

NLP 基础技术：词法分析（词性标注和词义标注）；句法分析（判断成分和句法结构，有完全/浅层句法分析）；语义分析；语用分析（具体运用）；篇章分析（整理解析分析）
NLP 应用技术：机器翻译，信息检索，情感分析，自动问答，自动文摘，社会计算，信息抽取，正则表达式：[A-Z]（从 A 到 Z），[123]（匹配 1 或 2 或 3），[^\a]（不是 a），[a^]（a 和 ^，因为只在 [后有效），a|bc（替代产生式），?（上一个字符是可选的），+（出现一次或者更多次），*（出现 0 次或者更多次），.（点，匹配任意单个字符），\$（在结尾匹配，加在 RE 的最后），^（在开头匹配，加载 RE 的最前）错误的类别：假阳性（不该匹配却匹配，精确度），假阴性（该匹配却没有匹配，覆盖率）
词元（Lemma）：同一个词干（stem）和词性（part of speech），大致相同的词义
词形（Wordform）：词的表面形式（把单复数啥的变化都加上）
词型（Type）：一个单词
词例（Token）：词型在文章中的一个实例
中文分词：Baseline 方法是贪心(最长匹配)法

形态学（Morphology）：研究单词是如何从语素构造语素（Morpheme）：词干（Stem）和词缀（Affix）两种广义的构造形式：①屈折（Inflectional）：不改变词类的词缀（walk, walking）②派生（Derivational）：改变意思和词类（clue, clueless）
词干还原（Stemming）：只关心词干，不关心结构，常用于信息检索应用；比如 Porter Stemmer，基于规则去词缀，不保证产生真实词干，但不影响 IR。
断句（Segmenting Sentences）：用二分类器（EOS/NotEOS），基于规则或 ML 来判断句号是否为一句话的结束
最小编辑距离：在插入，删除和替代意义下的最少编辑距离；应用：评估机器翻译和语音识别的效果；命名实体（Named Entity）识别和指代（Entity Coreference）识别；解法：设 D(i,j) 为 A[1...i] 和 B[1...j] 的最短编辑距离，目标是让 A 靠近 B。Levenshtein 插入和删除代价为 1，替换代价为 2。
初始化 D(i,0) = i, D(0,j) = j

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 & \text{deletion} \times \text{删除} \times 1 \\ D(i,j-1) + 1 & \text{insertion} \times \text{添加} \times 1 \\ D(i-1,j-1) + 1 & \text{substitution} \times \text{替换} \times 1 \end{cases}$$
$$D(i,j) = \begin{cases} \text{insertion} & \text{if } i=0 \\ \text{deletion} & \text{if } j=0 \\ \text{substitution} & \text{if } i>0 \text{ and } j>0 \end{cases}$$

如果 Xi=Yj，则认为在这里是对齐的；为了跟踪对齐情况，仿照 LCS 维护一个箭头数组跟踪 insert (LEFT), delete (DOWN), subst (DIAG)；时间复杂度 O(nm), 输出 O(n+m)
带权最小编辑距离：维护单个字母 del 和 ins 权重，两字母之间 sub 权重；用于修正一些拼写错误
语言模型：P(w₁|w₁, ..., w_{n-1})，但是由于数据不足，假设 Markov 性质成立；Bigram 为 Markov 链，预测用 MLE，即 P(W_i | W_[i-1]) = count(W_{i-1}, W_i) / count(W_{i-1})
Shannon 可视化方法：根据概率选 <s>，然后根据给定词为条件，出现下一个词的概率选，直到选择 <s>
Unigram(不用条件概率)；Bigram(用上个为条件)；Trigram；Quadrigram
封闭词汇任务和开放词汇任务（没见过的替换为 <UNK>）
评价 N-gram 模型：①外在评测，Word Error Rate；②内在评测，用困惑度（Perplexity，多用于先期自测）：

