分 类 号 学号

学校代码 10487 密级



硕士学位论文

基于循环分段残差网络的对抗关系抽取研究与实现

学位申请人：赵恢强

学科专业：信息与通信工程

指导教师：王玉明 副教授

答辩日期：2019年5月25日

A Dissertation Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Engineering

Design and Implementation of Adversarial Relation Extraction Based on Recurrent Piecewise Residual Networks

Candidate： Huiqiang Zhao

Major： Information and Communication Engineering

Supervisor：Prof. Wang Yuming

**Huazhong University of Science and Technology**

**Wuhan, Hubei 430074, P. R. China**

**May, 2019**

**独创性声明**

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除文中已经标明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期： 年 月 日

**学位论文版权使用授权书**

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权华中科技大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保 密□，在\_\_\_\_\_年解密后适用本授权书。

不保密□。

（请在以上方框内打“√”）

学位论文作者签名： 指导教师签名：

日期： 年 月 日 日期： 年 月 日

摘 要

作为信息抽取的核心任务，有效的关系抽取是信息爆炸时代的一大挑战。本文在进行关系抽取这一基础性研究时发现，现有的模型都无法有效的抽取长而复杂或无直接关系触发词的句子中的实体对关系，而这种句子在自然语言文本中是普遍存在的，这使得关系抽取的效果遇到了瓶颈。

本文深入调研并分析了国内外在关系抽取方面的研究。在融合更多更深次信息的思想启发下，为了丰富模型的表示，提出**实体类型嵌入(Entity Type Embedding, ETE)**，ETE和词嵌入、位置嵌入一起作为模型的联合嵌入层；为了更精细地编码模型的嵌入表示，精心设计了**循环分段残差网络(Recurrent Piecewise Residual Networks, RPRN)**结构，RPRN能有效获取句子上下文中的潜在表示。本文在进行关系抽取建模时还引入了对抗训练，从而进一步增强模型的泛化能力。本文提出的模型有效解决了现存模型所遇到的困境，并在纽约时报公开数据集上进行了验证，实验结果显示本文提出的胜过现存最先进的模型。

关系抽取不仅具有很高的理论研究价值，而且具有非常广泛的实际应用。本文将提出的模型应用到了科研文档中，并应用知识图谱可视化技术构建了一个语义检索系统。通过机器学习相关技术自动抽取自然语言文本语料中蕴含的关系事实，人们可以更加有效地获取文本中涉及的各主客体之间的语义关系，避免了以人工阅读的方式从不断生成的海量文本中提取关系事实等结构信息，这对文本语料的信息化建设具有重大的意义。同时抽取的关系事实是构建语义检索、问答系统等应用的重要基础。

**关键词：**关系抽取；实体类型嵌入；循环神经网络；残差网络

Abstract

As a core mission in information extraction, effective relation extraction is a great challenge in the information explosion era. In this paper, we found that existing models usually fail in extracting correct relation of entity pair effectively when the sentence is long and complex or there is not the firsthand trigger word of corresponding relation, whereas such sentences are ubiquitous in texts formed of natural language, which makes the effects of relation extraction hit a bottleneck.

In this paper, we penetrate into investigating and anlyzing the researches on relation extraction at home and abroad. Inspired by the idea of fusing more and deeper information, we propose **Entity Type Embedding(ETE)** to enrich the representation of the model, ETE, word embedding and position embedding serve as the joint embedding of our model; For encoding the representation of the embedding subtly, we devise an architecture of **Recurrent Piecewise Residual Networks(RPRN)** meticulously, which may acquire the latent representation underlying the context of a sentence. We introduce adversrial training when modeling relation extraction as well, which further enhance generalization capacity of our model. Our proposal effectively solve the dilemma of existing models, and is validated by experiments on public dataset of New York Times. Experiment results show that our proposal outperforms the existing state-of-the-art models.

Relation extraction is not only of great values for theory study, but also has extremely extensive application. We apply proposed model to documents of scientific research, and employ visuallization technology of knowledge graph to build a system of semantic search. Extracting the relation facts implied in the texts formed of natural language automatically by the technology of machine learning, pepole are able to acquire semantic relations between the subjects and objects involved in the text more effectively, and get rid of extracting structural information such as relation facts from the massive texts constantly generated by means of artificial reading, which is of great significance to information construction of text corpus. Simutaneously, the extracted relation facts are key foundation of building applications such as semantic search, question answering system and etc.

**Keywords**：Relation Extraction, Entity Type Embedding, Recurrent Neural Networks, Residual Networks

# 目 录

[摘 要 I](#_Toc480896054)

[Abstract II](#_Toc480896055)

[1 绪论](#_Toc480896057)

[1.1 课题背景及来源 (1)](#_Toc480896058)

[1.2 课题现状及分析 (2)](#_Toc480896059)

[1.3 研究内容及意义 (5)](#_Toc480896060)

[1.4 本文的组织结构 (6)](#_Toc480896061)

[2 相关理论及技术概述](#_Toc480896062)

[2.1 图论介绍 (7)](#_Toc480896063)

[2.2 SSH框架 (13)](#_Toc480896064)

[2.3 面向服务架构 (14)](#_Toc480896065)

[2.4 数据可视化 (15)](#_Toc480896066)

[2.5 本章小结 (16)](#_Toc480896067)

[3 推荐系统分析与设计](#_Toc480896068)

[3.1 总体需求分析 (17)](#_Toc480896069)

[3.2 总体架构设计 (18)](#_Toc480896070)

[3.3 数据清洗模块 (21)](#_Toc480896071)

[3.4 协作网络建模 (25)](#_Toc480896072)

[3.5 协作推荐模型 (29)](#_Toc480896073)

[3.6 本章小结 (37)](#_Toc480896074)

[4 推荐系统实现](#_Toc480896075)

[4.1 开发环境搭建 (38)](#_Toc480896076)

[4.2 数据清洗模块实现 (39)](#_Toc480896077)

[4.3 协作网络建模实现 (46)](#_Toc480896078)

[4.4 协作推荐模型实现 (50)](#_Toc480896079)

[4.5 本章小结 (60)](#_Toc480896080)

[5 系统测试及数据分析](#_Toc480896081)

[5.1 系统测试及性能分析 (61)](#_Toc480896082)

[5.2 协作关系网络探究 (66)](#_Toc480896083)

[5.3 本章小结 (72)](#_Toc480896084)

[6 总结与展望](#_Toc480896085)

[6.1 论文总结 (73)](#_Toc480896086)

[6.2 论文展望 (74)](#_Toc480896087)

[致 谢 (75)](#_Toc480896088)

[参考文献 (76)](#_Toc480896089)

# 略缩语

|  |  |
| --- | --- |
| **ETE** – Entity Type Embedding | 实体类型嵌入 |
| **RPRN** – Recurrent Piecewise Residual Networks | 循环分段残差网络 |
| **NLP** – Natural Language Processing | 自然语言处理 |
| **CNN** – Convolutional Neural Network | 卷积神经网络 |
| **PCA** –Principal Component Analysis | 主成分分析 |
| **SSH –** Struts, Spring, Hibernate | Web应用集成框架 |
| **RPC** – Remote Procedure Call | 远程过程调用 |
| **MVC –** Model View Controller | 一种软件设计模式 |
| **HOG** – Histogram of Oriented Gradient | 方向梯度直方图 |
| **CVPR** – IEEE Conference on Computer Vision and Pattern Recognition | 计算机视觉会议 |
| **GAN –** Generative Adversarial Networks | 生成对抗网络 |
| **CycleGAN –** Cycle-Consistent Adversarial Networks | 循环生成对抗网络 |
| **LSGAN –** Least Squares GANs | 最小方差生成对抗网络 |
| **SRGAN –** Super Resolution GAN | 超分辨率生成对抗网络 |
| **VGG –** Very Deep Cconvolutional Networks | 超深度卷积网络 |
| **AOP –** Aspect Oriented Programming | 面相切面变成 |
| **ORM** –Object Relational Mapping | 对象关系映射 |
| **HTTP –** Hyper Text Transfer Protocol | 超文本传输协议 |
| **SOA** –Service Oriented Architecture | 面相服务架构 |
| **JSON** –JavaScript Object Notation | JS对象标记 |
| **ReLU –** Rectified Linear Units | 一种激活函数 |
| **IDE** –Integrated Development Environment | 集成开发环境 |
| **JDK –** Jave Development Kit | Java开发工具包 |
| **CelebA** – CelebFaces Attributes Dataset | 名人面部数据集 |
| **SDK** – Software Development Kit | 软件开发工具包 |

# 绪论

信息时代的快速发展，一方面使得人们获取信息越来越便利和快捷，另一方面却使人们在海量的信息中获取有效信息变得越来越难。近年来推荐系统技术的发展极大的改善了信息过载问题，使得人们获取的信息更加有针对性、更加有效。目前推荐系统在商业领域的大规模应用取得了良好的效果，但却鲜有在科研合作者推荐方面的应用。随着经济、技术、学科的发展，科研合作变得越来越重要，科研合作者的质量在一定程度上决定了科研成果的质量[[[1]](#endnote-1)][[[2]](#endnote-2)]，如何高效、高质量的找到合适的合作者也成为亟待解决的问题[[[3]](#endnote-3)][[[4]](#endnote-4)]。

随着我国高校科研管理信息化建设的不断推进，积累了丰富的科研管理数据资源，历年积累的科研数据中蕴藏了巨大的价值，如何提高科研数据的利用率和发掘出科研数据的价值是一个值得探索的问题。

本课题尝试发掘已有的科研管理数据资源来解决科研过程中寻求合作者效率和质量低下的问题。本章首先介绍了本课题的背景及来源，并结合国内外具体案例分析了不同推荐系统理论及模型在各场景下的应用和不足之处，分析了本课题的重难点问题。然后对本课题的研究内容及研究意义做了简要总结，最后对本文的组织结构做了简要说明。

## 课题背景及来源

随着Web2.0的兴起，互联网信息交互变得越来越便捷，给用户带来了大量信息的同时也满足了用户在信息时代对信息的需求。但是网络信息的爆炸式增长，使得用户在面对海量网络信息时很难从中找到有效的信息，这就产生了信息过载问题。解决信息过载的有效手段之一是根据用户的需求、行为偏好、兴趣等，将用户感兴趣的信息通过推荐系统推荐给用户。目前推荐系统已经广泛应用于很多领域，尤其是商业领域的应用极大的丰富了电子商务、社交等的形式，但却鲜有在科研合作者推荐方面的应用。

现代科学研究的综合化、交叉化和深入化，使得单人完成科研活动正变得越来越困难。有研究表明：科研人员之间协作分工、共同探讨，能够极大的提升科研效率和科研成果水平[1][2]。目前大多数学者一般是通过参加学术会议、社交、寻找之前合作过的人等手段来寻求潜在的合作者。这种方式局限性很大，效率低下。对于各个领域大量的科研人员来说，如何有效、高质量的找到潜在的科研合作者是一个亟待解决的问题。

本课题的研究受到教育部哲学社科繁荣计划建设项目（2016）、华中科技大学双一流建设项目（2016-2018）、华中科技大学自主创新研究基金项目（No.2015AE027）的资助。本课题通过挖掘科研管理数据资源中科研人员的项目协作关系和研究领域等特征建立推荐模型来对给定的科研人员推荐其潜在的合作者，解决了科研人员寻找合作者效率和质量低下的问题，为提高现有科研管理数据资源的利用率和和发掘其价值提供了新的方向，有利于促进科研人员之间的有效合作者和有效交流，有利于促进学科发展、学科创新和增强科研质量。更进一步的推进了国内社会科学信息化建设，使之更加高效的为国内科研工作者提供信息化服务。

## 课题现状及分析

### 国内外研究概况

对于推荐系统的研究国外起步稍早，以1994年的研究团队推出系统为标志，推荐系统便成为一个相对独立的研究方向[[[5]](#endnote-5)]。该系统第一次提出了基于的思想来完成推荐任务，并为推荐问题建立了一个形式化的模型，该思想和模型引领了推荐系统在今后数十年的发展方向[[[6]](#endnote-6)]。

在推荐系统相关理论的发展和完善中，相继提出了基于内容的推荐[[[7]](#endnote-7)][[[8]](#endnote-8)]、基于物品协同过滤的推荐[[[9]](#endnote-9)]、基于用户协同过滤的推荐[[[10]](#endnote-10)]、基于人口统计学的推荐[8]和基于关系网络的推荐[[[11]](#endnote-11)][[[12]](#endnote-12)]等一系列推荐模型，极大的丰富了推荐系统的理论研究和拓展了推荐系统的应用范围。目前推荐系统的相关理论已经成功的应用到了商用推荐系统中，一般商用推荐系统是多个推荐系统模型和理论进行混合推荐的，下面结合国内外不同的推荐系统模型和商用案例做以简要介绍和分析。

1. 电商推荐系统

国内外知名的电商网站是推荐系统理论应用的高地，同时也推动着推荐系统理论和技术的发展，目前采用推荐系统的知名电商网站有亚马逊、淘宝和京东等。这些公司的推荐系统大都是通过分析用户历史行为来给用户做推荐，采用基于用户的协同过滤、基于物品的协同过滤、基于内容推荐和基于人口统计学的推荐等模型进行混合推荐。它们的基本思想就是通过分析用户历史行为给用户和物品建立模型，为同类用户推荐某一物品或为某用户推荐同类物品[6][10][[[13]](#endnote-13)]。推荐系统在这一类电商网站中的应用，一方面提高了其商品的销售；另一方面为用户提供了个性化的推荐服务，使得用户能够更容易的找到自己感兴趣的物品[6][13]。

由上可知，基于用户的协同过滤、基于物品的协同过滤、基于内容推荐和基于人口统计学的推荐等模型在商业领域的成功应用创造了极大的商业价值，也为用户在网上购物时提供了极大的便利。但是，由于它们在原理和模型方面的局限性，使得这类推荐原理及模型在应用范围上存在一定的限制，例如这类推荐系统只能较好的应用在给人推荐物品上，却不能较好的应用在给人推荐好友上。

1. 社交推荐系统

一般认为社交网络的理论基础最早来自于1967年六度分隔理论的创立，每一个较小的个人社交所组成的网络通过熟人就可以扩充为一个大的社交网络，且最多通过六个人就可以认识任何一个陌生人[12][[[14]](#endnote-14)]。国内外的社交巨头公司等的发展壮大均离不开社交好友推荐，通常情况下好友推荐系统结合用户画像和用户在社交网络中的特征进行推荐[[[15]](#endnote-15)][[[16]](#endnote-16)]。这类好友推荐系统通过搜集用户的基本信息和爱好信息建立用户画像，可以根据用户画像给该用户推荐一些有共同爱好或特点相似的其他用户作为好友，同时也可以分析用户在社交网络中的特征，推荐跟该用户具有共同好友的其他用户作为好友，也可以结合用户画像和共同好友进行混合推荐。

这一类社交推荐系统解决了给人推荐人的问题，极大的促进了人与人之间的交流。但是存在新用户因缺少信息而无法推荐好友的缺陷，一方面如果新注册用户基本信息很少，那么建立的用户画像就很模糊甚至无法建立用户画像，这样就会造成无法给该用户推荐好友的问题；另一方面新注册用户拥有很少好友或基本没有好友，那么该新用户在社交网络中会表现为独立的孤点或存在于一个很小的连通子网络中，就会造成在网络中几乎没有用户跟该用户拥有共同好友的现象，也没有办法给该新用户推荐好友。这就是所谓的推荐系统冷启动问题[6][13]。

目前也有把社交推荐应用于学术社交的案例，例如国外的和，国内的科研之友等。但是这些学术社交网站提供服务主要基于收集注册用户信息，建立用户画像，跟上面提到的等社交网站类似都存在冷启动问题，另一方面好友推荐普遍采用基于共同好友的推荐策略，这就限制了只能在该用户的二度人脉范围内进行推荐,极大的缩减了推荐范围，在一定程度上限制了用户的交流范围。这一类学术社交网站最为重要的缺陷是缺乏用户之间的合作关系数据，仅仅根据用户添加的好友和基本信息进行推荐，跟普通的社交网站没有本质的区别。就目前来看，这一类学术社交网站对于合作者推荐这一领域的研究指导意义有限。

早期，Newman为复杂网络分析做出了巨大的贡献，提出了一系列的复杂网络分析方法论[[[17]](#endnote-17)]和复杂网络节点相似度分析方法[[[18]](#endnote-18)]，最为重要的贡献就是跟Watts提出了NW小世界网络模型[[[19]](#endnote-19)]。Salton[[[20]](#endnote-20)]、Jaccard[[[21]](#endnote-21)]、Cosine[[[22]](#endnote-22)]、SimRank[[[23]](#endnote-23)]等节点相似度指标和模型的提出为网络科学的应用奠定了坚实的理论基础，随后链路预测模型及理论[[[24]](#endnote-24)][[[25]](#endnote-25)]的兴起推动了基于网络推荐模型及理论的发展。

近年来，有不少国内外学者对科学合作网络进行了研究[[[26]](#endnote-26)][[[27]](#endnote-27)]，提出了通过分析作者之间的合著网络或项目合作网络来分析作者之间的合作模式，并在此基础上进行合作者推荐。何贤芒等人通过科研人员之间项目合作关系构建了社会关系网络[[[28]](#endnote-28)]，冯欲等人对科研合著网络中合作者模式进行了分析[[[29]](#endnote-29)]。Kautz H[11]、Chen H H[[[30]](#endnote-30)]、Kong X [[[31]](#endnote-31)]和Lopes GR [[[32]](#endnote-32)]等人通过研究作者之间的学术合作关系组成的社会关系网络进行合作者推荐，Cohen S[[[33]](#endnote-33)]通过研究作者之间的关键词进行推荐。同时Chen H H[30]通过建立主题特征向量，采用主题贡献度来表征两个主题的相似度，Kong X[31]采用表征两个主题的相似度。这些前人的研究为本课题奠定了理论基础。

