 包, 包含预训练的 embedding;

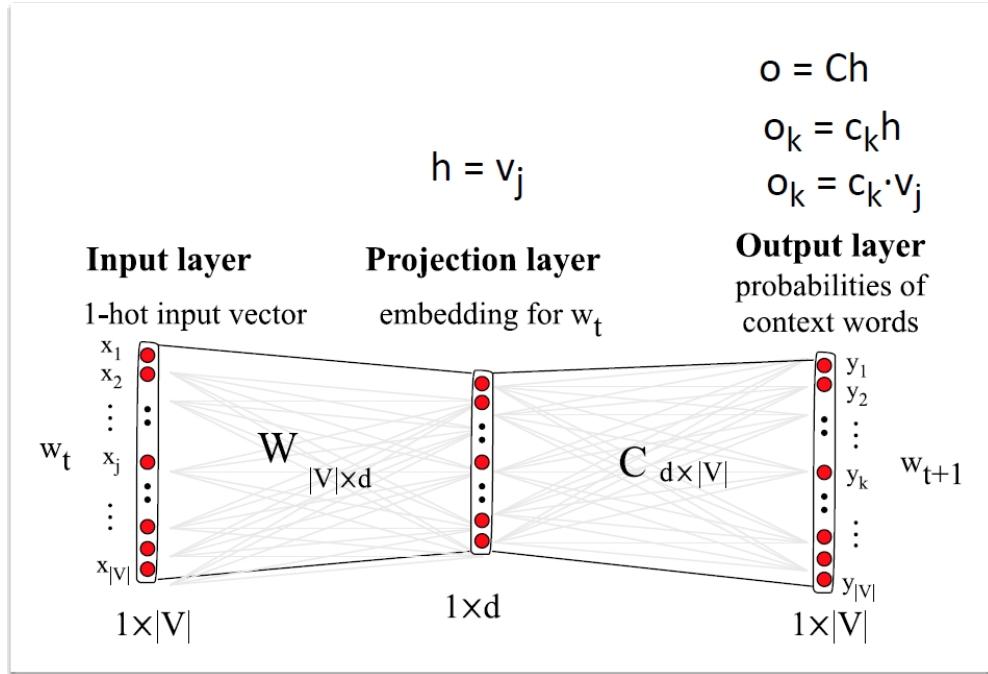
Skip-grams:

步骤:

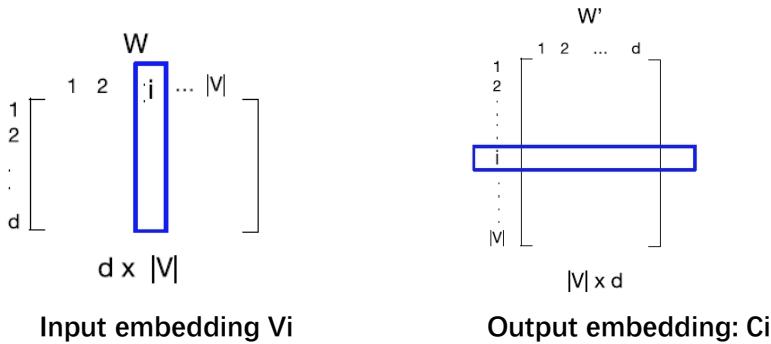
预测临近的 2C 个词; $C=2$: $[w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}]$

(因为训练目标是重建损失最小, 输出向量在训练完成时会基本等于输入向量, 则输出也是近似 One-hot 的)

记 $W(t)=W_j$, $W(t+1)=W_k$, 故需要算出 $P(W_k|W_j)$



2 embeddings:



$$p(w_k | w_j) = \frac{\exp(c_k \cdot v_j)}{\sum_{i \in |V|} \exp(c_i \cdot v_j)}$$

计算概率值：

问题：分母的计算需要全体数据集

$$\log \sigma(c \cdot w) + \sum_{i=1}^k \mathbb{E}_{w_i \sim p(w)} [\log \sigma(-w_i \cdot w)]$$

Negative Sampling:

