**物联网第三章大作业实验报告**

富诗涵2019213493

**1.数据源准备**

document 和 query里的“：”， “，”是中文字符，先替换成英文字符方便后续操作

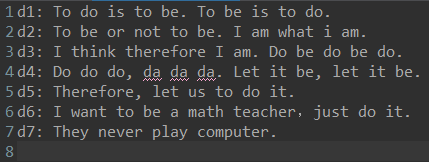


figure 1 document数据

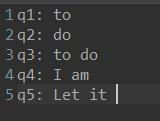


figure 2 query数据

**2.实现算法与结果分析**

**2.1布尔检索模型**

布尔模型是最早的IR模型，也是应用最广泛的模型，它目前仍然被应用于商业系统中。

该模型需要：①文档D：一个文档被表示为关键词的集合②查询式Q:查询式(Queries)被表示为关键词的布尔组合，用“与、或、非”连接起来，并用括弧指示优先次序。③匹配F:一个文档当且仅当它能够满足布尔查询式时，才将其检索出来

首先我将document中的数组按照指定格式读出，将其全部转化为规格字符（这里用英文形式的输入作为统一标准），并将其全部转化为小写表达，然后根据query中给出的搜索词构建出文档是否含有特定字符串的存储数据，其结果如图所示：

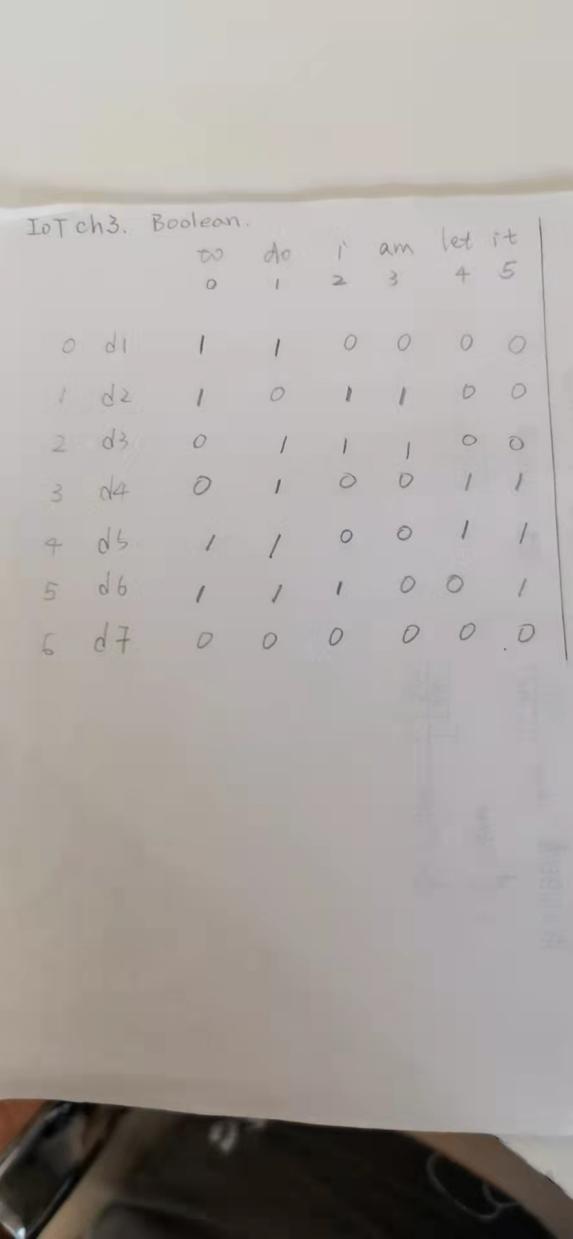


figure 3 构建结果存储于ArrayList<String> documentContent中

通过GUI，用户有两种选择方式，可以使用query文档，选择具体搜索qi。

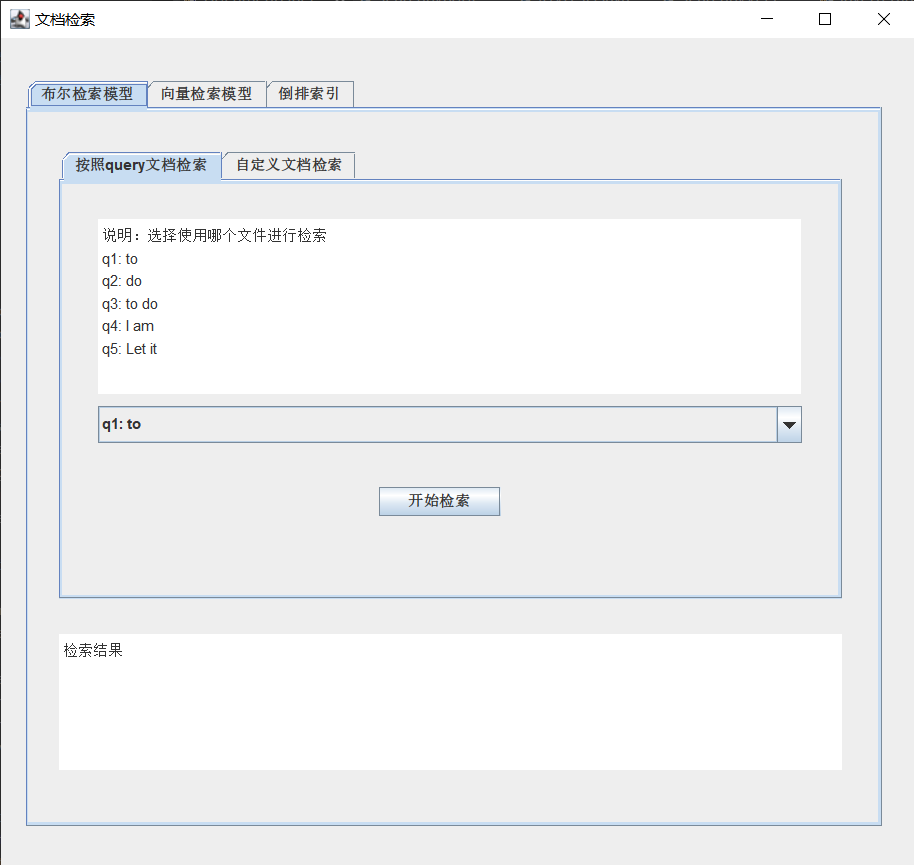


figure 4 布尔检索模型-按照query文档检索页面

例如，选择q1，点击开始搜索后，搜索结果将被显示在下面的结果框内。

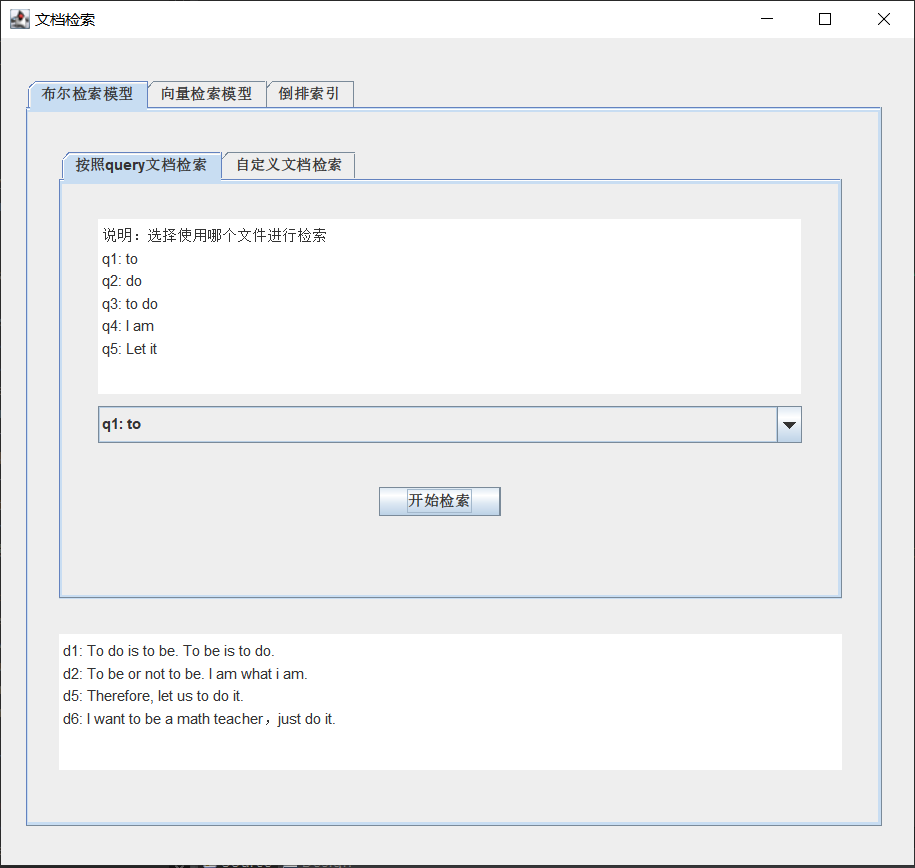