综上国内外研究概况可知，现有的推荐理论及模型主要应用于商业领域，主要基于分析用户的历史行为做推荐，均存在不同程度的冷启动问题。虽说近年来出现了少部分学术社交网站，但是这类网站缺乏具有说服力的用户数据和灵活的推荐策略，其本质上跟商用的社交网站一样，并不能看做专业的学术推荐系统，因此可参考性和指导意义有限。近年来，对合著网络模式和项目合作网络研究的兴起，为本课题奠定了理论基础。

### 重点难点问题

本课题具有权威、丰富的全国科研管理数据资源，主要包括历年项目数据、人员数据和机构数据等，为科研合作者推荐系统提供了权威的数据，并且在很大程度上解决了冷启动问题。基于项目协作关系的合作者推荐系统的设计与实现涉及到数据的清洗、协作关系的挖掘、协作关系建模、协作推荐模型和数据可视化等，重难点问题主要有：

1. 协作关系建模

本课题推荐系统采用基于关系网络的推荐理论，挖掘项目协作关系并选取恰当的网络模型图是协作关系建模重要的一步，如何合理、准确度量协作关系权重是整个协作关系建模的重点和难点部分。

1. 协作推荐模型

本课题的推荐模型主要由协作关系相似度模型和研究领域相似度模型混合组成，这两个模型的设计和实现是整个推荐系统的核心和难点部分。在模型设计上既要克服传统社交推荐系统推荐范围过小的问题，又要控制算法复杂度不能太高，同时还要保证推荐策略的灵活性和合理性。

1. 协作关系网络分析

基于网络科学和推荐系统的相关理论，可以对协作关系网络进行深入的分析。例如通过研究网络连通子图和巨片的变化等特征，可以分析大型网络的生长、演变、群组和社团的发展机制[[[34]](#endnote-34)]，为进一步探究科研协作关系提供新的方向；通过分析节点度的分布、节点权重和节点重要性等因素，可以将协作关系网络应用到寻找核心作者和挂名作者等方面，为确保科研质量提供保障。虽然在对协作关系网络的理论研究和应用方面还存在很多未知的重难点问题，但是仍然具有较高的理论研究和实际应用价值。

## 研究内容及意义

### 研究内容

本课题数据集基于历年全国科研管理数据，通过深度挖掘历年社会科学项目协作成员之间的协作关系和研究领域为基础来建立推荐模型，为用户推荐潜在的合作者，同时也是为社会科学数据在科研领域发挥效用做的一次尝试。

本课题的主要研究内容如下：

1）调研和学习目前推荐系统的相关理论，并结合国内外具体案例分析了不同推荐系统理论及模型在各场景下应用的不足之处。为发挥科研管理数据价值和解决当前寻找合作者过程中存在的问题，通过分析当前科研管理数据的现状并结合其优势，提出了基于项目协作关系的推荐系统。

2）设计了基于项目协作关系的推荐系统。通过对历年全国科研管理项目数据协作关系的分析，提出了一套高效数据清洗的规则，减少了计算和存储开销。采用无向有权图来建立协作关系网络，提出了多维权重的计算模型，其权重考虑了协作次数、协作时作者顺序和协作年份等多种因素，极大的改善了度量维度不足的问题。通过分析传统相似度度量方式的不足，提出了基于局部不定长路径协作关系相似度模型和基于研究领域重合度、研究领域侧重度的研究领域相似度模型，共同组成混合推荐模型确保推荐指标的合理性和准确性，同时改善了推荐范围不足和算法复杂度过高的缺点。

3）推荐系统的实现、测试和协作关系网络规律探究。实现了数据清洗规则、协作网络建模、协作关系相似度模型、研究领域相似度模型等系统核心部分，解决了推荐结果计算耗时和存储大量数据的问题。对系统基本功能及性能进行了测试，针对系统设计和实现过程中协作关系网络呈现的现象和规律加以探究和简要分析。

4）基于数据可视化的设计，实现了对协作关系的深入分析和多维度图表展示。

### 研究意义

本课题通过对现有大量科研管理数据资源的分析，以推进全国科研管理信息化建设和高效利用科研管理数据资源为出发点，结合科研人员合作趋势和现状分析中发现当前科研人员在寻求合作者中效率和质量低下的问题，挖掘全国科研管理数据资源中科研人员的协作关系和研究领域等特征，提出了一种基于局部不定长路径协作关系相似度模型以及基于研究领域重合度和研究领域侧重度的研究领域相似度模型所组成的混合推荐模型，给科研人员推荐潜在合作者。

本课题为提高现有科研管理数据资源的利用提供了新的方向，并在一定程度上解决科研人员寻找合作者效率和质量低下的问题。提高科研人员寻找潜在合作者的效率以及质量，可以极大的促进科研人员之间的有效合作者和有效交流，在一定程度上为促进学科发展、学科创新和增强科研质量提供了新的思路和方向。这是为发掘科研管理数据资源价值所做的一次尝试。

协作关系网络模型为进一步探究科研协作关系提供了基础，可以探究协作关系网络的演变趋势、群组及社团结构等，也可以衍生出推荐核心作者和挂名作者等应用方向，具有较高的理论研究和实际应用价值。

## 本文的组织结构

本文共计包含六个章节，各章节的内容简述如下：

第一章：绪论。本章节主要介绍了本课题的背景及来源，并结合国内外具体案例分析了不同推荐系统理论及模型在各场景下应用及存在的不足，分析了本课题的重难点问题，论述了本课题的研究内容及研究意义，最后简要介绍了本文的组织结构。

第二章：相关理论及技术概述。本章节主要介绍了在设计和实现基于项目协作关系的合作者推荐系统时用到的相关理论知识和关键技术，主要包括图论和网络拓扑相关理论、SSH框架，面向服务架构和数据可视化等相关技术。

第三章：推荐系统分析与设计。本章节主要介绍了推荐系统的总体需求分析和总体架构设计，设计和制定了数据清洗规则及接口、协作关系网络模型、推荐引擎模型、大数据计算和存储方案等。

第四章：推荐系统的实现。本章节首先简述了系统的开发环境，然后介绍了推荐系统各个模块和模型的实现，主要包括数据清洗的实现、协作关系网络模型的实现、推荐引擎模型的实现以及推荐指标计算和存储方案的实现。

第五章：系统的测试及数据分析。本章节首先对推荐系统的业务功能模块进行了测试和分析，然后对协作关系网络所呈现的规律进行了简要的分析和探究。

第六章：总结与展望。本章节对该课题的研究内容和研究成果做了简要总结，并分析了需要拓展的方面，同时对其可以深入研究和应用的方面做了展望和讨论。

# 相关理论及技术概述

基于项目协作关系的合作者推荐系统是将研究人员之间的协作关系抽象为图的模型，采用图的相关理论和属性进行建模和分析。系统采用了面向服务的架构，设计和开发过程中涉及到图论及相关算法模型、SSH框架、Thrift框架、数据可视化等技术。图论及相关算法模型是整个合作者推荐系统的理论基础，支撑着协作关系建模和推荐系统模型；SSH框架是系统功能开发的基础；Thrift框架及通信技术是整个系统构架的交互基础，是面向服务思想的具体实现；数据可视化是最终的结果呈现方式。本章依次对上述原理及技术进行介绍。

## 图论介绍

本节主要包含图的相关属性、节点相似性度量和图的搜索算法等，其中图的相关属性主要介绍了图的定义及表示、图的路径与连通性、节点的度与度的分布，图的搜索算法包括图的深度优先和广度优先搜索算法。

### 图的相关属性

本小节主要介绍了图的一些基本概念和属性，主要包括图的定义及表示、路径与连通性、度、度的分布。

1) 图的定义及表示：图可定义为一个由点集和边集抽象而成的元组。其中是顶点组成的非空有限集合，是无序集 的一个子集[14][16]。

按图中的边是否有向和是否有权，图可以分为四种类型：加权有向图、加权无向图、无权有向图和无权无向图。由于本课题基于加权无向图建立协作关系网络模型，故着重介绍加权无向图。加权无向图中的边是无向和有权的。边的无向是指任意点对与对应的是同一条边。顶点和也称为无向边的两个端点。边的加权是指由顶点和组成的无向边具有边权值。图2-1为加权无向图示意图。



图2-1 加权无向图

在图算法中，表示稀疏的无权图的最常用的方法是邻接表。它对每一个顶点建立一个单链表，这个单链表由邻接于顶点的所有顶点构成[[[35]](#endnote-35)]。邻接表示意图如图2-2，如果是加权图，则需要在图2-2边链表节点中增加一栏权重值。



图2-2邻接表示意图

图2-1加权无向图用邻接表表示如下：

 图2-3有权图邻接表示意图

表头节点由和三个域表示。是索引下标；是数据域，用来存储顶点的信息；是指针域，指向边链表的第一个节点。边链表由和两个域组成。是邻接点域，存储某顶点的邻接点在表头中的下标；域则存储指向边链表中下一个节点的指针。对于加权图，只需在边链表中加上权值域即可。

2) 路径与连通性

路径定义：若图中顶点和顶点经过顶点序列可达，并且任意两个相邻的顶点之间存在一条边，那么序列称为顶点到顶点的一条路径，其包含边的条数称为该路径的长度[35][[[36]](#endnote-36)]。

对于无向图，如果每一对顶点之间都至少存在一条路径则称该图是连通的；否则就称该图是不连通的。一个不连通图是由多个连通片组成的，如果该连通片满足以下两个条件：①连通性：该子图中任意两个顶点之间都存在路径；②孤立性：网络中不属于该子图的任一顶点与该子图中的任一顶点之间不存在路径。那么该连通片就称为连通子图。每一个网络图都是由若干个不相交的的连通片（连通子图）组成的。其中包含顶点数最多的连通子图就称为最大连通子图。

3) 度与度的分布

将一个节点连接到图的有向或无向链路的数量称为节点的度，标记为，度是有效描述单个节点属性的重要概念之一。网络中所有节点的度的平均值称为网络的平均度，记为[14]。

对于给定的网络，其邻接矩阵表示为，有：

 （式2-1）

 （式2-2）

网络节点的度与网络边数之间关系如下：

 （式2-3）

 （式2-4）

如果两个网络的节点数相同，网络节点的平均度相同就等价于它们具有相同的边数。为了更加深刻的刻画网络的一些特性，用统计学的方法来统计网络节点度的分布情况，记度为的节点数占网络总节点数比例为，那么这个概率分布即为度的分布。

### 节点相似性度量

社会网络分析中经典的三元闭包原则指出，如果两个人和拥有一个共同的好友,那么这两个人有很大概率成为朋友[13]。可以把这一原则推广到一般网络：两个节点的共同邻居数量越多，这两个节点就越相似，从而更倾向于相互连接。两个人共同好友数量的多少在一定程度上反映了其关系的强弱，可以基于此进行好友推荐。

1）基于共同邻居的节点相似度

最简单的基于共同邻居的节点相似度指标[14]定义如下：

 （式2-5）

其中表示的邻居，表示的邻居。通过对共同邻居数进行排序，即可推荐好友。但是该相似度指标仅仅只考虑了两个共同邻居的数量，并没有考虑他们各自拥有的邻居数量。

2) 基于共同邻居加权系数的节点相似度

在基于共同邻居的节点相似度的基础上，考虑了两个节点的共同邻居的相对数量，提出了Salton指标[20]、Jaccard指标[21]、Cosine指标[22]。

Salton指标定义如下：

 （式2-6）

其中、分别表示和的度，该指标综合考虑了和各自的邻居数量。

Jaccard指标定义如下：

 （式2-7）

该指标考虑了两个节点的共同邻居的同时也综合考虑了和邻居的整体情况。

Cosine指标定义如下：

 （式2-8）

以上两类节点相似度指标都是基于共同邻居的，只考虑了两度关系。大型网络中有时需要根据节点的路径情况来考量其相似度，下面介绍几种基于路径数的节点相似性指标。

3) 局部路径（Local Path,LP）指标[14]

它在共同邻居的指标上考虑了三阶邻居的贡献，定义如下：

 （式2-9）

其中为可调参数，为网络的邻接矩阵，给出了节点之间长度为的路径数。当时，LP指标就等价于最简单的基于共同邻居的节点相似度指标。

4) Katz指标[14]

Katz指标不仅考虑节点之间的最短路径，还考虑了非最短路径。它认为越短的路径越重要，对越短的路径赋予越大的权值。假设有一个与节点距离为的节点，对的相似度贡献为，。设为从节点到节点经过距离为的路径数目，则有：

 （式2-10）

其中即为从节点到节点经过的边数为2的路径的数目。同理可以得到，将这些值根据路径的长短赋予不同的权重然后累加，便可以得到相似性指标：

 （式2-11）

其中为权重衰减因子。对应的相似性矩阵如下：

 （式2-12）

为了保证收敛性，的取值必须不大于邻接矩阵的最大特征值的倒数。当衰减因子很小时，高阶路径的贡献就很小了，使得Katz指标的结果非常接近于局部路径指标[14]。

### 图的搜索算法

图论是网络科学发展的基础，同时网络科学的发展也在丰富和完善着图论。图论的基本算法包括图的搜索算法、拓扑排序算法、最小生成树算法、最短路径等[35][36]。本节主要介绍图的搜索算法，包括广度优先和深度优先算法。

1. 广度优先搜索算法

广度优先搜索（Breadth First Search，）是连通图遍历的一种策略。该算法的基本思想是：首先访问起始点的所有邻接点，然后再访问较远区域[35]。算法需要在访问所有距离源节点为的所有节点后才能访问距离源节点为的其他节点。广度优先搜索示意图如图2-4所示，A是起始节点，数字表示节点访问的顺序。



图2-4 广度优先搜索示意图

伪码实现如下：

|  |
| --- |
|  |

1. 深度优先搜索算法

深度优先搜索（Depth-First-Search，DFS）的基本思想是：对于最新发现的顶点，如果它还有以此为起点而未探测到的边，就沿此边继续探测下去。当顶点的所有边都被探寻过后，搜索将回溯到发现顶点的前驱节点。该过程一直进行到已发现从源顶点可达的所有顶点时为止。如果还存在未被发现的顶点，则选择其中一个作为源顶点，并重复以上过程。整个过程反复进行，直到所有顶点都被发现时为止[36]。其搜索示意图如图2-5所示，数字表示访问的顺序。



图2-5 深度搜索算法示意图

递归是该算法较为简洁的实现，但是递归层级不能过深，否则会造成堆栈溢出。

递归伪码如下：

|  |
| --- |
|  |

## SSH框架

### Struts2

Strut2[[[37]](#endnote-37)]是JavaWeb中非常成熟的MVC开源框架，核心控制器是Struts2框架的基础，涵盖了控制流程和处理机制。而业务控制器和业务逻辑组建不是框架本身完成的，它需要用户根据自身的业务逻辑去实现，在开发过程中还要编写相关的配置文件。其体系结构图如图2-6所示。



图2- 6 Struts2体系结构图

Struts2主控制器FilterDispatcher接收到客户端的请求后，会经过ActionMapper映射到对应的Action处理请求，通过读取Struts.xml构建对应的Action代理（ActionProxy）执行请求，并经过一系列拦截器返回最终结果。

### Spring

Spring[[[38]](#endnote-38)]是JavaEE 开发的全功能开源架构，它的分层架构极大简化JavaWeb 开发流程。Spring有两个最基础的功能特性：IoC和AOP。IoC[[[39]](#endnote-39)]，Inversion of Control，即控制反转,也叫依赖注入。Spring通过容器技术管理所有被注册的类，任何服务类都可以直接从容器中获取，而它的反转就是从原来的主动获取变成享受服务，使得依赖对象的获取方式发生了反转。AOP[39]，Aspect Oriented Programming，即面向切面编程。采用动态代理和字节码生成技术实现，在软件开发过程中，对于与业务功能无直接关系且比较通用的模块，比如事务管理与日志记录，把它们封装成组件，从而实现与业务功能的解耦这类事务就被称为横切点[[[40]](#endnote-40)]。

### Hibernate

Hibernate[[[41]](#endnote-41)]是一个开源的ORM（Object Relational Mapping）框架，通过对JDBC轻量级的对象封装与数据表建立联系，提供对数据库查询等节本操作，使开发者可以更好的专注于业务开发。Hibernate支持数据缓存、事务管理，开发效率高，易于扩展，是目前最为流行的ORM框架之一。

## 面向服务架构

### SOA

SOA（Service-Oriented Architecture，面向服务的架构）既是一种架构概念，又是一种系统架构风格。它使用服务作为构建块，通过松耦合、互操作来促进企业集成和组件的重复使用[[[42]](#endnote-42)]，并且不需要了解底层实现就可以进行跨平台调用服务。

SOA服务通过独立于底层平台和编程语言的某种格式定义（契约）交互接口，使得SOA服务具有调用的跨平台性，实现语言的隐藏性和技术的独立性[[[43]](#endnote-43)]。

 图2-7 SOA架构图

服务化的好处就是不限定服务的提供方使用什么技术选型，只要事先定义好相关的契约文件，就可生成相应的通信接口，与底层平台和编程语言无关。面向服务化的思想极大的实现了技术和系统解耦，使得服务就像组件一样可以为多个系统所调用，提高了服务的重用性。

### Thrift

Web Service[[[44]](#endnote-44)]和Restful[[[45]](#endnote-45)]为当前较为流行的服务调用方式，其消息格式分别基于SOAP[[[46]](#endnote-46)]和JSON，其数据传输分别基于XML和JSON，这些传输方式均存在一定的缺陷，如XML包含大量的标签，体积较大，传输效率低下；JSON体积较小，新颖，但不够完善。

Apache Thrift[[[47]](#endnote-47)]是由 Facebook 开发的基于接口描述语言定义服务的远程服务调用框架，支持跨语言开发并可以通过引擎创建高效、无缝的服务。其二进制数据传输格式较XML 和 JSON具有更明显的优势，体积小、支持高并发和多语言[[[48]](#endnote-48)]。