训练目标是让正例的概率大，负例概率小

和 PMI 的关系：将一个移位的 PMI 矩阵分解成两个嵌入矩阵

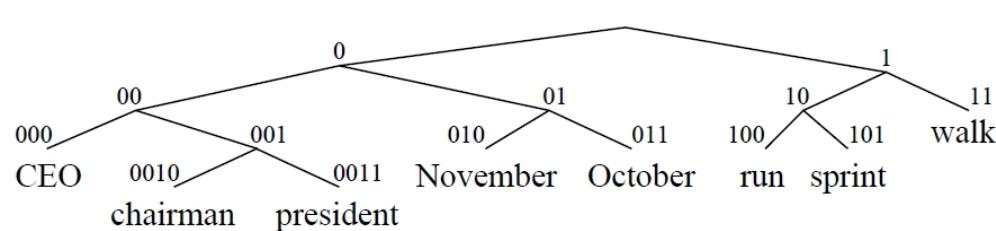
$$WW^T = M^{\text{PMI}} - \log k$$

(3) Brown Clustering: 一种根据词的前后词进行聚类的算法

算法：

一开始每个词都在一个等价类中，然后根据“Quality”=合并两个前面和后面的词有相似的概率的词“进行 cluster，这个 cluster 操作可以看成一种二元操作，其可以生成一个 bottom-up 的二叉树，这个二叉树按编号读出来就是其词向量 相似的词其向

量的距离也比较近。



Class-based language model

Suppose each word was in some class c_i :

$$P(w_i|w_{i-1}) = P(c_i|c_{i-1})P(w_i|c_i)$$

$$P(\text{corpus}|C) = \prod_{i=1}^n P(c_i|c_{i-1})P(w_i|c_i)$$

So What??

WEEK 7

□ Lexical Semantics (词汇语义学)

- Word Sense
- WordNet
- Word Similarity
- Inside Words: Thematic Role
- Beyond WordNet

Word sense (词义)

Homonyms

同形异义: bat;

同音异义: write;

(多义关系) Polysemy

有多个有关联的意思: blood bank; money bank

借喻(指同样的实体): school → institution/organization

(同义词) Synonyms: 可以相互替代的词, 但很少 perfect

是词义而不是词语之间的关系(large/big sister)

(反义词) Antonyms

vehicle → hypernym → car → hyponym(instance) → vehicle

(整体-部分关系)Meronymy: weel → Meronymy → carr → holonym → wheel

Wordnet: 反应词汇之间关系的稀疏图数据库，单词和同义词集之间的真值表(属于为 1, 不属于为 0), 采用 Meaning-based 遍历; 词汇本体。

语义关系: 词之间关系, 概念之间关系

同义词集: 同义词是最重要的关系, wordnet 中的关系是建立在同义词集之间的, 同义词是概念的例示

Super sense: 作为词义的粗粒度表示

动词的同义词集: 使用方式词连接; 支持继承; 时序关系更重要

形容词的同义词集: 1. 描述类 2. 关系类 3. 有情感色彩类

结构: 语义关系(Lexical relations)+概念关系→同义词集

限制: Wordnet 可以被 POS 限制, 分为 Paradigmatic (组合) relations (within POS) 和 Syntagmatic (聚合) relations (across POS);

有 4 类彼此未连接的 wordnet: 动词、名词、形容词、副词

问题: 稀疏; 关系未加权; 非有向关系; 未实现相互唤起; Types 和 roles 未分离

Word Similarity

同义: 0/1 关系, **相似性**: 两个单词的某个语义相似即可, 更宽松;

关联性: car-gasoline(related); car-bicycle(similar)

相似性算法:

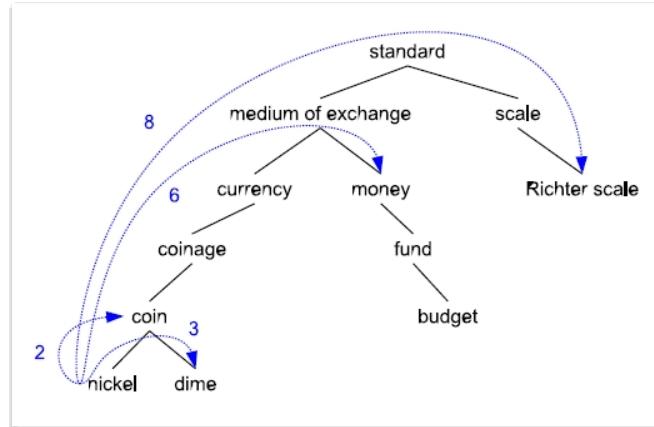
Thesaurus-based/Path-based(基于词典):

c 为含义结点(sense nodes)

$\text{pathlen}(c_1, c_2) = 1 + \text{edges}(c_1, c_2)$

$\text{simpath}(c_1, c_2) = 1/\text{Pathlen}(c_1, c_2)$

$\text{wordsim}(w_1, w_2) = \max \text{simpath}(c_1 \in w_1, c_2 \in w_2)$



问题: 每条边权重一样, 层次越高越抽象 $\text{sim}(\text{nickle}, \text{money}) < \text{sim}(\text{nickle}, \text{standard})$

我们希望每条边权重相互独立, 只通过抽象节点连接的相似性较低

Information Content Similarity(+P):

