Web 关键词信息检索系统

多媒体信息检索期末项目 51255901138 舒翔

1 实验方案概述

本次期末项目实现了一种 Web 关键词信息检索系统, 输入要搜索的关键词, 通过调用网络搜索 API, 进而实现对内容简介、相关问题链接以及图片信息的获取和展示, 系统的初始化界面如图 (1) 所示.

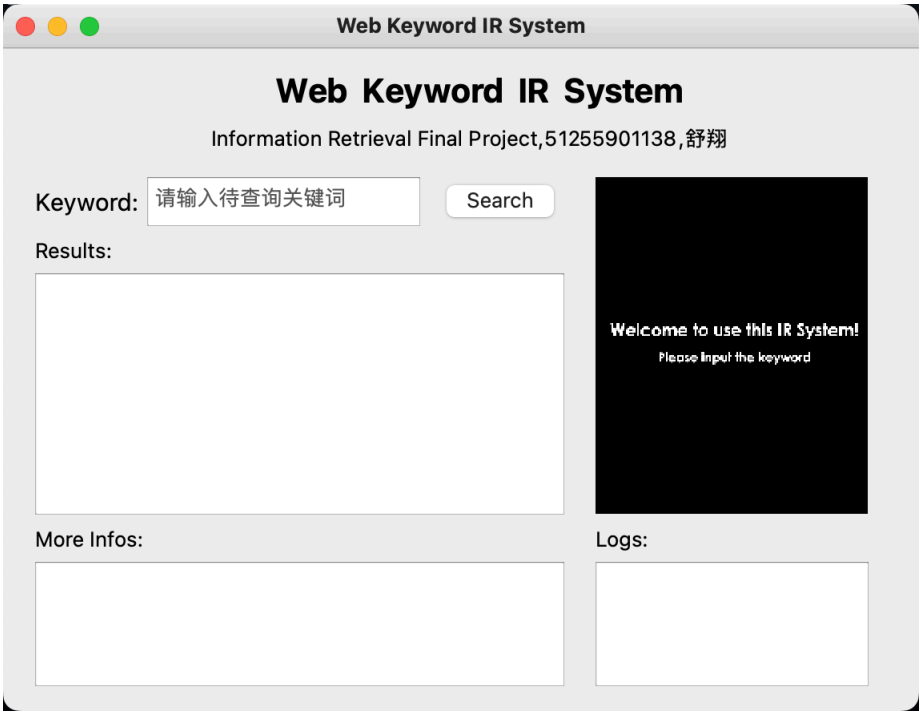


图 1: Web 关键词信息检索系统初始化界面

本系统使用了百度[1] 搜索引擎提供的 API 接口, 通过 Python[2] 爬虫对有关信息进行爬取, 使用 Qt Designer[3] 和 PyQt5[4] [5] 库对搜索界面进行设计实现, 进而实现整个系统.

在附录中, 提供了本次期末项目的源代码和实验环境的配置方法, 按照说明安装对应版本的 Python 库, 运行源代码, 即可复现本项目的全部内容. 源代码也已经放在 `https://github.com/Hsiang-1/IR_hw` .

[1] http://www.baidu.com

[2] https://www.python.org

[3] https://build-system.fman.io/qt-designer-download

[4] https://www.qt.io

[5] https://contribute.qt-project.org

2 具体实现细节

具体的实现可以分成两个部分: **窗口界面的设计**和 **检索功能的实现**.

在窗口界面设计部分, 首先是有 Qt Designer 设计出窗口的大概示意和相对位置, 然后对导出的 Python 代码进行精细修改, 撰写各个部分的按键接口. 整个窗口界面的设计示意图如图 (2) 所示.