最小化 PP(W) = P(w₁w₂...w_N)^{-1/N}
最小化 PP 就是最大化整个句子在模型中的出现概率；开 N 次根号用来做某种关于模型状态空间的归一化补偿；可以使用条件概率展开
问题：①过拟合，测试集和训练集相差很大则效果不好；②很多概率是 0：进行平滑：(1)Laplace 平滑，每个 C_i 都加一，则 $P_{Laplace}(w_i) = (C_i + 1) / (N + V)$ ；

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

Bigram 如上。
加 k 法缺点是由于 0 太多的数据集，非 0 的概率会极大稀释 (2)Good-Turing 平滑法，用 p0 = N1/N
c' = (c_x + 1)N_{x+1}/N_x，其中 N_x 为出现 x 次的词的出现次数。
回避（Backoff）：如果更高阶的 Markov 没有出现，就回退到用低阶的 Markov 过程对概率进行估计
内插（Interpolation）：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）
应用上我们一般用 exp {Σ log p_i} 来算对应的乘法；避免下溢+加快速度
N-gram 优点：容易构建，可以使用平滑来适应新数据；缺陷：只有在测试集与训练集比较相似的情况下表现较好，只能捕捉到较短的结果；神经网络：适应能力强，但训练消耗相对较大

词类标注（POS Tagging）：①基于规则的方法②概率方法 (HMM)
基准方法：无脑选最大类，查表+无脑选，RE 方法
HMM：一些状态 S，一些观测值 O，关于状态 S 的转移概率矩阵，输出概率矩阵 B_i(O_i)，即 S=i 时观测值为 k 的概率，以及初始状态 S 的概率分布

HMM Tagging：隐状态是各个 POS，输出是各个词本身的 HMM；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？②（解码问题）给定观察序列，如何计算最优的隐状态序列？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？定义 a 为状态转移概率，b 为观察概率。
对于①，定义 $\alpha_t(i) = P(o_1...o_t, q_t = S_i | \lambda)$ ，即第 t 个观察值对应的隐状态为 i 时的输出概率，则有递推关系

$$\alpha_{t+1}(j) = \left| \sum_{i=1}^N \alpha_t(i) a_{ij} \right| b_j(o_{t+1}),$$

成立；P(O|λ) = Σ_{i=1}^N α_T(i)；此方法也可以用来反向计算（Backward Algorithm）：先初始化 β_T = 1
对于②，使用 Viterbi 算法，定义 $\delta_t(i) = \max_{j=1}^N P(q_1...q_{t-1}, q_t = i, o_1...o_t | \lambda)$ ，即给定前 t 个观察序列下前 t-1 个隐状态最优，第 t 个状态为 i 的概率，则递推关系如下：

$$\delta_t(i) = \pi_i b_i(o_1), \quad \delta_t(j) = \max_i \delta_{t-1}(i) a_{ij} b_j(o_t)$$

；同时，为了记忆最优状态，设置 ψ_t(j) = argmax_i [δ_{t-1}(i) a_{ij}]（每个 t 都要记录 N 次，最后回溯 t ≤ N 为了加速也可以进行对数化）。
对于③，现在还没有全局最优解，思路大概有 (1) 梯度下降 (2) EM 或者 Baum-Welch 迭代搜索；重估计的

$$\xi_t(i,j) = \frac{P(q_i = S_i, q_{t+1} = S_j, O | \lambda)}{P(O | \lambda)} = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O | \lambda)} = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}$$

一种方法如下：
则 Σ_{i=1}^N ξ_t(i,j) 表示从 S_i 转移到 S_j 的期望总转移边数，Σ_{i=1}^N γ_t(i) 表示从 S_i 转移出去的总期望边数；那么，

$$\gamma_t(i) = \sum_{j=1}^N \xi_t(i,j) \quad \gamma_t(i) = P(q_t = S_i | O, \lambda)$$

