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（整体理解分析）**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），[^Aa]（不是 A 或者 a），[a^]（a 和^，因为只在[后第一个有效），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，加在 RE 的最前（注意在[]的外面）），匹配所有开头/结尾的字符。**有限状态自动机 Finite State Automata 与正则：**状态 Q，字母表 Sigma，初始状态，终极状态，转一函数。有限状态机定义了一种形式语言（Formal Language）。Generator & Acceptor

错误的类别：假阳性（1 类，不该匹配却匹配，降低假阳性提升精确度），假阴性（2 类，该匹配却没有匹配，降低假阴性提升覆盖率）**DFA:**同一个状态/输入只能通向一个状态，**ND-FA：**同一个状态/输入可能通向多种状态，需要遍历所有状态，支持通过空串 epsilon 转向另一种等价状态（tape 不前进）**分词:**分词，规范化，分句。**词元（Lemma）**：同一个词干（stem）和词性（part of speech），大致相同的词义(cat=cats)

词形（Wordform）：词的表面形式（把单复数啥的变化都加上）(cat!=cats) **词型（Type）**：一个单词

词例（Token）：词型在文章中的一个实例,**N 词例数>>V 词汇量，分词**时的歧义：Finland's, state-of, what're, PhD,法语，德语复合词；**方法**：空格分词，字符分词，subword-toeknization（BPE，Wordpiece，…）**Learner**:从语料库学习词汇表，**Segmenter:**分词

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（Morphemes）构造.**语素（Morpheme）**：词干（Stem,核心意思）和词缀（Affix，附加/修改意思）

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**，不需要词汇表，基于规则去词缀，不保证产生真实词干，但不影响 IR。（ization->ize->()）**切分（Segmentaion）：**不能单纯按照标点/空格切分句子/单词，？！没有歧义，“.”有很多歧义

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（词为单位）；**命名实体**（Named Entity）和（Entity Coreference）识别；**Levenshtein** 插入和删除代价为 1，替换代价为 2。**一般**代价都为 1

计算方法：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)

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

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

初始化 $D(i,0)=i, D(0,j)=j$

如果 Xi=Yj，则认为在这里是对齐的；为了跟踪对齐情况，用 ptr 维护一个箭头数组跟踪 insert(LEFT)，delete(DOWN)，subst（DIAG）；时间以及空间复杂度 O(nm)，Backtrace 输出 O(n+m)

带权最小编辑距离（Weighted Edit Dist）：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(有些字母对更容易打错；有些插入/删除更容易发生)

语言模型：$P(w_n|w_1,...,w_{n-1})$，用链式法则计算句子（单词序列）的概率。**但是由于数据不足，假设 Markov 性质成立（$P(x|a,b,c,d...)=P(x|a)$）**（二元情况，也可以 N 元）；Bigram 即为 Markov 链，预测用 MLE 选出最大可能的下一个词，

即$P(w_i|w_{i-1})=\frac{c(w_{i-1},w_i)}{c(w_{i-1})}$

句子生成：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-测试集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）：

$$PP(W)=P(w_1w_2...w_N)^{-1/N}$$

$$PP(W)=\sqrt[N]{\prod_{i=1}^{N}\frac{1}{P(w_i|w_{i-1})}}$$

最小化 PP 就是最大化整个句子在模型中的出现概率（希望测试集与训练集类似）；开 N 次根号用来做某种关于模型状态空间的归一化补偿；越小越好

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 Ci 都加一，则$P_{Laplace}(w_i)=(c_i+1)/(N+V)$;

Bigram 如上，$c_i'=(c_i+1)N/(N+V)$

加 k 法缺点是对于 0 太多的数据集，非 0 的概率会极大稀释，适合 0 不多的数据

$$P(w_{n-1}w_n)^*=(c(w_{n-1}w_n)+1)/(c(w_{n-1})+V)$$

(2)Good-Turing 平滑法，估计没出现的概率用$P_0=N_1/N$, $c_x^*=(c_x+1)N_{x+1}/N_x, p_x'=c_x'/N$，其中$N_x$为出现 x 次的词的出现次数。对于任何一个出现了 c 次的类别，都假设它出现了 c'次.