图2-8 Thrift系统架构图

用户事先编写好契约文件，根据契约文件生成相应的通信接口，服务提供端实现接口的具体业务，服务消费端直接调用相应的接口，无须关注相应的接口具体实现。使得服务提供端和服务消费端解耦，具有极高的可拓展性和易维护性。

TProtocol提供了一系列按类型进行读写的接口。

TTransport抽象了底层网络通信部分的接口。对socket进行了封装，提供阻塞和非阻塞式的Socket,也提供了对流的一系列操作接口。

## 数据可视化

数据可视化是数据展现的形式，从多个不同的角度和维度来度展现数据，从而达到对数据有更加直观、更加深入的理解和分析。数据可视化的目的在于借助图形化的方式，清洗、高效的解析信息展现和交流[[[49]](#endnote-49)]。

本课题采用目前较为流行的数据可视化插件Highcharts[[[50]](#endnote-50)]和D3[[[51]](#endnote-51)]来展现数据，Highcharts 和D3采用纯JavaScript开发，支持多种数据展现形式。结合HTML、CSS等技术，可以实现较为丰富的动态效果。

## 本章小结

本章主要介绍了基于项目协作关系的合作者推荐系统的设计和实现过程当中涉及的相关理论和技术，主要包括图论的基本概念和相关理论、JavaWeb框架、面向服务架构和可视化技术等。首先，介绍了图论的基本概念包括图的表示、图的分类、图的连通性、节点的相似性和图的搜索算法等；其次，介绍了JavaWeb开发框架SSH；接着介绍了面向服务的架构思想及关键技术Thrift;最后介绍了数据的可视化技术。

# 关系抽取分析与设计

本系统在架构上采用了面向服务的思想，整个服务端分为服务提供端和服务消费端，使得系统充分解耦。服务提供端负责提供推荐服务，其中推荐引擎模型的设计是本系统服务提供端设计的重点，主要包括关系网络模型、推荐模型和推荐指标计算及存储方案等的设计。服务消费端和客户端主要是负责数据的清洗、数据组装和数据展现。

本章从系统的总体需求出发，分别分析了系统的服务提供端的模型需求和服务消费端的功能需求。着重介绍了系统总体架构设计、服务消费端的数据清洗模块、服务提供端的协作网络建模和协作推荐建模等的设计。

## 总体需求分析

本课题以合作者推荐为设计目标，通过分析和总结国内外推荐系统原理和应用案例中的不足，并结合推荐系统的设计流程梳理出本课题的需求。基于项目协作关系的合作者推荐系统旨在为解决科研人员寻找合作者效率和质量低下问题，同时为挖掘科研管理数据资源的潜在价值提供了新的方向。为广大科研人员寻找合作者提供了渠道，在一定程度上可以促进科研人员之间的交流，提升科研效率、科研成果水平。

### 服务提供端

1. 协作关系网络建模

本课题采用基于关系网络模型的推荐理论，关系网络模型是本系统分析的基础，设计合理的关系网络模型是本系统极为重要的一步。合适的网络模型和全面的度量方式对于后续分析和深入探索科研人员的协作关系具有重要意义。

1. 推荐系统模型

推荐系统模型和算法设计是整个推荐系统的核心，本文基于协作关系相似度和研究领域相似度进行混合推荐。设计合理、高效的模型和算法对于推荐效率和推荐质量具有决定性的意义。

1. 推荐指标计算及存储

本数据集拥有丰富的数据资源，通过数据清洗之后仍然会有数以十万计的数据，这些数据经过推荐模型的计算会产生数以百万计甚至数以千万计的数据。高效计算和高效存储大量的数据对于推荐系统的性能十分重要。

### 服务消费端

1. 数据清洗

本课题基于历年全国科研管理数据资源，通过深度挖掘科研管理数据中科研人员之间的项目协作关系为基础建立模型进行分析和探究。面对大量的科研管理数据，如何选出有效信息找出隐藏的关联性，构建模型，发现有价值的信息与知识，为科研人员寻求潜在合作者提供帮助是本课题的研究重点之一。因此制定一套数据清洗规则有效筛选信息对于后续建模和算法设计具有重要意义。

1. 合作者推荐

合作者推荐主要是利用服务端提供的推荐引擎计算的指标来推荐潜在合作者，更好的为广大科研工作者服务，主要包括作者检索、作者推荐、作者信息展示等基本功能。

1. 数据可视化

数据可视化主要是展现作者协作关系及整个协作关系网络的基本情况，这些数据可以充分反映出作者的协作情况和各年度协作关系网络全图以及连通子图的演变趋势。借助于可视化的技术，能够以图表的形式，更加直观和清晰的把所有的分析结果都呈现给用户。

## 总体架构设计

本推荐系统的总体架构设计充分考虑了系统的可拓展性、技术的前沿性、结构上的伸缩性和推荐服务模型的通用性。运用面向服务的思想将推荐服务模型抽象为通用的服务化组件，便于其他系统的接入，力求打造平台级的系统，为科研管理数据中心和全国科研人员更好的服务。本节将从功能架构和系统架构分别予以介绍。

### 功能架构

推荐系统主要包含数据层、服务层和展示层组成，其中服务层又分为服务提供端和服务消费端，其总体功能架构设计如下：



图3-1总体功能架构

数据层主要包括生产数据库和数据仓库，其中生产数据来源于历年全国科研管理数据，数据仓库则是通过对生产数据库中的数据进行抽取、清洗、转换后得到的。有了数据仓库的支持，后续推荐系统相关建模才能顺利进行。

服务提供端是本系统的重点部分，主要包括数据解析及入库、协作关系网络模型的设计、推荐系统模型的设计和推荐结果计算及存储等。数据清洗及组装主要是服务消费端按照清洗规则把清洗后的数据组装成特定的格式推送到服务提供端，服务提供端进行处理后将这些数据存入数据仓库。协作关系网络模型主要是用来分析合作者之间的协作关系相似度，推荐系统模型主要是通过分析协作关系网络模型来建立协作关系相似度模型和研究领域相似度模型，然后融合成最终的推荐指标。推荐指标计算及存储方案是为了解决推荐模型处理数据过程中数以百万级数据的计算速度和查询效率问题。

服务消费端主要包含推荐查询、作者检索、数据分析和数据清洗及组装等。这个部分向上层为展示层提供服务，向下通过调用服务提供端的推荐服务来获取数据资源。推荐查询、作者检索、数据分析都是通过调用服务提供端提供的相应服务来获取数据并进行相应的处理来向上层提供服务。

展示层主要分为参数配置和数据展现。参数配置提供了对特定数据清洗和抽取的入口，使得服务消费端跟服务提供端更好的解耦。数据展现方式对于用户接受和理解数据具有决定性的影响，本文借助于Highcharts和D3等可视化插件多数据进行多维度的展现。

### 系统架构

整个推荐系统采用了面向服务的思想，把推荐服务服务化成服务组件的形式，可以为多个跨平台的系统提供推荐服务，使得系统结构解耦，具有极高的灵活性、可维护性、可拓展性和可重用性。

服务化组件为多个平台提供服务的示例如图3-2所示。



图3-2 服务化组件示意图

如果图3-2中的Web应用和App应用都需要推荐功能，按照传统的软件开发思路是直接在各自的应用中加入推荐业务，这样会有诸多缺点：①系统业务高度耦合，系统结构沉重。②可重用性低，需要编写两套业务一样的代码。③可维护性差，由于系统业务的高耦合，在维护推荐业务时可能影响其他相关业务。④跨平台性弱，只能在同一个平台开发。

采用面向服务的思想，可以把一些业务服务化成组件的形式对外提供服务。如图3-2所示，服务化组件Service跟Application之间是解耦的，它们之间通过事先定义的契约生成的统一接口进行通信，业务之间彻底解耦，同时也支持跨平台通信，具有极高的灵活性、松耦合性、可维护性、可拓展性和可重用性。

结合面向服务的思想和相关技术以及模型，推荐系统的总体系统架构如图3-3所示。



图3-3 总体系统架构图

系统采用面向服务的思想，开发框架整体基于B/S模式的三层架构，将服务端分为服务提供端和服务消费端，它们之间主要通过Thrift框架进行通信。

浏览器主要采用HTML、JS、CSS并结合Hightcharts和D3对数据进行展示和渲染，数据以JSON格式进行传输。

服务消费端分为视图层和业务层。业务层处理请求，返回数据至视图层，浏览器展示数据，这个过程主要由Struts和Spring控制。业务层主要是通过Thrift框架定义契约生成的接口调用服务提供端的相关服务获取相应数据并进行一定的处理为视图层提供服务，这个过程主要由Spring和Thrift管理和控制。

服务提供端主要是提供推荐服务，通过Thrift框架来跟服务消费端进行数据交互。服务提供端主要是解决和实现数据解析与入库、关系网络模型、推荐系统模型、推荐结果计算和存储方案等。

## 数据清洗模块

数据清洗模块主要包括数据来源及现状的分析、清洗规则的设计和清洗接口的设计。首先初步分析了数据来源及现状，说明了数据清洗的必要性；然后初步分析了关系网络图的特点，制定了一系列清洗规则和清洗流程，对清洗后的数据进行了初步的比对来说明清洗效果；最后设计了一套灵活的数据清洗接口。

### 数据来源及现状

本课题具有权威、丰富的全国科研管理数据资源， 本文选取了2010-2016年度全国科研管理数据中的项目立项数据和对应的人员数据，主要包括项目名称、项目立项年份、项目所属学科、项目参与者、作者姓名、作者性别、作者职称、作者单位等信息。其数据情况大体如下：项目立项数据共计24195条，与立项项目相关科研人员数据共计118187条。

为了初步分析科研人员之间的协作关系情况，本文以项目为中介，基于科研人员之间的项目协作关系，将科研人员和协作关系抽象为关系网络中的节点和边，建立无向无权网络，得到如下数据：

连通子图数9922；

最大连通子图节点数：73669；

图节点总数：118187；

连通子图节点数最多Top10 ：73669、106、86、74、70、70、68、67、65、62

通过以上数据可以看出：最大连通子图节点数占全图节点数的62.3%，而其余37.7%的节点却分布在9921个连通子图中。节点数较少的连通子图在全图中就像很多孤立的点，对于分析协作关系和合作者的推荐几乎没有贡献，而且还会造成计算和存储上的额外开销。

### 数据清洗规则设计

由上面分析可知，协作关系网络中存在大量单点节和含节点数较少的小连通子图。因此，我们需要制定规则相应的清洗规则，用以去除在关系图中表现为单点和包含节点数较少的连通子图，从而减少计算和存储开销。

清洗规则主要分为以下三步：

第一步：取出给定年度范围内的立项数据；

第二步：根据取出的立项数据取出其相应的申报人和合作者，如果参与该项目的人数大于2人，则继续进行第三步；否则直接舍弃该项目及相应合作者；

第三步：判定该项目的申报人和合作者在给定年度范围内是否参与过其它的项目。如果至少有一人参与过其它项目，则保留该项目及人员数据；否则舍弃该项目及人员数据。

经过清洗规则清洗后的数据情况如下：

连通子图数量 1563

最大连通子图节点数： 73669

图节点数： 93045

连通子图节点数最最多top10：73669、106、86、74、70、70、68、67、65、62

通过与清洗前的数据对比可以发现节点数减少了大约20%，连通子图数减少了大约83%，而最大连通子图和包含节点数较多的连通子图基本没有变化。从而可以推断出减少的这些节点和连通子图都是包含较少节点的连通子图或是单节点。数据清洗后减少了计算和存储开销。

数据经过清洗后可以得到一些项目和人员信息，我们可以据此进一步从数据源抽取其他相关信息，具体流程如图3-4



图3-4 数据清洗流程图

### 数据清洗接口设计

清洗后的数据组装成一定的格式并推送至服务提供端解析入库。这个过程中数据清洗是对基本数据的清洗过程，还需从数据源取出其他相关信息组装成数据仓库需要的格式。整个过程的数据组装处理都是在服务消费端完成的，数据格式组装完成后调用契约生成的统一接口将数据推送至服务提供端，由服务提供端解析数据并插入至数据仓库中。数据存储结构如下表所示：

表3-1 人员表user数据结构

|  |  |  |
| --- | --- | --- |
| 属性 | 数据类型 | 描述 |
| id | int | 主键，作者id |
| authorName | varchar | 作者姓名 |
| authorId | varchar | 生产环境数据人员id |
| gender | varchar | 性别 |
| title | varchar | 职称 |
| agency | varchar | 所属机构 |

表3-2 协作关系表 user\_relationship数据结构

|  |  |  |
| --- | --- | --- |
| 属性 | 数据类型 | 描述 |
| uId | int | 一作id |
| uName | varchar | 一作姓名 |
| uIds | varchar | 合作者ids,以;隔开 |
| uNames | varchar | 合作者姓名，以;隔开 |
| applicationId | varchar | 项目id |
| year | int | 协作年份 |

表3-3研究领域表user\_keyword

|  |  |  |
| --- | --- | --- |
| 属性 | 数据类型 | 描述 |
| uId | int | 作者id |
| uName | varchar | 作者姓名 |
| displine | varchar | 研究方向 |
| numbers | int | 该研究方向参与过的项目数 |

服务端之间的通信接口是契约统一生成的，其参数格式固定。服务消费端可以采用任何形式的数据源，包括数据库、excel表、xml、json等，只要组装成接口要求的数据格式推送到服务提供端即可。服务提供端也只需要接收固定格式的数据存储即可，而不管数据源的形式和来源，数据清洗接口示意图如图3-5所示。

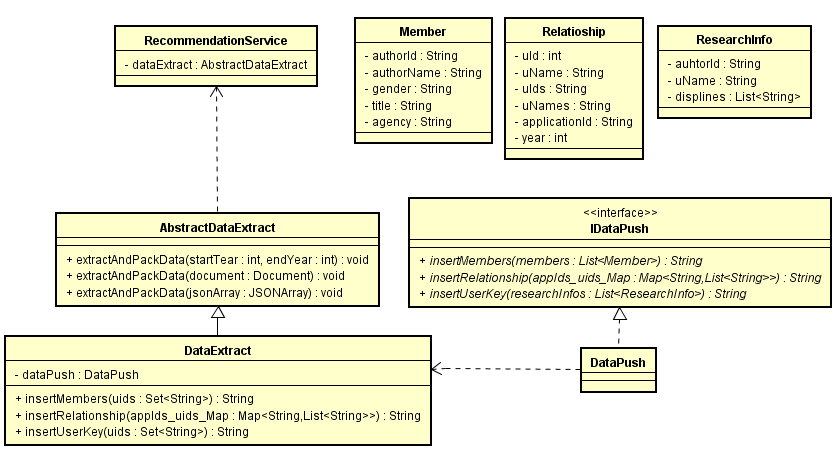


图3-5 数据清洗接口示意图

其中Member、Relationship、ResearchInfo等类定义了服务消费端和服务提供端的数据格式，IDataPush接口和DataPush类是服务消费端推送数据的接口和实现类，同时也是服务提供端处理数据的接口和实现类。以上这些接口和类都是通过Thrift契约定义的，通过Thrift工具可以生成对应的服务消费端与服务提供端接口及类。由于本系统服务消费端采用Java开发，服务提供端采用Python开发，故生成服务消费端接口和类均为Java类型，服务提供端需生成Python类型的接口和类。

AbstractDataExtract类给出了不同类型数据源的定义，由子类实现具体功能。extractAndPackData方法实现数据的清洗功能，子类继承AbstractDataExtract类并实现了数据组装功能，通过IDataPush接口推送数据至服务提供端。服务提供端实现IDataPush接口进行数据的存储操作。

RecommendationService拥有AbstractDataExtract子类实现的一个实例，为Action层提供服务。

## 协作网络建模

协作关系网络建模主要包含了网络模型图的选取、协作关系权重的设计和协作关系网络图的构建。首先分析了协作关系并将其抽象为两种网络图模型，分别分析了其特点；然后考虑了多维影响因子设计权重；最后利用合适的网络模型建立协作关系网络图。

### 网络模型图设计

每一个社科项目都会有申请人和参与者，通常情况下有些申请人或参与者可能是其他项目的参与者或申请人，这些申请人和参与者的关系就形成了一个协作关系网络。网络中节点表示作者，边表示协作关系。文献[28]认为同一申请书中申请人与参与者之间存在边，其他参与者之间没有边的关联。文献[29][30]中对于合作关系的表示是只要有共同署名的项目就把他们视为合作关系，他们之间就有一条边。

现在假设立项项目《X1》有合作者，其中数字代表作者的顺序，为一作也就是项目的负责人或申请人。项目与作者之间的关系抽象出来如图3-6所示。按照文献[28]和文献[29][30]中对于合作关系的定义，可以把项目《X1》的合作关系抽象为图3-7和图3-8所示的网络图。



图3-6 项目X1与作者示意图

图3-7项目X1合作关系图 图3-8 项目X1合作关系图

图3-7星型耦合网络模型[14]表示的合作关系能够较好的区分第一作者跟非第一作者的关系，可以非常容易的体现出项目的核心成员，但却没有体现出非第一作者之间的关系。通常情况下，可以认为一个项目的申报者跟参与者之间的合作关系是直接的，但该模型却分割了非第一作者的合作关系。

图3-8全局耦合网络模型[14]表示的合作关系很好的体现出了第一作者与非第一作者的合作关系，参与过同一个项目的成员之间可以视为直接合作，这与现实情况十分吻合，但却体现不了不同作者之间的重要性和紧密程度地位。

由于本课题主要是分析项目作者之间的协作关系，通过以上两个模型的对比，可知图3-8全局耦合网络模型较为合适。图3-8中边的是由项目参与者按照协作顺序排列组合形成的，的次序分别代表协作顺序，那么网络图中的边有。虽然协作关系网络图为无向的，为了方便接下来边权重的定义，我们规定边中端点的协作顺序在端点的协作顺序之前。

