西 南 交 通 大 学

本科毕业设计（论文）

基于实体链接的中文文本审校

年 级： 2013 级

学 号： 20131975

姓 名： 刘 畅

专 业： 计算机科学与技术

指导教师： 李天瑞老师

二零一七年五月

院系 信息科学与技术学院 专 业 计算机科学与技术

年级 2013级 姓 名 刘畅

题目 基于实体链接的中文文本审校

指导教师

评 语

指导教师 (签章)

评 阅 人

评 语

评 阅 人 (签章)

成 绩

答辩委员会主任 (签章)

年 月 日

**毕业设计（论文）任务书**

班 级 计算机2013-01班 学生姓名 刘畅 学 号 20131975

发题日期：2016年12月9日 完成日期：2017年5月27日

题 目 基于实体链接的中文文本审校

1、本论文的目的、意义

词语审校是中文文本审校的一个重要组成部分。在中文文本中经常会出现由于音近、形近、义近等因素导致的人名、地名、成语、机构名错误。但目前还没有开放的中文文本校对系统。本论文旨在构建一个针对文本中出现的词语错误进行校对的开放系统。本论文以分词和知识库为基础，用自然语言处理的方法对文本进行审校，主要包含命名实体识别，命名实体链接，命名实体消歧等算法。

命名实体链接是自然语言处理中的一项基础研究，在自然语言处理相关的研究和应用中有着重要作用，是不可缺少的一环。实体链接的主要目标是识别上下文中的名称指代那个现实世界中的实体。具体而言，实体链接是将给定文本中的一个名称字符串映射到知识库中的相应实体上去。本论文是将实体链接的技术因公到中文词语审校的问题中，主要研究中文文本中出现的人名、地名、成语、机构名等错误的审校。

2、学生应完成的任务

（1）查阅相关资料

（2）学习命名实体链接的相关技术

（3）学习词语审校相关内容

（4）算法设计

（5）系统代码实现

（6）完成论文撰写

3、本论文与本专业的培养目标达成度如何？（如在知识结构、能力结构、素质结构等方面有哪些有效的训练。）

4、论文各部分内容及时间分配：（共17周）

第一部分 查阅相关资料 ( 2周)

第二部分 学习命名实体链接相关技术 ( 1周)

第三部分 学习词语审校相关内容 ( 1周)

第四部分 算法实现 ( 3周)

第五部分 系统代码实现 ( 5周)

第五部分 毕业设计论文撰写 (4周)

评阅及答辩 (1周)

备 注

指导教师： 2016年12月9日

审 批 人： 2016年12月9日

摘 要

随着大数据时代的到来，大量的数据资源涌入到我们的生活当中，我们接触到越来越多的出版物。以出版工作为例，我国是一个出版大国，在出版工作中，审校工作占了很大一部分比重。工作人员需要确定在出版物中是否存在错误的信息，而审校交由人工进行则会耗费大量的时间和精力，同时还无法保证正确率。本文考虑这一情况，提出构建一个开放的中文实体词语审校系统。

词语审校是中文文本审校的一个重要组成部分。在中文文本中经常会出现由于音近、形近、义近等因素导致的人名、地名、成语、机构名错误。但目前还没有开放的中文文本校对系统。本论文旨在构建一个针对文本中出现的词语错误进行校对的开放系统。

本文应用实体链接技术来达到审校目的。实体链接技术指通过一定的手段将实体链接到现有数据库中的记录中，可以达到多种目的。本系统的数据存储采用的是非关系型数据库MongoDB，其优点是在存储方式上没有太多限制，可以灵活的存储数据，适合在大数据环境下进行数据存储。

本系统实现流程大致如下：首先基于百度百科建立一个基于百度百科的实体数据库；然后利用分词将审校文本中的实体关系抽取，将需要的人名、地名、机构名抽取出来，系统使用基于HanLP分词器的分词功能，能够添加用户自定义词典，增加了分词的可靠性和精度；其次，尽可能可能到出错的情况以及结果的提出，应用了多种相似度匹配算法，综合考虑设置审校规则；最后，对抽取出来的实体逐一进行审校。

关键词：大数据；实体链接；词语审校；分词；相似度计算

Abstract

With the advent of the Big Date Era, a lot of data resources into our lives, we come into contact with more and more publications. China is a publishing power, in the publishing work, the revision of the work accounted for a large part of the proportion. The staff member needs to determine whether there is a wrong message in the publication, and the review by the manual will take a lot of time and effort, but also can not guarantee the correct rate. This paper considers this situation, proposed to build an open Chinese entity word review system.

Word review is an important part of the Chinese text review. In the Chinese text often appear as the sound near, shape near, meaning near and other factors led to the people’s names, place’s names, idioms, organization’s name error. But there is no open Chinese text proofreading system. This paper aims to build an open system for proofreading errors in words.

This article uses the Entity-Linking technology to achieve the purpose of review. Entity- Linking technology refers to a certain means to link entities to existing records in the database, can achieve a variety of purposes. The system's data storage uses a non-relational database, MongoDB, which has the advantage of being that there are not too many restrictions on storage and can be stored flexibly for data storage in large data environments.

The system implementation process is roughly as follows: First, based on Baidu Encyclopedia to establish a Baidu Encyclopedia based on the entity database; and then use the word segmentation will review the text of the entity relationship extraction, the need for names, names, organization names extracted, the system based on HanLP word segmentation The word segmentation function can add the user custom dictionary, increase the reliability and precision of the word segmentation; Secondly, as far as possible to the error situation and the results of the proposed application of a variety of similarity matching algorithm, consider setting the review rules ; Finally, the extracted entities are examined one by one.

**Keywords:** Big Data; Entity Linking; word review; word segmentation; similarity calculation

目 录

[摘 要 IV](#_Toc482694680)

[Abstract V](#_Toc482694681)

[第1章 绪 论 1](#_Toc482694682)

[1.1 背景与意义 1](#_Toc482694683)

[1.2 国内外发展（应用）现状 2](#_Toc482694684)

[1.3 论文所做工作及思路 3](#_Toc482694685)

[1.4 论文章节安排 4](#_Toc482694686)

[1.5 本章小结 4](#_Toc482694687)

[第2章 相关知识概述 5](#_Toc482694688)

[2.1 MongoDB数据库 5](#_Toc482694689)

[2.1.1 MongoDB数据库简介 5](#_Toc482694690)

[2.1.2 MongoDB数据库特点 5](#_Toc482694691)

[2.1.3 基于百度百科的数据库 6](#_Toc482694692)

[2.2 实体链接 6](#_Toc482694693)

[2.2.1 背景与意义 6](#_Toc482694694)

[2.2.2 实体链接基本概念 7](#_Toc482694695)

[2.3 中文分词 7](#_Toc482694696)

[2.2.1 CRF分词 8](#_Toc482694697)

[2.2.1 N-最短路径分词 8](#_Toc482694698)

[2.4 词语相似度算法 9](#_Toc482694699)

[2.5本章小结 10](#_Toc482694700)

[第3章 设计与分析 11](#_Toc482694701)

[3.1 数据库配置 11](#_Toc482694702)

[3.1.1 MongoDB数据库安装 11](#_Toc482694703)

[3.1.2 数据库设计 12](#_Toc482694704)

[3.2 Java环境配置 13](#_Toc482694705)

[3.2.1 JDK环境配置 13](#_Toc482694706)

[3.2.2 Java连接MongoDB数据库 14](#_Toc482694707)

[3.3 HanLP分词器配置 14](#_Toc482694708)

[3.2.1 Java配置HanLP分词器 14](#_Toc482694709)

[3.2.2 HanLP分词器启用自定义词库 15](#_Toc482694710)

[3.4 相似度算法 16](#_Toc482694711)

[3.4.1 特殊情况考虑 16](#_Toc482694712)

[3.4.1 相似度算法实现 17](#_Toc482694713)

[3.5本章小结 18](#_Toc482694714)

[第4章 实验方法流程 19](#_Toc482694715)

[4.1 运行环境 19](#_Toc482694716)

[4.2 系统流程 19](#_Toc482694717)

[4.2.1 系统框架图 19](#_Toc482694718)

[4.2.2 流程详解 20](#_Toc482694719)

[4.3 本章小结 23](#_Toc482694720)

[第5章 实验及结果分析 25](#_Toc482694721)

[5.1 实验测试数据 25](#_Toc482694722)

[5.1.1 实体链接率 25](#_Toc482694723)

[5.1.2 审校效率 25](#_Toc482694724)

[5.1.3 审校正确率 26](#_Toc482694725)

[5.2 实验数据比较 26](#_Toc482694726)

[5.2.1 分词算法应用比较 26](#_Toc482694727)

[5.2.2 相似度算法应用比较 28](#_Toc482694728)

[5.3 本章小结 29](#_Toc482694729)

[总结和展望 31](#_Toc482694730)

[致 谢 32](#_Toc482694731)

[参考文献 33](#_Toc482694732)

第1章 绪 论

1.1 背景与意义

21世纪的今天，人们生活在一个被信息包围的时代。随着信息社会的迅速发展，数据信息呈现出井喷式的增长。而地球上使用中文的人数超过15亿，超大量的使用人数意味着超大量的数据信息。以出版行业为例，我国是一个出版大国，文本校对是保证出版物质量的重要环节，面对大量涌入的文本信息，如果使用人工来进行校对，不仅需要花费大量的人力物力，同时还无法保证审校结果的正确率与完整性，所以机器校对这一话题就被摆在了台面上。机器审校不同于人工审校，机器审校针对一类文本拥有超高的审校速度，同时一旦审校方法确定了，就可以不断改进，正确率和完整性都有了很大的保障。

机器审校逐渐发展，自动审校这一术语被提了出来。拿期刊论文为例，我国地域辽阔，人口众多，大小期刊数不胜数，每一天每一秒都有新东西出现，针对单一类型项目的机器审校已经很难满足人们的需求。自动审校要求对于任意类型的文本，都能够对其进行审校工作；同时能够跟上时代的发展社会的进步，不断地提升优化自己。自动校对并不只是应用在出版机构中，随着自动校对技术的不断发展，其应用的范围越来越广，大到刊物出版、文件印刷，小到自己写的文章审校、论文校准，都可以用到自动校对。自动校对都有很大的应用前景。

