**面向重复记录检测的数据清洗算法的研究**

**作者姓名 张 攀**

**学校导师姓名、职称 刘志镜 教授**

**企业导师姓名、职称 吴春苗 高工**

**申请学位类别 工程硕士**

**面向重复记录检测的数据清洗算法的研究**

**西安电子科技大学**

**硕士学位论文**

**作者姓名：**张 攀

**领　　域：**计算机技术

**学位类别：**工程硕士

**学校导师姓名、职称：** 刘志镜 教授

**企业导师姓名、职称：** 吴春苗 高工

**学　　院：**计算机学院

**提交日期：**2017年4月

**学　号　 1503121785**

**密　级　 公开**

**学校代码 10701**

**分类号 TP39**

Supervisor: Liu Zhijing

Supervisor: Wu ChunMiao

Title: Professor

Title: Senior Engineer

A thesis submitted to

XIDIAN UNIVERSITY

in partial fulfillment of the requirements

for the degree of Master

in Computer Technology

**Research of Data Cleansing Algorithm for Duplicate** **Elimination**

By

Zhang Pan

June 2017

**西安电子科技大学**

**学位论文独创性（或创新性）声明**

秉承学校严谨的学风和优良的科学道德，本人声明所呈交的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，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果；也不包含为获得西安电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同事对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

学位论文若有不实之处，本人承担一切法律责任。

本人签名： 日 期：

**西安电子科技大学**

**关于论文使用授权的说明**

本人完全了解西安电子科技大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权属于西安电子科技大学。学校有权保留送交论文的复印件，允许查阅、借阅论文；学校可以公布论文的全部或部分内容，允许采用影印、缩印或其它复制手段保存论文。同时本人保证，结合学位论文研究成果完成的论文、发明专利等成果，署名单位为西安电子科技大学。

保密的学位论文在 年解密后适用本授权书。

本人签名： 导师签名：

日 期： 日 期：

摘要

随着信息技术的发展与信息化建设的深入研究，激烈的市场竞争对于企业信息化程度的要求也越来越高。在形形色色的企业信息化系统进行不断地迭代和重构时，数据库中积累了大量的脏数据，主要包括错误数据、相似重复数据和缺失数据等类型。这些脏数据产生的原因多种多样，例如数据来源不同，存储于不同的操作系统以及硬件平台等。其中多源数据的归并造成的数据重复则是最关键的热点问题。

本文在对相似重复记录技术发展和研究现状进行简要介绍的基础上，首先详细地阐述了基于不同实现方式的重复检测技术，然后在前人工作成果的基础上，基于MPN算法提出了改进的IMPN算法。IMPN算法主要有以下三个创新：（1）传统的MPN算法主要依赖相关领域的专家经验生成排序关键字，不利于处理未知领域或未知类型的数据，IMPN算法使用基于统计字段区分度的方法进行排序关键字的选取，有效地修正了MPN的缺点；（2）IMPN使用动态调整滑动窗口大小的机制，与传统MPN算法所采用的固定大小的滑动窗口相比，算法操作过程更加灵活，可以更加全面地检测重复记录，同时，有效地避免了不必要的时间消耗，提高了算法运行效率；（3）对于排序关键字为空的记录，对其进行预标记，改善了MPN算法在处理排序关键字不完整的情况下的缺陷，提高算法应对缺失字段的能力，增强了鲁棒性。

虽然IMPN算法对MPN算法进行了三方面的改进，但是，随着数据量的增加，IMPN算法的查准率下降的问题随之凸显。对于这一问题，本文基于遗传算法和人工神经网络在相似重复记录检测领域的研究，提出了AMPN算法。AMPN算法将基于遗传算法改进的神经网络GA-ANN算法应用到IMPN算法的重复记录判断中，提高了算法的查准率。传统的GA-ANN算法检测相似重复记录通常做法是，首先通过带标签的数据集作为训练数据训练出多个BP神经网络，然后，将每一个训练完成的神经网络作为遗传算法的个体，所有神经网络构成遗传算法的种群，利用遗传算法使用全局信息的优势，弥补神经网络易陷入局部极小值的缺点。但是，GA-ANN在测试阶段，若对测试数据集中的所有记录进行全排列，然后使用训练好的神经网络判断重复记录，时间复杂度较高。本文综合GA-ANN算法与IMPN算法的优势，提出AMPN算法，仅使用训练好的网络对滑动窗口内的记录进行检测，提高了IMPN的查准率，同时减少了不必要的判断，改进了GA-ANN的运行效率。该算法相对于IMPN算法的主要优势在于，通过神经网络学习属性之间的内在关系来完成记录是否匹配的判断，随着环境的变化具有较强的自适应能力。经对比实验证明，AMPN算法的查准率取得了显著提升。

最后本文将相似重复记录检测技术应用到航天情报信息管理系统中，用以解决多源数据合并造成的重复记录问题，提高了数据质量。

**关键词**：数据清洗，相似重复记录检测，BP神经网络，机器学习，遗传算法

ABSTRACT

With the development of information technology and information construction, fierce market competition drives the enterprises to improve their information management. During the process of iteration and reconstruction of EIS, databases have accumulated a lot of “dirty data”, which consists of error data, duplicate data, missing data and so on. There are varies of reasons of these data, such as the different data source, the different OS, the different hardware and so on. The key issue of these data cleansing problem is deduplication caused by the merge of multi-source data, which becomes the research focus nowadays.

In this paper we introduce the development and research status of the duplicatoin detection technology firstly, including their theory and implemention. Then a promoted algorithm called IMPN is present on basis of previos work. There are three improvement, decreasing the dependence of expertise and experience by the statistic of fields’ discrimination when choosing sort key, decreasing time consumption and the amount of missing duplicate records by dynamically adjusting the size of sliding window, and promoting rubust by marking those records whose sort key is null.

The precision ratio’s decreasing becomes a fierce problem as the data grows too large. This paper provides a solution that the neural network improved by genetic algorithm is applied to the IMPN to judge whether two records are duplicate. Firstly, a small dataset is used to be the input of a BP network. After training process, the network will evolve as the chromosome, then this network could rapidly decide whether two records are the same entity with the similarity vector being the input. This solution’s advantage is that it gets the result through the inner relation of records rather than estimates the fields’ weight directly, which enhances its self-adaption ability as the environment changes. This algorithm reaches a higher accuracy as experiments prove.

At last, the duplicate record detection method is applied in the spaceflight information management system in order to eliminate those duplicate records which is caused by the merge of multi-source data, that improves the data quality.

**Keywords:** Data Cleansing, Duplicate Record Detection, BP Neural Network, Machine Learning, Genetic Algorithm

插图索引

**未找到图形项目表。**

表格索引

**未找到图形项目表。**

**未找到图形项目表。**

符号对照表

exp 以e为底的指数函数

min 取最小值函数

∑ 求和

缩略语对照表

EIS Enterprise Information System 企业信息化系统

SNM Sorted Neighborhood Method 近邻排序算法

MPN Multi-Pass Sorted Neighborhood 多趟近邻排序算法

IMPN Improved Multi-pass Sorted Neighborhood 改进的多趟近邻排序算法

ANN Artificial Neural Network 人工神经网络

BP Back Propagation 反向传播

GA · Genetic Algorithm 遗传算法

GA-ANN GA-based Artificial Neural Network 基于遗传算法改进的人工神经网络

目录

[摘要 I](#_Toc510644166)

[ABSTRACT III](#_Toc510644167)

[插图索引 V](#_Toc510644168)

[表格索引 VII](#_Toc510644169)

[符号对照表 IX](#_Toc510644170)

[缩略语对照表 XI](#_Toc510644171)

[第一章 绪论 1](#_Toc510644172)

[1.1 研究的背景和意义 1](#_Toc510644173)

[1.2 国内外研究现状 2](#_Toc510644174)

[1.3 论文研究的主要内容 3](#_Toc510644175)

[1.4 论文结构 4](#_Toc510644176)