为了解决全局耦合网络模型不能体现不同作者之间重要性和紧密程度的问题，本文把关系网络设计为无向加权网络，通过给定不同权重来区分作者之间的重要性和紧密程度，权重越大，重要性越高，联系越紧密。

### 协作关系权重设计

冯郁等人认为合作者之间不区分署名顺序，多次协作仅记录一次[29]。Hung-Hsuan Chen等人仅考虑合作者之间的协作次数，而不考虑合作者之间的署名顺序[30]。这些度量方式考虑的影响因素单一，并不能很好的衡量协作关系。

由多人共同完成的项目，一般而言第一作者和第二作者的贡献较大，联系也更加紧密；如果两个项目协作年份不同，那么我们有理由认为项目协作年份近的比项目协作年份远的作者之间的关系更紧密。基于以上事实，本文综合考虑了协作时作者的署名顺序、协作次数和协作年份等因素对协作关系的影响，设计了一套基于多维影响因子的协作关系权重度量方式。

一般而言第一作、第二作的贡献较大，联系也更加紧密；作者顺序越靠后，他们的重要性越弱，联系强度也更弱。我们有必要区别衡量项目协作时不同作者次序间的关联性强度。

定义两节点某次协作的权重为，其中分别表示某次合作时他们的作者排序，其表达式如下：

 （式3-1）

其中，为调节因子，且有，为常数，且有。

该表达式可以看出协作时作者次序越靠前，其权重越大，作者之间的协作关系更加紧密；协作时作者次序越靠后，其权重越小，作者之间的协作关系更加疏远。

一般而言，协作年份越近，说明当前协作关系越紧密；协作年份越远，说明当前协作关系越疏远。协作年份的远近跟协作关系衰减程度正相关，年份越远，协作关系的衰减越大。衰减因子的定义如下：

 （式3-2）

其中表示协作年份，表示当前年份，且有**，**为调节因子，一般可取20~50之间。根据该定义可知：指数不大于0，当且仅当协作年份等于当前年份时，其衰减因子为1，协作关系不衰减；当协作年份小于当前年份时，其衰减因子小于1，且距离当前年份越远，其衰减因子越小，协作关系衰减越快；为调节因子，越大衰减越慢，越小衰减越快。

在给定的年度范围内，有些作者会多次协作，定义作者对总的协作权重如下：

 （式3-3）

其中为作者协作的次数，为第次协作对应的年份。可以看出该权重定义综合考虑了协作时的作者顺序、协作年份、协作次数等因素，权重越大表明关系越紧密，再次协作的可能性也越大。该度量方式较为全面的衡量了多种因素对协作关系的影响，具有较高的参考性。

### 协作关系网络构建

协作关系网络采用的是无向有权图，构建协作关系网络的重点就是要计算出边信息和权重信息等。下面分别介绍了协作关系网络的融合过程、边信息和权重信息的计算。

按照3.4.1网络模型图的设计一节知，每一个项目的所有参与者都可以组成一个全局耦合网络。如果两个项目有共同的参与者，那么这两个较小的全局耦合网络就可以通过该共同参与者作为媒介连接成为一个较大的网络。如果这种项目有共同参与者的现象很多，那么很多小的全局耦合网络就会连接成为一个大的网络，最后组成的网络就会显得很紧密。

假设现有项目《X1》、《X2》，协作年份分别为，分别包含参与者和其中作者的次序代表他们协作时的作者顺序。按照3.4.1节网络模型图，两个项目的全局耦合网络和融合之后的网络图如图3-9所示。



图3-9 全局耦合网络和网络融合示意图

其中图3-9-a和图3-9-b分别为项目《X1》和《X2》的协作关系网络图表示。由于两个项目中有共同的参与者和，所以两个协作关系网络图可以通过参与者和作为媒介连接成为一个大的网络图，如图3-9-c所示。可以看出作为媒介的和在网络中拥有的边数较其他的节点多，从这一方面可以看出作者和在整个网络中协作的重要性。

同时对每个项目参与者在网络图中所组成的节点对计算协作权重作为该节点对组成边的权重，对于在多个网络中出现过的边，由于网络是无向的，把它们的累加权重作为最终边的权重。

对于节点在项目《X1》中分别对应第一作者和第二作者，在项目《X2》中分别对应于第二作者和第一作者。图3-9-c中边的权重是由图3-9-a中边和图3-9-b中边权重累加而成。

根据式3-1可知，对于图3-9-a有，对于图3-9-b有。假设项目年度范围中的项目数据结束年份为，根据式3-2有，项目《X1》协作关系衰减因子，项目《X2》协作关系衰减因子。由式3-3有融合后的网络图3-9-c中边的最终权重为：



同理，可计算出其它所有边的权重。

有了节点信息、边信息和权重信息，就可以据此建立一个无向有权图，该网络模型表征了协作关系，又称为协作关系网络。

## 协作推荐模型

推荐模型设计主要包括协作关系相似度模型设计、研究领域相似度模型设计和推荐指标模型的设计。通过对传统节点相似度度量方式的分析，提出了一种基于局部不定长路径协作关系相似度模型，该模型极大的改善了传统相似度度量方式维度不足和算法复杂度过大的缺点。同时本文提出了基于研究领域重合度和研究领域侧重度的概念和模型来计算研究领域相似度，该模型能够很好的反映研究领域相似度。

首先分别通过协作关系相似度模型和研究领域相似度模型计算给定作者和待推荐作者的协作相似度和研究领域相似度，然后将两个相似度融合为最终的推荐指标，根据推荐指标的大小进行排序推荐。推荐系统模型示意图如图3-10。



图3-10 推荐系统模型示意图

推荐模型数据存储结构如下表：

表3-4 推荐数据表user\_recommendation数据结构

|  |  |  |
| --- | --- | --- |
| 属性 | 类型 | 描述 |
| uId | int | 作者id |
| rId | int | 待推荐作者id |
| score | decimal | 推荐指标 |
| rel\_score | decimal | 合作关系相似度 |
| rea\_score | decimal | 研究领域相似度 |
| overlap\_ratio | decimal | 研究领域重合度 |
| cos\_ratio | decimal | 研究领域侧重度 |

### 协作关系相似度模型设计

如何衡量两个节点的相似性是一个及其重要和值得探讨的问题。分析节点相似性和重要度有很多方法，第二章介绍了基于共同邻居的节点相似度和基于共同邻居加权系数的节点相似度，这两类节点相似度指标都是基于共同邻居的，只考虑了两度关系。在大型网络中有时需要根据节点的路径情况来考量其相似度，局部路径指标仅考虑了路径长度为2和3的情况，并且没有体现出节点的权重和各路径的相似度之间的区别；Katz指标考虑的路径范围相对来说较为合理，但其认为相同长度路径对相似度的贡献因子相同，且考虑路径长度过长时复杂度会过高。

Chen H H通过路径分析，提出了基于路径长度的节点相似度模型[30]。该模型能够很好的处理复杂网络中节点相似度的计算，但是在计算节点相似度时仅仅只考虑了两节点的共同协作次数，而且考虑的节点范围跟路径范围一致，缺乏灵活性。

本文在基于路径长度的节点相似度模型的基础上，考虑综合考虑了协作次数、协作时署名顺序和协作年份等因素对节点相似度的影响，并分别考虑节点范围和路径长度，提出了一种基于局部不定长路径协作关系相似度模型来度量节点相似度。该模型先确定给定节点的三度关系内的所有节点，然后综合考虑协作次数、协作时署名顺序和协作年份等因素计算该节点与中节点的相似度，在计算两个非相邻节点的相似度时采取的是找到该两节点之间的路径长度在一定范围内的所有路径，然后计算所有符合条件的路径的相似度的累加和作为最终两节点的相似度。该模型考虑了三度范围以内的节点，在一定程度上兼顾了节点范围太小和计算复杂度过大的问题。同时考虑了路径长度在内的所有路径，避免了路径长度过大寻址效率低下的问题。分别考虑节点范围和路径长度，使得度量方式更为灵活。最为重要的是，该模型考虑综合考虑了协作次数、协作时署名顺序和协作年份等因素，能够很好的区分不同路径对相似度的贡献程度。下面分别从协作关系相似度模型定义和算法两部分做介绍。

1. 协作关系相似度

假设和****是关系网络图中直接相连的两个节点，节点的邻接点集合记为，节点和节点组成边的权重为，节点的权重记为，定义如下：

 （式3-4）

通过的定义可以看出，越大，该节点在关系网络中的重要性越大。在某种程度上反映了节点在关系网络中的重要性，考量节点的值和节点的度等其它因素，可以找出关系网络中处于核心地位的节点，并且这些节点极有可能是该领域或学科内具有重要影响力的学者。

定义节点和节点****的关系强度为，其表达式如下：

 （式3-5）

可以看出关系强度是非对称的，且有。一般情况下有，原因是通常情况下，这种定义与实际情况也是符合的。实际情况中，如果作者对应的权重较大，说明作者跟很多人合作过或在合作中占据重要位置导致与相邻的边权较大；而为的众多合作者中一个，为定值。若越大，对于来说，的重要性越弱，关系强度也越弱；若越小，对于来说，的重要性越强，关系强度也越强。反过来，对于来说，的重要性取决于。

对于不相邻的两个节点和节点，假若节点通过节点可以到达节点，也就是说节点组成一条可达路径，那么节点和节点的关系强度取决于节点和节点以及节点和节点的关系强度。节点和节点的关系强度定义如下[30]：

 （式3-6）

由于保证了，故一定有且。这意味着两个节点之间路径越长，其关系强度越弱；当路径长度大到一定程度时，其关系强度及其微弱甚至可以忽略不计。这与实际情况也是符合的。

上述定义推广到更加一般的情况，假设从节点到节点存在一条简单的路径，路径依次包含节点为。定义节点和节点在通过路径的相似度为[30]：

 （式3-7）

在复杂网络情况下，通常两个不相邻节点之间会有多条不同的路径。假设节点到节点之间存在条不同的简单路径可达，那么节点和节点之间的相似度可以定义为[30]：

 （式3-8）

在巨大的复杂网络中寻找两节点之间符合条件的所有简单路径是本模型的一个重点问题。

1. 协作关系相似度模型

上述协作关系相似度计算在理论上是可行的，在一个拥有数十万计节点的复杂网络中寻找任意两个节点之间的所有简单路径需要的计算资源和时间是极其巨大的。假设网络图顶点数为，最坏情况下所有单对顶点个数为，利用深度优先遍历搜索算法的时间复杂度为，计算所有点对的时间复杂度为。

由于本模型是为了找到相似度比较高的一系列节点，根据式（3-7）可知：路径越长，节点的相似度越小，路径长度达到一定时，其节点相似度很小甚至可以忽略不计。为了降低对计算资源的消耗和减小算法复杂度，有必要缩小节点的搜索范围和路径长度。

本文选取了待推荐节点三度范围内的所有节点，假设网络中节点的平均度为。考虑到搜索路径过长会带来较大的开销，并且其相似度会很小这一事实，本文仅考虑路径长度在以内的情况。利用深度优先遍历搜索算法的时间复杂度为，在有个顶点的网络图中计算所有点的时间复杂度为。在特定的网络中、皆为定值且数值较小，亦可视为常数，若，则时间复杂度可近似为。这样极大的降低了计算性能，从而使得该模型到达应用可接受的范围。从某种程度上说，该模型的关系强度也是基于局部的，它考虑的邻域层次比一般的基于局部的相似度邻域更多。

给定网络中的一个节点，要找出跟该节点相似度较大的节点就是要计算出该节点跟其三度范围内的节点的相似度。计算某节点的协作关系相似度算法步骤定义如下：

步骤一：找出给定节点三度范围内的所有节点，并将该局部关系网络图以邻接点的形式表示为graph；

步骤二：对给定的点和三度范围内的点所组成的点对，采用深度优先搜索算法结合回溯算法寻找路径长度在以内的所有路径paths；

步骤三：结合点对的所有符合条件的路径paths，按照式3-8计算该点对的相似度。

其模型示意图如图3-11所示。



图3-11 协作关系相似度模型示意图

### 研究领域相似度模型设计

Hung-Hsuan通过建立主题特征向量，采用主题贡献度来表征两个作者的主题相似度[30]，Kong X采用表征两个作者的主题相似度[31]。这些度量方式考虑的因素较为单一，且特征主题向量较稀疏，并不能较好的度量主题相似度。

为了从多个维度衡量研究领域相似度，本文提出了研究领域重合度和研究领域侧重度的概念和模型来综合衡量研究领域相似度。

把作者参与过的所有项目所属的学科集合作为该作者的研究领域，并分别统计在某学科内参与过项目的次数。假设作者和作者的研究领域集合分别为和，定义其研究领域重合度（overlap ratio）为：

 （式3-9）

很显然，两个作者的研究领域的交集占比越大，其研究领域重合度越大；两个作者的研究领域的交集占比越小，其研究领域重合度越小。这个跟实际情况也是相符的。

一般情况，两个作者在重合的研究领域里参与过的项目并不是均匀分布的，各个作者都会有侧重的研究领域。研究领域重合度并不能很好的衡量两个作者的研究领域相似度，为了更好的衡量作者之间的研究领域相似度，本文提出了研究领域侧重度这一概念。

定义两个作者研究领域的交集为，假设，其中表示参与项目所属学科。定义作者在研究领域交集下参与过的项目数量组成的向量分别为和，其中分别表示作者在学科下参与过的项目数量。研究领域侧重度定义如下：

 （式3-10）

可以看出即为两向量夹角的余弦值，向量和的夹角越小，表明作者的研究侧重点越相似，那么研究领域侧重度的值也越大；反之，向量和的夹角越大，表明作者的研究侧重点越偏离，那么研究领域侧重度的值也越小。

上面我们已经分别定义了研究领域重合度和研究领域侧重度来描述作者研究领域的相似情况，下面将这两个指标综合成一个指标来作为研究领域相似度的最终度量。研究领域相似度定义如下：

  （式3-11）

为了避免研究领域侧重度所占比例过大，有如下约束条件：

 （式3-12）

由式3-11联合式3-12可以得到：

 （式3-13）

式3-12针对每一节点对都成立，将所有节点对的和代入上式并求和有： （式3-14）

对式3-14两边同时除以节点对个数有：

  （式3-15）

式3-15等价于：  （式3-16）

亦即有： （式3-17）

其中分别表示的统计均值，通过求的统计均值可以求出参数的范围。

### 推荐系统推荐指标模型设计

通过以上协作关系相似度模型和研究领域相似度模型，可得到某节点和待推荐节点的协作关系相似度和研究领域相似度，把这两个相似度融合为一个最终的推荐指标，然后根据推荐指标的排序进行推荐。

定义最终推荐指标综合相似度为：

 （式3-18）

其中，为协作关系相似度，为研究领域相似度，为脉冲函数，当时；当时。由上述定义可知：

当研究领域相似度时，有；

当研究领域相似度时，有

分别对合作关系相似度和研究领域相似度取以自然常数为底的指数，是为了避免相似度为0时最终得到的结果为0。指数减1，是为了对整个结果进行归一化。参数的确定方法可以参考研究领域相似度融合时采用的方法。

### 推荐指标计算与存储方案设计

本课题选取2010-2016年度全国高校社会科学立项项目24195条，相关人员数118187条。经过数据清洗后保留立项项目数19232条，相关科研人员数93045条。每个立项项目平均有4.84个人参加，每一个作者三度范围内平均有个作者，这意味着平均可以给每一个作者推荐个潜在合作者。那么所有科研人员共计可以产生大约个推荐结果，该估计值在690万左右。那么如何高效计算、存储以及查询如此大量的数据显得尤为重要。

1. 推荐指标计算方案设计

为了提高运算效率和节省运算时间，本课题采取多机多线程的形式，将推荐模型服务部署到多台主机上，每一个线程负责部分作者的推荐结果的计算。为了使集群上的每一个节点任务均匀分配，采用Ngnix对任务请求进行负载均衡，每一个节点接受到请求后都会开启两个线程来处理自己的任务。具体任务部署如下：

表3-5 推荐任务部署

|  |  |  |
| --- | --- | --- |
| 主机名称 | 线程数量 | 任务分段 |
| Node1 | 2 | 1-9999 |
| 10000-19999 |
| Node2 | 2 | 20000-29999 |
| 30000-39999 |
| Node3 | 2 | 40000-49999 |
| 50000-59999 |
| Node4 | 2 | 60000-69999 |
| 70000-79999 |
| Node5 | 2 | 80000-89999 |
| 90000-93045 |

每一台主机运行两个线程，各自负责10000个作者推荐结果的计算，任务分段起始标记即为作者id。例如：集群节点Node1的线程1负责计算作者id在1-9999的推荐结果，线程2负责计算作者id在10000-19999的推荐结果。推荐结果计算出来后统一存入到数据服务器上。任务计算方案集群环境技术架构图如图3-12所示。



图3-12 数据计算方案集群环境技术架构图

发送task任务请求后，代理Nginx通过轮询算法实现负载均衡，将task任务分配到集群不同的节点，节点接受到任务后开启两个线程分段执行任务，将计算的推荐结果存入到数据服务器。

1. 数据存储方案设计

通过分析可知，推荐结果数据大概在690万条。如果把这690万条数据存储在一个表中，那么查询效率是极其低下的，在计算推荐结果任务时对表的访问也是极其低效的。可以采取分库分表或数据库服务器集群化的措施来提高存储及查询效率，此外还可以对推荐结果建立索引以提高查询速度。

考虑到实际情况和推荐结果数据的大小，我们采用分表存储和索引技术来进行查询和存储优化。上述表3-5中每一个节点对应的任务产生的推荐结果将会存储到不同的表中，每一段都包含10000个作者id，大概会产生69万条数据，我们把这69万条数据存入推荐结果表中，并在作者id上建立索引以提高查询速度。推荐结果存储规划如表3-6所示：