figure 5 布尔检索模型-按照query文档检索结果示例页面

除此之外，用户也可以自定义搜索，通过点击想要包含的（AND）与不想包含的（NOT），进行自定义搜索。

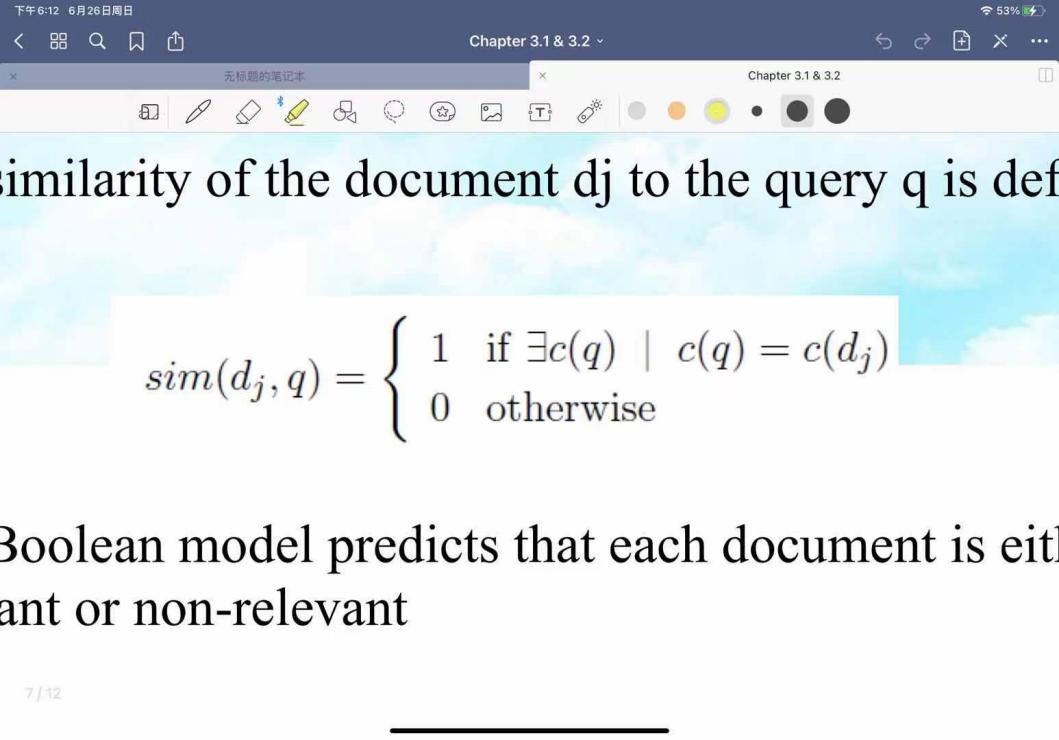


figure 6 布尔检索模型-自定义文档检索页面



figure 7 布尔检索模型-自定义文档检索结果示例页面

在布尔检索模型中，文档与搜索命令的相似度只有0、1之分。其数值由下列方程式定义。当sim为1时，找到文档，反之则未找到。“有就是有，无就是无。”

 （1）[1]

**2.2向量检索模型**

向量检索模型类似于VSM，是把对文本转换为空间向量, 通过向量的计算来处理文本。它把文本内容转换为空间向量 (一般使用词的tf-idf值作为向量来表征文本内容)，并计算向量之间的相似度来度量文档间的相似性 (常用余弦值来度量相似性)。