可以得到新的估计转移概率和输出概率 π_t = γ_t(i), a_{ij} = Σ_{i=1}^N ξ_t(i,j) / Σ_{i=1}^N γ_t(i), b_j(k) = Σ_{i=1}^N γ_t(i) o_t=w_j γ_t(i) / Σ_{i=1}^N γ_t(i) (v_k 表示在 O_t 观察到符号 k)；<UNK> 在处理中可以 根据形态学猜测其词性 (比如 -s, -able 等)

Forward 就是加起来
重新估计转移矩阵：比如 1-1, 1-2, 用有 1-1 转移的序列概率乘转移次数，1-2 转移的序列概率乘转移次数，两者归一化。
最大嫡马尔可夫链：状态是由观察值和上一个状态生成的，优化可以直接用最大似然估计，给定观察序列之后哪种隐状态的序列概率最高

Pi: Urm 1: 0.9; Urm 2: 0.1

A	Urm 1	Urm 2
Urm 1	0.6	0.4
Urm 2	0.3	0.7

B	Urm 1	Urm 2
Red	0.7	0.4
Blue	0.3	0.6

Blue Blue Red
1 1 1
(0.9*0.3)*(0.6*0.3)*(0.6*0.7)=0.0204

评估：用人来分词，作为金标准（97%左右），和机器结果进行比较，得到 Confusion Matrix

英语核心构成：句子（表达完整的思想），子句（有一个动词），短语（一种词的聚合）

用 CFG 解析文法的问题：①一致关系（名词代词和动词的复数形式应该对应，可以细化语法规产生式，但不够简洁美观，泛化能力不够强）②次范畴化（谓语句和后边的“参数”类型应该满足一些语义上的约束）；对谓语句（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③移位（倒装等特殊句式使动词和宾语的位置发生变化，用 CFG 很麻烦）

依存文法：记录词之间的论旨角色，一般是二元的；没有非终结符。

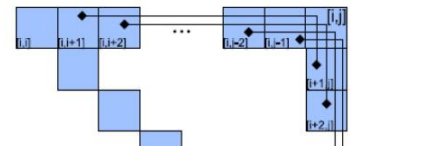
文法库的来源：①手工构造②TreeBanks，根据 POS 过的句子自动生成文法，自动解析，手工修正。

解析文法：DP 方案，有效存储二义句；CKY 自底向上 O(n^3)，Earley 自顶向下 O(n^3)

CKY：文法全部写为 Chomsky 范式 (A->BC 或 A->w)；索引 0,1,...,n 分别代表第一个单词前，第二个，第 n 个单词后；CKY 表格中填写可能的非终结符，计算 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，不是 Chomsky 范式的文法可以重写为 Chomsky 的。

缺陷：可能产生一些无关的成分
Partial Parsing：只处理大的组块（chunk）在哪；基于规则的部分 Parsing：去掉规则集中的递归元素（变成正则文法）；基于 ML 的 Chunking：一个序列分类任务，其把词分为三类“开始，内部和外部”，外部表示其不在任何组块中，内部表示其在某组块中（比

如 I_NP 在 NP 中），开始表示其为某块的开始（比如



B_NP)

概率 CFG：给文法规则都赋一个概率，用来解决二义性，同时模仿人类解析语言的过程；非二义的情况下就是该文法的概率，二义的情况是 n 者相加；解析树的每个 NT 都放有一个概率值；注意，对 A → BC 来说概率其实是 P(A → BC) * P(B) * P(C)，只是 DP 的时候 P(B) 和 P(C) 肯定都已经计算出来了；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
求 PCFG 文法概率：用 Treebank 里面的数据进行统计)

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

或者按确定性语法进行 Parse 后进行重新估计得出概率。由此可以有改进的 maxprob CKY。

PCFG 的问题：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

解决方案：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 (NPNP_{Subject})，但是增加了语法复杂性，并且需要更多数据②中心词（NP 的中心词是名词，VP 的中心词是动词，PP 的中心词是介词），现在在文法符号变成类似 VP(中心词)；由于句子太少，规则不变，PCFG 的概率变成类似 P(r|VP^dumped) = Count(这个规则用在 dump 上) / Count(dump 作为中心词出现次数)；这其实是一种次范畴化；同时，应该加入一种优先级，描述中心词和上层文法之间的选择亲和性。

