南京大学
NANJING UNIVERSITY

研 究 生 毕 业 论 文

（申请硕士专业学位）

论 文 题 目　基于 SpringBoot 与 Vue 框架的中文社科论文分析系统的设计与实现

作 者 姓 名　叶济凡

专 业 名 称　软件工程

研 究 方 向　软件工程

指 导 教 师　郑滔 教授

2020 年 5 月 15 日

学　　号：　**MF1832220**

论文答辩日期：　**2020　年　5　月　19　日**

指 导 教 师：　　（签字）

Design and Implementation of Chinese Social Science Paper Analysis System Based on SpringBoot and Vue.js Framework

By

Ye Jifan

Supervised by

Professor Zheng Tao

A Thesis
Submitted to the Software Institute
and the Graduate School
of Nanjing University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Engineering

Software Institute

May 2020

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：基于SpringBoot与Vue框架的中文社科论文分析系统的设计与实现

工程硕士（软件工程领域） 专业 2018 级硕士生姓名： 叶济凡

指导教师（姓名、职称）： 郑滔 教授

摘 要

随着社科研究知识水平的发展，社科知识也在快速的更新迭代，论文库也随之日益庞大。在日益庞大的论文库面前，当研究人员希望全面了解某一领域的相关研究并在此基础上继续深造，前期往往要耗费大量的时间精力查找相关论文，甚至错过一些重要的发展方向。因此，提高论文的检索效率，以此提升科学研究生产力是一件非常值得研究的内容。另一方面，随着大数据时代的到来，数据挖掘，机器学习算法日渐成熟，自然语言处理技术也被广泛应用于文本提取，主题提取等相关方面，为使用自动化处理大批量的论文提供了技术上的支持。通过自动化而不是通过人力对论文进行检索分析，可以节省大量的人力物力，提高生产效率。

本文针对目前国内社科论文分析研究所面临的问题进行了分析，同时结合目前对大数据文本方面处理技术的发展，提出采用Citation-LDA（Citation Latent Dirichlet Allocation）与BERT（Bidirectional Encoder Representation from Transformers）模型进行运算，同时与数据分析相结合的方法，对社科论文相关信息进行挖掘，整合与展示。Citation-LDA模型是基于引文的LDA模型，由于论文引文信息包含的信息较多且篇幅较小，使用引文信息来进行模型的运算可以大大加快运算的速度，同时降低了噪声影响。通过Citation-LDA模型，不仅能够发现论文主题，同时根据引用信息，还可以总结出主题流变以及发现主题下的里程碑论文。而BERT模型则从另一个角度来对文章主题进行挖掘。通过BERT模型，所有的论文都可以表示为一个词向量，而通过对词向量的聚类，可以得到主题相近的论文簇。通过对论文簇进行主题提取，可以得到所有的主题以及每个主题下的论文以及论文的排名。以上两种模型互相结合，能够较为准确的总结出论文与主题之间的关系。结合对论文其他相关信息的处理，最终可以向研究人员展示包括论文主题，主题流变与发展，论文作者研究领域，论文相关研究方向的里程碑论文等一系列深层次的信息，方便研究人员进行相关论文发展方向的探索。

在结构方面，项目是一个web项目，主要采用Springboot进行项目的搭建，使用Elasticsearch来作为存储引擎，方便信息的快速查找。模型使用python脚本

进行编写，模型运算结果储存在Elasticsearch以及文件系统中。前端方面则采用BootStrap框架与Vue.js框架来实现相关运算结果与数据的可视化。本文运用到的所有数据来源为南京大学数据中心以及社科类论文全文PDF文件。本人在项目中承担了数据分析中关联作者分析以及数据统计部分，分词训练，以及项目前端部分的设计与实现。

关键词： 社科论文分析，Citation-LDA，BERT模型，主题流变与发展

南京大学研究生毕业论文英文摘要首页用纸

THESIS: Design and Implementation of Chinese Social Science Paper Analysis System Based on SpringBoot and Vue.js Framework

SPECIALIZATION: Software Engineering

POSTGRADUATE: Ye Jifan

MENTOR: Professor Zheng Tao

Abstract

With the development of the social science research knowledge level, the social science knowledge is also rapidly updated and iterated, and the thesis library is becoming larger and larger. In the face of an increasingly large thesis library, when researchers want to fully understand relevant research in a certain field and continue to further their studies on this basis, they often spend a lot of time and energy to find relevant papers in the early stage, and even miss some important development directions. Therefore, improving the retrieval efficiency of papers to improve the productivity of scientific research is a very worthwhile content. On the other hand, with the advent of the era of big data, data mining and machine learning algorithms are becoming more and more mature. Natural language processing technology is also widely used in text extraction, topic extraction and other related aspects, providing technical support for the use of automation to process a large number of papers. Retrieval and analysis of papers through automation rather than manpower can save a lot of manpower and material resources and improve production efficiency.

This article analyzes the current problems faced by the domestic research institute of social science thesis analysis, and combines the current development of processing technology for big data text, using the Citation-LDA (Citation Latent Dirichlet Allocation) and BERT (Bidirectional Encoder Representation from Transformers) to perform calculations, combined with data analysis, mines, integrates and displays relevant information of social science papers.The Citation-LDA model is a citation-based LDA model. Because the citation information of the paper contains more information and is smaller in length, using the citation information to perform model calculations can

greatly speed up the calculation and reduce the impact of noise. Through the Citation-LDA model, not only the thesis topic can be found, but also the topic changes and the milestone papers under the topic can be summarized based on the citation information. The BERT model mines the topic of the article from another angle. Through the BERT model, all papers can be represented as a word vector, and by clustering the word vectors, a paper cluster with similar topics can be obtained. By subject extraction of the paper clusters, you can get all the topics and the papers and the ranking of the papers under each topic. The above two models are combined with each other, which can more accurately summarize the relationship between the paper and the topic. Combined with the processing of other relevant information of the paper, a series of deep-level information such as the subject of the paper, thematic changes and development, the research area of the author of the paper, and milestone papers related to the research direction of the paper can be shown to the researchers, which is convenient for the researchers to make relevant Exploration of the development direction of the thesis.

In terms of structure, the project is a web project. Spring-boot is mainly used to build the project. Elasticsearch is used as the storage engine to facilitate the rapid search of information. The model is written using a python script, and the model calculation results are stored in Elasticsearch and the file system. On the front-end side, the BootStrap framework and Vue.js framework are used to realize the visualization of related operation results and data. All data sources used in this article are from Nanjing University Data Center and full-text PDF file in the social science. In the project, I undertook the author analysis and data statistics part of the data analysis, word segmentation training, and the design and implementation of the front-end part of the project.

Keywords: Analysis of Social Science Papers, Citation-LDA model, BERT model, Theme Rheology and Development

目　录

表　目　录

图　目　录

第一章　引言

1.1　项目背景

当今世界，社科知识研究发展水平日益提高，社科知识也在不断更新迭代。社科论文发表量的显著上升也使得其相关论文库日益庞大 [1]。研究人员在研究初期，往往需要通过耗费大量时间精力查找相关论文以达到尽可能全面了解研究领域的目的，奠定研究基础。但此过程很容易受到干扰，乃至可能因此错过重要的发展方向和重点研究内容。在面对纷繁复杂的学科论文时，研究人员如何提高论文的检索效率，进而提升科学研究的生产力成为了做好科学研究的关键性因素。在论文的分析统计方面，已经形成了一门专业的科学，即文献计量学。

21世纪以来，随着计算机技术也随着时代的进步高速发展，在大数据时代背景下，数据挖掘，机器学习算法日渐成熟，文献计量学也在向信息化，自动化方向发展。如何通过软件，自动化地对论文进行研究分析是文献计量学的重要研究方向。随着自然语言处理技术的快速发展，对论文文本进行分析的技术已经日渐成熟，这也为自动化处理大批量论文提供了技术上的可能 [2]。如何更好地利用自动化代替人力对论文进行检索分析，节约人力物力，提高科研生产的效率是十分值得研究的内容[3]。

在自动化语言处理中，LDA（Latent Dirichlet Allocation）模型是使用的较为广泛的无监督机器学习模型。通过三层贝叶斯模型[4]，LDA模型对文本主题的提取达到一个比较令人满意的程度 [5, 6]。但同时，LDA模型对中文的处理依赖于中文语料分词的准确性，特别对于专业的社科类论文来说，很多专业名词很可能对分词结果产生影响。除了LDA以外，文本聚类也是自然语言处理的经典场景。通过将论文转化为句向量或者词向量 [7–10]，论文库中的论文可以统一的表示起来，通过对向量的聚类，可以将论文分为若干类，使得一类之中的文档相似度尽可能大 [11–14]。这些技术都为自动化论文处理提供了宝贵的思路。

以上两种技术均涉及到中文分词，分词的准确性直接关系到模型的效果[15]。目前针对中文分词，效果比较好的工具有Hanlp，jieba等。在使用这些工具的同时，还可以加入对专有名词的处理，以保证分词效果的准确性。

1.2　国内外文献计量学发展与研究现状

我国文献计量学发展可以分为三个阶段，即起步阶段，发展初期阶段与全面发展阶段。自1988年至今，我国文献计量学进入了全面发展的阶段，不仅提出了很多全新的理念，同时在科学评价与科技管理方面开展了大规模的运用，理念与运用相结合，获得了很多具有意义的结果。1987年，赵红洲等学者通过美国SCI对我国主要大学发表的论文进行了统计分析，得出了相关的论文排名，引起了强烈反响；中国科技情报研究所建立了“中国科技论文与引文数据库”，从而可以更加科学，系统，客观地对主要大学以及研究院的学术水平作出评价；同时，《文献计量学》等教材相继出版，标志着我国文献计量学研究和发展获得了很大的进步，进入了新的层次。

我国文献计量学相较于国际，仍然起步较晚。早在20世纪60年代初，美国就编制了《科学引文索引》(SCI)这一在文献计量学史上划时代的数据库，对文献的研究与发展起到了巨大的作用，很大程度上推动了文献计量学的进步。我国于1988 年开始建立“中国科技论文与引文数据库”，每年发表相应的统计分析报告；此后，中国科学院文献情报中心研制了“中国科学引文数据库”(CSCD)，南京大学社会科学研究评价中心出版了“中国社会科学引文索引”(CSSCI)，都为我国文献计量学发展起到了推动作用 [16]。

进入新世纪以来，计量学逐渐呈现出科学化，信息化，网络化，自动化的趋势，结合计算机的日益普及以及机器学习，数据挖掘等相关算法的发展，计算机辅助研究成为一股热潮。目前较为流行的有SATI，citespace等辅助分析软件。

SATI全称Statistical Analysis Toolkit for Informetrics，是一款针对期刊论文题录信息进行统计分析的专业工具 [17]。通过聚类分析，共现分析与一般统计分析等分析方法，挖掘出论文深层次的信息，并以可视化的方式展现出来，为文献的学术研究提供了专业化的统计与分析方法。SATI支持多种来源数据库导出的题录格式，包括CNKI，CSSCI，WoS等主流论文数据库，同时还支持用户自行生成SATI专有格式的题录。通过使用SATI，用户可以对文献进行深层次的研究。SATI提供包括基础统计，自然语言处理，共现分析，聚类分析等一系列分析方法。基础统计主要用于统计论文的基本信息，包括作者，机构，关键字等出现的频率和频次。自然语言处理则支持用户从题录信息中抽取除了主题词以外的其他关键词 [18]。共现分析可以生成主题词，关键词，作者等信息的共现矩阵，并且支持用户自定义矩阵大小。聚类分析将对题录信息中的作者，机构，关键词等信息进行聚类展示，并且支持多种聚类算法。SATI将对数据的分析结果通过图表的方式直观的展示给用户，并支持用户自定义可视化图形的参数。

通过以上的分析方法，SATI能够对题录信息中隐藏的信息进行挖掘，帮助研究人员节省大量的人力物力。

citespace中文名称为“引文空间”，顾名思义，是一款针对引文进行统计分析的专业化工具 [19]。citespace的数据一般来源于WoS或者中国知网（CNKI），对于非WoS数据库导出的数据，则一般需要对数据格式进行转化。citespace专注于四个方面的分析：共被引分析，共词分析，突现分析以及聚类分析。共被引分析主要针对引用进行相关的分析。所谓共被引，指的是两篇文章同时被第三篇文章所引用。共被引的次数越多，说明两篇文章之间关联性越强，研究的主题越接近 [20]。因此，通过构建共被引矩阵，可以得到文章与文章之间的相关性，通过将共被引矩阵可视化，可以得到共被引网络。共词分析则对关键词信息进行挖掘。如果两关键词在一组文档中频频共现，则可以说明这两个关键词存在着比较接近的关系 [21]。突现分析功能用于检测某一个时间段，引用量是否发生较大的起伏，根据引用量的大量增加或者减少，可以从中推断出主题的兴盛或衰落。聚类分析则是将相似的文章通过聚类算法进行聚集，从而整理出相似文章组成的文章簇。citespace支持多种聚类算法。

通过以上分析不难看出，目前针对论文文献进行分析的工具有很多，不同的工具针对的文章内容也有所不同。SATI主要针对题录信息而citespace主要针对引文信息 [22]。然而，由于专业之间差别较大，在中文论文领域，专业词汇的分词结果可能不尽如人意，造成分析结果可能有所误差。本文主要针对单一的中文社科类论文进行分析，同时因为数据量相对来说较小，可以将全文纳入到分析运算的范围之中，从而得到更为详细准确的结果。

1.3　论文的主要工作和组织结构

本文主要介绍中文社科分析系统的设计以及相关实现。本系统主要分为数据管理，模型管理，数据分析以及分词训练几大模块，本人主要负责数据分析中关联作者分析以及数据统计部分，分词训练，以及项目前端部分的设计与实现。

- 关联作者分析主要负责提供作者的所有作品，作者发表的期刊以及作者研究主题的演变等信息，同时能够直观的以年份为单位展示该作者发表的作品、在同一时间与该作者研究同一主题的学术圈等深层次信息。系统使用者还可以找到该作者发表的最有价值的论文以及成就最高的研究主题等。
- 数据统计包含了除了作者信息之外的其他一些内容的统计。系统将统计某指定文件分词结果的词频，并展示给用户分词文件的词频总览图和最高词

频关键词的作者以及机构，用户可以分析词频统计结果，如果用户对词频的统计结果不满意，用户可以反馈给研究人员，研究人员会对关键词进行调整。同时，用户可以通过点击总览图中的某个关键词，查看该关键词详细统计结果。用户还可以通过搜索的方式查看指定的关键词信息。

- 分词训练模块主要包含分词结果展示，自定义词典的添加以及重新分词等功能。由于分析结果是否准确很大程度上依赖于分词结果，因此分词模块是半人工监督的。研究人员可以在界面上观看某分词文件的分词结果，当用户对结果中某些分词结果效果不满意时，用户可以通过增加自定义分词词典来修正分词结果。在添加词典后，研究人员可以选择重新分词，观察新的分词结果，直到对分词结果满意为止。

- 系统的前端部分，主要采用了BootStrap框架与Vue.js框架，在界面风格上参考了中国社会科学评价研究中心的相关网站。同时根据大数据量等特点，采用分页，延时加载等操作，确保所有信息在3s内显示给用户。

本文的组织结构主要包括以下章节：

第一章：引言部分。主要介绍了开发中文社科分析系统的项目背景，以及该系统所服务的文献计量学的发展现状，总结目前国内研究仍然存在的不足。介绍了论文的主要工作以及本人负责的相关工作。