[第二章 重复记录检测相关算法概述 5](#_Toc510644177)

[2.1 相似重复记录概述 5](#_Toc510644178)

[2.2 相似度匹配算法 6](#_Toc510644179)

[2.2.1 基于单字段相似度匹配 6](#_Toc510644180)

[2.2.2 基于多字段相似度匹配 7](#_Toc510644181)

[2.3 相似重复记录检测算法 8](#_Toc510644182)

[2.3.1 近邻排序算法 8](#_Toc510644183)

[2.3.2 多趟近邻排序算法 9](#_Toc510644184)

[2.3.3 其它算法 11](#_Toc510644185)

[2.4 BP神经网络理论基础 12](#_Toc510644186)

[2.4.1 神经元模型 12](#_Toc510644187)

[2.4.2 梯度下降法 13](#_Toc510644188)

[2.4.3 BP网络前向传播和反向传播 14](#_Toc510644189)

[2.6 算法的衡量标准 33](#_Toc510644190)

[2.7 本章小结 17](#_Toc510644191)

[第三章 改进的IMPN算法 19](#_Toc510644192)

[3.1 基于字段区分度提取关键字的方法 19](#_Toc510644193)

[3.1.1 传统的提取关键字的方法 19](#_Toc510644194)

[3.1.2 改进的字段区分度方法 22](#_Toc510644195)

[3.2 自适应大小的滑动窗口检测方法 24](#_Toc510644196)

[3.2.1 传统的滑动窗口检测方法 24](#_Toc510644197)

[3.2.2 改进的自适应大小的滑动窗口 25](#_Toc510644198)

[3.3 基于标记的处理不完整排序关键字方法 27](#_Toc510644199)

[3.3.1 MPN的排序方法存在的问题 27](#_Toc510644200)

[3.3.2 改进的基于标记的方法 28](#_Toc510644201)

[3.4 IMPN算法设计 28](#_Toc510644202)

[3.4.1 算法流程设计 29](#_Toc510644203)

[3.4.2 时间复杂度分析 31](#_Toc510644204)

[3.5 实验设计与结果分析 31](#_Toc510644205)

[3.5.1 实验数据介绍 32](#_Toc510644206)

[3.5.2 SNM、MPN、IMPN综合对比实验 33](#_Toc510644207)

[3.6 本章小结 38](#_Toc510644208)

[第四章 基于遗传神经网络改进的IMPN算法 41](#_Toc510644209)

[4.1 遗传神经网络用于相似重复记录检测 41](#_Toc510644210)

[4.1.1 BP神经网络的设计 41](#_Toc510644211)

[4.1.2 遗传神经网络与重复记录检测 43](#_Toc510644212)

[4.2 改进的遗传神经网络 47](#_Toc510644213)

[4.2.1 神经网络过拟合现象 47](#_Toc510644214)

[4.2.2 “Dropout”防止过拟合 49](#_Toc510644215)

[4.3 对比实验 49](#_Toc510644216)

[4.4 本章小结 51](#_Toc510644217)

[第五章 航天情报系统中的相似重复记录检测 53](#_Toc510644218)

[5.1 系统需求分析 53](#_Toc510644219)

[5.1.1 系统建设背景与目标 53](#_Toc510644220)

[5.1.2 需求分析 53](#_Toc510644221)

[5.2 系统设计与实现 54](#_Toc510644222)

[5.2.1 系统概要设计 54](#_Toc510644223)

[5.2.2 系统实现 57](#_Toc510644224)

[5.3 数据清洗模块 59](#_Toc510644225)

[5.3.1 “脏数据”产生原因 59](#_Toc510644226)

[5.3.2 重复记录检测算法的应用 60](#_Toc510644227)

[5.3.3 实验设计与结果分析 60](#_Toc510644228)

[5.4 本章小结 61](#_Toc510644229)

[第六章 总结与展望 63](#_Toc510644230)

[参考文献 67](#_Toc510644231)

[致谢 71](#_Toc510644232)

[作者简介 73](#_Toc510644233)

# 第一章 绪论

## 1.1 研究的背景和意义

人类正在由工业化时代进入信息化时代，经济学家们普遍认为，进入21世纪后，信息将成为第一生产要素，同时将构成信息化社会的重要技术物质基础。为了在激烈的市场竞争中占据先机，各行业如保险、金融等纷纷加快了信息化的步伐。随着数据库技术的快速发展和广泛应用，形形色色的企业信息化系统应运而生，数据库的信息量也与日逐增。

从规模庞大的数据库中提取重要信息，从而对企业单位的发展提供参考，为决策者提供技术支持，是近年来数据挖掘的研究重点。由于不可避免的人为录入错误，或者是不同的数据表示方法，抑或是从不同的数据源合并数据甚至数据存储于不同的操作系统和物理设备，都不可避免地降低了系统的数据质量，从而产生各种“脏数据”。脏数据的类型主要包括重复数据、不完整数据、错误数据等[[[1]](#endnote-2)][[[2]](#endnote-3)]。如果这些数据不能被正确清洗，则会影响信息化系统的正确运行，使得数据中提取的信息不再可靠，为企业决策支持和商务应用带来负面影响。因此，为了确保数据的准确性、一致性，数据清洗显得尤为重要。

最早的数据清洗过程需要大量的人为操作，所以当遇到较大规模的数据集，就会凸显出人为操作的低准确性和低效性。所以在当前数据规模急剧加大的情况下，只有借助计算机技术，数据清洗才能实现其高效性。目前的信息化清洗过程中，仍不能完全离开专家的经验、人工的操作等行为，所以研究的一个重要方向就是尽可能减少人为的参与和影响[[[3]](#endnote-4)]。

相似重复的记录是数据库中降低数据质量最重要的一个原因，所以如何高效地检测和去除重复数据是数据清洗研究范畴的一个热点问题[[[4]](#endnote-5)][[[5]](#endnote-6)]。

同一个实体在数据库中不同的展现形式是相似重复记录的本质，它主要会引发以下的问题：

（1）资源浪费：重复记录会造成数据冗余，导致存储空间的极大浪费。

（2）破坏数据一致性：相似重复记录之间的关系可能是互为补充，也可能存在部分的冗余，甚至互相矛盾。它们共同对应的现实中的实体发生变化会导致这些记录中只有某个或者某些记录发生改变，而其余无法同步更新。

相似重复记录的检测与消除，保证了数据的一致性、减少资源的浪费，是数据清洗的重要环节。

## 1.2 国内外研究现状

早在上个世纪50年代，数据清洗已经开始了相关研究。将出自不同数据源的数据集进行整合被认为是一个困难而且极为重要的问题，最早的研究主要是从数据连接[[[6]](#endnote-7)]、数据实体识别[[[7]](#endnote-8)]、对象识别等问题来展开，是商业保险、医疗、等领域中的研究重心之一。美国清除全美社会保险号数据集中的错误数据被视为数据清洗技术研究的开端[[[8]](#endnote-9)]。

数据清洗的研究重点包括：重复记录检测、异常数据检测、缺失数据的处理。数据仓库的出现以及数据挖掘相关技术的发展和应用，造成了多源数据进行合并容易出现大量重复数据的问题。因而相似重复记录的检测与清除成了数据清洗领域的研究重点。

在重复记录清洗方面，国外展开了大量的研究，主要的工作有两个方面——属性匹配和重复检测。属性匹配问题的解决方法[[[9]](#endnote-10)]主要有编辑距离算法、Smith-Waterman算法、递归属性匹配算法等。

相似重复记录检测领域一种主流的算法是“排序/归并”法，即先将数据连接成一整个数据集，之后按照某种规则进行排序，将相似重复的记录排列在附近，最后通过某种相似判断方法检测出重复的记录。最基本的算法是Jaro提出的“排序/合并”（Merge/Purge）算法[[[10]](#endnote-11)]。这种算法存在明显的缺点，许多研究人员在此基础上提出了各种各样的改进思路和算法实现，主要的改进方向包括对字段相似度匹配算法的改进和对相似记录判断方法的改进。