中文文本审校分为多个方面，词语审校是中文文本审校的一个重要组成部分。在中文文本中经常会出现由于音近、形近、义近等因素导致的人名、地名、成语、机构名等实体名称错误的情况。但目前还没有开放的中文文本校对系统。本论文旨在构建一个针对文本中出现的实体词语错误进行校对的开放系统。本论文以分词和知识库为基础，用自然语言处理的方法对文本进行审校，主要包含命名实体识别，命名实体链接，命名实体审校等步骤。

命名实体链接是自然语言处理中的一项基础研究，在自然语言处理相关的研究和应用中有着重要作用，是不可缺少的一环。实体链接的主要目标是识别上下文中的名称指代哪个现实世界中的实体，并将之正确的链接到数据库中的实体指称上。具体而言，实体链接是将给定文本中的一个名称字符串映射到知识库中的相应实体字符串上去。实体链接的工作并不止于此，实体链接技术从提出到现在一直在补充完善，最主要的原因是实体链接的特性：实体链接的链接率并不是百分之百，对于无法连接的实体，多数情况是实体名称出错，但也有较少的情况是数据库数据量不足导致链接失败，而后者的情况更为重要，实体链接的下一步任务之一就是根据连接的情况来补充数据库，根据实体链接的情况来完善数据库也是当下比较重要的一个课题。本论文是将实体链接的技术应用到中文词语审校的问题中，主要研究中文文本中出现的人名、地名、成语、机构名等错误的审校。

本项目目的在于对于给定文本，对于其中的实体进行抽取识别并连接到知识库中，通过与链接到的记录进行匹配判别文中实体是否有拼写错误，如有，则根据数据库中的实体名称提出修改意见。

1.2 国内外发展（应用）现状

随着Web技术的飞速发展和不断普及，人们以网络为媒介，接触到了更多、信息量更加庞大的各色数据资料，中国作为人口大国，中文作为一门博大精深的语言体系，中文文本审校的意义和作用也越来越大，越来越多的国内外学者投入到这一方面的研究。其中有很多优秀的研究成果，如张鑫[1]根据多种模型对面向社会媒体的中文文本校对方法进行了研究和实现；刘亮亮，曹存根[2]二人针对中文“非多字词错误”自动校对方法也进行了研究；李岩[3]的基于深度学习的短文本分析和计算方法有着不错的研究成果。

2009年美国国家标准与技术研究院（National Institute of Standards and Technology，NIST）在文本分析会议（Text Analysis Conference, TAC）的知识库扩充（knowledge base population，KBP）任务中提出了实体链接（Entity Linking）这一子任务，是将文本中的实体指称表述项正确链接到知识库中实体的过程。中文实体链接的研究在近几年也有所起步，但是面临很多难题，其中之一就是没有一个统一的、规模足够大的中文语料数据库。在舒佳根做过的工作中由曾提到过根据维基百科的中文实体链接语料库的建设，并进行了实践，成果较好。王睿对英文实体链接进行了测试，提出了基于上下文和排序学习的实体链接方法；Moro A，Raganato A，Navigli R等人也采用实体链接对词语进行消歧，并提出了一种同一方法，见解深刻：Demartini G，Difallah D E和他们的同伴一起提出了利用概率推理和众包技术来实行大规模的实体链接计数；舒佳根[4]针对实体链接算法的实现也做的很多工作。

目前绝大多数研究工作在文本表述特征的基础上增加了文本语义信息的获取，文件[8]从文本中抽取多种语义特征，对每个候选实体对依次进行排序过滤；文献[9]在此基础上，对从上下文文本中选取的多种语义特征进行加权融合，一定程度上减少了单特征的偶然性；为了更好的体现文本的语义性，文献[10]基于维基百科的连接结构构造图模型，通过图节点的路径长度反应实体与实体之间的相关性。高艳红，李爱萍等人综合上述多个文献，提出了面向实体链接的多特征图模型实体消歧方法并取得了一定的成果。

中文分词算法在中文文本处理中占有很重要的席位。中文文本并不像英文文本那样具有空格间隔这样的天然分词优势，中华文化博大精深，中文的构建形式也多种多样，单字成词、长词、成语等多种组合情况在中文文本中屡见不鲜。从机器开始处理中文文本开始，中文分词算法就不断地被研究改进。最初的分词算法是基于规则的，再考虑到尽可能多的情况下人为的设置分词规则来进行分词；到后来算法的加入，机器学习和深度学习的应用让分词算法的精度和效率有了飞速提升。

综上所述，实体链接技术的研究已经取得了一些成果，但同时还有很大的研究空间，尤其是在中文文本审校方面有着很大的优势。实体链接技术不仅能够对词语进行审校，后期还可以结合前后文的语义对句子中不合理的地方进行修改；同时随着实体链接技术的发展，其本身是可以设置成不断地进行自我完善和优化的。分词算法的结果直接关系到实体抽取的情况，提高分词的精度也是提高连接度和审校效果的途径之一。本文所要做的就是对给定文本进行命名实体抽取，实现最基本的实体链接，通过优化分词并结合数据库进行中文词语审校。

1.3 论文所做工作及思路

首先需要构造一个项目所需要的中文知识库，用于对实体进行存储识别，在得到要审校的文献时，首先对文献进行分词处理，将文中的实体一一找出并进行提取，然后对每一个实体在知识库中进行匹配，找到相似度最高的知识库实体对象进行链接，然后对文献中的实体和知识库中的实体进行比较，如果发现不同，则用知识库中的实体指称代替文中的实体指称并做出标注。同时设置一个阈值，当知识库中的最优实体指称与文本实体指称的相似度低于该阈值时，默认实体链接失败，没有找到对应实体，返回一个空指针。项目流程如图1-1所示，主要过程如下：



图1-1 项目流程

（1）通过网络现有资源，进行自动标注和人工标注后生成项目所需的知识库。

（2）考虑到重名、多名问题，在对知识库实体进行拼音标注的同时构建一个同义词表方便查找。

（3）拿到所需校对的文献后，对整个文献进行分词提取，筛选出全文中的实体表述对象。

（4）对每一个实体对象在知识库中进行匹配，可能在知识库中找到多个实体与文献中实体相关，经计算得出的是知识库中实体对象与所需校对文献中实体对象的相似度。

（5）相似度越高说明两个实体匹配度越高，选取相似度最好的知识库实体进行链接，返回链接指针。

（6）人工设定一个阈值，当相似度小于该阈值时，认为知识库中没有实体与文献中实体对象一致，链接失败返回空指针。

（7）对文献中所有实体对象链接完成后，比较链接结果和原文献中实体对象是否一致，若一致则继续审校；若不一致，则用知识库中的实体对象代替文献中的实体对象，完成校对。

1.4 论文章节安排

论文共分4章，各章内容如下：

第1章介绍项目的应用背景和意义，包括项目的发展过程、基本流程和工作安排。

第2章介绍本论文所涉及到的一些方法和技术原理，主要包括MongoDB数据库的基本知识，实体链接的基本知识、词语相似度算法以及CRF分词算法和N最短路径分词算法的相关应用。

第3章介绍项目的算法设计和运行流程，主要包括系统环境的配置，以及主要部件之间的配置连接，以及关键算法的实现方法。

第4章介绍项目的实现流程，主要通过界面的布置搭建，链接的方法实现以及审校功能的实现对实验流程进行细致的讲解。

第5章介绍项目的运行结果，主要包括项目的运行结果以及对结果进行初步的评估测量，测试了分词方法对结果的影响以及相似度阈值的不同设置对精确度的影响。

1.5 本章小结

本章首先介绍了本文的研究背景和意义，介绍实体链接技术和中文文本审校的研究现状。然后介绍了本文的主要方法和研究进展。接着对全文的内容进行了总结。最后介绍了本文的章节安排。

第2章 相关知识概述

2.1 MongoDB数据库

2.1.1 MongoDB数据库简介

关系数据库是建立在关系数据库模型基础上的数据库，借助于集合代数等概念和方法来处理数据库中的数据；数据库的元数据都是实体与实体之间联系的集合，特点是将具有相同属性的数据独立的存储在一个表中。关系型数据库遵从ACID原则，分别原子性、一致性、独立性和持久性。关系型数据库的特点是简单易懂，结构性强，具有较强的逻辑性，但缺点是不够灵活，较为死板。

非关系数据库是随着互联网web2.0网站的兴起而逐渐发展起来的，用于处理超大规模的数据集合和多重数据，其没有明确的定义，但相比于关系型数据库需要实现定义表结构，非关系型数据库的一大特点是并不需要预先定义模式，每条记录储不需要固定的模式，无需多余操作就可以横向扩展，因此可以存储较为复杂的数据类型。

MongoDB是一个介于关系数据库和非关系数据库之间的产品，是非关系数据库当中功能最丰富也是最像关系数据库的一种综合性数据库。MongoDB是由C++语言编写的，是一个基于分布式文件存储的开源数据库系统。MongoDB将数据库存储为一个文档，数据结构有键值对组成，文档类似于JSON对象，字段可以包含其他文档，数组及文档数组。

2.1.2 MongoDB数据库特点

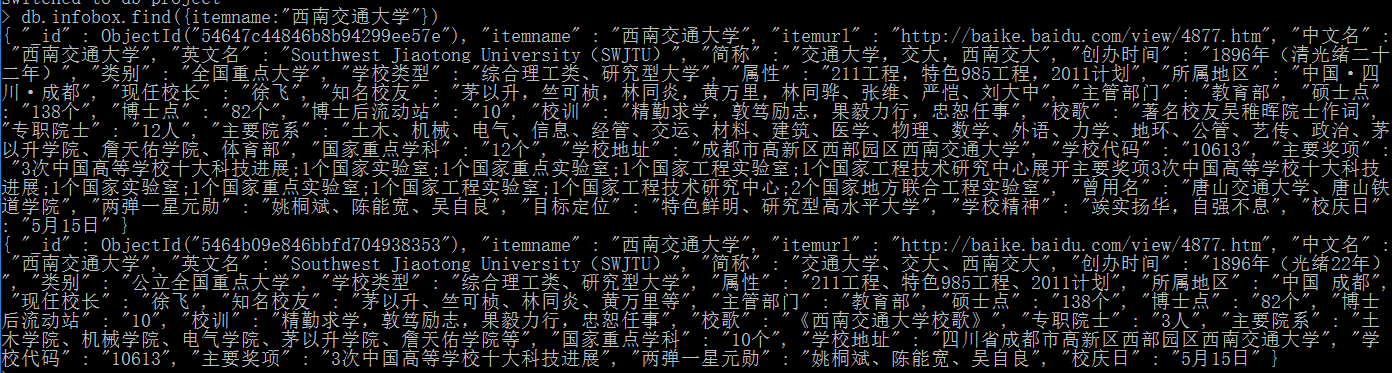


图2-1 MongoDB数据库中数据存储格式

MongoDB数据库属于非关系型数据库中的“键值（Key-Value）存储数据库”，类似于关系数据库中的表，MongoDB数据库的基本结构是集合，集合在数据库中唯一标识，不需要定义任何模式；集合中每一条数据成为一个记录，存储方式为“key：value”的方式，即“键名：数据”的方式，每一对键值可以看成关系数据库表结构中的一个字段（列），该条记录所有的键值对构成完整的一条数据，相当于关系数据库中的一条记录（行）。具体格式如图2-1所示。

