搜索系统算法解析-OAA

倒排召回

倒排召回（Inverted Index Retrieval）是搜索系统中最经典、广泛应用的一种召回技术，其核心是利用倒排索引的高效结构，在海量文档中快速找到包含查询关键词的文档集合。相比于业界主流的向量召回技术，倒排召回由于其易于调试、成本低、检索效率高等优点，在一些场景下仍有不可替代的作用。

基本概念

Query切词

词权重

Term同义改写

倒排索引

倒排索引是一个词到文档的映射，记录了每个词项（term）在哪些文档中出现，以及它们的位置或频率等信息。

- **索引项（Term）**：文档中每个独立的词
- **倒排表（Posting List）**：包含每个词出现的文档 ID，以及可能的词频、位置等元信息
- **数据结构**：`Term -> [DocID1, DocID2, DocID3, ...]`
- **构建过程**：
  - 对文档进行分词，并去停用词、归一化
  - 按词项建立倒排表，并存储相关信息（如文档频率、位置）

特别注意的是，为了保证召回的文档与查询高度相关，避免漏召回和误召回，需要确保检索词Query和文档Doc的**分词一致性**，关于索引分词的详细介绍见Query分析中的Query切词。

倒排召回流程

倒排召回利用倒排索引实现关键词检索：

- 查询分词： 将用户的查询语句分解为若干词项，召回的效果强依赖于**分词粒度**（详细介绍见Query分析中的Query切词）
- 匹配倒排索引： 根据每个词项，从倒排索引中提取对应的文档列表
- 文档合并： 结合多个词项的文档列表，使用交集、并集或加权策略获取最终候选文档集
- 排序： 根据词频、文档权重（如 TF-IDF）、BM25或其他评分规则对候选文档进行初步排序

文本匹配计算

对于倒排索引召回的文档集合，通常需要通过文本匹配分计算查询Query和文档Doc之间的相似度，为每个召回的文档打分，以评估其与查询的相关性。尽管在语义理解方面存在局限性，但其简单高效的特点使其在现代搜索系统中仍被广泛使用，常作为更复杂模型的基线或初筛方法。

常见的相似度匹配算法有TF-IDF和BM25，其基本思想是，Query中的词在Doc中出现次数越多，则Query和Doc越相关。

TF-IDF

TF-IDF（Term Frequency-Inverse Document Frequency）用于衡量一个词在文档中的重要性，其结合了词在文档中的频率（TF）和词在整个语料库中的稀有性（IDF）来作为判断依据。

TF-IDF的计算公式为：

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \tag{1}$$

其中，t为目标词项（term），d为目标文档，D为语料库（包含所有文档）。

- **词频（TF）**

$$\text{TF}(t, d) = \frac{f(t, d)}{\sum_{t' \in d} f(t', d)} \tag{2}$$

  - $f(t, d)$：词 t 在文档 d 中出现的次数
  - $\sum_{t' \in d} f(t', d)$：文档 d 中所有词出现的总次数
  - TF 是一个局部指标，表示一个词在当前文档中的重要性。词在文档中出现得越多，其重要性可能越高

- **逆文档频率（IDF）**

$$\text{IDF}(t, D) = \log \frac{|D|}{1 + |\{d \in D : t \in d\}|} \tag{3}$$

  - $||D||$：语料库中文档的总数
  - $||d \in D : t \in d||$：包含词 t 的文档数量（文档频率）
  - IDF 是一个全局指标，表示词在整个语料库中的稀有性。词在整个语料库中出现得越少，其区分性越强

通过将 TF 和 IDF 相乘，TF-IDF 能有效提升那些对当前文档具有区分性的关键词的权重，降低全局常见但区分性较弱词的权重。

BM25

BM25 对 TF-IDF 进行了改进，BM25的计算公式为：

$$\text{BM25}(t, d, D) = \text{IDF}_{\text{BM25}}(t, D) \times \text{TF}_{\text{BM25}}(t, d, D) \tag{4}$$

其中，t 为目标词项（term），d 为目标文档，D 为语料库（包含所有文档）。

- **词频（TF修正）**

$$\text{TF}_{\text{BM25}}(t, d, D) = \frac{f(t, d) \cdot (k_1 + 1)}{f(t, d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{\text{avgdl}}\right)} \tag{5}$$