表3-6 推荐结果分表存储

|  |  |  |  |
| --- | --- | --- | --- |
| 数据表 | 作者id段 | 数据表 | 作者id段 |
| user\_recommendation\_00 | 1-9999 | user\_recommendation\_01 | 10000-19999 |
| user\_recommendation\_02 | 20000-29999 | user\_recommendation\_03 | 30000-39999 |
| user\_recommendation\_04 | 40000-49999 | user\_recommendation\_05 | 50000-59999 |
| user\_recommendation\_06 | 60000-69999 | user\_recommendation\_07 | 70000-79999 |
| user\_recommendation\_08 | 80000-89999 | user\_recommendation\_09 | 90000-99999 |

通过分表和索引技术，在查询某作者的推荐结果时，不用再在690万条数据中查询，而只需通过作者id直接定位到对应的表中根据索引查询推荐结果，其查询范围缩小了10倍，查询效率可以提高数百倍甚至数千倍。同时在集群节点进行数据存储时，每一个任务只需要对相应的表进行I/O操作，避免了同时对一个表进行I/O带来的效率低下问题。

## 本章小结

本章对推荐系统主要模块和模型进行了分析和设计。首先，对推荐系统服务提供端和服务消费端进行了分析；紧接着对系统的的整体架构进行了设计，主要包括功能架构和系统架构；其次，着重围绕着服务提供端的相关模型和算法进行了介绍和设计，包括数据清洗模块的设计、协作关系网络模型设计、推荐系统模型设计和推荐指标计算及存储方案的设计。其中，数据清洗模块的设计包括对数据来源及现状的分析、数据清洗规则的设计和数据清洗接口的设计等；协作关系网络模型设计主要包括网络模型选择、协作关系权重设计和协作关系网络模型构建等；推荐系统模型设计主要包括协作关系相似度模型设计和研究领域相似度模型设计，协作关系相似度模型主要是提出了协作关系的度量方式和算法——基于局部不定长路径的协作关系相似度模型，研究领域相似度主要是提出了研究领域重合度和研究领域侧重度的度量方式——基于研究领域重合度和研究领域侧重度的研究领域相似度模型。

# 关系抽取实现

在上一章中，详细介绍了推荐系统的总体需求，并从整体架构到具体模块和模型的设计进行了深入的分析，夯实了推荐系统实现的理论基础。基于上一章的分析和设计，本章将对推荐系统的整体环境和框架搭建到具体模块和模型的实现做了详细介绍。

## 开发环境搭建

1）开发工具与框架

推荐系统主要分为服务提供端、服务消费端和客户端。下面分别介绍推荐系统的开发环境，如表4-1所示：

表4-1 推荐系统开发环境

|  |  |
| --- | --- |
| 工具/环境 | 软件名称/版本 |
| 服务提供端IDE | PyCharm 5.0.3 |
| Python | Python 3.4 |
| 服务消费端IDE | MyEclipse 10.0 |
| JDK | Sun JDK 1.8.0\_25 |
| 应用服务器 | apache-tomcat-7.0.72 |
| 开发服务器 | apache-tomcat-7.0.72 |
| 服务器系统 | Cent OS 5.7 |
| 数据库 | Oracle 11gR2 |
| 客户端软件 | 市面上常见的浏览器 |

推荐系统框架配置如表4.2所示，其中前五项为服务消费端所需组件，后三项为服务提供端所需组件。

表4.2 推荐系统框架配置

|  |  |
| --- | --- |
| 框架/组件 | 版本 |
| Highcharts | 3.0.6 |
| D3 | 3.3.9 |
| Struts2 | 2.3.15 |
| Spring | 3.0.5 |
| Hibernate | 4.1.9 |
| Networkx | 1.11 |
| Numpy | 1.8.1 |
| Scipy | 0.15.1 |

2）负载均衡配置

根据3.5.4节推荐结果数据计算方案分析和设计，我们把推荐结果计算服务部署到5个服务节点上，其IP为192.168.88.176-192.168.88.180。由一台服务器作为反向代理利用Ngnix作为负载均衡，其IP为192.168.88.160。

其核心配置如下：

|  |
| --- |
|  |

## 数据清洗模块实现

数据清洗是建模的第一步，数据清洗模块实现主要包括数据清洗规则和数据清洗接口的实现，图4-1为数据清洗模块流程示意图。



图4-1数据清洗模块流程图

第一步：用户进入数据清洗配置页面，配置数据清洗的相关信息，包括数据开始年份和数据结束年份等。

第二步：服务消费端接受到请求和参数后开始按照数据清洗规则进行数据清洗。在数据清洗完成后，进行人员数据的抽取及组装、研究领域数据抽取及组装和合作关系数据处理等操作，最后将三类数据推送至服务提供端。

第三步：服务提供端接受到推送数据后，对数据进行解析并入库。

其中服务消费端数据清洗主要是数据清洗规则的实现，人员数据的抽取及组装、研究领域数据抽取及组装和合作关系数据是数据清洗规则的后续实现，为调用数据推送接口做的准备。各类数据按照定义好的格式组装好后，通过调用统一接口推送数据至服务提供端，由服务提供端解析数据并入库，整个数据清洗流程结束。

### 数据清洗规则

根据3.3.2节中的数据清洗规则，其数据清洗规则实现框图如图4-2所示。



图4-2 数据清洗规则实现流程图

详细步骤如下：

第一步：进入数据清洗配置页面，选择数据清洗的年度区间，即配置起始年份和结束年份。

第二步：客户端将带有参数的请求发送至服务消费端，服务消费端根据配置的参数抽取该年度范围内的立项项目id集合applicationIds。

第三步：遍历立项项目id集合applicationIds，如果遍历完成则整个清洗过程结束，否则进行第四步。

第四步：取出立项项目id，并由项目id取出该项目的所有参与者members。如果参与者数量小于2，则返回到第三步；否则进行第五步。

第五步：遍历参与者集合members，如果已经遍历完成则返回至第三步，否则进行第六步。

第六步：取出成员member并查询其参与过项目的次数。如果参与过项目次数为1，则跳转至第五步；否则存储memberId、项目id与members的关系。

为了保证筛选后人员id的唯一性，其中memberId是以Set<String>的结构存储的；为保存项目id与人员id的对应关系和人员在项目参与过程中的合作次序，本文采用Map<String,List<String>>数据结构存储，其中key对应项目id,List<String>存储项目对应的人员id,并且其次序跟合作次序一致。

### 数据清洗接口

根据图3-5数据清洗接口示意图可知，数据清洗接口主要包含数据抽取、数据组装、数据交互等接口，其中服务消费端主要实现数据清洗规则、数据抽取及组装等接口，服务提供端主要实现数据交互接口。数据清洗规则接口包含了数据清洗及相应数据的抽取实现；数据抽取及组装接口主要是将数据封装为定义的格式；数据交互接口是由Thrift契约定义生成的，主要是在服务提供端实现数据的解析和入库操操作，服务消费端直接远程调用数据交互接口推送相关数据至服务提供端；人员数据和研究领域数据格式均由Thrift契约定义生成。上一小节已经介绍了数据清洗规则的实现，本小节主要介绍契约定义、数据抽取封装接口和数据交互接口的实现。

1. Thrift契约定义

首先通过Thrift定义服务消费端和服务提供端的通信契约，然后根据契约生成各自的接口和类。Thrift契约server.thrift文件定义如下：

|  |
| --- |
|  |

struct ResearchInfo定义的是研究领域ResearchInfo类，struct Mebmber定义的人员Mebmber类，service Server定义的是通信接口。通过运行命令

thrift –gen java server.thrift

thrift –gen py server.thrift

可以分别生成Java端和Python端接口和类。其接口分别如下：

Server.java生成接口源代码：

|  |
| --- |
|  |

Server.py生成接口源代码：

|  |
| --- |
|  |

1. 数据抽取组装接口实现

数据抽取及组装接口主要包含人员数据抽取及组装和研究领域数据抽取及组装。由于人员协作数据需要人员数据入库后的人员id，故人员协作关系数据直接传递初始关系数据至服务提供端，由服务提供端进行数据解析、转换、封装和 入库操作。人员数据和研究领域数据抽取及组装实现框图如图4-3所示。



图4-3 数据抽取及组装实现框图

服务消费端在数据清洗阶段主要是抽取、筛选和组装数据，数据组装成约定的格式后就可以推送至服务提供端进行数据入库。

其中数据的清洗和筛选已经在数据清洗规则实现一节介绍过，此处主要介绍数据的组装和数据推送的一些实现。

a）人员数据组装及推送

经过数据清洗规则后，可以得到保留的人员id集合uids。我们根据该人员id集合uids抽取出相应的人员信息，组装成Member对象并加入到List集合。人员数据组装成功后推送到服务提供端进行数据入库。核心代码如下：

|  |
| --- |
|  |

b）协作关系数据推送

协作关系数据是由项目id和项目参与者id组成的Map集合，直接将该协作关系推送至服务提供端，由服务提供端进行二次数据抽取、转换和组装。核心代码如下：

|  |
| --- |
|  |

c）研究领域数据组装及推送

首先根据保留的人员id集合uids抽取出对应的项目所属学科的集合作为该人员的研究领域集合，并组装成契约定义的ResearchInfos类的形式，最后将其推送至服务提供端进行解析入库。核心代码如下：

|  |
| --- |
|  |

1. 数据交互接口实现

数据交互接口主要定义了服务消费端跟服务提供端进行通信的标准和数据格式,服务消费端利用Thrift框架提供的底层通信机制跟服务提供端进行交互，数据交互接口主要在服务提供端实现，其框图如图4-4所示。

 图4-4数据交互接口框图

服务提供端接收服务消费端推送的数据并进行相应的处理。数据交互接口的实现主要是在服务提供端实现的，包括人员数据、研究领域数据和协作关系数据等的解析和入库等操作。服务消费端通过Thrift框架提供的通信机制与服务提供端进行RPC通信。

服务提供端数据交互接口实现核心代码如下：

|  |
| --- |
|  |

各个接口均调用了server层的具体实现，server层对各个接口的具体实现就是解析数据，然后插入进数据库,最后返回是否插入成功等信息。这样服务提供端就可以不用关注数据来源，只要数据格式符合给定的格式就可以对外提供服务。

## 协作网络建模实现

协作关系网络建模实现主要包括数据初始化、协作关系权重的计算和协作关系网络构建的实现，其流程图如图4-5所示。



图4-5 协作关系网络建模流程图

数据初始化主要是实现了加载和解析协作关系数据，将项目的协作关系组装为特定格式，以便于生成网络边信息；协作关系权重计算主要是实现了边权重的计算；协作关系网络构建主要是实现利用已有的边信息和权重信息进行协作网络的构建。

### 协作数据初始化

在整个推荐系统建模和对外提供服务过程中会多次用到协作关系数据，为了减小数据库的读写压力和便于管理，我们把数据库中的协作关系数据提取出来，按照一定格式以文本文件的形式存储，以后要用到协作关系数据时直接读取文件并解析即可。其文本格式如下：



每一行数据都代表着一个项目的所有参与者和项目年份，“-”将整行数据分隔成了三部分，第一部分表示该对应的作者是项目参与者中的第一作者；第二部分由“;”分隔，代表的是项目参与者中的非第一作者，它们的位置次序代表了协作时的作者顺序，例如是第二作者，是第三作者；第三部分为项目协作年份，用于计算该项目的协作关系衰减因子。

协作关系数据初始化包含数据加载和数据解析，首先从文本文件中加载所有数据，然后遍历每一行数据，每一条数据经解析后都会计算相应边的权重并加入关系网络，所有的数据初始化完毕后建模就完成了，其实现流程图如图4-6所示。



图4-6 协作关系数据初始化实现流程图

第一步：加载文本文件中的协作关系数据datas。

第二步：遍历协作关系数据datas，如果还有数据则转至第三步；否则整个建模过程结束。

第三步：取出一条数据并解析，根据数据格式分隔数据后，将项目所有的参与者按照协作次序加入list集合。将list数据和项目协作年份传至权重计算和网络建模模块。

数据加载核心代码如下：

|  |
| --- |
|  |

数据解析核心代码如下：

|  |
| --- |
|  |

### 协作关系权重

数据加载过程中，根据作者集合生成边的信息计算边权重，并结合项目协作年份计算衰减因子，最后计算衰减后的权重，并把该衰减后的权重作为初步构建网络模型中边的权重。计算协作关系权重具体实现流程图如4-7所示。



图4-7 协作关系权重计算实现流程图

第一步：根据协作年份结合式3-2计算协作关系衰减因子，并根据作者集合authors生成边的信息，其中为作者对应的协作顺序，且有。

第二步：根据式3-1计算边的权重

第三步：根据式3-3计算边的最终权重，并将权重作为边(author[i],author[j])的权重初步构建网络。

计算权重核心代码如下：

|  |
| --- |
|  |

上述两个for循环就是边的生成规则：按照排列组合的方式生成边，且满足。然后计算边的权重，最后计算衰减后的协作关系作为构建网络的边权。

### 协作关系网络

在数据初始化以及协作关系权重等信息计算完成后，利用networkx的Python库就可以很容易的构建一个无向有权图，该无向有权图表征协作关系网络。其协作关系网络构建的实现流程图如图4-8所示。



图4-8 协作关系网络构建实现流程图

第一步：利用之前已经生成的边信息和边权信息作为输入，判断现有协作关系网络模型中是否存在该输入的边。

第二步：如果网络模型中存在该边，那么取出该边的边权，并重置网络模型中该边的边权为该边在当前项目中的边权和网络模型中该边的边权之和。

第三步：如果网络模型中该边不存在，那么直接把边信息和对应的权重信息加入到网络模型中。

协作关系网络构建核心代码：

|  |
| --- |
|  |

## 协作推荐模型实现

协作推荐模型的实现主要包含协作关系相似度模型和研究领域相似度模型的实现、推荐系统的推荐指标以及推荐结果计算与存储方案的实现，下面分别予以介绍。

### 协作关系相似度模型

协作关系相似度模型是以协作关系网络为基础建立的。首先，进行网络初始化，加载并解析协作关系数据构建协作关系网络；其次，对给定节点的三度范围内的节点进行预处理，将局部关系网络图表示为邻接表的形式；再次，遍历三度范围内节点，针对每一个节点与给定节点组成的节点对找出其路径长度在内的所有路径；最后，计算节点对的每一条路径的相似度累加和，将其作为节点最终的协作关系相似度。具体流程框图如图4-9所示。

 图4-9协作关系相似度模型流程图

由于网络初始化主要是协作关系网络的建模，其实现在4.3节有详细的介绍，本节主要介绍节点的预处理、全路径寻址和相似度计算的具体实现。

1. 节点预处理实现

节点预处理主要是取出给定节点的三度范围内的所有节点，并把该局部关系网络表示成邻接表的形式。局部关系网络表示成邻接表主要是为了满足全路径寻址算法的输入要求。节点预处理实现框图如图4-10所示。



图4-10节点预处理实现流程图

首先给定一个作者author，在关系网络图中寻找其邻接点即为一度节点，将其加入nodes集合。针对每一个一度节点，再寻找其邻节点，即为author的二度节点，将其加入nodes集合。针对每一个二度节点，再寻找其邻节点，即为author的三度节点，将其加入nodes。这个过程结束以后就得到了author的所有三度范围内的节点。

遍历author三度范围内的节点，取出节点node的邻接点并用字典表示为dict[node]=neighbors的形式。

核心代码如下：

|  |
| --- |
|  |

1. 全路径寻址实现

全路径寻址就是要找出两节点之间符合条件的所有路径。经过上述节点的预处理流程后，对于给定节点*node*的三度范围内的关系已经表示成为了邻接表的形式。在该三度范围内给定一个节点对，要找出从节点到节点路径长度不大于的所有可达路径，可以采取深度优先搜索结合回溯法进行寻址。其主要思想如下：

第一步：把当前节点加入路径节点集合*path*末尾，并做以下校验：①路径*path*长度是否大于给定限制路径长度，若大于则回溯至的上一层节点；②节点是否跟目标节点*end*相等，若相等则返回该条路径*path*；③节点是否为叶子节点，若为叶子节点则回溯至的上一层节点。若校验后没有返回值，则转至第二步。

第二步：取出节点的邻接点，校验节点是否在路径*path*中已经存在，若节点在路径*path*中已经存在，则说明存在环路，此时不对做任何处理，直接处理的下一个邻接点并把其当做新的节点；若节点不在路径*path*中，则把节点置为当前节点重复第一步。

第三步：当回溯至节点时，表明从节点至节点的某一邻接点的所有路径寻找完毕，将该返回的路径加入*paths*，以同样的方式处理节点的其他邻接点。节点的所有邻接点处理完毕后返回最终的路径*paths*。

图4-11所示为包含6个节点的网络图，假设需要寻找节点之间路径长度在3以内的所有可达路径，即，图4-11中虚线和数字标识寻找方式和顺序。



图4-11 全路径寻址示意图

首先将当前节点=放入路径*path*的末尾，做三层校验通过后将的邻接点作为当前节点做同样的处理，此时路径*path*中顺序存在节点。将节点的邻接点作为当前节点做同样的处理，此时路径*path*中顺序存在节点。将节点的邻接点作为当前节点做同样处理，发现在校验路径长度时其路径*path*顺序存在节点，其长度为4，大于限制长度3，故返回至上一层节点。此时取的邻接点（因节点已经包含在*path*中了，故不需重复处理）做校验时发现该节点等于，说明已经成功找到一条路径*path*，其中顺序存在节点，返回该路径*path*。由于节点的邻接点已经遍历完毕，故将返回的路径*path*加入到路径集合*paths*中，并返回至的上一层节点。接下来遍历的其它邻接点，此处即为节点，做校验时发现该节点跟节点相同，说明成功找到一条路径*path*,其中顺序包含节点，返回该路径*path*。由于节点的所有路径已经遍历完毕，故将返回的*path*加入路径集合*paths*，并返回至上一层节点。

对的其他邻接点，此处为做同样的处理，会找到一条路径*path*，其顺序包含节点。由于节点的路径已经遍历完毕，故将返回的路径*path*加入路径集合*paths*,并返回至上一层节点。

经过以上处理，节点的所有路径均已遍历完成，此时返回最终的路径集合*paths*。

其核心实现代码如下：

|  |
| --- |
|  |

第2行是把当前节点顺序加入*path*变量

第3-8行是做的三层校验，分别校验了长度、是否成功找到路径、是否是叶子节点。满足校验条件就会返回至上一层节点进行下一次递归。

第9行是当前递归的路径集合变量，用来存储经过当前节点的所有符合条件的路径。

第10行是处理当前节点的邻节点。

第11行保证了不会形成环路

第12行把当前节点的邻接点视作当前节点进行递归操作

第13-15行把返回的符合条件的路径加入路径集合变量*paths*.

