第二节网络爬虫 反爬虫策略 (1):User Agent 利用访问网站时浏览器发布的 Request Headers 信息中的 User-Agent 信息，判断用户使用何种方式浏览。爬虫往往 U-A 部分为空应对策略：利用 Python 的 Request 库允许用户自定义请求头信息的手段在请求头信息中将 User-Agent 的值改为浏览器的请求头标识，从而绕开反爬虫机制常见反爬虫策略（2）：IP/账号访问次数/频率 • 通过限制特定 IP 地址/账号访问频率和次数进行反爬 • 其本质在于判断浏览行为是否是人类行为 • 应对手段：• 构造 IP 代理池，然后每次访问时随机选择代理 • Github 中有相关服务，通过各种提供免费 IP 的网站来提供代理池 • 每次爬取行为后间隔一段时间 • 注册多个账号以保障数据收集 • 挑战：账号本身的成本问题、账号被查封的危险常见反爬虫策略（3）：验证码 • 通过各类验证码，判断浏览者属于人类还是机器 • 应对手段：• 简单的字符识别：基于机器学习与模式识别相关技术 • 复杂的逻辑推理：人工辅助破解验证码是一把双刃剑，在抵御爬虫的同时也在伤害用户体验常见反爬虫策略（4）：动态网页 • 从网页的 url 加载网页的源代码之后，会在浏览器里执行 JavaScript 程序 • 网页内容由脚本加载，而直接抓取则只能得到空白页面 • 此类情况在抓去在线播放的音视频文件时尤为常见 • 应对手段：• 核心思路：模拟调用请求 • 使用审查元素分析 ajax 请求，如此循环直到获得包含数据信息的 json 文件常见反爬虫策略（5）：蜜罐技术 • 网页上会故意留下一些人类看不到或者绝对不会点击的链接。由于爬虫会从源代码中获取内容，所以爬虫可能会访问这样的链接 • 只要网站发现有 IP 访问这个链接，立刻永久封禁访问者，从而难以继续爬取 • 应对手段：• 核心思路：干涉爬虫路径 • 通过工具库判断页面上的隐含元素，使爬虫避开这些元素，可以部分回避蜜罐的诱导常见反爬虫策略（6）：用户权限限制 • 不同类型/级别的用户给予不同的内容权限 • VIP、SVIP、蓝钻、红钻、绿钻、各种钻……• 应对手段：• 氪金，就可以变强其他的反爬虫策略 • 不同的网页结构：每个相同类型的页面的源代码格式均不相同 • 同样也是双刃剑：增加爬取难度的同时降低用户浏览体验 • 多模态的呈现方式：文字转为图像或视频 • 应对策略：OCR、语音识别、图像/视频标签技术第三节网页文字处理 近似重复文档的检测方法：指纹表示法 • 1）对文档进行分词处理，并进行 n-gram 组合 • 2）挑选部分 n-gram 用于表示这一文档 • 3）对被选中的 n-gram 进行散列 • 4）存储散列值作为文档指纹基于匹配的分词方法 • 又称机械分词方法，它按照一定的策略将待分析的汉字串与一个"充分大的"机器词典中的词条进行匹配，若在词典中找到某个字符串，则匹配成功。• 常用的机械分词方法 • 正向最大匹配分词（FMM）• 反向最大匹配分词（RMM）• 双向最大匹配分词（BM：FMM+RMM）• 最少切分分词（最短路径分词）基于匹配分词的一般模型 • 对于机械分词，可以建立一般模型，其形式化表达如下：• ASM(d,a,m)，即 Automatic Segmentation Model。其中：• d，表示匹配方向，+1 为正向，-1 为逆向 • a：每次匹配失败后增/减字符数，+1 为增字，-1 为减字 • m：最大/最小匹配表示，+1 为最大匹配，-1 为最小匹配 • 例如，ASM（+, -, +）即正向减字最大匹配（即 FMM 方法）正向最大匹配分词 • 从左至右尽可能查找最长的词，直到当前字符与已经处理的字符串不构成词，输出已经识别的词，并从识别出来的词后面接着继续查找下一个词。• 分词速度较快，但错误率较高（约 1/169）反向最大匹配分词 • Reverse（也作 Backward）Maximum Matching method，RMM。• 从右至左尽可能查找最长的词，直到当前字符与已经处理的字符串不构成词。• 统计证实 RMM 分词效果更好（错误率约 1/245)。双向最大匹配分词 • 综合比较 FMM 与 RMM 两种方法的切分效果，从而选择正确的切分。• 有助于识别分词中的交叉歧义。最少切分分词方法 • 使句子中切出的词数目最少。• 等价于在有向图中搜索最短路径的问题。• 将每个字元视作节点，每个词形成一条边。• 边权重可都视为 1，也可根据词频决定（尽量切出高频词）• 结合权重/概率之后，实际上可视作基于统计的分词方法基于匹配分词方法的优劣 • 优点：效率高、直观性好 • 缺点：对词典的依赖性 • 维护高质量词典需要极大的开支 • 永远难以应对新生词汇 • 词汇频率/重要性往往对结果不产生影响基于统计的分词方法 • 没有词典，怎么办？从海量文档中去找答案。• 字与字相邻共现的频率或概率能够较好的反映成词的可信度。• 如果某两个词的组合在统计上出现的几率非常大，那么我们就认为分词正确。统计分词的形式化表达 • c = c1c2...cn，c 是待分词的句子（字串）。而 w = w1w2...wm 是切分的结果。• 设 $P(w|c)$ 为 c 切分为 w 的某种估计概率。• wa，wb，...，wk 为 c 的所有可能的分词方案 • 那么，基于统计的分词模型就是找到目标词串 w，使得 w 满足：• $P(w|c) = maxP(wa|c), P(wb|c), ..., P(wk|c)$ 统计分词的一般化过程 1. 建立统计语言模型 2. 对句子按不同方案进行分词 3. 计算不同分词方案的概率，选出概率最大的分词结果 • 理论上，基于统计的分词方法可以不需要词典，但实际应用中第 2 步可以采用机械分词方法进行分词，以获得候选的分词集合。• 既发挥匹配分词切分速度快、效率高的特点。• 又利用了无词典分词结合上下文识别生词、自动消除歧义的优点。N-gram 模型与马尔科夫假设 • N-gram 指一个由 N 个单词组成的集合，各单词具有先后顺序。• N-gram 模型的马尔可夫假设：• 当前状态出现的概率仅同过去有限的历史状态有关，而与其他状态无关。• 具体到分词任务，就是文本中第 N 个词出现的概率仅仅依赖于它前面的 N-1 个词，而与其他词无关。• 常见的 N-gram 模型：• N = 1，一元文法模型（最大概率模型），$P(w) = P(w1)P(w2)...P(wn)$ • N = 2，Bigram 模型，$P(w) = P(w1)P(w2|w1)...P(wn|wn-1)$ • N = 3，Trigram 模型，$P(w) = P(w1)P(w2|w1)P(w3|w1w2)...P(wn|wn-2wn-1)$ N-gram 模型的概率估计 • 以 Bigram 模型为例，基于最大似然估计进行推断 $P(wn|wn-1) = \frac{C(wn-1wn)}{C(wn-1)}$ • 其中，$C(wn-1wn)$ 指词序列 $wn-1wn$ 在语料库中出现的次数。• 而 $C(wn)$ 指某个单词 wn 在语料库中出现的次数。N-gram 模型的分词过程 • 以 Bigram 模型为例 1. 首先，构造训练语料库，计算所有的 $C(wn)$ 与 $C(wn-1wn)$。2. 其次，对于每一个可能的分词序列 w，计算以下公式 • $P(w) = P(w1)P(w2|w1)...P(wn|wn-1)$ • 其中，$P(wn|wn-1) = \frac{C(wn-1wn)}{C(wn-1)}$ 3. 最后，返回最大的 $P(w)$ 所对应的分词序列作为结果。特殊形式：一元文法模型 • 当 N=1 时，N-gram 模型退化为一元文法模型，此时词与词之间是独立的。(独立性假设，一元文法) $P(W) = P(w1, w2, ..., wi) = P(w1)P(w2)...P(wi)$ $P(wi)$ = wi 在语料库中出现的总次数 n / 语料库中的总词数 N 基于统计文法模型的优劣 • 优点：减轻了对于词典的依赖性 • 然而，这种依赖并非完全消除，取决于性能与效率的平衡 • 如果深度结合机械分词（匹配分词），则效率提升但依赖词典 • 如果减少对词典的依赖，则需要更多地遍历潜在的组合（解空间巨大！）• 缺点：依赖已有数据中词频的统计，对于新生词汇或专业词汇不友好 • 冷门领域的稀有词汇往往难以准确划分 • 易受数据集先验偏差（Bias）的影响隐马尔可夫模型（HMM） 输入模型 $\lambda = (A, B, \pi)$ 和观测 $O = (o_1, o_2, ..., o_T)$，输出最优路径 $I^* = (i_1^*, i_2^*, ..., i_T^*)$ 初始化 $\delta_1(i) = \pi_i b_i(o_1) \quad i = 1, 2, ..., N$，$\psi_1(i) = 0 \quad i = 1, 2, ..., N$ 递推，对 $t = 2, 3, ..., T$，$\delta_t(i) = \max_{1 \le j \le N}\{\delta_{t-1}(j) a_{ji}\} b_i(o_t), i = 1, 2, ..., N$，$\psi_t(i) = \arg\max_{1 \le j \le N}\{\delta_{t-1}(j) a_{ji}\}, i = 1, 2, ..., N$ 终止 $P^* = \max_{1 \le i \le N} \delta_T(i)$，$i_T^* = \arg\max_{1 \le i \le N}\{\delta_T(i)\}$ 最优路径回溯，对 $t = T-1, T-2, ..., 1$，$i_t^* = \psi_{t+1}(i_{t+1}^*)$ 停用词的概念与意义 • 停用词，Stopwords，指文档中频繁出现或对实际语义影响不大的词语 • 例如，英文中的 The、of，中文中的"的"、"是"等。• 数字、副词等与语义关系不大的词常作为停用词被处理。• 为什么要去除停用词？• 重复率很高，会造成索引中的倒排表很长，影响查询性能。• 对最后结果的排序没什么贡献，反而可能产生干扰停用词类型与识别 • 停用词的设定与语料库的性质有关 • 除通用停用词表外，特定领域或特定场景下会有其专用的停用词 • 例如，URL 中的 www，Wikipedia 中的 wiki • 常用的停用词识别方法 • 较为成熟的停用词识别方法有：文本频率、词频统计、熵计算等。• 更为复杂的算法将结合统计与句法或内容分析。• 常用的停用词表：哈工大停用词表、百度停用词表、四川大学机器智能实验室停用词表去除停用词可能导致的隐患 • 有些停用词在特定场景下是有意义的 • 例如，"非"、"不"表示否定；"较"、"稍微"表示程度等。• 有些停用词的组合是有意义的 • 例如，"的士"、"To be or not to be"。• 依赖于分词的效果。未来停用词的使用趋势 • 现代搜索引擎的趋势是逐渐减少对停用词的使用。• 现代搜索引擎更关注利用语言的统计特性来处理常见词问题。• 采用压缩技术，降低停用词表的存储开销。• 引入词项权重，将高频词的影响降至最低。• 索引去除技术，降低停用词对结果的影响归一化处理 • 归一化/词根化，指还原词语的特殊形式的过程。• 例如：Ran, running run • Universities university • 往往针对英语等语言，汉语并不需要这一步。• 词根化可以有效降低词项的数量并减少歧义。词干提取 • Stemming，指去除单词前后缀，获得词根的过程。• 常见的前后缀通常有"复数形式"、"过去分词"、"进行时"等词形还原 • Lemmatisation，指基于词典，将单词的复杂形态转变成最基础的形态。• 并不是简单地将前后缀去掉，而是会根据词典将单词进行转换。词干提取与词形还原的异同 • 词干提取与词形还原的相同点 • 目标一致。两者的目标均为将词的不同形态简化或归并为基础形式。• 结果交叉。两者不是互斥关系，其结果有部分交叉。• 方法类似。目前两者主流方法均是利用语言中存在的规则或利用词典映射提取词干或获得词形。• 词干提取与词形还原的不同点 • 在原理上，词干提取主要采用"缩减"，而词形还原主要采用"转变"。• 在复杂性上，词形还原方法相对更复杂。• 在实现上，词干提取主要利用规则变化进行词缀的去除和缩减，而词形还原依赖于词典进行词形变化和还原。• 在结果上，词干提取不一定得到完整单词，而词形还原是完整单词。第四节网页索引 倒排索引的概念与意义 • 信息检索中非常流行的，基于词项的索引模式 • 主要包括以下两部分结构：• 词汇表（Lexicon, Dictionary）：语料集合中所有不同单词构成的集合 • 倒排表（Posting List）：文档 ID 列表，列举词项在哪些文档中出现正排和倒排 • 正排索引 (key) Document Term • 倒排索引 (key) Term Document 倒排索引的实例：建立倒排表的流程 • 第一步：对文档中的词项进行预处理 • 第二步：生成词项-文档 ID 序列对 • 第三步：对词项排序 • 第四步：合并 • 对于使用 AND 连接的两个词项，合并两个倒排表 • 如果两个倒排表的长度分别为 x 和 y，则合并操作的时间复杂度为 O(x+y) 次操作倒排表的优化问题 • 处理查询的最佳顺序是什么？• 对于使用 AND 连接的查询，其本质是倒排表的合并 • 按照文档频率的顺序进行处理 • 先处理文档频率小的，再处理大的 • 例如：(Brutus OR Caesar) AND (Antony OR Cleopatra) • 同样，按照文档频率的顺序进行处理 • 首先，获得所有词项的文档频率 • 其次，保守地估计出每个 OR 操作后的结果大小 • 考虑 x+y 的最坏情况 • 最后，按照结果从小到大的顺序执行 AND • 通过设置跳表指针跳过部分文档，从而实现更快速的合并 • 如何利用跳表指针实现快速合并？