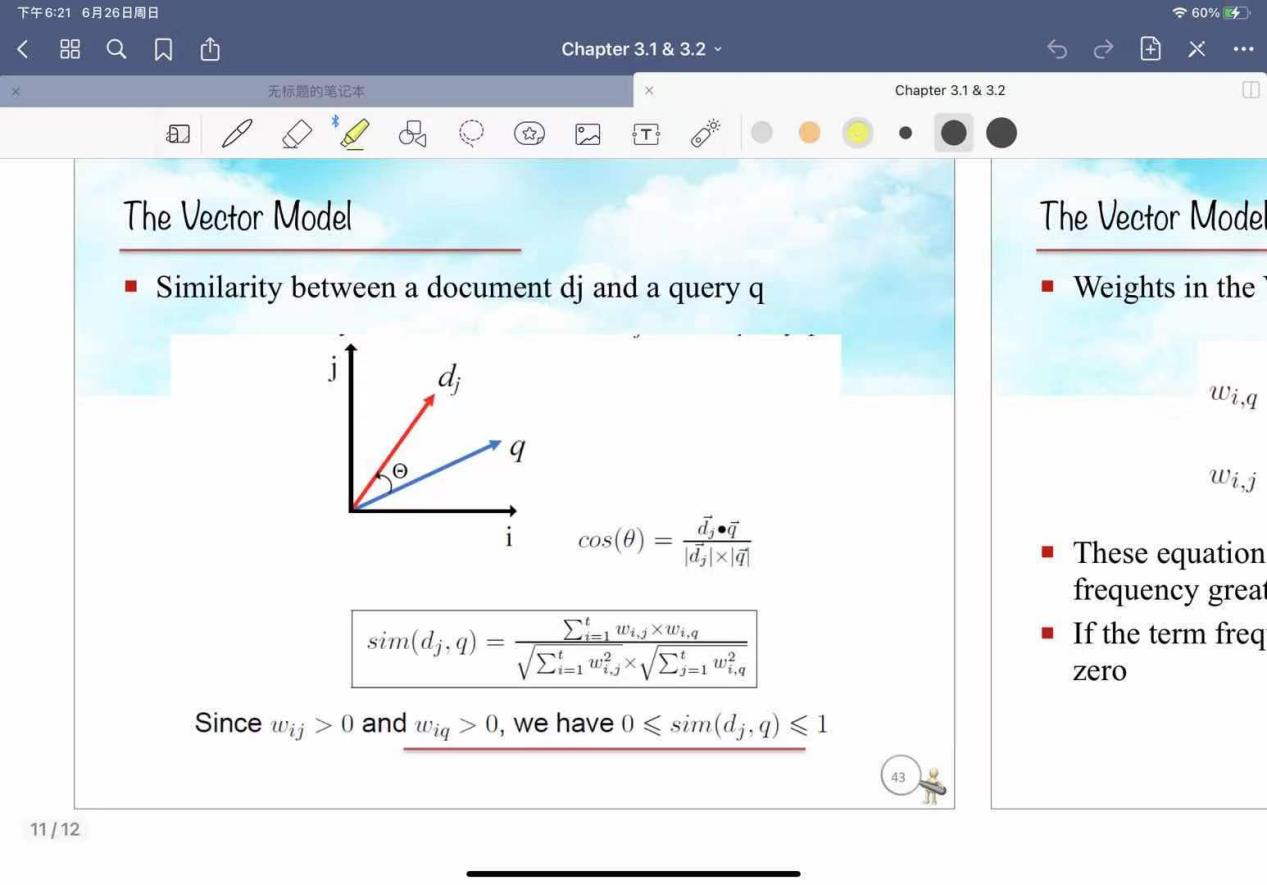
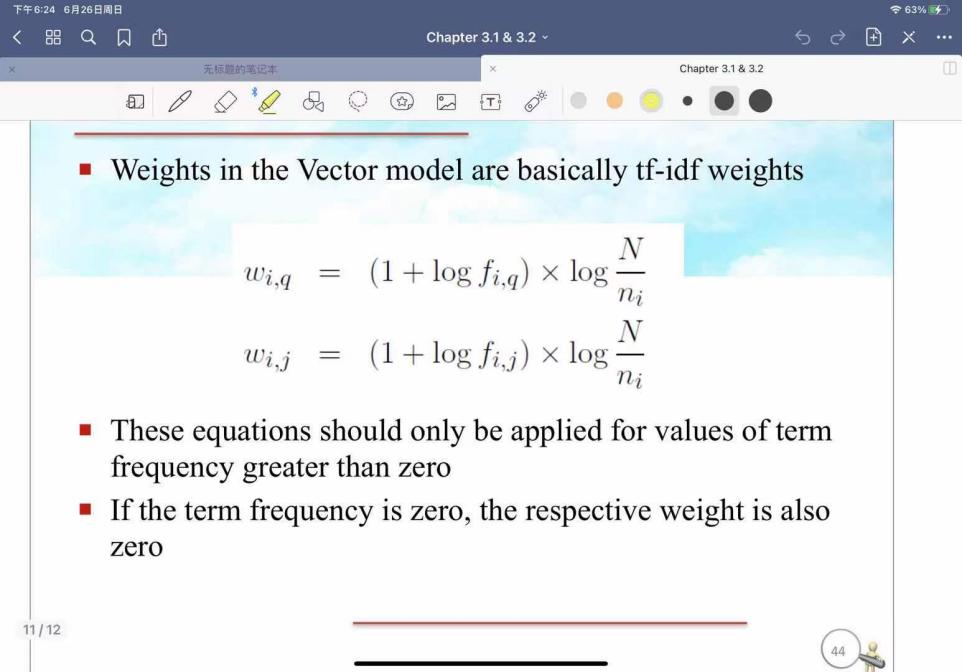