MongoDB数据库高性能、易部署、易使用。MongoDB最大的特点是它支持的查询语句非常强大，其余发有点类似于面向对象的查询语言，几乎可以实现类似关系数据库表单查询的绝大部分功能，而且还支持对数据建立索引。MongoDB数据库查询语句简单易懂，同时可以使用正则表达式来进行查询，更为方便的是查询语句可以嵌套，这样的相互组合之间让查询功能变得无比强大。

2.1.3 基于百度百科的数据库

本系统采用基于百度百科的数据。爬取百度百科的数据内容，经过整理存入到MongoDB数据库中，本文爬取了40余万条数据，较为规格化的进行了存储。百度百科对每一条数据都有一种非常规格化的数据样式，以表格的形式放在每个词条的首位，通过爬虫爬取对应的数据内容即可获取实验所需要的数据。

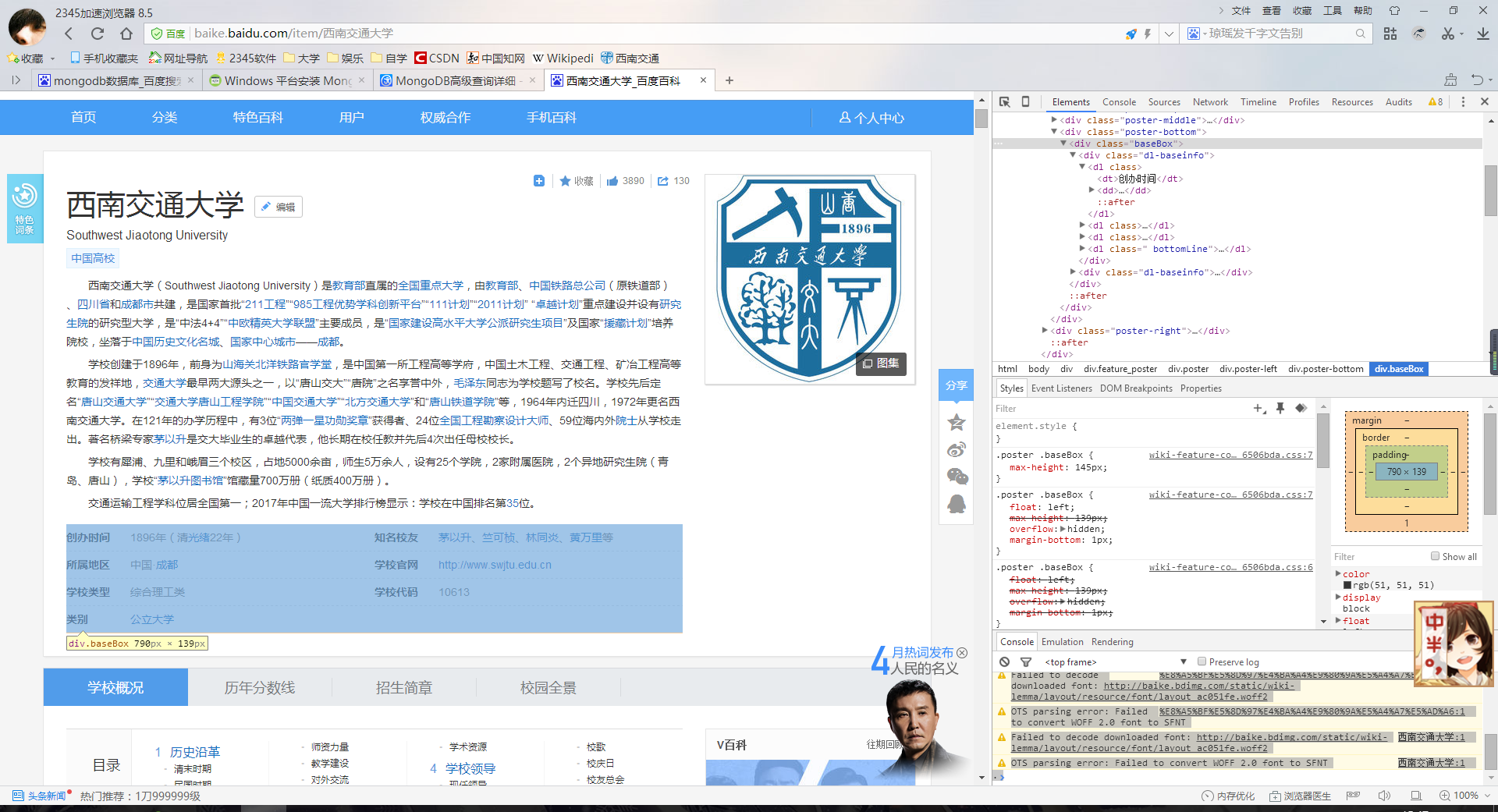


图2-2 百度百科表格数据

2.2 实体链接

2.2.1 背景与意义

2009年美国国家标准与技术研究院NIST（National Institute of Standards and Technology）在文本分析会议TAC（Text Analysis Conference）的知识库扩充KBP（knowledge base population）任务中提出了实体链接（Entity Linking）这一子任务，是将文本中的实体指称表述项正确链接到知识库中实体的过程。

实体链接是命名实体消歧的重要方法，该方法通过将具有歧义的实体指称项链接到给定的知识库中从而实现实体歧义的消除。命名实体歧义是指同一个实体指称项在不同上下文环境中对应不同真实世界实体的语言现象。实体歧义问题给信息处理领域的很多任务带来了严重问题，信息检索和抽取、知识工程等任务都需要功能强大的实体消歧系统做支撑。研究高性能的实体消歧技术具有重要的学术和应用价值。

2.2.2 实体链接基本概念

首先介绍命名实体歧义现象。歧义现象是由于多种原因导致实体指向性不明确，常见的两种情况是多名和重名，如下两种情况：  
 （1）“【孙中山】先生是中国革命事业的先驱。”

（2）“【孙逸仙】先生和【陈仲甫】先生将参加今晚的会议。”

（3）“【迈克尔·乔丹】建立了【公牛】王朝。”

（4）“前阿森纳球员，门将【迈克尔·乔丹】转会切斯特菲尔德。”

在（1）和（2）中，“孙中山”和“孙逸仙”指的都是国民革命时期的著名领袖孙中山先生，但前者是本名，后者喊的是字；而（3）和（4）则不同，（3）中的“迈克尔·乔丹”是指美国NBA联赛中非常出名的篮球运动员乔丹，（4）中“迈克尔·乔丹”指的则是英超联赛中的一名足球运动员，两者并不是同一个人，但是起的是相同的名字。

上述的两种情况都会导致命名实体歧义现象。而实体链接的目的则是将实体指称正确的链接到数据库中的实体上，即将（1）和（2）中的“孙中山”和“孙逸仙”同时链接到孙中山实体指称上，（3）中的“迈克尔·乔丹”链接到篮球运动员乔丹上，（4）中的“迈克尔·乔丹”链接到足球运动员乔丹上。

2.3 中文分词

中文分词在中文文本处理中是必不可少的一步。英文文本因为英文单词的独立性，所以单词与单词之间有空格分隔，处理是直接按空格切分就可以获取所有的单词。不同于英文文本，中华文化博大精深，中文的用法多样，有时一个字都可以看成一个词，有的时候几十个字也是一个词，没有通用的分隔方法。几乎所有的中文文本处理，所以中文分词的结果可以说在很大程度上决定了实验结果的好坏与否。本实验使用了开源的Java分词器HanLP分词器，其自身集成了很多分词方法。实验使用了CRF分词方法和N最短路径分词方法。

2.2.1 CRF分词

CRF算法全称为条件随机场算法（Conditional Random Field algorithm），是一种机器学习模型，同时是判别式机器学习技术中较为重要的一个模型。条件随机场最早由Lafferty提出，结合了最大熵模型和隐马尔科夫模型的特点，是一种无向图模型。

设是一个无向图，是以*G*中节点*v*为缩印的随机变量构成的集合。在给定*X*的条件下，如果每个随机变量服从马尔科夫属性，即，则就构成了一个随机条件场，其中*u~v*表示*u*和*v*是相邻的边。

CRF代表了一种新一代的机器学习技术分词，其基本思路是对汉字进行标注，即由字构词（组词），不仅考虑了文字词语出现的概率信息，同时考虑上下文语境，具备较好的学习能力，因此其对歧义词和未登录词的识别都具有比较好的效果，但其不足之处也较为明显，需要大量的语料库进行训练，同时训练周期较长，运行时的计算量比较大。

CRF分词原理简单介绍可以理解为字的词位分类问题，具体步骤如下：

（1）首先定义词位信息如下：

词首，通常用B表示

词中，通常用M表示

词尾，通常用E表示

单字词，通常用S表示

（2）CRF分词的过程就是对词位标注后，将B和E之间的字，以及S单字构成分词。

分词实例：我爱北京天安门。

CRF标注后：我/S 爱/S 北/B 京/E 天/B 安/M 门/E。

CRF分词结果：我/ 爱/ 北京/ 天安门。

当下有许多开源的分词器都使用了CRF分词算法，本系统使用了HanLP开源的Java分词器，使用CRF分词能够较好的对未登录词（数据库中没有记录的词）进行识别，但缺点是无法使用自定义的词库。

2.2.1 N-最短路径分词

最短路径匹配算法是根据词典，找出字串中所有可能的词（也称全分词），然后构造词语切分有向无环图。这样，每一个词对应图中的一条有向边。若赋给相应的有向边一个权值（该权值可以是常数，也可以是构成的词的属性值），然后针对该切分图，在起点到终点的所有路径中，求出最短路径，该最短路径上包含的词就是该句子的最优切分结果。即找到*N*条从起点到终点的路线，然后根据权重和规则选择出最优的一条路线作为分词的结果。