• 在什么位置设置跳表指针？• 以何种策略设置跳表指针？• 设置较多的指针 + 较短的步长 → 更多的跳跃机会 • 相应的，需要耗费更多的存储空间 • 设置较少的指针 + 较长的步长 → 更少的指针比较次数 • 存储空间消耗更少，但跳跃机会也更少 • 一个简单的启发式策略：如果倒排表长度为 L，则每隔 $\sqrt{L}$ 均匀放置跳表指针 • 该策略没有考虑查询词项的分布，未必最优。• 同时，索引的动态变化也会影响跳表指针的设置 • 如果索引相对固定，建立有效的跳表指针就相对容易 • 反之，经常更新的索引很难建立合适的跳表指针倒排表的扩展问题 • 除了文档 ID，搜索引擎往往在倒排表中加入更多元素 • 例如：词项频率（Term Frequency）、词项类型等 • 同时，除了基本的词项查询之外，倒排表还可能面临更多需求 • 词组查询："中国科学技术大学" • 词项邻近："肥西"＋在附近出现的"老母鸡" • 文档区域：作者是"D.Manning"的文档短语查询的需求 • 用户希望能够对类似"stanford university"的查询中的两个词看成是一个整体 • 当前倒排表实现下，类似"I want to university at stanford"这样的文档是不能够满足用户需求的。• 大部分的搜索引擎支持双引号的短语查询，这种语法很容易理解并成功地被用户使用。然而，有很多查询在输入时没有加双引号，其实都是隐式的短语查询第一种解决方案：二元词索引 • 将文档中每个连续词对看成一个短语 • 例如，文本"Friends, Romans, Countrymen"将生成如下的二元连续词对：• Friend Roman • Roman Countrymen 以上的每一个二元词对都将作为词典中的词项。• 经过上述的处理，可以构建面向二元词的倒排表，并处理两个词（或多个词）构成的短语查询 • 更长的短语查询：可以分成多个短查询来处理 • 例如，文本"stanford university palo alto"，将分解成如下的二元词对布尔查询：• stanford university AND university palo AND palo alto • 或采用更长的多元词索引来处理。• 如果采用二元词索引拼接的方式，对于该布尔查询返回的文档，我们不能确定其中是否真正包含最原始的四词短语。第二种解决方案：位置信息索引 • 二元词（或多元词）索引是一个昂贵的选择 • 词汇表迅速膨胀 • 在记录词项的同时，记录它们在文档中出现的位置，可以达成更广泛的查询 • 在这种索引中，对每个词项，采用以下方式存储倒排记录表：• 对于短语查询，仍采用合并算法（AND），查找符合的文档。• 不只是简单地判断两个词是否出现在同一文档中，还需要检查他们出现的位置情况是否符合要求 • 更为重要的是，位置信息可以用于邻近查询（例如，间隔 k 个词）倒排索引的存储问题 • 一种常用方式：词典与倒排表一起存储 • 便于同时读取，但文档数量较大时，会影响性能 • 另一种常用方式：两者分开存储 • 词典与倒排表分别存储为不同的文件，通过页指针关联 • 倒排表文件也可采用分布存储的方式 • 优点：性能大幅提升 • 词典可以常驻内存，至少常驻一部分（例如索引）• 可以支持并行、分布式读取 • 常见的词汇表存储结构 • 顺序存储 • 哈希存储 • B/B+ 树 • Trie 树索引压缩的意义 • 为什么要压缩索引？• 节省磁盘空间，提高效率（内存利用率或数据传输速度）• 前提：快速的解压缩算法。目前的压缩算法效率都非常高。• 对于词典而言，压缩的好处是：压缩得足够小，可以直接放入内存中，提升效率 • 对于倒排记录表而言，压缩的意义在于：减少所需的磁盘空间，可以更多地写入内存 • 减少从磁盘读取倒排表所需时间两种索引压缩策略 • 从词典和倒排表两个维度入手，实现索引的压缩 • 词典压缩：定长存储 • 在词项数目为 400K 的情况下，考虑每个词项占 28 字节，一共需要 11.2 MB • 词项本身设置定长为 20 字节，另外需要记录文档频率和倒排指针的信息 • 定长存储造成空间浪费压缩词项列表：将词典视作单一字符串 • Dictionary-as-a-String，词项之间用指针分割 • 指向下一个词项的指针同时也标识着当前词项的结束 • 期望节省 60% 的词项空间：(20-8)/20×100% = 60% 字符串词典的空间大小 • 每个词项平均占用 19 个字节，而不是原先的 28 个字节 • 首先，在词项字符串中，每个词项平均占用 8 个字节 • 其次，词项文档频率与倒排表指针各需 4 个字节不变 • 最后，词项指针需要 3 个字节 • 字符总长度约为 400K*8=3.2M，用大约 22bit 3B 长度可以标记 • 8+4+4+3=19，400K 词项一共仅需 7.6MB（定长存储时为 11.2MB）进一步压缩：按块存储（Blocking） • 单一字符串在词项指针上需要占用较多额外空间 • 通过为每 k 个词项存储一个指针，减少指针的数量 • 需要额外 1 个字节用于表示词项长度。• 例如，下图中的例子为 k=4 • 按块存储通过牺牲少量存储空间获得指针数量减少，可以节省更多的指针开支 • 例如，当块的大小 k=4 时 • 不采用按块存储时，每个指针花费 3 个字节，共需 12 字节 • 采用按块存储时，只需要花费 3+4*1=7 字节 • 由此，原先的存储空间可以进一步减少到 7.1MB • 讨论：k 增大时，开支进一步降低，但为什么不选取更大的 k? 按块存储在搜索上的问题 • 未压缩的词典的搜索，标准二分法 • 假设词典中每个词项被查询的概率相同（实际上并非如此）• 则平均比较次数为：(1+2*2+4*3+4)/8=2.625 次 • 采用按块存储后，二分查找只能在块之间进行 • 块内采用线性查找方式。• 当块的大小 k=4，平均比较次数为：(1+2*2+2*3+2*3+2*4+5)/8 = 3 次 • 显然，随着 k 上升，线性查找部分增多，效率更低 • 另一种改进：前端编码（Front Coding） • 前端编码中，同一块的词项往往具有公共前缀 • 使用特殊符号表示前缀的公共部分，如下图的 倒排表存储的问题所在 • 倒排表所需的空间远远大于词典本身 • 最迫切的需求在于如何紧密地存储每一个倒排表，尤其是文档 ID • 如果使用 4 字节整数来表示文档 ID，每个文档 ID 需要 32bit 再进一步：可变长度编码 • 我们的需求：对于一个间距值 G，想用最少的所需字节来表示它 • 关键问题：需要利用整数个字节来对每个间距编码 • 这需要一个可变长度编码，对小数字使用短码来实现这一点 • 可变长度编码的基本流程大致如下：• 先存储 G，并分配 1bit 作为延续位 • 如果 G<128，则采用第一位延续位为 1 + 7 位有效二进制编码的格式 • 如果 G>=128，则先对低阶的 7 位编码，然后采取相同算法对高阶位进行编码。最后一个字节（8bit）的延续位为 1，其他字节延续位为 0 • 可变长度编码的实例 • 例如，5 的二进制为 101，加上延续位为 10000101. • 214577 的二进制为 1101/0001100/0110001，因此拆分为 3 个字节。• 相比于 4 字节整数，可变字节码在小数字上的短码可以节省更多空间第五节查询与评估 何为相关性反馈（Relevant feedback） • 用户在查询后标记相关/不相关，然后迭代更新查询，以获得更好的结果 • 相关性反馈的动机 • 你也许无法表达想要找的内容，但是你至少能够判断所看到的内容 • "为我提供更多相似的文档……" • 精准的查询条件或许无法一蹴而就，但可以通过迭代逐渐趋于精准相关性反馈的基本流程 1. 首先，用户提出一个查询条件（Query）2. 对于返回的文档，用户标出相关与不相关的部分 3. 系统根据用户反馈，获得用户信息需求更为准确的描述 a) 基于相关性反馈，更新查询条件 b) 基于新查询条件，获取新的结果文档并再次提交用户进行评估 上述过程将根据情况进行一次或多次的迭代，从而不断接近最优查询条件相关性反馈的最终目的 • 通过相关性反馈，获得用于表达用户查询意图的最优查询条件 • 常见方式：为已有词项添加不同权重，或增加新的词项 • 这一过程应对用户隐藏相关性反馈存在的问题 • 相关性反馈可能影响用户体验 • 用户不愿意提供显式的相关反馈 • 用户不希望因为相关性反馈（迭代）而显著延长搜索时间 • 相关反馈生成的新查询往往很长，降低系统效率，增加计算开支 • 一种做法是只改变重要词项权重而不增加新词项，但效果有限 • 有时很难理解，为什么经过相关性反馈后，会返回不相关的文档 • 被相关性反馈捕捉的词项，未必是用户需要的内容常见的相关性反馈类型 • 显式反馈（Explicit Feedback）• 用户显式地参与交互过程 • 隐式反馈（Implicit Feedback）← 更为常见！• 系统追踪用户行为来推测返回文档的相关性 • 伪反馈（Pseudo Feedback）• 在没有用户参与的前提下，直接假设返回结果是相关的，并进行反馈显式反馈 • 最基础的显式反馈：用户点击记录 • 显而易见的缺陷：只有正样本 • 用户不点击，不代表完全不相关！• 拓展的显式反馈：收集负面评价的渠道日益丰富更为复杂的显式反馈：用户评论 • 基于用户评论，可以收集更为完整的相关性反馈 • 同时，对于网页质量有更为可靠的判断隐式反馈 • 通过观察用户对当前检索结果采取的行为，来判断检索结果的相关性 • 判定不一定很准确，但省却了用户的显式参与行为 • 常见的用户行为种类 • 鼠标键盘动作，如点击、停留、翻页、拷贝等 • 用户眼球动作，如凝视、移动、拉近、拉远等鼠标键盘动作可能揭示用户身份特征 • 他山之石：《暗算》，"手迹"识别报务员 • 不同用户在击键频率、时延、习惯、错误率等方面存在一定差异 • 用户眼球动作，可以揭示用户关注的内容及关联（视觉注意特征）• 借助眼球动作捕捉，还可以支撑其他相关的应用 • 例如，判断文本之间的相关性，甚至揭示问题的答案隐式反馈的优缺点 • 优点：• 不需要用户显式参与，减轻用户负担，提升用户体验 • 用户行为某种程度上可以反应其兴趣，因此具有可行性 • 缺点：• 对行为分析有着较高的要求 • 准确度难以保证 • 某些情况下需要增加额外设备（且很贵！）伪相关性反馈 • 无需用户参与反馈过程，而直接根据检索结果自动反馈，较为简单 • 对于用户查询返回的有序结果，假定前 K 篇文档是相关的 • 在此基础之上，进行相关性反馈 • 实验证实，利用伪相关性反馈技术可以提升检索的效果 • 但是，伪相关性反馈存在显著的隐患：• 结果未经用户判断，难以保证其准确性 • 某些查询可能结果很差，甚至出现查询漂移（被 Top K 文档带了节奏）查询扩展：一种用户相关性反馈的特殊方式 • 在相关性反馈中，用户针对文档给出反馈，这些反馈将被用来更新查询条件。• 而在查询扩展中，用户针对词项的合适程度给出反馈，这些反馈将被用来构建更为完整的查询条件。• 暗含的功能：用户选择和确认的查询扩展能够更好表达其查询意图。内容回顾：拼写检查 • 拼写检查在本质上也是对查询的修改和完善 • 通常采用基于词典或编辑距离的方式进行检查和校对。• 编辑距离（Levenshtein Distance）：两个字符串间转换最少需要的编辑步数查询扩展的实现 • 利用同义词辞典，可以实现查询条件的扩展 • 对于某个查询词汇，使用辞典中的同义词或相关词进行扩展 • Feline（猫科）→ Feline cat • 相对于原始的查询词汇，可以给扩展的词汇设定更小的权重 • 一般而言，查询扩展有助于提升查询的召回率（找得更全）• 但是，可能会影响准确率，尤其在扩展词存在歧义的情况下 • 编纂和维护同义词词典需要很大的代价查询扩展的类型 • 利用人工编纂的同义词词典 • 例如，第三节"网页文字处理"中提到的 How-Net、大词林等 • 全局分析与同义词词典的自动生成 • 基于统计词汇之间的共现关系（Co-occurrence），自动构建词典 • 基于搜索日志进行优化 • 通过查询日志，挖掘查询的等价类自动构建相关词词典的两种思路 • 通过分析文档集中的词项分布，来自动生成同义词/近义词词典。• 基本的想法是计算词语之间的相似度，常见的两种思路如下：• 思路 1：如果两个词经常和相似的词共同出现，则它们很可能是相似的 • E.g.，Car 和 Motocycle 很可能相似，因为它们和 road、gas 等经常共现 • 思路 2：如果两个词经常与相同的词以一种特定的语义关系共同出现，那么它们很可能是相似的 • E.g.，可烹调并食用的实体往往都属于食物基于搜索日志的查询扩展 • 搜索日志目前是搜索引擎查询扩展的重要方式 • 实例 1：提交查询 Herbs（草药）后，用户往往搜索 Herbal remedies （草本疗法）。