P(c): 在语料中随机抽取一个词, 属于 concept c 的概率, 特别地 $P(\text{root}) = 1$
等级越低, 概率越低

Self-information(surprisal 惊异度):

$\text{Information Content IC}(c) = -\log_2 P(c)$ 从事件发生得到多少信息量

$\text{LCS}(c_1, c_2) = \text{最小公共包含的节点}$

$\text{Sim_resnik}(c1, c2) = -\log P(\text{LCS}(c1, c2))$
 $\text{Sim_lin}(c1, c2) = 2 * \ln P(\text{LCS}(c1, c2)) / (\ln P(c1) + \ln P(c2))$
 $\text{Sim_jiangconrath}(c1, c2) = 1 / (\log P(c1) + \log P(c2) - 2 \log P(\text{LCS}(c1, c2)))$
 $\text{Sim_lesk} = \sum \text{overlap}(\text{gloss}(r(c1)), \text{gloss}(q(c2))) = 1 + (\# \text{overlap})^2$

分布式相似性判断：是否有相似的上下文

Inside words

Thematic Roles: 词语本身语义，使用“主题”表达(takers, givers→agent)

Thematic Role	Example
AGENT	<i>The waiter spilled the soup.</i>
EXPERIENCER	<i>John has a headache.</i>
FORCE	<i>The wind blows debris from the mall into our yards.</i>
THEME	<i>Only after Benjamin Franklin broke the ice...</i>
RESULT	<i>The French government has built a regulation-size baseball diamond...</i>
CONTENT	<i>Mona asked "You met Mary Ann at a supermarket?"</i>
INSTRUMENT	<i>He turned to poaching catfish, stunning them with a shocking device...</i>
BENEFICIARY	<i>Whenever Ann Callahan makes hotel reservations for her boss...</i>
SOURCE	<i>I flew in from Boston.</i>
GOAL	<i>I drove to Portland.</i>

除了 Wordnet 还有 MeSH

在特定领域 (e.g. 生物) 使用 wordnet 效果更好。

WEEK 8

□ Word Sense Disambiguation (词义消歧)

□ Compositional Semantics (组合语义学)

语义消歧 (Word Sense Disambiguation)

任务：给定有上下文的一个词；可能有的意思：决定这个词是哪个意思

对特定词语集合的 WSD 任务：可以使用机器学习训练分类器

适用于所有词语的任务：数据量大，关系稀疏，故不能使用针对特定词语集合的分类器

监督学习：

Need:

标签集 每个词语所有可能的含义

训练语料 (带标注)

特征提取：Collocational & Bag-of-Words

搭配 (考虑位置，待分类词左右加减 window size 出现的词一起构成向量)

guitar and bass player stand

$[w_{i-2}, \text{POS}_{i-2}, w_{i-1}, \text{POS}_{i-1}, w_i, \text{POS}_i, w_{i+1}, \text{POS}_{i+1}, w_{i+2}, \text{POS}_{i+2}, w_{i-2}^{i-1}, w_i^{i+1}]$

词袋 (不考虑位置，先构建一个可能出现的词的集合，在待分类词左右加

减 window size 的窗口内统计有无出现预定集合内的词，可以计数或用 binary：出现处记 1，否则为 0）

[fishing, big, sound, player, fly, rod, pound, double, runs, playing, guitar, band]

- The vector for:
guitar and bass player stand
[0,0,0,1,0,0,0,0,0,0,1,0]

分类器：可以使用朴素贝叶斯+平滑、基于规则的决策

Input:

- a word w in a text window d (which we'll call a "document")
- a fixed set of classes $C = \{c_1, c_2, \dots, c_j\}$
- A training set of m hand-labeled text windows again called "documents" $(d_1, c_1), \dots, (d_m, c_m)$

Output:

- a learned classifier $y: d \rightarrow c$

		Doc	Words	Class
$\hat{P}(C) = \frac{N_c}{N}$	Training	1	fish smoked fish	f
		2	fish line	f
		3	fish haul smoked	f
		4	guitar jazz line	g
$\hat{P}(w C) = \frac{\text{count}(w, C) + 1}{\text{count}(C) + V }$	Test	5	line guitar jazz jazz	?