在图2-3中，边的起点为词的每一个字，边的终点为词尾的下一个字。单字边表示单字成词，如“我”；跨字边表示跨越的所有字成词，如“只是”。每个边都拥有一个权值，表示该词出现的概率，最简单的做法是采用词频作为权重，也可以采用TF-IDF值（TF，term frequency，词频；IDF，inverse document frequency，逆向文件频率。某一特定词语的IDF可以由总文件数目除以包含该词语文件的数目，然后对商去对数得到。TF-IDF实际上就是TF\*IDF）作为权重提高对低频词的分词准确度。

N最短路径分词即在上述有向无环图中寻找N条权值和最大的路径，路径上的边标志了最有可能的分词结果，通常我们只求权值和最大的那一条路径。图2-3的分词结果即为“我/只是/做/了/一些/微小/的/工作”。

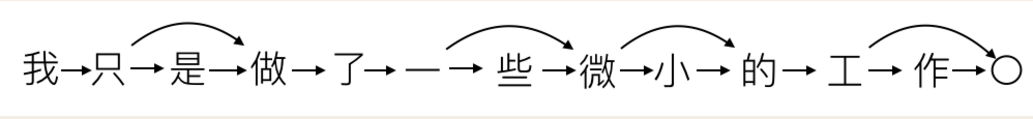


图2-3 “我只是做了一些微小的工作”形成的有向无环图

最短路径匹配算法的规则是使切分处理的词数最少，符合汉语自身的语言规律。但是，同样发现在实际应用中，同样不能正确切分出许多不完全符合规则的句子。如果有多条最短路径，往往只能保留其中一个结果，这样对其他同样符合要求的结果是不公平的，也缺乏理论依据。

2.4 词语相似度算法

命名实体抽取出来后，在数据库中进行链接是，相似度算法必不可少。词语相似度算法，顾名思义就是计算两个词语相似程度。常用的方法有最长公共子串、最长公共子序列、最少编辑距离和基于空间向量的余弦值计算等方法，本文采用的是词语编辑距离相似度算法，即最少编辑距离。

编辑距离，又称Levenshtein距离（也叫做Edit Distance），是指两个字符串之间，有一个转换成另一个所需要的最少操作次数，如果他们的距离越大，则说明他们越是不相同。许可的编辑操作包括将一个字符替换成另一个字符，插入一个字符，删除一个字符。以kitten转化为sitting为例：

替换操作 kitten→sitten（k→s）

插入操作 sitten→sittieng（插入i，g）

删除操作 sittieng→sitting（删除e）

编辑距离就是上述三种操作次数的总和。假设字符串*S*的长度为*m*，字符串*T*的长度为*n*，则编辑距离的计算方法如下所述。首先假设表示从*S*到*T*的编辑距离，那么阶矩阵就可以通过公式（2-1）计算得到：

公式（2-1）包含了删除、插入、替换三种操作。*ins\_cost*表示进行插入操作的编辑常量，在实验中设为1；*sub\_cost*表示进行替换的编辑常量，当比较位上的两个字符，相同时*sub\_cost*设为0，不同时设为1；*del\_cost*表示进行删除操作的编辑常量，实验中设为1。该算法从两个字符串左边开始比较，记录已经比较过的编辑距离，然后进一步得到下一个字符位置时的编辑距离。逐行逐列获取，最终表示**的值，即*S*和*T*的编辑距离。编辑距离越大，相似度越小。对编辑距离归一化处理得到归一化编辑距离（Normalized Edit Distance，*NED*）,字符串*S*，*T*的归一化编辑距离计算公式如下：



其中*m*为*S*的字符串长度，*n*为*T*的字符串长度。当*S*与*T*完全相同是，*NED*=0；当*S*与*T*完全不同时，*NED*=1；即。为了更加符合通常情况下的思维，将归一化的编辑距离转换为次遇见的词语编辑距离相似度，如下所示：



除了词语编辑距离相似度，本文还应用了拼音编辑距离相似度。实验所使用的HanLP分词器自带有词语转化为拼音的接口组件，同时对于转化来的拼音也有着不同的形式。分词器提供的分词形式有很多种，常用的有带声调标注的拼音转换、不带声调标注的拼音转换、单独声母提取的拼音转换和单独韵母提取的拼音转换4种。考虑到上文刚才提到的词语编辑距离相似度，而不带声调的拼音转换的结果较为符合上述算法形式，可以在不使用其他方法的前提下进行拼音相似度的计算，减少了不必要的算法实现和工作量。

2.5本章小结

本章介绍了项目中所用到的内容的相关知识，包括非关系数据库MongoDB的应用，实体链接任务的介绍和意义，中文分词算法的简单介绍和词语相似度的计算方法。MongoDB在处理大量非规格化数据是有较大的优点，但是难于描述。实体链接技术用处多样而有效率；单一的中文分词方法总是具有或多或少的的缺陷，相互结合着来用会使结果更加精确。相似度算法的应用有效的解决了模糊查找困难的问题，但是单一的相似度局限性很高，所以从两方面考虑使用了两个方面的相似度相互配合，相互补充。

第3章 设计与分析

本章介绍了实验环境配置的软件环境配置，包括实验环境的安装配置、数据库的安装和配置以及关键的算法方法的细致讲解。

3.1 数据库配置

3.1.1 MongoDB数据库安装

通过MongoDB官网下载最新版数据库MongoDB-3.4.1.msi文件，该文件是一个Windows的数据包，简单来说就相当于一个可执行文件。执行该文件后进入安装程序。记住数据库安装的路径。

MongoDB数据库是没有自带的用户图形界面的，所有的命令都要通过Windows控制台来完成。通过控制台跳转命令（cd+路径）跳转到安装目录的bin文件夹下，执行mongod.exe文件来配置MongoDB数据库，手动确定数据库目录所在的位置。命令如下：

|  |
| --- |
| MongoDB根目录/bin>mongod.exe --dbpath +db文件夹所在目录 |

文件夹db存放的是MongoDB所保存的数据，这个数据库目录数据库本身不会创建，需要用户创建好了之后通过mongod命令进行配置。运行情况如图3-1所示。

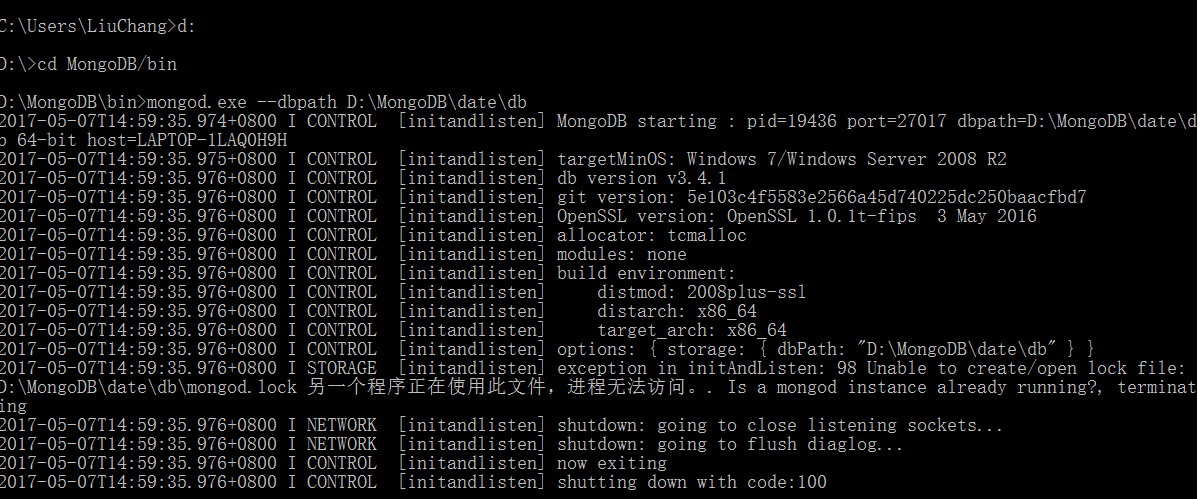


图3-1 MongoDB基本配置完成图

如果只是偶尔的使用MongoDB数据库，每次使用的时候手动启动也可以，但是做工程时每次都要手动启动会相当的麻烦，好在MongoDB提供一种机制，可以将MongoDB服务器设置为Windows服务运行，这样每次开机是MongoDB都会作为一种后台程序自动启动。虽然占资源，但是MongoDB本身并不是很大，所以建议经常使用时有必要配置一下。配置命令如下：

|  |
| --- |
| MongoDB根目录/bin>mongod.exe --dbpath +db文件夹所在目录 –logpath +日志文件夹所在目录 –port+占用的端口 –serviceName “数据库名字” –install |

该命令只列出了必要的几个选项，首先dbpath不用说，是数据库目录文件；logpath则是数据库日志文件，这个文件在数据库备份恢复时非常重要，建议和db文件夹放在一起；port是服务器的端口号，默认为27017，可以更换，但是要保证更换的端口号不能被其他程序所占用；serviceName则是服务运行时显示的名称；最后，install则是启动命令，将MongoDB设置成Windows服务程序。具体命令如图3-2所示：



图3-2 MongoDB配置成Windows服务命令

3.1.2 数据库设计

MongoDB是一种非关系型数据库，所以在存储格式上样式没有固定模式，向本次实验的数据本身有40余万条，但所包含的字段（暂时这么称呼，便于理解，但MongoDB并没有这一定义）就达到了80多万个。虽然存储格式本身没有固定格式，但是仍有相同的地方，这里列举相同之处作为一种参考。

表3-1 数据库集合中相同部分列举

|  |  |  |
| --- | --- | --- |
| 列名 | 含义 | 备注 |
| id | 记录编号 | MongoDB对每一条记录都有一个对应的编号，相当于关系数据库中表的主键，用于对记录进行唯一标识 |
| itemname | 记录名称 |  |
| itemurl | 记录来源 | 记录爬取的网站URL |
| 中文名 | 记录的中文名称 | 大量数据总存在外国资料，所以存储时记录中文表述 |
| Describe | 描述 | 数据库存储了各个方面的数据，无法用统一的格式来描述，但每条数据都有各自的内容记录形势，如自然界的界门纲目科属种、对人的描述包含生平事迹等 |