• 此时，Herbal Remedies 是 Herbs 的潜在扩展查询 • 实例 2：用户搜索 Flower Pix 与 Flower Clipart 时，往往都会点击 URL：photobucket.com/flower • 此时，Flower Pix 与 Flower Clipart 可能互为潜在扩展查询相关词扩展的潜在问题 • 词项关联的质量是一个问题 • 有歧义的查询词可能导致统计上相关，而意思上不相关的词 • 如"Apple Computer"可能导致"Apple Red Fruit Computer" • 同时，由于扩展的查询词与原查询词高度相关，扩展后的查询也未必能够获得更多的相关文档结果评价的常见内容 • 评价性能（Effectiveness）的常见指标 • 面向单个查询的评价指标 • 无序/二元结果：Precision、Recall、F-value⋯ • 有序/多元结果：P@N、R@N、AP、NDCG⋯ • 面向多个查询的评价指标 • MAP、MRR 及其各种拓展指标基本评价指标的矩阵化表示 • 两个视角下对于数据集的切分 • P/N：Positive or Negative，表示算法对样本的判断 • T/F：True or False，表示算法判断的正确与否（和结果是否一致）• 四种简写的含义：• TP：True Positive，样本为正例，且被判定为正，即真正 • FN：False Negative，样本为正例，但错误地被判定为负，即假负 • FP：False Positive，样本为负例，但错误地被判定为正，即假正 • TN：True Negative，样本为负例，且被判定为负，即真负面向单查询的基本评价指标 • 准确率（Precision）• 指检索出的文档中，相关文档所占的比例，也称查准率 • 计算公式为 TP/(TP+FP) • 召回率（Recall）• 指所有相关文档中，被检索出来的部分的比例，也称查全率 • 计算公式为 TP/(TP+FN) 为什么某种方案被抛弃？• 既然 TP 与 TN 都是正确结果，为什么不直接计算 (TP+TN) 的全局比例？• (TP+TN)/(TP+TN+FP+FN)，即 Accuracy，在模式分类中经常被使用 • 然而，它在信息检索的相关任务中并不常见准确率与召回率的平衡 • 不同应用场景中，对于准确率和召回率有着不同的侧重 • 邮件分类：宁愿放过一些垃圾邮件，也不能错杀正常邮件 • 牺牲（对垃圾邮件的）召回率，保证较高准确率 • 智慧医疗：宁愿多判断一些疑似患者，不能漏掉一个真实病人 • 牺牲（对确诊病人的）准确率，保证较高召回率召回率的近似计算 • 解决方法：缓冲池（Pooling）方法 • 针对某一检索问题，各个算法分别给出检索结果中的 Top N 个文档 • 将这些结果汇集起来并进行人工标注，从而得到一个相关的文档池 • 潜在假设：大多数相关文档都在这个文档池（Doc Pooling）中 • 这一方法的可行性在于，虽然它实际上仍然无法得到全部相关文档，因此并不能得到召回率的绝对值。但是，它可以比较各个算法的相对优劣 • 因此，这一算法在各个测评中被广泛采用，N 通常取 50-200 P-R 的平衡到 F 值 • 如前所述，准确率与召回率之间存在权衡 • 如何综合评价一个算法在这两项指标上的性能？• F 值（F-measure），即准确率与召回率的加权调和平均数 $F = \frac{1}{\alpha\frac{1}{P}+(1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P+R}$ • 通常情况下，我们取 =0.5 或 =1（即两者同等重要）• 此时，可得基本的 F1 值，即 F=2PR/(P+R) • 为何不使用算数平均，而使用调和平均综合这两个指标？如何综合评价一个算法在这两项指标上的性能？• 本质上说，我们希望的目标是 Precision 和 Recall 都比较高 • 调和平均在本质上对一方偏低的情况是有惩罚的，必须两方都较高才会高 • 算数平均和几何平均在处理极端情况下的效果并不够合理 • 一个不好的例子：如果采用算术平均计算 F 值，那么一个返回全部文档的搜索引擎的其 F 值就不低于 50%，显然这不是一个好结果 • 准确率、召回率与 F 值是信息检索任务中最为基础和常用的三个指标P-R 曲线与 ROC 曲线 • 在分类问题中，准确率与召回率的平衡是通过选定不同阈值实现的 • 例如，通过调控相关性阈值，可以控制检索所得的文档数量 • 较低的阈值可以使得返回更多文档，但也混入大量不相关的文档 • 较高的阈值可以保障文档的相关性，但也会遗漏许多相关的文档 • 如何选择合适的阈值？• 通过绘制不同阈值下的指标变化曲线，可以帮助我们做出选择P-R 曲线 • 以准确率和召回率分别作为两条轴线 • 通过选定不同的阈值得到不同的 P-R 点并连接成线 • 通过 P-R 曲线，可以直观地看出准确率与召回率之间的平衡关系ROC 曲线 • 以真正率【TP/(TP+FN)】和假正率【FP/(FP+TN)】作为两条轴线 • 真正率/命中率，假正率/误报率 • 通过选定不同的阈值得到不同的真正率- 假正率点并连接成线 • 对角线表示区分能力为 0，即随机猜测 • 在对角线上端越远，效果越好 • 低于对角线的结果无意义（无区分度）如何基于 P-R 曲线或 ROC 曲线判别算法好坏 • 如果线 A 将线 B 完全包住，显然线 A 对应的算法效果更好 • 如果两条线发生重合，则可依据以下规则判别：• 计算 AUC，AUC 更高者效果更好 • Area under curve，即曲线下面积 • 可通过积分近似计算 • 另外，当使用 P-R 曲线时，可使用平衡点计算 • 平衡点即 Precision = Recall 的点，值越高越好P-R 曲线与 ROC 曲线的选择 • ROC 曲线兼顾正负样例，更为全面，而 P-R 曲线则只考虑正例 • 用户往往更关心正样本，如果面向特定应用场景（如检索），P-R 曲线是个好选择 • 单份数据正负样本比例失调时，P-R 曲线更合适 • 当负样本比重过高时，负例的数目众多致使 FPR 的增长不明显，导致 ROC 曲线呈现一个过分乐观的效果估计，从而难以体现出性能的差异性 • P-R 曲线受分布影响大，多份数据且正负比例不一时 ROC 曲线更合适 • ROC 曲线两个指标各自针对正负样本，而 Precision 只针对正样本，受影响较大P、R@N 的概念与意义 • P@N，即 Precision@N，指前 N 个检索结果文档的准确率 • 由于大多数用户只关注第一页或前几页，因此 P@10、P@20 等对于大规模搜索引擎来说是很合适的评价指标。• 如果相关文档数小于 N，P@N 的理论上限必定小于 1 • 同理，可得 R@N，即 Recall@N，指前 N 个检索结果找回的相关文档比例 • 由于返回结果有限，Recall@N 值，甚至其理论上限往往都远小于 1 • 理论上限为 N/相关文档数，即使通过 Pooling 加以控制仍然较小一种特例：R-Precision • 检索结果中，在所有相关文档总数位置上的正确率 • 由于 N 往往小于相关文档总数，因此设计了这一特殊指标 • 例如，如果相关文档总数为 3，那么考察 P@3 作为 R-Precision • 实例：如果查询 2 的标准答案集合为 d1,d2,d13，那么：• 系统 1：R-Precision=1/3；系统 2：R-Precision=2/3；面向 P、R@N 的 P-R 曲线 • 在有序结果情况下，可以不再采用不同阈值作为 P-R 值的依据，而是通过依次计算前 N 个结果对应的 P-R 值绘制曲线 • 新的不相关文档被检索时，Recall 不变，Precision 下降平均准确率 AP • 用于对不同召回率点上的正确率进行平均 • 通常情况下，AP 有三种不同的定义与计算方法 • 未插值 AP：查询 Q 共有 6 个相关结果，排序返回了 5 篇相关文档，位置分别是第 1，第 2，第 5，第 10，第 20 位，则 AP=(1/1+2/2+3/5+4/10+5/20+0)/6（注意补 0）• 插值 AP：事先选定插值点数并进行插值。例如，当我们计算 11 点平均时，计算在召回率分别为 0（第一条）, 10%, 20%,⋯, 100% 的十一个点上的正确率求平均（注意补 0 前面补 1）• 简化 AP：只对返回的相关文档进行计算，AP=(1/1+2/2+3/5+4/10+5/20)/5，倾向那些快速返回结果的系统，没有考虑召回率和补零的情况（不补 0）累计增益 CG • 用于衡量位于位置 1 到 p 的检索结果的相关度之和。$CG_p = \sum_{i=1}^{p} rel_i$ • Rel 用于描述文档相关性，可以根据需求选取多个数值 / 级别。• 较高的 CG 表明文档的整体相关性较高。• 然而，由于 CG 并未考虑文档位置，并不能体现靠前部分文档的质量。改进的相关度加和 • 如何区别位置的作用？直观想法：对不同位置赋予不同折损 • 折损累计增益（Discounted Cumulative Gain, DCG）• 基本思想：若搜索算法把相关度高的文档排在后面，则应该给予惩罚。• 一般用 log 函数来表示这种惩罚，如 log(i+1)，i 为文档位置 • 往往有以下两种计算公式（后者采用指数，更突出相关性）：$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{log_2(i)}$ $DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i}-1}{log_2(i+1)}$ 归一化的相关度加和 • 然而，DCG 由于其随着长度单调非减的特性，仍具有其局限性 • DCG 与具体查询和结果列表的长度 p 有关，不利于不同算法之间的对比 • 不同查询的结果有多有少，因此其 DCG 值无法实现相互比较 • 对 DCG 进行规范化：归一化折损累计增益（Normalized DCG, NDCG）• 基本思路：将 DCG 除以完美结果下得到的理想结果，iDCG（ideal DCG）• 即：NDCG = DCG / iDCG • 其中，iDCG 是根据文档根据相关性从大到小排序得到理想化的最优序列，并对此序列计算 DCG 值所得到的。从 AP 到 MAP • 从单查询拓展至多查询评价，可以更全面地体现排序算法的综合性能 • 如何对多查询的结果进行综合？• MAP（Mean AP），对所有查询的 AP 求算数平均 • 例如：假设有一个检索系统 • 对查询 1 返回 4 个相关网页，其 rank 分别为 1，2，4，7 • 对查询 2 返回 3 个相关网页，其 rank 分别为 1，3，5 • 查询 1 共有 4 个相关文档，查询 2 共有 5 个相关文档查询 1AP=0.83，查询 2AP=0.45，MAP=(0.83+0.45)/2 • MAP 可以反映全部查询的综合效果，但在查询难度不平衡的条件下有误导注重首个相关文档的 MRR • 在许多查询任务中，用户只关心第一个相关的文档，越靠前越好 • 该位置的倒数被称作 Reciprocal Rank（RR），数字越大效果越好 • 对多个查询所得的倒数排序求平均，即 Mean RR，MRR • 例如，两个查询，第一个查询的第一个相关文档在位置 2，第二个查询的第一个相关查询在位置 4 • 则 MRR 为 (1/2+1/4)/2 = 3/8 • 即平均在 8/3 的位置上找到第一个相关文档为什么要考虑多样性 • 一方面，用户的单次搜索可能体现出多方面的需求 • 另一方面，用户搜索可能存在歧义，需要展示多方面内容加以确认 • 此外，也要避免信息茧房的产生多样性的形式化定义 • 基本形式：给定一个查询 q，返回一个多样化的结果文档集合 R(q)。• 其中，R(q) 作为一个整体，应满足以下条件：• R(q) 中所有的结果文档都与查询 q 本身有较大的相关性。• 总体上要有较小的冗余度，以覆盖 q 的不同方面。• 核心思想：降低用户无法获得所需信息的风险 • 尽可能确保排序靠前的结果中至少有一个结果满足用户的需求。多样性的两种衡量方式 • 总体需求：衡量不同文档之间的主题差异性。• 一般而言，衡量方式有以下两种：• 隐式模型：只计算文档之间的差异性 • 文档是什么内容，不会也无法进行详细考量 • 显式模型：更加具体地考量文档所对应的用户意图 • 会从文档中抽取主题，并显式地实现主题的多样化第六第七节排序 量化方法（1）：Jaccard 系数 • 布尔模型的一个假设：所有的查询条件都必须满足 • 假如打破这一约束，能够实现相关性的 [0,1] 量化？• 1901 年，Paul Jaccard 提出计算两个集合重合度的方法 • 如果 A 与 B 为两个集合，那么有：$JACCARD(A, B) = \frac{A \cap B}{A \cup B}$ 量化方法（2）：词项频率 TF • 比 Jaccard 更进一步，如何通过词项区分文档权重？• 一个直观的想法：查询词在文档中出现得越多，该文档越相关 • 词项频率 TF(t,d)，指词项 t 在文档 d 中出现的次数（Term Frequency）• 利用原始的 TF 值，可以粗略计算文档的相关性 • 但存在一定问题，相关性与频率并不线性相关 • 某个词在文档 A 中出现 100 次，在文档 B 中出现 10 次，A 比 B 相关 10 倍？• 原始 TF 基础之上的改进：引入对数词频 $wf_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d}, tf_{t,d} > 0 \\ 0, otherwise \end{cases}$ • 通过这种方式，数量级的差异性所造成的影响变得更为缓和 • 由此，可以将文档与词项的匹配得分定义为所有同时出现在查询与文档中的词项其对数词频之和，即 $\sum_{t \in q \cap d}(1 + \log_{10} tf_{t,d})$ • 如果没有公共词项，显然得分为 0 量化方法（3）：文档频率 DF • 如何再近一步？