第16行当前节点的所有邻接点遍历完毕就返回经过当前节点的路径集合*paths*作为上一层的路径集合的子集。

1. 相似度计算实现

计算给定*author*和节点*node*的协作关系相似度就是要使用以上全路径寻址算法找出两节点间路程长度在内的所有路径*paths*，然后根据3.3.1节提出的相似度模型计算所有路径的相似度累加和作为两节点协作关系相似度。具体实现如图4-12所示。



图 4-12协作关系相似度实现流程图

第一步：初始化相似度值。

第二步：遍历两节点之间的路径*paths*，如果路径遍历完毕则直接返回相似度；否则进行第三步。

第三步：取出路径*path*，令下标，单路径相似度r=1。

第四步：判断下标是否越界，若越界则单路径r相似度累加至最终相似度s上，并转至第二步；否则取出相邻两节点。

第五步：根据式3-3计算边权，根据式3-4计算权重，根据式3-7计算单路径相似r，同时下标i加1。转至第四步。

### 研究领域相似度模型

由3.5.2节知，节点对的研究领域相似度由其研究邻域重合度和研究领域侧重度融合而成，研究领域相似度实现流程图如图4-13所示。



图4-13研究领域相似度实现流程图

对于给定的节点对，首先分别找出节点参与的所有项目所属学科组成的集合作为其各自的研究领域和，然后分别计算出其研究领域交集和并集。

按式3-9计算研究邻域重合度

按式3-10计算研究领域重合度

分别求出的统计均值。将统计均值代入式3-17：可以求出，取，即有研究领域相似度为：

 （式4-1）

### 推荐系统推荐指标

推荐指标是由研究领域相似度和协作关系相似度融合而成，其实现流程图如图4-14所示。



图4-14推荐指标融合实现流程图

在3.5.3节中推荐指标综合相似度为：



当研究领域相似度时，有；

当研究领域相似度时，有

其实现核心代码如下：

|  |
| --- |
|  |

### 推荐指标计算与存储方案

在3.6推荐指标计算和存储方案设计一节分析了推荐系统的数据量，并从数据计算和数据存储方面给出了设计方案。本节主要分别介绍数据计算方案和数据存储方案的具体实现，其中数据计算方案的实现主要介绍多线程下推荐结果数据的计算；数据存储方案主要介绍了数据存储结构、数据入库和数据查询等的具体实现。

1. 数据计算方案实现

数据计算所需的平台搭建在4.1.2小节已经介绍过，该小节主要介绍多线程下推荐结果数据的计算实现。

推荐结果数据计算的核心代码如下：

|  |
| --- |
|  |

第一步：获取锁，以便保持后续操作的原子性。

第二步：传入节点id，根据推荐模型计算推荐结果信息，主要是找节点三度范围内的所有节点nodes，根据模型计算 id跟nodes组成节点对的协作关系相似度、研究领域重合度和研究领域侧重度。

第三步：组装数据。将上述通过推荐模型计算的数据组装成特定的格式以便后续批量插入数据库。

第四步：释放锁，以便其他的线程执行计算任务。

第五步：将以上推荐结果组装后的数据插入对应的数据库。

上述代码为推荐结果数据计算的底层实现，下面介绍多线程处理推荐结果计算，核心代码如下：

|  |
| --- |
|  |

第一步：接收推荐结果计算任务后，加载协作关系数据并创建协作关系网络图。

第二步：根据参数起始任务段来批量获取相应段的作者id。

第三步：创建线程并将推荐结果数据计算方法跟该线程关联，同时传入该段的作者id集合。

第四步：启动线程。

1. 数据存储方案实现

根据3.5.4节分析可知，推荐结果总数据量在690万左右(实际推荐结果总数约为680万)，为了提高存储效率和查询效率，我们采用了分表和数据库索引技术。

存储推荐结果数据的表结构和分表存储如表3-4和表3-6所示。其数据库中分表存储设计示意图如图4-15所示。

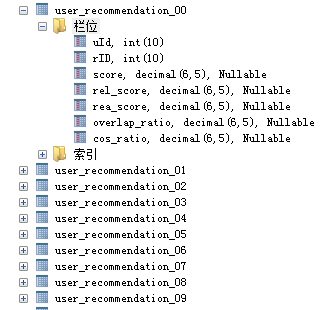


图4-15 推荐结果数据分表存储示意图

在计算不同任务段的作者推荐数据时，根据作者id所属段来将其存储至对应的分表中，其对应关系见表3-6，其对应关系核心代码实现如下：

|  |
| --- |
|  |

给定作者的推荐结果数据计算完成后就组装成为特定格式，然后进行批量插入，其核心代码如下：

|  |
| --- |
|  |

上述是作者id所属段，有。根据获取其对应存入的表名，然后执行批量插入操作。

根据3.5.4节分析可知，每一个分表的数据量在69万左右。如果在建表时不做任何处理，对推荐结果进行查询时每一次都需要全表扫描，这样查询操作非常耗时、效率极其低下。为了解决查询效率耗时和效率低下的问题，除了采用分表技术以外，本文还采用了数据库的索引技术。在每一个分表的uid字段上都建立索引，这样在通过uid字段查询推荐结果时就不需要进行全表扫描。需要注意的是，对分表uid字段建立索引是要在该分表所有推荐结果数据入库以后建立，如果在建立分表的时候就建立索引，那么每一次推荐结果的入库都会造成该索引的更新，使得索引的维护成本极高，从而降低了数据库的性能进而影响整个推荐结果计算的性能和存储的性能。

在分表uid字段建立索引代码如下：

|  |
| --- |
|  |

user\_recommendation\_00表的uid字段建立了索引，同理可以在其他分表的uid字段建立索引。

## 本章小结

本章详细介绍了基于项目协作关系的推荐系统的各个模块和模型的实现。首先，简单的介绍了开发环境与框架的配置；其次对数据清洗模块的实现做了详细的介绍，包括数据的清洗规则和清洗接口的实现等；然后介绍了协作关系建模的实现，主要包括协作数据初始化、协作关系权重和协作关系网络的实现；最后介绍了协作推荐模型的实现，主要包括协作关系相似度模型、研究领域相似度模型、推荐系统推荐指标模型和推荐指标计算与存储方案的实现，其中协作关系相似度主要分为网络初始化、节点预处理、全路径寻址和协作相似度计算等部分；研究领域相似度主要包括研究领域重合度和研究领域侧重度等；推荐指标主要是协作关系相似度和研究领相似度的融合等。

# 模型评估及案例分析

第三章和第四章详细介绍了推荐系统的设计和实现，本章将从不同的角度对推荐系统进行测试和分析，主要包括系统的性能测试及分析和协作关系网络数据分析两大部分。系统测试及性能分析主要是从系统的基本功能、响应时间和推荐质量来进行分析，协作网络数据分析主要是探索和分析协作关系网络呈现的一系列规律。为了便于描述，本章将采用图片和表格来进行展现和分析测试结果及相关数据。

## 系统测试及性能分析

测试主要是为了保证系统功能足够完整、系统足够健壮、用户界面友好并且响应时间合理。推荐系统的测试主要将采用黑盒测试的方式，同时辅之以白盒进行测试。测试内容主要包括以下几个部分：基本功能测试，即系统对外提供基本推荐功能的测试；响应时间测试，主要测试各推荐功能模块的响应时间；推荐质量分析，主要是对推荐质量进行测试。对于本课题的测试环境，客户端采用Chrome浏览器和Firebug调试工具，服务端采用Mysql数据库和Tomcat应用服务器。

### 系统测试方案

推荐系统的基本功能测试包括作者检索、合作者推荐、协作关系图展现、协作关系数据概况和系统内部功能数据抽取和推荐结果计算等，响应时间测试主要包括基本功能的响应时间，推荐质量分析主要是分析推荐的准确率、召回率和覆盖率。

用户进入合作者推荐页面，通过提供的搜索框检索待推荐的作者，返回检索结果，用户可以选择某一作者查看其推荐结果，推荐结果以列表的形式展现，待推荐者和推荐结果之间的协作关系以图的形式展现出来。用户进入协作网络数据概况导航栏可以查看协作关系网络数据概况。数据抽取和推荐结果计算主要是系统内部功能，配置好数据抽取年份后进行数据清洗、数据抽取、数据推送、数据解析和数据入库，推荐结果计算主要是采用了多机多线程的和分表的技术。

对系统基本功能和系统内部功能进行测试并记录下请求响应的时间。

### 系统测试结果分析

1. 基本功能与响应时间测试结果分析

根据5.1.1节中描述的测试方案对推荐系统进行测试，基本功能测试结果如表5.1所示。

表5.1基本功能测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 功能 | 输入与动作 | 预期响应 | 实际响应 | 是否通过 |
| 01 | 进入推荐页面 | 点击菜单栏 | 显示作者检索框 | 显示作者检索框 | 是 |
| 02 | 作者检索 | 输入作者并检索 | 显示检索的相关作者列表 | 相关作者列表展示 | 是 |
| 03 | 作者推荐 | 点击按钮触发推荐操作 | 展现推荐作者列表和关系图 | 推荐结果列表展示 | 是 |
| 展示作者关系图 | 是 |
| 04 | 数据概况展现 | 点击页面导航栏 | 展示相关数据图表 | 展示相关数据图表 | 是 |
| 05 | 抽取数据 | 配置参数提交 | 提示数据抽取成功 | 提示数据抽取成功 | 是 |

表5.1所示的测试结果显示，推荐系统的基本功能均正常运行。用户点击和输入操作也的到正常响应。表5.2为以上基本功能的响应时间。

表5.2 基本功能响应时间

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 功能 | 处理数据量 | 平均响应时间（ms） |
| 01 | 作者检索 | 数百条 | 32 |
| 02 | 推荐作者列表 | 数百条 | 92 |
| 03 | 推荐作者关系图 | 数百条 | 98 |
| 04 | 数据概况展现 | 数百条 | 24 |
| 05 | 抽取数据 | 数十万条 | 256000 |

除了数据抽取的耗时较长之外，其他响应时间均在100ms以内，极大的增强了用户体验，这是由于系统事先已经计算出了推荐结果，推荐服务会直接查询相关结果。由于数据抽取处理数据量较大，且涉及流程较多故其耗时较长，但是数据抽取只在数据更新后由系统维护人员进行操作，并不影响用户体验。

整个推荐系统耗时最大的就是推荐结果的计算，其处理数据量达到数百万条甚至数千万条，共计有近十万节数据节点，以3000数据节点为单位进行推荐结果计算测试，取多次测试的平均值作为在该不同环境下的测试的结果，测试结果如表5.3所示。

表5.3推荐结果计算耗时测试

|  |  |  |  |
| --- | --- | --- | --- |
| 环境 | 节点数 | 实际推荐结果平均值条数 | 平均耗时（h） |
| 单机单线程 | 3000 | 237500 | 18.25 |
| 单机多线程（2线程） | 3000 | 226197 | 12.62 |
| 集群多线程（2线程） | 3000 | 221965 | 2.53 |

根据表5.3计算结果可以看出通过使用集群多线程可以大大的提高计算效率，集群数和线程数取决于硬件资源。

图5-1至图5-3为推荐系统推荐结果可视化（部分敏感信息做打码处理）。



图5-1 待推荐作者检索结果

在图5-1中检索结果列表为用户键入的待推荐的作者相关信息，包含了作者姓名和作者所属机构用以区分同名的不同作者。最后一列是操作列，点击可以查看推荐给该作者的潜在合作者信息，如图5-2所示。

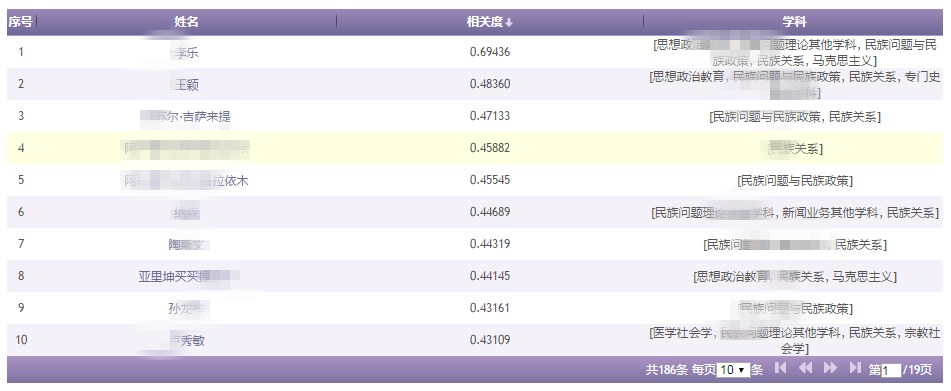


图5-2推荐结果列表

图5-2中为某作者的潜在合作者推荐结果，包含了姓名、推荐指标和相应作者的学科，其中按照推荐指标降序排列。

图5-3为某作者跟其潜在合作者在协作关系网络中的协作关系示意图，该图仅展示了该作者在协作关系网络中三度范围内的协作关系。最大的顶点为给定的作者，其他节点为推荐的潜在合作者。

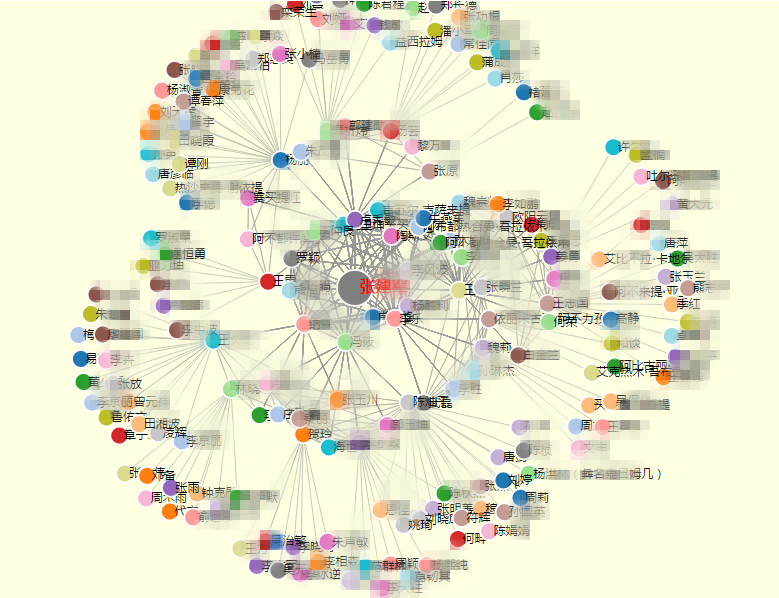


图5-3推荐作者协作关系图

1. 推荐质量分析

推荐质量主要从推荐准确率、召回率和覆盖率三方面分析[[[52]](#endnote-52)]，记推荐结果总人数为,其中推荐且成功合作的人数为，测试集总人数（合作总人数）为，推荐结果去重后总人数为，训练集总人数为，定义准确率，定义召回率为，定义覆盖率。

分别实现了基于协作关系相似度跟研究领域相似度的混合推荐模型（CNR）和基于共同邻域（Jaccard）跟研究领域相似度模型的混合推荐模型（CMR）作为对比，以2017年度社会科学数据为测试集，每一个节点返回推荐结果数以5递增，其模型准确率、召回率和覆盖率如图5-4、5-5和5-6所示：

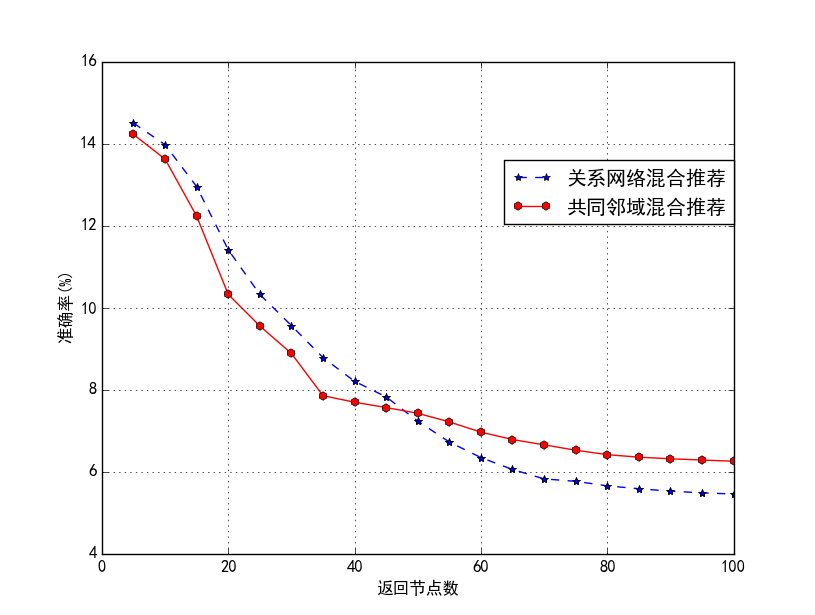


图5-4 准确率随推荐返回节点数的变化

由图5-4可知，在返回推荐节点数小于50时，本文提出的推荐模型准确率优于基于共同邻域的混合推荐模型，说明本文提出的模型在二度范围内具有较好的推荐性能；但在推荐节点数大于50时，基于共同邻域的混合推荐模型准确率优于本文提出的基于协作关系相似度跟研究领域相似度的混合推荐模型，且基于共同邻域的混合推荐模型准确率变化较小。这是因为基于共同邻域的混合推荐模型分析的是二度范围内的节点，当推荐节点数大于50时，这些节点大多数属于三度节点，而该模型处理不了三度节点，对最终结果影响较小，故呈现出准确率变化较小的趋势。另一方面，本文提出的模型具有较大的推荐范围，在返回推荐节点数大于50时，其对推荐准确率的影响较大（返回推荐节点数越多，其准确率会降低）。