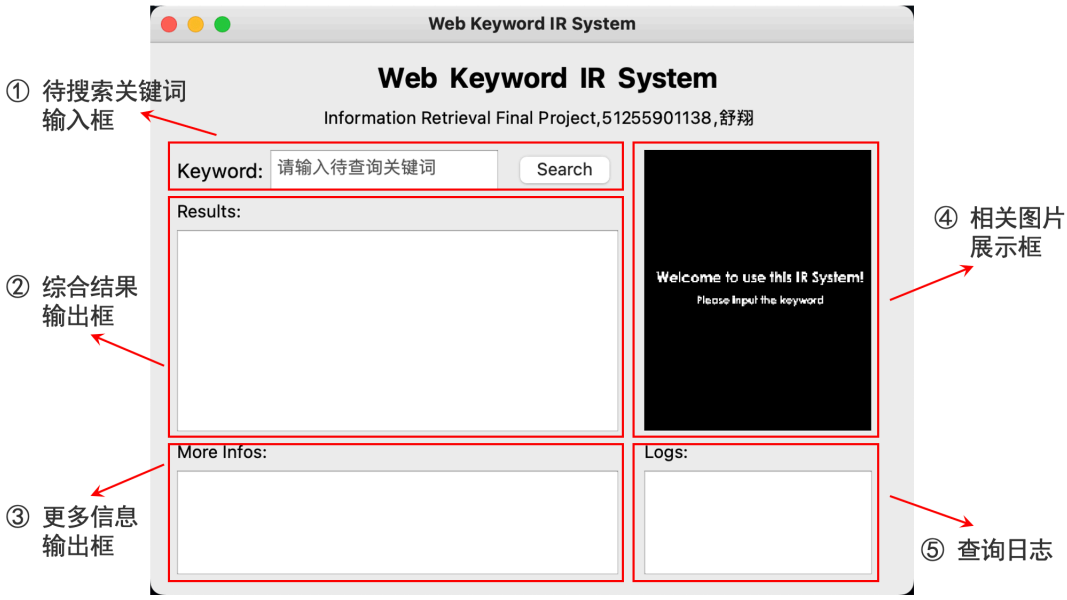


图 2: Web 关键词信息检索系统各模块示意图

从图 (2) 可以看出, 界面可以被拆分成五个主要的模块, 分别通过文本框、按钮、标签等多种控件实现. 在模块 1, 用户需要在文本框内正确输入要查询的关键词信息, 输入之后点击“search”按钮, 即可触发信息检索功能, 并在其他 4 个模块展示对应的检索结果. 在模块 2 (综合结果输出框) 中, 会展示关键词检索的简介信息; 在模块 3 (更多信息输出框) 中, 会展示关键词的有关问题和话题, 并提供相应的可点击的链接; 在模块 4 (相关图片展示框) 中, 会展示与该关键词相关的一张图片; 模块 5 为本次使用的查询日志, 会记录查询时间和历史记录.

下面简介检索功能的具体实现方法.

检索功能主要基于 Python 的 requests[6] 库和 beautifulsoup4[7] 库实现, 并调用 baiduspider[8] 库提供的方便使用的百度搜索接口进行实现.

在模块 2 中, 主要使用了百科检索接口 `BaiduSpider().search_baike()` 进行实现. 通过爬取并输出最相关的一个百科的概述资料, 进而实现对关键词简介信息的输出. 同时, 由于界面大小的限制, 往往无法完全展示完整信息, 此时还会在有关信息下提供“详细信息”超链接, 供用户点击跳转到完整信息的界面.

[6]https://github.com/requests

[7]https://beautifulsoup.readthedocs.io/zh_CN/v4.4.0/

[8]https://baiduspider.github.io/index.html

在模块 3 中, 主要使用百度问题检索列表 (百度知道) 的接口 `BaiduSpider().search_zhidao()` 进行实现. 通过调用该接口, 展示与关键词最相关的几个话题或问题及其链接, 供用户选择. 用户也可以通过这些链接跳转访问到对应的网页界面.