第二章：技术概述部分。主要介绍了本人在项目实现过程中运用到的相关技术。

第三章：需求分析与概要设计部分。主要介绍了系统的需求分析，用例描述以及系统的功能性需求以及非功能性需求。介绍了本人负责模块的详细设计。

第四章：系统的实现。对本人负责的模块与功能相应的实现以及使用的技术进行详细的介绍，并展示关键代码。

第五章：总结与展望。总结中文社科分析系统已经实现的功能，指出目前系统仍然存在的不足，并提出未来的系统改进以及发展方向。

第二章　中文社科论文分析系统技术概述

2.1　SpringBoot框架

2.1.1　SpringBoot框架简介

在java开发过程中，Spring框架的使用率很高。在javaweb开发过程中，SSH（Spring+Struts+Hibernate）架构或SSM（Spring+SpringMVC+MyBatis）架构经常被开发人员所使用。而使用上述架构，一般都要进行以下的步骤：首先，需要配置Maven的依赖项，通过修改pom.xml文件，添加项目所需要的所有依赖；然后使用javaconfig来进行服务层或dao层bean的配置，包括使用@EnableTransactionManagement启用基于注解的事务管理，@PropertySource注解定义从相关的properties文件中加载需要的属性，定义数据源等一系列配置；其次需要对Spring MVC Web Layer Beans进行配置，启用注解来进行MVC的配置以及指定url静态资源的位置等；其次注册Spring MVC FrontController Servlet DispatcherServlet，编写数据库的相关配置，创建Spring MVC控制器。当开发另一个类似的项目时，以上的配置过程需要重复进行，无疑耗费了开发人员大量的时间。因此，能够自动化完成绝大部分操作SpringBoot框架应运而生。

正如名字SpringBoot所示，boot代表着引导，指引。SpringBoot框架正是基于Spring框架的二次开发。SpringBoot开发的初衷在于帮助开发者在使用Spring的过程中，简化繁琐的配置流程，使得开发者可以不再花费大量时间用于重复化的配置，快速完成应用的搭建过程。SpringBoot框架使得创建独立的，生产级的Spring应用程序变得十分容易 [23]。

2.1.2　SpringBoot特性

SpringBoot旨在减少繁琐的spring配置过程，完成快速的应用搭建。它具有以下的特性：

1）创建了独立的Spring应用程序

2）极大简化了配置依赖管理的过程

当默认添加springboot-starter-web依赖后，将提供spring-webmvc，jackson-json等一系列依赖的库，不再需要单独在pom.xml中进行配置。

3）提供自动的配置

SpringBoot自动配置了常用的需要配置的一系列注册bean，如ResourceHandles，MessageSource等，并具有合理的默认值。同时，如果程序类路径中有内存式数据库驱动程序，比如HSQL，SpringBoot将自动创建datasource并注册TransactionManager bean，EntityManagerFactory。如果使用mysql，也只需要在application.properties文件中配置MySQL的连接参数，SpringBoot将使用这些属性创建datasource。

4）提供了嵌入式的servlet容器

spring-boot-starter-web将自动提取spring-boot-starter-tomcat，它将自动启动tomcat来作为嵌入式的servlet容器，因此开发人员不再需要外部安装tomcat 容器，同时，也不需要部署相应的WAR文件。

2.2 Elasticsearch搜索引擎

2.2.1 Elasticsearch简介

Elasticsearch是以Apache Lucene的全文搜索引擎为基础开发的进阶版搜索引擎 [24]。其关键性概念则主要涵盖节点（实例之一，一个节点一般部署于一台主机上）、集群（数个节点组成一个集群，通信某一集群内节点即可通信此集群）、分片（存储一个索引需要多个分片，且建立后分片的数量是固定不可更改的）、副本（用于提高系统容错率和检索效率的分片拷贝）、索引（与数据库的库类似的概念）、类型（与数据库的表类似的概念）、文档（与数据库的行类似的概念，可能包含多个字段）以及字段（最小搜索单元）。其优点主要包括性能高，近实时，零配置以及分布式等 [25]。

在已使用JDK配置了本地计算机中，一键式Elasticsearch部署使得用户只需将官方网站下载的压缩包解压缩后，于bin目录中运行Elasticsearch程序。Elasticsearch实例只要拥有相同的cluster.name，在多个主机上启动即可自动实现分布式集群。默认情况下Elasticsearch的分布式模式包含5个分片加一个复制项。对于每一个集群，各台主机上都会按照一定的算法来存储相应的分片，以保证集群的正常运转不会受到任一主机可能发生的意外下线的影响。

2.2.2 Elasticsearch数据存储结构

Elasticsearch是一种文档搜索引擎，即Elasticsearch 中会存储整个对象或者文档。但是，Elasticsearch并不仅仅提供存储功能，同时还提供了搜索功能。为了实现搜索功能，Elasticsearch会对存储的文档内容进行索引。在Elasticsearch中，用户可以对文档进行搜索，排序，过滤等相关操作。索引JSON格式文档，针对常规字段如num、string、bool等，Elasticsearch可以自动

识别；而对于特殊字段，例如ip（192.168.0.1）以及地理位置类型（[lat，lon]）等，Elasticsearch手动指定ip字段或geo字段即可实现索引。Elasticsearch极大地提高了索引效率，可以进行批量索引，包括全文索引和关系型数据库的结构化索引等。Elasticsearch的每个文档都必须包含3个元数据节点：_index，_type以及_id。其中_index是存储文档的单元，类似于传统数据库中的库。_id是一串字符串，与_type与_index组合可以成为文档的唯一标识。_id既可以在插入时指定，也可以通过默认方式，由Elasticsearch自动生成64位的字符串。

2.2.3　Elasticsearch High Level Rest Client 相关API

Java High Level Rest Client是目前新版本Elasticsearch比较推荐的使用Java访问Elasticsearch并进行增删查改等操作的封装包。本质上，High Level Rest Client是在地基客户端的基础上进行封装，可以与早期的接口TransportClient接受相同的参数以及返回相同的对象。它暴露出一系列操作Elasticsearch的相关方法，接收相应的请求对象并返回一个响应对象。本文中主要使用了其中的新增，删除索引，查询数据，批量操作的相关方法。在使用时，需要将相关的参数以json的格式进行编码并作为相关方法的参数。

2.3　jieba分词

2.3.1　jieba分词简介

jieba是基于Python，且可以同时兼容Python2和Python3的中文分词库。一般实现中文分词可以采用两种方法，其一是利用人工设置的匹配规则在自带词典中执行分词，其二则是通过对常见单词匹配及手动标记语料库概率分布的总结，学习训练而得以实现分词 [26]。jieba同时结合了以上两种分词方法，首先，jieba内置了一个包含很多单词的词典dict.txt，在分词时将内置词典加载入字典树中进行分析，根据树生成有向无环图（DAG），通过动态规划的方式得到最终的分词结果。jieba提供了三种分词模式供使用者选择：

1）精确模式：适用于文本分析的方式，会将句子进行最精确的分析并得到分词结果。

2）全模式：该模式下将句子中所有出现的单词全部扫描出来，但是可能会有歧义以及产生噪声词。

3）搜索引擎模式：在精确模式的分词结果上，对其中出现的较长的单词再次进行分词，从而可以提高搜索的覆盖面，常用于搜索引擎中。

同时，jieba还支持用户通过添加自定义词典的方式自定义分词的方式，从而使分词结果更加符合用户的心理预期。

2.3.2　jieba分词的相关算法

1）jieba分词实现词图扫描主要是通过Trie树结构算法来实现的。先扫描出一句话里词汇的多重组合成词的可能，生成有向无环图（DAG）。jieba分词自带了一个有2万多词条的词典（dict.txt）。Trie树结构的算法使得这些词条可以被放入其中，与扫描词条进行比对，通过寻找相同前缀以实现快速查找。

2）为了找到最大概率路径，jieba采取动态规划查找的方法，保证了最大概率路径。jieba分词在将词典生成Trie树的同时，对词频做出了计算，从而依此查找出最大切分的组合。

3）对于不包含在词典内的新词，jieba分词则基于Viterbi算法，应用隐马尔科夫模型（HMM）来实现。

2.4　Vue.js框架

2.4.1　Vue.js简介

Vue.js是一个可以按照用户需求组织应用程序的解决方案，其自身重点着眼于MVVM前端View和Model部分，可以在单页基础上嵌入现有页面，亦可配合其他库一同使用 [27]。模块构建系统可以被设置在应用中，Ajax 和路由则不包含于Vue.js核心功能内。Vue.js的优点主要包括灵活开放、组件化、模块化、响应式编程等。相比之下，AngularJS和ReactJS涉及较广，涵盖了页面应用所需的全部阶段及功能，基础功能的冗余重复使得现有框架难以直接应用，使用之后需要耗费很大的成本学习及维护基础框架。而Vue.js着眼于数据源头，无需复杂的DOM 操作，DOM元素的变化亦不会对其产生连带影响，展示了数据驱动的优越性 [28]。

2.4.2　vue-resource.js

vue-resource是Vue.js的一款插件，通过vue-resource插件，可以做到ajax所能做到的事情，并且使用vue-resource相比于ajax来说更为的简介。vue-resource 主要通过XMLHttpRequest或JSONP发起请求并处理响应，并具有如下特点：

1）vue-resource体积小，相较于jQuery非常小巧

2）vue-resource支持除了IE9以外的其他主流浏览器

3）支持拦截器

4）支持URI_Templates和Promise API

2.5 BootStrap框架

2.5.1 BootStrap框架简介

BootStrap一种广受欢迎的，用于开发响应式布局WEB项目的CSS、HTML、JS框架。BootStrap问世于2011年，主要由twitter的工程师MARK OTTO和Jacob Thornton 设计开发，是一种容易使用的可扩展前端工具集 [29]。最初仅仅用于公司内部工程师提高管理分析的效率，之后在github上实现了开源，得到了很多人的关注和支持。不仅有很多的工程师积极贡献代码，促进代码版本的进化，创作质量极高的官方文档；而且随之产生了很多界面简洁排版利落的基于BootStrap建设的网站。BootStrap框架的优势主要在于其响应式设计和充足的组件，可以在同一个网站内实现不同分辨率设备的兼容，极大地提升了用户的视觉体验，界面也简洁美观，符合主流审美。

2.5.2 BootStrap框架特点

1）因为基于html5、css3存在而使得此框架具备很高的兼容性、响应式布局、适应性较好的学习曲线、样式向导文档等

2）包含css、html、javascript工具集，可用于架构满足当下需求的交互接口和用户界面，拥有简单且易于操作的使用体验

3）可以进行JQuery插件的自定义操作等

2.6 Echarts库

2.6.1 Echarts简介

Echarts是Enterprise Charts的缩写，其程序语言为JavaScript，可适用于多种类的浏览器。ECharts内包含了多种实用性很强的图表类型及图标组件（如详情、图例等），且可以支持两种或多种配合使用以更加完备地展示，显示出直观、交互、自定义、可视化数据等优点 [30]。在引入Echarts前，需要先根据所需宽高计量DOM，接着通过init进行初始化操作，使用setOption进行后续的自定义选择，自由搭配所需图标种类进行绘制组合。其引入方法简单易操作，类似于其他JavaScript库；图表类型、图表组件多样，可以满足不同的用户需求。

2.6.2　Echarts与Highcharts对比

同样以JavaScript为基础的图表框架HighCharts与Echarts相比有很多的类似之处，都需要用到浏览器的渲染技术。两者操作均比较简单，基本上只要简单的JavaScript知识就可以掌握其使用方法，其示例也都有非常详细的图表及其对应的文档介绍，对于初学者十分友好。两者也各有所长，总的来说Highcharts的使用自由度更高，自定义选项更丰富，在进行图表配置及同种类图表绘制时，用户会拥有更多的选择 [31]。其应对数据量较大的图表时，表现力也很强，不容易出现卡顿，而Echarts的数据管理则需要使用额外的部件如datazoom来进行，或者仅截取部分数据进行后续操作。Echarts的比较优势在于其丰富的图表种类和开源免费。其图表的3D表现力较为先进，图表外观也十分顺应时代潮流，并且可以连同百度地图一起使用，方便便捷。两者都是数据可视化领域的重要存在。

2.7　本章小结

本章节主要介绍了在系统具体实现过程中主要使用的技术。主要介绍了搭建系统的主题框架Spring-boot，系统存储引擎Elasticsearch，分词模块使用的中文分词技术jieba，前端使用的BootStrap框架以及Vue.js框架。对于Spring-boot框架，主要介绍了Spring-boot框架的特性，正是由于Spring-boot框架的优点所以本系统采用该框架进行开发。对于Elasticsearch引擎，主要介绍了文档存储的方式以及操作Elasticsearch的相关api。对于jieba分词，主要介绍了jieba分词的使用以及相关算法。对于BootStrap框架以及Vue.js框架，主要介绍了框架的优点以及特性。最后介绍了画图表所使用的的Echarts库，并对比了Echarts与Highcharts各自的优缺点。

第三章　中文社科论文分析系统需求分析与概要设计

3.1　项目总体规划

本系统主要针对两类用户，数据管理员是系统后台数据的维护者，主要负责对数据进行管理；研究人员则可以查看数据统计分析的一系列结果，方便做更深层次的研究。因此，数据管理员可以操作系统完成一系列对数据进行处理的相关操作，包括将原始PDF论文文件转化成模型运算以及数据分析所需要的相关数据，将数据导入到Elasticsearch中，在Elasticsearch中查看存储的论文信息以及运算结果并能够进行编辑模型代码，运行模型等操作。研究人员则可以通过界面查看根据模型运算以及相关数据分析后整理的研究结果，包括主题的演化流变，相关作者信息，以便于后续更深层次的研究。同时，由于分析过程依赖于对文章标题等信息进行分词的结果，而通过修改自定义词典的方式可以对分词结果进行修正，因此开放了分词界面供研究人员查看，并可以联系数据管理员对分词结果进行相关修改。

3.2　系统用例分析

由总体规划分析可得，系统面向用户主要分为两类，一是数据管理员，负责后台数据处理的相关操作，包括数据处理，模型编辑与运行，存储系统的更新与维护等工作；另一类是研究人员，是系统的使用者，使用系统分析的相关数据来进行进一步的研究。因此，数据管理员主要需要对数据以及模型进行管理。数据管理方面包括对原始PDF文件的处理，对Elasticsearch存储的内容进行增删改查，对源文件格式进行转换最终生成满足模型以及运算需要的数据。模型管理主要包括对LDA模型以及BERT模型的相关代码进行编辑，运行。同时，研究人员主要可以查看对源文件的一系列分析运算结果，同时，因为分词结果对分析运算结果有较大影响，研究人员还可以帮忙通过添加或修改自定义分词来调整相关分词结果，使得分析结果更加符合预期。根据以上分析可得用例图3.1。