Priors:
 $P(f) = \frac{3}{4}$ $P(g) = \frac{1}{4}$

Conditional Probabilities:
 $P(\text{line}|f) = \frac{1+1}{8+6} = 2/14$
 $P(\text{guitar}|f) = \frac{0+1}{8+6} = 1/14$
 $P(\text{jazz}|f) = \frac{0+1}{8+6} = 1/14$
 $P(\text{line}|g) = \frac{1+1}{3+6} = 2/9$
 $P(\text{guitar}|g) = \frac{1+1}{3+6} = 2/9$
 $P(\text{jazz}|g) = \frac{1+1}{3+6} = 2/9$

V = {fish, smoked, line, haul, guitar, jazz}

Choosing a class:
 $P(f|d5) \approx 3/4 * 2/14 * (1/14)^2 * 1/14 \approx 0.00003$
 $P(g|d5) \approx 1/4 * 2/9 * (2/9)^2 * 2/9 \approx 0.0006$

对测试排序(Decision Lists):

使用 $\log[P(\text{sense 1} | \text{feature}) / P(\text{sense 2} | \text{feature})]$ 来做 ranking

测评：

外在指标(extrinsic): 放到具体任务中看能不能完成得更好

内在指标(intrinsic): 准确率和验证集

Baseline:

(1) 使用最常出现的意思，人类准确率 80%;

(2) Lesk 算法：

选择 context 和某个 sense 的 signature(包括词典中注解(gloss)和例句(examples))

重合的词，用 $\text{idf}_w = \log(\# \text{文章} / \# \text{有单词 } w \text{ 的文章})$ 计算重合的词的值并求和，作为对该 sense 的评分，目的是降低 function words 的重要性。

$\text{score}(\text{sense}_i, \text{context}_j) = \sum \text{idf}_w \text{ for } w \in \text{overlap}(\text{signature}_i, \text{context}_j)$

半监督学习：半监督学习需要大量人工标注的数据，使用 bootstrapping 解决：

种子：已知的固定搭配(collocation)；或者含义基本完全一致的一篇语料(discourse)
使用种子训练的分类器对所有样本进行分类，将结果可信度较高的添加到种子集中，重复步骤。

问题：

为了鲁棒性可能需要对每个歧义词训练一个分类器

需根据应用场景选择合适的训练集，包括 tags/labels/sense

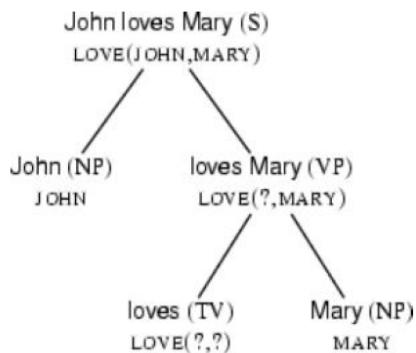
Thesaurus/Dictionary Methods 也可以用来消歧

Intuition: word's context 含有丰富的信息；依赖 counting 的方法效果很好

组合语义学 (Compositional Semantics)

只提取感兴趣的实体、关系和角色，只需大致知道整体意思，保证效率。

1. 使用一阶逻辑，根据语法分析建立逻辑表达式，注意量词顺序



Standard example: "Every man loves a woman":

- Reading 1: the women may be different
 $\forall x(\text{man}(x) \rightarrow \exists y(\text{woman}(y) \wedge \text{love}(x,y)))$
- Reading 2: there is one particular woman
 $\exists y(\text{woman}(y) \wedge \forall x(\text{man}(x) \rightarrow \text{love}(x,y)))$

2. 使用信息抽取

WEEK 10

Information Extraction (信息抽取)

Named Entity Recognition

Relation Detection and Classification

Template Filling

Biomedical Information Extraction

命名实体识别 (NER)

基于规则

直接用 LIST 存储或人工构造正则表达式，或者词出现的语法规则

基于机器学习

训练：对文本进行编码 → 人工标注训练数据 → 特征提取 → 训练分类器（抽取 substring）

测试：接受 doc → 运行模型来推断 label → 输出识别出的实体

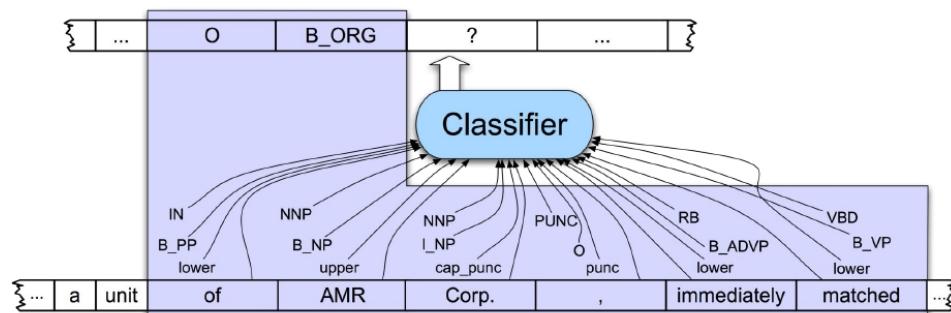
IOB 编码：对于 N 类需要 $2N+1$ 个标签。对于每一类：有 B-类别 表示属于某一类实体的开始，I-类别 表示实体的后续，O 表示不属于任何类。

