网络实习Lab报告

组长： 刘子渊

组员：熊思亚东

张泽华

赵睿哲

# 操作与功能

本组项目主要实现关于CS方向研究生留学项目的搜索。使用Node.js提供后端数据接口与机器学习算法、使用Redis保存搜索引擎索引和数据、使用Python进行项目数据的爬取、以及使用Angular和Bootstrap框架搭建前端页面，最终实现了一个利用爬虫技术获取数据的留学项目搜索引擎系统。本项目最终爬取了美国排名较高院校的研究生留学项目共计59个。

本搜索引擎系统实现了两种搜索方式，一为朴素的传统搜索引擎文本匹配搜索方法。而是基于机器学习及文本分类技术的新型搜索引擎匹配方式。

主要使用流程如下，下图所示图片全部源自项目运行时截图。

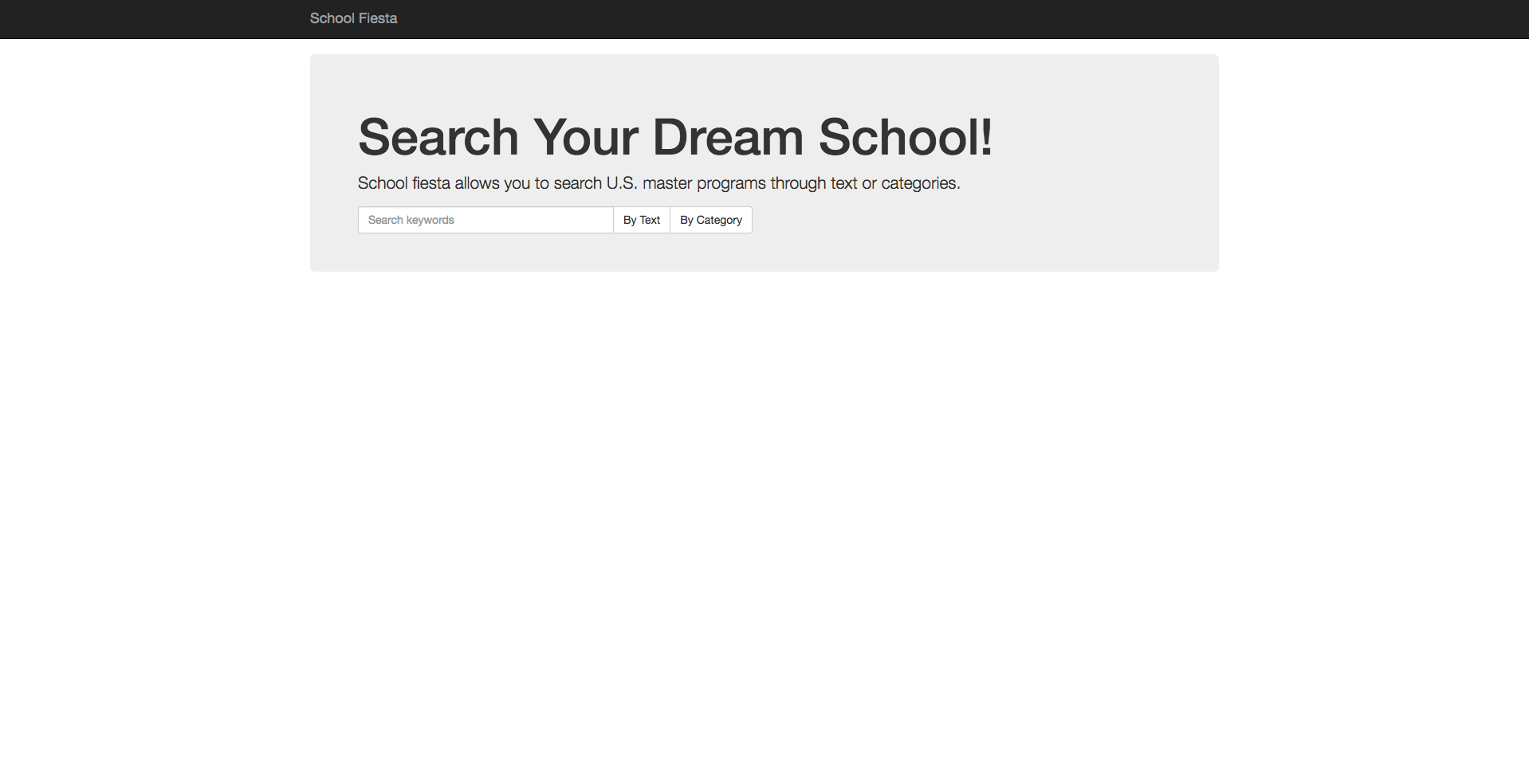


