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文字信息。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，而审校工作交由人工进行则会耗费大量的时间和精力，同时还无法保证审校的正确率。在这种情况下，中文文本的审校工具应运而生。

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、义近等因素而导致的人名、地名、成语、机构名等实体名称出现错误，因为中文实体的数目非常多，审校的工作较为繁琐，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中文文本中出现的词语错误进行校对的开放系统，通过采用实体链接技术来完成对中文文本中的实体指称审校。本系统采用了将词语相似度和拼音相似度相结合来完成实体链接任务，根据实体链接的结果完成审校工作。具体流程如下：首先对需要审校的文本进行读取，经过分词处理后进行实体抽取，得到文本中的实体集合；然后对实体集合中的每一个实体应用相似度算法，在知识库中进行搜索匹配，完成实体链接；最后，根据实体链接的结果给出最终的审校意见，审校结果不会对原文进行修改，而是在可能出错的地方给出审校意见，便于用户进行选择。

关键词：大数据；实体链接；词语审校；相似度计算

Abstract

With the advent of the Big Date era, a large number of data resources have come into our lives. We have the access to more and more text information. To take publishing as an example, China has a lot of publishing. In the publishing work, the proofreading accounts for a large proportion of part. The staff member needs to determine whether there is any wrong information in the publication. The proofreading by manual will take a lot of time and effort, and the accuracy of the proofreading will not be guaranteed. In this case, Chinese text proofreading tools have come into being.

Word proofreading is an important part of the Chinese text proofreading. In Chinese texts, entity name errors, e.g., name, place name, idiom, organization name and so forth, often happen because of the factors such as similar sound, shape and semantics. The work of proofreading is complicated since the number of Chinese entities is very large. Presently, there is no open Chinese text proofreading system.

The purpose of this thesis is to construct an open system for correcting the errors of words in the Chinese text. Through the use of entity linking technology, the entity proofreading in the Chinese text is completed. The system uses the word similarity and phonetic similarity to complete the entity linking tasks and complete the proofreading work according to the results of the entity linking. Its concrete process is as follows: Firstly, the text that needs to be revised is read. The entity is extracted after the word segmentation and the entity set in the text is obtained. Then, the similarity algorithm is applied to each entity in the entity set. The search matching is carried out in the knowledge base and the entity linking is completed. Finally, the final proofreading advices are given according to the results of the entity linking. The original text will not be modified by the proofreading, however, the proofreading advices will be given in the possible wrong place which are easy for the user to choose.

Keywords: Big Data; Entity linking; Word review; Similarity calculation

目 录

[摘 要 III](#_Toc484272681)

[Abstract IV](#_Toc484272682)

[第1章 绪 论 1](#_Toc484272683)

[1.1 背景与意义 1](#_Toc484272684)

[1.2 国内外发展现状 2](#_Toc484272685)

[1.3 论文所做工作及思路 3](#_Toc484272686)

[1.4 论文章节安排 3](#_Toc484272687)

[1.5 本章小结 4](#_Toc484272688)

[第2章 相关知识概述 5](#_Toc484272689)

[2.1 MongoDB数据库 5](#_Toc484272690)

[2.1.1 MongoDB数据库简介 5](#_Toc484272691)

[2.1.2 MongoDB数据库特点 5](#_Toc484272692)

[2.1.3 基于百度百科的数据库 6](#_Toc484272693)

[2.2 实体链接 6](#_Toc484272694)

[2.2.1 背景与意义 6](#_Toc484272695)

[2.2.2 实体链接基本概念 7](#_Toc484272696)

[2.3 中文分词 7](#_Toc484272697)

[2.3.1 CRF分词 8](#_Toc484272698)

[2.3.2 N-最短路径分词 8](#_Toc484272699)

[2.4 词语相似度算法 9](#_Toc484272700)

[2.5 本章小结 10](#_Toc484272701)

[第3章 系统设计 11](#_Toc484272702)