IO encoding IOB encoding

Fred	PER	B-PER
Showed	O	O
Sue	PER	B-PER
Mengqiu	PER	B-PER
Huang	PER	I-PER
's	O	O
new	O	O
painting	O	O

特征选择：

(1) word shape: 将单词的长度、大小写等特征进行区分性映射(如 Cpa1 → Xxxd)



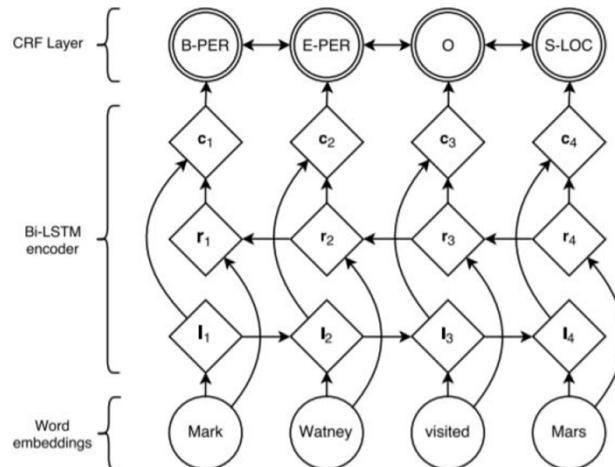
(2) 序列模型：

1. MEMM (最大熵马尔科夫模型) 基于当前信息和之前的决策进行决策

2. Conditional Random Fields (CRFs):

全序列条件决策模型，非局部条件决策；训练较慢，但能避免局部偏差

$$P(c|d, \lambda) = \frac{\exp \sum_i \lambda_i f_i(c, d)}{\sum_c \exp \sum_i \lambda_i f_i(c, d)}$$



测评：准确率、召回率、F1: $P=TP/(TP+FP)$; $R=TP/(TP+FN)$; $F1 = 2PR/(P+R)$

boundary errors: FP+1/FN+1(First Bank of Chicago)

关系发现和关系分类

Tuple(a, b) 存放 a, b 之间的关系

1. 首先判断是否存在关系
2. 对关系进行分类

原因：在训练时通过第一步能过滤掉大多数词语对；不同任务可以选择不同特征集
特征：

1. 命名实体本身的特征（类型、首字母等）
2. 命名实体周围词语的特征（window size 内的词语）
3. 命名实体所在的语法环境（产生式、依赖式等）

半监督 bootstrapping：

从已知有关系的实体出发，在语料中提取更多构成关系的特征，再利用这些关系特征得到更多实体之间的关系，之后重复操作。

<Mark Twain, Elmira> **Seed tuple**

- Grep (google)
- “Mark Twain is buried in Elmira, NY.”
 - X is buried in Y
- “The grave of Mark Twain is in Elmira”
 - The grave of X is in Y
- “Elmira is Mark Twain’ s final resting place”
 - Y is X's final resting place.

模板填充(Template Filling): 填充模板化数据

(1) Cascades of Transducers

(2) Machine Learning: 1. One seq classifier per slot 2. One big sequence classifier

问题：跟语言相关、需要特定领域知识

采用 IBOES 标注(E=end of entity; S=singleton word entity)

错误会传递（错误的命名实体识别产生错误的关系）；信息抽取的准确率不高

生物信息抽取

特定领域问题：语料充足，主要研究问题是 NER 和（相互反应）关系分析。

WEEK 11

□ Question-Answering 问答

□ Information Retrieval 信息检索

□ Summarization 摘要

问答(QA)

类型：在一段材料中寻找答案；关系数据库的接口；交互式问答

主要步骤：问题分类+关键词提取 -> 文章信息抽取 -> 提取回答

问题分类:

回答类型: 事实型问题(who, where); 分类界限不明: who sells(org), which president(person); 使用人为规则和机器学习

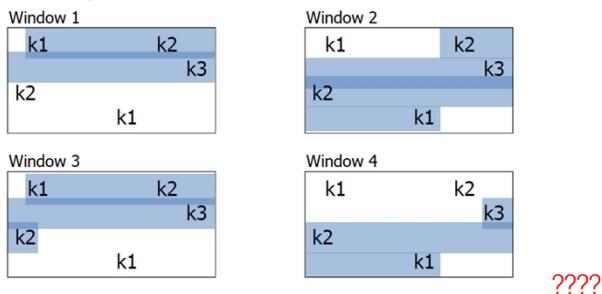
问题关键词提取: 提取出若干无关联重点词(类似 bag-of-words)

