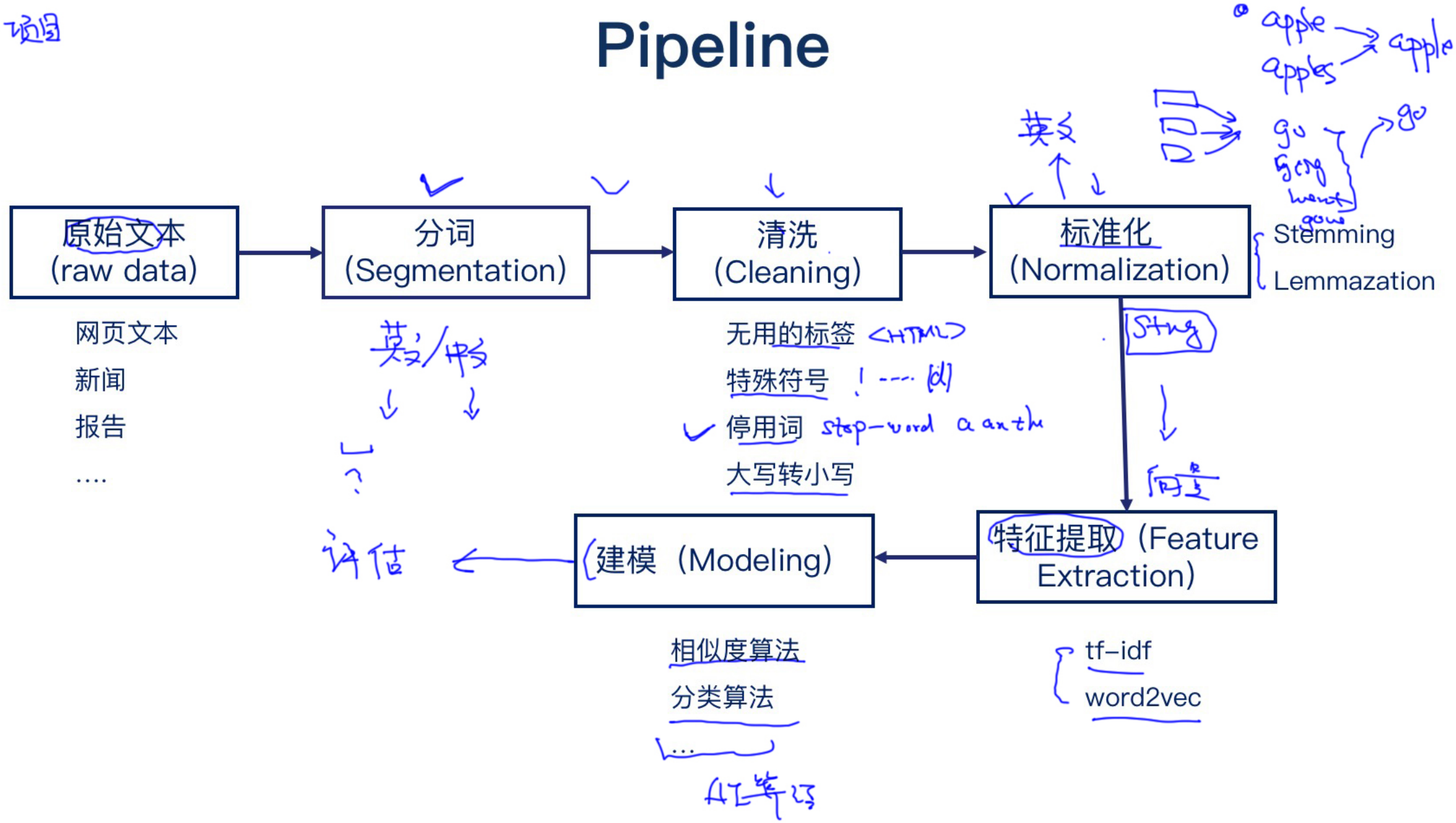
# 自然语言处理训练营（3）

NLP 项目

# Pipeline



# What we cover today

- Word Segmentation
- Spell Correction
- Stop Words Removal
- Stemming

# Word Segmentation（分词）

# Word Segmentation Tools

Jieba分词 https://github.com/fxsjy/jieba

SnowNLP https://github.com/isnowfy/snownlp

哈工大 LTP http://www.ltp-cloud.com/

HanNLP https://github.com/hankcs/HanLP/

.....FudanNLP

# Segmentation Method 1: Max Matching(最大匹配)

Global Optimal (不能保证)