• 有些词在单个文档中出现的多，是因为这个词本身就很常用 • 总结起来：罕见词的信息量更为丰富，而频繁词的信息量相对较少 • 相应的，如果查询中包含某个罕见词，则包含这个词的文档可能很相关 • 基于上述思想，引入新的度量机制：文档频率（Document Frequency）• dft，指出现词项 t 的文档数量 • dft 是与词项 t 的成反比的一个值 • 相应的，我们一般采用逆文档频率（Inverse DF）来衡量 $idf_t = log_{10}\frac{N}{df_t}$ • 需要注意，这里同样引入对数计算方式来抑制 DF 中数量级的影响量化方法（4）：集大成的 tf-idf • TF 与 DF，从两个角度为衡量文档相关提供了量化依据 • 如何将两者整合起来，实现更完整的描述？• 乘起来就完事了 $W_{t,d} = (1 + \log tf_{t,d}) \cdot \log\frac{N}{df_t}$ • 同时，解答了一个问题：什么样的词适合衡量文档相关性？• 在少数文档内多次出现的词 • 指标随词项频率增大而增大，随词项罕见度增大而增大量化方法（5）：向量空间模型 • 每个文档和查询视作一个词项权重构成的向量 • 查询时通过比较向量之间相似性来进行匹配 • 如果用开头提到的形式化表述加以概括？• D：文档表达，每个文档可视作一个向量，其中每一维对应词项的 tf-idf 值 • Q：查询表达，可视作一个向量，其中每一维对应词项的 tf-idf 值 • F：非完全匹配方式 • R：使用两个向量之间的相似度来度量文档与查询之间的相关性 • 向量空间模型的总结 • 首先，将文档与查询表示成词项的 tf-idf 权重向量（也可采用其他方法）• 其次，计算两个向量之间的某种相似度（如余弦相似度）• 最后，按相似度大小进行排序，将 Top-K 的文档返回给用户 • 优点：简洁直观，可以支持多种不同度量或权重方式，实用效果不错 • 缺点：缺乏语义层面的理解和匹配，同时依赖 tf-idf 值也可能造成干扰 • 用户无法描述词项之间的关系，词项之间的独立性假设实际上不成立 • 例如：贝利 + 足球，郎平 + 排球查询与文档的匹配 • 如何计算查询与文档之间的相似度？• 最基本的方法：欧氏距离 • 然而，欧氏距离并不是一个好的选择，它对于向量长度（文档长度）非常敏感 • 更为合适的方案：余弦相似度 • 按照文档向量与查询向量的夹角大小来计算基于迭代的查询意图更新 • 上节课我们提到，用户的查询意图可能无法一蹴而就，而需要通过相关性反馈实现逐步更新 • 在本质上，这一过程是使查询意图的表达逐步逼近用户目标文档的过程 • 如何接近目标文档？可以从目标文档的质心为出发点 • 某种意义上说，由向量表示的文档，可以视作高维空间中的一个点 • 由此，"质心"就是一系列点（文档）的重心 • 我们可以用如下公式来计算一类文档的质心 $\vec{\mu}(C) = \frac{1}{|C|}\sum_{d \in C}\vec{d}$ • 其中，C 是文档的集合 • Rocchio 算法提供了一种将相关反馈信息融入到向量空间模型的办法 • 其思想在于试图找到一个完美的最优查询向量，以满足以下目标 $\vec{q}_{opt} = \arg\max_{\vec{q}}[\cos(\vec{q}, \vec{\mu}(C_r)) - \cos(\vec{q}, \vec{\mu}(C_{nr}))]$ • 即：使得查询尽可能离与之相关的文档更近，离与之不相关的文档更远 • 其中，理想情况是，在可知完整的相关/ 不相关文档集合的情况下，可以使用以下公式来获得一个完美的查询：$\vec{q}_{opt} = \frac{1}{|C_r|}\sum_{\vec{d}_j \in C_r}\vec{d}_j - \frac{1}{|C_{nr}|}\sum_{\vec{d}_j \notin C_r}\vec{d}_j$ • 问题在于，我们事实上并不可能获得完整的相关/不相关文档集合 • Rocchio 算法（1971）实际使用的近似方法如下：$\vec{q}_m = \alpha\vec{q}_0 + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j$ • 其中，Dr 为已知相关文档的向量集合，Dnr 为已知不相关文档的向量集合 • q0 为初始查询向量。、、 为权重，根据手工调节或经验设定 • 由此，新的查询向量将逐渐向相关文档向量移动，远离不相关文档向量正反馈 vs 负反馈 • 正反馈的价值往往大于负反馈 • 用户更关心符合需求的标准答案，而不是错误答案 • 相应的，可以通过设置 $\beta > \gamma$ 来给予正反馈更大的权重 • 很多系统甚至只允许正反馈，即 $\gamma = 0$ • 收集真正的负反馈往往比较困难TF-IDF 技术的局限性 • 使用 tf-idf 来表示词项简单快速而且容易理解 • 但是，tf-idf 仅以"词频"度量词的重要性，无法体现词项的位置信息以及词项与上下文的关系：• E.g."武松打虎""虎打武松" • 因此，我们需要一种能够表示出文本上下文关系的词项表示方法如何描述词项之间的语义关联 • 如何通过设计模型以及优化目标，来获得词项间更为一般化的关系表征 • Word2vec 模型的出现：两种设计思路 • 根据上下文预测中心词（CBOW）• 根据中心词预测上下文（Skip-gram）Word2Vec 模型的基本流程 • 我们以 CBOW 模型为例，介绍其工作原理（Skip-gram 模型基本思路类似）• CBOW 本质上是获得上下文的词项 → 中心词的映射 • 如何获得这样的映射？• 基于深度学习的 CBOW 算法详细步骤：• 使用 one-hot（0/1）向量表示词项：• 计算隐藏层（深度学习基操）：• 计算预测的中心词的概率：• 计算交叉熵损失，反向传播优化模型：• 基于多个不同滑动窗口进行多次训练 • 最终得到各个词项的表征：Wxi Word2Vec 的优缺点 • 优点 • 有效表征了词项之间的上下文关系 • 无监督，通用性强，可适用于各种 NLP 任务 • 缺点 • 无法解决一次多义的问题，例如 play music 和 play football • Word2vec 是一种静态的方式，其词项表征一旦训练确定就不会再做更改。因此，虽然通用性强，但是无法针对特定任务进行动态优化三类常见的排序学习算法 • Pointwise：将排序退化为分类或回归问题 • 输出：网页对应的分类（有序）、回归值或有序回归值 • Pairwise：比较一对网页之间的相关度，e.g.，相关 > 不相关 • 输出：网页对之间的偏序关系 • Listwise：对整个网页集合进行排序 • 输出：整个集合的完整排序，往往依赖特定排序指标Pointwise 类排序算法 • 基本假设：训练样本中的任何一个查询-文档对，都可以映射到一个分值或一个有序的类别（如优良中差）• 相应的，给定一个查询-文档对，Pointwise LTR 将试图预测其分值/类别。• 常见的模型类别包括：• 回归，将查询-文档对映射到具体分数 • 分类，将排序问题转化为一个面向有序类别的二分类/多分类问题 • 有序回归，在映射到具体分数的同时保持样本之间的有序关系 • Pointwise 类排序算法可以简单且广泛地套用已有回归、分类算法 • 然而，其局限性也较为明显 • 首先，Pointwise 类方法往往更为注重文档的相关度得分，而并不注重文档之间的相关性排序 • Pairwise 类方法的出现，为解决这一问题提供了新的手段 • 其次，不同查询所对应的文档，尤其相关文档数量不同，对损失函数的贡献也各不相同，一定程度上影响效果Pairwise 类排序算法 • 基本假设：同样是将排序问题转化为分类问题（二分类或三分类）• 每次比较一个查询与两个文档，衡量两个文档的偏序（Partial Order）• 分类器的目的在于判断哪个文档应该排在前面 • 对应的标签为 1，-1 或 1，0，-1（0 表示两个文档可以并列）• 相比于 Pointwise 类算法，Pairwise 类排序算法通过衡量样本之间的偏序关系，实现了从绝对相关性（分值）到相对偏序的改进 • 然而，Pairwise 类算法也具有自己的缺陷：• 首先，两两成对导致样本数大为提升，计算资源开支增加 • 其次，Pairwise 类算法仍然受样本不平衡问题的影响 • 最后，Pairwise 类算法无法体现全局排序的合理性 • 由此引出了最后一类算法：Listwise 类算法 • 局限性：样本不平衡性的影响 • 如果不同 Query 的 Doc 数量相差很大，从偏序的对来说，Doc（对）数量较多的 Query 将掩盖其他的 Query，从而产生干扰 • 计算指标时可通过加权等方法加以缓解 • 局限性：算法无法体现全局排序 • 某个成对文档排序错误，发生在不同位置，在 Pairwise 类条件下是等价的 • 例如：7 篇文档，相关性从 1 到 3，理想状态下的排序为 3 2 2 1 1 1 1 • 假如有如下两个排序：• Rank A：2 3 2 1 1 1 1 Rank B：3 2 1 2 1 1 1 • 这两个排序，在 Pairwise 类算法中是等价的，然而，后者显然更合理 • 如前所述，用户往往更关心排在靠前位置的文档Listwise 类排序算法 • 基本思路：直接面向整体排序结果进行优化 • 直接将排序的完整队列作为学习的对象 • 通常情况下，解决这一问题采用以下两种思路 • 直接采用某种 IR 指标对排序进行优化 • 直接设计面向完整排序的损失函数 • 通常情况下，由于全局优化的作用，Listwise 类排序算法可取得相比于 Pointwise 和 Pairwise 类算法更好的效果。• 然而，Listwise 类算法也会面临一些小的挑战，例如两个网页并列的情况 • 相当于 Pairwise 中文档对标签为 0 的情况PageRank 的设计思路 • 将网页视作一个点，网页间的超链接视作一条边，形成一个巨大的有向图 • PageRank 背后的总体思路：优质网页引用或推荐的网页，一定还是优质网页 • 三重衡量标准：• 链入链接数：单纯意义上的受欢迎程度 • 链入链接本身是否推荐度高：欢迎本身是否具有权威性 • 链入链接源页面的链接数：被选中的几率 • 体现源网页是否滥发推荐PageRank 的计算方法 • PageRank 的核心公式如下：$PR(p_i) = \frac{1-d}{N} + d\sum_{p_j \in M(p_i)}\frac{PR(p_j)}{L(p_j)}$ PageRank 转移概率矩阵为：$A = dR + \left[\frac{(1-d)}{N}\right]ee^T$，$ee^T$ 是全 1 矩阵 • PR(pi) 为网页 pi 的 PageRank 值 • PR(pj) 为指向网页 pi 的某个网页 pj 的 PageRank 值 • L(pj) 为网页 pj 发出的链接数量 • d 为阻尼系数，取值在 0-1 之间 • N 为网页总数，M(pi) 为链入 pi 的页面集合 • R 为归一化邻接矩阵转置 • M 为邻接矩阵转置PageRank 的实现过程 • PageRank 的迭代计算过程：• 采用近似迭代算法计算 PageRank 值 • 首先给每个网页赋予一个初值，例如 1/N • 然后，利用之前的公式进行迭代有限次计算，得到近似结果PageRank 中的两类特殊情况 • PageRank 的计算过程，实际上是一个马尔科夫过程 • 如果计算中出现陷阱节点或终止节点，如何处理？• 陷阱节点：只有一条指向自己的边，没有其他出边 • 终止节点：没有任何出边，如同黑洞 • 此外，孤立节点的存在也会产生一定影响 • 没有任何入边，单纯使用马尔科夫链无法跳转 • 仅有初始概率，不再更新，也不影响其他节点几类特殊情况的解决：Restart 机制 • 回顾 PageRank 的公式 • 其中的 (1-d)/N 的部分，相当于以一定概率重新选择起点 • 此时，所有节点以一定等概率被选中作为新起点 • 由此，跳出了陷阱和黑洞的干扰（所有节点全联通）• d 一般选择为 0.85 左右（Google 的选择）PageRank 中的收敛性问题 • 如前所述，PageRank 的计算过程，实际上是一个马尔科夫过程。• 取 A = dM + [(1-d)/N]eeT，其中 M 为跳转矩阵，e 为所有元素都为 1 的列向量 • 则有 Pn+1 = APn，形成马尔科夫过程 • 前述论文已从理论上证明，不论初始 PageRank 值如何选取，这种算法都保证了网页排名的估计值能收敛到他们的真实值。• 马尔科夫过程的三个收敛条件：• 转移矩阵 A 为马尔科夫矩阵 • A 矩阵所有元素都大于等于 0，并且每一列的元素和都为 1 • 转移矩阵 A 为不可约的 • 当图是强连通时，A 为不可约，而 Restart 保障了这一条件 • 转移矩阵 A 为非周期的 • 这三个条件，PageRank 算法都满足，因此保障了其收敛性。HITS 算法的两个基本假设 • 基本假设与核心概念相互对应 • 基本假设 1：好的 Authority 会被很多好的 Hub 指向 • 基本假设 2：好的 Hub 会指向很多个好的 Authority • 因此，在 HITS 算法中，每个网页需要计算两个权威值HITS 算法的计算过程 • 基于先前的基本假设，HITS 的计算过程如下：HITS 归一化迭代方法为 $a_{k+1} = \frac{M^T h_k}{|M^T h_k|}$，$h_{k+1} = \frac{M a_{k+1}}{|M a_{k+1}|}$ 取初始 Hub, Authority 为全 1 向量 • 首先，根据关键字获取与查询最相关的少数页面，及与这些页面有链接关系的页面，作为待选集合 • 其次，对所有网页的 a(p) 与 h(p) 进行初始化，可都设为 1 • 最后，迭代计算两个步骤，即基本假设所对应的两个公式 • 重复这一步，直到最终收敛为止 • 注意每一步中的归一化过程！• 输出 Authority 或 Hub 值较高的页面第八九节个性化检索 基于内容的推荐：核心思想 • 基本想法：用户的偏好一般相对稳定 • 因此，给用户推荐他/她以前喜欢的项目准没错（偏保守的想法）• 例如：电影推荐 • 推荐同演员、同导演、同主题……• 例如：新闻推荐 • 推