文章抽取: 先选择带有所有(6个)key words 的文章, 再根据得到的文章数是否达到门限"判断限制是否需要调整, 若文章太多(少)则增添(去掉)一个 key words

Passage Scoring

Passages are scored based on keyword windows

- For example, if a question has a set of keywords: {k1, k2, k3, k4}, and in a passage k1 and k2 are matched twice, k3 is matched once, and k4 is not matched, the following windows are built:



文章排序: 1. 关键词窗口中词出现的顺序与问题中一致 2. 两个关键词之间最长距离 3. 窗口中与 kw 不相关的词数量

潜在答案排序: 设置 Features; 答案的 pattern 有不同的 precision score; wiki/wordnet 中的答案更可靠。

AskMSR: 根据重写规则重构问题-> 使用搜索引擎收集摘要 -> 建立 N-gram 模型 (权重为在文章中出现的频率) -> 筛选 (与问题类型匹配的得分较高) -> 合并答案

评价指标: 平均排序倒数 (1/(第一个正确答案的在预测出的答案集序号))

信息检索(Information Retrieval)

基本假设: 文章含义能够根据文章中出现的词 (bag of words) 的频率推测

Inverted index: {w: documents contain w}; **stop list**: 包含无关词 (of, a 等, 去掉可以减少 Inverted index 的 size); **stemming**: 关注词干

Phrases: 短语搜索可以通过将短语拼凑在一起递增地操作

特定型检索 向量词空间: 文章和查询语句被表示为数字向量, 这些数字来自文本中的单词。

$d_j = (t_1, t_2, \dots, t_n)$ 表示 n 个词语类型的布尔向量(所有词的权重相同), 可以通过点乘来判定相似性; 考虑加权.

局部权重: $f=w$ 在 doc 中的频率;

全局权重(#docs 出现 w, 越少越重要) $\text{idf}_i = \log(\#\text{docs}/\#\text{docs with term } i)$;

TF-IDF: $f * \text{idf}_i$

通过余弦相似度来衡量相似性

步骤: 根据 kws 找到所有相关 doc, 将 query 和 doc 向量化, cos 衡量相关性并排序, return docs

摘要(summarization): 将最多的信息用最少的空间输出(hard and open)

种类: **报道性**(替代)&**指示性**(形容); **摘抄**&生成; 多 docs&单 doc; **通用的**&基于用户的

单文章摘要: 选取内容 -> 对抽取的句子进行排序 -> 重新组织, 删除冗余信息

选取句子: (1)TF*IDF 加权, 用 4 种特征: 线索词、开头或者结尾词、句子位置、主题词频
(2)词语链: (指代关系构成的) 链的强度用长度和同义性衡量, 选取强度较强的链的第一个句子
(3)主题词: 使用 log 概率并设置阈值判断一个词能不能作为主题词, 包含主题词较多的句子被抽出。

机器学习方法: 一组手写摘要的训练集 → 贝叶斯分类器估计给定句子出现摘要的概率 → 根据该概率对文档句子进行排序, 产生新的摘要; 特征包括大写字、主题字等。

句子匹配: Direct match, Direct join, Unmatchable, Incomplete.

测评: 交叉验证, 留一法训练; **performance**: Direct Sentence Matches 和 Direct Joins 占的比例 ; **correct**: Has direct sentence match & present in manual summary; **automated**: Rouge, compare with human, BLEU

WEEK 12

□ Machine Translation

□ MT Evaluation

□ Statistical Machine Translation

□ Neural Machine Translation

机器翻译 基于概率和规则

评价指标

1. **人为评价**: 忠实度、流畅度 (不同人的感受和评价不同); Kappa 系数 = $k = (p(A) - p(E)) / (1 - p(E))$, $p(A)$ =为评价者给出一致评价的可能性, $p(E)$ =平均得到一致评价的可能性 (比如共 5 个评分点, 2 个 evaluator, $P(E) = 1/5$)

2. **自动评价指标**: 给定机器翻译和人的参考翻译, 要求给出两者之间的相似性

WER(Word error rate) = (替换、查找和删除操作总数)/(参考翻译的长度)

BLEU(n-gram 准确率) = $\min(1, \text{output length} / \text{ref length}) (\prod_i \text{precision}_i)^{1/n}$; precision_i =(output 中的 i-gram 在 ref 里出现的占 output 里所有 i-gram 的比例)。有短词惩罚并且不能用 ref 中相同的词两次, 可以使用多个 ref 增大可信度。

BLEU in Action

SYSTEM A: [Israeli officials] responsibility of [airport] safety
2-GRAM MATCH 1-GRAM MATCH