Monge等人将数据库中的一条记录视为一个字符串，在排序和比较的时候采用优先级队列的方法，检测相似重复时则使用了基于字符串的编辑距离[[[11]](#endnote-12)]。Hernandez等提出了多趟近邻排序算法（Multi-Pass Sorted Neighborhood，记为MPN）[[[12]](#endnote-13)][[[13]](#endnote-14)]，该算法独立地执行多次近邻排序算法（Sorted Neighborhood Method，记为SNM），每次采取不同的排序关键字段以及较小的滑动窗口，最后使用C语言重写的OPS5[[[14]](#endnote-15)]规则编程判定记录是否相似。Qiu首先计算每条记录的N-gram统计值，然后根据这个N-gram值对数据集进行排序，最后再用用优先级队列的方式聚类检测重复记录[[[15]](#endnote-16)]。Gianni Costa等人采用文本聚类中的增量技术将新数据划分到最近的已知重复的聚类中，解决了大文本库中的相似检测问题[[[16]](#endnote-17)]。Alfredo Ferro使用了基于q-grams的相似度衡量函数[[[17]](#endnote-18)]，可以避免许多不必要的比较和判断，提高了时间效率。

国内的相关研究主要是对已知算法的改进和创新以实现更高的精度和效率。复旦大学周傲英等比较早开始数据清理的研究工作[[[18]](#endnote-19)]。邱越峰等提出了基于N-Gram的相似记录检测算法[[[19]](#endnote-20)]。算法以一条数据的N-Gram值作为排序键，该算法在对因为拼写错误而造成的重复记录进行检测时表现良好。陈伟提出了基于权重进行相似重复记录检测的方法[[[20]](#endnote-21)]，具体实现按照字段的等级划分权重，并结合长度过滤的思路减少冗余的字段相似度计算。

在数据清洗市场化领域，国内外涌现了一批优秀的数据清洗软件以及框架，包括商业上和各大学以及研究机构开发的数据清洗软件[[[21]](#endnote-22)][[[22]](#endnote-23)]。

在相似重复记录检测领域，国内外研究人员已取得了诸多进展，但仍旧或多或少存在适用局限性或者检测效率和精度不足等问题，所以仍然有研究的价值和改进的空间。

## 1.3 论文研究的主要内容

由目前的研究现状可以看出，相似重复记录检测领域的发展已取得诸多有效成果，但仍旧存在一定问题，主要体现在：

（1）检测效率与查全率存在提升空间，尤其是较大数据量的相似重复检测问题。

（2）大多数数据清理只针对特定领域以及业务场景，各行业需要更加通用的相似记录检测方案。

（3）相似重复记录检测大多基于“排序/归并”的思想，排序的效果以及最终归并的结果受排序关键字影响较大，尤其是当数据库排序关键字对应的字段为空或者是错误数据时，部分重复记录无法被正确的检测到，从而影响数据清洗的质量。

如何高效地检测相似重复记录，进而剔除数据库中的冗余数据，一直是数据清洗研究的重点问题。本文在分析了常用相似重复记录检测算法的基础上，针对传统的多趟近邻排序算法MPN在时间消耗和检测精度的不足，提出了改进的IMPN算法。IMPN算法有三个改进点：

（1）通过统计字段区分度改善了传统的MPN算法在选择排序关键字时过于依赖专家经验的缺点。

（2）通过动态调整滑动窗口大小以节约时间并减少被遗漏的重复记录。

（3）通过标记排序关键字为空的记录提高算法应对缺失字段的能力，增强了鲁棒性。

随着人工神经网络研究的兴起，越来越多的跨学科研究正在如火如荼地展开。本文将反向传播神经网络应用于相似重复记录检测，将两条记录对应字段间的相似度组成的向量作为神经网络的输入，利用有监督的学习训练出多个三层BP神经网络，然后利用遗传算法将这些神经网络组成的种群进行迭代优化，选择出适应度最好的个体，这样可以克服BP神经网络容易陷入局部最小值的缺点。将遗传神经网络判断两条记录是否相似的方法应用到IMPN算法中，提高了IMPN算法的查准率。

## 1.4 论文结构

论文一共分为六章，每一章的主要内容如下：

第一章是绪论。主要介绍了的数据清洗研究的背景和意义，相似重复记录的国内外的研究与发展现状，简单描述了论文的主要研究目的以及研究内容，展示了论文的组织架构。

第二章，主要介绍了相似重复记录检测的相关算法。首先简单介绍了衡量字段相似度的相似度检测有关算法，分析了它们各自的优缺点以及适用条件。然后介绍了最基本的近邻排序算法和多趟近邻排序算法，包括了算法的基本原理、设计思路以及算法步骤和算法的优缺点。除此之外还介绍了其它常用算法包括优先级队列算法、N-Gram算法等。然后对本文用到的BP神经网络理论基础进行说明。最后介绍了相似重复记录检测领域衡量算法的几个常用标准及其计算方法。

第三章，首先介绍了改进的IMPN算法的提出背景，然后详细介绍了算法的设计思路和其改进点，并采用SNM算法和MPN算法作为对比，通过实验验证了IMPN算法查全率较高的优势，并分析了算法的缺点，即随着数据量的增大查准率不够理想。

第四章，首先介绍了神经网络在相似重复记录检测中的应用，以及遗传算法对其的改进，并说明了如何使用改进的BP神经网络进行相似重复记录检测。接下来重点介绍了遗传BP神经网络应用于IMPN算法中，并通过实验验证了遗传BP神经网络对IMPN算法查准率的提升。

第五章，主要介绍了航天情报信息管理系统中的数据清理模块，该模块是IMPN算法在该系统中的应用，主要内容包括数据清理模块的设计、重复记录产生的原因、IMPN算法在系统中的应用方式以及该算法对数据质量的提高等。

第六章，总结了本文的内容以及相似重复记录检测算法研究过程中遇到的问题，并对未来的研究方向进行了展望。

# 第二章 重复记录检测相关算法概述

## 2.1 相似重复记录概述

相似重复记录是指，数据库中存在这样的两条记录、，它们的内容相同或者相似，且都对应着同一个现实实体，则记录对互为相似重复记录[[[23]](#endnote-24)]。实际数据库中可能存在多对互为相似重复的记录，它们的存在降低了数据的质量，可能会妨碍系统的正常运行，甚至会影响企业信息管理系统的决策正确性。

表2.1给出了学生信息表中的相似重复记录示例：

表2.1 学生信息表中的重复记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stu\_ID | Name | Gender | Brithday Date | School |
| 1801001 | Sam Water | M | 1993/01/02 | School of Computer Science, Xi’an University of Electronic Science and Technology |
| 1802002 | Jack Panda | Female | 1990/07/20 | School of Artificial Intelligence, Xi’an University of Electronic Science and Technology |
| 1801003 | S. Water | Male | 1993/1/2 | Schol of Computer Science, Xi’an University of Electronic Science and Technology |
| 1802004 | Jack Panda | Female | 1990/07/20 | School of Artificial Intelligence, Xi’an University of Electronic Science and Technology |
| 1801005 | Mr.Sam W | Male | 1993-01-02 | College of Computer Science, Xi’an University of Electronic Science and Technology |

表2.1展示了5条学生记录，其中Stu\_ID为1802002和1802004的两条记录的所有字段内容完全一致，说明这两条记录对应现实世界中的同一个学生的信息，所以它们互为相似重复记录。表中Stu\_ID为1802001、1802003、1802005的三条记录表面上内容是不一样的，它们的区别在于：Name字段值分别为“Sam Water”、“S. Water”、“Mr.Sam W”，是由“Sam Water”采用了不同的书写方式而产生的；Gender字段值“Male”、“M”则是全称和缩写的区别，均代指男性；Brithday Date字段则是使用的不同的时间格式，但它们都是代表相同的一天“1993/01/02”；所属学院字段中，出现了“Schol”这样的拼写错误；经过以上观察分析可以发现，这三条内容相似的记录同样对应同一个学生。