  - $f(t, d)$：词 t 在文档 d 中的词频（Term Frequency）
  - $||d||$：文档 d 的长度（即词的总数）
  - avgdl：语料库中文档的平均长度

- k_1：调节词频饱和程度的参数，通常取值范围为 [1.2, 2]
- b：调节文档长度归一化的参数，通常取值范围为 [0, 1]
- 词频饱和：对 $f(t, d)$ 引入非线性加权，防止高词频对得分过度影响
- 文档长度归一化：引入参数 b 平衡长文档和短文档

- **逆文档频率（IDF修正）**

$$\mathrm{IDF}_{\mathrm{BM25}}(t, D) = \log \frac{N - n(t, D) + 0.5}{n(t, D) + 0.5} \tag{6}$$

- N：语料库 D 中的文档总数
- $n(t, D)$：包含词 t 的文档数
- 增加了平滑项，避免在极端情况下分母为 0，同时使常见词的权重更低

BM25 的改进体现在对 TF 进行非线性调节（饱和处理）和引入文档长度归一化，同时对 IDF 进行了平滑处理，解决了 TF-IDF 中的高词频偏差和长文档劣势问题。

丢词召回（松弛召回）

丢词召回是一种通过部分匹配扩展召回范围的有效策略，通过策略移除部分查询词项（即“丢词”），提升对包含部分关键词但不完全匹配的文档的召回率。有效缓解因用户查询中的关键词缺失或文档内容匹配不足导致的召回不足问题。

给定检索词 Query 可分词为 $t_1, t_2, \ldots, t_n$，n 为可分词的数量，则基于倒排索引的布尔检索查询串为：

$$t_1 \text{ and } t_2 \text{ and } \ldots \text{ and } t_n \tag{7}$$

即要求检索召回的文档中需要同时包含词 t_1 到 t_n。丢词召回则根据策略去除查询串中部分词 t，常见的丢词策略有：

- **基于词权重**：通过Query分析给到的词权重信号，丢掉低价值的词（详细算法逻辑见Query分析中词权重）
- **基于模型预测**：训练一个排序模型，对多种丢词后的查询串（基于词权重）进行打分，通过打分排序选择最佳的丢词方式

同义词召回

倒排召回的布尔查询串同时可以结合同义词扩展查询词项，提升召回率，同义词信号的详细算法逻辑见Query分析中的Term同义改写。

> 假设有`Query`可以被分词为`[A, B, C]`，基于布尔检索的召回会根据分词构建查询串表达式：`A and B and C`，即检索结果`Doc`的文本内容需要同时包含Term词`A`、`B`、`C`，若`A`有同义Term词`A1`、`A2`、`A3`，`B`有同义Term词`B1`，则查询串表达式可以构建为：`(A or A1 or A2) and (B or B1) and C`。

当以结合同义词的布尔查询串进行检索时，由于原Term词和同义Term词是`OR`的关系，即检索命中同义词的笔记也会被召回，由此可以较大的提高相关结果召回量。

Query锚点召回

对于高频查询词，利用其历史点击数据来召回文档是一种有效的策略。这些文档由于历史点击数据的支撑，通常与查询词相关性较高。另外，通过利用现成的历史数据，可以减少实时计算的复杂度和资源消耗。

具体的实现方式为：

1. 基于搜索点击日志挖掘每篇文档历史上点击关联的Query作为召回锚点
2. 将每个Query映射到其点击过的文档列表及其相关性评分以构建Query-Doc倒排表
3. 召回策略：
    - **直接召回：** 当接收到一个高频查询词时，直接从倒排映射表中召回对应的文档
    - **加权召回：** 根据历史点击的频率或时间衰减等因素对文档进行加权，优先召回那些被多次点击或最近被点击的文档

利用高频查询词的历史点击数据进行召回是一种有效的策略，尤其适用于那些查询模式相对稳定、用户行为数据丰富的场景。通过结合其他召回策略和动态更新机制，可以进一步提高召回结果的质量和多样性。

总结

倒排检索召回具有极高的检索效率，适合处理大规模文本语料。但是倒排召回仅基于词项匹配，无法捕捉深层次语义关联，另外长查询的分词可能导致稀疏性问题，降低召回效果。