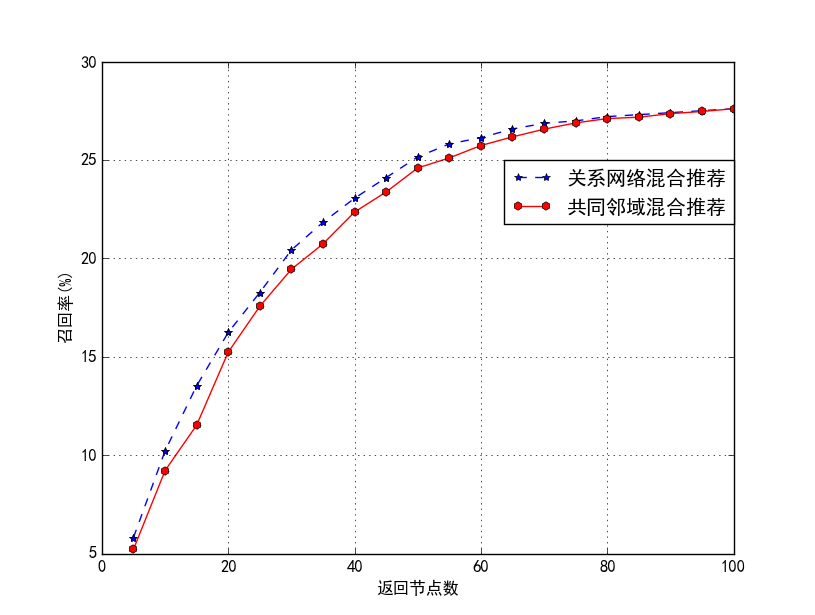


图5-5 召回率随推荐返回节点数的变化

由图5-5可知，召回率随着推荐总节点数的增加而增加，跟准确率的变化趋势相反，可以看出在返回推荐结果数相同的情况下，本文提出的模型召回率优于基于共同邻域的混合推荐模型。

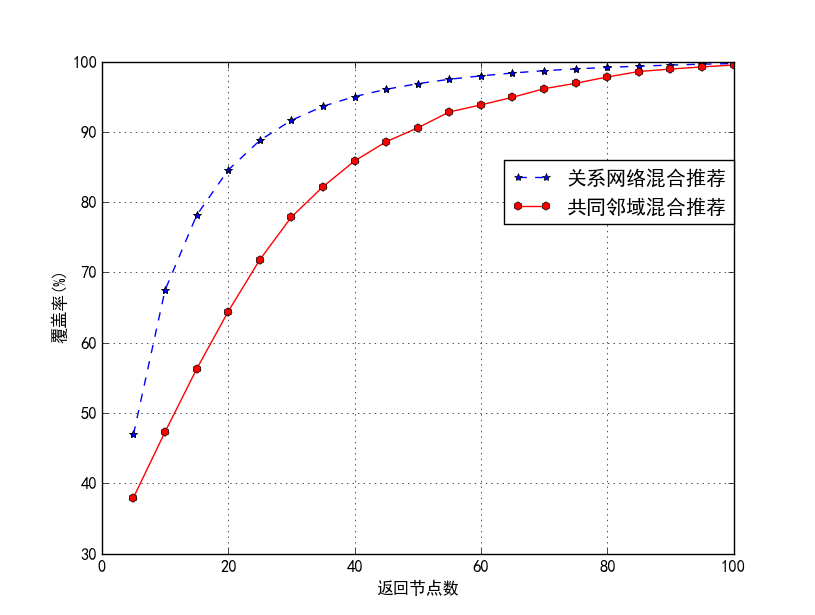


图5-6 覆盖率随推荐返回节点数的变化

由图5-6可知，覆盖率随返回推荐总节点数的增加而增加，初始增加幅度较大，最终趋于100%，本文提出的模型在返回节点数相同的情况下覆盖率优于基于共同邻域的混合推荐模型。结果表明返回推荐节点数越多，其覆盖率会越大，召回率越大，准确度越低。

综合以上三个指标可以看出，本文提出的基于协作关系相似度跟研究领域相似度的混合推荐模型总体要优于基于共同邻域（Jaccard）跟研究领域相似度模型的混合推荐模型。

## 协作关系网络探究

在整个推荐系统的设计和实现过程中，我们发现了协作关系网络呈现出的一些有趣的现象，包括协作关系概况数据、连通子图数据和度分布数据等呈现的一些规律，下面分别展示这些现象并尝试解释这些现象。

### 协作关系概况数据分析

协作关系概况数据主要包括协作关系网络历年全图节点数、最大连通子图节点数以及最大连通子图节点数占全图节点数的比例。其各年度数据示意图如图5-7和表5.4所示。

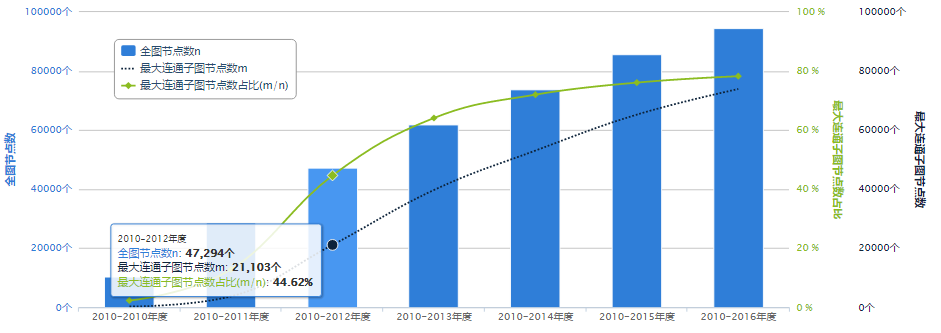


图5-7 协作关系概况数据图

表5-4协作关系概况数据

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 年度 | 全图节点数(n) | 最大连通子图节点数(m)个 | 最大连通子图节点数占比（m/n）% | 占比增量 |
| 2010-2010 | 10297 | 235 | 2.28 | 2.28 |
| 2010-2011 | 28724 | 3682 | 12.82 | 10.54 |
| 2010-2012 | 47294 | 21103 | 44.62 | 31.8 |
| 2010-2013 | 61946 | 39705 | 64.09 | 20 |
| 2010-2014 | 73688 | 53050 | 72.01 | 7.92 |
| 2010-2015 | 85737 | 65235 | 76.08 | 4.07 |
| 2010-2016 | 93045 | 73906 | 79.43 | 3.35 |

通过图5-7可以看出，随着全图节点数的增加，其最大连通子图节点数也在增加，但最大连通子图节点数占比的增量是先大后小。最大连通子图节点数增加是因为新加入的节点有部分是原来最大连通子图中已经存在的节点，这部分节点作为媒介将较小的连通子图融合到大的连通子图中，这样最大连通子图就会不断的增长。

全图节点数的变化和最大连通子图节点数都是呈现出先是增量较大后是增量较小的趋势，导致这一变化趋势的原因可能是：随着人文社会科学的发展，后面立项的作者很有可能在之前已经立项过，在统计作者时不会新增作者。所以随着数据的积累，如果以后立项项目中有很多作者之前已经立项过的，那么其作者的增量不会很大，且其作者增量越小，表明其立项项目越是集中在之前立项过的作者之中，进一步可以推测新一代的社会科学类研究人员科研成果增长缓慢。

### 连通子图数据分析

通过以上协作关系概况可以知道，每一个年度范围内的协作关系网络都存在一个最大连通子图，且该最大连通子图包含了全图绝大多数的节点。为了进一步探究连通子图的的变化规律，我们统计了每个年度范围内协作关系网络包含节点数前十的连通子图，如图5-8所示，横坐标表示子图节点数的排序，纵坐标表示该子图所包含的节点数。

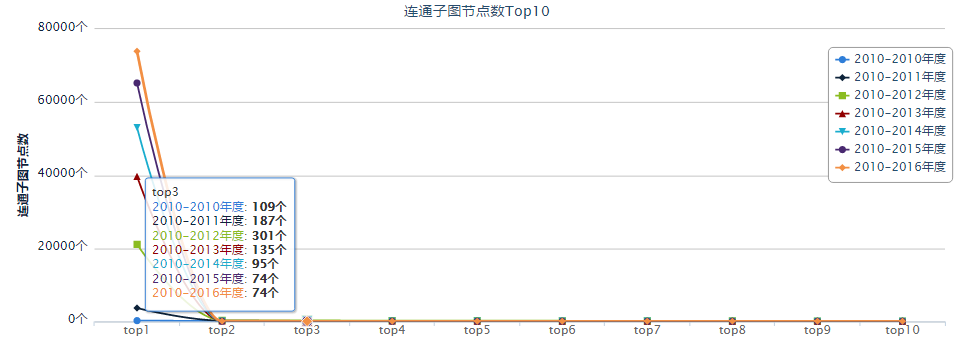


图5-8连通子图节点数Top10

通过图5-8可以看出，每一个协作关系网络中存在一个最大连通子图，并且也仅仅只存在一个巨片（包含节点数较多的连通子图），其他连通子图节点数都较少。另外，除最大连通子图之外的连通子图呈现出越接近当前年份其节点数越少的趋势。

随着社会科学项目数据的积累，新加入的项目参与者很有可能是已经存在于之前的数据之中的，这样在挖掘项目协作关系时，新加入的小的协作关系网络就会通过那些已经存在于协作关系网络中的节点作为媒介而融合成为一个较大的协作关系网络，而那些没有媒介的新加入的较小的协作关系网络就会表现为较小的连通子图，这就会造成具有巨片特征的网络节点数会越来越大。小连通子图在不断的生成，也在不断的被巨片“吞噬“而形成更大的巨片，这就会使整个网络呈现出少数巨片和众多小连通子图的现象。

巨片越少，且巨片节点越多，表明协作关系网络整体较紧密；反之表明协作关系网络整体较松散。图5-8所呈现的巨片仅有一个，且节点数较多，说明该协作关系网络较紧密，这也是我们分析协作关系的基础。

通过图5-8，观察到小连通子图呈现出越接近当前年份其节点数越少的趋势，并且子图节点数从较远年份到当前年份呈现出先增加后减少的趋势。为更好的证实和探究这一趋势，我们选取了每一个年度范围内协作关系网络包含节点数在前三十的小连通子图，如图5-9所示，横坐标为连通子图按节点数排序，纵坐标为该连通子图节点数。

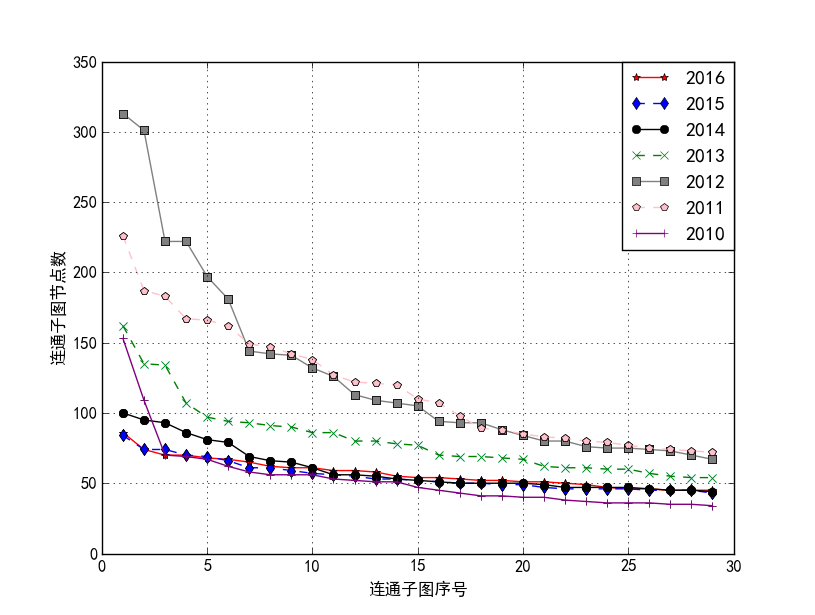


图5-9 连通子图节点数top2-top30

可以看出，2010-2012年连通子图节点数整体基本呈现递增的趋势，但到2013年及以后就呈现出递减的趋势。造成这一现象的原因可能是在数据积累初期，新加入的节点组成的较小连通子图由于缺乏媒介而无法融合到巨片中，就只能以较小的连通子图存在于网络中；随着数据的积累，新加入的节点存在于原来网络中的概率变大，也就是小连通子图跟巨片之间存在媒介的概率变大，也就更容易与巨片融合形成更大巨片，这就造成了一种连通子图被巨片“吞噬”的现象，且连通子图节点数越多，其被巨片“吞噬”的概率越大。

连通子图节点数递增趋势为何会在2013年打破还有待探究。

### 度分布数据分析

以上探究了协作关系网络的宏观特征，本小节着重分析节点度的分布特征，各年度范围内节点度的分布如图5-10所示。

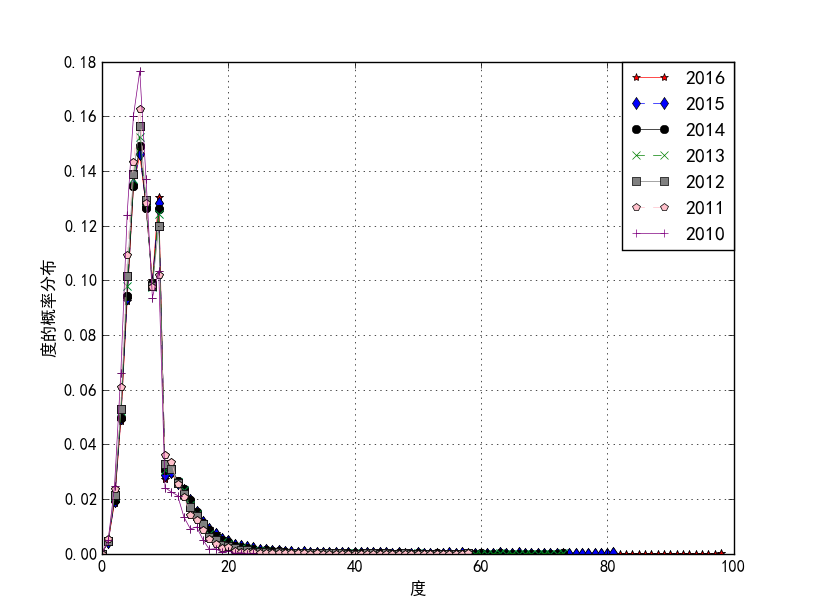


图5-10历年协作关系网络节点度分布

可以看出各年度节点的度大部分都分布在20以内，同时各年度范围内度的分布曲线重合度较好，曲线最高点有向下偏移的趋势，同时曲线尾部有外扩的趋势，见局部图5-11和图5-12。

各年度协作关系网络大部分节点的度分布在20以内，具有较大的度的节点很少，这和统计学现象非常吻合。节点具有较大度在一定程度上可以表明该作者参加项目次数较多，而具有度较大的节点数目极少，这种现象表明项目立项率极高的人极少，这一类人在历年项目立项和项目协作中具有极为重要的地位，极有可能是该领域的权威学者或有重要影响力的学者。通过分析节点的度、节点总权重和总权重对度的平均等因素，结合核心作者和挂名作者在协作关系网络中的特征，建立相应模型可以进行寻找核心作者和挂名作者等应用。

随着年份越接近当前年份，曲线最高点有向下偏移的趋势，同时曲线尾部有外扩的趋势，这表明节点平均度有增大的趋势。随着数据的积累，新加入的节点存在于原来网络中的概率变大，也就是小连通子图跟巨片之间存在媒介的概率变大。在连通子图被巨片“吞噬”的同时媒介的度也在增加，所以随着时间的推移，节点的平均度有增大的趋势，只是这个变化趋势越不明显。