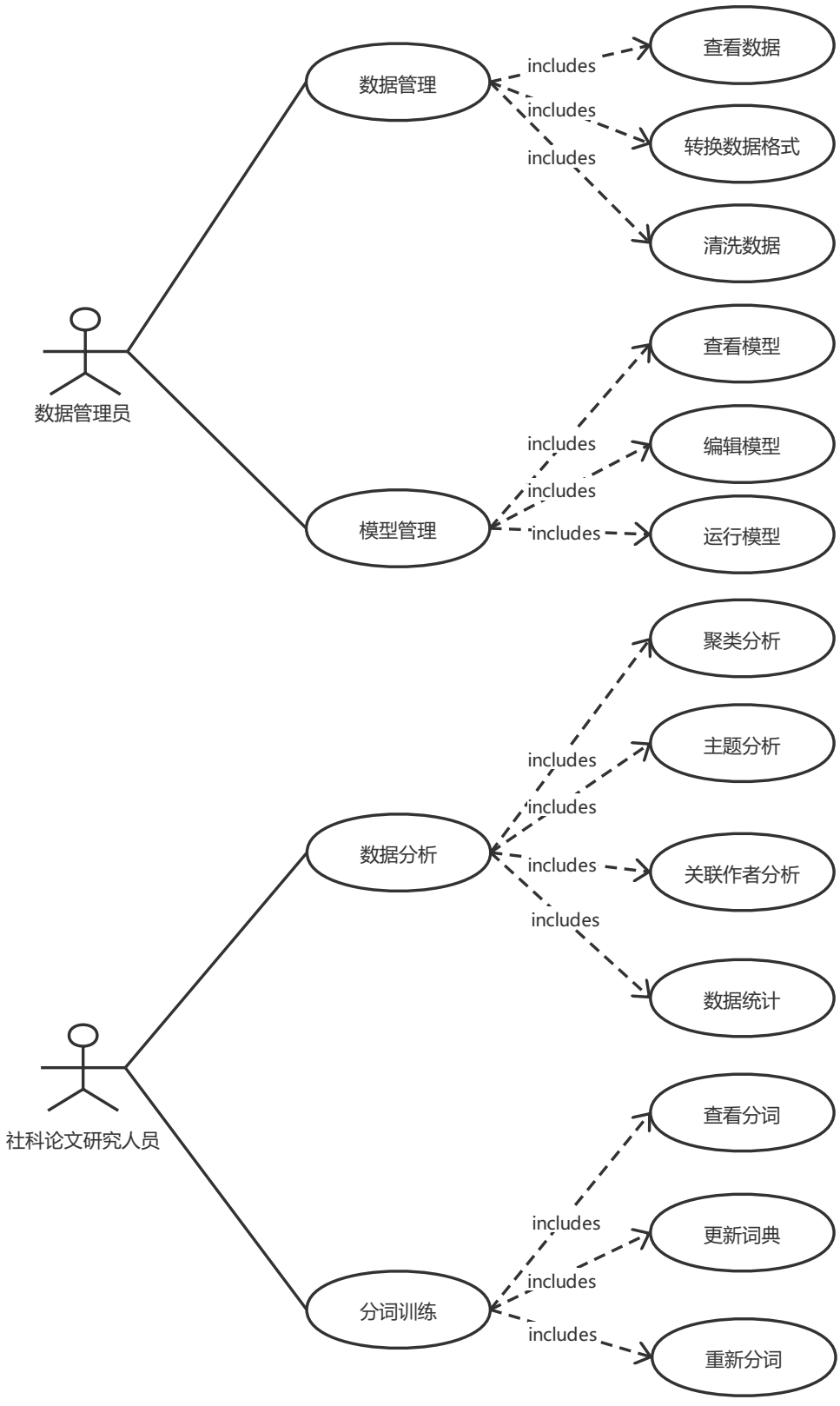

图 3.1: 系统用例图

如图 3.1所示，系统主要可以分为四个模块，包括数据管理，模型管理，数据分析以及分词训练。其中与数据管理员相关的是数据管理以及模型管理模块，与研究人员相关的则是数据分析与分词训练模块。数据管理模块主要负责对PDF源文件进行处理，整理出系统需要的数据并对数据进行转换，存储。因此数据管理模块主要分为查看数据，转换数据格式以及清洗数据三个子用例。查看数据负责对存储数据进行增删改查以及查看，清洗数据负责对PDF源文件进行清洗，整理出关键信息，转换数据格式负责将从PDF提取出的信息进行转换，生成系统模型以及分析需要的数据格式文件。模型管理模块主要负责对LDA模型以及BERT模型代码进行管理，主要包括查看模型，编辑模型以及运行模型三个子用例。数据分析主要负责对源数据采取一系列方法进行分析整合，最终呈现研究人员需要的数据内容。主要包括聚类分析，主题分析，关联作者分析以及数据统计四个子用例。分词训练模块主要负责将分词结果展示给研究人员，研究人员根据自己的需要调整分词并进行重新分词。主要包括查看分词，更新词典以及重新分词三个子用例。

3.3　数据分析模块需求分析

3.3.1　数据分析模块用例图

由系统用例图可得，数据分析模块主要分为聚类分析，主题分析，关联作者分析以及数据统计四个部分。本人主要负责了关联作者分析以及数据统计的设计与开发工作。

为了方便系统使用者更好地了解论文的信息，系统将提供论文作者的信息。经过分析，关联作者用例主要包含以下需求：能够查询该作者的所有作品，该作者发表的期刊以及作者研究主题的演变，同时能够直观的以年份为单位展示该作者发表的作品，同时在同一时间与该作者研究同一主题的学术圈。系统使用者还可以找到该作者发表的最有价值的论文以及成就最高的研究主题等。详细的需求及用例描述将在小节中展示。

对分词文件，本系统为管理人员与研究人员提供了数据统计用例。通过该模块，研究人员可以了解某分词文件中出现的词频以及最高词频关键词的气泡图的作者及机构词频的分布图。同时，点击气泡图中的关键词，用户能够查看某一个词的具体信息。同时，研究人员还能够手动输入某一关键词进行搜索。详细的需求及用例描述将在小节中展示。

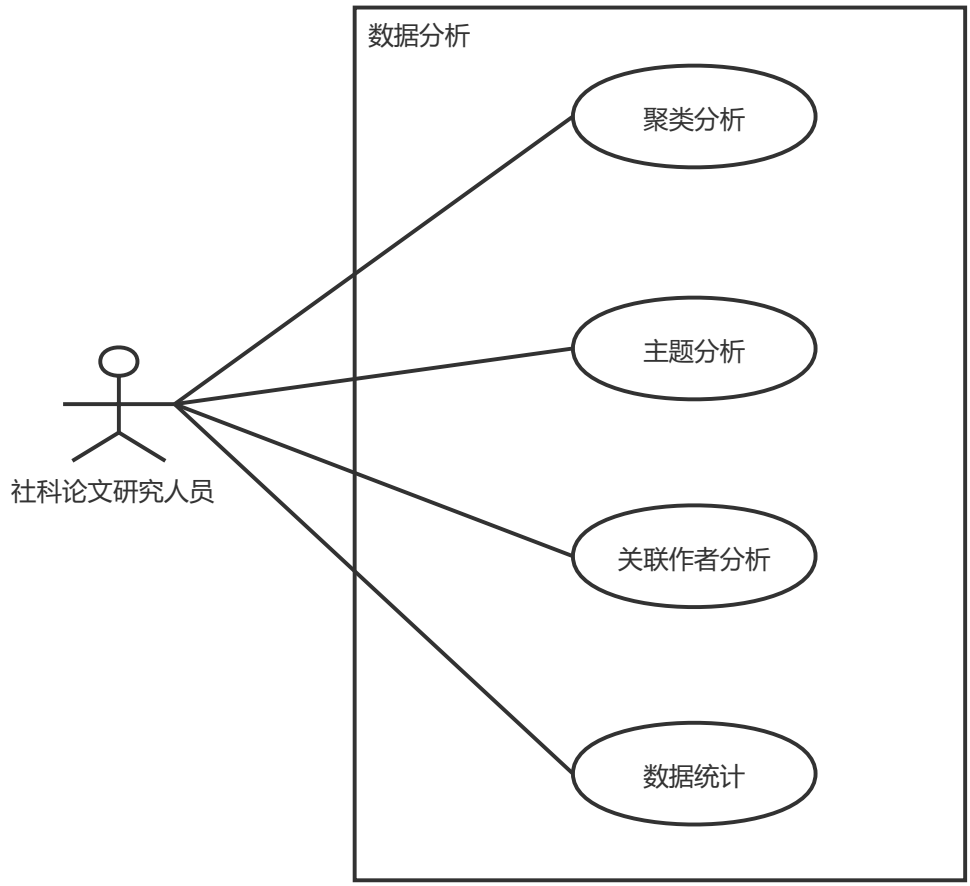

图 3.2: 数据分析模块用例图

3.3.2　数据分析模块用例描述

关联作者，展示作者所发表的所有作品以及期刊，是数据分析模块的核心用例之一，主要功能在于当用户点击论文信息中作者这一栏的时候，将会跳转到作者信息展示页面，系统将会以年份为单位以表格和视图的方式展示作者所有的作品，期刊以及作者研究成果最高的主题及相应论文。当用户点进详细的年份时，可以查看到该年作者研究的详细信息，包括论文，期刊以及研究同一主题的学术圈等。具体描述见表 3.1。

表 3.1: 关联作者分析用例描述

ID	UC01
名称	关联作者分析
参与者	用户
前置条件	用户点击作者按钮
后置条件	跳转至展示页面
主要事件流	1.用户点击作者姓名按钮。 2.系统按年份显示作者发表的论文及期刊 3.系统显示论文以及期刊发表数量的图表 4.系统显示作者研究成果最高的主题以及作品 5.用户点击详细的年份 6.系统按照年份显示当年研究的详细信息
次要事件流	无

数据统计，是除了作者信息以外的其他信息的集中展示，其中包括分词文件中的词频统计展示以及搜索关键词的统计结果展示。具体描述见表 3.2。

表 3.2: 数据统计用例描述

ID	UC02
名称	数据统计
参与者	论文管理员
前置条件	论文管理员查看选中分词文件的词频统计
后置条件	系统显示统计结果
主要事件流	1.用户选中分词文件并点击查看 2.系统显示该分词文件的词频总览气泡图以及词频分布图 3.用户分析词频统计结果 4.用户检索某一关键词查询相关统计结果 5. 系统显示该关键词词频统计 6.用户点击气泡图查看某词详细结果 7.系统显示该关键词词频统计结果
次要事件流	3a.用户对词频统计结果不满意 3a.1.用户反馈给社科人员，对关键词进行调整 4a.用户搜索的关键词不存在 4a.1.系统提示该关键词不存在 4a.2.系统展示词频总览气泡图

3.3.3　数据分析模块相关需求分析和非功能需求分析概述

由用例分析可以看出，数据分析模块是系统的根本模块。所有系统展示的内容都依赖于该模块对数据的处理。该模块一共分为四个用例，本人在项目中

负责关联作者分析与数据统计部分的设计实现，以下将介绍两个部分的需求分析。

本系统对社科论文的分析主要包括两个方面：主题分析与关联作者分析。由此可见，关联作者分析是项目中核心功能之一。由用例分析可知，关联作者分析主要负责相关作者信息的挖掘以及展示。对于关联作者相关信息的挖掘，系统主要从基本信息以及深度信息两方面对作者进行全面的分析展示。作者的基本信息主要包括：

- 作者所发表的所有论文
- 作者发表的相关论文的时间轴
- 作者发表的期刊

对于作者信息的更深一步挖掘，系统将结合主题分析的结果。由于每篇论文有其对应的主题，因此，将主题分析的结果与作者发表的论文信息进行结合，系统将更有针对性的对作者的深层信息进行发掘。作者深度信息分析主要包括以下方面：

- 作者所有论文分别属于哪些主题，这些主题代表了该作者的研究领域
- 作者发表的最具有代表性的论文
- 作者研究最深入的主题
- 根据作者研究的相关主题，整理作者研究方向的流变
- 整理与作者同一主题的相关联作者，并根据研究水平进行相关作者的推荐

对作者信息挖掘完毕后，将对以上信息进行展示。为了让用户能够清晰的获得相关信息，关联作者分析模块的展示将分为作者概要信息展示以及作者详细信息展示。概要信息方面能够展示作者的总体信息，包括以图表的信息按照年份展示作者发表的所有期刊、论文，作者成就最高的论文以及研究方向，作者研究方向的流变。当用户点击具体年份，可以查看相关年份的详细信息，包括作者在该年发表的所有的论文，该论文所属的主题，以及研究同一主题的论文及学者的相关推荐。结合以上分析，整理关联作者相关的功能需求如表 3.3所示。

表 3.3: 关联作者功能需求

需求ID	需求名称	需求描述
R01	作者基本信息展示	用户能够通过点击作者名字，进入该作者信息总览界面，系统将会按照年份将用户发表的论文以及期刊展示出来
R02	作者成就	系统将根据作者发表的论文与主题，通过被引量，文章对主题的贡献价值等方面，对用户展示该作者研究成就最高的论文以及主题
R03	作者研究方向流变	系统将根据作者发表的论文所对应的的主题，展示作者研究的领域随着时间发生的变化
R04	作者详细信息	用户能够通过总览中相应的年份，进入查看该年份作者的详细信息页面，包括当年该作者具体发表的论文和期刊
R05	关联作者推荐	针对作者研究的相关主题，系统能够根据该主题下的论文排名，推荐同一主题的优秀论文以及优秀学者

数据统计包含了除了作者信息之外的其他一些内容的统计。系统将统计某指定文件分词结果的词频，并展示给用户分词文件的词频总览图和最高词频关键词的作者以及机构，用户可以分析词频统计结果，如果用户对词频的统计结果不够满意，用户可以反馈给研究人员，研究人员会对关键词进行调整。同时，用户可以通过点击总览图中的某个关键词，查看该关键词详细统计结果。用户还可以通过搜索的方式查看指定的关键词信息。数据统计相关功能需求如表 3.4。

表 3.4: 数据统计功能需求

需求ID	需求名称	需求描述
R06	统计结果总览	用户选定分词文件后，系统将对分词结果进行统计，向用户展示词频总览气泡图和最高词频关键词的作者以及机构
R07	用户反馈	当用户对词频统计结果整理的关键词不满意时，可以通过系统提交结果反馈，系统将会通知研究人员
R08	关键词修改	研究人员收到用户反馈后，可以对关键词做出调整
R09	查看关键词详细统计结果	用户点击总览图上的某个关键词，系统将展示某个关键词的详细信息
R10	搜索关键词	用户可以直接搜索想查看的关键词，系统将展示搜索的关键词的详细信息，如果无该关键词，系统将提醒用户该关键词不存在

对于数据分析模块而言，数据展示是其中关键的部分，对于数据展示，系统应能做到3s内将用户需要的信息展示出来。同时，系统应对用户有足够的引导，使用户能够快速便捷地查看自己想要查看的内容。对于搜索功能，当用户搜索的内容不存在时，需要给用户明确的提示。对于展示模块，系统应具有良好的可扩展性，方便用户随时增加展示的内容需求。

3.4　分词训练模块需求分析

3.4.1　分词训练模块用例图

由系统用例图可得，分词训练模块包括查看分析，更新词典，重新分词三个用例。

查看分析是分析训练模块的核心用例之一，因为目前所采用的的分词方法，对专业名词的分词可能出现偏差，需要研究人员手动进行校对，因此研究人员可以选定文件查看相应的分词结果进行确认。

在工作人员确定分词效果之后，可以通过更新词典的方式矫正分词结果。

在更新完词典后，研究人员选择重新分词，系统将会根据词典中新加入的名词进行重新分词，达到更好的效果。

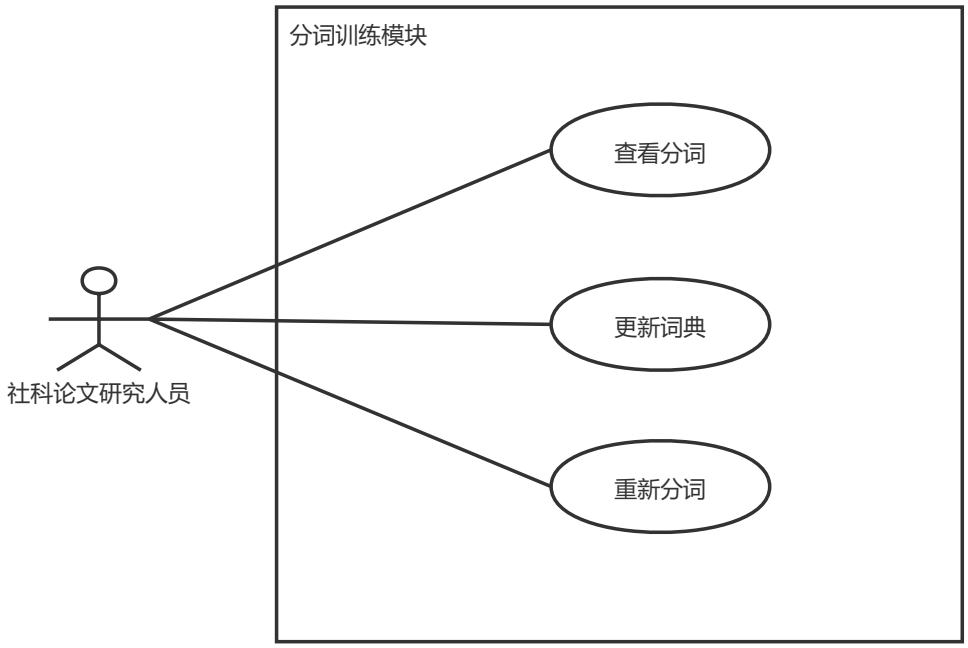

图 3.3: 数据分析模块用例图

3.4.2　查看分词结果模块用例描述

查看分析分词结果，是分词训练模块的核心用例之一。研究人员选定指定的分词文件，系统展示相应的分词结果供研究人员核对。具体描述见表 3.5。

表 3.5: 查看分词结果用例描述

ID	UC03
名称	查看分词结果
参与者	研究人员
前置条件	研究人员选择分词文件
后置条件	系统分词后展示分词结果
主要事件流	1.研究人员选择分词文件 2.系统显示分词结果
次要事件流	无