[3.1 系统流程 11](#_Toc484272703)

[3.2 系统模块详解 11](#_Toc484272704)

[3.2.1 用户界面 11](#_Toc484272705)

[3.2.2 分词算法 12](#_Toc484272706)

[3.2.3 相似度算法 14](#_Toc484272707)

[3.2.4 实体链接 16](#_Toc484272708)

[3.3 本章小结 17](#_Toc484272709)

[第4章 实验及结果分析 18](#_Toc484272710)

[4.1 实验运行环境 18](#_Toc484272711)

[4.2 数据库配置 18](#_Toc484272712)

[4.2.1 数据库安装 18](#_Toc484272713)

[4.2.2 数据库设计 19](#_Toc484272714)

[4.3 Java环境配置 20](#_Toc484272715)

[4.3.1 JDK环境配置 20](#_Toc484272716)

[4.3.2 连接MongoDB数据库 21](#_Toc484272717)

[4.4 HanLP分词器配置 21](#_Toc484272718)

[4.4.1 Java配置HanLP分词器 21](#_Toc484272719)

[4.4.2 启用自定义词库 21](#_Toc484272720)

[4.5 实验测试 22](#_Toc484272721)

[4.5.1 实体链接率 23](#_Toc484272722)

[4.5.2 链接效率 23](#_Toc484272723)

[4.5.3 链接正确率 23](#_Toc484272724)

[4.6 实验结果分析 23](#_Toc484272725)

[4.6.1 分词算法测试 23](#_Toc484272726)

[4.6.2 相似度算法阈值测试 26](#_Toc484272727)

[4.7 本章小结 27](#_Toc484272728)

[总结和展望 28](#_Toc484272729)

[致 谢 29](#_Toc484272730)

[参考文献 30](#_Toc484272731)

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，如果单纯的通过人工进行校对，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，自动校对[1]这一术语被提了出来。自动校对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和社会的进步，不断地进行提升和优化。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，小到个人文章校对、论文审查，都可以用到自动校对。

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。本论文以相似度计算和知识库为基础，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现实世界中的哪个实体，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。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不断完善，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，对于无法链接的实体，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链接的情况来补充数据库，根据实体链接的情况来完善知识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。

本项目目的在于对给定文本，将其中的实体进行抽取识别并链接到知识库中，通过与链接到的记录进行匹配，从而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[2]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；刘亮亮和曹存根[3][23]针对中文“非多字词错误”自动校对方法也进行了研究；李岩[4]给出了一个基于深度学习的短文本分析和计算方法，具有良好的性能。

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。实体链接是实体消歧的一种重要方法，该方法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[5]。它能够丰富文本语义信息，在自然语言处理、信息检索等领域有着广泛的应用前景[6]。中文实体链接的研究在近几年也有所起步，众多国内外学者纷纷加入研究的行列。Moro等[7]采用实体链接对词语进行消歧，并提出了一种同一方法；Demartini等[8]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；陈玉博[9]等则提出了融合多个特征进行实体链接并进行了试验，效果理想。大多数的实体链接研究都是基于维基百科的数据而进行的英文链接，而中文实体链接的研究随着理论不断充实也在不断发展，杨光和刘秉权[10]等将图论研究引入中文研究进行了中文实体消歧算法的研究；刘峤[11][24]等也应用图论的方法完成了实体链接算法的实现；张涛[12]等则是在图模型的基础上引入了维基概念，并实现了一个实体链接的系统；罗念[13]等总结已有的实体链接技术，对未来的发展进行了展望。实体链接技术方兴未艾，还有很大的研究空间。

随着实体链接技术的发展，中文文本处理的方式也越来越多，对语料的预处理则显得尤为重要。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，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使得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，尤其是在中文文本审校方面有着很大的优势，但同时还有很大的研究空间。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链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知识库进行中文词语审校。

1.3 论文所做工作及思路

本文工作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，在得到要审校的文本时，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，然后将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。主要过程如下：