figure 8 the vector model [1]

权重的计算应该是其tf-idf权重，即

 （2）[1]

在初期wi,q与wi,j的选择上，简化了计算，使用了下图所示数据计算dj与q。

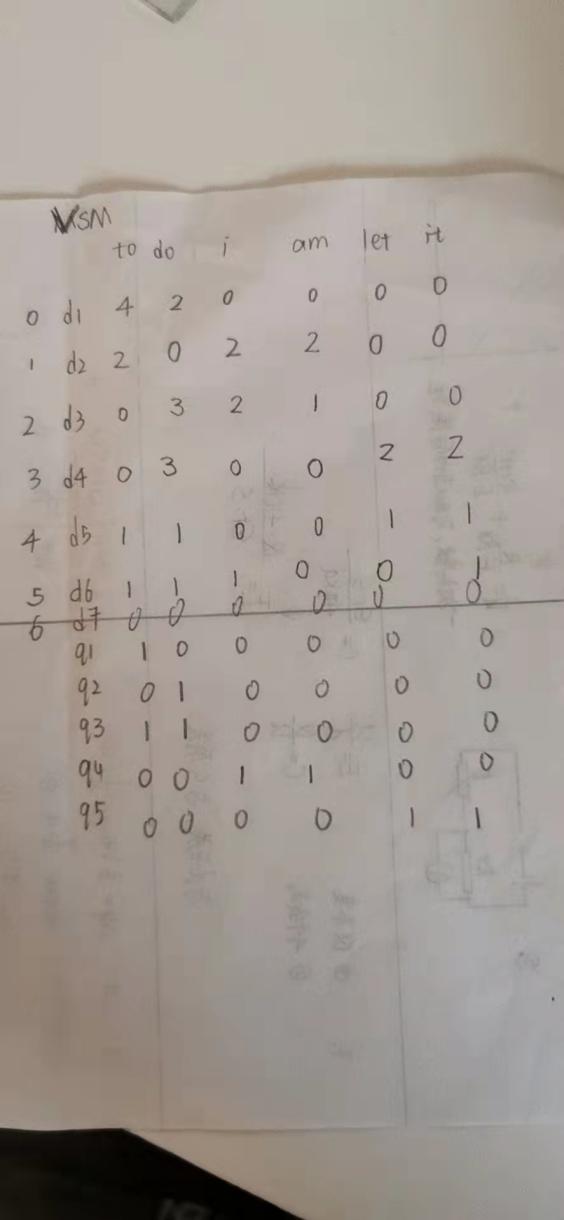


figure 9 VSM d,q

在此过程中我使用了cosine判断其相似性，即

（3） [2]