相似重复记录产生的原因多种多样，包括人工操作过程中的录入错误或者管理错误造成的重复、不同来源的数据集进行合并时产生的重复、信息系统重构时新旧版本的数据库合并造成的重复等。

相似重复记录检测目前应用最广泛的手段是基于“排序/合并”的方法：首先对包含重复记录的数据集进行排序，排序使用的关键字按照某种固定的方式（如某字段的前三个辅音字母等）从记录的相应字段中提取，排序之后相似重复的记录汇聚在相邻的位置，然后通过对相邻位置的记录进行对比判等，可以检测出相似重复记录。

## 2.2 相似度匹配算法

相似重复记录检测过程中需要对不同的记录对进行整体相似性判断，这就需要用到字段相似度匹配算法。数据库中的每条记录均由不同的字段组成，相似重复记录各字段内容也相似重复，通过计算不同记录字段之间的相似度可以判定记录是否互为重复。目前该领域的算法主要有两大类：基于单字段的匹配算法和基于多字段的匹配算法。

### 2.2.1 基于单字段相似度匹配

基于单字段的相似度匹配算法在相似重复记录检测中的应用过程是，通过计算两条记录相同字段对应内容的相似度来衡量记录整体的相似程度，这是一个从部分到整体的过程。常用的算法包括：编辑距离算法、Smith-Waterman算法、Jaro算法等。

编辑距离算法是Levenshtein于1965年提出的一种基于字符的相似度匹配算法，又名L-距离算法[[[24]](#endnote-25)]。两个字符串和的编辑距离是指：变成需要对其单个字符进行插入、替换、删除操作的次数。编辑距离越小代表和越相似。



图2.1 编辑距离示意图

如图2.1所示，字符串“change”经过3次插入操作和一次删除操作可以变成字符串“challenge”，所以这两个字段的编辑距离为4。计算两个字符串间的编辑距离的经典解法是使用动态规划方法。L-距离算法在应对字母书写错误、缩写等场景下效果较好。

Smith-Waterman算法[[[25]](#endnote-26)]最早是在生物学序列比对领域被提出的，用于匹配遗传序列。S-W算法也是一种动态规划算法，它是Needleman-Wunsch算法的一个变种，主要思路是通过罚分和空位计算不同字段内容的相似度。S-W算法可以有效应对包含不正确值的相似重复记录，但处理字符串缩写、字母颠倒情况的能力较差。

Jaro算法[[[26]](#endnote-27)]由Jaro在1976年提出的基于字符串公共子集的相似度匹配算法。Jaro距离用来衡量两个字符串的相似度，对于给定的字符串和，两者的Jaro距离如下公式(2-1)所示：



其中，代表匹配的字符个数，代表换位的数目。Winkler提出的Jaro-Winkler相似度匹配算法在Jaro算法的基础上，赋予相同的字符串更高的分数，减小了原算法对于字符距离限制的影响，提高了算法在面对较分散的长字符串时的检测准度。Jaro-Winkler距离的计算如公式(2-2)所示：



其中，是Jaro距离，是前缀的匹配长度，是一个常数，作用是可以调整前缀匹配的权值，0.25。

### 2.2.2 基于多字段相似度匹配

基于多字段的相似度匹配算法的思想是将一条记录视为一个整体，通过计算两条记录整体上的相似度判断是否互为相似重复记录。常用的算法包括余弦相似度匹配算法、基于监督训练的机器学习方法等。

余弦相似度[[[27]](#endnote-28)]是一种基于TF-IDF加权算法的多字段相似度匹配方法。算法的步骤如算法2.1所示：

|  |
| --- |
| **算法2.1：**余弦相似度匹配算法 |
| 1.将需要匹配的字段内容进行分词，得到互相独立的单词；  2.对每个单词分配权重，，其中单词出现的次数（词频）用表示，表示记录总数除以包含的记录个数（逆文档频率）；  3.将待匹配的字段转化成向量和；  4.计算向量的余弦相似度：；  5.余弦结果越接近1证明记录间相似度越高，将结果与阈值进行比较判断记录是否相似。 |

除此之外，机器学习领域中的分类技术可以用来检测判断重复记录[[[28]](#endnote-29)]。使用基于单个字段的相似度匹配算法进行相似重复记录检测时，不同的字段拥有不同的判等权重，对记录相似与否的影响程度也不同，字段之间相似到记录整体的相似度关系是非线性的，这是一种适合使用基于训练样本的有监督学习的场景。通过确定数据集（可以明确不同记录之间相似与否）对神经网络进行训练，然后采用训练好的网络对由记录对生成的输入向量进行计算，网络输出的结果若大于阈值则判定两条记录是重复记录，否则判定为不重复记录。

## 2.3 相似重复记录检测算法

相似重复记录检测领域最直接的方法是对数据集中的数据进行一对一地判等，这种做法简单，查重效果好，但是时间复杂度为，当数据量比较大时效率过低。“排序/归并”是目前相似重复记录检测算法所采用的主要思想，按照预先设定或者按照算法计算出的排序关键字对数据集进行排序，将相似记录汇聚到邻近位置，然后使用相似度匹配算法进行重复检测。常见的算法有近邻排序算法（Sorted-Nighborhood Method，记为SNM）、多趟近邻排序算法（Multi-Pass Sorted-Neighborhood，记为MPN）、优先队列算法（Priority Queue Strategy，记为PQS）、N-Gram算法等。

### 2.3.1 近邻排序算法

SNM算法的设计思路是：首先，根据数据领域的专家知识经验指定数据集排序所使用的关键字的生成方式，然后，遍历数据集对每一条记录生成排序关键字并附加到记录后，之后，对数据集按照对应关键字字段进行排序，根据相似记录对应的关键字内容也相似的原理，不同的重复记录在排序完成后理论上会处于邻近的位置，最后采用固定大小的滑动窗口方式对数据集进行重复检测。SNM算法步骤如下：

|  |
| --- |
| **算法2.2：**SNM算法 |
| 1.确认排序关键字的生成方案；  2.对每条记录生成排序关键字；  3.按照对数据集的记录进行排序（只考虑内部排序）；  4.确定滑动窗口的大小，每次比较时将新进入窗口的记录与窗口内剩余的条记录进行相似性判断。 |

SNM算法滑动窗口的过程如图2.3所示：



图2.2 SNM算法滑动窗口过程示意图

若数据集大小为，使用SNM算法生成排序关键字过程的时间复杂度为，排序过程的时间复杂度为，滑动窗口归并过程的时间复杂度为，其中为窗口的固定大小。SNM算法的优点在于使用简单，滑动窗口判重过程效率较高，运行速度较快。但它也存在比较明显的缺点：

（1）过于依赖生成的排序关键字。选择不当的关键字生成方案可能导致相似重复记录相距较远，不相似的记录却处于邻近位置，这就导致算法的检测效果大打折扣。

（2）滑动窗口的大小较难选择。若太大，虽然检测效果可能提高，但是会导致算法的运行时间增大；若太小，很可能导致相似重复记录无法被窗口覆盖到，导致算法查全率下降。

### 2.3.2 多趟近邻排序算法

多趟近邻排序算法（Multi-Pass Sorted Neighbothood，记为MPN）是在SNM算法的基础上提出来的一种改进算法。该算法的改进点在于：

（1）对数据集互不干扰地执行多趟近邻排序算法，每次采用不同的排序关键字生成方案，并且滑动窗口的大小相对于传统的SNM算法可以更小。

（2）对执行完多趟SNM算法的结果求传递闭包。

传递闭包的定义[[[29]](#endnote-30)]是：设是定义在集合上的关系，代表传递性，满足下列所有条件的关系称为的传递闭包：

（1）；

（2）满足性质；

（3）如果存在集合上的关系，满足性质并且，则。

传递闭包的计算多采用Warshall算法，伪代码如下：

|  |
| --- |
| **算法2.3：**Warshall算法 |
| 1.*W* := *MR*  2.FOR *k*:= 1 to *n*  FOR *i* := 1 to *n*  FOR *j* := 1 to *n*    3.Output *W*； |