更新词典，是分词训练的核心用例。当研究人员在查看分词结果后，如果对分词结果不满意，可以选择将一些专业分词手动写入词典中，用于调整分词方式。具体描述见表 3.6。

表 3.6: 更新词典用例描述

ID	UC04
名称	查看分词
参与者	研究人员
前置条件	研究人员更新完分词词典并上传
后置条件	系统显示上传结果
主要事件流	1.研究人员在选择框中增加自定义分词 2.系统将原词典替换为新的词典 3.系统显示更新词典结果
次要事件流	无

在研究人员更新完词典后，可以选择重新分词，验证新的分词结果，不断迭代，获得满意的分词效果。具体描述见表 3.7。

表 3.7: 重新分词用例描述

ID	UC05
名称	查看分词
参与者	研究人员
前置条件	研究人员点击重新分词按钮
后置条件	系统重新分词并展示
主要事件流	1.研究人员点击重新分词按钮 2.系统根据新词典重新分词 3.系统显示重新分词结果
次要事件流	无

3.4.3　分词训练模块相关需求分析和非功能需求分析概述

由用例分析可以看出，分词训练模块是系统正常运作的前提。论文原始信息需要通过分词处理来生成系统需要的文件。因此，分词是否准确关系到系统一系列分析结果的展示是否客观准确。然而，中文分词具有其特殊性，如某些专业名词，特殊用词可能无法准确的分出。因此，分词训练模块需要人工的干预。通过分词- 人工检查-更新词典-重新分词的不断迭代，最终获得令人满意的分词结果。因此系统需要提供分词结果展示，词典更新，以及运行分词的功能。分词训练相关的功能需求如表 3.8。

表 3.8: 分词训练功能需求

需求ID	需求名称	需求描述
R11	分词结果展示	研究人员选择文件，系统将分词结果进行展示
R12	修改词典	研究人员添加或删除自定义分词，系统在词典中新增或删除对应的分词
R13	重新分词	研究人员点击**rerun**按钮进行重新分词操作，系统将会根据新的词典对源文件重新分词生成新的分词结果
R14	验证修改效果	研究人员观察日志，确认系统重新分词，随后刷新页面，系统将显示新的分词结果

分词训练模块的相关操作，系统均需要在用户操作后3s内予以操作反馈，系统能够支持主流浏览器的访问。对于展示部分，系统应具有良好的可扩展性，方便用户随时增加展示的内容需求。

3.5　系统总体设计与模块设计

3.5.1　系统总体设计

由以上的用例分析以及相关功能需求的整理，在系统设计方面可以进行相对应的设计。首先，系统数据量以及访问量目前相对而言较少，因此采用单服务器进行部署即可满足日常访问需求。在功能方面，系统需要实现数据管理，模型管理，数据分析以及分词训练四个模块，同时需要Elasticsearch引擎存储与文件系统存储相关功能支持。在架构方面，系统将采用MVC架构，将数据，逻辑与界面进行分离，便于分工合作进行相关功能的开发，同时视图与业务分离，使得系统具有良好的扩展性。

为了更加直观的显示系统的功能，模块设计以及架构，以下将从逻辑视图，架构视图，开发视图以及进程视图四个方面进行详细描述。逻辑视图主要根据面向对象的原则，详细描述了系统包含的对象模型。架构视图则从系统组织架构方面对系统予以阐述。开发视图站在开发者的角度，将系统划分为子系统以及相关功能模块，方便开发过程中的分工合作。进程视图则抽象的反映了系统的进程结构，强调了并发性，系统容错能力等非功能需求。

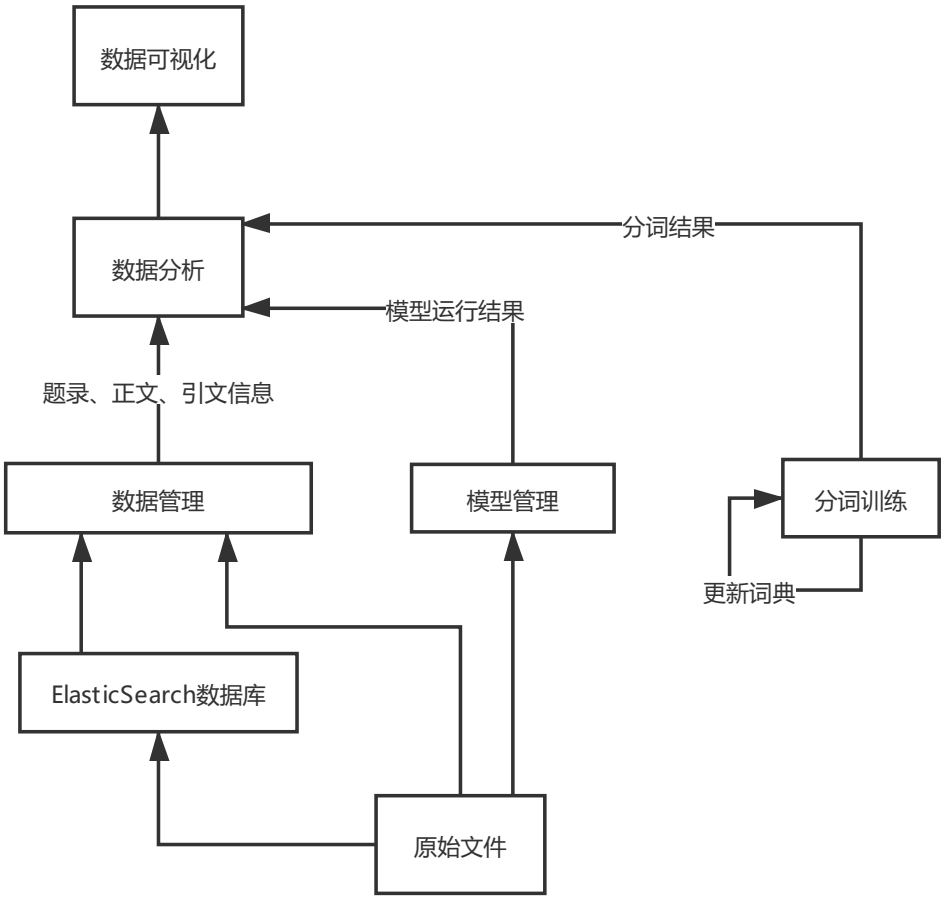

图 3.4: 系统逻辑视图

图 3.4展示的是系统的逻辑视图。系统分为数据可视化，数据分析，数据管理，模型管理，分词训练五个模块。系统的数据持久化分为两部分，处理过后的数据存储在ElasticSearch数据库中，如论文的名称，期刊，作者等，而另一些中间文件，如pdf，分词文件，则存储在文件系统中。模型管理模块负责模型的训练与存储。分词训练模块负责将分词文件进行分词，词频统计，同时能够根据研究人员的干预，通过不断更新词典的方式，达到一个满意的分词效果。数据分析模块负责对数据进行加工，通过调用模型以及数据挖掘等处理方式，将数据加工整理成用户感兴趣的形式，并通过数据可视化部分与用户进行交互，向用户展示处理结果。

图 3.5展示的是系统架构图。如图所示，系统采用springmvc架构，包括View，Http，Model，Controller，Dao五个模块。用户与界面进行交互，浏览器发送Http请求到Http模块，Http模块根据vue-resource的规则将请求转到Controller中。Controller将其中的逻辑交给相应的Model模块进行处理。Model模块需要的数据

通过调用Dao模块来获取。

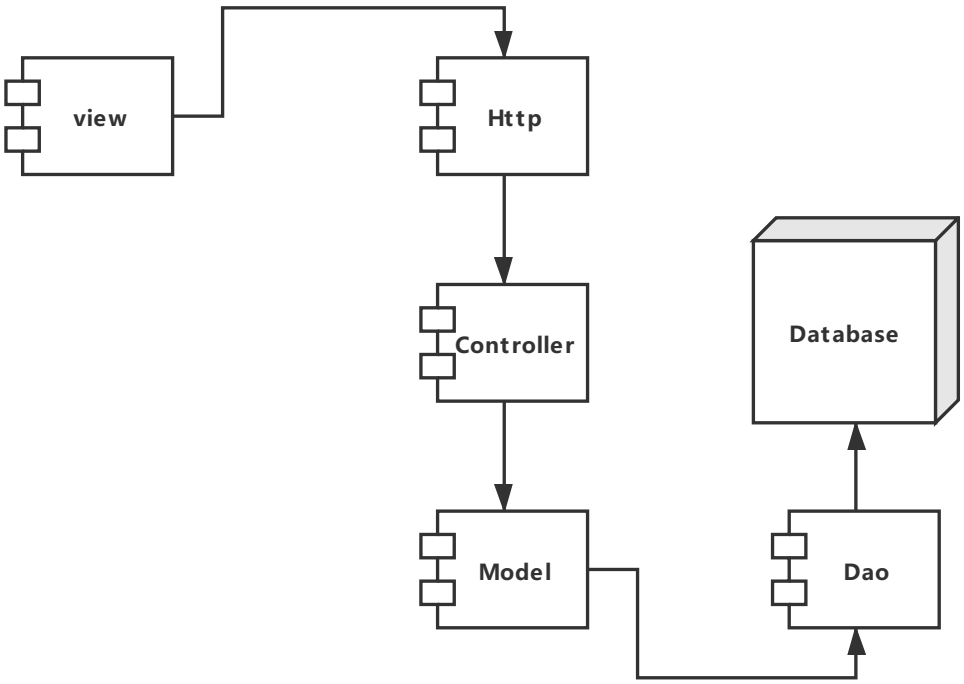

图 3.5: 系统架构视图

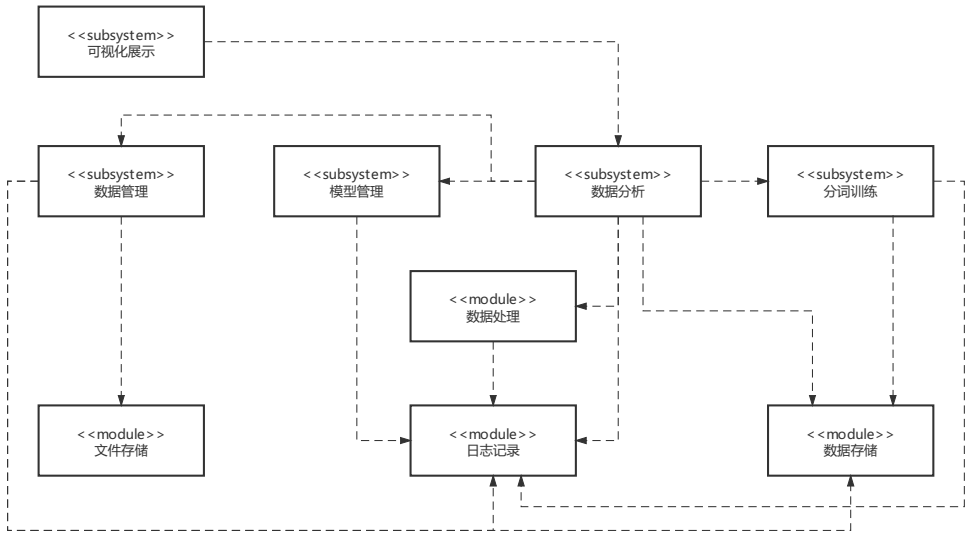

图 3.6: 系统开发视图

图 3.6展示的是系统开发视图。系统可以分为数据管理，模型管理，数据分析，分词训练，可视化展示五个子系统以及文件存储，日志记录，数据存储，数据处理四个模块。其中数据管理，分词训练，数据分析三个子系统向数据分

析子系统提供数据，在数据分析子系统中进行处理，包装后发送至前端展示。所有的子系统与数据处理模块都有相应的日志记录供研究人员分析，都依赖于日志记录模块。文件存储模块负责存储各种中间结果，分词结果，原始数据等文件，数据存储模块则以数据库的形式对处理过的数据进行持久化。数据管理，模型管理，分词训练所用的数据都来源于这两个模块。数据处理模块根据已经训练好的模型以及其他统计学的方法，对收集的数据进行各种加工，数据分析的过程中依赖于数据处理模块。

图 3.7展示的是系统进程视图。系统分为四个子系统，但是在运行时，系统可以抽象为展示进程与负责数据处理的主进程。由于网络请求异步进行，所有的数据请求都会在单独的进程中进行处理。

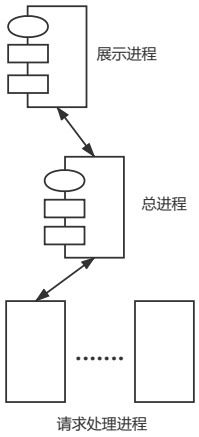

图 3.7: 系统进程视图

3.5.2　数据分析模块设计

由之前的用例分析可知，数据分析模块分为聚类分析，主题分析，关联作者分析，数据统计四个部分。这四个部分相互独立，分析的方式也有所不同。本人负责关联作者分析与数据统计两个部分，各部分的详细设计将在下面小节中展示。

由需求分析可知，关联作者分析是系统的核心功能，主要分为两部分：作者概要信息展示以及作者详细信息展示。作者的相关信息来源于两方面：作者的一些基本信息来源于对原始信息的整合，从原始数据中，对作者发表的论文，期刊按照时间顺序进行归纳整理，得到作者作品、期刊的集合。同时，根据已有的基本信息，结合主题分析模型运算的结果，对作者信息进行更深层次的挖掘。通过向主题运算的相关接口进行请求，可以获得作者研究的论文所属的主题，在该主题下的排名等重要信息，将主题相关的信息进行处理，最

