基础工具与作业 1

陈雨农

更新：*September 29, 2020*

1 Python 介绍

“人生苦短，我用 Python。”Python 是一门十分简洁的语言，得益于他活跃的开发社区和完善的生态，Python 有许多开源的包，可以帮助我们用简单的代码完成复杂的任务。而且 Python 的语法也很简单，网上有很多教程和公开课可供学习。为了更好地了解 Python 的运行和之后介绍的软件安装与使用，需要先了解以下概念。

1.1 解释性语言

我们常见的 C++ 是一种编译性语言，它通过编译器的编译汇编得到最后的机器语言，以后的每次运行都不再需要重新编译，而是直接运行机器语言代码，效率比较高。而 Python 与此不同，它是一种解释性语言，程序不需要编译，而是在运行时通过解释器逐句地翻译成机器语言运行，跨平台性好。

1.2 环境与包

Python 的包与 C++ 中的类似，它封装了许多变量与函数，通过关键字 import 进行调用。某个版本的 Python、Python 解释器和各种包等构成了一个环境，Python 程序需要在某个环境中运行，而程序中用到的包也应提前安装在运行的环境中。

2 软件安装与使用

2.1 Anaconda

Anaconda 指的是一个开源的 Python 发行版本，其包含了 Python、Python 的一些科学包及其依赖项，还有一些 Python 的 IDE 如 jupyter、spider，类似于 Python 界的“360 全家桶”，帮助你进行 Python 的包管理与开发 (**图 1**)。可在官网进行下载，一键安装。

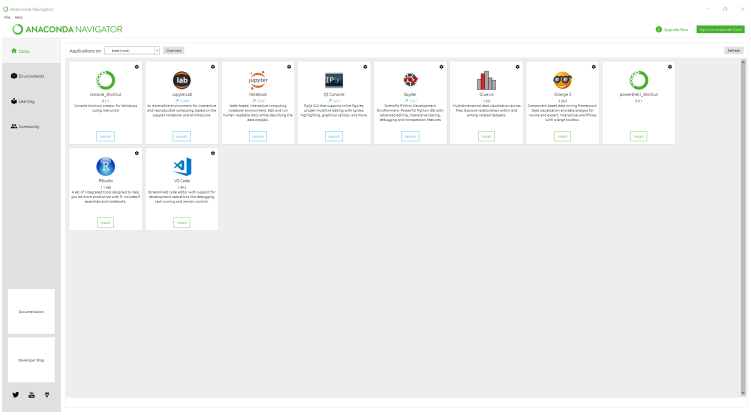

图 1: Anaconda 界面

2.1.1 环境管理

不同的程序可能需要在不同的环境下运行，Anaconda 可以让我们方便地进行环境管理。(**图 2**)

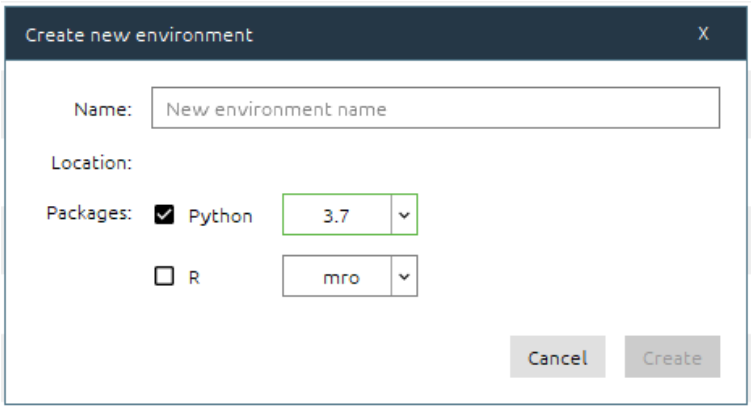

图 2: 使用 Anaconda 创建新环境

2.1.2 包管理

在不同的环境下可能会需要安装不同的包，Anaconda 提供可视化界面进行包及其依赖项的查询和安装，同时在命令行中使用 conda 或 pip 也能够进行包的安装。(**图 3**)

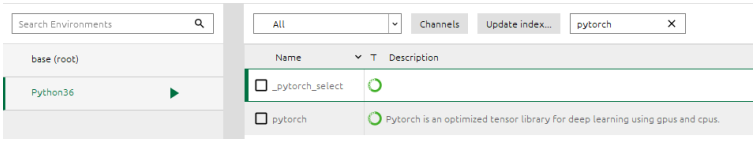

图 3: 使用 Anaconda 管理包

2.2 Jupyter Notebook

Anaconda 全家桶自动帮我们安装好了 Jupyter，它的本质是一个 Web 应用程序，能让用户将说明文本、数学方程、代码和可视化内容全部组合到一个易于共享的文档中。(图 4)