贪心 → 选择当前最优解

策略

DP → Global optimal

暴能看到全局

**前向最大匹配** (forward max matching)

max_len = 5

例子：我们经常有意见分歧

词典：["我们", "经常", "有", "有意见", "意见", "分歧"]

① 我们经常有意见分歧 ×

O(字符)

[我们经常] ×

[我们经] ×

[我们] ✓

② [经常有意见] ×

[经常有意] ×

[经常有] ×

[经常] ✓

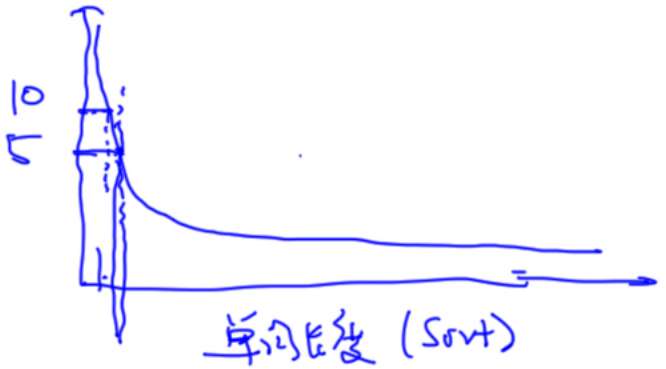
③ [有意见分歧] ×

[有意见分] ×

[有意见] ✓

④ [分歧] ✓

Beam Search



# Segmentation Method 1: Max Matching(最大匹配)

## 后向最大匹配（backward-max matching）

例子：我们经常有意见分歧

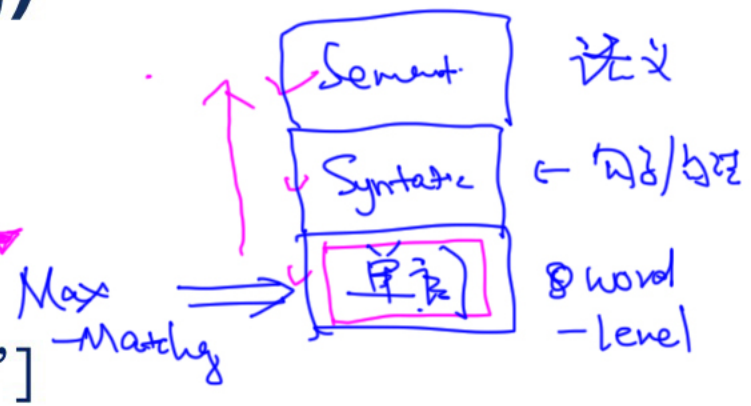
词典：["我们", "经常", "有", "有意见", "意见", "分歧"]

# Segmentation Method 1: Max Matching(最大匹配)

**最大匹配的缺点?**

例子：我们经常有意见分歧

词典：[“我们”, “经常”, “有”, “有意见”, “意见”, “分歧”]