终构成了该作者完整的信息分析。最后，将分析结果存储在对象中并批量导入Elasticsearch中进行存储。该过程的详细流程图如图 3.8所示。

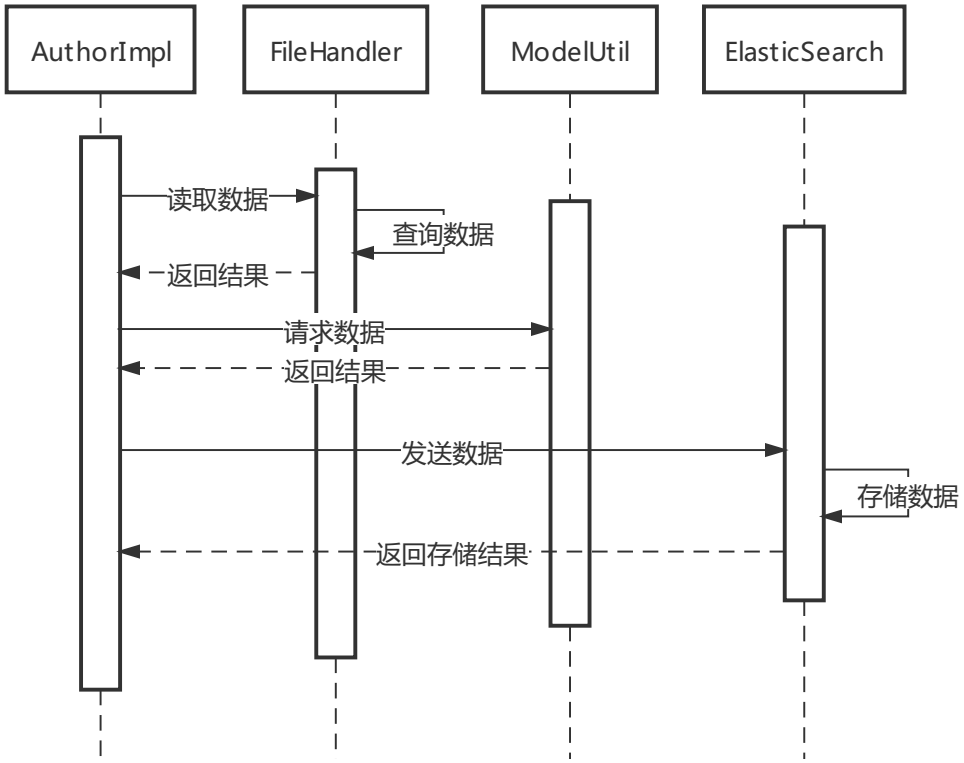

图 3.8: 数据分析模块关联作者分析流程图

由于作者分析的相关计算需要结合主题运算结果进行，而主题运算速度较为缓慢，因此作者分析完成后的分析结果存储在Elasticsearch中，在展示时对相关数据库进行查询并将结果传递至前段进行展示。关联作者分析结果展示主要包括作者概要信息展示与作者详细信息展示两部分组成。作者概要信息将会以图表的方式展示历年来作者发表的论文与期刊以及根据算法得出的该作者成就最高的研究论文以及研究主题。作者详细信息则会展示具体年份中作者发表的论文，期刊以及该年作者研究的主题。同时展示研究该主题的一些作者以及该主题下有价值的论文。为了实现上述需求，根据面向对象的原则，可以得到类图如 3.9 所示。

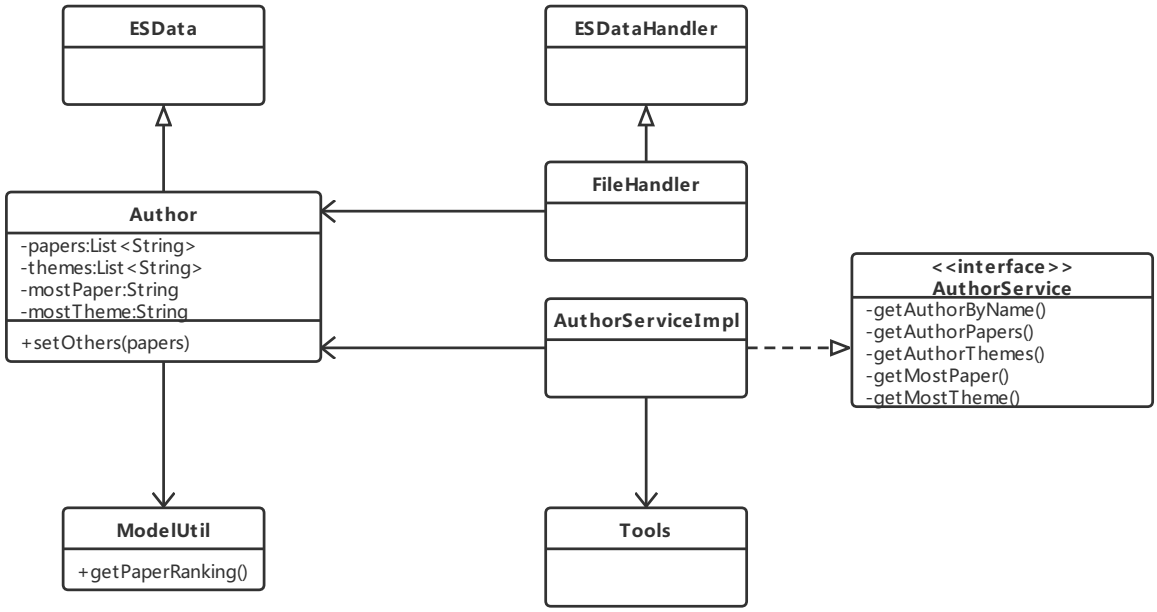

图 3.9: 数据分析模块关联作者部分类图

如图 3.9所示，Author类继承自数据基类EsData，定义了作者所包含的一系列属性，包括作者发表的论文列表，作者发表的论文所属主题列表，作者最有价值论文以及作者成就最高的主题等信息，其中主题相关的信息包括文章属于哪个主题，文章在主题下的排名等则依赖于模型预算的结果。因此需要模型运算部分的接口ModelUtil提供相应的方法返回主题与论文对应的列表。在Author类封装完毕后，AuthorHandler类中相关方法负责将作者相关信息导入至Elasticsearch中。AuthorHandler类继承自基类EsDataHandler，实现了自定义的handleData()方法，对封装完成的作者类进行导入操作。界面展示过程中，则由AuthorService接口定义了从Elasticsearch中查询作者信息的相关方法，并由AuthorImpl类进行具体实现。Elasticsearch查询部分代码由于可以共用，因此在Tools类中进行实现，供AuthorImpl类调用。

由于作者信息被预先处理好存储在Elasticesarch中，作者概要信息与作者详细信息的流程图流程大致相同。如图 3.10所示，研究人员选择用户并点击查看用户信息，界面向AuthorService接口发送请求参数，AuthorService通过实现类AuthorImpl，调用工具类中的Elasticsearch相关查询方法，查询Elasticsearch并获得相应的返回对象response，在AuthorImpl中对返回对象进行解析封装，返回至界面，最后进行展示。

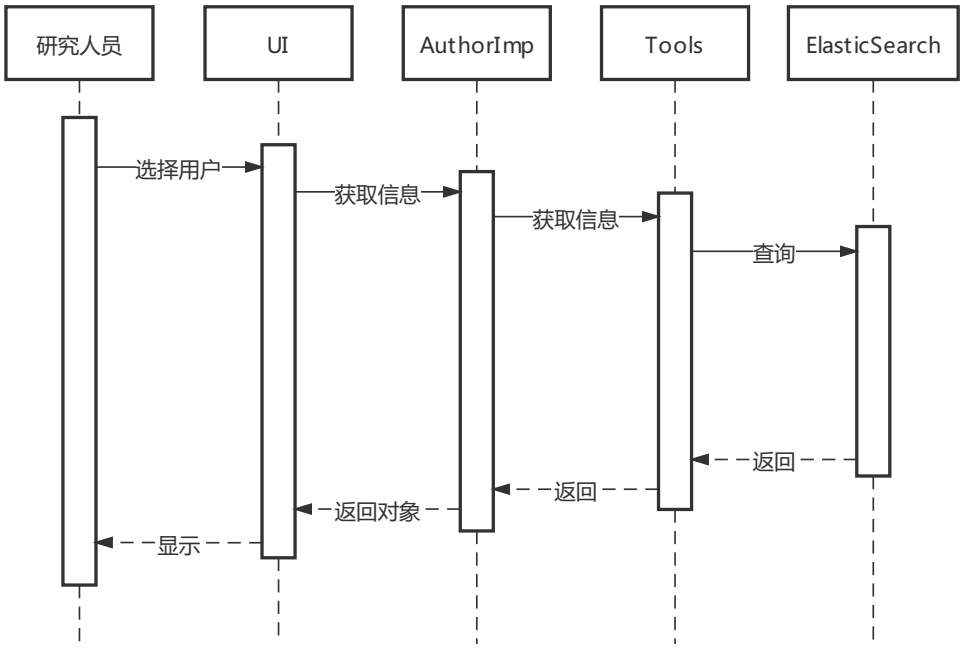

图 3.10: 关联作者概要信息展示流程图

数据统计模块是对词频等信息的统计展示，主要分为三个部分：数据读取，数据分析以及数据展示。

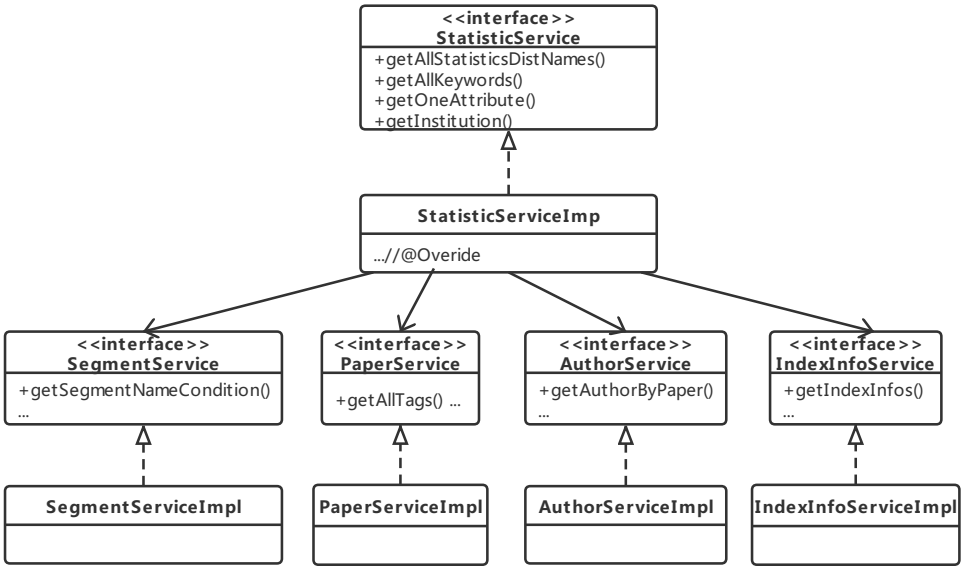

图 3.11: 数据分析模块数据统计部分类图

系统从数据库以及文件存储系统中提取数据进行计算，向用户展示关键词的词频，热词榜以及选择的词语的作者，期刊以及机构分布。为了实现上述需

求，根据面向对象的原则，可以得到类图如 3.11所示。接口StatisticsService定义了实现需求所需要的一系列方法。包括获取关键词，获得词频等方法。statisticsImpl是对接口方法的实现，通过底层数据定义的接口向存储系统请求数据。SegmentService，AuthorService，PaperService，IndexInfoService接口分别定义了从底层存储中获取分词信息，作者信息，论文信息以及其他相关信息的一系列方法。SegmentImpl，AuthorImpl，PaperImpl以及IndexInfoImpl类则是对上述接口的具体实现。

图 3.12为用户查看数据统计的流程图。当用户进入界面时，界面向数据统计接口发送请求，接口通过实现类StatisticsImpl向底层数据接口发送请求，底层数据接口通过相应的实现类，向数据库或文件存储系统请求数据并返回。最后在StatisticsImpl类中进行处理统计，返回给界面进行显示。

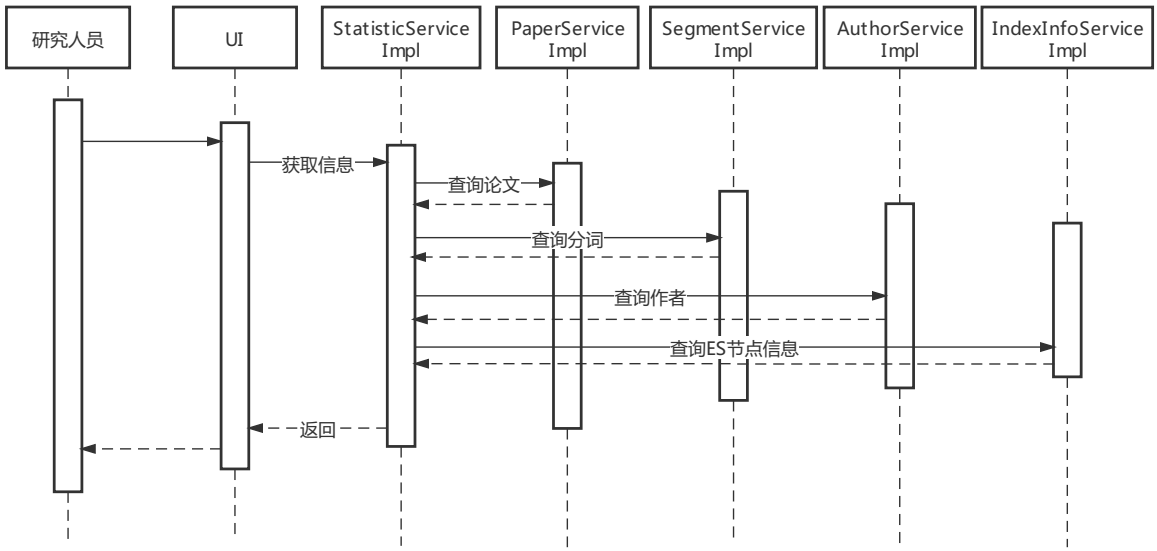

图 3.12: 数据统计展示流程图

3.5.3　分词训练模块设计

分词训练模块需要向用户展示指定文件的分词结果，并支持用户对词典进行添加或删除操作。在更新词典后，用户可以进行重新分词查看更新词典的效果。为了实现以上需求，根据面向对象原则，设计的类图如 3.13所示。

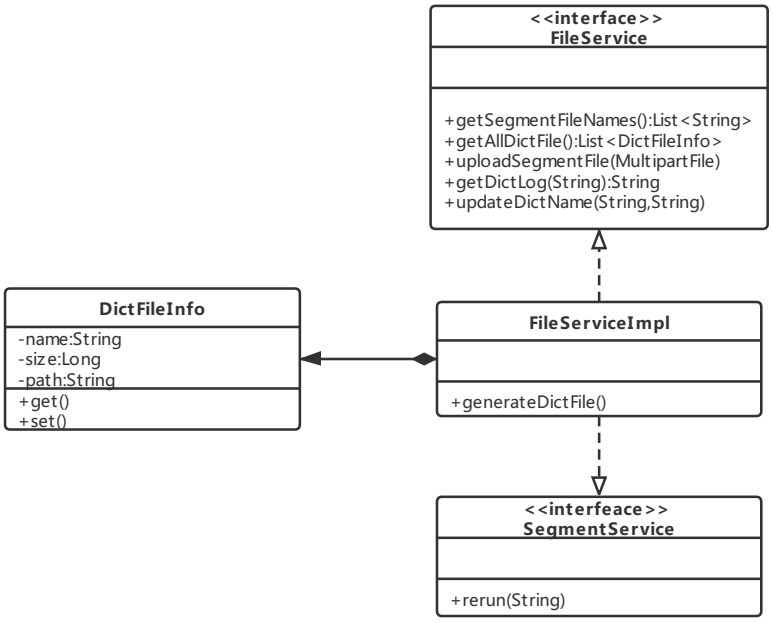

图 3.13: 分词训练类图

如图 3.13所示，Segment类定义了分词的基本属性，包括原字符，分词后的字符列表。当分词文件过大，为了显示速度与查看方便，将进行分页显示，PageDataInfo类定义了分页的属性，包括页码，一页的数据等。接口SegmentService定义了获取分词，运行分词脚本，增加词典，删除词典，记录日志的相应方法，SegmentServiceImpl类实现了上述接口。

分词模块是需要研究人员参与完成的模块，分词结果需要研究人员进行查看，修改，迭代进行，最终得到满意的结果。因此与研究人员的交互主要分为两大流程：查看分词与重新分词。相关的流程图如下所示。研究人员点击查看分词可以看到上一次分词结果，如果不满意，可以在更新词典后选择重新分词，系统将重新运行分词程序并展示新的结果，直到得到满意的分词结果。相关流程图如下所示。

图 3.14为用户查看分词的流程图。当用户选择指定文件查看该文件的分词结果时，界面向接口发送请求，接口通过实现类SegmentServiceImpl从底层文件中获取数据返回给界面，界面对数据进行展示。