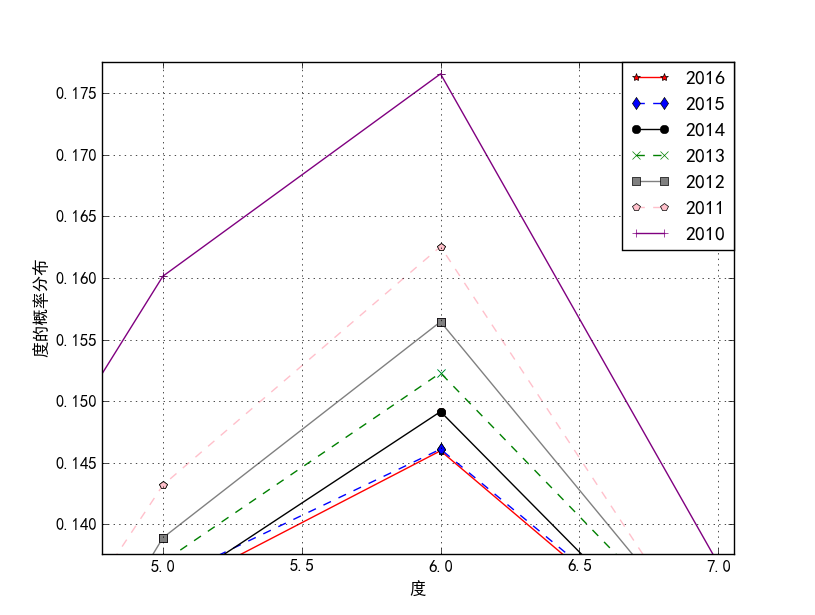


图5-11 曲线最高点局域图

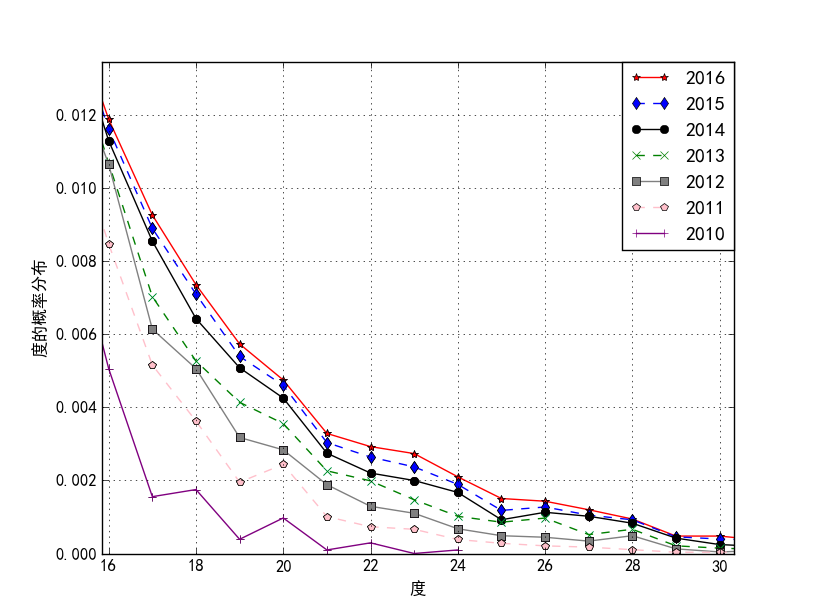


图5-12 曲线尾部局域图

通过对协作关系网络的分析，我们得到各年度范围内节点平均度如表5.5所示：

表5.5各年度范围内节点平均度

|  |  |  |
| --- | --- | --- |
| 年度 | 节点平均度 | 增量 |
| 2010-2010 | 6.61 | - |
| 2010-2011 | 7.11 | 7.56% |
| 2010-2012 | 7.38 | 3.79% |
| 2010-2013 | 7.53 | 2.03% |
| 2010-2014 | 7.71 | 2.39% |
| 2010-2015 | 7.78 | 0.91% |
| 2010-2016 | 7.85 | 0.89% |

可以看出随着时间的推移，节点平均度呈增加的趋势，但时间越接近当前年份，其平均度增量越小，即增长趋势越不明显。因为随着数据的积累，节点数越来越多，充当媒介节点所引起度的增长被大量新增的节点和原有节点所稀释。图5-12中各分布曲线向外扩的趋势逐渐减弱以及各曲线具有较好的重合度也证明了节点平均度的变化趋势越发不明显的特征。可以预见的是：随着数据的积累，曲线外扩程度会越来越小，尾部会逐渐趋于重合。

## 本章小结

本章主要对推荐系统的基本功能进行了测试，并对测试结果和推荐质量进行分析，从对测试结果和推荐质量分析上可知，推荐系统达到了设计要求和预期效果，满足了为科研合作者推荐潜在合作者的服务需求。同时本章尝试对协作关系网络呈现的一系列现象和规律进行探索和分析，为进一步研究协作关系网络夯实基础。

# 总结与展望

## 论文总结

本文是对提高科研管理数据资源利用率和解决广大科研工作者寻求合作者效率和质量低下问题的一个初步尝试，试图通过先进的计算机技术和理论，实现对科研管理数据资源的利用，并在一定程度上提高广大科研人员寻找合作者效率和质量。促进科研人员之间的交流，一定程度上可以起到增强科研成果和协作质量的效果。以下是本文的主要内容：

1）确定了课题需要解决的问题。通过对科研管理研究数据现状和广大科研工作者在寻求高质量合作者过程中遇到的问题进行分析，确定了本课题要解决的实际问题，并对该问题进行了拆分。

2）对相关理论、技术及架构的学习和选型。通过对传统推荐系统的相关理论和技术的学习，并结合本课题所要解决问题的实际情况，确定了本推荐系统所采用的理论原型——基于关系网络的推荐。调研并学习了基于图的相关理论、基于SSH的Web开发框架技术、基于面向架构的Thrift通信技术、基于Python数据处理的Networkx框架、基于前端可视化的Highcharts和D3技术等。

3）系统设计和实现。通过分析传统推荐系统原型在解决本课题实际问题时表现的不足，设计并提出了一套适用于该课题的推荐模型。包括数据清洗的设计、协作网络模型的设计、协作关系相似度和研究领域相似度的设计等。同时基于前期对相关理论和技术的调研、学习和研究，实现了推荐模型的基础框架，并在此框架上对外提供推荐服务和相关的数据分析。

4）系统测试及数据分析。对推荐系统进行了测试并对结果进行了简要的分析，同时针对在系统设计与实现过程中协作关系网络呈现的一些现象和规律做了简要的探究和分析。

综上所述，本文从系统的需求分析、技术及架构选型、系统设计、编码实现、系统测试以及协作关系网络探究等方面介绍了基于项目协作关系的合作者推荐系统的设计与实现过程。该系统借助于挖掘科研管理研究数据中的项目及人员信息，提高了数据的利用率，同时为科研工作者快速高效的寻找合作者提供了服务，能够促进科研人员之间的有效合作和有效交流，进而提高协作质量，在一定程度上为促进学科发展和增强科研质量提供了新的思路和方向，进一步推进了高校信息化平台的建设。

## 论文展望

到目前为止，本课题的研究已经告一阶段，通过基于项目协作关系的合作者推荐系统可以实现对给定作者推荐其潜在合作者，但是由于本人知识水平和时间等因素的限制，仍然还存在一些需要完善和值得探索的地方，现简述如下：

1. 用户反馈收集

目前实现的合作者推荐系统不包含用户对推荐结果的反馈功能。一方面可以通过分析用户对推荐结果的反馈来表征推荐系统的推荐质量；另一方面可以通过分析用户对推荐结果的反馈来完善推荐模型和调节参数，使得推荐系统更好的为用户服务。下一步可以考虑加入收集用户推荐结果反馈和反馈结果分析功能，为进一步优化模型和提高推荐质量服务。

1. 协作关系网络探究

目前仅对协作关系网络进行了简要的探究和应用，后续通过深入学习网络科学理论，可以进一步分析协作关系网络的生长、演变状况以及群组的生成发展机制；通过对网络社团结构的分析和检测，可以分析网络社团的构成情况；通过分析节点度的分布、节点强度等因素，可以进一步推荐核心作者和检测挂名作者等。除了提及到的探索方向之外，还有众多的未知的方向值得探索，但是以上任何一个方向都具有进一步分析和探究的意义。

1. 应用的深化

目前合作者推荐系统仅仅只是对给定的作者进行潜在合作者推荐，我们可以深化应用，通过推荐系统把该平台打造成科研合作者的交友平台，使得科研人员之间除了项目协作交流之外还可以进行在线交流，极大的促进了科研人员之间的交流，有助于新的科研成果、科研创新和研究方向的产生。

# 致 谢

弹指一挥间，时光如白驹过隙，转眼间充实而又忙碌的三年研究生生涯即将结束。在这三年的学习生活中，有过喜悦、也有过悲伤、有过成功、也有过挫折、有过付出、也有过回报。研究生期间不仅积累了专业知识，还学会了对人生的思考，不仅开拓了视野，还学会了为人处世，这些都将成为宝贵的财富，陪伴着我。在此，我衷心的感谢这三年来默默支持、关心和鼓励我的人。

首先，要感谢我的导师王玉明副教授，本论文在是王老师悉心指导下完成的。王老师治学严谨、待人真诚、学识渊博，不仅在学习上给予指导，在生活中也给予我莫大帮助和支持。王老师教会了我如何思考，如何做事，如何沟通，这些让我终生受益。在此，向王老师表示由衷的感谢。

感谢曾经帮助过我的老师和同学，在我最迷茫的时候，有你们的支持和鼓励，使得我能够不断前行。感谢实验室和宿舍的小伙伴们，他们是冯哲奇、彭亮、张楠、刘甲、林晓佳、冯子健、黄骏、姚雨涛、欧霄、钟远维、汪贝贝、黄钊伟、涂云、黄山、佘健夫等。你们在生活上给予我莫大的包容，在学习上给予我很大的帮助和支持，是你们使我这三年的研究生生活丰富多彩，祝愿大家在以后的日子一帆风顺。

感谢一直支持我的朋友们，他们是汤显才、熊兵兵、刘浩、于国星、张弛、徐江伟、张陶盛、张毛等，虽有大家身处天南地北，但是在我遇到困难和挫折的时候，你们的鼓励和支持让我有了不断前行的动力。

最后，要感谢我的家人，你们给了我最为无私的爱和无条件的支持。感谢你们在二十多年里对我的关怀、照顾和包容，亲情是至高无上的，值得我一辈子珍惜，因为你们，我才有更大的勇气和信心不断前行。希望你们身体健健康康，快乐幸福每一天。

# 参考文献

1. 1. Abramo G, D’Angelo C A, Di Costa F. Research collaboration and productivity: is there correlation?. Higher Education, 2009, 57(2): 155-171

   [↑](#endnote-ref-1)
2. 1. Lee S, Bozeman B. The impact of research collaboration on scientific productivity. Social studies of science, 2005, 35(5): 673-702

   [↑](#endnote-ref-2)
3. 1. Franceschet M, Costantini A. The effect of scholar collaboration on impact and quality of academic papers. Journal of informetrics, 2010, 4(4): 540-553

   [↑](#endnote-ref-3)
4. 1. Smith C L. Global scientific collaboration and global problems. The Academic Executive Brief, 2011: 2-5

   [↑](#endnote-ref-4)
5. 1. Resnick P, Iacovou N, Suchak M, et al. GroupLens: an open architecture for collaborative filtering of netnews.Proceedings of the 1994 ACM conference on Computer supported cooperative work. ACM, 1994: 175-186

   [↑](#endnote-ref-5)
6. 1. Ricci F, Rokach L, Shapira B. Introduction to recommender systems handbook. springer US, 2011:20-30

   [↑](#endnote-ref-6)
7. 1. Balabanović M, Shohom Y. Content-based, callaborative recommendation. Communications of the ACM, 1997, 40(3)

   [↑](#endnote-ref-7)
8. 1. Pazzani M J. A framework for collaborative, content-based and demographic filtering. Artificial intelligence review, 1999, 13(5-6): 393-408

   [↑](#endnote-ref-8)
9. 1. Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms.Proceedings of the 10th international conference on World Wide Web. ACM, 2001: 285-295

   [↑](#endnote-ref-9)
10. 1. Bobadilla J, Ortega F, Hernando A, et al. Recommender systems survey. Knowledge-based systems, 2013, 46: 109-132

    [↑](#endnote-ref-10)
11. 1. Kautz H, Selman B, Shah M. Referral Web: combining social networks and collaborative filtering. Communications of the ACM, 1997, 40(3): 63-65

    [↑](#endnote-ref-11)
12. 1. Newman M E J. The structure of scientific collaboration networks. Proceedings of the National Academy of Sciences, 2001, 98(2): 404-409

    [↑](#endnote-ref-12)
13. 1. 项亮. 推荐系统实践. 北 京: 人 民 邮 电 出 版 社, 2012:144-165

    [↑](#endnote-ref-13)
14. 1. 汪小帆, 李翔, Chen G. 网络科学导论. 高等教育出版社, 2012:178-185

    [↑](#endnote-ref-14)
15. 1. 方锦清, 汪小帆, 郑志刚, 等. 一门崭新的交叉科学: 网络科学 (上). 物理学进展, 2007, 27(3): 239-343

    [↑](#endnote-ref-15)
16. 1. 方锦清, 汪小帆, 郑志刚, 等. 一门崭新的交叉科学: 网络科学 (下). 物理学进展, 2008, 27(4): 361-448

    [↑](#endnote-ref-16)
17. 1. Newman M E J. The structure and function of complex networks. SIAM review, 2003, 45(2): 167-256

    [↑](#endnote-ref-17)
18. 1. Leicht E A, Holme P, Newman M E J. Vertex similarity in networks. Physical Review E, 2006, 73(2): 026120

    [↑](#endnote-ref-18)
19. 1. Newman M E J, Watts D J. Renormalization group analysis of the small-world network model. Physics Letters A, 1999, 263(4): 341-346

    [↑](#endnote-ref-19)
20. 1. Salton G. Automatic text processing: The transformation, analysis, and retrieval of Information by Computer . Reading: Addison-Wesley, 1989

    [↑](#endnote-ref-20)
21. 1. Hamers L, Hemeryck Y, Herweyers G, et al. Similarity measures in scientometric research: the Jaccard index versus Salton's cosine formula. Information Processing & Management, 1989, 25(3): 315-318

    [↑](#endnote-ref-21)
22. 1. Salton G, McGill M J. Introduction to modern information retrieval. 1986:234-245

    [↑](#endnote-ref-22)
23. 1. Jeh G, Widom J. SimRank: a measure of structural-context similarity.Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2002: 538-543

    [↑](#endnote-ref-23)
24. 1. Lü L, Jin C H, Zhou T. Similarity index based on local paths for link prediction of complex networks. Physical Review E, 2009, 80(4): 046122

    [↑](#endnote-ref-24)
25. 1. Chen H H, Gou L, Zhang X L, et al. Discovering missing links in networks using vertex similarity measures.Proceedings of the 27th Annual ACM Symposium on Applied Computing. ACM, 2012: 138-143

    [↑](#endnote-ref-25)
26. 1. Gollapalli S D, Mitra P, Giles C L. Similar researcher search in academic environments.Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries. ACM, 2012: 167-170

    [↑](#endnote-ref-26)
27. 1. Wang S, Avagyan M, Skardal P S. Evolving network structure of academic institutions. arXiv preprint arXiv:1612.01901, 2016

    [↑](#endnote-ref-27)
28. 1. 何贤芒, 陈银冬, 李东, 等. 基于项目合作的社会关系网络构建. 计算机研究与发展, 2016, 53(4): 776-784

    [↑](#endnote-ref-28)
29. 1. 冯郁. 科研合作网络的描述与分析. 南京信息工程大学学报: 自然科学版, 2009, 1(4): 360-364

    [↑](#endnote-ref-29)
30. 1. Chen H H, Gou L, Zhang X, et al. Collabseer: a search engine for collaboration discovery.Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries. ACM, 2011: 231-240

    [↑](#endnote-ref-30)
31. 1. Kong X, Jiang H, Yang Z, et al. Exploiting Publication Contents and Collaboration Networks for Collaborator Recommendation. PloS one, 2016, 11(2): e0148492

    [↑](#endnote-ref-31)
32. 1. Lopes G R, Moro M M, Wives L K, et al. Collaboration recommendation on academic social networks.International Conference on Conceptual Modeling. Springer Berlin Heidelberg, 2010: 190-199

    [↑](#endnote-ref-32)
33. 1. Cohen S, Ebel L. Recommending collaborators using keywords.Proceedings of the 22nd International Conference on World Wide Web. ACM, 2013: 959-962

    [↑](#endnote-ref-33)
34. 1. Botta F, Del Genio C I. Finding network communities using modularity density. Journal of Statistical Mechanics: Theory and Experiment, 2016, 2016(12): 123402

    [↑](#endnote-ref-34)
35. 1. West D B. Introduction to graph theory. Upper Saddle River: Prentice hall, 2001:43-71

    [↑](#endnote-ref-35)
36. 1. Cormen T H. Introduction to algorithms. MIT press, 2009:594-611

    [↑](#endnote-ref-36)
37. 1. Brown D, Davis C M, Stanlick S. Struts 2 in action. Dreamtech Press, 2008:3-20

    [↑](#endnote-ref-37)
38. 1. Walls C, Breidenbach R. Spring In Action, Updated For Spring 2.0. Dreamtech Press, 2007:97-120

    [↑](#endnote-ref-38)
39. 1. 王福强. Spring 揭秘. 人民邮电出版社, 2009:122-142

    [↑](#endnote-ref-39)
40. 1. Miles R. AspectJ cookbook. " O'Reilly Media, Inc.", 2004:13-29

    [↑](#endnote-ref-40)
41. 1. Bauer C, King G. Hibernate in action. 2005:3-14

    [↑](#endnote-ref-41)
42. 1. Josuttis N M. SOA 实践指南. 2008:11-46

    [↑](#endnote-ref-42)
43. 1. 崔弘亮, 肖刚. 网络管理服务设计技术研究. 计算机工程与设计, 2011, 32(9): 3027-3030

    [↑](#endnote-ref-43)
44. 1. Curbera F, Duftler M, Khalaf R, et al. Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. IEEE Internet computing, 2002, 6(2): 86-93

    [↑](#endnote-ref-44)
45. 1. Richardson L, Ruby S. RESTful web services. " O'Reilly Media, Inc.", 2008：10-20

    [↑](#endnote-ref-45)
46. 1. Box D, Ehnebuske D, Kakivaya G, et al. Simple object access protocol (SOAP) 1.1. 2000:

    [↑](#endnote-ref-46)
47. 1. Abernethy R. The programmer’s guide to Apache Thrift. 2014:5-10

    [↑](#endnote-ref-47)
48. 1. 韩冰, 祝永志. 一种基于 Thrift 的跨平台单点登录实现方法. 软件导刊, 2014, 13(2): 48-50

    [↑](#endnote-ref-48)
49. 1. 王媛媛, 丁毅, 孙媛媛, 等. 数据可视化技术的实现方法研究. 现代电子技术, 2007, 30(4): 71-74

    [↑](#endnote-ref-49)
50. 1. Kuan J. Learning Highcharts 4. Packt Publishing Ltd, 2015:7-21

    [↑](#endnote-ref-50)
51. 1. Meeks E. D3. js in Action. Manning, 2015:6-37

    [↑](#endnote-ref-51)
52. 1. Shani G, Gunawardana A. Evaluating recommendation systems.Recommender systems handbook. Springer US, 2011: 257-297

    [↑](#endnote-ref-52)