荐类似主题/倾向性的文章 • 可能造成局限性和错觉基于内容的推荐：项目画像 • 为实现基于内容的推荐，对于每个备选项目，需要给出相应的画像（Profile）• 更一般的形式：画像往往以向量的格式存在 • 例如：电影推荐 • 可以将元数据，如标题、类型、演员、导演等抽象为向量 • 例如：新闻推荐 • 可以从文章中抽取关键词，或采用类似 tf-idf、主题模型等方法基于内容的推荐：用户画像与评分预测 • 相应的，用户的画像可由他/她曾经评分过的项目画像所估计 • 一般采用加权平均的方式得到用户画像向量（基于评分进行加权）• 基于用户与项目画像，可采用相似性度量进行评分 • 一般采用两个向量之间的余弦相似度（Cosine Similarity）• , = cos , = · • 也可根据实际情况选择其他度量方式基于内容的推荐：优点 • 每个人的推荐过程相互独立，不需要其他用户的数据 • 可以为具有独特偏好的用户进行有效推荐 • 不受大众倾向性和热度的影响 • 可以推荐新项目或非热门项目 • 推荐结果有着较好的可解释性 • 可列举内容特征作为推荐的依据基于内容的推荐：缺点 • 找到合适的特征是一件困难的事 • 对于非结构化信息，如图像、视频、音频等尤其如此 • 部分特征的提取可能存在误导性（归因错误）• 给新用户推荐项目，永远是一个困难的任务 • 如何建立新用户的用户画像？• 过度特化（Overspecialization）现象 • 永远只能给用户推荐局限于其画像中的内容——信息茧房问题 • 用户的多方面兴趣难以体现 • 难以通过他人的评价对推荐结果进行评估协同过滤的基本思想 • 基于内容的推荐方案只基于单一用户记录向该用户进行推荐 • 在实际应用中，我们发现，其他用户的浏览行为对当前用户有借鉴作用 • 例如，你和你的狐朋狗友们往往具有相似的口味和行为 • 相应的，这种相似的口味导致了在评分上的相似性 • 如前所述，推荐系统的本质就是矩阵补全问题 • 相应的，协同过滤的思想在于基于矩阵的其他行，协助填补本行的空缺协同过滤的基本类别 • 大体上，可将协同过滤技术分为以下两大类 • 基于内存（Memory-based）的协同过滤 • 基于现有数据与简单度量运算进行推荐 • 可进一步细分为基于用户（User-based）与基于项目（Item-based）• 基于模型（Model-based）的协同过滤 • 基于现有数据训练模型，通过模型进行推荐 • 代表性方法如矩阵分解（Matrix Factorization）、深度学习等基于用户推荐 • 如前所述，具有相似偏好的用户，往往在过去与未来的评分行为上相似 • 基于用户（User-based）推荐的目的，即在于找到这些相似用户，并基于这些用户的历史行为进行推荐 • 相似用户往往被称作"邻居"，类似于寻找最近邻的思想 • 基于共同评分的项目，衡量用户之间的相似性如下：$\mathrm{sim}(a,b) = \frac{\sum_{p\in\mathrm{product}(P)}\left(r_{a,p}-\bar{r}_a\right)\left(r_{b,p}-\bar{r}_b\right)}{\sqrt{\sum_{p\in\mathrm{product}(P)}\left(r_{a,p}-\bar{r}_a\right)^2}\sqrt{\sum_{p\in\mathrm{product}(P)}\left(r_{b,p}-\bar{r}_b\right)^2}}$ 基于用户推荐的评分预测 • 在得到用户之间的相似性后，针对待预测的项目，可以根据历史上其他用户对于该项目的评分，结合用户之间的相似性作为加权，预测评分结果 • 相似性计算与评分预测中，都通过减去平均值来抹去个人评分偏好的影响 • 不同用户打分范围不同，有些人倾向于打高分，有些人更苛刻$\mathrm{pred}(a,p) = \bar{r}_a + \frac{\sum_{b\in\ \mathrm{neighbors}\ (n)}\mathrm{sim}(a,b)\cdot\left(r_{b,p}-\bar{r}_b\right)}{\sum_{b\in\ \mathrm{neighbors}\ (n)}\mathrm{sim}(a,b)}$ • 有些时候，我们不需要考虑所有邻居的评分，而只考虑 K-最近邻的情况 • 相应的，前页公式中的 neighbor(n) 即缩小为 K-最近邻的集合 • 几点需要注意的情况：• 只考虑对指定 Item 有评分的邻居，无评分的跳过（即使 Similarity 较高）• 通常情况下，低于 0 的相关性可以被忽略 • 因此，K-最近邻集合可能实际上小于 K 个邻居 • 基于项目（Item-based）的推荐采用类似的设定基于项目推荐 • 与用户行为相似性类似，相似的项目，往往在大众眼中的评分也比较接近 • 与基于内容的推荐不同，基于项目（Item-based）的推荐，其衡量相似项目的标准，并不是项目本身的属性，而是不同项目的评分历史 • 同一个人给两个项目打出相似分数，说明他认为两个项目相似；越多这样的人，两个项目越相似 • 基于项目推荐的计算公式$r_{ix} = \frac{\sum_{j\in N(i,x)} s_{ij}\cdot r_{jx}}{\sum s_{ij}}$ • 其中，相似度计算与 User-based 类似 • 即构造两个向量，然后计算 Pearson 相关系数 • 此外，同样可以基于平均分对分数估计进行修正 • 如果未评分，则直接设为 0，不用减去平均数 • 思考：最终的 rix 为何不需要平均分修正？因为只针对该用户自己的评分算加权和两类方法的对比 • 在实践中，往往基于项目的推荐效果更好，为什么？• 项目的属性相对单一，而用户的偏好则更为丰富多样 • 相应的，某样项目受欢迎的理由相对固定，而用户可能在不同情境下体现出不同的偏好 在你没有办法区分这些不同情境时，得到的用户偏好反而是不准的基于内存的推荐的优缺点 • 优点：可适用于任意种类项目，效果较好 • 不受多模态、非结构化信息表征与特征选取的困扰 • 缺点：冷启动、稀疏性、热度偏差等 • 冷启动：用户与项目都存在冷启动问题 • 稀疏性：用户评分记录严重稀疏，很难找到评价过同一项目的用户 • 热度偏差：更倾向于推荐热门项目，对具有独特偏好的用户推荐效果差 • 寻找相似用户/项目时，小众偏好很容易被热门偏好所淹没基于模型的推荐：基本思路 • 基于内存的推荐技术，仅对数据进行简单处理，适用于各种数据 • 然而，数据的稀疏性、计算最近邻的高复杂度，限制了其有效性 • 与此同时，我们知道，从效用矩阵的视角来看，推荐系统的本质是矩阵补全 • 那么，矩阵的各个元素是如何生成的？• 基本思路：用户对项目的评分，本质上是用户的偏好，与项目的属性之间的相似度。相似度越高，评分越高基于模型的推荐：基本形式 • 借鉴矩阵分解的思路，揭示潜在因子 • 评分矩阵 R 被近似视作项目属性矩阵 Q 与用户偏好矩阵 P 的乘积 • P 与 Q 的维度，一方面与用户/项目的数量有关，另一方面体现了潜在因子的数量 • 当用户与项目的潜在因子已知，则任何缺失的评分，均可以通过对应的 P、Q 矩阵相应的行列运算估计得到 • 那么，如何估计出两个潜在因子矩阵？• 既然用户评分是根据潜在因子的乘积所得，那么，基于这种方式估计得到的用户评分，应与历史评分记录尽可能接近，即 $\min_{P,Q}\sum_{(i,x)\in R}\left(r_{xi}-q_i\cdot p_x\right)^2$ 基于模型的推荐：过拟合与正则化 这一公式，其本质是一个误差平方和（SSE）。• 因此，我们的目标正是通过优化这一 SSE，以获得潜在因子的估计值 • 然而，潜在因子的维度 K，也是一个潜在的麻烦 • 我们需要训练的参数一共（M+N)*K 个，随着 K 的增长而增长 • 如果 K 足够的大，在数据稀疏的情况下，可能并没有那么多训练样本 • 因此，过高的 K 可能导致过拟合的问题 • 如何解决过高的 K 带来的过拟合问题？• 一种思路是引入更多训练数据，但这并非易事 • 另一种思路则尝试通过"收缩"参数的方式来提升泛化性 • 过拟合问题的本质是对于训练样本的过分迁就，从而影响了模型的泛化能力 • 相应的，我们的目的在于通过控制参数数值，使其不那么"迁就"训练样本 • 解决方法：引入正则项 • 避免过大的参数值矩阵分解的衍生模型（2）概率矩阵分解 • 另一种思路则着眼于参数本身的生成规律 • 在数据稀疏且有噪声的情况下，是否可能引入某些规律，可以实现对参数更好的描述。比如，参数符合高斯分布？• 假设 1：参数本身与噪声均符合高斯分布 • 假设 2：两个潜在因子矩阵相互独立，各用户/项目独立同分布 • 概率矩阵分解 • 在概率矩阵分解中，两个参数矩阵均服从均值为 0，预设方差的高斯分布 • 开放问题：如何求解 PMF 中的矩阵参数？基于 R 与 U、V 联合分布优化求解 • 在参数完成训练后，即可简单地得到相应的完整评分矩阵矩阵分解的拓展 • 前面我们介绍了基于矩阵分解推荐技术，然而，这只是最基本的形式 • 如何在模型中加入更多的信息，以获得更符合我们需要的结果？• 行内的黑话：给概率图加圈，给矩阵分解加约束，给神经网络加层 • 通过在矩阵分解中加入约束的方式来加入更多信息与假设，往往可以提升效果 • 例如，右边的公式实际上融入了基于内存的协同过滤思想一种最常见的约束方式：社交约束 • 合理而常见的假设：好友们在偏好与行为上十分相似 • 因此，如果我们知道好友关系作为 Side Information，可以对模型进行补充 • 一个基于社交约束方面的经典工作：SoRec • 右半边表示社交约束，C 表示社交关系 • 存在社交关系的，属性必然相似 • 基于偏好相似的社交约束虽然效果很好，但未必恰然成立 • 例如：社交关系可能并非通过偏好间接影响决策，而是直接影响决策 • 社交影响力的体现："碍于面子"的决策行为 • 右图的框架同时体现了这两种思路 • R1 为好友直接影响评分 • R2 为好友通过影响属性间接影响评分情境的概念与意义 • 在上述两个例子中，我们都是用了其他信息帮助我们理解用户意图。这一类信息被我们称作"情境信息"（Context Information）• 从计算机学科的视角出发，"情境"一词可定义为"所有与人机交互相关，用于区分标定当前特殊场景的信息"。• 基于这一定义，服务提供者可借助情境信息，为用户提供更精确的信息检索和过滤服务。搜索中的基础情境：上下文 • 直观上，查询上下文可以帮助更好的理解用户查询词。• 用户的搜索行为往往具有一定的连贯性 • 相应的，同一查询会话中的查询词和点击的 URL 往往是相关的基础的上下文感知方法 • 考虑特定长度以内的所有序列模式 • 保留频率高于阈值的模式，并存储为后缀树（Suffix Tree）• 当面临上下文感知任务时，根据已有序列找到相应节点，从而获得候选查询建议进阶的上下文感知方法 • 引入隐马尔科夫模型（HMM），将用户意图 s 视作隐变量，查询上下文 q 与点击记录 u 视作观察值。• 当前查询可能不止与前一个查询相关，如何处理？• 引入变步长的隐马尔可夫模型，打破 HMM 中每一个状态只与前一时间状态相关的约束 • 前述方法虽然有效利用了上下文信息，但将查询词限制在一个查询意图中，同时仅能推荐查询扩展，而不能帮助判断文档相关性细化的上下文感知不同类型 • 类型 1：查询重组 • 用户的后续查询仅仅只是先前查询的重新表述，目的不变或类似 • 在此情况下，先前点击的内容往往不再被点击（即使内容相关）类型 2：查询特化 • 在用户的后续查询中，对先前查询中部分内容进行了更为具体、深入的查询 • 在此情况下，先前查询中较为泛指的内容将被略过类型 3：查询泛化 • 在用户的后续查询中，对先前查询中部分内容进行了更泛化的查询 • 体现了用户对于该查询更广泛的兴趣，而不是局限在某一特定话题类型 4：一般关联 • 借助先前查询，可以补全用户在当前搜索中的特定意图 • 更接近之前所笼统叙述的"上下文"情境感知的概念推荐系统为什么要感知情境？• 用户需要考虑当下的情境进行决策 • 购买服装：冬季 or 夏季？• 看电影：和孩子一起 or 和伴侣一起？• 网上冲浪：办公室 or 回家路上？推荐系统中的情境有哪些？ • 情境是那些在重复进行同一活动时可能发生变化的因素 • 看电影：时间、地点、同伴等 • 听音乐：时间、地点、情绪、场合等 • 派对或餐厅：时间、地点、场合等 • 旅行：时间、地点、天气、交通状况等 • 根据应用场景，收集和整理相关的信息情境感知的用户需求 • 描述用户所处状态及意图的，也不再是单一的上下文（查询或查询概念），而是一组复杂且相互关联的情境信息。• 单一解读某一种情境要素，可能无法得到用户当前状态的准确描述如何获取用户所处的情境？ • 显式获取 • 直接询问用户或预先定义情境，并让用户从中进行选择 • 隐式获取 • 通过应用的使用数据获取用户当前的位置等情境信息 • 推断获取 • 从用户的近期行为（序列）中进行推断和归纳情境筛选 • 显然，并非所有的情境信息都会影响到用户当下的决策 • 直接询问用户哪些情境信息对于当下是重要的 • 通过特征选择 • 例如，主成分分析（PCA）和线性判别分析（LDA）• 通过统计分析 • 统计测试，如信息增益、互信息、Freeman-Halton Test 检验