- 细分（有可能是更好）
- 局部最优
  - 效率（Max-len）
  - 歧义（不能考虑语义）

# Segmentation Method 2: Incorporate Semantic (考虑语义)

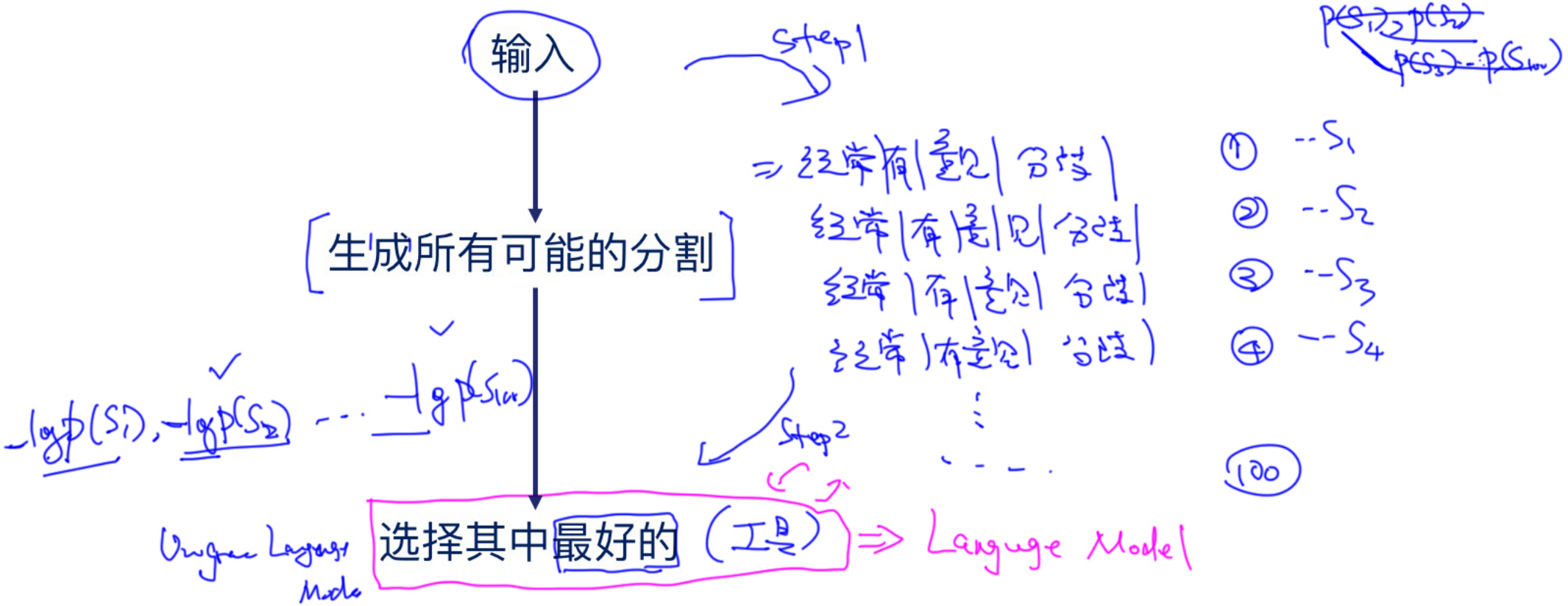
- 例子：经常有意见分歧

词典：[“有”，“有意见”，“意见”，“分歧”，“见”，“意”,]

# Segmentation Method 2: Incorporate Semantic (考虑语义)

例子：经常有意见分歧

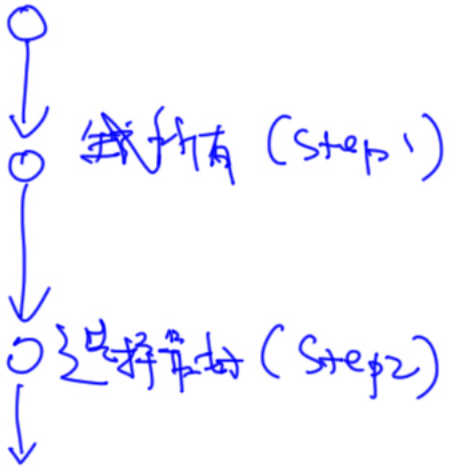
词典：["有"，"有意见"，"意见"，"分歧"，"见"，"意"]



# Segmentation Method 2: Incorporate Semantic (考虑语义)

怎么解决效率问题?

Step1 > Step 分开
Step1 + Step2 ?
Viterbi
(Dp)



Ask him!

Viterbi (USC → 通讯)

例子：“经常有意见分歧”

词典：[“经常”，“经”，“有”，“有意见”，“意见”，“分歧”，“见”，“意”，“见分歧”，“分”]

概率：[ 0.1，0.05，0.1，0.1，0.2，0.2，0.05，0.05，0.05，0.1 ]

-log(x)：[ 2.3，3，2.3，2.3，1.6，1.6，3，3，3，2.3]