回退（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

$$P(z|xy)\approx P(z|x)\approx P(z)$$

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；先固定 N-gram prob，再找到最优权重使 held-out dataset prob 最大，可以采用搜索算法找到最优权重（比如 EM 算法）

$$P(z|xy)'=\lambda_1P(z|xy)+\lambda_2P(z|x)+\lambda_3P(z)$$

词汇外词汇(OOV): 不能用 GT，因为不知道总数，依旧全转化为一个<UNK>来看

应用:我们一般用$\exp\{\sum\log p_i\}=\prod p_i$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**：非线性，前/反向传播，SGD 适应能力强，但训练耗相对较大，词嵌入，词向量，softmax 需要对所有单词求和

词类标注（POS(Parts of speech 词类) 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 正则法。**HMM**:一些状态 S，一些观测值 O，关于状态 S 的转移概率矩阵A_{ij}，输出概率矩阵$B_i(k)$，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？遍历所有可能的状态序列，乘上对应的状态->输出序列得到概率，累加。

②(**解码问题**)给定观察序列，如何计算最优的隐状态序列？选出最大概率对应的状态序列$t_1^n=argmaxP(t_1^n|w_1^n)=argmaxP(w_1^n|t_1^n)P(t_1^n)\approx argmax\prod P(w_i|t_i)P(t_i|t_{i-1})$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$\alpha_t(i)=P(\boldsymbol{o}_1...\boldsymbol{o}_t, q_t=S_i|\lambda)$，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

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

成立；P(O|λ)=$\sum_{i=1}^{n}\alpha_T(i)$；此方法也可以用来反向计算（Backward Algorithm）：先初始化βT = 1

对于②，使用 Viterbi 算法，定义$\delta_t(i)=\max P[q_1...q_{t-1}, q_t=i, \boldsymbol{o}_1...\boldsymbol{o}_t|\lambda]$，即给定前 t 个观察序列下前$t^t$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：

$$\delta_t(j)=\max_{1\le i\le N}\left[\delta_{t-1}(i)a_{ij}\right]b_j(\boldsymbol{o}_t),$$

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

同时，为了记忆最优状态，设置$\psi_t(j)=\text{argmax}[\delta_{t-1}(i)a_{ij}]$（每个 t 都要记录 N 次，最后回溯）；为了前递也可以进行对数化。

$$\delta_1(i)=\pi_i\,b_i(\boldsymbol{o}_1),$$

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；重估计的一种方法如下：则$\sum_{t=1}^{T-1}\xi_t(i,j)$表示从 Si 转移到 Sj 的期望总转移边数，$\sum_{t=1}^{T-1}\gamma_t(i)$表示从 Si 转移出去的总期望边数；那么，可以得到新的估计转移概率和输出概率$\pi_i=\gamma_1(i)$，$\bar{a}_{ij}=\sum_{t=1}^{T-1}\xi_t(i,j)/\sum_{t=1}^{T-1}\gamma_t(i)$，$\bar{b}_j(k)=\sum_{t=1\text{ and }\boldsymbol{o}_t=v_k}^{T-1}\gamma_t(j)/\sum_{t=1}^{T-1}\gamma_t(j)$（$v_k$表示在$\boldsymbol{O}_t$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

$$\xi_t(i,j)=\frac{P[q_t=S_i, q_{t+1}=S_j, \boldsymbol{O}|\lambda]}{P(\boldsymbol{O}|\lambda)}=\frac{\alpha_t(i)a_{ij}b_j(\boldsymbol{o}_{t+1})\beta_{t+1}(j)}{P(\boldsymbol{O}|\lambda)}=\frac{\alpha_t(i)a_{ij}b_j(\boldsymbol{o}_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(\boldsymbol{o}_{t+1})\beta_{t+1}(j)}$$

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(对一个状态出发的所有转移归一化)。

最大熵马尔可夫链：$p(c|x)=(1/z)\exp(\sum w_if_i)$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。**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，可以找出最常见错误。**POS 在中文中：**更迷惑，没见过的字有问题，神经网络最佳。