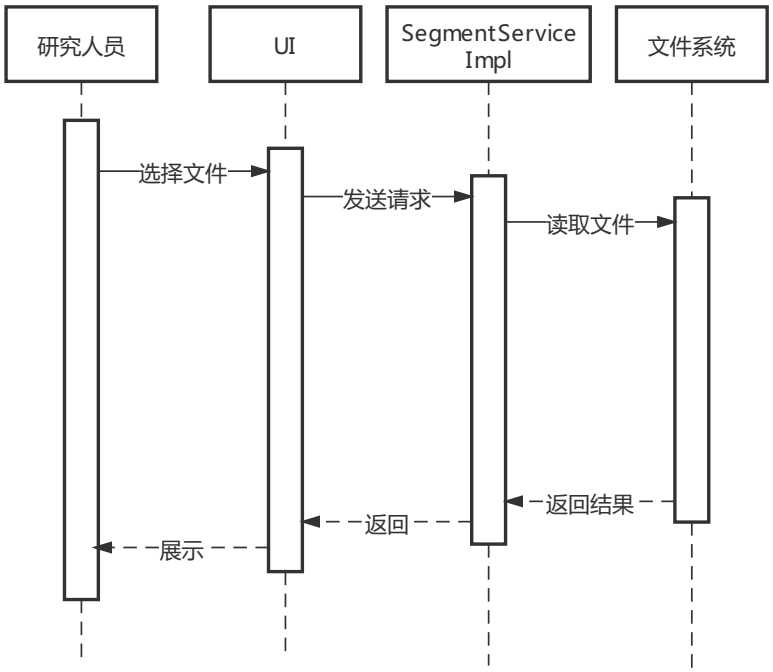

图 3.14: 查看分词流程图

图 3.15为用户更新词典后重新分词的流程图。当用户点击重新分词后，界面向接口发送请求，接口通过实现类SegmentServiceImpl调用python脚本重新进行分词，并记录相应日志。

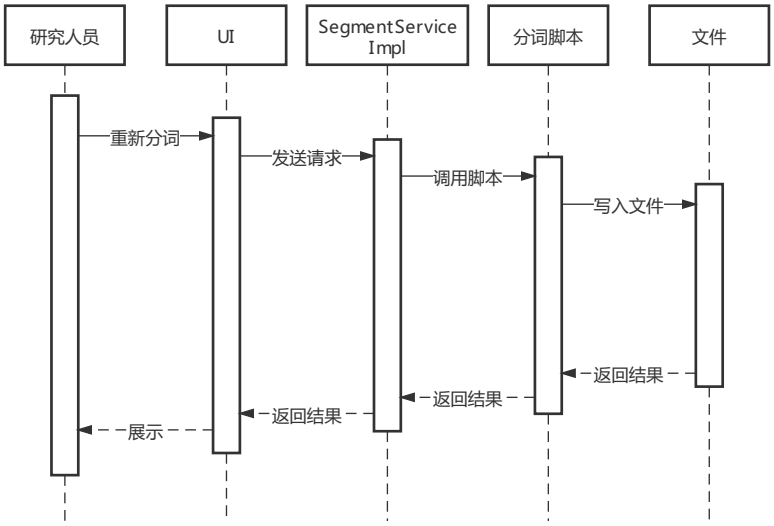

图 3.15: 重新分词流程图

3.6　本章小结

本章主要对系统的需求进行了分析，整理出了相关用例，并根据用例对系统进行了概要设计。在用例分析部分，主要整理得出了系统的总用例图，并根据总用例图拆分得到各个模块的用例图，并对模块中的各个用例进行了详细的分析，整理了功能需求以及非功能需求。在需求分析完毕后，本章对系统做了概要设计。首先分析了系统的总体设计，并给出了系统的逻辑视图，架构视图，开发视图以及进程视图。最后，根据整理的需求对模块进行设计，对各个模块给出了概念类图以及某些交互的流程图。

第四章　中文社科论文分析系统的相关实现

4.1　导入数据的相关实现

为了方便数据的分析以及搜索，本系统的数据主要存储在Elasticsearch中，需要存储在Elasticsearch中的数据包括经过处理的论文的概要信息，包括论文标题，作者等概要信息，论文的全文，摘要等详细信息，以及模型运算的中间结果，模型相关代码等等。不同的数据源存储格式可能有所差别。如论文概要信息的中间文件存储于csv文件中，而运算中间结果可能来自于数据库。同时，导入数据功能相对于系统而言并非实时需要的功能，而是定期更新即可。因此，导入数据的功能需要满足以下条件：支持配置文件配置数据来源以及处理数据的方法，支持单独索引的导入更新。

4.1.1　导入数据功能设计

由上述分析可知，导入数据功能需要能够处理来源不同的数据源。不同的数据源的处理方式会有相当的不同，但是通过处理后将数据导入Elasticsearch中的部分是完全一致的。因此，在设计方面，采取工厂模式来通过输入的参数为不同的数据源选取相应的处理类，通过处理后，调用相应的方法将处理后的数据导入进Elasticsearch中。因此，导入数据功能主要分为三个主要方面：数据源方面，Elasticsearch相关操作以及对数据进行处理并调用Elasticsearch操作接口完成导入操作的处理类。以下将对代码进行展示。

4.1.2　数据源方面代码

如图 4.1所示，所有的数据类型都有特定的数据类，其中定义了需要导入到Elasticsearch的所有的属性值。同时，所有需要导入到Elasticsearch中的数据类都继承自基类EsData。EsData类中定义了所有数据类都必须实现的方法，如toString()，使得导入Elasticsearch的代码得以复用，同时也利于今后导入数据需求发生变化时对代码进行更新。

```
/**
 *所有进入 es 的数据类都必须继承此接口，此接口规定了数据类必须实现的方法
 *目前只有 toString()方法需要实现
 * @author YEJIFAN
 *
 */
public abstract class EsData {

    public abstract String toString();

}
/**
 * 论文概要信息类，主要包括标题，作者，期刊，年份，关键词，摘要等属性
 * @author YEJIFAN
 *
 */
public class ResumeData extends EsData{

    private String title;
    private String author;
    private String journal;
    private String year;
    private String keywords;
    private String abstracts;

    public ResumeData(String title, String author, String journal, String year, String ke
String abstracts) {
        this.title = title;
        this.author = author;
        this.journal = journal;
        this.year = year;
        this.keywords = key;
        this.abstracts = abstracts;
    }

    @Override
    public String toString() {
        // TODO Auto-generated method stub
        return "title:"+title+"author:"+author+"journal:"

+journal+"year:"+year+"keywords:"+keywords+"abstracts:"+abstracts;
    }

}
```

图 4.1: 数据基类代码

4.1.3　Elasticsearch相关方法

```
/***
 * 该接口提供对 elasticsearch 中 index 的一些操作
 * createIndex 是创建新的索引
 * deleteIndex 是删除指定索引
 */
public interface ESIndexService {
    //创建新的索引，以及设置别名
    void createIndex(String name,String alias,String type);
    //删除指定索引
    void deleteIndex(String name);
    //判断索引是否存在
    boolean isIndexExist(String indexname) throws IOException;
     //批量将数据导入指定的 index 中
    void bulkData(String indexName, List<? extends EsData> dataList) throws
OException;
}

public class ESIndexImpl implements ESIndexService {
    private Logger logger = LoggerFactory.getLogger(this.getClass());
    @Autowired
    private RestHighLevelClient client;

    @Override
    public void createIndex(String name, String alias,String type) {
        …… //创建索引相关代码
    }

    @Override
    public void deleteIndex(String name) {
        …… //删除索引相关代码
    }

    @Override
    public boolean isIndexExist(String indexname) throws IOException {
        …… //判断索引是否存在
    }

    @Override
    public void bulkData(String indexName, List<? extends EsData> dataList) throws
OException {
        …… //批量操作相关代码

    }
}
```

图 4.2: 操作Elasticsearch相关方法

如图 4.2所示，EsIndexService接口定义了使用Elasticsearch的一系列方法，包括新增，删除Index，判断某一个Index在Elasticsearch 中是否存在，以及批量的将对象插入到Elasticsearch 中。EsIndexImpl类实现了EsIndexService接口，提供了相关方法的实现。对应于多个数据源的处理方式，在处理批量操作的时候，bulkdata()方法接受的参数中对象的List采取了泛型，只接受所有继承自基类EsData的相关对象组成的List参数。通过Java High Level Rest Client api，将对象批量的导入Elasticsearch中，并处理返回对象BulkItemResponse，所有没有成功完成的操作都将以日志的形式将信息记录下来，方便后期的补录等操作。

4.1.4　处理类与处理工厂

```
    /**
     * 插入 ES 的数据处理基本类，将元数据进行转换处理，包装成 ES 处理类接受的数据
格式并插入到 es 中
     * handleNum：批量处理的个数
     * @author YEJIFAN
     *
     */
    public abstract class EsDataHandler {

        Logger logger;
        int handleNum;
        ESIndexService service;
        public EsDataHandler(int num) {
            handleNum = num;
            this.service = new ESIndexImpl();
        }

        public Integer getNum() {
            return handleNum;
        }

        public void setNum(int num) {
            handleNum = num;
        }

        public abstract void handleData(String... params);
    }
```

图 4.3: Handler基类代码

如图4.3所示，所有的处理类都继承自抽象类EsDataHandler。EsDataHandler中定义了所有Handler需要的属性值，包括记录日志的Logger，负责Elasticsearch相

关操作的接口EsIndexService以及批量操作每次批量处理的条数num。同时，所有的Handler都必须实现抽象类中定义的handleData()方法，该方法用于处理数据来源的数据并调用EsIndexService接口的相关方法完成相关index索引的创建以及数据导入的工作。

```
public class ResumeHandler extends EsDataHandler{
    public ResumeHandler(int num) {
        ……  //类初始化
    }
    @Override
    public void handleData(String... params) {
        //新建进入 ES 的总条目数
        int total = 0;
        String filename = params[0];
        try {
            //通过 csvreader 读取 csv 文件，并储存进对象中
            CsvReader reader =   new CsvReader(filename, ',', Charset.forName("UTF-8"));
            int temp = 0;
            List<ResumeData> list = new ArrayList<>();
            while (reader.readRecord()) {
                ……  //读取 csv 每一行内容并赋值
                ResumeData data = new ResumeData(title, author, journal, year, keywords,
abstracts);
                list.add(data);
                temp++;
                if(temp == handleNum) {
                    buildEs(list);
                    total += handleNum;
                    list.clear();
                }
            }
        } catch (Exception e) {
            logger.info("total:"+total+"========"+e.getMessage());
        }
    }
    //调用接口将数据批量上传至 ES，若因为网络原因失败，则尝试 3 次，无法成功则抛出
异常
    public void buildEs(List<ResumeData> list) throws IOException{
        int n = 3;
        while(n > 0) {
            try {
                service.bulkData("resume", list);
                n = 0;
            } catch (IOException e) {
                if(n == 1) {
                    throw e;
                }
                n--;
            }}}
```

图 4.4: 论文概要信息导入Es相关实现代码

如图4.4所示，图中展示了论文概要信息导入的相关过程。ResumeHandler类继承自EsDataHandler，并实现了handleData()方法。论文的概要信息存储在csv文件中，因此在handleData()方法中，首先使用csvreader的相关api，将csv文件按行读取，每一行数据生成一个ResumeEnity对象，并存放在相应的list中。根据预设的批处理的条目上限num，当处理到条目上限后，将调用EsIndexService中的bulkdata() 方法，将该批数据批量导入到Elasticsearch中，成功后开始接着对文件进行处理。

定义了Handler基类以及相关导入的自定义Handler实现后，通过处理类工厂来实现选择正确的处理类来处理相应的元数据。通过输入的参数，工厂类HandlerFacroty中的getHandler()方法将会返回相应的具体的实现类，从而使得系统可以调用不同的处理类来实现对不同来源的数据进行处理。

4.2　分词训练模块的相关实现

分词训练模块主要包括上传分词文件，调用jieba进行分词并展示分词结果，以及添加自定义分词等功能。分词训练模块主要针对论文标题进行分词，通过选择指定的源文件，系统将调用jieba代码对相应文件中的文字进行分词，并展示分词结果。当用户对分词结果不满意的时候，可以选择通过添加自定义分词的方式，更新jieba中的用户自定义词典，并对文件进行重新分词并继续查看分词效果。

4.2.1　分词训练功能设计

由上述分析可知，分词模块的实现主要通过Python调用jieba代码库进行实现，分词结果存储在本地文件中，在界面显示时进行读取。因此，后端代码实现相对简单，主要包括两个部分：分词部分通过java调用Python脚本，传入文件路径等参数，将结果存储在本地文件中。显示部分则通过读取本地文件，传递至前端进行分页显示。

4.2.2　分词训练模块前端部分代码

分词训练前端部分主要使用了BootStrap框架组件来搭建了相关页面，同时使用了Vue.js框架来完成数据动态更新以及监听动作，前后端之间的数据传递则使用了vue-resource。如图 4.5所示，segment.html界面主要采用css编写，交互功能使用javaScript脚本实现，图中代码展示了切换分词文件的相关操作代码。如图所示，getAllSegment()主要利用了vue-resource中的get方法，从后台获取到分词的相关数据。

```
segSelect:function () {
        console.log("切换了分词文件");
        this.showRealSeg=false;
        console.log("隐藏真正的分词文件"+this.showRealSeg);
        this.segments=[];
        this.segments = [{title:"正在加载分词文件...",segments:["请稍等..."]}];
        $("#reRunButton").attr("disabled",true);
        this.nowSegmentFile=$("#segFileSelect").val();
        console.log("切换文件，新文件请求URL
/seg/getSegments/"+this.nowSegmentFile);
        this.getAllSegments();
        // this.$http.get("/dict/getDict").then(function (response) {
        this.$http.get("/dict/getDict/"+this.nowSegmentFile).then(function (response) {
            if (response.data.length>0){
                this.myselfSegments=response.data;
            }
            else {
                this.myselfSegments=[];
            }
        });
        // toastr.success("加载文件"+this.nowSegmentFile+"成功")
    }
```

图 4.5: 分词部分界面JavaScript代码

如图 4.6展示的是vue部分代码的相关功能。该段代码用于鼠标选中对输入框进行自动填充。

changeInput()方法用于获得鼠标选中的内容，v-on是vue中的相关语法，用于绑定相关事件，在该段代码中，绑定了鼠标点击的相关事件，用户在鼠标点击后将会调用changeInput()方法。

v-show用于控制元素的显示能够根据条件来在界面上进行相关展示，在代码中将根据showRealSeg的值来决定是否显示。

v-for是循环语句，对同一类型的数据可以做到以相同的格式进行呈现。

```
<div class="row pre-scrollable" id="showSegScroll" v-on:MouseUp="changeInput"
v-show="showRealSeg">
    //循环
    <div v-for="segment in segments" id="segShow">
        <div id="eachSeg">
            <div class="row">
                <div class="col-xs-1"></div>
                <div class="col-xs-8 segTitle">
                    <label>题目：{{segment.title}}</label>
                </div>
                <div class="col-xs-3">
                    <button class="btn btn-default btnSeg"
                    v-on:click="chooseAddSeg(segment)">
                        //修改分词
                    </button>
                </div>
            </div>
            <div class="row">
                <div class="col-xs-1"></div>
                <div class="col-xs-9 segParts">
                    <span v-for="eachS in segment.segments">
                        {{eachS}}/
                    </span>
                </div>
            </div>
            <div class="col-xs-1"></div>
        </div>
    </div>
</div>
```