（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，结合相似度算法在知识库中进行匹配。设置相似度阈值，对相似度高于阈值的实体记录进行统计。

（5）相似度越高说明两个实体匹配度越高，选取相似度最高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，包括项目的意义背景及发展过程、实验的基本流程和论文的工作安排。

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章介绍系统的算法设计和运行流程，主要包括相似度算法的实现过程，系统的具体流程以及各步骤的详细实现方法。

第4章介绍了系统的环境配置和测试数据分析。主要包括数据库以及分词器的配置，Java环境的配置以及与数据库、分词器的连接，同时对系统的测试数据进行分析，展示系统的性能。

1.5 本章小结

本章首先介绍了本文的研究背景和意义，概述了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。最后给出了本文的章节安排。

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事先定义表结构，非关系型数据库的一大特点是并不需要预先定义数据模式，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较为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，数据结构由键值对组成，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

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）。

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，其语法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。图2-1给出MongoDB数据库查找记录的截图。可以看到通过查询语句可以轻松定位想要查找的数据记录，同时结果的显示也较为规整，均为“键值：内容”格式的显示，相比于其他的非关系型数据库来说更加便于用户解读。

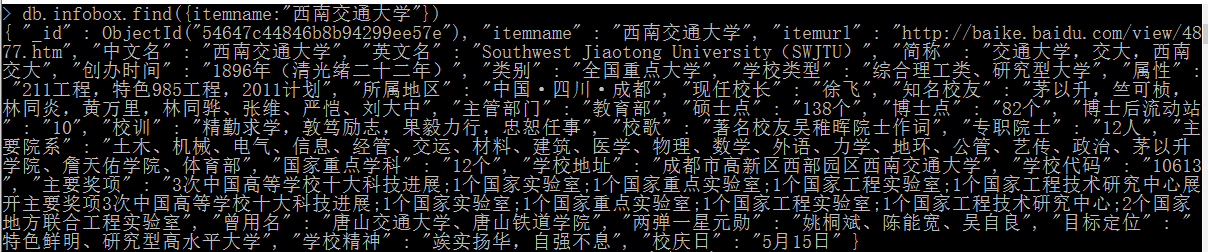


图2-1 MongoDB数据库查找记录的截图

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。百度百科对每一条数据都有一种非常规格化的数据样式，如图2-2，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。