爬取百度百科相对简单，因为百度百科对每一条记录都有一个对应的表格来描述该项纪录，只需要将表格爬取下来，将表头作为键，将表的内容作为值存放进数据库中即可。

3.2 Java环境配置

3.2.1 JDK环境配置

本实验说用到的语言为Java，配置环境变量自然是基础中的基础。下载并执行最新版的JDK-8u66-windows-x64.exe安装文件，可以改变安装路径。配置JDK最关键的一步就是设置系统的环境变量，配置JDK和JRE的路径。

JDK是Java的开发工具，JRE是Java的运行环境。配置系统变量是为了在执行程序是可以找到这两个关键文件，一旦这一步出错，所编写的程序根本无法正确运行。配置系统变量的命令如下：

（1）新建系统变量：JAVA\_HOME

变量值：JDK文件的安装路径

（2）新建系统变量：CLASSPATH

变量值：.;%JAVA\_HOME%\lib\dt.jar;%JAVA\_HOME%\lib\tools.jar

（3）修改已存在的环境变量：Path

增加变量值：%JAVA\_HOME%\bin;%JAVA\_HOME%\jre\bin

配置完成后进入dos命令窗口，依次输入“javac”、“java”、“java -version”，观察输出情况。三条命令：javac调用编辑器将文件转化成.class文件；java可以运行.class文件，java -version用于查看Java版本。正确显示这表明JDK配置成功。

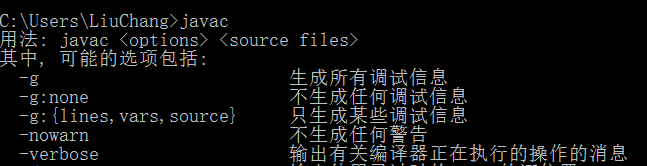


图3-3 Dos运行javac指令结果图

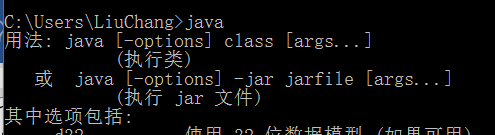


图3-4 Dos运行java命令结果图

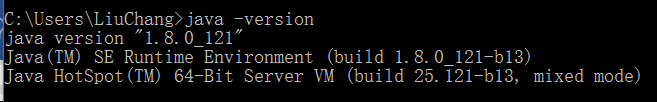


图3-5 Dos运行java -version命令结果图

3.2.2 Java连接MongoDB数据库

Java语言是当下较为流行的语言之一，大多数数据库都针对Java做了连接数据包，MongoDB也不例外。将mongo-java-driver-3.4.2.jar包放在工程目录下面，通过Eclipse配置驱动路径导入jar包。jar包提供了连接所必要的接口以及各种查询语句方法的调用接口，直接创建接口对象就可以链接到数据库中。

3.3 HanLP分词器配置

3.2.1 Java配置HanLP分词器

HanLP是一个开源的Java中文分词器，其优点是集成了大量的已经训练好的分词方法，体积小且便于配置使用，缺点是只能单一的用Java语言进行调用，使用面不够广阔。配置有两种方法，一种是使用Maven工程时的配置，直接修改xml配置文件即可，本项目没有使用，不多做介绍；另一种是普通Java工程的配置方法。

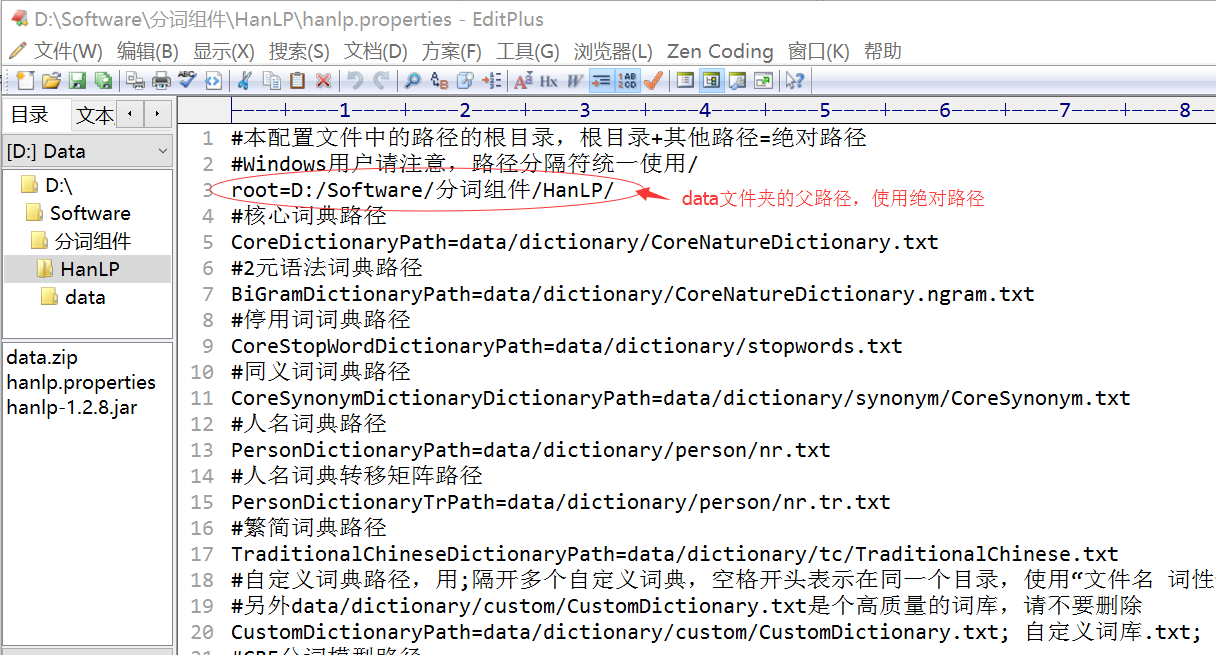


图3-6 hanlp.properites文件中修改data文件夹父路径

HanLP官网对两种方法进行了区分。普通工程方法所需要的文件分为3类：Java连接HanLP所需要的jar包、分词组件文件夹data以及配置文件hanlp.properties。使用HanLP时，首先确定data文件夹的所在路径，打开hanlp.properties，将data的父路径配置在文件的首行，如图3-6所示。然后将配置文件放到工程的bin文件夹下，最后配置工程，放入jar包，直接导入头文件后就可以直接调用HanLP提供的分词器。

3.2.2 HanLP分词器启用自定义词库

时代是在进步发展，软件也在不断地更新迭代。单机版的HanLP分词器也不会是一成不变的。有时分词的效果可能达不到用户的理想情况，原因有很多，通常原因为用户使用了HanLP所不能分辨的新词或者HanLP遇到了没有记录的词语。考虑到这些情况，HanLP分词器提供了一种机制：允许用户添加自定义词库。

用户添加自定义词库大大方便了用户，提高了不同用户对分词情况的不同需求。使用自定义词库有两种不同形式：第一种方法用户通过代码强制在词库中添加、删除、修改词语，这种方法针对添加较少词语的情况，但不推荐使用，因为这样会在基本目录中进行修改，大多数用户会在网上进行分享交流，当基本目录进行修改后就会对软件系统的维护造成困难，所以更建议使用第二种方法。

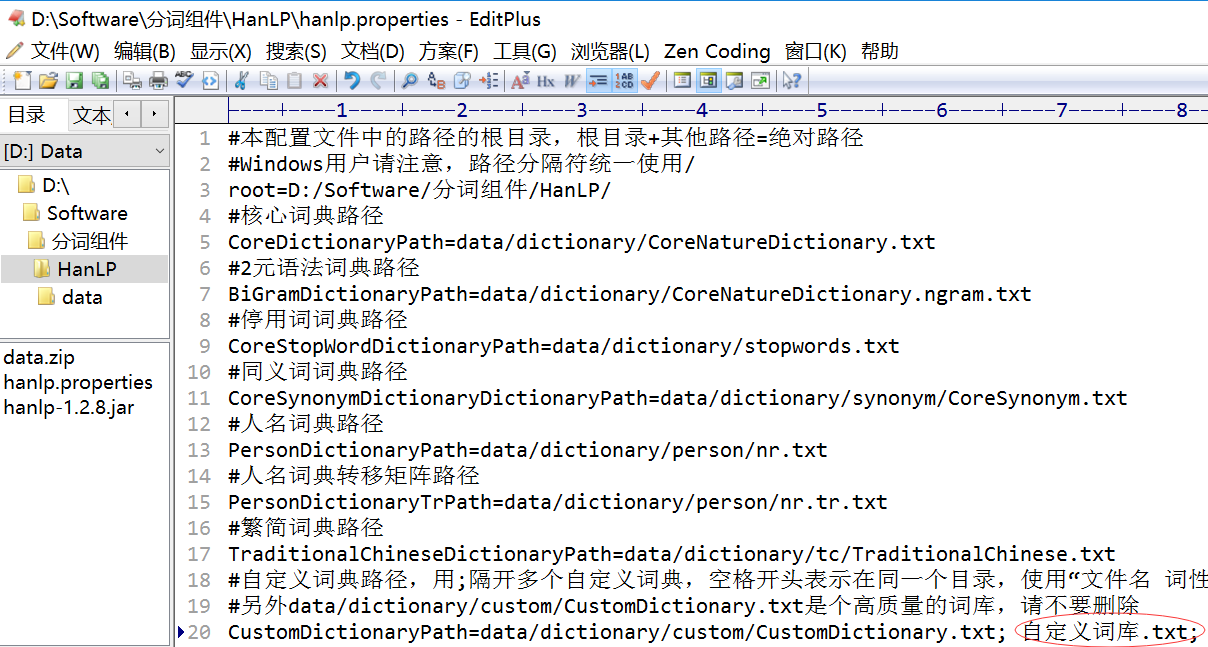


图3-7 修改配置文件中的自定义词库路径

第二种方法则是建立一个自定义词库文件（文本文档即可），然后修改配置文件中的的自定义词库路径（如图3-7），将自定义词库的路径添加进去，自定义词库如果是定义在data/dictionary/custom目录下，则直接在已有文件分号后加空格和文件名，标明该文件去前面文件的路径相同，否则要将绝对路径加入配置文件。然后删除data/dictionary/custom路径下的CustomDictionary.txt.bin文件。这个文件是一个二进制缓存文件，是自定义词库训练后的结果，每次修改自定义词库后都要对该文件进行删除，否则新增修改无法生效。然后运行HanLP分词器会重新生成这个文件。首次使用自定义词库比较慢，因为需要生成CustomDictionary.txt.bin，以后在使用的时候就不需要经过该文件的生成步骤。