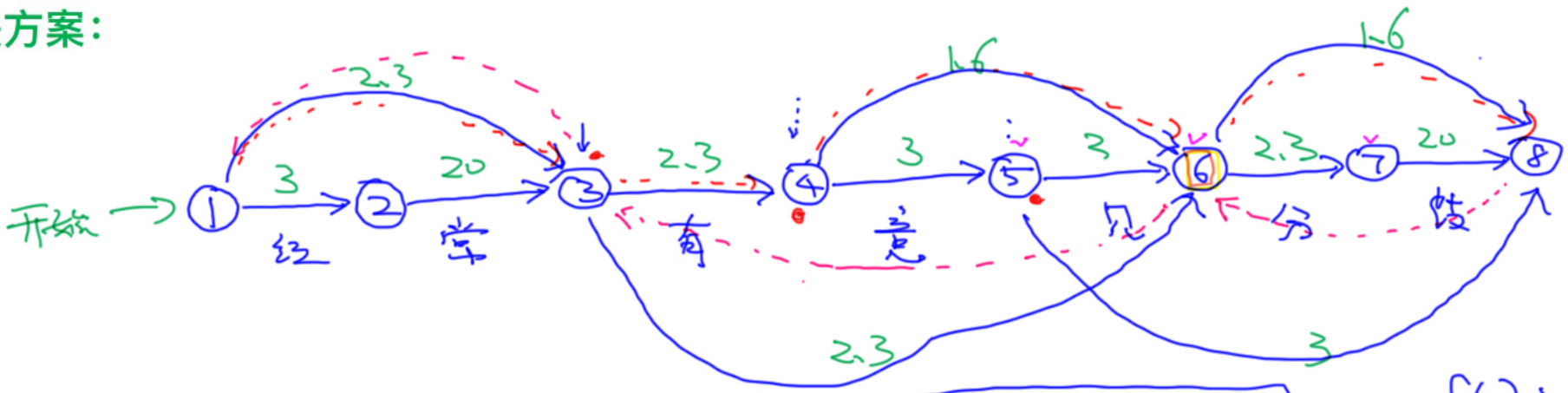
$P(\text{某}) = 10^{-8}$

10个

$P(\text{经常}) = 0.1$

$-\log(10^{-8}) = 20$

**解决方案：**




定义：
- f(8)：从节点1到8的最短路径的值
- f(7)：从节点1到7的最短路径的值
- f(6) = ……

“经常，有意见分歧！”

$f(7) = f(6) + 2.3$

$f(6) = \min\{f(3)+2.3 = 4.6,\ f(4)+1.6 = 6.2,\ f(5)+3 = 10.6\}$

$f(8) = \min\{f(5)+3 \cdots$ 从5到8节点
$f(6)+1.6$ 从6到8节点
$f(7)+20$ 从7到8节点…

$P(\text{经常，有意见}) > P(\text{经常，有意})$

$\log P(\text{经常，有意见}) > \log P(\text{经常，有意})$

$-(\log P(\text{经常}) + \log(\text{有}) + \log(\text{意见})) > -\log($

f(8)：…

| 0 | 3 | 2.3 | 4.6 | 7.6 | 4.6 | 6.9 | 6.2 |
|---|---|---|---|---|---|---|---|
| f(1) | f(2) | f(3) | f(4) | f(5) | f(6) | f(7) | f(8) |
| 开始 | 1 | 1 | 3 | 4 | 3 | 6 | 6 |

1368

# Word Segmentation Summary

**知识点总结：**

- 基于匹配规则的方法
- 基于概率统计方法（LM, HMM, CRF..）
- 分词可以认为是已经解决的问题

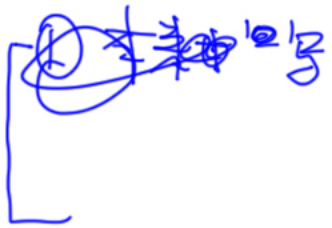
**需要掌握什么？**

- 可以自行实现基于最大匹配和Unigram LM的方法

搜索相

# Spell Correction（拼写错误纠正）

# Spell Correction (拼写错误纠正)



用户输入（input） → AI → 用户输入(correction)