图 4.6: vue部分语法展示

如图 4.7展示的是分词训练模块的最终界面。左侧部分是分词训练的界面。通过选择相关的分词文件，点击ReRun按钮，系统将对该文件进行分词，并将分析结果展示在界面上。其中，上行加粗部分是被分词的论文的原标题，如文物文化遗产之术语辨析，下方则是对该标题的分词结果展示，如前一个标题被分成了文物/文化遗产/术语/辨析。工作人员通过对分析结果的观察，可以通过添加或删除自定义分词的方式，对分词结果进行人工干预。因此在界面右侧的窗口，主要提供了该项功能。工作人员可以通过创建自定义分词按钮，来对自定义词典进行增加或者删除。

图 4.7: 分词训练主界面展示

如图 4.8展示的是添加自定义分词的界面。当工作人员点击添加自定义词典，系统将展示如下界面。工作人员可以选择单独添加某个分词，也可以选择批量添加一组分词。

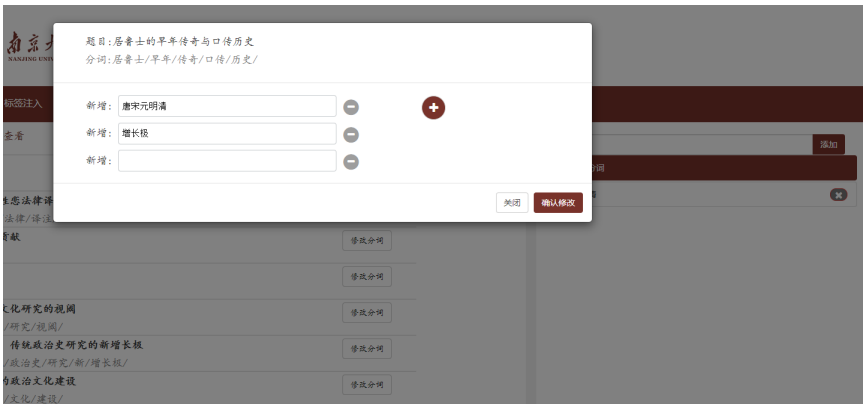

图 4.8: 分词训练批量添加分词展示

4.2.3　分词训练模块后端部分代码

如图 4.9所示，分词训练后端代码主要通过jieba分词来实现。首先从文件系统中读取用户自定义词典，停用词词典以及根据传入的待分词分拣的文件路径读取的相关分词文件，调用jieba分词的相关api进行分词，并将分词结果储存在本地文件中。

```
stop_word_file    = open('E:/finalDesign/citation_lda/references/stopwords.txt'
,'r',encoding='utf-8')
stopwords = stop_word_file.read().splitlines()

argv_num = len(sys.argv)
raw_file_path = sys.argv[1]
raw_file = open(raw_file_path, 'r', encoding='utf-8')
raw_file_content = raw_file.read().splitlines()
titles = []
for each in raw_file_content:
    titles.append(each.split(' ')[0])
raw_file.close()

dict_file_path = sys.argv[2]
jieba.load_userdict(dict_file_path)

# dict_file = open(dict_file_path, 'w', encoding='utf-8')
# dict_file.close()

seg_file = open(raw_file_path, 'w', encoding='utf-8')

try:
    for title in titles:
        title = re.sub("[\s+\.\!\/_,$%^*()+\"\']+|[+——！，。？、~@#￥%……&*（）《》·“”—]",
"", title)
        seg_file.write(title + ' ')
        split = list(jieba.cut(title,cut_all=False))
        for each_split in split:
            if each_split in stopwords:
                split.remove(each_split)
        seg_file.write(','.join(split))
        seg_file.write('\n')
    seg_file.close()
except:
    print('Error')
print('Complete')
```

图 4.9: 分词训练后端部分代码

4.3　数据统计分析图表的相关实现

数据统计模块主要分为三个部分：数据读取，数据分析以及数据展示。其中，数据展示部分采用了大量的图表进行展示，使得分析结果一目了然。为了实现相关的图表，前端部分主要使用了Echarts图表，对相关数据进行绘制，包括折线图，气泡图等。后端部分主要包括文件的读取以及相关处理，本节将主

要展示图标的绘制部分。

4.3.1　Echarts绘制气泡图的相关实现

```
statisticsVue.generalShowChart.setOption({
    series:[
        {name:'keyword',
            type:'graph',
            //设置图类型为力导向图
            layout: 'force',
            data:authorArry,
            roam:true,
            symbolSize: function (data) {
                //根据出现频率设置对应气泡的大小
                return Math.round(15 + data * 80 / authorNum[0]);
            },
            label: {
                normal: {
                    show: true
                },
                emphasis: {
                    show: true,
                    formatter: function (param) {
                        return param.name;
                    },
                    color: 'black'
                }
            },
            force:{
                //斥力，设置节点之间的距离
                repulsion: 65
            }
        },
    ]
})
```

图 4.10: 词频气泡图相关实现代码

如图 4.10所示，由于Echarts中并没有气泡图的实现，因此可以通过力导向图来进行模拟。力导向图又称关系图，主要是由点以及点之间的关系（边）组成。而为了模仿气泡图，可以选择删去边的属性，而只保留节点。将节点绘制成不同的大小来模拟相应的节点出现的频率。

如 4.11显示了气泡图的实际效果，气泡越大的代表该词出现频率最高。为了避免绘制卡顿，只选取前70个词语进行展示。

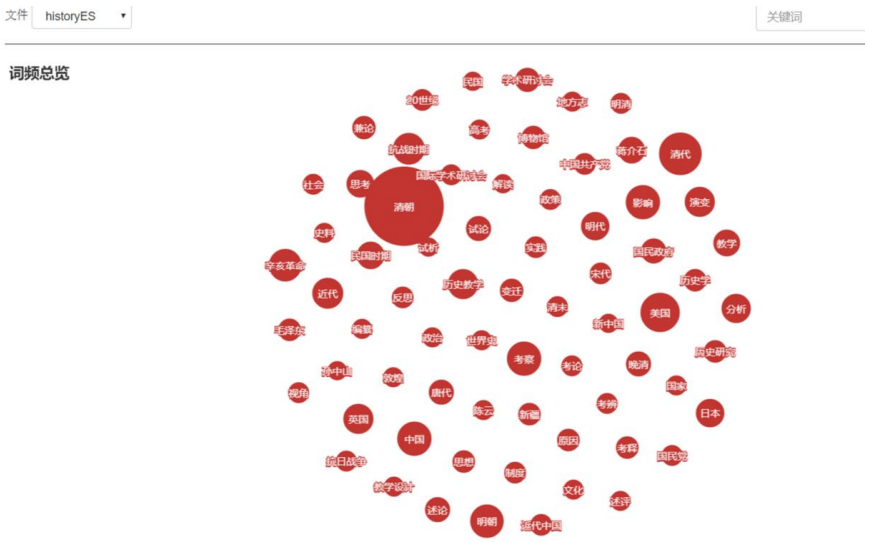

图 4.11: 气泡图效果演示

4.3.2　Echarts绘制直线图相关代码

```
this.venueKeyShowChart.setOption({
    title: {
        text: '期刊词频分布:'+this.nowKeyWord
    },
    tooltip: {
        trigger: 'item'
    },
    grid: {
        //设置相关参数
        left: '3%',
        right: '4%',
        bottom: '3%',
        containLabel: true
     },
     xAxis: {
          data: []
     },
     yAxis: {},
     series: [{
          //绘制折线图
          type: 'line',
     }],
})
```

图 4.12: 折线图相关实现代码

如图 4.12所示，在series中声明type为line，并设定相关属性，通过Echarts可以方便的绘制出折线图。

折线图效果如图 4.13所示。

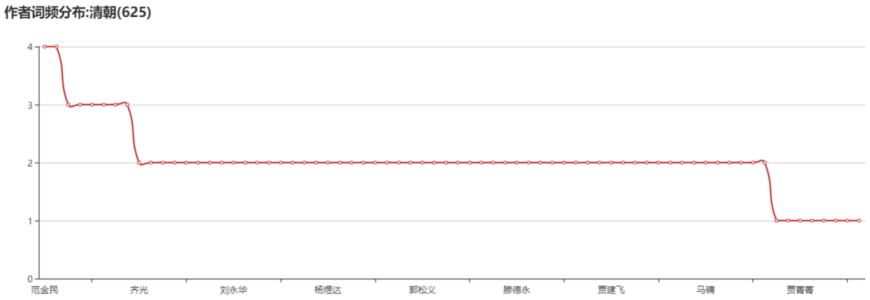

图 4.13: 折线图效果演示

4.4　关联作者分析功能的相关实现

关联作者分析主要包括作者概要信息展示以及作者详细信息展示两部分。由于作者信息包括发表的论文，研究的主题等相关内容，而研究主题的相关内容又依赖于主题研究的相关内容，主题研究通过BERT模型以及LDA模型运算得到。因为模型运算需要大量的时间以及资源，因此不能在实时运算中进行。因此，作者关联分析与模型运算一起，独立运算完毕后，通过Elasticsearch导入数据的功能，将运算结果储存在Elasticsearch中，在使用时通过查询Elasticsearch将数据导出并进行包装处理传递到前端进行显示。

4.4.1　关联作者分析功能设计

由第三章系统需求分析与概要设计分析可知，关联作者分析是系统的核心功能，不仅包括对原始数据的分析整合，还通过主题分析的结果对作者相关信息进行进一步的挖掘，最终得到完整的作者信息。因此，关联作者分析功能实现主要包括两大部分：作者信息整理以及作者信息展示。作者信息整理部分代码负责对原始数据文件处理，并通过调用主题分析工具类提供的相关接口，获得相关论文主题的信息，经过整合，生成作者对象，调用导入Elasticsearch数据模块，将作者信息批量存储在Elasticsearch中。作者信息展示部分负责将Elasticsearch中的作者信息根据指定参数进行提取，封装，并传递给前端进行展示。与导入功能实现类似，为了实现代码的复用，将Elasticsearch查询部分代码封装成为工具类进行调用。以下是相关代码核心部分的展示。

4.4.2　作者信息整理部分代码

作者信息主要来源于两个部分：初始论文信息的CSV文件，以及模型运算的结果。初始文件中存储了作者发表论文的基本信息，之后通过调用模型运算的结果，得到所有论文的主题以及该篇论文在该主题下的排名。对于排名越高

的论文，可以认为这篇论文价值越高。从而可以得出该作者发表的最有研究价值的论文以及该论文所属的成就最高的研究主题。

```
//通过模型处理结果文件以及文章标题，获得文章所属的所有主题。
//根据排名决定最有价值的文章与主题，并进行赋值
public void setOthers(List<String> papers){
    //调用模型运算接口获得论文以及该论文在所属主题中的排名
    Map<String, ThemeRanking> paperRanking = ModelUtil.getPaperRanking(papers);
    Set<String> themeSet = new HashSet<>();
    for(ThemeRanking themeRanking : paperRanking.values()) {
        themeSet.add(themeRanking.getTheme());
    }
    //通过 HashSet 去重后获得主题列表
    this.themes = new ArrayList<String>(themeSet);
    //通过 entrySet 转成 List 后使用排序获得排名最高的论文及主题
    List<Map.Entry<String, ThemeRanking>> list = new ArrayList<Map.Entry<String,
ThemeRanking>>(paperRanking.entrySet());
    list.sort(new Comparator<Map.Entry<String, ThemeRanking>>() {
        @Override
        public int compare(Entry<String, ThemeRanking> o1, Entry<String, ThemeRanking>
o2) {
        return o1.getValue().compareTo(o2.getValue());
        }
     });
     this.mostPaper = list.get(0).getKey();
     this.mostTheme = list.get(0).getValue().getTheme();
}
//一篇文章所属的主题以及该篇文章在该主题下的排名，实现了 Comparable 接口
public class ThemeRanking implements Comparable<ThemeRanking>{
    String theme;
    int ranking;

    public ThemeRanking(String theme, int ranking) {
        ……  //构造函数
    }
    //排名较小的说明排名靠前
    @Override
    public int compareTo(ThemeRanking o) {
        if(ranking > o.ranking) {
            return -1;
        }
        if(ranking < o.ranking) {
            return 1;
        }
        return 0;
    }
}
```

图 4.14: 作者信息整理部分代码

如图 4.14所示，首先，处理论文概要信息的csv源文件，整理出每个作者名下的所有论文信息，通过对初始论文信息进行处理，按照作者将论文进行归类，可以得到所有作者的所有著作的列表。然后，通过与主题模型运算进行结合，通过对该作者名下的所有论文进行主题的运算并根据重要度进行排序，可以得到该作者所有研究的主题以及成就最高的主题等相关信息。定义ThemeRanking类，将主题与相应排名形成一个整体，该类实现Comparable接口，排名越靠前的ThemeRanking值越大，从而可以实现对该作者名下的所有论文根据在相关主题下的排名进行排序，进而可以筛选出该作者最有价值的论文以及最有成就的研究主题。将信息进行整合后，通过作者类Author将信息封装起来，并通过AuthourHandler处理类，将作者信息通过调用导入Elasticsearch的功能代码，存储在Elasticsearch中。

4.4.3　作者信息相关查询代码

由于主题、作者等相关分析速度较慢，故采取先进行运算，将运算结果储存在持久化的Elasticsearch中，在展示时快速取得数据，保证系统的响应速度。作者信息展示部分，主要通过查询Elasticsearch来获取数据，进行包装后传递给前端进行展示。由于主题分析结果，作者分析结果等包含大量的查询代码，为了减少重复工作，将查询部分代码进行封装，成为工具类，在查询时通过指定索引名以及语句进行相关查询。当用户点击作者页面后，后端获得作者的姓名等相关信息，通过SearchAuthor方法，从Elasticsearch中查询相关作者的信息。通过调用Java High Level Rest Client相关查询api，通过对返回结果进行解析，重新组装成Author类，并传递给前端进行相关信息的展示。

如图 4.15所示，查询关联作者方法通过组装bool语句，并调用查询工具类Tools中Tools.getSearchHit()方法，将作者信息从Elasticsearch中导出并进行相关处理。下方则给出了getSearchHit()方法的具体实现。通过Java High Level Rest Client中的api的要求，封装SearchRequest，调用api并返回相应的response。

```
//根据索引名字以及作者名查询相关联作者
public ArrayList<StatisticsInfo> getRelativeAuthorByAuthor(String indexName, String
authorName) {
    //组装查询语句
    BoolQueryBuilder boolBuilder = QueryBuilders.boolQuery();
    //使用 must 语句进行查询
    boolBuilder.must(QueryBuilders.matchQuery("ZZMC.keyword", authorName));
    SearchHits hits = tools.getSearchHits(indexName+"authors",boolBuilder,0,20000);
    SearchHit[] searchHits = hits.getHits();
    ArrayList<String> snos=new ArrayList<>();
    for (SearchHit s:searchHits){
        ..... //对查询数据进行组装
    }

    return relativeAuthors;
}

//在 ES 中通过索引名字以及相应的语句，获得查询结果
public SearchHits getSearchHits(String indexName, BoolQueryBuilder boolBuilder){
    logger.error("在 getSearchHits(String indexName, BoolQueryBuilder boolBuilder)中寻找
index 为"+indexName+"的数据");
    SearchSourceBuilder sourceBuilder = new SearchSourceBuilder();
    //查询 bool 查询
    sourceBuilder.query(boolBuilder);
    SearchRequest searchRequest = new SearchRequest(indexName);
    searchRequest.source(sourceBuilder);
    //获得 ES 返回体 response
    SearchResponse response=new SearchResponse();
    try {
        response = client.search(searchRequest, RequestOptions.DEFAULT);
    }catch (IOException e){
        e.printStackTrace();
    }

    SearchHits hits = response.getHits();
    return hits;
}
```