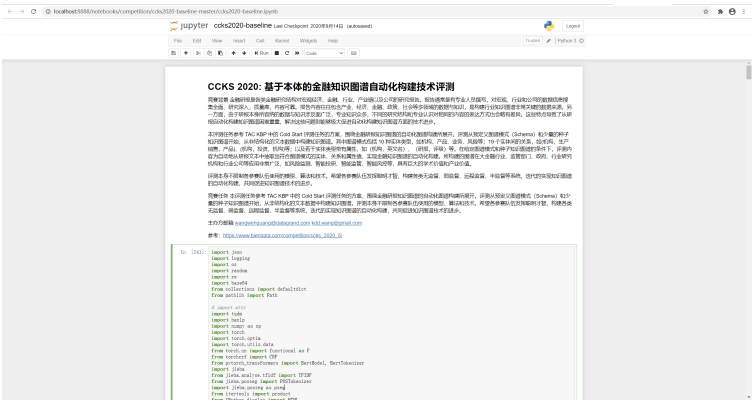

图 4: jupyter 界面

2.3 PyCharm

如果觉得 Anaconda 为我们装的 Jupyter 和 Spyder 写代码不方便或者界面不美观，可以在jetbrain官网下载使用 PyCharm(图 5)，其中社区版可以免费使用，专业版需要破解。

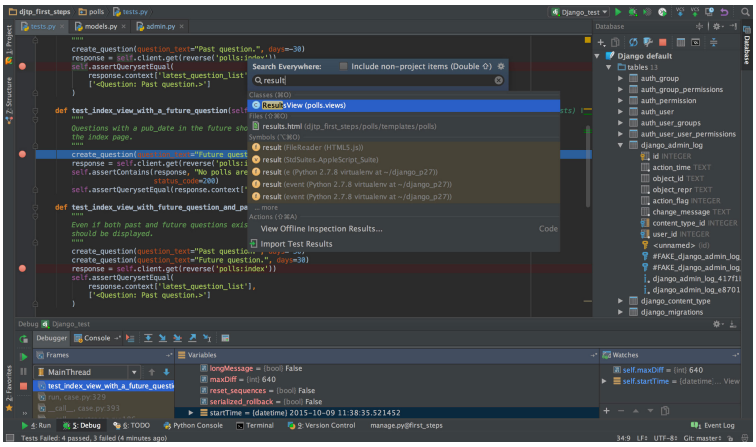

图 5: pycharm 界面

3 实现布尔检索

本次作业要求实现布尔检索模型，其大致可以分为索引构建、布尔表达式解析，结果查询合并三个部分，下面会对各个部分的思路进行讲解并给出框架代码，因本人水平有限，希望大家能批判性地阅读代码，鼓励自己的创新和改进。

3.1 索引构建

本次作业对英文文本进行检索，需要用到自然语言处理工具包 NLTK，这个包 Anaconda 已经为我们安装好了，但这次作业用到的标点符号和停用词的数据需要自行下载，可以通过运行以下代码下载：

```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
```

如果因网络问题下载失败，可从网盘(提取码 q3k1) 下载并将两个文件分别放在对应的新建目录下 (图 6图 7)。

图 6: stopwords 文件路径

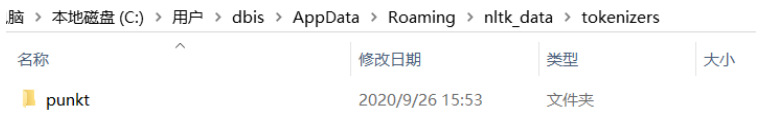

图 7: punct 文件路径

在索引构建模块中有两个重要的变量：

```
files = []  # 存储文档名称的列表，可用文档名位置下标作为ID
index = defaultdict(list)  # 键为单词，值为文档列表的字典，如{"word": [1, 2, 9],
    ...}
```

假设我们所有的文档存储在同一个文件夹下，我们需要遍历获取该文件夹下所有符合要求的文件加入 *files* 中。对 *files* 中的每一个文档，我们用 NLTK 对其内容进行分词，在 *index* 对应词的文档列表中添加文档 ID，构建好倒排索引。

尽管索引构建应该是检索系统的重点所在，但 Python 让我们走了很多捷径，如使用 NLTK 进行分词，使用字典类型变量存储索引而不用对单词按字典序排列等，并且检索效率还不一定低，这也体现了 Python 的强大之处。

3.2 布尔表达式解析

该部分的实现参考了南京大学计算机系统基础PA 实验指导, 现将基本思想总结如下。

3.2.1 词法分析

词法分析就是识别出表达式中的单元 token，在这次作业中主要有六类 token: '&&', '||', '!', '(', ')', WORD 。比如说对于查询"boolean && retrieval"，我们就需要识别出boolean, &&, retrieval，并记录他们的 token 类型。具体的实现用到了 Python 的正则表达式。