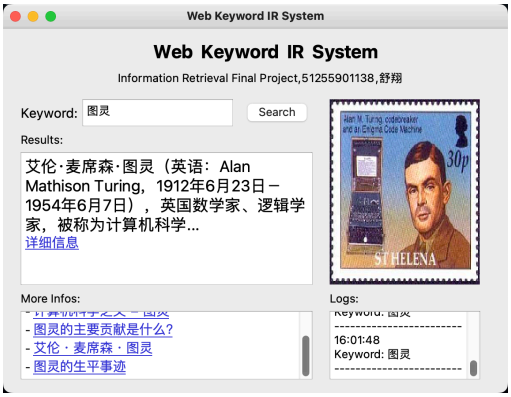
在模块 4 中, 主要使用图片检索接口 `BaiduSpider().search_pic()` 进行实现. 在图片展示的部分, 通过随机展示前 5 页搜索结果中的某一个结果, 实现“同一个查询、不同的查询图片结果”的效果, 丰富了用户信息检索的体验.

在模块 5 中, 主要通过调用系统时间和用户输入, 对用户查询的历史记录进行保留和输出. 既方便了用户翻阅查找历史, 又能方便开发人员调试程序.

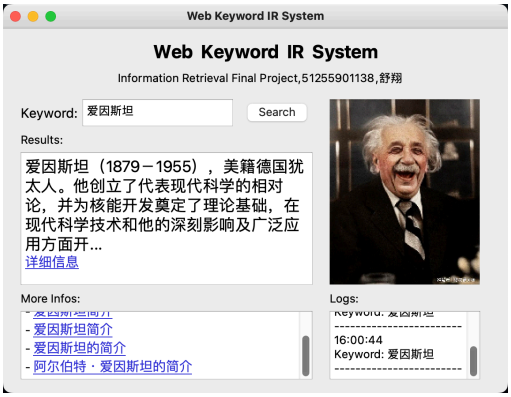
3 实验结果

在本章中, 结合对检索系统程序的设计, 进行了一系列关键词检索任务.

首先, 尝试对人名进行检索. 实验对“图灵”、“爱因斯坦”、“牛顿”、“肖邦”进行了检索, 检索结果如图 (3) 所示.

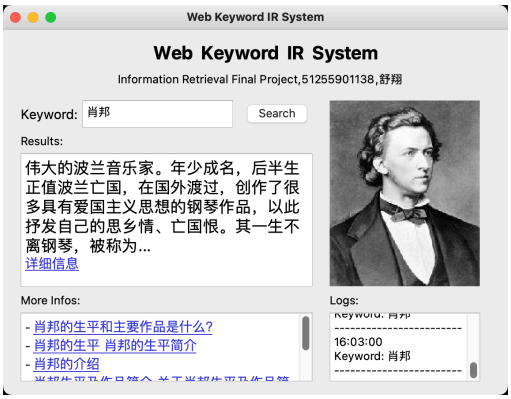


(a) 检索关键词: 图灵



(b) 检索关键词: 爱因斯坦

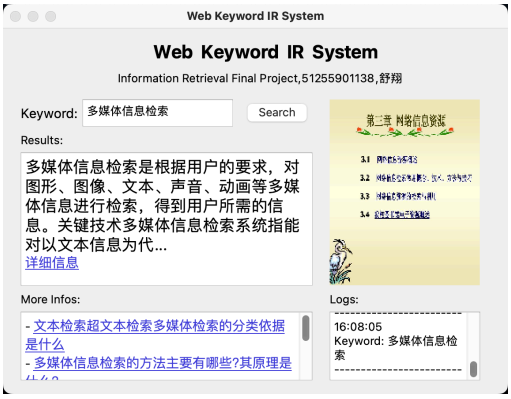
(c) 检索关键词: 牛顿



(d) 检索关键词: 肖邦

图 3: 人名检索结果

同时, 还尝试对更复杂、更多样的关键词进行检索, 如“多媒体信息检索”、“人工智能”、“中国”、“长江”等关键词进行检索, 系统均能较好地完成检索任务, 检索结果如图 (4) 所示.