具体实现代码如下图所示。在结果排序方面，结果将按照sim的数值，由大到小，即由最接近到最不接近的方式展示出来。



figure 10 cosine计算代码块

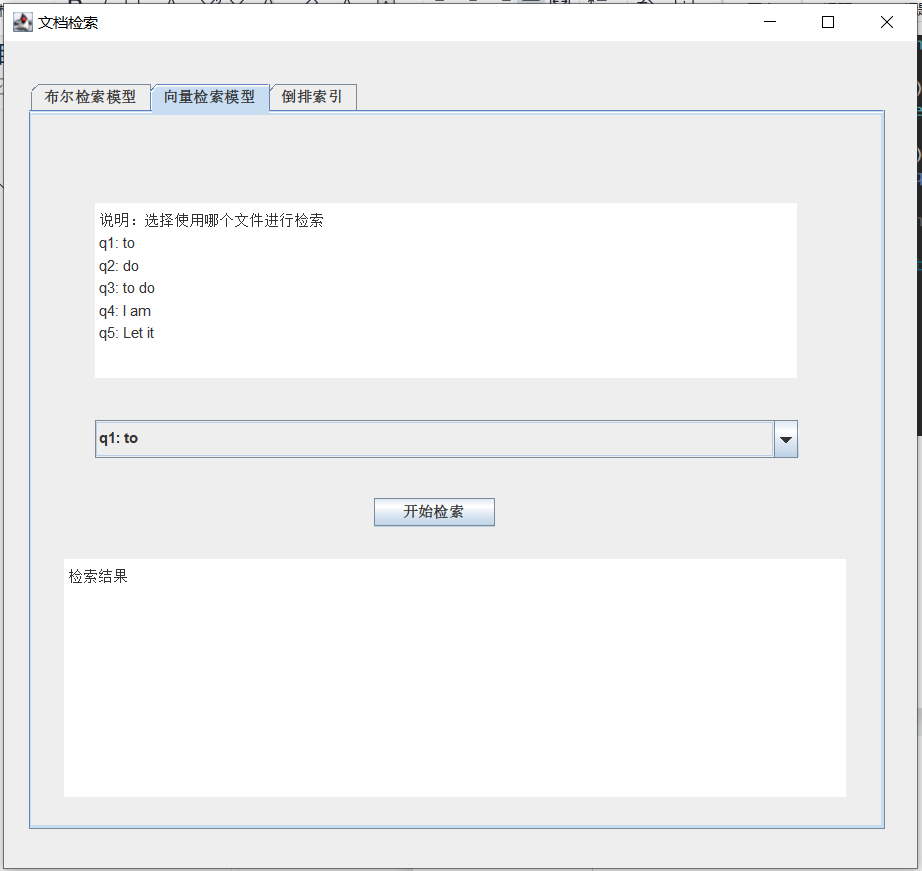


figure 11 向量检索模型-检索页面

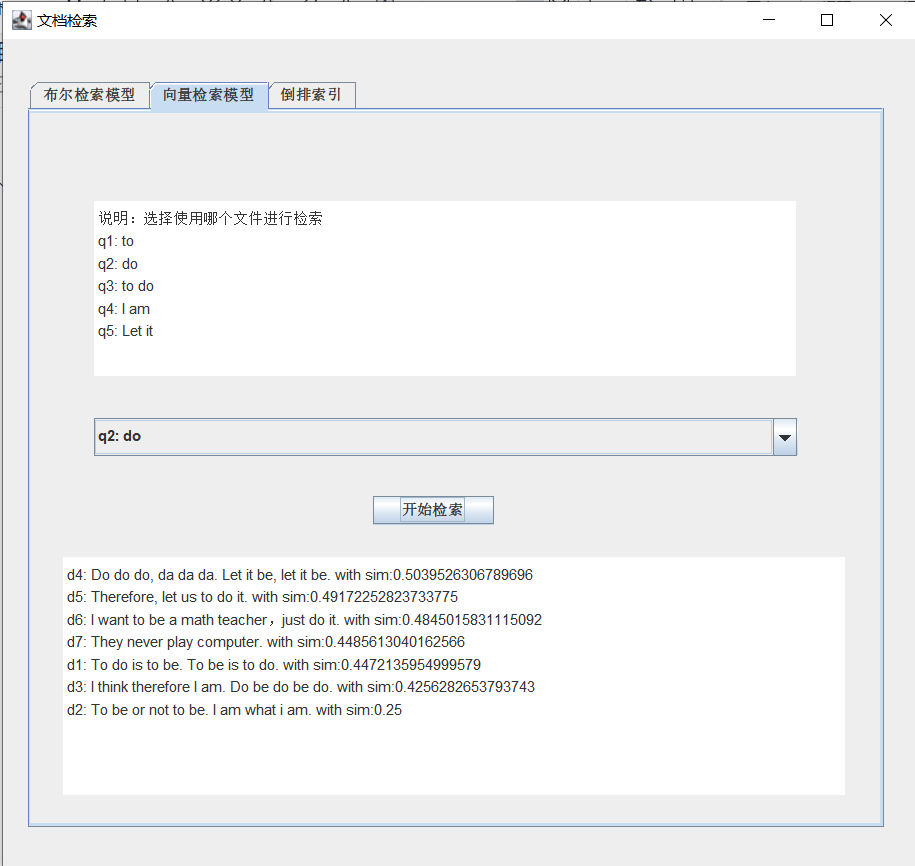


figure 12 向量检索模型-检索结果示例页面

在实验过程中，发现对于有一些文档中并未出现的搜索词，其cosine相似度也不一定为0，也会被作为结果显示在结果框内。

**2.3 倒排索引**

先找到用户要搜索的词条，根据词条得到保护词条的文档的id，然后根据id获取文档。是根据词条找文档的过程。

创建倒排索引：①将每一个文档的数据利用算法分词，得到一个个词条②创建表，每行数据包括词条、词条所在文档id、位置等信息③因为词条唯一性，可以给词条创建索引，例如hash表结构索引。