图 4.15: 作者信息查询代码

4.5　系统测试

本小节主要是对本系统的相关功能进行功能测试以及对系统的相关非功能性需求进行测试。

功能测试方面，考虑到时间成本以及同时测试界面交互功能，选择通过交互的方式对相关功能能进行测试，主要包括数据导入功能，分词训练功能，关

联作者信息查询的相关功能。

对非功能性测试，主要对系统的性能进行了性能测试，在设计时，对系统的响应速度，相关错误保护机制，出错恢复提出了相关要求，着重对上述非功能性需求进行了测试。

4.5.1　数据导入功能测试

如表 4.1所示，数据导入功能主要从数据处理以及数据导入两方面功能点进行测试。经测试，所有的测试结果均与预期相符，测试通过。

表 4.1: 数据导入功能测试用例

测试项	操作/输入	预期结果	测试结果
数据处理	1.准备测试用少量数据 2.通过java -jar ****.jar **** EsConfig 命令指定运行导入类代码 3.查看日志观察是否有报错	日志无处理错误相关的报错信息，导入处理顺利完成	程序无报错顺利运行完毕
数据导入	1.登入Elasticsearch管理界面 2.查看数据导入格式是否正确 3.查看数据量是否正常	Elasticsearch中数据符合设计，且数据量与准备的数据量一致	界面中查看的数据量与测试数据量相同，导入成功

4.5.2　分词训练功能测试

如表 4.2所示，分词训练功能主要从选择文件分词，添加自定义分词两方面功能点进行测试。经测试，所有测试结果均与预期相符，测试通过。

表 4.2: 分词训练功能测试用例

测试项	操作/输入	预期结果	测试结果
选择文件分词	1.选择需要分词的文件 2.点击分词按钮 3.查看日志以及分词结果	日志无报错信息，分词结果展示正常	能够正常展示分词结果，并且日志无报错信息
添加自定义分词	1.在输入框输入需要添加的分词 2.点击添加分词按钮 3.点击重新分词按钮 4.观察分词结果是否有变化 5.查看相关分词日志	日志无报错信息，分词结果根据自定义的分词发生了改变	重新分词结果因为添加了自定义分词而发生了改变

4.5.3　关联作者信息查询功能测试

如表 4.3所示，关联作者信息查询功能主要从作者概要信息展示，详细信息展示两方面功能点进行测试。经测试，所有测试结果均与预期相符，测试通过。

表 4.3: 关联作者信息查询功能测试用例

测试项	操作/输入	预期结果	测试结果
作者概要信息展示	1.点击用户按钮 2.查看页面跳转是否正常 3.查看跳转后的概要信息展示界面中信息是否正确	界面跳转正常，作者概要信息展示正确	界面正常跳转至概要信息展示页面，展示的信息与元数据一致
作者详细信息展示	1.点击某一年份进入该年份详细信息展示 2.查看页面跳转是否正常 3.查看详细信息展示是否正确	界面跳转正常，作者详细信息统计正确	界面跳转正常，跳转至某一年份的详细信息展示，展示的信息正确

4.5.4　非功能性需求测试

非功能性需求测试主要针对系统的性能，可靠性，安全性等相关方面进行测试，如系统的错误日志记录是否完善，系统在大数据量访问的过程中是否存在卡顿，加载缓慢的现象，在系统发生意外停止后能否在短时间内自动恢复，系统对网络错误等错误是否有相应的保护措施等。通过非功能性需求测试，能够保证系统能够稳定地提供相关服务，给用户良好的用户体验。以下将介绍系统性能及可靠性的相关测试用例。

如表 4.4所示，对系统的相关性能进行了测试。测试方式通过点击展示按钮，查看展示是否迅速，有无卡顿现象发生。对于设计了分页展示，延迟展示的相关模块，如分词模块，图表绘制相关功能，采用了大数据量进行了测试，查看延迟展示，分页展示等策略是否符合预期，所有界面能否在3s内予以响应。测试结果基本符合预期。

表 4.4: 系统性能测试用例

测试功能点	系统性能
测试级别	非功能测试
测试目的	查看系统对指令的响应速度是否符合需求，界面是否能在3s内加载出来
测试流程	对系统各个展示界面进行点击测试，特别对分词展示相关部分，上传较大的分词文件，查看结果展示速度
预期结果	系统展示正常，所有结果均能在3s内进行展示
实际结果	系统能够顺利对相关信息进行展示，在展示内容较多时，能够进行延迟展示，无卡顿现象
测试结果分析	测试结果符合预期

如表 4.5所示，对系统的可靠性进行了相关测试。在系统发生意外宕机，系统崩溃等情况时，系统应保证在5分钟内进行自动重启，从而保证能够正常的提供服务。通过模拟系统意外停止来对系统可靠性进行测试，经测试，系统在停止后5分钟内自动重启，恢复了服务，测试结果与预期基本相符。

表 4.5: 系统可靠性测试用例

测试功能点	系统可靠性
测试级别	非功能测试
测试目的	查看系统在意外停止的情况下能否自动重启
测试流程	通过kill命令将系统进程停止，观察系统是否能够恢复
预期结果	系统在5分钟内重启进程，恢复服务
实际结果	系统在预计时间内成功重启
测试结果分析	测试结果符合预期

4.6　本章小结

本章主要介绍了从数据导入功能，分词训练功能，关联作者信息分析功能三个方面来介绍了系统的主要功能的部分实现以及相关功能的界面展示。在数据导入方面，主要介绍了在方便代码进行扩展方面做出的相关设计与其具体的实现，并给出了操作Elasticsearch以及文章摘要信息导入的实现范例；在分词训练方面，主要介绍了分词训练模块前端到后端的实现过程，给出了主要运用的技术如vue，JavaScript的相关具体代码实现；在数据统计功能方面，主要展示了前端在页面展示上使用Echarts绘制图表的相关实现；在关联作者信息功能实现方面，主要介绍了作者信息整理，储存的相关方法，并且展示了与模型训练结果相结合而得出更深一步的结果的具体实现细节以及从Elasticsearch 方面根据

关键词得到作者详细信息的实现代码。最后，介绍了系统的相关功能测试以及非功能性测试的结果。

第五章　总结与展望

5.1　总结

本文对我国目前社科论文研究现状进行了分析，在论文水平日益发展，论文库愈加庞大的前提下，结合近些年高速发展的数据挖掘技术以及机器学习技术，提出一种自动化的方式，即通过Citation-LDA模型以及BERT模型运算与数据分析方法相结合，对论文进行归纳整理，提取论文的主题，进而更深层次的挖掘出论文相关的信息，比如主题的流变，作者分析等。基于该思想，设计开发了中文社科论文分析系统。

本文所设计实现的中文社科论文分析系统主要完成了以下工作：

- 数据处理方面，主要完成了对PDF源文件的清洗处理，整理出系统所需要的论文信息，如标题，正文，作者等相关内容，并对信息进行整合，处理，转换为模型运算以及数据分析所需要的数据格式。系统将Elasticsearch作为储存工具，将处理完成的数据批量导入至Elasticsearch中。

- 模型方面，主要针对数据集，运用了Citation-LDA（Citation Latent Dirichlet Allocation）以及BERT（Bidirectional Encoder Representation from Transformers）两种模型对数据集进行处理。Citation-LDA模型通过对引文的计算，可以得到主题的流变，里程碑论文等相关信息。而BERT模型将论文转化为统一的词向量，再根据聚类算法与主题提取算法，可以得到所有论文的所有主题与主题与论文的对应关系。二者相辅相成，使得结果更加的准确与详细。

- 系统的后端部分，系统是一个web项目，主要采用SpringBoot框架进行快速的搭建，减少了繁杂的配置，兼容性较高。

- 系统的前端部分，主要采用了BootStrap框架与Vue.js框架，在界面风格上参考了中国社会科学评价研究中心的相关网站。同时根据大数据量等特点，采用分页，延时加载等操作，确保所有信息在3s内显示给用户。

系统开发工作目前已经基本完成，相关的功能设计也已经基本实现，运行的效果也与预期比较相符。本人在项目中主要负责了数据导入，关联作者分析，分词训练等相关后台代码的实现以及系统模块的前端部分实现。

5.2　工作展望

目前来说，中文社科分析系统设计的功能点已经基本实现，相关的数据分析结果也基本符合预期。但是仍然有地方需要改进。

首先，对于系统功能方面，仍然有很多地方可以扩充。如在论文展示页面，可以根据主题内容以及主题相关流变方向，做到与主题相关的论文，作者的推荐功能，在分词训练功能方面，可以根据某一类别的论文，通过分析已有的相关论文的关键词等信息，自动生成分词库，使得初始的分词更加符合研究人员预期，降低研究人员手动添加分词的工作量。在系统性能方面，目前在数据量变大的情况下，模型运算过程中可能有内存溢出的情况发生，需要进一步优化。目前系统只是对社科论文进行分析，在未来可能会扩展到更多的学科，在数据量大量增加的情况下，系统可能需要转变为分布式的部署方式，以加快处理响应的速度。

参考文献

[1] 鄂丽君, 从论文产出角度看图情国家社科基金项目研究现状, 情报科学 (08) 68–72+88.

[2] 张引, 陈敏, 廖小飞, 大数据应用的现状与展望, 计算机研究与发展50 (s2) (2013) 216–233.

[3] 王灿辉, 张敏, 马少平, 自然语言处理在信息检索中的应用综述, 中文信息学报.

[4] 朱军, 胡文波, 贝叶斯机器学习前沿进展综述, 计算机研究与发展52 (1) (2015) 16–26.

[5] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, Journal of Machine Learning Research 3 (2003) 993–1022.
URL http://dl.acm.org/citation.cfm?id=944919.944937

[6] X. Wang, C. Zhai, D. Roth, Understanding evolution of research themes: a probabilistic generative model for citations, in: I. S. Dhillon, Y. Koren, R. Ghani, T. E. Senator, P. Bradley, R. Parekh, J. He, R. L. Grossman, R. Uthurusamy (Eds.), The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013, ACM, 2013, pp. 1115–1123.
URL https://doi.org/10.1145/2487575.2487698

[7] X. Rong, word2vec parameter learning explained, cite arxiv:1411.2738 (2014).
URL http://arxiv.org/abs/1411.2738

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, p. 5998 – 6008.
URL https://papers.nips.cc/paper/7181-attention-is-all-you-need

[9] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, cite arxiv:1810.04805 (2018).
URL http://arxiv.org/abs/1810.04805

[10] Y. Cui, W. Che, T. Liu, B. Qin, Z. Yang, S. Wang, G. Hu, Pre-training with whole word masking for chinese BERT, CoRR abs/1906.08101.
URL http://arxiv.org/abs/1906.08101

[11] V. V. Das (Ed.), An Iterative Improved k-means Clustering, Vol. 2 of 3, Institute of Doctors Engineers and scientist(IDES), The Association of Copmuter Electronics and Electrical Engineers, 1133 Broadway,Suite 706,New York,NY10010,USA, 2011.
URL /brokenurl#doi.searchdl.org/01.IJNS.02.03.183

[12] T. Kim, S. Park, S. Yang, Improving prediction quality in collaborative filtering based on clustering, in: 2008 IEEE / WIC / ACM International Conference on Web Intelligence, WI 2008, 9-12 December 2008, Sydney, NSW, Australia, Main Conference Proceedings, IEEE Computer Society, 2008, pp. 704–710.
URL https://doi.org/10.1109/WIIAT.2008.319

[13] G. Gan, J. Wu, Z. Yang, A genetic fuzzy k-modes algorithm for clustering categorical data, Expert Syst. Appl. 36 (2) (2009) 1615–1620.
URL https://doi.org/10.1016/j.eswa.2007.11.045

[14] G. Gan, Z. Yang, J. Wu, A genetic *k*-modes algorithm for clustering categorical data, in: X. Li, S. Wang, Z. Y. Dong (Eds.), Advanced Data Mining and Applications, First International Conference, ADMA 2005, Wuhan, China, July 22-24, 2005, Proceedings, Vol. 3584 of Lecture Notes in Computer Science, Springer, 2005, pp. 195–202.
URL https://doi.org/10.1007/11527503_23

[15] 黄昌宁, 赵海, 中文分词十年回顾, 中文信息学报.

[16] 我国文献计量学发展的回顾与展望, 科学学研究 (2) 32–37.

[17] 刘启元, 叶鹰, 文献题录信息挖掘技术方法及其软件sati的实现——以中外图书情报学为例, 信息资源管理学报 (1) (2012) 50–58.

[18] 徐戈, 王厚峰, 自然语言处理中主题模型的发展, 计算机学报34 (8) (2011) 1423–1436.

[19] C. Chen, Citespace ii: Detecting and visualizing emerging trends and transient patterns in scientific literature, Journal of the Association for Information Science and Technology 57 (3) (2006) 359–377.

[20] 马楠, 官建成, 利用引文分析方法识别研究前沿的进展与展望perspectives on research fronts identification based on citation analysis, 中国科技论坛000 (4) 110–113,128.

[21] 冯璐, 冷伏海, 共词分析方法理论进展co-word analysis, 中国图书馆学报032 (2) 88–92.

[22] 李纪, 李莘, 基于sati及citespace的学科服务研究知识图谱对比分析, 兰台世界No.487 (29) 140–142.

[23] 张峰, 应用springboot改变web应用开发模式, 科技创新与应用000 (23) 193–194.

[24] elasticsearch, elasticsearch/elasticsearch (2015).
URL https://github.com/elasticsearch/elasticsearch

[25] 王占宏, 王战英, 分布式弹性搜索研究与实践, 微型电脑应用030 (007) 9–12.

[26] 祝永志, 荆静, 基于python语言的中文分词技术的研究, 通信技术.

[27] J. Song, M. Zhang, H. Xie, Design and implementation of a vue.js-based college teaching system, iJET 14 (13) (2019) 59–69.
URL https://www.online-journals.org/index.php/i-jet/article/view/10709

[28] 麦冬, 陈涛, 梁宗湾, 轻量级响应式框架vue.js应用分析, 信息与电脑(理论版) 377 (7) 64–65.

[29] A. C. Davison, D. V. Hinkley, G. A. Young, Recent developments in bootstrap methodology, Statistical Science 18 (2) (2003) pp. 141–157.
URL http://www.jstor.org/stable/3182844

[30] D. Li, H. Mei, Y. Shen, S. Su, W. Zhang, J. Wang, M. Zu, W. Chen, Echarts: A declarative framework for rapid construction of web-based visualization., Vis. Informatics 2 (2) (2018) 136–146.
URL http://dblp.uni-trier.de/db/journals/vi/vi2.html#LiMSSZWZC18

[31] 骆文亮, 绘图插件highcharts浅析, 科技视界 (12).

致　　谢

首先要感谢我的导师郑滔教授，郑老师在我研究生的生涯中为我提供了巨大的帮助，使我获益良多。在项目开始阶段，郑老师指导我们查询相关领域的研究成果，确定了研究方向，并且培养了我们阅读论文，独立思考的能力。在项目中期做出基本成果时，郑老师与我们认真研究项目目前仍然存在的不足，并提出接下来的研究思路，并为我们介绍了相关领域的专家为我们提供了帮住。在论文的撰写阶段，郑老师悉心指导，指出我论文撰写方面存在的缺陷，提供了宝贵的建议。

其次，我要感谢我们项目组的同学们。大家一起完成了项目的开发工作，一起寻找更好的方法实现系统。在遇到问题时，大家一起思考，讨论，最终得到令人满意的结果。

最后，我要感谢我的父母。正是你们的支持，让我能够心无旁骛的在学校中进行学习，研究，在研究生生涯中学习到了宝贵的知识，在学习生活中取得了满意的成果。同时，感谢学院的每一位老师，传授给我大量的科学知识以及人生经验，在我遇到困难时，主动提供帮助，为我的人生道路打下了坚实的基础。

版权与原创性说明

任何收存和保管本论文的单位和个人，未经作者本人授权，不得将本论文转借他人并复印、抄录、拍照或以任何方式传播，否则，引起有碍作者著作权益的问题，将可能承担法律责任。

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含其他个人或集体已经发表或撰写的作品成果。本文所引用的重要文献，均已在文中以明确方式标明。本声明的法律结果由本人承担。

作者签名：

日期: 2020 年 5 月 28 日