(a) 检索关键词: 多媒体信息检索

(b) 检索关键词: 人工智能

(c) 检索关键词: 中国

(d) 检索关键词: 长江

图 4: 其他检索结果

4 总结

本次期末项目实现了一种 Web 关键字检索系统, 通过调用百度信息检索 API, 设计 Qt 界面, 实现了对关键词的概要信息、相关话题、相关图片的信息检索.

本系统设计了友好的用户界面, 能够同时展示多种信息 (文本、网页链接、图片), 能够在不关闭窗口的情况下连续完成多次检索任务, 并存储多次检索的历史记录, 较好地实现了任务需求.

相关代码和允许环境配置见附录; 更多实验结果截图见附件.

附录：实现代码

```
import sys, time
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import *
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from pprint import pprint
from baiduspider import BaiduSpider
import requests
import random


class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(572, 413)
        self.label = QtWidgets.QLabel(Form)
        self.label.setGeometry(QtCore.QRect(170, 10, 261, 31))
        font = QtGui.QFont()
        font.setPointSize(21)
        font.setBold(True)
        font.setWeight(75)
        self.label.setFont(font)
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(Form)
        self.label_2.setGeometry(QtCore.QRect(130, 40, 331, 31))
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(Form)
        self.label_3.setGeometry(QtCore.QRect(370, 80, 171, 211))

        img = QImage('./images/init.png')
        size = QSize(170, 210)
        pixImg = QPixmap.fromImage(img.scaled(size, Qt.
                                              IgnoreAspectRatio))
        self.label_3.setPixmap(pixImg)
        self.label_3.setText("")
        self.label_3.setObjectName("label_3")

        self.textEdit = QtWidgets.QTextEdit(Form)
        self.textEdit.setGeometry(QtCore.QRect(90, 80, 171, 31))
        self.textEdit.setObjectName("textEdit")
        # self.textEdit.setText("请输入待查询关键词")
        s = '<b style="color:#575757;font-size:13px">{}</b>'.format("
                                              请输入待查询关键词")
        self.textEdit.setText(s)


        self.label_4 = QtWidgets.QLabel(Form)
        self.label_4.setGeometry(QtCore.QRect(20, 290, 111, 31))
        self.label_4.setObjectName("label_4")
        self.pushButton = QtWidgets.QPushButton(Form)
```

```
        self.pushButton.setGeometry(QtCore.QRect(270, 80, 81, 31))
        self.pushButton.setObjectName("pushButton")
        self.pushButton.clicked.connect(self.search)
        self.textEdit_2 = QtWidgets.QTextEdit(Form)
        self.textEdit_2.setGeometry(QtCore.QRect(20, 140, 331, 151))
        self.textEdit_2.setObjectName("textEdit_2")
        self.label_5 = QtWidgets.QLabel(Form)
        self.label_5.setGeometry(QtCore.QRect(20, 110, 111, 31))
        self.label_5.setObjectName("label_5")
        self.textEdit_3 = QtWidgets.QTextEdit(Form)
        self.textEdit_3.setGeometry(QtCore.QRect(20, 320, 331, 78))
        self.textEdit_3.setObjectName("textEdit_3")
        self.textEdit_4 = QtWidgets.QTextEdit(Form)
        self.textEdit_4.setGeometry(QtCore.QRect(370, 320, 171, 78))
        self.textEdit_4.setObjectName("textEdit_4")
        self.label_6 = QtWidgets.QLabel(Form)
        self.label_6.setGeometry(QtCore.QRect(370, 290, 111, 31))
        self.label_6.setObjectName("label_6")
        self.label_7 = QtWidgets.QLabel(Form)
        self.label_7.setGeometry(QtCore.QRect(20, 80, 81, 31))
        font = QtGui.QFont()
        font.setPointSize(15)
        self.label_7.setFont(font)
        self.label_7.setObjectName("label_7")

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Web Keyword IR System
                                        "))
        self.label.setText(_translate("Form", "Web  Keyword  IR
                                        System"))
        self.label_2.setText(_translate("Form", "Information
                                        Retrieval Final Project,
                                        51255901138, 舒翔"))
        self.label_4.setText(_translate("Form", "More Infos:"))
        self.pushButton.setText(_translate("Form", "Search"))