MPN算法对多趟检测结果可以计算传递闭包的理论基础是相等的传递性：若记录和是相似重复记录，记录和是相似重复记录，则和也互为相似重复记录。

MPN算法的步骤如下所示：

|  |
| --- |
| **算法2.4：**MPN算法 |
| 1.确认个排序关键字的生成方案，独立重复地执行步骤2~4次；  2.对数据集生成排序关键字；  3.按照对数据集的记录进行排序（只考虑内部排序）；  4.确定滑动窗口的大小，每次比较时将新进入窗口的记录与窗口内剩余的条记录进行相似性判断得到重复记录集合；  5.对个重复记录集合求传递闭包得到最终的重复记录集合。 |

含有两趟SNM过程的MPN算法的流程图如图2.4所示：



图2.3 MPN算法流程图

正是由于MPN算法引入了传递闭包的计算，使得一些容易被遗漏的重复记录被检测了出来，提高了算法的查全率，同时每轮SNM过程的滑动窗口也可以变得更小，缩短了滑动归并的执行时间。但是MPN算法也存在缺点：依旧没有克服SNM算法对于排序关键字的依赖性，计算传递闭包容易导致算法的误识别率上升。

### 2.3.3 其它算法

N-Gram算法是一种基于聚类思想的算法[[[30]](#endnote-31)]。N-Gram值由记录中每个单词出现的概率综合计算得出，相似重复记录的N-Gram值在数值上也相似，算法的优势在于对常见拼写错误表现较好。韩京宇等人提出了一种基于N-Gram层次空间的聚类算法DGHS[[[31]](#endnote-32)]，记录、的N-Gram相似性计算方法如公式(2-3)所示：



其中表示记录的所有字段组成的集合。将记录整体作为字符串，由N-Gram映射相应的子空间，采用层次聚类进行归并，实现对相似重复记录的检测。

优先级队列算法进行相似重复记录检测的思想是：用含有不同重复记录簇的优先级队列来替换传统SNM算法中的滑动窗口，算法在扫描过程中，若队列中不含有当前记录则赋予其最高的优先级然后加入到队列中，如果含有该记录，则将相应的重复记录簇的优先级设为最大。采用多趟优先级队列算法进行相似重复记录检测的过程如图2.4所示：



图2.4多趟优先队列扫描算法过程示意图

## 2.4 BP神经网络理论基础

### 2.4.1 神经元模型

人工神经网络简单模拟大脑处理信息的机制，它是由许多互相连接并传递信息的神经元组成的非线性处理系统[[[32]](#endnote-33)]，每个组成单元的结构功能并不复杂，整体却能以任意精度逼近线性或者非线性函数，从而可以表征真实社会中更加复杂的问题。神经网络中的一个神经元所起到的作用是接收来自其他神经元的加权输入，然后结合自身的阈值(偏置)，最后经过非线性函数的处理，得到输出结果[[[33]](#endnote-34)]。典型的神经元模型如图2.5所示：



图2.5 神经元模型

其中，~是输入向量的不同分量，~是神经元各个突触的权值，表示偏置，是加权求和操作，则是激活函数。则对于输入向量经过此神经元时，经过加权求和以及激活函数得到的输出为：。许多的类似于这样的神经元则组成了人工神经网络。

### 2.4.2 梯度下降法

梯度下降法[[[34]](#endnote-35)]是一种经典的最优化方法，其主要思想是不断沿着负梯度方向进行搜索。给定目标函数如公式(2-4)所示，是要学习的参数，是第个输入特征向量的第个分量，，表示偏置，共有维特征。



采用均方误差损失函数，如公式(2-5)所示，共有个训练样本，表示第个训练样本的真实类标向量，表示第个训练样本的预测类标向量，当损失函数的值最小时，说明所训练出的模型参数最能拟合训练样本，因此求解参数的过程就是最小化损失函数。



首先求出损失函数对参数的导数，如公式(2-6)所示。然后，根据损失函数对参数的负梯度方向更新参数，如公式(2-6)所示。是梯度下降法的学习速率，一般情况下，随着学习次数的增加，参数逐渐减小，即参数在学习过程中的变化越来越小。





由公式(2-7)可以看出，对每一个参数，都需要使用全部样本来学习该参数的变化量，将这种梯度下降法的实现方式称作“批梯度下降法”。在实际操作中，由于样本个数较大，所以这种参数更新方法会使导致训练过程缓慢，难以应用于实际问题。

为了克服批梯度下降法的缺点而出现了随机梯度下降法和小批量梯度下降法，这两种方法使用样本全集中的一个或部分样本来更新参数，这样操作使得每一次并不是按照严格意义上的最优方向来更新参数，但是从整体来看，依旧是朝着负梯度的方向更新参数，这两种梯度更新方式使参数学习的速度大大提高，适用于大规模训练样本的情况。

### 2.4.3 BP网络前向传播和反向传播



图2.6 典型三层BP神经网络

一个典型的三层BP网络如图2.6所示，第一层是输入层，共有个输入神经元；第二层是隐层，共有个隐层神经元，第三层是输出层，共有个输出神经元。表示第个输入神经元与第个隐层神经元的连接权值，表示第个隐层神经元的偏置，表示第个隐层神经元与第个输出神经元的连接权值，表示第个输出神经元的偏置，表示第个输入神经元的输入特征值，表示第个隐层神经元的输出，表示第个输出神经元的输出，表示第个隐层神经元的输入，表示第个输出神经元的输入。

BP网络的前向传播是指将上一层的输出与层间的对应权值相乘并求和，最终经非线性函数的处理，得到下一层对应神经元的输出值。常用的非线性函数有sigmoid函数(如公式(2-8)所示)、tant函数(如公式(2-9)所示)、ReLU函数(如公式2-10所示)等。









如图2.6所示，前向传播过程如公式(2-11)和公式(2-12)所示，设使用sigmoid函数。





对于第个训练样本，，，设神经网络对于该样本的输出为，使用均方误差损失函数，则对样本的损失函数如公式(2-13)所示，对整个样本的损失函数如公式(2-14)所示，其中第一项表示在所有样本上的平均损失，第二项表示正则项，表示所有参数组成的向量，正则项可以避免过拟合。Sigmoid函数对参数求导结果如公式(2-15)所示。BP网络反向传播的目的是求得使损失函数最小时的参数，使用梯度下降法进行求解。







根据链式求导法则，损失函数对参数和的导数分别如公式(2-16)和公式(2-17)所示。





在样本上，参数的更新如公式(2-18)所示。



BP误差反向传播算法的基本过程为：

**输入：**训练集，学习率.

**输出：**连接权值与偏置值确定的多层前馈神经网络.

**过程：**

产生(0,1)范围内的随机数，初始化所有连接权值和偏置值；

Repeat

For all ：

根据当前参数执行前馈传播，计算当前样本的输出值；

根据公式(2.17)进行反向传播，更新当前参数；

End for

Until 达到终止条件

## 2.5 本章小结

本章首先介绍了相似重复记录的概念以及产生的原因，然后介绍了常用的基于单字段和多字段的相似度匹配算法。

然后重点介绍了几种不同的相似重复记录检测算法。从算法的设计原理，实现步骤，优缺点等对SNM算法、MPN算法等进行了介绍说明。

最后介绍了神经网络中的神经元模型、梯度下降法、BP神经网络的正向传播和反向传播的过程。

# 第三章 改进的IMPN算法

多趟近邻排序算法（Multi-Pass Sorted Neighbothood，MPN）在传统的近邻排序算法（Sorted Neighbothood Method,，SNM）的基础上，通过计算每趟近邻排序算法重复检测结果的传递闭包，实现了更好的检测效果。它能够以更小的滑动窗口进行重复检测，并且通过计算传递闭包可以检测到一些人工难以发现的重复记录。