Parser 评估：采用构成评估（Constituent-level Evaluation），分为覆盖率（Recall，你的解析结果中正确【即有相同节点标签和正确分点的节点】的节点数/treebank 中的相应正确节点数）和精度（Precision，你的解析结果中正确的节点数/你的解析结果中的节点数）
Recall = 2/5; Precision = 2/4;

交叉括号（Cross Brackets）：treebank has (X Y Z) and candidate has (X (Y Z)) 的情形；这个应该作为客观的函数来最小化

上述方法的缺点：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

Candidate	gold
X:a	X:a
Y:b	Z:b
Z:cd	V:cd
--	Y:bcd
W:abcd	W:abcd

况；将所有节点一视同仁，而不是更关注核心的语义关系

四种向量语义模型：①稀疏向量表示(1)以互信息（Mutual-information）为权重的单词关联矩阵②稠密向量表示(1)奇异值分解和潜在语义分析（Latent Semantic Analysis）(2)各种神经网络模型（Skip-grams, CBOW）(3)Brown clusters

词项-文档矩阵（Term-Document Matrix），tf(t,d) 表示词 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（word-word）矩阵：又叫词-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多，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

直接计数（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，比如“the”和“of”一起出现的频率非常高，但是词义不一定是最贴近的；

为了修正，用 PPMI（点间互信息，Positive Pointwise Mutual Information）来进行修正：

PPMI(X,Y) = $\log \frac{p_{ij}}{p_i p_j}$ ；f_{ij} 定义为 w_i 出现在以 c_j 为上下文的中心词当中的频数，则 p_{ij}, p_i(i) 和 p_j(j) 均容易求得。

PMI 会向出现较少的事件倾斜，解决方法：①给稀有事件更高概率：其中 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②用加一平滑法（差不多）

相似性度量：采用向量夹角来归一化内积；

Vi 和 Wi 分别是 v 和 w 在上下文 i 的 PPMI 值；因为 PPMI 非负，故 cos 值在 0-1 之间

其它定义上下的方法：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

图 10.10 展示了 PPMI 值的计算过程。图中包含两个表格：一个是“Unit product”表格，另一个是“Unit vectors”表格。在“Unit product”表格中，列出了词 v 和 w 在不同上下文 i 中的 PPMI 值。在“Unit vectors”表格中，列出了词 v 和 w 在不同上下文 i 中的 PPMI 值。图中还包含了一个公式，用于计算 PPMI 值。

vi is the PPMI value for word v in context i
wi is the PPMI value for word w in context i

Vi 和 Wi 分别是 v 和 w 在上下文 i 的 PPMI 值；因为 PPMI 非负，故 cos 值在 0-1 之间

其它定义上下的方法：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;于是,一个向量用 Relation*|V| 个关系进行修饰,比如"subject-of,absorb"这种;另一种方法是將上下文定义成"counts of words that occur in one of R Dependencies",而不是直接用滑动窗口,比如" M("cell","absorb") = count(subj(cell,absorb)) + count(obj(cell,absorb)) + count(pobj(cell,absorb)), etc."

PPMI的替代指标: tf-idf (term-frequency 词的频率和 inverse document frequency), 其中 df_i 表示 number of documents with word i, N 是文档的总数;则 成立。

PPMI 方法本身构造的是稀疏向量
稠密向量的优势: 稠密向量更容易捕获"同义"
构造稀疏向量的方法: (1)SVD, PCA, 成分分析; 压缩后的词项-文档矩阵可以作为一种潜在语义分析, 而 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, 优势是比 SVD 快很多倍, 并且可以调 word2vec 包; Skip-grams 预测临近的 2C 个词: (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)