下图为to的倒排索引表构建：

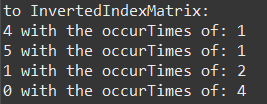


figure 13 示例to的倒排索引表

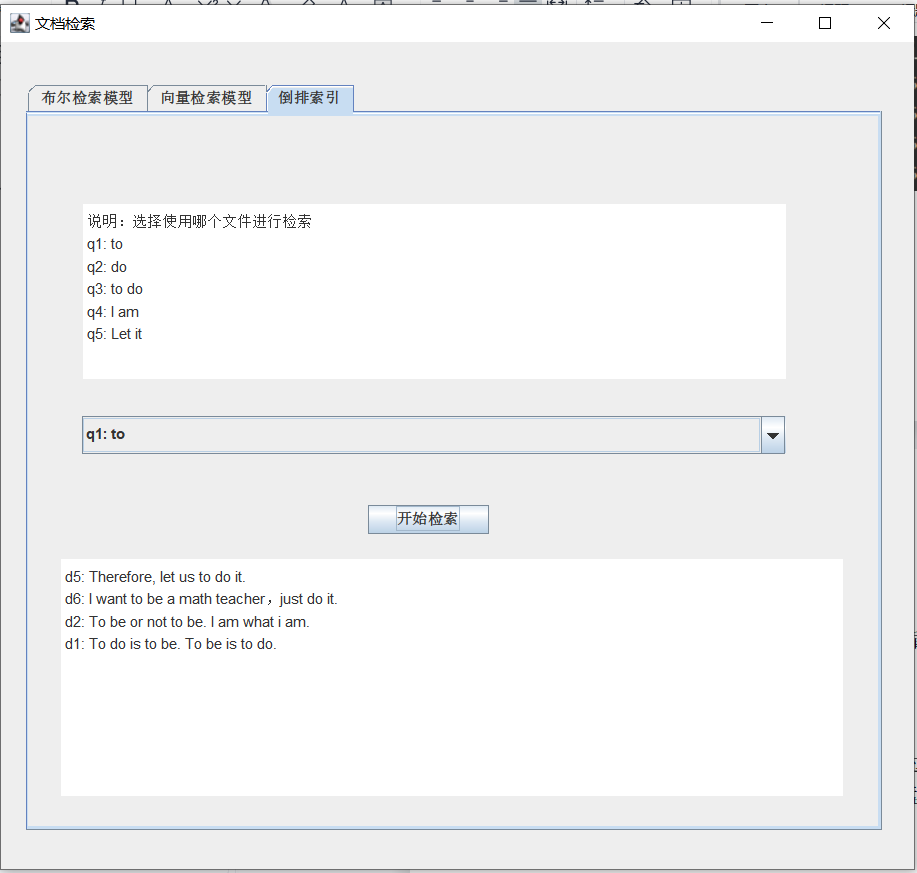


figure 14 倒排索引检索结果示例

由于我的每个词条的倒排索引表中，按照频率又低到高排列，其结果的相似性也是由低到高排列的。

倒排索引的结果与布尔模型结果显示相似，遵循“有就是有，无就是无。”。

**3.总结**

**3.1 布尔模型**

优点：

由于查询简单，便于理解。并且通过使用复杂的布尔表达式，可以很方便的控制查询结果。因此布尔模型是最为常用的检索模型。并且通过某种训练，用户可以很容易的写出布尔查询q。

缺点:

布尔模型没有考虑索引词和文档中词的权重；

布尔模型是命中模型，不涉及对召回结果的排序（结果无序）；

布尔模型是完全匹配，不支持部分匹配，而完全匹配会导致太多或太少的结果被召回；

原则上布尔模型会返回所有匹配的文档，因此很难控制检索的文档的数量；

当用户认为检索结果是相关或者不相关的时候，很难对于布尔式进行修改，很难进行自动的反馈[3]。

**3.2向量模型**

优点：

术语权重的引入使得结果的质量更高；

部分匹配允许检索近似查询条件的文档,更接近用户的检索需求；

余弦排序公式根据与查询的相似度对文档进行排序（结果有序）；

文档长度规范化自然内置于排名中；

缺点：

它假设索引项独立；

用户想查询的是一个跟query相关度大的文档，并不一定是这个文档中出现了查询词。换句话说就是即便这个文档中出现了查询词，并不一定相关性就大;

各个词之间被认为是相互独立的，随着Web中的信息量的增加，会产生很多噪声信息，影响计算结果。

**3.3 倒排索引**

优点：

不需要锁，提升并发能力，避免锁的问题；  
根据词条搜索、模糊搜索时，速度非常快

缺点：

只能给词条创建索引，而不是字段；

无法根据字段做排序[4]。

4.代码备注

GUI：DocumentQuery.java

布尔模型：SearchByBoolean.java

向量模型：SearchByVectorModel.java

倒排索引：InvertedIndex.java, occurance.java

**5.参考文献**

1. 2022物联网信息技术处理课件ch3.1&3.2