但是MPN并没有克服SNM算法在选取排序关键字时过于依赖专家经验的缺陷，同时，如果某一条记录的排序关键字字段为缺失值时，MPN算法的检测效果很差。基于MPN算法的不足，本文在MPN算法基础上提出了一种改进的多趟近邻排序算法（Improved Multi-Pass Sorted Neighbothood，IMPN）。IMPN对MPN的改进在于三个方面，首先，IMPN采用基于统计规律的提取关键字的关键字选取方法；其次，IMPN采用动态可伸缩的自适应滑动窗口；最后，IMPN增加了对排序关键字有缺失的记录的特殊处理。

## 3.1 基于字段区分度提取关键字的方法

### 3.1.1 传统的提取关键字的方法

基于“排序、合并”思想的重复记录检测算法，都需要依据预先提取排序关键字，然后依据提取的排序关键字对记录进行排序[[[35]](#endnote-36)]。在完全理想的状态下，重复记录会具有相同的关键字，所以，这些重复记录排序后会汇聚到邻近的位置，从而可以缩小检测范围，在小邻域内进行重复记录的检测。

在相似重复记录检测问题上，排序关键字是指，从记录中提取出来的不同属性组成的序列或者属性的字符串子集[[[36]](#endnote-37)]。对于传统的SNM算法和MPN算法，关键字的选取十分关键，数据集中的记录需要根据关键字进行排序，只有选择了恰当的关键字，才能保证排序后的重复记录聚集在相邻的位置，进而保证重复记录能够被一个滑动窗口所覆盖，从而准确地检测到重复记录。一方面，准确的排序关键字还可以保证较小的滑动窗口就具有较大的检测准确度，另一方面，准确的排序关键字会使得不同的记录经排序后处于较远的位置，从而可以避免无意义的判等比较，节约检测时间。综合来看，排序关键字的选择不仅影响了算法的整体重复检测效果，还影响了算法的时间运行效率。

排序关键字的选取方式不是唯一的，传统的SNM算法和MPN算法需要根据专家经验，人为地为待处理的数据集提取准确的排序关键字，这也是这类算法的一个缺点。所以，针对特定的数据集，准确地选取合适的排序关键字，才能达到准确检测重复记录的目的。

表3.1 四条相似重复记录的例子

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Record | First Name | Last Name | Address | ID |
|  | Jack | Stolfo | 123 First Street | 12345678 |
|  | Jack | Stolfo | 123 First Street | 12345673 |
|  | Jack | Stolpho | 123 First Street | 12345678 |
|  | Jacon | Stiles | 123 Forest Street | 12345432 |

表3.1列举了四条记录、、和，它们由四个字段组成，分别是First Name、Last Name、Address和ID，其中，、、互为重复记录，与其他记录不同。记录和的First Name、Last Name和Address字段都完全一致，ID字段因人为录入或印刷错误而不同，但是他们是重复字段。和的Last Name字段不一致，这也是由于人为错误导致的。而和的Last Name字段完全不一致，而且两者的Address字段的取值也相差甚远，它们对应着现实世界的两个不同的实体。相似重复检测算法就是要从已有的脏数据集中，准确地检测到重复数据，例如，对于表3.1所示的4条记录，就需要算法准确得到的结果。

对于表3.1所示的数据，依据传统的关键字选取方法，因为主要依赖专家经验，容易因人而异，本文列举三种不同的排序关键字选取方式，并详细阐述每种选取方法所得到的排序关键字，并分析不同的排序关键字对检测结果的影响。

**方式1、**排序关键字的构成包含以下几个部分：

（1）Last Name的所有部分

（2）Address的数字部分加字母部分的前三个辅音字母

（3）ID的前三个连续数字

**方式2、**排序关键字的构成包含以下几个部分：

（1）Last Name的前三个辅音字母

（2）First Name的前三个连续字母

（3）Address的数字部分加字母部分的前三个辅音字母

（4）ID的前三个连续数字

**方式3、**排序关键字的构成包含以下几个部分：

（1）Last Name的前三个辅音字母

（2）First Name的前三个辅音字母

（3）Address的数字部分加字母部分的前三个辅音字母

（4）ID的前三个连续数字

由方式1、方式2和方式3得到的针对表3.1所示数据的排序关键字分别如表3.2-表3.4所示。

表3.2 相似重复记录及其生成的关键字

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Record | First Name | Last Name | Address | ID | Key |
|  | Jack | Stolfo | 123 First Street | 12345678 | STOLFO123FRT123 |
|  | Jack | Stolfo | 123 First Street | 12345673 | STOLFO123FRT123 |
|  | Jack | Stolpho | 123 First Street | 12345678 | STOLPHO123FRT123 |
|  | Jacon | Stiles | 123 Forest Street | 12345432 | STILES123FRT123 |

表3.3 相似重复记录及其生成的关键字

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Record | First Name | Last Name | Address | ID | Key |
|  | Jack | Stolfo | 123 First Street | 12345678 | STLJAC123FRT213 |
|  | Jack | Stolfo | 123 First Street | 12345673 | STLJAC123FRT123 |
|  | Jack | Stolpho | 123 First Street | 12345678 | STLJAC123FRT123 |
|  | Jacon | Stiles | 123 Forest Street | 12345432 | STLJAC123FRT123 |

表3.4 相似重复记录及其生成的关键字

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Record | First Name | Last Name | Address | ID | Key |
|  | Jack | Stolfo | 123 First Street | 12345678 | STLJCK123FRT213 |
|  | Jack | Stolfo | 123 First Street | 12345673 | STLJCK123FRT123 |
|  | Jack | Stolpho | 123 First Street | 12345678 | STLJCK123FRT123 |
|  | Jacon | Stiles | 123 Forest Street | 12345432 | STLJCN123FRT123 |

由表3.2可以看出，记录和对应的关键字相同，均为“STOLFO123FRT123”，记录和对应的关键字不同，经过排序后和会聚集在邻近位置，而和则不会处于邻近的位置。对于记录和，它们原本是相同的记录，但是依据这种方式对它们所选取的排序关键字是不同的，所以将导致无法准确地检测出记录和是一对重复记录。

由表3.3可以看出，按照方式2对4条记录所选取的排序关键字均为“STLJAC123FRT213”，然而从表3.1和表3.3可以看出：和具有相同排序关键字是合理的；和对应的现实实体是一致的，选择辅音字母可以一定程度上克服发音上的错误，和的排序关键字的相同也是合理的；然而，和则是对应着现实世界的两个不同的实体，由于“Stolfo”和“Stiles”的连续的三个辅音字母都是“STL”，“Forest”和“First”的前三个连续的辅音字母都是“FRT”，并且关键字段中所选取的ID也是相同的，所以导致和具有相同的排序关键字，经排序后，和将会与、、处于近邻的位置，从而会导致额外的判等计算，甚至可能导致错误地将与、、识别为重复记录，从而影响算法的准确性。

由表3.4可以看出，、、的排序关键字相同，它们对应的现实实体也是一致的，和则对应不同的排序关键字，这说明所选取的排序关键字可以有效地将重复数据聚集，将不同数据隔离，因此，这是针对该数据集的理想的排序关键字选取方式。

### 3.1.2 改进的字段区分度方法

衡量排序关键字优劣的一条重要的原则是，不同的记录对应的关键字不同，相同的记录对应的关键字理应相似或一致。对于真实重复的记录，无论排序关键字以何种方式选取，理论上会得到相同或近似相同的排序关键字值，所以衡量排序关键字最关键的标准是它们在区分不同记录时的表现，即对应着现实世界中不同实体的两条记录在合适的排序关键字方案下生成的排序关键字值应当不一致。所以在选择排序关键字时，应当选择有足够区分度的字段来提取关键字。为定量地衡量关键字的区分不同记录的能力，本文提出了“字段区分度”的概念。