天起 → 天气

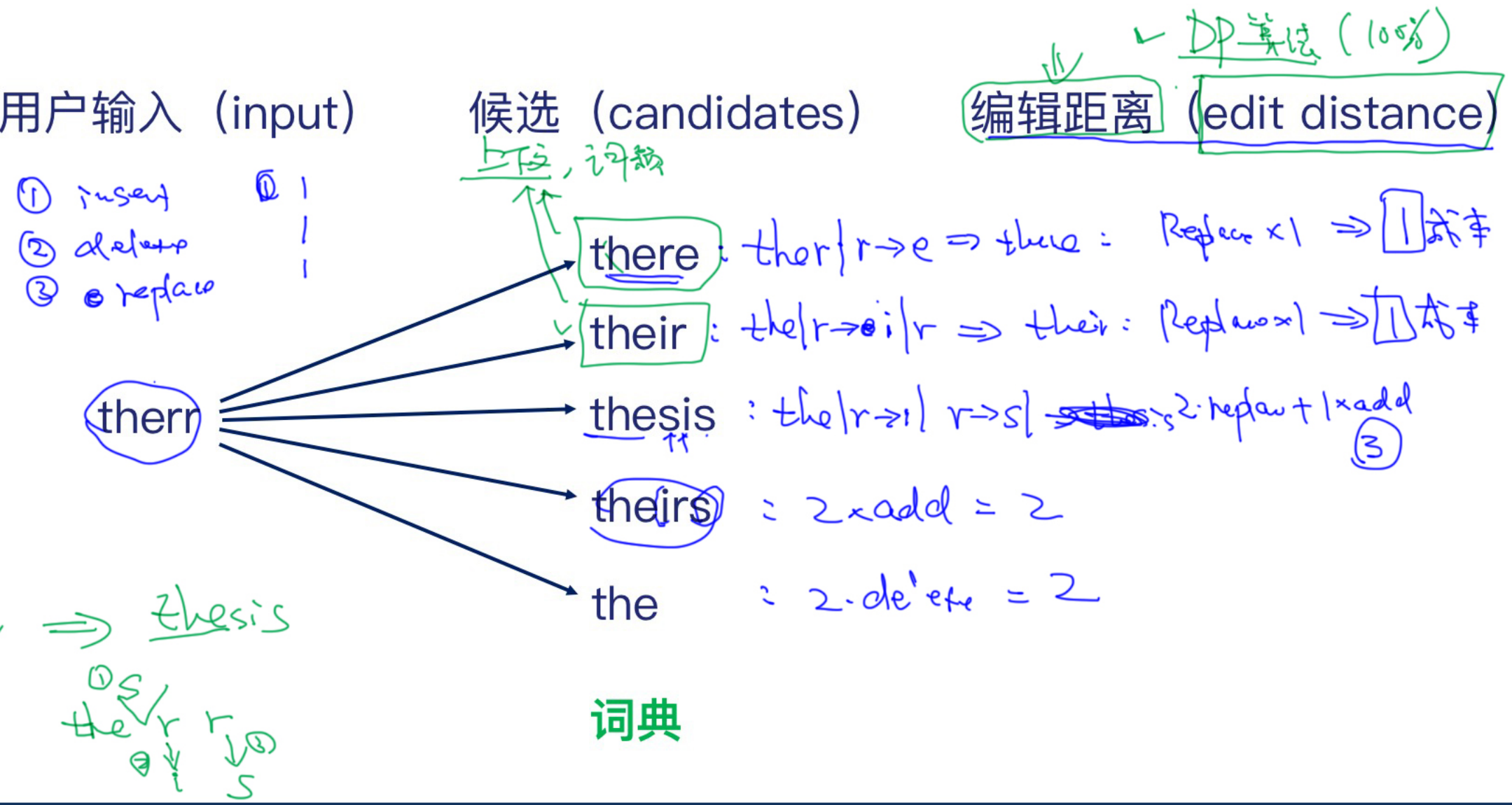
theris → theirs

机器学系 → 机器学习

**怎么做?**

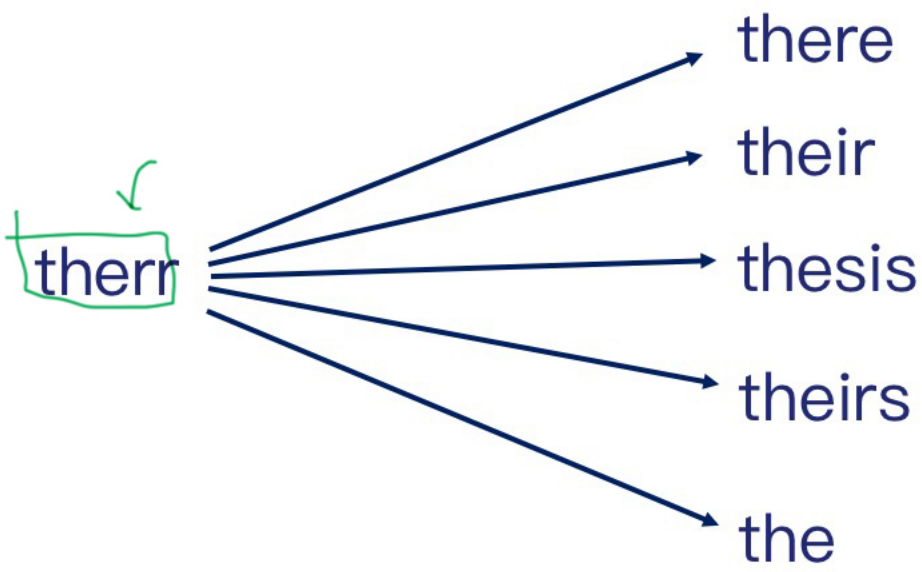
# Find the words with smallest edit distance