图 1：首页截图，用户可以在搜索框中输入自己的目标关键词

在进入系统尚未进行任何操作时网站界面如图1所示，主要组件为用于键入搜索关键词的文本输入框和选择搜索类型的两个按钮，分别代表本组项目中实现的两种搜索方式。

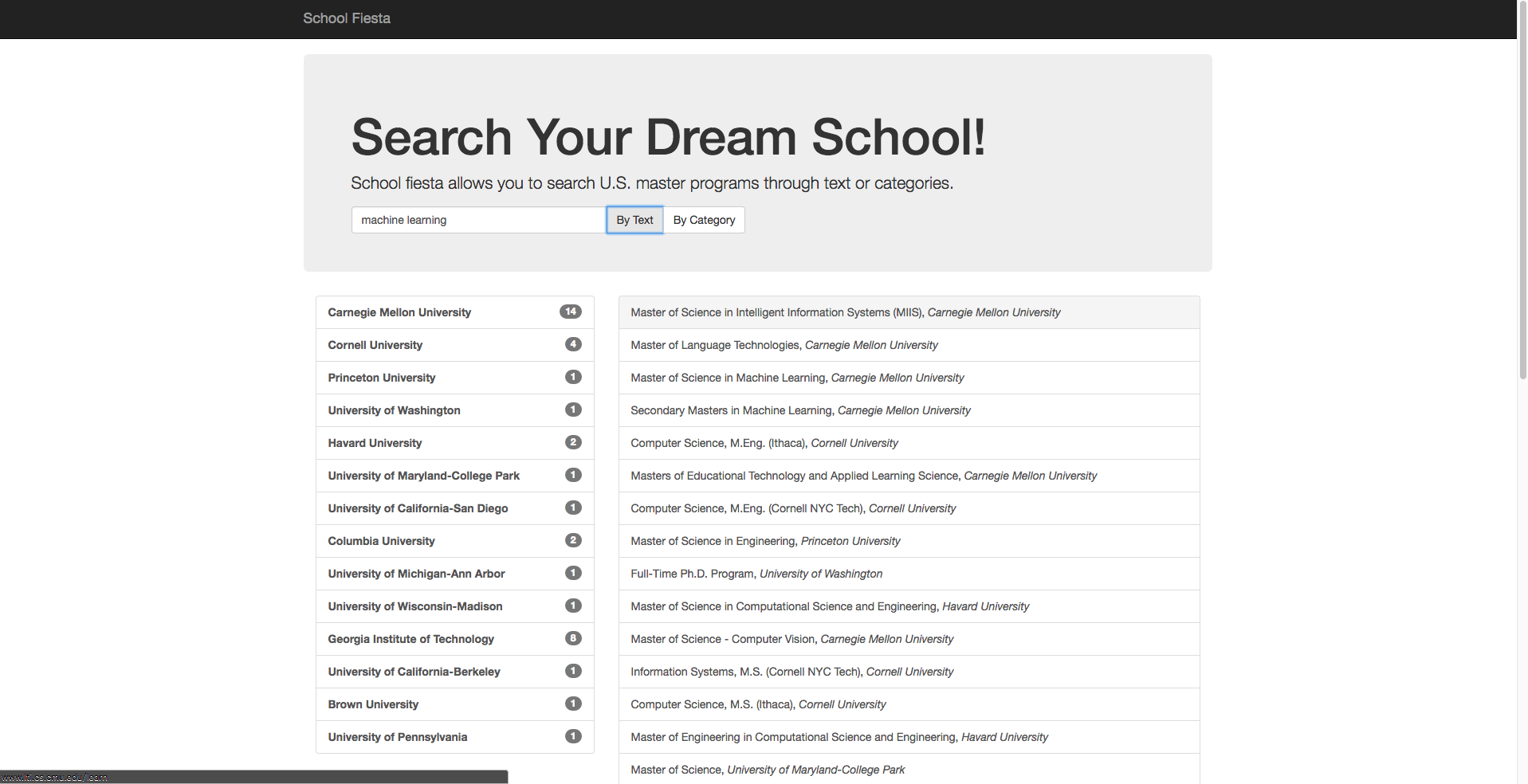


图 2：显示搜索结果，左侧边栏为结果中出现的全部学校，右侧部分为项目名称。点击左侧学校可以筛选右侧的结果列表

再输入搜索关键词之后，在本页面内会刷新出搜索得到的结果并根据院校进行了分类。可以看到左边一列展示了返回结果中所属学校的列表并表明了每个院校所包含的项目数量，便于用户进行筛选和检索。右方列表当中展示的内容则是项目的名称信息，点击后会直接跳转至对应项目主页。