自定义词库中的词语有其固定的格式，分为词频词性词典和词频词典两种类型。词频词性词典每一行代表一个单词，格式遵从“[单词] [词性A] [A的词频] [词性B] [B的词频]…”；词频词典每一行代表一个单词，格式遵从“[单词] [单词的频次]”。HanLP自带的词典均为词频词性词典。用户自定义词典也沿用同样的格式，但词性词频可以不进行标明。在修改配置文件是可以再文件名后加入词性标志标明该词库内的单词均是这一词性；如果词性没有标明，则会默认为名词（n）。HanLP分词器是根据人民日报的语料库来进行训练的，词频也是由此而来，如果用户想要自己定义的单词优先级较高，可以将词频设置较大。见图3-8。

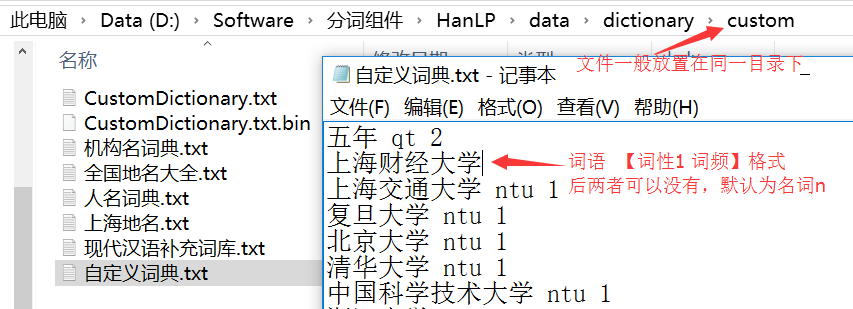


图3-8 自定义词库

3.4 相似度算法

前面已经介绍过相似度算法的计算方法，这里着重讲述一下算法具体的实现过程。考虑到最终结果为词语审校，在相似度计算时考虑的方面比较多，同时加入多种考虑情况，让特殊情况下的计算变得简单。

3.4.1 特殊情况考虑

拼音相似度计算类似于词语相似度计算，故两者统一考虑，后如无特殊说明相似度计算均代表两种相似度。特殊情况下相似度一目了然，如果再用算法计算就会非常浪费时间和资源，所以对特殊情况单独处理。因为相似度归一化公式直接关联与编辑距离，所以考虑的均是编辑距离的情况。

1. 比较的两个词语有一个为空，则直接令编辑距离等于存在的词语。
2. 两个词语存在包含关系，即一者在另外一者中完全存在，则令编辑距离等于两者长度差值的绝对值。
3. 比较的两者长度差异较大，明显不会有很高的相似度。设定比较的两者如果长度较长的一方的长度大于较短一方长度的2倍，则认为不存在相似度，编辑距离等于长度较长词语的词长，即归一化相似度等于0。

当然还存在其他的特殊情况，但是大部分情况下进行特殊处理的花销和运行算法的花销相差不大，就不进行过多的考虑了。

3.4.1 相似度算法实现

词语编辑距离的计算就是对于长度分别为*m*，*n*的两个字符串，构造一个阶矩阵*d*，然后不断对矩阵进行填充，当矩阵填充完整时，的值就是两个字符串的编辑距离。得到编辑距离后根据归一化编辑距离相似度的计算公式即可计算相似度。以“ivan1”和“ivan2”两个字符串计算为例说明算法流程。

首先排除特殊情况，在可以用一般情况进行处理时，第一步进行矩阵的初始化。两个字符串的长度均为5，所以构造一个6\*6阶矩阵，矩阵的（0,0）位置为0，首行和首列为从0开始的依次递增，如表3-2所示，字符串显示的行列不计算在矩阵中，只是为了方便显示。下面进行计算。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | i | v | a | n | 1 |
|  | 0+0=0 | 1+1=2 | 2 | 3 | 4 | 5 |
| i | 1+1=2 | 0 |  |  |  |  |
| v | 2 |  |  |  |  |  |
| a | 3 |  |  |  |  |  |
| n | 4 |  |  |  |  |  |
| 2 | 5 |  |  |  |  |  |

表3-2 d矩阵初始化 表3-3 d矩阵计算

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | i | v | a | n | 1 |
|  | 0 | 1 | 2 | 3 | 4 | 5 |
| i | 1 |  |  |  |  |  |
| v | 2 |  |  |  |  |  |
| a | 3 |  |  |  |  |  |
| n | 4 |  |  |  |  |  |
| 2 | 5 |  |  |  |  |  |

矩阵计算填充时计算的是当前位正上方、左上方、正左方三个位置的值经过公式2-1计算后的值。如果比较的两位字符相同，则*sub\_cost*替换代价为0，否则为1。然后比较（正左方计算结果值，插入操作），（正上方计算结果值，删除操作）和（左上方计算结果值，替换操作）三者的大小，选取最小值填到当前位，如表3-3所示，当第一位进行比较时，三者中最小值为0，所以当前位为0，意味着第一位进行比较是不需要编辑代价，即编辑代价=0。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | i | v | a | n | 1 |
|  | 0 | 1 | 2 | 3 | 4 | 5 |
| i | 1 | 0 | 1 | 2 | 3 | 4 |
| v | 2 | 1 | 0 | 1 | 2 | 3 |
| a | 3 | 2 | 1 | 0 | 1 | 2 |
| n | 4 | 3 | 2 | 1 | 0 | 1 |
| 2 | 5 | 4 | 3 | 2 | 1 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | i | v | a | n | 1 |
|  | 0 | 1 | 2 | 3 | 4 | 5 |
| i | 1 | 0 | 1 |  |  |  |
| v | 2 | 1 | 0 |  |  |  |
| a | 3 | 2 | 1 |  |  |  |
| n | 4 | 3 | 2 |  |  |  |
| 2 | 5 | 4 | 3 |  |  |  |

表3-4 d矩阵计算 表3-5 d矩阵计算结果

通过表3-4和表3-5的计算，可以得出两个字符串的编辑距离为1.根据归一化编辑距离相似度的计算公式，两个字符串的相似度为。

通过上述步骤，可以得到词语方面的两个词语的相似度。上述的例子同样可以用在拼音相似度的计算上，或者说上述的样例更加近似于拼音相似度的计算。两种相似度计算方法的应用是实验中较为重要的一步之一，算法的正确性也是重中之重。

3.5本章小结

本章介绍了项目所需要的基本环境的配置情况，MongoDB数据库适用于大量数据的存储，同时能够给无统一格式的数据以一定的统一形式来显示；HanLP分词器继承了多种分词方法，可以结合使用是分词情况更为优化。详细介绍了相似度算法的实现方法以及多方面情况的考虑，尽可能多的考虑到各种情况。下一章将介绍项目的具体实现流程。

第4章 实验方法流程

本章则是对实验运行的环境配置进行了说明，对实验的流程进行了详细的讲解，并对其中需要特殊说明的地方进行则特殊的说明。

4.1 运行环境

表4-1 实验运行环境

|  |  |
| --- | --- |
| 操作系统 | Windows10 家庭版 |
| 软件环境 | JDK 1.8.0\_121 |
| 数据库 | MongoDB 3.4.2 |

4.2 系统流程

4.2.1 系统框架图



图4-1 项目流程图

图4-1给出了整个实验的基本流程。实验针对给出的文本数据，在数据库中进行链接。实验中较为重要的两步为分词和基于规则进行链接，这两步的结果直接影响了实验结果的好坏。图4-2给出了分词的具体流程，介绍了CRF分词和N最短路径分词结合使用的方法；图4-3则对链接过程进行了细致划分，灵活运用相似度算法结合数据库知识完成实体链接过程。

图4-2 分词流程详解 图4-3 实体链接流程

4.2.2 流程详解

**1. GUI界面**

用户图形界面（Graphical User Interface，简称GUI）又称为图形用户接口，是指采用图形方式显示的计算机用户界面。本实验有一个Java语言编写的简易图形化界面，不仅方便与使用者操作，同时能够清晰明了的显示实验的结果数据。

本实验通过java.swing组件创建了一个用户界面。界面整体分为两个部分，左侧的文本框用于显示需要进行审校的文本，可以用户手动键入，也支持通过文件菜单栏的打开按钮浏览电脑中存在的文件，暂时只支持打开txt文本文件。文件编码应为中文编码utf-8，否则读取时会显示乱码。审校按钮会调用实验的审校功能。GUI界面比较简单，仅仅方便文本的读取和结果的显示，方便用户和程序之间的交互。审校完成后用户可以再左侧对原始文本进行修改，界面提供了文本保存功能，在文件菜单栏里。界面如图4-1所示。



图4-1 实验的GUI界面

**2. 分词部分**

HanLP集成了多种分词方法，综合考虑速度、精度和开销的多个方面，决定使用CRF分词和N最短路径分词相结合的方法。

CRF分词和N最短路径分词前面已经介绍过原理，如果单独使用N最短路径分词，则分词的效率不够高，且分词的精度达不到要求。对于同一段文本进行测试，N最短路径分词的分词效果很难令人满意，很多可以划分的词都没有进行分割，而很多不是词的词则被认为是一个实体名词，这样的情况下不仅增加了运算的开销，同时精度达不到要求，最终的结果也让人难以满意。

CRF分词实体识别度比较高，同时对未登录词也有较好的识别率，但是在经过测试后发现，对于一些错误的实体名词，CRF分词后的结果并不是理想中的情况（词性标注为“/nr”“/ns”“/nt”），而只将词性标注为“/nz”，其意义为其他专属名词。面临的问题就是一部分实体名称标注为其他专属名词，需要额外处理；但不是所有的其他专属名词都需要处理，这样就对错误实体的处理增加了难度。所以考虑增加一种分词方法，在CRF分词后的结果基础上进行辅助分词。

对于CRF分词结果，对于其中词性为“/nz”的字词进行重新分词，并判断新分词后的记过，如果结果为实体名称（词性为“/nr”“/ns”“/nt”），则对其进行审校，否则跳过。这样在精度提高的同时让开销达到了最小值。

**3． 基于规则的实体链接**

实体链接技术是项目的核心，项目的最终意义在于词语审校，所以考虑通过规则来实现实体链接。基于规则的考虑是因为便于实验，实体链接技术说到底是用于服务词语审校的，基于规则的实体链接是完整实体链接技术的简化版，词语审校的主要功能还是放在审校上面，基于规则的实体链接技术可完全以达到目标。