REFERENCE: Israeli officials are responsible for airport security

SYSTEM B: [airport security] [Israeli officials are responsible]
2-GRAM MATCH 4-GRAM MATCH

Metric	System A	System B
precision (1gram)	3/6	6/6
precision (2gram)	1/5	4/5
precision (3gram)	0/4	2/4
precision (4gram)	0/3	1/3
brevity penalty	6/7	6/7
BLEU	0%	52%

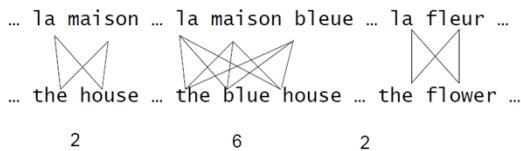
基于统计的机器翻译

大体思路: $f \rightarrow$ 翻译模型 (给出候选 $p(f|e)$) \rightarrow 语言模型 (确定翻译的合理性 $p(e)$) \rightarrow 解码输出($\text{find argmax}\{p(e)p(f|e)\}$)

需要从语句对齐的语料中学习如下几个概率:

1. $n(x|y)$ 词 y 在译文中产生 x 的概率
2. p 某个单词在译文中不出现, 被删除的概率
3. t 实际的翻译概率表
4. $d(j|i)$ 原文中位置 i 的词出现在译文中位置 j 的概率

词语对齐: 开始时假设所有词语的对应都是等概率的, 然后观察两种语言的哪些词经常一起出现, 并提升他们的概率;



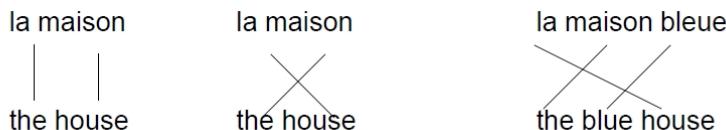
How many alignments are there?

EM 算法:

1. 找出所有的 $p(f|e)$, 并假设等可能

(l a the) 1/4	(l a house) 1/3	(l a blue) 1/3	(l a flower) 1/2
(m the) 1/4	(m house) 1/3	(m blue) 1/3	(f flower) 1/2
(b the) 1/4	(b house) 1/3	(b blue) 1/3	
(f the) 1/4			

2. 计算可能的组合绑定情况的概率并标准化 $p_i(f&e) = p(\text{当前匹配}) / \sum p(\text{所有种类的匹配})$



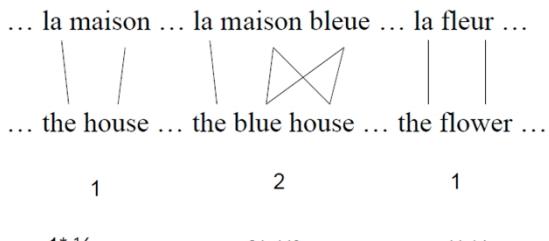
$$\frac{1}{4} * \frac{1}{3} = 1/12$$

$$\frac{1}{4} * \frac{1}{3} = 1/12$$

$$\frac{1}{4} * \frac{1}{3} * \frac{1}{3} = 1/36$$

$$(1/12) / (2/12) = 1/2 \quad (1/12) / (2/12) = 1/2 \quad 1/6 \text{ For each}$$

3. 对一组绑定, 计算 $\text{Count}(f|e) = \sum_i \#(\text{different assignments for a sentence with } f \& e) * p_i(f \& e)$, 并且标准化



8/6 = Discounted count for (la|the)

$(la the) = 8/6$	$(la the) = 8/6 / 18/6$	$(la the) = .44$
$(m the) = 5/6$	$(m the) = 5/6 / 18/6$	$(m the) = .27$
$(b the) = 2/6$	$(b the) = 2/6 / 18/6$	$(b the) = .11$
$(f the) = 3/6$	$(f the) = 3/6 / 18/6$	$(f the) = .16$

4. 概率最大的即为需要的绑定 $p(f | e)$

对于给定的 f 寻找使得 $p(f | e) * p(e)$ 最大的 e , 使用动态规划求解。

Word based 翻译缺陷：对多个已知单词对应同一个未知语言的情况难以处理，于是有 **phrase based** 翻译：1. 先将待翻译语料划分为词组（很多可能的划分）2. 对应词组翻译并算概率 $P(F|E)$ 3. 解码；**好处**：多对多映射；减少单个词语的歧义性

换位概率(distortion) $d(ai-b_{\{i-1\}})=\alpha^{|ai-b_{\{i-1\}}|}$, 其中 ai 是第 i 个英语词组翻译成的目标语言后目标语言的开始位置, $b_{\{i-1\}}$ 是第 $i-1$ 个英语词组翻译成的目标语言后目标语言的结束位置