用户输入（input）　　候选（candidates）　　编辑距离（edit distance）



# Find the words with smallest edit distance

用户输入（input）　　候选（candidates）




词典

DP

DP算法核心：

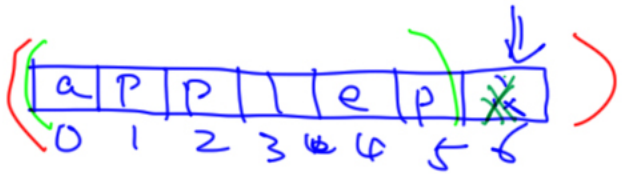
insert char

Big Problem ⟹ Smaller Problem

$$F(n) = F(n-2) + F(n-1)$$

$$F(n+1) = F(n-1) + F(n)$$

$S_1$ :




$S_2$ :


$$d(S_1[0:6], S_2[0:8]) = d(S_1[0:5], S_2[0:7])$$

$$\begin{bmatrix} F(n) \\ F(n-1) \end{bmatrix} = \begin{pmatrix} & \\ & \end{pmatrix} \begin{bmatrix} F(n-1) \\ F(n-2) \end{bmatrix}$$

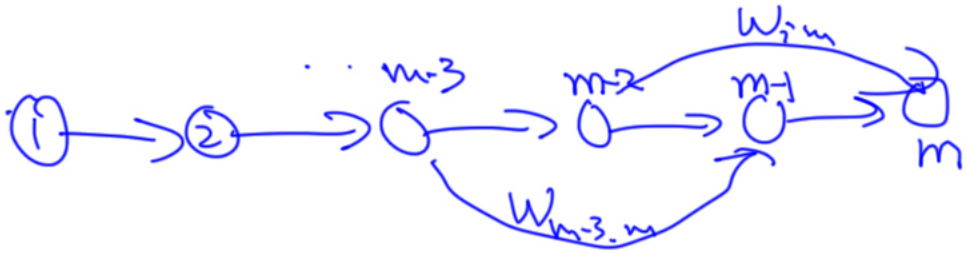
$$\begin{bmatrix} F(n) \\ F(n-1) \end{bmatrix} = \begin{pmatrix} & \\ & \end{pmatrix} \begin{bmatrix} F(1) \\ F(0) \end{bmatrix}$$