等如何将情境信息整合到推荐系统中？ • 情境感知推荐的三种范式 • 情境预过滤：事先根据情境信息筛选、分隔数据记录 • 情境后过滤：产生推荐结果后，根据情境信息进行筛选 • 情境建模常用的数据采样方法 • 最基本的采样技术为简单随机采用（Simple Random Sampling）• 对于所有对象，采用简单的等概率方式进行采样 • 一般采用两种方式进行 • 无放回采样：被采中的对象从整体中删除，仅可选中一次 • 有放回采样：被选中的对象不会从整体中删除，可再次被选中 • 两种方式无本质区别，但有放回采样较为简单（概率不变）• 更为复杂的方法包括分层采样（Stratified Sampling）• 对数据进行分组，从预先指定的组里进行采样启发式的采样规模确定方法 • 通过分组采样的方式，可以近似确定适当的样本容量 • 例如，将一个数据集分为 10 组，每组数量大致相等 • 组内的数据高度相似，而不同组的对象差异性较大 • 右图展示了不同样本容量下，每组至少取到一个数据点的概率（体现代表性）• 可见，至少 40 个点，才能保证 10 组都取到的概率接近 90% 渐进式采样 • 渐进式采样从训练结果的角度确定采样规模：• 从一个小样本开始，然后逐步增大采样规模 • 在模型的准确率趋于稳定的时候停止采样，从而确定采样规模 • 优点：不需要在一开始就确定采样的容量 • 缺点：计算开销大（需要多次迭代）维度归约的代表性方法：主成分分析 计算 n 个维度特征的平均值 $\mu_1,\ldots,\mu_n$ 原始特征减去各自对应的平均值得到归一化样本矩阵 X 协方差矩阵为：$C = \begin{bmatrix}\mathrm{cov}(x_1,x_1) & \mathrm{cov}(x_1,x_2)\\ \mathrm{cov}(x_2,x_1) & \mathrm{cov}(x_2,x_2)\end{bmatrix}$ 设 C 特征值为 λ，则 $|\lambda I - C| = 0$ 解得特征值 λ_1,λ_2 对应的单位特征向量为 α_1,α_2 则第一主成分为 α_1，第二主成分为 α_2 降维后各样本点新特征表示为:$y_1 = X\alpha_1$ 数据离散化旨在将连续属性变换为分类属性 • 例如，病人的年龄是一个连续值，但是在实际治疗中，往往仅需要一些年龄段的信息即可（如成年/未成年，儿童/成人/老人等）• 也可用于分类属性值过多的情况，例如大学的专业较多，可以用文理工农加以概括 • 对于特定数据挖掘问题，特别分类问题，通过合并减少类别数目是有益的非监督离散化 用于分类的离散化方法，最根本的区别在于其离散化过程是否有监督 • 即是否使用类别信息 • 对于不使用类别信息的非监督离散化方法，往往根据数据本身的特性进行离散，常见的方法包括等宽、等频率、等深等有监督离散化更注重问题导向，其目的在于取得更好的结果 • 基于熵（Entropy）的方法是最重要的有监督离散化方法之一 • 采用如下方法定义某个区间的熵 • 总的熵可以定义为加权和 • 显然，熵越小，区间内的纯度越高（标签越一致），越符合我们的要求 • 因此，一种做法是先进行二分，选择熵最小的点进行第一轮分割。进而，对其中具有较大熵（即纯度不高，信息较混乱）的部分再进行下一轮分割，以此类推。第十一节知识图谱概述 信息抽取的含义 • 从语料中抽取指定的事件、事实等信息，形成结构化的数据 • 被抽取的信息以预先定义的、结构化的形式描述。• 为后续的情报分析、自动文摘、问答系统等一系列应用提供服务。• 从整体文档中抽取细粒度、结构化信息信息抽取的内容 • 核心的 8 字方针："抽取实体，确定关系"• 实体：即命名实体，指文本中的基本构成块，如人、机构等 • 属性：实体的特征，如人的年龄、机构的类型等 • 关系：实体之间存在的联系，也称事实，如公司和地址之间的位置关系、公司与人之间的雇佣关系 • 事件：实体的行为或实体参与的活动信息抽取的基本任务 • 命名实体 NE、模板元素 TE、共指关系 CR、模板关系 TR、背景模板 ST 命名实体 NE（实体抽取） • 命名实体抽取是信息抽取最重要的任务 • 命名实体是文本中基本的信息元素，是正确理解文本的基础 • 狭义：指现实世界中具体或抽象的实体，如人、组织、地点等 如：水果摊/Org，老板郝哥/Person • 广义：还可以包含日期和时间、数量表达式等模板元素 TE（属性抽取） • 模板元素又称为实体的属性，目的在于更加清楚、完整地描述命名实体 • 通过槽（Slots）描述了命名实体的基本信息 • 槽：名称、类别、种类等 例如：郝哥表示，这都是大棚的瓜，你嫌贵我还嫌贵呢。 TE：瓜是大棚里产的（属性）共指关系 CR • 如果不同的命名实体表达了相同的含义，即为共指关系，也称为等价概念 • 共指关系的抽取任务在于抽取关于共指表达的信息 • 包括那些已在命名实体和模板元素任务中作了标记的，对于某个命名实体的所有表述 例如：我开水果摊儿的，能卖给你生瓜蛋子啊……要是不熟，我自己吃了它，满意了吧？ CR：生瓜蛋子和它均代指郝哥卖的瓜模板关系 TR（关系抽取） • 实体之间的各种关系，又称为事实 • 通过关系抽取，将实体关联起来，并为推理奠定基础 • 例如，职务（Post_of）、雇佣关系（Employee_of）、生产关系（Product_of）等 如：Post_of（老板，郝哥） Employee_of(水果摊，郝哥) 场景模板 ST（事件抽取） • 又称事件，是指实体发生的事件 • 例如：会议（Time<…>，Spot<…>，Convener<…>，Topic<…>）• 常见的新闻事件描述模板 5W1H Who、When、Where、What、Why、How 例如：那天（When），有一个人（Who）来水果摊（Where）买瓜，因为秤有问题（Why），用西瓜刀（How）"萨日朗"（What）了郝哥知识图谱的符号表示 • 由前述各种网络衍生而来，知识图谱呈现出类似的形式 • 由结点和结点之间的边组成，结点表示概念（或实体），边表示关系（或属性）。• 在数学上，知识图谱表现为一个有向图 • 方向表示主被动关系，表达相互关系请用双向图知识图谱的基本元素：节点 • 一般而言，知识图谱中的节点用来表示概念（Concept）和实体。• 实体（Entity / Object / Instance）• 能够独立存在的，作为一切属性的基础和万物本原的东西——黑格尔知识图谱的基本元素：边 • 一般而言，知识图谱中的边用来表示关系（Relation）和属性（Attribute）。• 关系：侧重实体（Entity）之间的关联，例如"高于"：姚明高于小四 • 属性：用于描述实体的特征，例如尺寸，颜色、组成等等 • 点和边组成知识图谱的基本单位：三元组（实体-关系-实体）知识图谱的优点 • 知识图谱至少可从以下三个层面提升搜索的效果：• 找到最想要的信息：不再需要用户自行浏览、阅读和总结，而将信息直接呈现 • 提供最全面的摘要：对搜索对象进行总结，使得用户获得更完整的信息和关联 • 让搜索更有深度和广度：构建完整知识体系，使用户获得意想不到的新发现多模态知识图谱 • 多模态知识图谱可笼统分为属性多模态与实体多模态两大类。属性多模态 实体 • 抽象概念 • 具有多模态的表示关系 • 实体之间的概念联系优点 • 易于从现有图谱中进行扩展，即通用图谱可以轻松扩展，概念和关系不用改变 • 可以推理视觉知识缺点 • 实体仍仅限于文本概念描述 • 1 个概念可以对应 N 张图片，但 1 个图片无法对应 N 张概念实体多模态 实体 • 多模态的实体关系 • 逻辑关系——相同模态 • 物理关系——跨模态 • 语义关系优点 • 视觉语义信息丰富，场景多源化 • 关系丰富缺点 • 图谱庞大 • 符号复杂第十二十三节知识抽取与表达 命名实体识别 • 识别出文本中的人名、地名等专有名称，和有意义的时间、日期等数量短语等，并加以归类。• 命名实体识别是信息抽取中的核心任务，它往往包含两个子任务 判别实体边界 判别实体类型命名实体识别的内容 • 一般按照 MUC-7 的定义，分为 3 大类，7 小类 • 实体类：人名、地名、机构名 • 时间类：日期、时间 • 数值类：货币、百分比 • 严格定义下哪些不属于命名实体？（部分例子）• 重复指代的普通名词：如飞机、公司等 • 人的团体名称以及以人命名的法律、奖项等：如共和国、诺贝尔奖等 • 非时间、日期、货币、百分比的数字命名实体识别方法（1）：基于词典 • 基于词典的识别方法（List Lookup）• 经常作为 NER 问题的基准算法（baseline）• 预先构建一个命名实体词典，出现在词典中的词汇即识别为命名实体 • 词典的来源：来自于领域公开数据，例如 人名/组织名，可以来自于黄页、电话簿、公开名单等 地点，可利用一些现有的地理信息列表 • 优点：简单快速，与具体语境/领域无关，容易部署和更新（只需更新词典）• 缺点（与基于词典/匹配分词存在类似问题）： 大部分情况下很难枚举所有的命名实体名 构建和维护词典的代价较大 难以有效处理实体歧义命名实体识别方法（2）：基于规则 • 采用手工构造规则模板，对符合规则的实体进行识别 • 选用特征包括统计信息、标点符号、关键字、指示词和方向词、位置词（如尾字）、中心词等 • 以模式和字符串相匹配为主要手段 • 例如：报告人：[韩伟力] 教授（[复旦大学]）• 可借助结构/半结构化或固定表达生成模板 • 基于规则模板的方法在深度学习技术发展之前被广泛使用，多数参加 MUC-7（1997）会议评测的系统，都采用了此方法 • 例如：[组织] 位于 [位置] 的总部 我们访问了 [华为] 位于 [东莞] 的总部 该类方法的局限性明显：不同句式意味着不同的模板 例如，上述句式可改为：我们访问了华为的 [总部]，它坐落于 [东莞] • 优点：当提取的规则能较精确地反映语言现象时，性能较好 • 缺点： 不同表达对应不同规则，导致规则库极其庞大，使用不便 规则往往依赖于具体语言、领域和文本风格不同领域的句法往往差异极大，如学术圈与二次元 代价太大，系统建设周期长、移植性差而且需要建立不同领域知识库命名实体识别方法（3）：基于统计 • 基于统计的命名实体识别方法是当下的主流方法 • 基于统计模型的分词方法，进一步抽象而言，可以得到一个序列标注问题 • 四类标注：B（词的开始）、M（词的中间）、E（词的结束）、S（单字词）分支一：基于分类的命名实体识别方法 • 将 NER 视作一个多分类问题，通过设计特征训练分类器的方法加以解决。• 事实上，早期基于统计的方法，需要精心设计大量的相关特征基本的词性标注方法 • 从思路上说，词性标注方法与实体识别，乃至分词，总体思路都是类似的 • 基于规则的方法：人工或通过大规模语料学习规则 • 核心思想是按兼类词搭配关系和上下文语境建造词类消歧规则 • 基于统计模型的标注方法：HMM 等面向词序列的方法 • 统计方法与规则方法相结合的词性标注方法 • 先基于规则排除明显歧义，再基于统计模型标注，最后人工校验基于统计的命名实体识别方法具有以下特点： • 对特征选取的要求较高，需要从文本中选择对 NER 有影响的特征来构建特征向量（尤其是早期工作，深度学习技术发展后相对要求降低）• 通常做法是对训练语料所包含的语言信息进行统计和分析，从中挖掘出特征 • 对语料的依赖也较大，目前缺少通用的大规模语料 • 对深度学习技术影响尤甚，特定专业领域影响最为明显 • 大部分技术仍需要进行人工标注训练数据分支二：基于序列模型的命名实体识别方法 • 与分词中的序列标注方法思路类似，区别在于标注的不同 • 针对命名实体的类别不同，引入了更多、更细致的标签种类 • 常用模型亦采用 HMM、CRF 以及各种序列深度学习方法（如 LSTM）等 • 深度学习方法的一般框架如下图所示：第一步，将单词和对应的标签实现初步（分布式）向量化 第二步，基于编码器对文本进行表征，其中通过 RNN 等方法捕捉上下文 第三步，基于解码器进行分类实体对齐 • 实体对齐（Entity Alignment），也称实体匹配（Entity Matching）• 指对于异构数据源知识库中的各个实体，找出属于现实世界中的同一实体。• 例如，不同药物可能在不同数据库中采用不同的名称 • E.g.，利君沙（琥乙红霉素片）• 一般而言，利用实体的属性信息判定不同源实体是否可对齐基础任务：基于表征的知识图谱实体对齐 • 近来，针对跨知识图谱（KGs）的实体对齐任务，研究者提出并改进了多种基于表征（Embedding）的模型 • 不仅利用实体的属性和语义信息，还利用实体间的关系 • 要求 KGs 的表征（包括关系表征和实体表征）落在同一个向量空间 • 换言之，这些模型更关注于关系三元组（relationship triple）• 一个类似的任务：跨社交网络用户匹配 • 不仅考虑用户画像，也考虑社交关系相似性实体链接 • 实体（entity），提及 or 指代（mention）• 如何将文本中的提及链接到知识库中的实体上呢？应用与挑战 • 应用 • 文本挖掘，知识图谱补全，信息检索和问答系统等 • 挑战 • 语言的多样性：同一个实体可能会有多个不同的提及 • 语言的歧义性：同一个提及可能会对应多个不同的实体任务框架 • 提及识别：其本质与前述分词/实体识别类似 • 候选实体生成 • 候选实体排序基于模式的关系抽取 • 首先由种子关系生成关系模式，然后基于关系模式抽取新的关系，得到新关系后，从中选择可信度高的关系作为新种子，再寻找