句法和语法：CFG(Context-Free Grammar)上下文无关语法：规则或者产生式，捕捉顺序和组成性

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）S->NP VP,

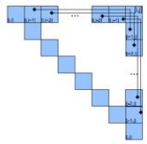
S->VP, S->Aux NP VP，递归: N->N PP

用 CFG 解析文法的问题：①**一致关系**（名词代词和动词的单复数形式应该对应，可以细化语法产生式，但不够简洁美观，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，（find 需要 NP），(补语)，V-with-NP-complement，**VP -> V-with-NP-comp NP**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，需要考虑很多基础语法）**依存文法：**记录词之间的论旨角色（thematic roles），一般是二元的；没有非终结符。适合印度、欧洲。United <= canceled (NSUBJ)**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,很多冗余。

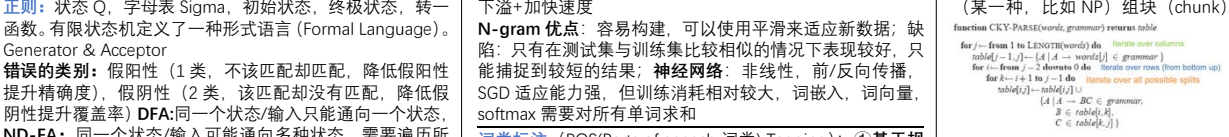
剖析文法：DP 方案：有效存储二义句子，避免重复工作，在多项式时间内完成指数任务。

Earley 自顶向下 O（n^3）：从树库出发，有可能有和输入无关的树；**CKY 自底向上** O(n^3)：从输入出发，有可能有非法树。**CKY：**文法全部写为 Chomsky 范式（CNF）(A->BC 或 A->w(两个非终止/一个终止))；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。

缺陷：非二元语法需要转成二元的 CNF，不是 Chomsky 范式的文法可以重写成 Chomsky 的，可能产生一些无关的成分，S=>ABC，S=>XC, X=>AB, X 不在其他地方出现。



Partial Parsing:速度较快，只处理大的（某一种，比如 NP）组块（chunk）



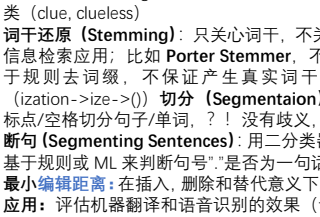
在哪；**基于规则的部分 Parsing**：去掉规则集中的递归元素（变成正则文法），POS=>句法短语=>更大的动词/名词组=>句子层面规则

基于 ML 的 Chunking：一个序列分类任务，按序列顺序处理，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），这样就能把所有 NP Chunk 找出来

概率 CFG：对于特定领域的语言以及难以理解的语言高有效。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；解析树的每个文法都放有一个概率值，**树的概率**是树中所有规则概率相乘，**句子的概率**是所有可能树的概率之和。**求 PCFG 文法概率：**用 Treebank 里面的数据进行统计）

若没有 TB，假设有确定性语法进行 Parse 后进行重新统计得出概率。对 $A\to BC$ 来说概率其实是 $P(A\to BC)*P(B)*P(C)$，只是 DP 的时候 P(B)和 P(C)肯定都已经计算出来了；如果有 B 和 C（来源于不同的分点，但是都合法）则取最大概率的

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



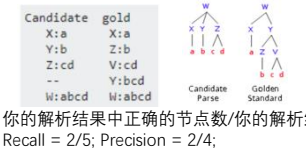
那种。由此可以有改进的 maxprob CKY。

结构依赖：

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（$\text{NPNP}_{\text{Subject}}$），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），现在文法符号变成类似 VP(中心词)；由于句子太少，规则不变，PCFG 的规则变成类似 VP(dumped)=>V(dumped)NP(sacks)PP(in), P(rule|VP^dumped^sack^in),以中心词为条件的规则的概率 P(rule|VP^dumped) = Count(这个规则用出现次数)/Count(VP^dump^rule 出现次数)；这其实是一种次范畴化捕捉到了中心词和语法的关联度；**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关联度，比较 dumped 作为 head 且有一个 PP 的 head 是 into 的概率和 sacks 作为 head 且有一个 PP 的 head 是 into 的概率即可选择正确的。