训练目标是让重建旁边词的损失最小 (用自己这个词取猜旁边的词=skipgram), 同时让输出更不像随机选择的 k 个单词; Skipgram 可以证明和 PMI 有关
(3)Brown clusters: 一种根据词的前后的词的聚合进行聚类的算法; 一开始每个词都在一个等价类中, 然后让相似的上下文的词进行 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。
Lexical Semantics (词汇语义学)
Word sense (词义)

-Homonyms (同形/同音异义词); Polysemy (多义关系)
WordNet
是一个反应词汇之间关系的数据库, 和传统词典不同: Meaning-based 遍历; 结构: 语义关系+同义词集

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 可以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

Wordnet 可以看作一个稀疏数据库, 是单词和同义词集之间的真值表 (属于为 1, 不属于为 0)

Wordnet 可以被 POS 限制, 分为 Paradigmatic (组合) relations (within POS) 和 syntagmatic (聚合) relations (across POS); 主要有 4 类彼此不连接的 wordnet: 动词、名词、形容词、副词

Wordnet 存在的问题: 1. 关系未加权 2. 非常稀疏 3. 关系可能不直接 4. 未实现相互唤起 5. Types and roles 未分离

Wordnet 是一种词汇上的本体论

Word Similarity
同义是一种严格的 0/1 关系, 而相似性是两个单词的某个语义相似即可, 更宽松; 相似性和关联性不同

基于词库的相似性判断

1. Path-based 相似性
c1, c2 是两个含义节点
 $Pathlen(c1, c2) = 1 + edges(c1 \text{ to } c2)$
 $Simpath(c1, c2) = 1/Pathlen(c1, c2)$
Wordsim(w1, w2) = 两个词最相近的两个含义的 simpath
问题: 每条边的权重一样, 层次越高越抽象; 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

2. Information Content similarity
 $P(c)$ 是在语料中随机抽取一个词, 属于 concept c 的概率, 特别地 $P(\text{root}) = 1$
Self-information: $l(w) = -\log_2 P(w)$ 表明我们从事件发送得到信息量的多少
 $Information \text{ Content} = -\ln P(c)$
最小公共包含: $LCS(c1, c2) =$ 同时包含 c1, c2 的最低层的节点
 $Sim_resnik(c1, c2) = -\ln P(LCS(c1, c2))$
 $Sim_lin(c1, c2) = 2 \ln P(LCS(c1, c2)) / (\ln P(c1) + \ln P(c2))$

分布式相似性判断
Inside words

词语本身语义, 使用"主题"表达, 加入限制? 局限性: 语言太灵活了。
在特定领域 (e.g. 生物) 使用 wordnet 效果更好。
语义消歧 (Word Sense Disambiguation)
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: 数据量太大, 关系稀疏, 不能使用针对特定词语集合的分类器
监督学习:

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: 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), 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, 出现处记 1, 否则为 0)

分类器: 可以使用朴素贝叶斯+平滑、基于规则的决策

对测试排序: 可以使用 $P(\text{sense } 1 | \text{feature}) / P(\text{sense } 2 | \text{feature})$ 来评估两种意思之间的区分程度

内在评价标准: 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: 选择和词典中 gross 和例句重合最多的意思, 给词语加权 $idf_i = \log(N/df_i)$ N 为文章总数, df_i 为单词 i 在几篇文章中出现过 (在越少文章中出现说明越重要, 否则不那么重要), 评分变成加权和中

图的观点: 选取最中心的意义

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:

使用已知的固定搭配; 或者含义基本完全一致的一篇语料作为种子, 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合中, 重复步骤

问题: 可能需要对每个歧义词训练一个分类器, 需要选择合适的训练集合

组合语义学 (Compositional Semantics)
不需要准确知道每个词语含义, 只需大致知道整体意思

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

2. 使用信息抽取

信息抽取

命名实体识别 (NER)

基于规则

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

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 subtring)

IOB 编码: 对于 N 类需要 $2*N + 1$ 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 表示实体的继续, O 表示不属于任何类

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

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

评价: 准确率、召回率、F1

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. 命名实体所在的语言环境 (产生式、依赖式等)