新的模式和新的关系。• 如此不断迭代，直到没有新的关系或新的模式产生代表性方法 1：DIPRE • 双重迭代模式关系提取（Dual Iterative Pattern Relation Extraction），由谷歌联合创始人 Sergey Brin 于 1998 年提出。（同年，PageRank 诞生）• 其大致思路在于先给定一些已知关系类型的种子实体对，找到出现了这些实体对的 Occurrences，再学习 Occurrences 的模式（Pattern）。• 进而，根据学到的模式，寻找更多符合该模式的数据，并加入到种子集合中，不断迭代这个过程以实现关系抽取DIPRE 的基本元素 • 元组：表示关系实例，如 <Foundation, Isaac Asimov> —<Title, Author> • 模式：包含常量和变量，例如?x , by ?y 的形式（可表示"title" by "author"）DIPRE 的基本假设 • 元组往往广泛存在于各个网页源中 • 元组的各个部分往往在位置上是接近的 • 在表示这些元组时，存在着某种重复的"模式"一种启发式方法（类似基于规则的基本方法） • 通过检查部分网站来获取潜在的"模式"，并利用正则表达式来描述 • 例如：letter = [A-Za-z.], title = letter5,40, author = letter10,30, (title) by (author) • 这种方法的缺陷是明显的（与基于规则的方法类似）：• 网站/系统存在特殊性，某个网站上的模式未必适用于其他网站 • 与规则类似的问题：人类不可能穷尽所有的潜在模式 • 更好的方法：DIPRE 正式登场 • 考虑模式和元组实例之间的双重影响关系 • 既要找到符合模式的元组，也要找到用于生成元组的模式DIPRE 的算法流程 • 首先，输入一组种子元组实例 R，如若干 <title, author> 的实体对 • 其次，基于种子实例集合 R，找到这些元组在网页中出现的内容 O （Occurrence），注意寻找的时候保留上下文信息（Surrounding Context）• 进而，基于找到的元组实例 O，生成模式 P • 最后基于生成的模式，找到更多的元组实例 R • 此时可选择停止，或返回第二步继续基于新实例生成新模式 • 注意，此时生成的新模式可能与之前的模式有所差异！• 从更多的元组实例中提炼种子元组，再去构建新的模式模式的概念与实例 • 将同一关系的不同实例在网页上所呈现的不同 Occurrence 中，相同内容保留下来，不同内容采用通配符取代，即可得到近似的模式基于 DIPRE 算法，生成模式的基本步骤 • 首先，将 Occurrence 归纳为 Order（元素的顺序）和 Middle（中间部分）• 其次，定义模式如下：• 模式的 Order 和 Middle，即为 Occurrence 集合的 Order 和 Middle • 模式的 URLPrefix、Prefix、Suffix，分别为 Occurrence 集合中最长的公共（Shared）URL 前缀与前、后缀。• 其他部分采用通配符填充远程监督的由来和意义 • 面向文本的关系抽取方法，最大的难点在于获取足够数量的、高质量的标注 • 人工标注开支过高，且存在主观性问题；而算法标注可能有累积误差 • 如何借助某种启发式方法，方便快捷地扩充训练数据？• 远程监督的思想：如果某个实体对之间具有某种关系，那么，所有包含这个实体对的句子都是用于描述这种关系。远程监督的基本思路 • 例如，我们已知"马云"和"阿里巴巴"之间是创始人/董事长关系 • 那么，我们默认以下包含这一实体对的句子，均描述这一关系 • "马云再谈悔创阿里巴巴：再有一次机会，尽量不把公司做这么大" • "马云：不当阿里巴巴董事长，但绝不等于我不创业了" • "港交所披露阿里巴巴集团招股书：马云持股 6.1% " • 接下来，我们将这些语料打包，从中训练用于关系识别的模型，并进而用于判断更多的实体对之间的关系。• 某种意义上，这一迭代思路类似于前面介绍的 DIPRE 算法。• 从上面的例子中我们可以看到，这一过程具有非常明显的局限性 • 语义漂移（Semantic Draft）现象：不是所有包含该实体对的句子都表达该关系，错误模板会导致关系判断错误，并通过不断迭代放大错误 • 例如，如果基于"港交所披露阿里巴巴集团招股书：马云持股 6.1%"这个句子进行训练，那么所有控股关系都会被错判为"创始人"。• 如何解决？• 可通过人工校验，在每一轮迭代中观察挑出来的句子，把不包含这种关系的句子剔除掉，但开支实在过高。远程监督的优化方案（1）：动态转移矩阵 • 尽管噪音数据不可避免，但是对噪音数据模式进行统一描述是可能的。• 例如，一个人的工作地点和出生地点很有可能是同一个地点，这种情形下远程监督就很有可能把 born-in 和 work-in 这两个关系标签打错。• 解决方案：引入一个动态转移矩阵，描述各个类之间相互标错的概率。• 在利用算法得到的关系分布的基础上乘以这一转移矩阵，即可得到相对更为准确的关系分类结果。• 然而，采用这种方式随机性较高，并不能完全保障其可靠性。远程监督的优化方案（2）：规则学习 • 远程监督试图通过使用知识库作为监督来源，从文本中提取实体之间的关系。这种启发式方法因噪声的存在可能会导致一些句子被错误地标记。• 针对这一问题提出一个新的生成模型，直接模拟远程监督的启发式标签过程。• 其中，设计相应的否定模式列表 NegPat(r)，专门用于去除错误的标签，即某些关系的判断是否为错误。• 对于单一关系的判断，可以通过这种方式进行比较高效的复检。但规则的生成成本较高远程监督的优化方案（3）：注意力机制 • 即使是被打入同一个包里的句子，不同句子对于训练关系判别模型的贡献度也不相同，这一贡献度可以采用注意力模型加以衡量。• 采用深度学习技术，获取对于整个句子的表示。• 进而，通过注意力机制，将最能表达这种关系的句子们挑选出来。• 最为有效的办法，但依赖于一个高质量的样本集合。事件抽取的概念 • 事件是信息的一种表现形式，其定义为特定的人、物，在特定时间和特定地点相互作用所产生的客观事实。• 例如，可对应先前所说的 5W1H 基本要素 • 一般信息呈现为句子级别（相比之下，关系往往表现为短语级别）• 推理任务（事件时序推理、事理推理）的前提事件抽取的基本要素 • 通常而言，事件往往包含以下基本要素：• 事件触发词：表示事件发生的核心词，多为动词或名词 • 相应的，事件触发词的检测与分类是事件抽取的基本任务 • 事件类型：与触发词相对应，往往可以通过触发词分类加以识别 • 例如，前例中的触发词 Pass away 对应着"死亡"的事件类型 • 事件元素：事件的参与者，主要由实体、时间等组成。• 例如，前例中的 Henry 是事件的主体 • 事件元素角色：事件元素在事件中充当的角色。• 例如，前例中的 Henry 在事件中是一个"受害者"的角色事件抽取的模板 • 通过触发词识别和分类，判定事件及其类型后，可以借助模板实现抽取。• 我们曾提到过模板元素 TE，其目的在于更加清楚、完整地描述实体 • 其中，模板元素通过槽（Slots）描述了命名实体的基本信息 • 槽的内容可包括名称、类别、种类等，不同类型的事件，对应的模板也不尽相同 • 通过事件元素与元素角色的识别，将元素填入模板合适的槽，即完成了事件抽取第十四节知识图谱与图计算 图表示学习算法 • 将图数据进行向量化表征，映射到一个低维的向量空间，在这个低维向量空间中，图的结构特征和语义特征得到最大限度的保留图的基本表示形式之一：邻接图矩阵 • 其中，每一行表示一个节点，1/0 分别表示与对应节点是/否连接 • 启发式想法：这一行可以视作该节点的一个表示向量 • 下游应用：该思路可用于最基础的图聚类问题 • 局限性：未能充分融入节点结构信息，节点属性信息无法加入几种不同的图表示学习算法 • 基于随机游走的图表示学习 • 基于随机游走的邻居节点序列，挖掘图结构信息 DeepWalk、Node2vec、Metapath2vec、LINE… • 基于图神经网络的图表示学习 • 利用神经网络来学习图结构数据，提取和挖掘图结构中的特征和模式 GCN、SGCN、VGAE、GraphSAGE、SDNE、GAT…知识图谱常见的图学习任务 知识图谱 ＋ 图学习：表征学习 • 为知识图谱中的实体和关系提供向量表示，同时保留语义信息 • 下游任务：• 知识图谱补全、关系预测 -> 稍后介绍 • 推荐系统、问答系统 -> 下节课介绍知识图谱 ＋ 图学习：关系抽取/补全 • 图学习可以帮助解决困难的关系抽取/补全 • 尤其是 OOKB 的实体问题为知识库中出现的关系补全未知实体为出现的实体补全未知的关系知识图谱 ＋ 图学习：实体对齐 • 知识图谱对齐（上上节课已进行了介绍）• 将描述同一目标的多个实体进行合并，减少冗余，同时增强完整性知识图谱 ＋ 图学习：图谱推理 • 借助符号逻辑或表征学习，实现基于图谱的推理过程，增强可解释性知识图谱推理 • 主要关注围绕关系的推理 • 基于图谱中已有的事实或关系来推断未知的事实或关系 • 许多现实中的问题（如链接预测、因果推理、基于 KG 的问答和推荐等）都可以表述为图谱推理知识图谱推理的分类 • 基于符号逻辑的知识图谱推理 • 显式的知识表示方法，一般需要人工定义 • 可解释性好 • 基于表示学习的知识图谱推理 • 易于捕获隐式知识 • 对机器友好，但不利于人理解知识图谱中的类型语言 • Abox（Assertion Component）• 引入概念、关系名称（A 是 B 的一种）• 例子：北京；北京是一座城市 • Tbox（Termination Component）• 表示对象是否属于概念 • 表示对象间是否存在关系 • 例子：首都（北京，中国）基于规则的知识图谱推理 • 知识库 KR = <T, A> T: Tbox, A: Abox • 特性：• 准确，可解释性好推理路径可以直接解释结果 • 存在效率问题：真值表规模随实体指数增长 • 对隐性关系效果很差基于符号表示的演绎推理 • 用 axioms 或规则进行简单的逻辑推导 • 效率问题：规则学习的主要困难在于确定可能的规则结构和搜索支持三元组以决定其真值时的巨大搜索空间 • 三元组稀疏性问题：它依赖于图的结构或路径来提取特征基于表示学习（嵌入）的归纳推理 • 当有大量的关系或三元组需要推理时，基于嵌入的推理更有效率 • 可解释性问题：我们知道预测的结果，但不知道为什么 • 三元组稀疏性问题：对三元组不足的稀疏实体的编码能力差基于图神经网络的图谱推理 • 按关系类型的图卷积实体分类和链接预测任务知识图谱上的关系推理任务 • 链接预测 • 给定两个实体，预测它们之间是否存在 r 关系 • 实体预测 • 给定头实体和关系，预测未知的尾实体 • 事实三元组预测 • 给定一个三元组判断其是否为真或假知识图谱嵌入模型 • TransE • 受 Word2vec 翻译不变性的启发：v(king)-v(queen) = v(man) -v(woman) • TransH • 通过向量 wr 将实体投影到关系 r 对应的超平面上，解决一对多问题 • TransR • 通过矩阵 Mr 将实体进行坐标转换到关系 r 对应的关系空间中 • 不同的关系对应实体不同的属性什么是图谱补全？ • 知识图谱"生来"不完整，多数的现有知识图谱都是稀疏的，由此引出了图谱补全来向知识图谱添加新的三元组 • 我们需要做的，是基于图谱里已有的事实，去推理出缺失的事实 • 本质上说，图谱补全和推理任务非常类似 • 推理的结果可以作为新"知识"加入图谱 • 给定知识图谱 G = E, R, F，其中 E 表示所有实体的集合，R 表示所有关系的集合，F 为所有三元组的集合 • 知识图谱补全的任务是预测出当前知识图谱中缺失的三元组 F'=(h, r, t) | (h, r, t) F, r R • 具体子任务包括头、尾实体预测和关系预测：• 给定三元组 (__,r,t)，预测头实体 • 给定三元组 (h,r,__)，预测尾实体 • 给定三元组 (h,__,t)，来预测关系图谱补全方法：Rule-based Reasoning（规则推理）Embedding-based（基于表征的方法）Pathfinding-based（基于路径查找的方法）图谱补全方法：引入大规模预训练模型 • 把常规的图谱三元组补全问题纳入预训练语言模型的处理框架中来，将知识图谱补全的任务视作序列分类任务 • 为了让预训练模型适应图谱中的 fact，将模型在具体任务上继续 fine-tuning • 对于头、尾实体补全问题：• 将三元组转换为文本序列模式 • 获得 [CLS] 的表征向量并进行二分类，判断该三元组是否成立 • 把常规的图谱三元组补全问题纳入预训练语言模型的处理框架中来，将知识图谱补全的任务视作序列分类任务 • 对于关系补全问题：• 模型也可以完成关系分类，输入两个实体或实体描述，输出多类分类别 • 采用负对数损失函数第十六节社会网络 社会网络的基本元素 • 从数学抽象的角度，可以将社会网络表示为图（Graph）的结构 • 节点（Node/Vertex）：用于表示网络中的实体，如社会网络中的人 • 边（Link/Edge）：用于描述网络中的关系，如人们之间的社交关系社会网络的基本元素：有向/无向边 • 网络中的边可能有向，也可能无向，各自表达不同含义社会网络的基本元素：邻居、出入度 • 对于任意节点而言，与其直接相连的节点被称作"邻居"（Neighbor）• 对节点 v 而言，往往采用 N(v) 表示其邻居集合 • 在有向网络中，入边邻居和出边邻居集