Parser 评估：构成评估（Constituent-level Evaluation）：句子水平太粗糙了，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划分的节点】的节点数/treebank 中的相应 constitent 节点数）和**精准度**（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，统计多少对括号穿过 tb 的括号。**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，一个节点的错误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

语义：单词含义，三种向量语义模型：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）**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-上下文（word-context）矩阵**，a_ij 表示在段落里每次单词 i 的±4（或者别的数）的范围内出现单词 j 的次数之和；矩阵会很稀疏，并且窗口大小和目标有关（1-3 窗口越小语法信息越多，4-10 窗口越大语义信息越多）两个行相近则两个词相近。

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：$PMI(X,Y)=\log_2 P(x,y)/P(x)P(y)$，$PPMI=\max(PMI,0)$；负数会很麻烦。 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*)和 p_(*j)均容易求得。Pi*对行求和，P*j 对列求和。

PMI 会向出现较少的事件倾斜，解决方法：①给稀有上下文单词 c 更高概率：$P(c)=>P_\alpha(c)=count(c)^\alpha/\sum count(c)^\alpha$，其中__,α为一个预先给定的值（如 0.75），c 很小时会有提升。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，点乘除以；向量长度，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

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

其它定义上下文的方法：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；

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；$w_{ij}=tf_{ij}idf_i, idf_i=\log N/df_i$

PPMI 方法本身构造的是**稀疏向量**，学到的是**稠密向量**，稠密向量优势：稠密向量更容易捕获"同义",更容易在神经网络中 tune。**构造稀疏向量的方法**：(1)**SVD(奇异值分解)：**$X^{w*c}=W^{w*m}S^{m*m}C^{m*c}$，S 是对角阵，中间是奇异值；W 每一列代表新的空间中的一维，相互正交；每一行代表一个单词。**LSA**：对 Term-Doc 的 SVD，使用 top-k 奇异值而非 m 个。压缩后的 Term-Doc 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(可尝试去掉 top1-50 维），其他方法：PCA，成分分析(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，预训练 embedding；Skip-grams 预测临近的 2C 个词，同时也能获得词向量。会产生输入向量与输出向量。输入层（1-hot, V）-预测层（词向量, d）-输出层（和上下文的概率, V,使用 softmax+similarity）**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(负采样)；

$$loss=\log\sigma(c\cdot w)+\sum E\log\sigma(-w^{'}\cdot w)$$

Skipgram 可以证明和 PMI 有关，**问题:**统一单词不同意思会有同一向量，需要训练上下文 based 向量。

Transformer：自注意力机制；LM 只使用单向 embedding，**BERT:**bidirectional 能让单词看见他们自己。**Masked LM:**随机替换/遮罩/不变需要预测的单词去训练。**Input:**是单词，segment，position 的 embedding 叠加。**模型:**Megatron-LM, T5, GPT3,XLNet, RoBERTA

Lexical Semantics（词汇语义学）Word sense（词义）

Words/lemmas/stems,celebrations/celebration/celebrate **词间关系-Homonyms**(同形 homograph/同音 homophone 异义词)；**Polysemy**(多义关系)语义关系/借喻; zeugma test 如果奇怪则多义；**Synonyms** 同义词，所有情况下可以互换；**Antonyms** 反义词; **Hyponym** 下位词更具体，**Hypernym** 上位词更抽象

Meronym 部分词**; Holonym** 整体词 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 sense 之间关系 **同义词集：**每个概念都有同义词集以及注释 gloss，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（最高级上位词）

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 **数据库视角:**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）**局限性** Wordnet 被 POS 限制，大部分 link 都在某一种 POS 内(因为必须可替换)，分为 Paradigmatic(聚合)relations(within POS)和 syntagmatic (组合) relations (across POS)；主要有 4 类彼此**未连接的 wordnet**：动词、名词、形容词、副词 **网络问题**：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 和 roles 未分离 dog->poodle(type), dog->pet(role) 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(可能相关不相似)