3.2.2 递归求值

根据表达式的归纳定义特性, 我们可以很方便地使用递归来进行求值. 首先我们给出布尔查询表达式的归纳定义:

```
<expr> ::= <number>     # 单独一个词是布尔查询表达式
  | "(" <expr> ")"      # 在表达式两边加个括号也是表达式
  | <expr> "||" <expr>  # 两个表达式相与也是表达式
  | <expr> "&&" <expr>  # 接下来你全懂了
  | ! <expr>
```

根据以上定义，我们可以采用分治法的解决方案，可以用递归来实现，框架如下：

```
def evaluate(p, q):
    if p > q:
        # 解析错误
        pass
    elif p == q:
        # 单个token, 一定为查询词
        pass
    elif check_parentheses(p, q):
        # 表达式被括号包围, 直接去掉外层括号
        return evaluate(p + 1, q - 1)
    else:
        op = find_operator(p, q)
        # files1为运算符左边得到的结果, files2为右边
        files1 = evaluate(p, op - 1)
        files2 = evaluate(op + 1, q)
        # 根据运算符类型递归求解
        pass
```

其中 *check_parentheses*() 函数用于判断表达式是否被一对匹配的括号包围着, 同时检查表达式的左右括号是否匹配, 如果不匹配, 这个表达式肯定是不符合语法的。

上面框架中最后一个 else 所做的就是对一个最左边和最右边不同时是括号的长表达式进行分裂，我们定义" 主运算符" 为表达式人工求值时, 最后一步进行运行的运算符, 它指示了表达式的类型，要正确地对一个长表达式进行分裂, 就是要找到它的主运算符。寻找运算符时有一些要点：

1. 非运算符的 token 不是主运算符.

2. 出现在一对括号中的 token 不是主运算符. 注意到这里不会出现有括号包围整个表达式的情况, 因为这种情况已经在 *check_parentheses*() 相应的 if 块中被处理了.
3. 主运算符的优先级在表达式中是最低的. 这是因为主运算符是最后一步才进行的运算符.
4. 当有多个运算符的优先级都是最低时, 根据结合性, 最后被结合的运算符才是主运算符。但是如果你想优化布尔表达式，则需要综合各个子表达式的结果大小进行考虑。

要找出主运算符, 只需要将 token 表达式全部扫描一遍, 就可以按照上述方法唯一确定主运算符。

3.3 结果查询与合并

我们已经对文档中存在的单词构建好了倒排索引，在单个词汇的查询时只需要将其取出即可。对于与或非操作，在 Python 中实际上可以将文档列表转换为集合，通过集合的与或非运算来完成。但为了让大家更好地理解和练习算法，要求按照课堂上讲的算法遍历列表来进行合并。(图 8图 9)

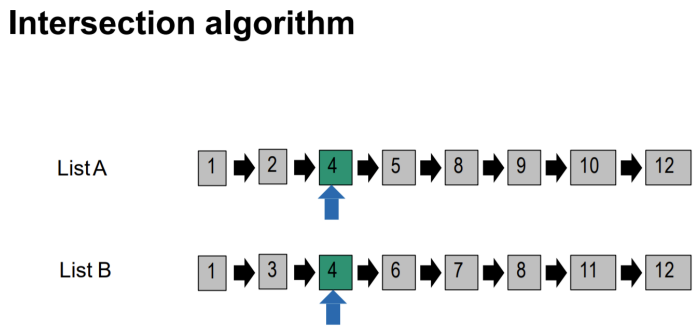

图 8: 两个列表的 intersection 操作

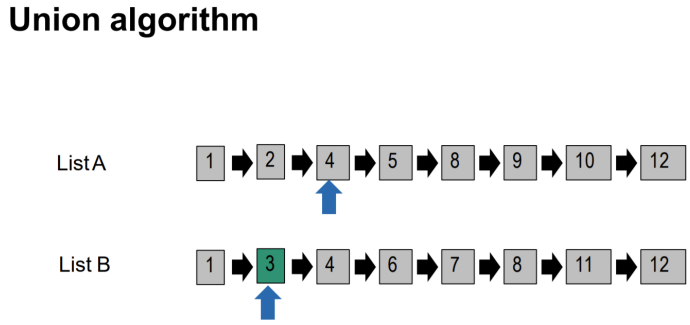

图 9: 两个列表的 union 操作

3.4 作业要求

本次作业已给出大部分代码,只需要同学们在此基础上实现布尔表达式解析中 *check_parentheses*() 和 *find_operator*() 两个函数，以及文档列表的与或非操作。

如果要对布尔表达式的与或非顺序进行优化，需要在 *find_operator*() 函数中不考虑运算符的结合性，而是比较被优先级相同的运算符切割出来的子表达式结果的大小，来确定先进行哪一步的运算。该部分不做强制要求。

助教的水平有限，所以也鼓励有能力的同学不被代码本身所局限，构建自己的框架或者改进代码的结构与细节。最后需要大家提交包括代码、运行结果、说明文本的 jupyter 文件（命名“学号 _ 姓名 _hw1”）至yunongchen@foxmail.com。