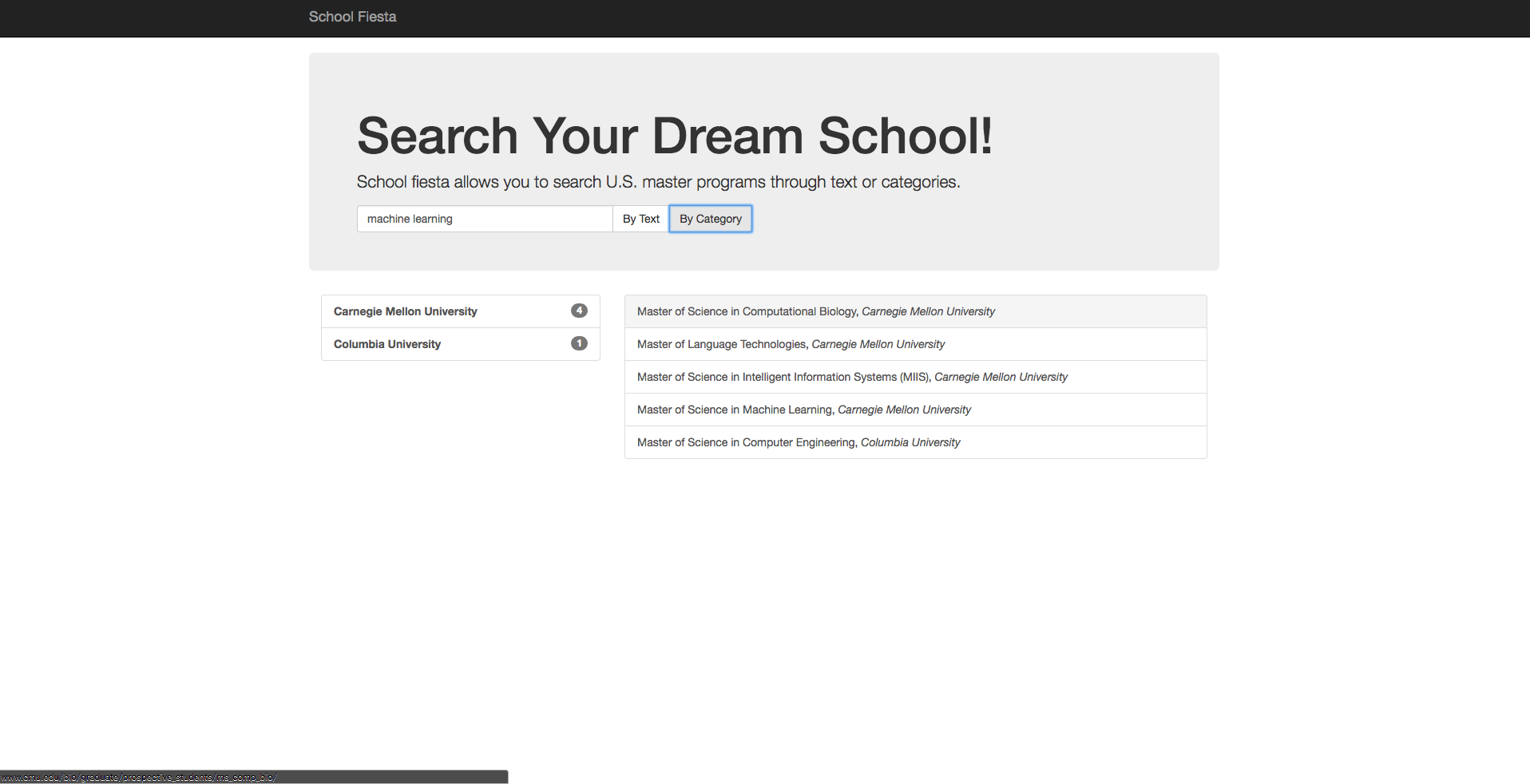


图 3：使用机器学习算法的搜索结果

点击”By Category”按钮后会使用基于机器学习的匹配技术进行检索，返回结果更加精确，展示形式上和上文所述并无不同。

# 数据集

本项目中所使用的数据集共包括来及不同学校的共59个研究生留学项目。获取信息使用的爬虫模块使用Python下beautifulsoup4开源爬虫组件实现。由于各个学校的网页路由规则并不相同，因此无法使用传统的统一规则爬虫进行实现。于是在本次项目的实现过程当中人为加入了网页列表进行启发式搜索，从预先设定的主页列表中选取起点，进行递归的链接检索。最终获取到的数据内容以Json形式进行存储以便后续使用。

爬虫相关的代码在crawler/目录下，使用script.py可以进行交互式的数据爬取。保存的JSON数据格式如下：

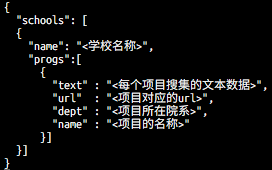


图 4：数据集的JSON格式

# 能否动态部署

本项目的代码由于使用Node.js实现后端的服务，因此非常容易进行动态的部署。本章提供两种方案，一种是基于一般的Linux服务器的部署，一种是在Heroku[[1]](#footnote-1)平台上的部署。

基于一般Linux服务器的部署需要在本机上安装好Node.js环境，最好使用NVM[[2]](#footnote-2)进行Node.js的版本管理。当Node.js环境配置好之后，可以直接在项目的根目录中执行npm install。本项目中的前端库均使用bower进行配置，在npm install之后会自动执行bower install，不需要手动安装。当安装完毕之后，直接执行npm start，随着主机环境变量PORT的取值不同，本项目的后端服务会侦听不同的端口。