1. Path-based 相似性（同义词词库，在上位词等级关系中的位子相同）c1, c2 是两个含义节点()

$pathlen(c1,c2)=1+shortest\_path\_edges(c1,c2)$

$simpath(c1,c2)=1/pathlen(c1,c2)$，自己对自己:1

$Wordsim(w1,w2)=$ 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(root) = 1，层次越低 p 越小。$word(c)$ =c 的所有孩子的集合，则$P(c)=\sum count(w)/N, w\in word(c)$

Self-information, suprisa 惊异度表明我们从事件发送得到信息量的多少 $l(w)=-\log_2 p(w)$ Information Content $IC(c)=-\log p(c)$

最小公共包含/最有信息包含：MIS/LCS(c1, c2) = 同时包含 c1, c2 的最低层的节点

$$sim_{resnik}(c1,c2)=-\log p(LCS(c1,c2))$$
$$sim_{lin}(c1,c2)\propto IC(common(A,B))/IC(des(A,B))$$
$$=2log\,P(LCS(c1,c2))/(log\,P(c1)+log\,P(c2))$$
$$sim_{jiangconrath}(c1,c2)=1/(logP(c1)+\log P(c2)-2logP(LCS(c1,c2))$$
$$sim_{eLesk}(c1,c2)=\sum overlaps(gloss(r(c1)),gloss(q(c2))$$

$r,q\in sense, overlaps$ =按重叠的词计算 score=n^2,n 为长度, Differ = Description-Common

除了 Wordnet 还有 MeSH(药学方面)

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）选出一个词在语境中的意思。**对文章中特定词语集合的 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 window size 出现的词的 identity/POS 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, count)】**分类器：**可以使用**朴素贝叶斯**($c=argmax P(d|c)P(c)$参数过多，需要数据过大；**多项式贝叶斯**,将 P(d|c)中 d 的每个 feat 看成关于 c 独立($c=argmax\,P(c)\prod P(x|c)=argmax\log P(c)+\sum logP(x|c)$),计算频率时可以把所有文档合并计算，为防止缺少样本可采用+1 平滑；未知词以及停止词(the,a)删掉不考虑/（还有：基于规则决策/NN/SVM/k 近邻)**二元多项式贝叶斯:**出现比次数更重要，每篇 doc 中 c 出现就算 1 次，删除每篇 doc 中重复的 w 后，令 textc=所有有 c 出现的 doc 集合，计算 P（w|c）=textc 中 w 出现次数的+alpha 平滑。**决策表:**一次只针对一元决策，排序每个可能的 rule 按照准确度，选择 top-k 作为决策表 **对测试排序:**可以使用 P(sense 1| feature)/P(sense 2 | feature) 来评估两种意思之间的区分程度。**外在评价**：任务驱动；**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（标注了 sense）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，否则不重要），评分变成加权和 sum idf(w), w is the overlap.**半监督学习问题:**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（手工/每个搭配/文章仅有一种意思），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，用每个搭配仅一种意思来筛选结果，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

信息抽取:命名实体识别（NER）**基于规则:**(名字表)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，朴素的分辨 substring 是否被抽取。**IOB 编码**：对于 N 类需要 2*N + 1 个标签，对于每一类：B 表示属于某一类实体的开始，I 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，2.POS, 3.previous label…**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; 3.NN：双向 LSTM+CRF 输出层。**评价：** Precise=TP/TP+FP 、Recall=TP/TP+FN 、F1=2PR/P+R **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 内的词语，相对位置，bag of word） 3. 命名实体所在的语法环境（POS 产生式、类型依赖式，chunk 内 POS 路径等）**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:**根据文章填充信息模版。**方法: 1.**Cascades of transducers

2.ML: 1. One seq classifier per slot 2. One big sequence classifier 问题：重应用，跟语言相关，IOBES (end, singleton)，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，名称变化大，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+问题规范化 -> 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。**问题规范化**：提取出若干无关联重点词（bag-of-word），规范化问题。**文章抽取:** 先选择带有所有重点词的文章，再根据得到的文章数是否达到"阈值"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. 窗口中关键词出现的顺序与问题一致的数量 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：**匹配的术语数目、顺序、平均修改距离…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，解方案：**根据重写规则重构问题（变成成数据）-> 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（多个答案合并分数更高）。**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)）。**更难的问题:**答案不固定，需要多种信息来源/材料