$$A = (\Sigma \cup D)$$

$$A^n = (\Sigma \cup D)(\Sigma \cup D)$$

$$[\ ]^n =$$

$f(n) = f(n-2) + f(n-1)$

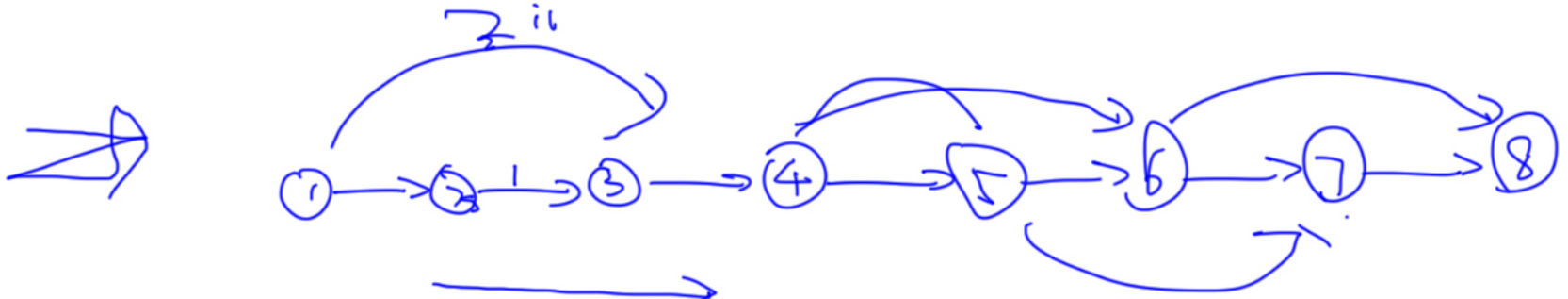


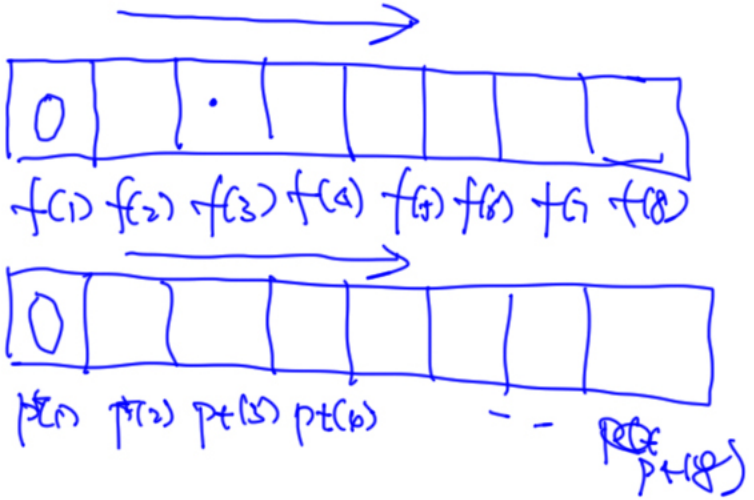

$f(m)$：从节点1到节点，m的最短路径 值

$f(m_1) = 1$

$f(n) = \min_{i \in \Omega(m)} \{ f(i) + W_{im} \}$

$\Omega(m)$：all the incoming links




~~f(1)~~ = 0

f[1] = 0, pt[1] = 0

for i = 2 ... 8

  for j in incoming_links

    best_sum = +∞

    best_incoming_link_idx = i

    sum = f(j) + w(j → i)

    if sum < best_sum:

      best_sum = sum;

      ~~pt[i]~~

      pt[i] = j

$f(3) = f(2) + 1$

$= f(1) + 2$

input: 经常有点儿含蓄

$S_1$ -
$S_2$ - [ ] 、、
$S_3$ - -
⋮
$S_{100}$

$P(S_1), P(S_2) \cdots P(S_{100})$

$P(\text{经常, 有点儿, 含蓄}) = P(\text{经常})P(\text{有})P(\text{点儿})P(\text{含蓄})$

$= 0.0001 \cdot 0.000035 \cdot 0.000002 \cdot 0.0001$

$= 0$ -inf / underflow

计算: double / float

$\log P(\text{经常, 有点儿, 含蓄}) = \log P(\text{经常}) + \log P(\text{有}) + \lg P(\text{点儿}) + \lg P(\text{含蓄})$

$\log(x \cdot y \cdot z) = \lg x + \lg y + \lg z$

等同

$P(S_1) \geq P(S_2) > P(S_3) > P(S_4)$

$\log P(S_1) > \log P(S_2) > \log P(S_3) > \log P(S_4)$

$S_1$ = ① 经常 | 有 | 意见 | 分歧 |

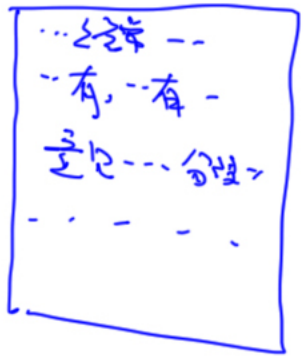
$S_2$ = ② 经常 | 有意见 | 分歧 |

Unigram Model ←

Unigram Lange Model

$$p(\text{经常}) = \frac{100 \leftarrow}{\#\text{of words in 书} \leftarrow}$$

书




经常：100

有：1000

意见：500

分歧：200

…

…

词 ⇒ 语言模型 [Langage Model]

$p(S_1) = 0.3$

$p(S_2) = 0.3+$

① $p(\text{经常}, \text{有}, \text{意见}, \text{分歧}) = p(\text{经常}) \cdot p(\text{有}) \cdot p(\text{意见}) \cdot p(\text{分歧}) = 0.3$

$p(\text{经常}, \text{有意见}, \text{分歧}) = p(\text{经常}) \, p(\text{有意见}) \, p(\text{分歧}) = 0.3+$

$p(\text{我们今天上课}) = p(\text{我们}, \text{今天}, \text{上课})$

$= p(\text{我们}) \, p(\text{今天}) \, p(\text{上课})$