使用Heroku部署更加容易，在其网站上注册一个账号并创建一个应用之后，将该应用与本项目的GitHub库进行连接，之后Heroku便可通过GitHub的webhook事件进行自动部署。由于本项目还使用了Redis，因此在Heroku上注册一个新的Redis插件即可。本项目在Heroku上的部署URL如下：<https://school-fiesta.herokuapp.com/>

# 修改意见反馈

老师对于数据集的要求将根据实际需求很具学校排名列表继续扩大搜索范围进行对数据集的补充。此外还可以考虑将其他小组的不同类型数据进行一定程度上的整合最终获得较好的实现效果。

1. 组员分工：

刘子渊：机器学习算法实现

熊思亚东：爬虫部分实现

张泽华：项目前端实现

赵睿哲：项目架构设计与实现（爬虫+前端/后端整合部署）

# 亮点

**爬虫部分**：由于各个学校的网页路由规则并不相同，因此无法使用传统的统一规则爬虫进行实现。于是在本次项目的实现过程当中人为加入了网页列表进行启发式搜索，从预先设定的主页列表中选取起点，进行递归的链接检索。尽管如此，本项目实现的爬虫对于手工检索数据也非常友好，管理人员只需要把项目的名称、院系与URL填充，后台即可自动的爬取数据并添加到数据源中。

**搜索引擎部分**：搜索引擎实现部分传统匹配方式运用了大量的自然语言处理方法对文本内容进行预处理。而运用机器学习算法的文档评分模块使用Softmax和基于特征向量提取算法实现了对目标文本的分类和内容分析，具有一定的技术和实现难度，代码量不少。

**服务端、前端实现**：整个项目架构后端使用Node.js的Express框架，结合Redis提供了数据存储。前端使用Angular与Bootstrap进行页面逻辑和界面实现。本项目的服务端架构非常方便部署于任一云服务平台，比如本项目选用的Heroku，只需要git push便可完成部署。

1. <https://heroku.com> 提供了非常简便的web应用的部署工具 [↑](#footnote-ref-1)
2. NVM是类似于RVM的Node.js版本管理工具，可以在一台机器上安装多个Node.js版本并且不互相影响。 [↑](#footnote-ref-2)