合应加以区分 • 同时，节点所连边的数量（或邻居的数量）被称作"度"（Degree）• 对节点 v 而言，往往采用 dv 表示其度数 • 同样，出度与入度应加以区分 • 显然，对于一个网络而言，节点出度之和等于入度之和等于边数 • 一般而言，真实网络中的节点度数往往符合幂律（Power-law）分布 • 少数节点拥有大多数的边 • 这些少数节点即形成了所谓"影响力节点"，也就是俗称的"大 V"社会网络的基本元素：连通性 • 两个节点是连通的（Connected），当且仅当节点之间存在一条路径（Path）• 注意：这并不意味着两个节点之间是直接相连的 • 进而，一个图是连通的，当且仅当图中任意两个节点都是连通的 • 对于有向图而言，有强 / 弱连通的区别 • 任意两个节点之间存在双向的连通路径，即为强连通 • 忽略方向的前提下任意两个节点之间存在一条路径，即为弱连通 • 如果存在两个节点无法连通，则图是不连通的社会网络的基本元素：连通组件 • 对于一个图而言，其中的一个连通组件（Component），即一个连通的子图 • 换言之，该子图的任意两个节点之间都是连通的 • 在有向图中，我们将强连通组件定义为该有向图的一个强连通子图 • 即任意两个节点之间存在双向连通路径 • 相应的，弱连通子图对应着弱连通路径节点的角色不同 • 研究者也发现，在同一个社会网络中，不同节点往往也扮演不同角色 • 例如，一个部门，往往有以下分工：• 部门经理：负责领导部门 • 技术专家：负责提供技术指导 • 项目经理：负责外部需求沟通节点的角色不同 • 第一类常见的节点角色，即所谓的"意见领袖" • 意见领袖的挖掘，可参考 HITS 算法及两类网页的区分 • 权威（Authority）网页与枢纽（Hub）网页的区分 • 权威网页：指某个领域或某个话题相关的高质量网页 • 如科研领域的中科院之声，视频领域的优酷与爱奇艺等 • 中心网页：类似中介，指向了很多高质量的权威网页 • 如"hao123"，各个浏览器自带的首页节点的角色不同 • 第二类常见的节点角色，即所谓的"结构洞" • 结构洞的主要作用，在于为组织引入外部的信息 • 例如，不同部门、不同社团之间的信息沟通："项目经理"的角色 • 因此，一种直观的结构洞的判定方法为：• 如果一个节点，移除该节点就会使网络变成多个连通组件，则该节点即为一个结构洞 • 结构洞的主要作用，在于为组织引入外部的信息 • 通过连通组件判定，假设可能过强，计算也较为不便 • 另一种衡量方式：聚集系数 • 某个节点的聚集系数为：它的任意两个好友也互为好友的概率（比重）• 显然，聚集系数越低，该节点作为中介的作用也就越大 • 结构洞的"意义"：各方沟通的桥梁 • 相应的，这种"意义"也成为了结构洞的"权力" • 例如，网络交换实验中，结构洞可能获得更大收益 体现出结构洞具有排他性，该权力来自于"选择的自由"信息级联现象 • 信息级联（Information Cascade），直译为"信息瀑布" • 这个词用来描述人们在信息流中的"从众"行为，非常生动形象 • 信息级联现象，可以描述人们从他人行为中获取消息/进行决策的这一过程 • 这个过程具有鲜明的"传播"特性信息传播的元素 • 一般而言，我们将信息传播过程中涉及的元素归为以下三类：• 发送者（Sender），也称作信息源（Source）或"种子节点"（Seed）• 指在信息传递开始时拥有信息的那一小部分用户集合 • 接收者（Receiver），指作为潜在传播目标的广大用户集合 • 接收者集合的规模要远大于前者，且不同发送者的目标集合存在重叠 • 媒介（Medium），指传播过程发生的平台 • 例如，寻找红气球比赛中的社交媒体 / 论坛等信息级联中的基本假设 • 首先，信息级联发生在一张有向图上 • 对于无向图，可以将其连边转化为双向边进行处理 • 对于网络中的节点来说，这些边就是信息传递的媒介 • 其次，每个节点仅能将信息传递给与其直接相连的节点 • 例如，大 V 可以将信息传递给其粉丝，但不能传递给未关注他/她的人 • 信息传递的局部可达性！• 特别需要注意的是，网络中节点的状态是二元（Binary）的 • 激活（Active/Activated）：表示节点已经收到了这一信息 • 未激活（Inactive）：表示节点尚未接收到这一信息 • 不存在薛定谔的状态！ 一个小问题：什么情况下算是激活？• 接收到信息，并且尝试将信息传给别人才叫激活 • 两个动作缺一不可（是否合理？）• 已激活的节点才具备激活其他节点的能力 • 而且，激活能力有一定的时限！• 传播中存在着时间"轮次"的概念 • 类似于传染病模型的设定 • 这个设定是否普遍合理？如何确定一个合理的时限？• 后面我们会展示去除这一约束的特殊模型 • 激活是不可逆的过程 • 可以从未激活到激活，但不能从激活退回未激活 • 这个约束又是否普遍合理？• 核心争议在于：是否接受信息的"二次传播" • 后面我们同样会展示去除这一约束的特殊模型基本模型（1）独立级联模型 • 独立级联模型 • "独立"体现在，每次激活都是一次独立事件，相互不产生影响 • 激活的尝试相当于一次以特定概率抛硬币的过程 • 同时，每个已激活节点，只有一次机会尝试激活他/她的未激活邻居节点 • 一旦尝试失败，不会再有第二次尝试机会独立级联模型中的重要概念：轮次 • 如果某个节点在第 t 轮被激活，那么，他仅有一次机会，即仅能在 t+1 轮，尝试激活他所有未被激活的邻居节点 • t = 1 时，仅有种子节点可以尝试激活其他节点 • 对于节点 v 而言，他激活邻居节点 w 的概率采用 Pvw 表示 • 以 Pvw 为概率进行抛硬币 • 整个传播过程直到所有节点都被激活，或没有新节点可以被激活为止 • Pvw 的取值方式：• 基本传播模型里，为简化考虑，一般将 Pvw 设为 1/N，N 为 w 节点的入边的数量 • 当然，也有实现确定带权图的做法（如后续的例子）• 此外，也可以基于主题等因素对 Pvw 进行扩展基本模型（2）线性阈值模型 • 线性阈值模型 • 另一种视角：将信息传递过程视作多人影响的叠加过程 • 一个用户会被某个信息激活，如果来自他已激活邻居的影响超过某个阈值 $\sum_{v_j\in N_{\mathrm{in}}(v_i),v_j\in A_{t-1}} w_{j,i} \geq \theta_i$ • 阈值预先设定，往往为从 [0,1] 均匀分布中随机抽取的一个数值 • 更复杂的情况：可以根据用户对信息的兴趣等决定 • 线性阈值模型与独立级联模型的区别：随机性 • 对于独立级联模型来说，其随机性在于抛硬币的过程 • 因此，独立级联模型是完全随机过程，每一次的结果可能都不相同 • 一般需要重复多次以确定个体节点被激活的可能性 • 对于线性阈值模型来说，其随机性在于边权重/阈值的确定 • 如果采用启发式方法确定边权/阈值，则该方法结果完全由方法设计决定 • 一旦确定边权/阈值（无论何种方式），其结果具有唯一性信息传播最大化问题 • 为什么会有传播最大化问题？• 口碑营销的常见应用：通过优惠来吸引潜在客户 • 例如，通过发放优惠券 / 赠品的方式来扩大客户群 • 然而，商家的预算是有限的 • 因此，往往仅能通过收买少数用户来扩散消息 • 这个时候，选择目标用户就至关重要！• 传播最大化问题的定义 • 假定初始的种子节点集合为 S，预期激活的节点集合为 f(S) • 信息传播最大化的目的，在于限定 S 集合的前提下，最大化 f(S) 的规模 • 常见的约束为限定 S 集合的规模，即 |S| • 如果 S 集合中的节点价值不等，则可将约束进一步扩展 • 解决传播最大化问题的启发式思路：寻找网络中最具影响力的节点信息传播最大化的启发式方法 • 启发式方法（1）PageRank 及其衍生模型 • 在网页排序部分，我们曾经介绍过，PageRank 可用来衡量网页权威性 • 因此，PageRank 及其各种衍生算法如 HITS 都可以采用 • 启发式方法（2）核心性度量 • 用于衡量网络中最重要的节点。常见核心性度量如度、紧密度、介数等一般化的信息传播最大化方法 • 基于前述的传播模型（例如：ICM / LTM），对于给定 S 集合计算 f(S) • 此时，传播最大化问题可转化为一个带约束的最优化问题 • 以 f(S) 为目标函数，找到一个集合 S，使得 f(S) 的期望最大 • 同时，S 应满足 |S| = k （即只选择 k 个节点作为初始节点集合）• 注意，是期望最大（因为整个过程是随机过程）• f(S) 的一些有趣的性质：子模特性 • 1. f(S) 函数是非负的（显而易见）• 2. f(S) 函数是单调非减的，即 $f(S+v) \geq f(S)$ • 也很好理解，新增加一个节点，至多不增加新激活，不至于减少 • 3. f(S) 函数是具有子模特性（Submodularity）的，即：• 对于任何集合对 S，T，且满足 $S \subseteq T$ 时，给定节点 v，有 $f(S\cup\{v\}) - f(S) \geq f(T\cup\{v\}) - f(T)$ 信息传播最大化问题的解决方案 • 基于前述问题定义，我们有坏消息，也有好消息 • 坏消息：在 ICM / LTM 等模型定义下，传播最大化问题是个 NP 难问题 • 简要证明思路：将这两个模型定义下的传播最大化问题，归约为集合覆盖（Set Cover）和节点覆盖（Vertex Cover）问题 • 好消息：由于 f(S) 函数具有子模特性，我们可以采用贪心算法近似求解 • 以空集合为起点，即初始 S = 空集 • 经过 k 次迭代，每次选择最大化 $f(S\cup\{v\}) - f(S)$ 的节点 v • 效果如何？论文证实贪心算法可以实现至少 (1 -1/e) 的近似效果 • 这就意味着，贪心法所得 S 可以激活至少 63% 最优解能激活的节点数独立级联模型的局限性 • 独立级联模型具有易于求解，假设直观的优点，但也存在一些缺点 • 小问题 1：每个节点只有一次传播信息的机会，是否过于苛刻？• 实际情况下，只要信息还在，就可以持续输出影响 • 小问题 2：节点状态未必二元化，也难以获得清晰明确的激活轮次 • ICM 适合类似微博等具有明确转发记录的场景 • 然而，很多场景下并没有明确的信息传播轨迹 • 节点是否真的被激活？何时被激活？无法回答稳定状态传播模型 • 独立级联模型的松弛版本：稳定状态传播（Steady State Spread, SSS）• 对 ICM 的改动体现在以下两点 • 节点状态不再二元化，而是引入一个变量表示当前被激活的概率 • 薛定谔的节点出现了！• 如果被激活概率不为 0，则节点可以持续对外输出信息 / 影响 • 稳定状态传播模型的核心公式：$1-\pi(i) = \prod_{l\in N(i)}(1-\pi(l)\cdot p_{li})$ $\pi(l)$ 表示 1 节点的当前状态，即被激活的概率当 $\pi(l) = 0$ 时，显然不影响邻居节点激活状态稳定状态传播的一个隐患：不合理的反向传播 • 我们观察形如下图的网络结构，假设信息源为节点 1 • 在此种情况下，信息只可能途经节点 4 到达节点 5 • 在 ICM 中，节点 5 被激活意味着节点 4 已被激活，但 SSS 呢？• 由于 SSS 只有薛定谔的激活状态，节点 4、5 都只有激活概率 • 在此种情况下，将出现节点 5 反向提升节点 4 激活概率的现象 • 然而，这在道理上是说不通的（子节点激活父节点？）• 一种解决方法是遍历网络，找到所有类似这样的子节点 • 但这种做法显然过于费时费力，且结构特性难以判断 • 另一种：启发式方法，限制每个节点的迭代次数，超过一定轮次阈值，则该节点被激活的概率不再更新 • 阈值大致根据节点到信息源的最长 / 短路径决定信息级联的另一个小问题 • 独立级联模型在场景应用上还有另一个局限性：信息传播与接收的捆绑 • 回想一下，ICM / LTM 等模型在假设有一个前提，一个转发行为 = 传播 ＋ 接受，两者缺一不可 • 但实际上，两者不可混为一谈 • 例如，有些人可能看到并接受了信息，但由于种种原因并没有转发；而另一些转发的人可能实际上并没有接受信息 $\arg\max_S F(S) = E(|I(S)|) + E(|\cup_{a\in I(S)} N(a)|), s.t. |S| = k$ • 信息覆盖最大化问题同样具有信息传播最大化问题的性质 • 因此，可以采用类似的贪心算法加以求解 • 更进一步实现信息传播与接收的解绑，修改模型框架 • 第二种思路：单独对信息接受过程进行建模 $F(S) = Adopt(S) = \sum_{u\in V}[f_u(A_u)]$ • 其中，引入函数 fu(Au)，描述 u 节点接受信息的概率 • 显然，这一概率与有多少个邻居已经接受了信息正相关 • Au 的定义：$u \cup N^{in}(u) \cap Active(S)$ • 更进一步实现信息传播与接收的解绑，修改模型框架 • 第二种思路：单独对信息接受过程进行建模 • 由此，衍生出第二个新问题：信息接受最大化问题 • 即，什么种子节点集合会导致接受信息的节点数期望最大 • 该问题在 fu(Au) 符合一定特性时，同样与信息传播最大化具有类似属性 • 事实上，信息接受最大化问题可视作一个更为一般化的框架 • 当满足如下条件时，该问题可退化为信息传播最大化问题：$f_v(A_v) = \begin{cases}1 & v\in A_v\\ 0 & v\notin A_v\end{cases}$ • 当满足如下条件时，该问题可退化为信息覆盖最大化问题

$$f_v(A_v) = \begin{cases}1 & A_v \neq \varnothing\\ 0 & A_v = \varnothing\end{cases}$$