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 = (t1, t2, ⋯, tn) 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，**局限:**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/n_i),n_i:有 i 词语的文档数 **TF-IDF Weighting：**由频率推导得到的权重 * idf，**缺点：偏好长文章（点乘局限）**通过向量空间中的方向来衡量相似性和相关度（余弦相似度）$sim=sum(wil*wij)$

大体步骤：根据关键词找到所有相关文章 -> 将查询语句

$$sim(qk,dj)=\frac{\sum_{i=1}^{N}w_{i,k}\times w_{i,j}}{\sqrt{\sum_{i=1}^{N}w^2{}_{i,k}}\times\sqrt{\sum_{i=1}^{N}w^2{}_{i,j}}}$$

和文章编码 -> 用余弦相似度衡量相关性并排序

摘要(summarization)informative/indicative, extractive/generative 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：TF*IDF 加权，筛去 stop word，近似词只算一次。**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；**篇章凝聚；**

主题词：使用 log 概率并设置阈值判断一个词（是否有说明性/在 input 和背景中概率相同/不同）能不能作为主题词，包含主题词较多的句子被抽出。也可使**用机器学习**方法抽取，用手工摘要，配合标注每一个句子与原文的关系进行训练。

聊天机器人:聊天/助理；**人类对话性质**:轮流，打断，判断插话；**言语行为:**断言/指令/承诺/致谢；**Grounding:**让别人知道你知道;**结构:**adjacency pairs, subdialogue, presequences; **交换主动:**对 NLP 很难，推理：也很难。**规则驱动:**查找关键词，寻找 rank 最高的规则（最具体），变换生成回应 **词汇库驱动:** 信息提取: response(q,c)=argmax q•r/|q||r|, r in C/生成

机器翻译; **任务/框架驱动:**GUS/dialogue state，结合 rule 填写 frame 中的槽(正则表达式）并且做数据库查询;**pro:**准确，在狭窄领域可全覆盖，**con:** recall problem， 花销大且慢

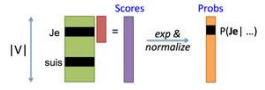
Dialogue state: 对话状态追踪，对话策略, NLU: 解析对话中的 slot filler（BIO 标注->ENR->normalize），NLG: 生成回应 **订正：**很难; **对话策略选择**:基于现在状态，和上一次输入输出; 用**自信**判断是否确认/拒绝; **NLG:** 内容规划以及**句子实现，用占位符替换词语** **评估：chatbot:** 人工，**task-based: 自动/slot error rate**

翻译 1. **基于概率和规则**；2.**评价指标**

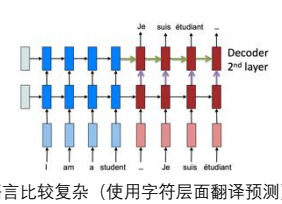
1. **人为评价**：忠实度、流畅度（不同人的感受和评价不同）；2. **自动评价指标：**给定机器翻译和人的参考翻译，要求给出两者之间的相似性 **WER**(Word error rate) = (替换、查找和删除操作数)/(参考长度)**BLEU**(n-gram 准确率)：n-gram 在参考中的出现可能性，短句惩罚，不能连续使用相同词两次，计算公式 $BLEU=min(1,(output\,length)/(ref\,length))(\prod_i 1-4\,precision\_i)^{1/4}$ 其中 i=1,2,3,4 使用代表 i-gram 的准确率（output 中的词组是否在 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(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；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：**对不同句法有惩罚；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

基于机器学习的翻译 RNN：直接计算 p(target | source) 概率，进行端到端训练 h=sig(Wx+Wh)输入输出使用 word embedding，encoder, decoder 各两层，输出概率使用 softmax 标准化。



• Hidden states ↦ scores ↦ probabiliHes.



训练时在 decoder 每次输入模版翻译，测试时选择每次选择概率最大的单词作为输出以及下一次 decoder 输入。**Loss：输出的-log(p)之和**

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 <unknow> 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）

LSTM：用 cell 存储信息。