半监督 bootstrapping: 从已知有关系的实体出发, 在语料中提取更多关系特征, 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

模板填充
Cascades of transducers

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
信息抽取的准确率不高: 错误会传递 (错误的命名实体识别产生错误的关系)

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。

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: 决定回答类型; 一般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文章抽取: 先选择带有所有重点点的文章, 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, 增添或者去掉若干重点词

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. 窗口中不相关词数量

潜在答案排序: 可以从维基百科或者 wordnet 中收集潜在答案

问题表述可能不同, 解决方案: 根据重写规则重构问题 (变成填空问题) -> 使用搜索引擎收集答案 -> 建立 N-gram 模型 (权重, 改写后的问题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

倒排索引: 文章 ID+文章中查询词出现的次数; stop list: 去除主题无关词 (of, a 等); stemming: 关注词干

特定型检索:

向量词空间: 文章和查询语句被表示为向量
 $D = (t_1, t_2, \dots, t_n)$ 表示 n 个词语类型, 每个词语类型在文章中出现的次数, 可以通过点乘来判定相似性

上述方法对所有词的权重相同, 考虑加权 (局部权重 (表达意义) 和全局权重 (区分性))

局部权重: 词频的函数; 全局权重: $idf_i = \log(N/df_i)$

TF-IDF Weighting: 由频率推导得到的权重 * 倒排索引

通过向量空间中的方向来衡量相似性和相关性 (余弦相似度)

大体步骤: 根据关键词找到所有相关文章 -> 将查询语句和文章编码 -> 用余弦相似度衡量相关性并排序

摘要(summarization)
单篇文章摘要: 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: $TF * IDF$ 加权, 特征选择: 线索词、开头或者结尾词、句子位置、主题词频

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

主题词: 使用 log 概率并设置值域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

也可以使用机器学习方法抽取。

机器翻译

1. 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) 为平均得到一致评价的可能性 (比如共 5 个评分点, $P(E) = 0.2$)

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、查找和删除操作数)/(参考长度)

BLEU (n-gram 准确率): 短句惩罚, 不能连续使用相同词两次, 计算公式 $BLEU = \min_{i=1,2,3,4} \left(\frac{\text{output length}}{\text{ref length}} \right) \left(\prod_{i=1,2,3,4} \left[\frac{\text{precision}_i}{\text{precision}_i} \right] \right)^{1/4}$

其中 $i=1,2,3,4$ 使用代表 i-gram 的准确率 (output 中的词组是否在 ref 里出现), 也可以使用多个 ref 增大可信度

3. 基于统计的机器翻译

大体思路: 输入 -> 翻译模型 (给出候选 p(f|e))

-> 语言模型 (确定翻译的合理性 p(e)) -> 解码输出 ($\text{find } \text{argmax}_p(p(e)p(f|e))$)

需要从语料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, 并提升它们的概率;

EM 算法: 1. 找出所有可能的以已知语言词汇为条件的未知绑定词汇的概率, 并假设等可能 2. 计算可能的组合绑定情况的概率并标准化 3. 对一组绑定, 寻找出现的证据, 并乘响应的概率, 标准化 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 3. 解码; 好处: 多对多映射; 减少单个词语的歧义性

换位概率 $d(a_i - b_i - 1) = \alpha^{(|a_i - b_i - 1|)}$, 其中 a_i 是第 i 个英语单词翻译成的目标语言后所在的位置, $b_i - 1$ 是第 i-1 个对应的位置

$P(f|e)$ 是对应词组翻译概率 乘 换位概率

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, 之后使用概率最大的单词对齐组合。解码过程同样可以使用动态规划或者 A* 搜索。

基于机器学习的翻译

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

输入输出使用 word embedding, 输出概率使用 softmax 标准化, 测试时选择每次选择概率最大的单词输出

局限: 1. 受限于词库大小, 希望扩大词汇范围 (拷贝技巧, 加入 <unknown> 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)