基于规则的实体链接虽然因为规则而具有局限性，但是随着后期的不断完善，精度也随着不断提高。下述是本实验所列的规则，对于需要审校的实体：

**步骤1：**首先搜索数据库中是否有该实体的对应实体，如果存在（不论个数），则认为实体指称正确，退出审校。否则进入步骤2。

**步骤2：**搜索是否有包含审校实体的记录，如果存在，抽取出来进行存储，通过相似度匹配算法找出相似度最高的一个进行审校匹配，给出审校意见，退出审校。否则进入步骤3。

**步骤3：**考虑到大多数情况下词语的键入错误是由于拼音出错导致的，所以对需要审校的实体指称在数据库中进行拼音相似度匹配。设定拼音相似度阈值，对高于阈值的数据库实体进行记录，选取相似度最高的实体进行审校匹配，给出审校意见，退出审校。否则进入步骤4。

**步骤4：**默认实体指称的首字（人名的姓氏、机构名的开头）是不会出错的，对数据库进行搜索，查找包含首字的记录，通过相似度匹配算法选出最优的匹配实体，进行审校匹配，给出审校意见，退出审校。否则认为链接失败，链接为空。

考虑的多种情况，列出了上述的规则，对于正规的实体名称链接行较好，但对于别名、简称等非正规实体名称效果较差，这与规则无关，造成对非正规实体名称效果太差的原因是因为数据库信息不够，无法达到理想效果。基于上述规则，没有考虑到语义情况，所以审校结果的提出会有一些不相匹配的地方。

**4． 审校结果的提出**

基于规则的实体链接方法的应用只是单独的从词语本身来考虑，并没有考虑到语义和语境的情况，这样就意味着任何一个实体指称放在对应位置理论上都是成立的，这样就对审校结果的提出造成了困难。

考虑到上述情况，审校结果的提出是建议性的，并不会在原文上进行修改，这就是为什么GUI界面上会有两个编辑框。审校结果会指出哪个实体指称出错，并给出一个可能的结果，之后考虑的没有语义的最佳结果可能并不是最佳的，所以将规则中匹配到的实体根据相似度从高到底列出最高的3到5个，以供用户参考。这样即使审校出错用户也不用因为系统的错误而做无用功。

4.3 本章小结

本章介绍了实验的开展流程。重点介绍了本实验的实现过程和实验方法。使用简化的实体链接结束让效率达到最优，同时最大程度上的考虑各方面的情况，在不影响精度的情况下尽可能的提高实验的运行效率。在下一章将重点介绍实验的运行结果和最后的结果评测。

第5章 实验及结果分析

本节通过测试数据对实验代码进行测试，测试数据分为大型数据和小型数据两种类型，大型数据的数据量较大，字数一般在几十万字到一百万字；小型数据的数据量则远远小于大型数据，一般在几千到几万字。大型数据用于测试实验代码的性能，而小型数据则是用于进行功能测试。本章节对比了不同分词算法的应用和相似度计算阈值对实验结果的影响，并做了详细的分析。

5.1 实验测试数据

本实验的测试数据是从网上下载的，包含一些其他学者进行实体关系抽取以及命名实体识别是用到的实验数据；除了这些数据，下载了人民日报的内容数据作为实验测试数据之一。

实验的结果会显示文本中的实体数目、链接到数据库中的数目、链接失败的数目以及实体名称可能出错的数目。这些数据统计会帮助我们分析原文本的情况以及对实验结果的评价。

本实验的链接过程没有考虑语义情况，所以对于链接结果显示的修改意见并不能说原文中的实体一定出错，所以数据集进行测试时，考虑的数据包含链接成功率、审校的正确率等可信度较高的数据。给出各数据的定义和计算方法。

5.1.1 实体链接率

实体链接率（Entity Linking rate）是指链接到数据库中的实体占所识别实体的比例，用来表征实体链接的效率：链接率越大，链接的效率越高，审校的可信度也就越强。用ELR表示实体链接率，则

ELR=链接到的实体数目/数据实体总数目

但是实体链接率并不能保证链接的成功率，在没有语义考虑的情况下就没有链接正确率的概念定义。

5.1.2 审校效率

实验的最终目的是对测试数据中的中文实体名称进行识别审校，审校的效率自然也是判定实验结果好坏的标准之一。审校效率（Revised Efficiency）定义为审校成功的实体占整体实体的比例，用RE表示，则

RE=审校成功的实体数目/整体实体总数目

审校效率直接体现了实验本身性能的高低，但单一的审校效率无法全面正确的评价实验结果，所以还需要别的数据来进行辅助评价。

5.1.3 审校正确率

审校正确率定义为审校正确的实体数目在审校成功实体数目中的比例。用Cor表示审校正确率，则

Cor=审校正确的实体数目/审校成功的实体数目

审校正确率较为直观的展现了实验结果的精确度。审校工作要保证的不仅仅是能够对数据中的实体名称进行审校，同时要达到一定的实验精度。如果实验的效率较高但是正确率很低，那么意味着审校的精度不够，实验的结果仍然是不完美的。

5.2 实验数据比较

5.2.1 分词算法应用比较

实验中应用了CRF分词法和N最短路径分词法，针对同一数据集，测试单独使用两者时和两者结合使用的情况。

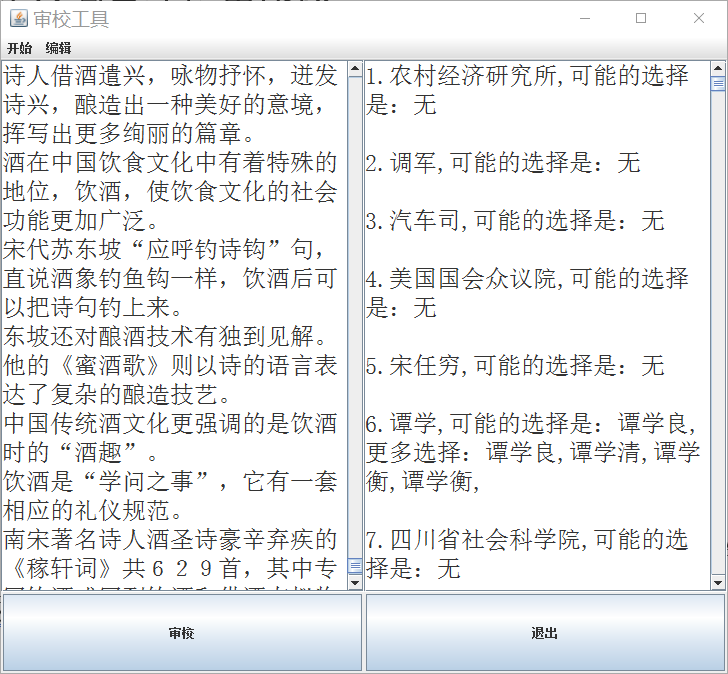


图5-1 测试结果局部截图

单独使用CRF分词，虽然分词的效果比较好，但是当实体名称出现错误是，CRF分词对其的识别并不会判定为实体名称，这样就导致了CRF分词完成后实体名称的数量会比实际的实体名称少，这种忽略错误名称的情况下所链接的实体正确率非常高；单独使用N最短路径分词则比较麻烦，因为N最短路径分词的花销相对CRF较大，同时精度达不到CRF分词的效果，但是N最短路径对于出错的实体名称的识别度比CRF要高很多，这样就较好的保证了原文的实体数目，但同时会有很多令人啼笑皆非的分词结果出现，导致分词结果不是实体名称的词语最终判定成了实体名称，这样的情况下自然是链接不到的，导致实验的链接失败率比较高；而实验中先使用CRF分词保证分词精度，然后对分词结果进行二次分词，这样不仅较为完整的保存了测试数据中的实体数目，同时较少了非实体项的错误判定而导致结果的偏差。

测试采用了一个分词的测试数据，因为是对分词方法进行比较，同时分词测试数据所包含的信息量比较丰富，包含的词语词性比较全面，更具有测试价值。图5-1是实验测试是的一个结果截图，可以看到左侧是打开的测试数据，右侧则是审校之后的结果。结果中可以看出，“调军”本身并不是一个实体名称，但是在分词识别是出现了错误的词性标注，可见CRF的分词效果并非完全精确，N最短路径常见的错误CRF也有出现的可能。

针对上述问题，综合考虑后，提出修改意见：原来的想法只是对CRF分词结果中的“其他特殊指称（/nz）”词性进行二次筛选，这样的想法是完全信任CRF分词的结果，但是CRF也有出错的时候，所以对CRF分词的结果中的实体指称全部进行N最短路径分词的二次审校，因为两种方法的工作机制并不相同，默认当两种分词方法同时认定某一词语是实体指称时，该词语就是一个实体指称。将改机后的方法也进行测试对比，与其他结果进行比较。

表5-1 测试数据的测试结果

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 使用方法 | 实体数目 | 链接数目 | 链接率 | 审校效率 | 正确率 |
| 原始数据 | 4365 |  |  |  |  |
| CRF分词 | 3290 |  |  |  |  |
| N最短路径分词 | 5304 |  |  |  |  |
| CRF+N最短路径 | 4541 | 3883 | 85.50% | 81.56% | 78.44% |
| CRF+N最短路径（改） | 4435 | 3626 | 81.75% | 82.95% | 79.15% |

通过表5-1可以看出，对于同样的测试数据，CRF分词所识别的实体数目比实际的实体数目小了很多，原因是因为对于很多的实体名称，比如测试数据中人为修改了“佟大为”变为“佟大伟”，前者CRF分词的结果没有错误（“/ns”），但是对于后者CRF分词并没有将其认定为实体名称，而是判定为了指代性的特殊专有名词（“/nz”），类似的情况有很多，从而导致了结果实体数量的减少。

N最短路径分词虽然不会出现CRF分词中错误词性标注的问题，但是分词的精度却达不到CRF分词，很多并不是实体名称的词语被判定为实体指称，比如原句中的短语为“五年就五年”，在使用N最短路径分词时，最开始的字“五”和最后一个字“年”被认为单字成词的分了出来，而中间的“年就五”三个字被拿了出来，因为“年”有姓氏的含义，所以“年就五”三个字就被N最短路径分词认定为是一个实体指称（人名），类似这样的结果还有很多，导致了分词结果中实体指称的数目比实际的数值要高出很多。