字段区分度：字段区分数据库中不同记录的能力，某一字段取不同值的记录个数越多，该字段的字段区分度越大。

设数据库中一共有条记录，每条记录都个字段组成，即，对于第个字段，它的区分度计算公式如公式(3-1)所示：



其中，代表数据集中在字段一共有种取值，即如果将数据集按照字段的不同取值进行聚类，一共有簇。的取值介于0到1之间，值越高，说明对应的字段对于整体数据集的区分能力越大。

通过计算字段区分度，可以有效地衡量不同字段对整体数据集的区分能力，这是因为，相似重复记录在每个字段的内容理论上是相等的，由于印刷错误、格式不一致、人工录入等导致部分相似重复记录在某个字段表现不一致的情况在数据集中只占有较小的比例，因此，无论数据集中的重复数据所占的比例是多少，对数据集具有较大区分能力的字段的“字段区分度”值也较大。基于字段区分度的排序关键字选取方法的主要操作过程如算法3.1：

|  |
| --- |
| **算法3.1：**基于字段区分度的排序关键字选取算法 |
| 1.读取数据集，得到待检测的数据；  2.计算数据集字段的字段区分度并排序；  3.确定排序关键字由个组成部分，以及每个部分的生成方案；  4.对数据集中的每条记录优先选取区分度较大的字段，按照生成有个组成部分的排序关键字； |

本文提出的基于字段区分度的关键字选择方法，依靠已知数据集的统计特性进行关键字选择，克服了MPN算法依赖专家经验进行人工选择排序关键字的缺陷，同时，对于未知类型的数据集，人工选择关键字往往不能准确把握该类数据集的特征，人为因素对算法效果的干扰较大，而基于字段区分度的方法从统计角度出发，更能挖掘数据本身的特性和规律，人为因素影响较小。因此，在没有人工参与或者较少人工参与选取排序关键字的情况下，基于字段区分度选择排序关键字的方法更有利于在排序后将对应着不同实体的记录区分开，将对应着相同实体的重复记录聚集到相邻位置，从而提高了算法的普适性。

“排序/归并”类算法在滑动窗口判重过程中，需要对邻近位置的记录进行相似判断，传统的MPN算法采用的是基于专家经验知识的规则产生式系统（OPS5），这种做法依赖人工，效率较低，适用范围较为局限。在本节提出的字段区分度概念的基础上，提出了一种基于字段区分度的加权方式，在判等过程中，为区分度高的字段分配更大的权值，字段相似度匹配采用编辑距离算法。对两条记录、（均包含个字段）进行相似性判断的步骤如下：

|  |
| --- |
| **算法3.2：**基于字段区分度的加权判等算法 |
| 1.计算记录和所有字段的编辑距离；  2.由字段区分度从小到大的顺序对字段编辑距离进行排序；  3.排序完成，将所有的编辑距离值组合得到相似度向量；  4.对每个字段分配权值，保证字段区分度大的字段，其权值也高；  5.计算两条记录的相似度：；  6.将和阈值进行比较，若，则两条记录为相似重复记录。否则判定记录不重复。 |

## 3.2 自适应大小的滑动窗口检测方法

MPN算法和SNM算法采用相同的滑动窗口选择方法。算法在操作过程中所采用的滑动窗口的大小都是一个固定值，记窗口大小为，本文在此基础上，提出了自适应大小的滑动窗口方法。

### 3.2.1 传统的滑动窗口检测方法

给定大小固定为的滑动窗口，首先，将数据集中的所有记录合并成一个线性序列，然后，将已知滑动窗口从第一条记录开始，每次向下滑动一条记录，每当滑动到一条新纪录时，移除原来窗口中的第一个记录，始终维持窗口内有条记录，直到数据集中的最后一条记录进入窗口内。

基于滑动窗口进行重复记录检测的主要操作过程为：初始时，取得滑动窗口内的所有条记录，将这条记录两两之间进行判断，检测任意两个记录是否是重复记录；然后，滑动窗口每次滑动所访问到的最新的一条记录分别与之前的条记录进行比较，判断与这条记录中的任意一条是否相同。

传统的滑动窗口扫描过程如图3.1所示。



图3.1 滑动窗口扫描过程

基于固定大小的滑动窗口检测方法需要提前设定窗口大小，所以人为因素干扰较大，其次，检测结果对窗口大小的依赖性较大，同时，对于大小为的滑动窗口，这种检测过程的时间复杂度为，所以，如果滑动窗口过大，则会增加算法的运行时间，如果窗口过小，则会降低算法检测的精度。

图3.2给出了不同窗口大小时的检测结果示意图。在图3.2中，record0\_origin和record2\_origin是两条不重复的原始记录，record0\_dup0、record0\_dup1、record0\_dup2是与record0\_origin重复的原始记录，分别用标号(1)-(5)简单标记这些记录。

图3.2中分别展示了两种大小的滑动窗口，一种窗口大小为2，一种窗口大小为5。若滑动窗口的大小为2，按照传统的滑动窗口归并方法，无法检测到记录(4)与记录(1)、记录(2)是重复记录；若滑动窗口的大小为，虽然能够实现最好的检测效果，但是检测时间是第一种检测时间的2.5倍。

通过观察可以得到，记录(1)、记录(2)相邻且是重复记录，所以应当采用较大的滑动窗口，以对尽可能多的记录进行检测，而当窗口移动到记录(3)、(4)、(5)位置的时候，这些记录互相不重复，较大的滑动窗口只会增加检测时间，所以适合采用较小的滑动窗口进行检测，既不会降低检测精度，又可以节约时间。通过以上分析可以得出，SNM和MPN所采用的固定大小的滑动窗口方式仍有较大的优化空间，一个较为合理的滑动窗口检测方案应当可以根据数据内容动态变化滑动窗口的大小，以减小算法复杂度。



图3.2 不同滑动窗口大小时的归并过程

### 3.2.2 改进的自适应大小的滑动窗口

为克服传统的固定大小的滑动窗口的不足，本文提出了自适应大小的滑动窗口检测方法。

滑动窗口的大小可以根据当前滑动窗口的数据重复情况而做出动态地调整，当窗口内的数据重复度比较高时，证明当前窗口正处于重复记录比较集中的位置附近，而且重复记录的数量可能更多，所以为了实现更精准的检测，应当扩大窗口尺寸以使得窗口包含更多的记录，对更多的记录进行检测；反之，如果滑动窗口内的数据集重复度比较低，即相似重复数据较少，则说明当前位置附近的数据之间可能互相不重复，会存在冗余的检测过程，所以如果窗口较大，就会造成这些互不重复的数据需要不断与新进入窗口的数据进行重复性检测之后才能退出滑动窗口，导致运行时间的增加，所以这时应当减小滑动窗口的大小。

现有的自适应窗口算法[[[37]](#endnote-38)]大多是统计出当前滑动窗口内的所有的重复记录数，依据重复数据的个数占滑动窗口的大小来决定当前窗口的尺寸变化，此类方式存在很大的弊端：原本MPN的归并过程的时间复杂度只有，其中是滑动窗口的大小，是待检测的相似重复记录集合中记录的条数，但是这种方法每次要对窗口内的记录进行一对一的重复检测，使得时间复杂度升高到。

考虑到MPN原本在滑动窗口内的比较是将新进入的一条记录与窗口内仍保留的-1条记录进行比较，当重复度较高的时候，下一次比较应当保证的是新进入的记录应该能够尽可能的照顾到即将要离开的记录，才能保证检测效果。也就是说，在当前滑动窗口内，即将离开滑动窗口的记录若和新进入的元素是相似重复记录，这时应当扩大窗口尺寸。设窗口内的重复记录在窗口内的位置依次为0、1、2...-1，其中-1是刚滑入的数据，0号位置的记录是即将被滑出的记录，则越是靠近0号位置的数据对于滑动窗口的尺寸影响越大。本文提出了动态计算滑动窗口大小的计算公式，如公式(3-2)所示：