$P(f | e) = \prod$ 对应词组翻译概率 * 换位概率

$$P(F | E) = \prod_{i=1}^l \phi(\bar{f}_i, \bar{e}_i) d(a_i - b_{i-1})$$

Position	1	2	3	4	5
English	Mary	did not	slap	the	green witch
Spanish	Maria	no	dió una bofetada	a la	bruja verde

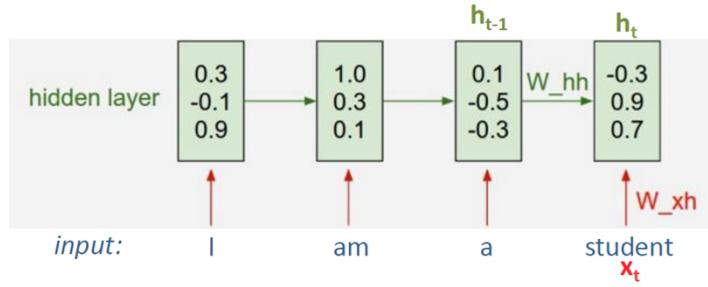
$$\begin{aligned} P(F|E) &= P(Maria, Mary) \times d(1) \times P(\text{no}| \text{did not}) \times d(1) \times \\ &\quad P(\text{dió una bofetada} | \text{slap}) \times d(1) \times P(\text{a la} | \text{the}) \times d(1) \times \\ &\quad P(\text{bruja verde} | \text{green witch}) \times d(1) \end{aligned}$$

为了训练对应词组翻译概率, 需要很大的双语语料, 并且做词组对齐。词组对齐首先进行单词对齐, 用“alignment templates”得到词组对齐。解码过程可以使用动态规划或者 A* 搜索。

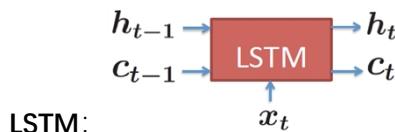
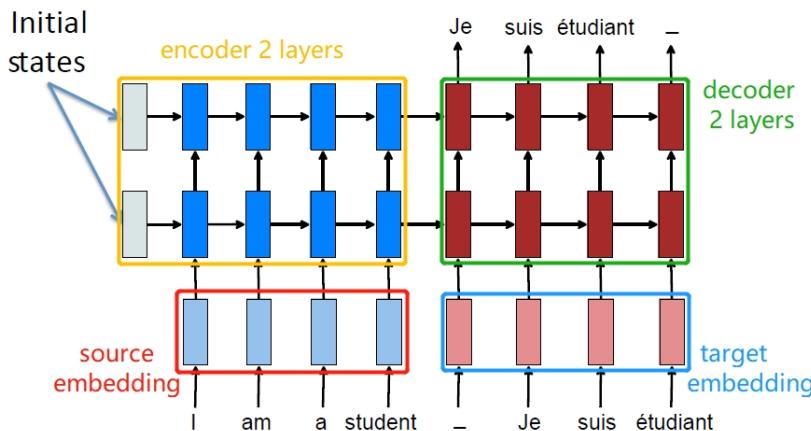
神经机器翻译

RNN: 直接对 $p(\text{target} | \text{source})$ 建模, 进行端到端训练

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1})$$



RNNs to represent sequences!

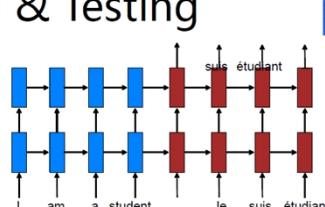


输入输出使用 word embedding, 输出概率使用 softmax 标准化, loss= $\sum_i -\log P(i)$, 误差向后传; 测试时只给定 source sentences, 每次选择概率最大的单词输出

Training & Testing

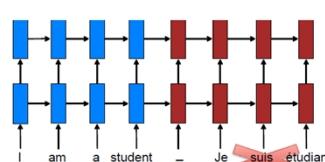
Training

- Correct translations are available.



Testing

- Only source sentences are given.



	mol	suis	étudiant	-
étudiant	0.1	0.1	0.5	0.1
-	0.1	0.1	0.2	0.6
je	0.3	0.1	0.1	0.1
mol	0.4	0.1	0.1	0.1
suis	0.1	0.6	0.1	0.1

Simple beam-search decoders!

局限: 1. 受限于词语库大小, 希望扩大词汇范围 (拷贝技巧, 加入标签) 2. 长句子翻译不理想 (使用 attention) 3. 有些语言比较复杂 (使用字符层面翻译预测)