        self.label_5.setText(_translate("Form", "Results: "))
        self.label_6.setText(_translate("Form", "Logs:  "))
        self.label_7.setText(_translate("Form", "Keyword: "))

    def search(self):

        # get keyword and print logs
        keyword = self.textEdit.toPlainText()
        t = time.strftime('%H:%M:%S', time.localtime(time.time()))
        self.textEdit_4.append(str(t)+"\nKeyword: {}".format(keyword)
                                        )
        self.textEdit_4.append("------------------------")

        # get baike knowledge
```

```
        self.textEdit_2.clear()
        font = QtGui.QFont()
        font.setPointSize(18)
        self.textEdit_2.setFont(font)
        result_baike = BaiduSpider().search_baike(keyword)
        self.textEdit_2.setText(str(result_baike[0].des))
        s = '<a href="{}" style="color:#3232CD;font-size:15px"><b>{}
                                        </b></a>'.format(
                                        result_baike[0].url, "详细
                                        信息")
        self.textEdit_2.append(s)

        # get a random picture
        result_pic = BaiduSpider().search_pic(keyword, pn=random.
                                        randint(1, 5))
        pic_link = result_pic.results[random.randint(0, 10)].url
        res2 = requests.request(url=pic_link, method='get')
        print(res2.cookies)
        content=res2.content
        with open('./images/f.jpg','wb') as f:
            f.write(content)
        img = QImage('./images/f.jpg')
        size = QSize(170, 210)
        pixImg = QPixmap.fromImage(img.scaled(size, Qt.
                                        IgnoreAspectRatio))
        self.label_3.setPixmap(pixImg)

        # get more informations
        self.textEdit_3.clear()
        result_more = BaiduSpider().search_zhidao(keyword).plain
        for i in range(5):
            # self.textEdit_3.append("- " + result_more[i]['title'])
            # self.textEdit_3.append(result_more[i]['url'])
            s = '<a href="{}" style="color:#3232CD;font-size:15px;
                                            font-family:Microsoft
                                            YaHei"><b>{}</b></a>'.
                                            format(result_more[i][
                                            'url'], result_more[i]
                                            ['title'])
            self.textEdit_3.append("- "+s)

if __name__=="__main__":
    QtCore.QCoreApplication.setAttribute(QtCore.Qt.
                                    AA_EnableHighDpiScaling)
    app=QtWidgets.QApplication(sys.argv)
    widget=QtWidgets.QMainWindow()
    ui=Ui_Form()
    ui.setupUi(widget)
    widget.show()

    sys.exit(app.exec_())
```

实验环境配置

```
## Information Retrieval Course Final Project

A simple Web Keyword IR System use Baidu API.
Use it by `python main.py`.
The folder `./ui` includes the orignal UI file of QtDesigner.


## Environment

Python 3.8.13 with:

Package             Version
------------------- -----------
BaiduSpider         1.0.2.6
beautifulsoup4      4.11.1
bs4                 0.0.1
certifi             2022.12.7
charset-normalizer 2.1.1
click               7.1.2
idna                3.4
pip                 22.3.1
PyQt5               5.15.4
pyqt5-plugins       5.15.4.2.2
PyQt5-Qt5           5.15.2
PyQt5-sip           12.11.0
pyqt5-tools         5.15.4.3.2
python-dotenv       0.21.0
qt5-applications    5.15.2.2.2
qt5-tools           5.15.2.1.2
requests            2.28.1
setuptools          65.5.1
soupsieve           2.3.2.post1
urllib3             1.26.13
wheel               0.38.4


That means you should do these to install:

pip install requests
pip install beautifulsoup4
pip install baiduspider
pip install PyQt5
```

Code is available on my github `https://github.com/Hsiang-1/IR_hw`.