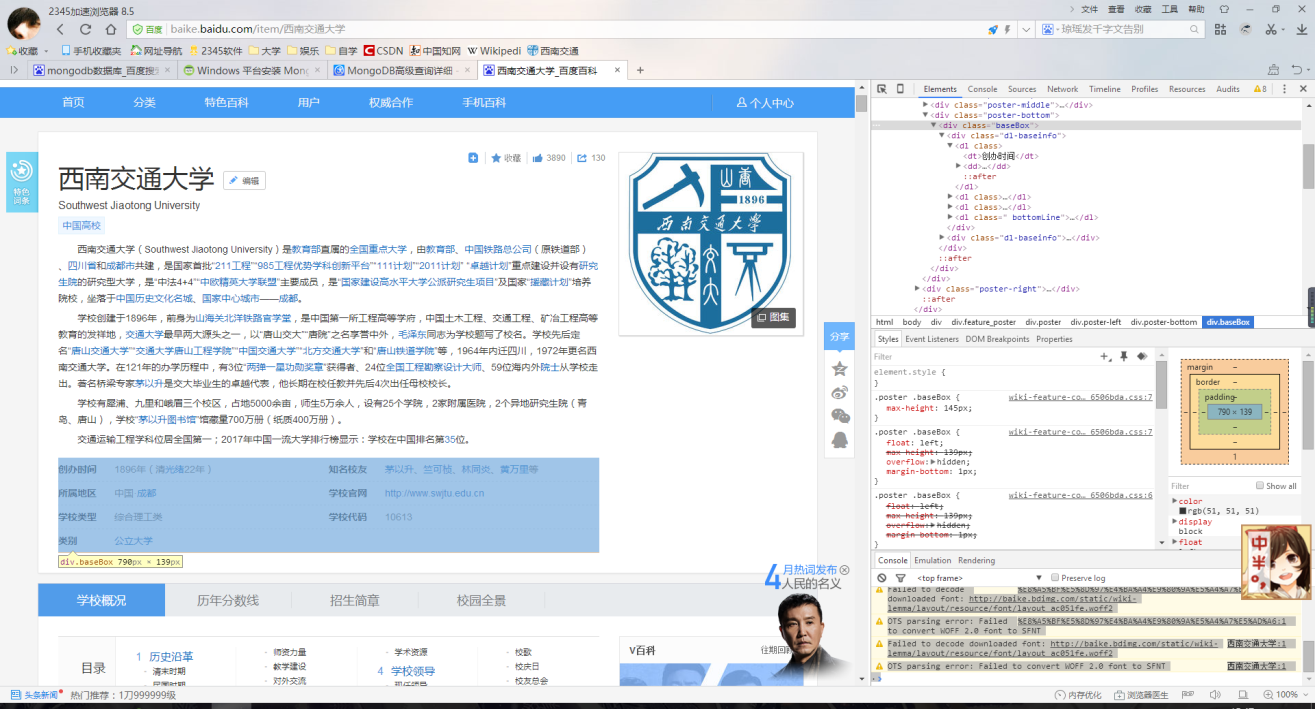


图2-2 百度百科表格数据

2.2 实体链接

2.2.1 背景与意义

实体链接技术早在2009年提出后，就有众多学者参与研究。作为自然语言处理的一项任务，实体链接在知识库填充、信息检索以及实体消歧方面等方面具有重要的实用价值。

实体链接是命名实体消歧的重要方法，该方法通常在知识库中搜索候选实体，然后消除歧义，返回最佳候选人或零[14]。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[15]，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，处理时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。在中文文本处理时都要对文本进行分词处理，所以中文分词的结果在很大程度上决定了系统结果的好坏[16]。本系统使用了开源的Java分词器HanLP分词器，其自身集成了很多分词方法。系统使用了CRF分词方法和N最短路径分词方法。

2.3.1 CRF分词

CRF（Conditional Random Field algorithm）算法全称为条件随机场算法，最早由Lafferty提出，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[17]。条件随机场算法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，但缺点对于一些实体名称的词性标注并不理想。

2.3.2 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[18]。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值作为权重提高对低频词的分词准确度，其中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值实际上就是TF\*IDF的结果。

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，路径上的边囊括了最有可能的分词结果，通常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”。

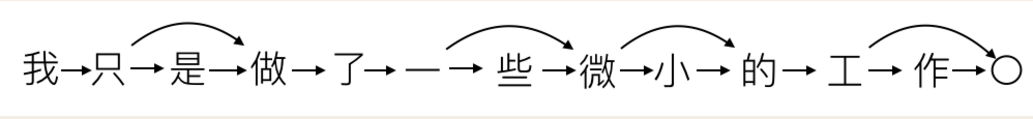


图2-3 “我只是做了一些微小的工作”形成的有向无环图

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，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，在知识库中进行链接时，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，由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：

 （2-1）

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（2-2）所示：

 （2-2）

其中*m*为*S*的字符串长度，*n*为*T*的字符串长度。当*S*与*T*完全相同时，*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，如公式（2-3）所示：

  （2-3）

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

本章介绍了系统中所用到的内容相关知识的介绍，首先介绍了非关系数据库MongoDB的概念及特点，其次详细介绍了实体链接技术的背景意义与基本概念，随后介绍了两种中文分词算法，并比较了两种分词方法的优劣，最后介绍了相似度算法的计算原理。