2. [信息检索—布尔模型(Boolean )和向量空间模型(VSM)\_凡眼观世界的博客-CSDN博客\_布尔模型和向量空间模型](https://blog.csdn.net/havorld/article/details/104900737?spm=1001.2101.3001.6661.1&utm_medium=distribute.pc_relevant_t0.none-task-blog-2~default~CTRLIST~default-1-104900737-blog-118386408.pc_relevant_multi_platform_whitelistv1&depth_1-utm_source=distribute.pc_relevant_t0.none-task-blog-2~default~CTRLIST~default-1-104900737-blog-118386408.pc_relevant_multi_platform_whitelistv1&utm_relevant_index=1)
3. https://blog.csdn.net/hei653779919/article/details/105969744
4. [倒排索引是什么？\_Code攻城狮的博客-CSDN博客\_倒排索引是什么](https://blog.csdn.net/helpluozhao123/article/details/123985694?ops_request_misc=%7B%22request%5Fid%22%3A%22165623523916782184622032%22%2C%22scm%22%3A%2220140713.130102334.pc%5Fall.%22%7D&request_id=165623523916782184622032&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~first_rank_ecpm_v1~pc_rank_34-7-123985694-null-null.142^v24^pc_rank_34,157^v15^new_3&utm_term=%E5%80%92%E6%8E%92%E7%B4%A2%E5%BC%95%E7%9A%84%E4%BC%98%E7%BC%BA%E7%82%B9&spm=1018.2226.3001.4187)