五组数据中的后两者为CRF分词和N最短路径分词的结合使用，可以看出两者的结果相差并不是很大，都较为接近实际测试数据中的实体数量。改就后的方案对实体指称的判定要求更高，所以链接率比改进前相对低了一点，但是因为精度有所提高，所以审校效率和审校的正确率都有了一定的提高。

考虑链接率、审校效率和成功率都不是很高，再次对实验数据以及实验结果，发现影响实验结果的原因是因为在测试数据中存在重复的实体，而一旦第一个实体链接失败或者没有正确链接，那么后续重复的实体结果也是一样，这样重复的数据导致错误的数据重复出现，出错实体的总数比真实的数据要大，这样的现象也出现在正确的实体集合中。最后采取措施，在审校前对实体集合进行梳理，将重复出现的实体去掉，保证测试数据的可靠性。

5.2.2 相似度算法应用比较

日常生活中的中文实体名称错误大多是由于拼音的错误导致，所以在进行相似度计算时拼音相似度就不能不说。而拼音相似度的计算方法有很多种，本实验中采用了词语编辑距离计算相似度的方法，因为这种方法可以同时计算词语以及由词语转换成的拼音两者的相似度，减少了运算步骤以及其他方法所需要的数据资料。使用相似度算法时需要设置阈值来进行限制，阈值设置的不同对结果也会有不同的影响。

表5-2 阈值对于词语编辑相似度影响折线图

阈值的设置关系到审校结果的精度，本实验中没有语义情况的考虑，所以模糊查找和链接审校时相似度算法就显得尤为重要，词语编辑相似度和拼音相似度有单独应用的时候也有一起使用的时候，阈值的变化也或多或少的改变着结果。

表5-2列举了词语编辑距离相似度的阈值变化是对于实验结果的影响。随着阈值的不断提高，对于相似度的标准也越来越严格，所以链接率不断走低；与之相反，在高相似度的限制下，链接的成功率却是逐渐走高的。所以并不是说阈值设置的越高越好，越高链接率越低，到时实体指称无法连接，实验的效果不会很好；也不能太低，阈值过低审校的正确率无法保证，审校的结果也很糟糕。综合考虑了上述情况，本实验单独使用词语编辑距离相似度时的阈值设置为0.8。

表5-3 阈值对于拼音相似度影响折线图

表5-3列出了一些阈值对于拼音相似度的影响，类似于词语编辑距离相似度，但是大部分情况下实体词语出错要么是拼音错误要么是同音字的存在，所以拼音相似度阈值确定时的链接率和正确率都比词语编辑相似度要高出一些，同时拼音相似度的阈值有1的存在。同样综合考虑各种情况，最后拼音相似度单独使用时的阈值定位0.9。

当两者共同使用时是在实验流程的规则4下使用的，毕竟从数据库中直接筛选候选项需要考虑的因素有很多，两种相似度同时使用能够提高精度。对于这两个的测试数据显示结果类似于表5-2和表5-3，并不在重复赘述，综合考虑后两者同时使用时词语编辑相似度的阈值设为0.85，拼音相似度的阈值设为0.8。

5.3 本章小结

本章对实验代码进行了数据测试，并对结果进行了分析。基于规则的实体链接有一定的缺陷性，但在本实验中也是可行的。基于规则的实体链接在相似度算法的辅助下也达到了一定的正确率。本实验最为核心的几步就是数据库的建立、数据的分词处理、基于规则的实体链接和最后审校结果的给出。通过对于分词算法的优化提高了实体名称的识别准确度，而改变相似度阈值则调整了链接的精确地。

分词算法的优化并不是唯一的，CRF分词可以通过大量语料的训练来提高分词精度，使用其他的分词方法也能够对分词效果进行优化。而相似度算法也不是唯一的，词语编辑距离相似度算法和本实验的情况比较相符，但还有其他的相似度算法可以使用，如笔画相似度等。阈值的设置给出了本实验的几组测试数据，但并不是唯一的，可以通过具体问题具体讨论。

总结和展望

本实验基于实体链接实现中文文本词语审校，考虑到各种情况，对于所用方法都有所简化。

实体抽取的过程取决于分词结果的词性标注，实验审校的实体集合的词性为人物名、地方名和机构名，分词的精度直接关系到实体抽取的好坏。实体链接的实现过程基于规则，考虑到大多数情况下可能出现的情况，总结出可以采用的审校规则并对规则的采用顺序进行了排列。对实验的结果进行了分析，并对实验中用到的参数进行了优化，是运行的结果达到了比较良好的状态。

实体链接技术将实体与数据库链接，链接的最好情况是综合考虑各种情况后的一对一的正确链接，这样的链接在文本审校中会有很好的应用：只要能够链接上，那么审校的成功率就会是100%。实验中根据测试结果可以看出，基于规则的实体链接的链接度和准确度都不算太高，一部分原因是因为规则的局限性，而另外一部分原因则是数据库的数据信息量不足。实体链接对于数据的要求分高，数据库数据量越大，链接的备选项越多，链接的成功概率则越高。

实体链接过程是将实体准确的链接到数据库中对应实体上，其深层次的含义是包含语义和语境的。本实验则是基于规则的实体链接，忽略了语义和语境的影响，所以在连接的准确性上并不够好。分词步骤的实现是使用分词器提供的分词方法直接分词，但方法并不是一种，如果可以，希望通过使用工具自己训练一个分词器来使用，这样训练出来的模型会和实验更加相配。基于规则的考虑有它的局限性，使用规则这一想法又是这样，规则是死的，而人是活的，不同的情况都会出现，不可能通过规则将所有情况完全考虑到，这样就对链接到效果产生了影响。

在未来的工作中，希望通过学习将分词效果进一步完善，然后将语义和语境的因素考虑到实验中，实现真正的实体链接。

致 谢

首先感谢我的导师李天瑞教授，从毕业设计题目确定开始，李老师就不断地跟进实验进展，对于实验也给了我很多参考意见，李老师严谨治学，认真工作的精神也深深地影响了我。

然后我要感谢云计算与智能技术实验室的师兄师姐们，在李老师较为忙碌的时候，学长学姐们经常挤出时间来给我们答疑解惑，同时也经常提出一些很有价值的意见，我们也经常参与学长学姐们的开会讨论，在和学长学姐们的交流中也时常有灵感迸发而出。不仅毕设工作上经常给我指导把关，平时的生活中学长学姐们也经常和我们沟通，让我对于未来的研究生生涯有了无比的憧憬。

我还要要感谢我的室友，在我懈怠的时候经常督促着我，在我沮丧的时候常常给我安慰和鼓励，在他们的带动下，我的实验也在不断地改进和完善，同时室友之间的“相互攀比”也让我更加拥有动力和活力。

最后，感谢我的家人，在精神和生活上都给予了我无私的帮助，有了他们的支持我才能够不断前行。

大学四年即将结束，有欢笑也有泪水，这些都将会是我最宝贵的回忆，都将作为我未来的生活前进的动力，希望大家都能够在未来谱写新的篇章。

参考文献

[1] 张鑫. 面向社会媒体的中文文本校对方法研究与实现[D]. 黑龙江大学, 2015.

[2] 刘亮亮, 曹存根. 中文“非多字词错误”自动校对方法研究[J]. 计算机科学, 2016, 43(10):200-205.

[3] 李岩. 基于深度学习的短文本分析与计算方法研究[D]. 博士学位论文. 北京科技大学, 2016.

[4] 舒佳根. 中文实体链接研究[D]. 硕士学位论文. 苏州大学, 2015.

[5] 王睿. 实体链接的研究与实现[D]. 硕士学位论文. 北京邮电大学, 2015.

[6] Moro A, Raganato A, Navigli R, et al. Entity Linking meets Word Sense Disambiguation: a Unified Approach[J]. 2014, 2(May):231-244.

[7] Demartini G, Difallah D E, Cudr, et al. ZenCrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking[C] International Conference on World Wide Web, 2012:469-478.

[8] 杨光, 刘秉权, 刘铭,等. 基于图方法的命名实体消歧[J]. 智能计算机与应用, 2015(5):52-55.

[9] 左乃彻. 基于维基百科的中英文命名实体消歧[D]. 硕士学位论文. 北京邮电大学, 2014.

[10] 郭宇航, 秦兵, 刘挺,等. 实体链指技术研究进展[J]. 智能计算机与应用, 2014, 4(5):9-13.

[11] 高艳红，李爱萍，段立国. 面向实体链接的多特征图模型实体消歧方法[J/OL]. <http://www.arocmag.com/article/02-2017-10-027.html>, 2016-10-12/2017-1-6

[12] 张仰森, 丁冰青. 中文文本自动校对技术现状及展望[J]. 中文信息学报, 1998, 12(3):50-56.

[13] Parikh A P, Täckström O, Das D, et al. A Decomposable Attention Model for Natural Language Inference[J]. 2016.

[14] 秦春秀, 赵捧未, 刘怀亮. 词语相似度计算研究[J]. 情报理论与实践, 2007, 30(1):105-108.

[15] 常建秋, 沈炜. 基于字符串匹配的中文分词算法的研究[J]. 工业控制计算机, 2016, 29(2):115-116.

[16] 李湘东, 高凡, 丁丛. LD A模型下不同分词方法对文本分类性能的影响研究[J]. 计算机应用研究, 2017, 34(1):62-66.

[17] Hachey B, Radford W, Nothman J, et al. Evaluating Entity Linking with Wikipedia[J]. Artificial Intelligence, 2013, 194(3):130-150.

[18] Nie P. An Automatically Learning Chinese Word Segmentation Algorithm Based on Particle Swarm Optimization[C] International Conference on Software Engineering and Information Technology. 2016:191-195.

[19] 刘亮亮, 曹存根. 基于局部上下文特征的组合的中文真词错误自动校对研究[J]. 计算机科学, 2016, 43(12):30-35.

[20] 张仰森, 丁冰青. 中文文本自动校对技术现状及展望[J]. 中文信息学报, 1998, 12(3):50-56.

[21] 傅士光, 林友芳, 万怀宇,等. 一种基于规则的中文分词算法[C] 中文信息处理国际会议. 2007.

[22] 余超, 蔡东风, 张桂平. 词汇语义相似度计算中相关技术的分析[C] 学术计算语言学研讨会. 2006.

[23] 陈正铭, 霍英. 编辑距离算法在中文文本相似度计算中的优化与实现[J]. 韶关学院学报, 2015, 36(12):8-12.

[24] 梁喜涛, 顾磊. 中文分词与词性标注研究[J]. 计算机技术与发展, 2015(2):175-180.