第3章 系统设计

本章介绍了系统的具体实现流程和关键算法的实现方法。

3.1 系统流程

图3-1给出了整个系统的基本流程。系统针对给出的文本数据，在知识库中进行链接。系统中较为重要的两步为文本分词和基于规则进行链接。



图3-1 项目流程图

3.2 系统模块详解

3.2.1 用户界面

用户图形界面（Graphical User Interface，简称GUI），又称为图形用户接口，指的是采用图形方式显示的计算机用户界面。本系统有一个基于Java语言编写的简易图形化界面，不仅方便使用者操作，同时能够清晰明了地显示实验结果以及审校意见。

本系统通过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utf-8编码格式，否则读取时会显示乱码。审校按钮会调用系统的审校功能。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在左侧对原始文本进行修改。界面提供了文本保存功能，这样在审校结果给出的时候，用户就可以直接根据审校意见来对原有文本进行修改。图形化界面如图3-2所示。



图3-2 用户图形化界面

基于规则的实体链接方法的应用只是单独地从词语本身来考虑，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，这样就对审校结果造成了困难。

考虑到上述情况，审校结果是建议性的，并不会在原文上进行修改，这就是GUI界面上设计两个编辑框的原因。审校结果会指出哪个实体指称出错，并给出一个最有可能的结果。由于没有考虑语义和语境，所以在给出推荐结果时会加上一些参考项供用户参考，方便用户进行考虑选择

3.2.2 分词算法

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

前面已经介绍过CRF分词和N最短路径分词的原理，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，很多词都没有进行分割，而很多不是词的词则被认为是实体名词，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。所以对分词算法进行调整，详细流程如图3-3。



图3-3 文本分词流程详解

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[20]，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”），而只将词性标注为其他专属名词（“/nz”）。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，并判断新分词后的结果，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。

除了分词通过影响实体集合的抽取间接影响系统结果外，实体重复项的存在也严重影响着系统结果。这些重复项的存在不仅影响着系统运行时的效率，同时对审校结果也产生了干扰。因此针对实体重复项提出改进：对每一个抽取的实体指称项进行记录，当遇到实体指称时，首先判断该实体是否已经完成审校，如果是，则跳过当前实体，否则更新实体集合，进行审校。根据测试结果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。改进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，所以链接效率和审校的正确率都有了一定的提高。

3.2.3 相似度算法

前面已经介绍过相似度算法的计算原理，这里着重讲述算法的具体实现过程。考虑到最终结果为词语审校，综合考虑各种可能的情况，分为两个部分实现了相似度算法。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指代两种相似度。

首先考虑相似度特征较为明显的特殊情况。这种情况下两个词语的相似度不需要通过算法计算就可以得出，如果再使用算法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，所以对特殊情况的考虑的均是基于编辑距离的情况。

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。

上述的几种情况是不用通过算法的复杂计算就可以直接计算出两个词语或拼音的相似度，这样会节省很多时间和资源，提高系统的运行效率。接下来实现相似度算法。

首先介绍词语编辑距离的方法实现。词语编辑距离的计算是对于长度分别为*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算法填充，当矩阵填充完整时，的值就是两个字符串的编辑距离[19]。得到编辑距离后根据归一化编辑距离相似度的计算公式（2-3）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。

首先这两个字符串不符合特殊情况中的任何一种，所以直接使用算法进行计算。进行算法处理时，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阶的矩阵，矩阵的（0,0）位置为0，首行和首列为从0开始的依次递增填充，填充的数字不仅可以表示为行列号，同时也作为参数参与矩阵运算。如表3-1所示，字符串显示的行列不计算在矩阵中，只是为了方便对算法进行解释分析而添加在矩阵的上方和左侧。下面进行计算。

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3-2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。

表3-1 d矩阵初始化 表3-2 d矩阵首位计算

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

通过表3-3和表3-4的计算，可以得出两个字符串的词语编辑距离为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1-1/5=0.8。

表3-3 d矩阵后续计算 表3-4 d矩阵计算结果

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

通过上述步骤，最后可以得到两个词语的相似度。