公式(3-2)中，常数表示滑动窗口的大小可以取的最大值，常数表示滑动窗口的大小可以取的最小值，表示当前滑动窗口的大小，表示即将滑出滑动窗口的记录在数据集中的索引位置，代表数据集中索引为的记录是否与位置的记录互为重复，若它们重复，则=1，否则=0。可以看出，若内的记录都是重复记录，则滑动窗口大小更新为最大值，相反，若内的记录互不重复，则滑动窗口大小更新为最小值；并且距离越远位置的记录对下一个滑动窗口大小的影响越大（当其与位置的记录互为重复记录时）。

图3.3 自适应大小的滑动窗口

图3.3展示了自适应大小的滑动窗口检测过程，在图3.3中，R0、R0\_0、R0\_1、R0\_2、R0\_3、R0\_4互为相似重复记录，记录R0、R1、R2、R3互不重复。滑动窗口内的归并过程从左向右进行。设定最大窗口为5，最小窗口为3，①表示初始时，滑动窗口大小为4；由公式(3-2)计算得到第②步所采用的滑动窗口大小变为5，因为此时窗口内的记录和R0\_2全部重复，所以第②步的窗口大小被扩大；然后由公式(3-2)计算得到第③步窗口大小取最小值3，因为此时滑动窗口内的数据和R1互不重复，所以窗口大小被缩小；同理得到第④⑤⑥的滑窗大小。图3.3简明生动地表现了自适应滑动窗口的变化过程，展现了IMPN算法所使用的自适应大小的滑动窗口检测方法的优势。

## 3.3 基于预标记处理排序关键字不完整的方法

### 3.3.1 MPN排序方法的缺陷

由于待清洗的数据集本身的数据质量并不高，所以记录中可能存在字段为空或者字段不完整的情况，表3.5给出了一种数据缺失情况的示例。

表3.5 缺失数据及不完整数据示例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Record | First Name | Last Name | Address | ID |
|  | Jack | tolfo | 123 First Street | 12345678 |
|  | Jack |  | 123 First Street | 12345673 |
|  | Jack | Stolpho | 123 First Street | 12345678 |
|  | Jacon | Stiles | 123 Forest Street | 12345432 |

在表3.5中，记录R1的Last Name字段原本应该是“Stolfo”，出现字段不完整而变成“Sto”， 在生成排序关键字时若采用3.1.1中的第3种方式，即提取Last Name的前三个辅音字母是则只能得到“s”和“t”两个字母，则排序关键字位数少了一位，所以生成的关键字为“TLFJCK123FRT213”。记录R2的Last Name字段则直接完全缺失，所以提取的关键字为“JCK123FRT213”，在排序的过程中，本来属于重复记录的R1和R2由于排序关键字的首字母的差异而无法聚集在近邻的位置，使得被检测为互为重复记录的概率减小。

传统的MPN算法在处理这种带有不完整数据和缺失数据的数据集时，就会遇到这种问题，从而使检测精度降低。为了克服这一缺点，本文提出了针对排序关键字不完整的改进方法，详细介绍在3.3.2节。

### 3.3.2 改进的基于预标记的方法

缺失数据的处理是数据清洗的另一个分支研究领域，面对缺失值常见的做法主要有三种[[[38]](#endnote-39)]：

（1）以同一指标的计算结果（包括均值、中位数、众数等）填充缺失值

（2）以业务知识或者经验填充缺失值

（3）从本数据集或者其他来源的数据集推测出来

其中第（1）种做法填充结果不够精细甚至过于粗糙，对检测结果可能造成负面影响；第（2）种做法填充结果可能较为准确但是需要人工干预，工作量较大；第（3）种做法对数据集的数据质量要求较高并且能达到的效果下限很低。

针对相似重复数据检测问题，考虑到不完整数据和缺失数据会造成记录的排序关键字缺失或不完整，进而会对记录排序后的位置产生影响，所以本文提出了基于标记的处理数据缺失的方法，该方法的主要操作过程为：

（1）对所有关键字不完整的记录的ID进行标记；

（2）从数据全集中去除第（1）步所标记的数据，只对排序关键字完整的记录进行排序和归并；

（3）处理被标记的带有缺失值的记录，分别对这些记录进行检测，将其一一聚类到第（2）步得到的重复数据簇中。

本文这种基于预标记处理缺失值做法能弥补MPN算法在排序关键字缺失的情况检测效果差的缺点，同时，对于对含有缺失字段的记录占数据集比例较低的数据集合进行操作时，时间耗费在合理的范围内，对于真实数据集，缺失值所占比例较小，所以该方法是可行的。

## 3.4 IMPN算法设计

### 3.4.1 算法流程设计

结合3.1~3.3的内容可以看出，IMPN算法的改进思想在于以下三点：

（1）基于字段区分度选取排序关键字，避免了对专家经验的依赖性。

（2）采用可伸缩的滑动窗口检测方法，根据数据特点动态调整检测窗的大小，减少不必要的比较次数。

（3）预标记含有不完整排序关键字的记录，更适用于真实应用场景。

有趟SNM过程的IMPN算法步骤如下：

|  |
| --- |
| **算法3.3：**IMPN算法 |
| 1.读取数据集，得到待检测的数据；  2.计算数据集字段的字段区分度并排序；  3.优先选取区分度较大的字段去生成组排序关键字；  4.独立地执行步骤5~8次；  5.按照排序关键字的产生方式对每条记录提取其排序关键字；  6.对数据集按照关键字排序，如果某条记录的不完整或者为空则将该记录的ID加入到缺失关键字记录集合中，完整则正常排序；  7.进行可伸缩大小的滑动窗口重复检测得到重复集合；  8.将与进行重复归并，然后计算此集合的传递闭包；  9.将次SNM重复检测得到的集合进行归并，然后计算传递闭包得到最终的重复记录集合。 |

含有两趟SNM过程的IMPN算法流程如下图3.4所示：



图3.4 含有2趟SNM过程的IMPN算法流程图

### 3.4.2 时间复杂度分析

IMPN算法是在MPN算法的基础上进行的改进与创新，因此主要对这两种算法的时间复杂度进行分析。在相似重复记录检测算法的处理过程中，理想状态下所有的数据都可以在内存中处理，不考虑磁盘I/O的情况。

MPN算法首先创造排序关键字需要对数据集进行整体遍历，所以该阶段的时间复杂度为；排序过程采用快速排序，算法时间复杂度为；滑动窗口的归并检测过程需要进行次比较，所以该阶段的时间复杂度为；其中代表待检测数据集中的记录总数，代表滑动窗口的大小；在传递闭包的计算过程中，假设重复记录数据集的大小为，则该阶段的时间复杂度为。所以对于MPN算法来说，总的时间复杂度为：



由公式(3-3)可以看出，IMPN算法首先需要对数据集中的所有字段进行区分度统计，假设每条记录的字段总数为，则区分度统计阶段的时间复杂度为；生成排序关键字过程、排序过程、以及滑动窗口归并过程与MPN算法的时间复杂度相同，分别为、、；设因为排序关键字为空而被标注的数据集包含记录数为，则与已检测识别出的数据集进行重复记录检测过程的时间复杂度为；传递闭包过程中的时间复杂度同理为，为检测结束后重复记录集合的大小。所以IMPN算法的时间复杂度为（一般情况下带有不完整排序关键字的记录数满足>k并且>）：



观察公式(3-3)和公式(3-4)组成部分可以发现，两者复杂度在同一数量级，所以两者的时间复杂度在特定的数据集上是一致的。若用代表常数，两个公式都可以简化成下面的式子：



当重复记录较多时，即的值较大，此时两个算法的时间复杂度均为，这体现出了MPN算法和IMPN算法的时间消耗受重复数据比例的影响均较大。

## 3.5 SNM、MPN、IMPN综合对比实验

### 3.5.1 实验数据介绍

为了方便研究使用，本文实验采用的数据集是由第三方的数据生成器“febrl”[[[39]](#endnote-40)]（开源地址：[https://sourceforge.net/projects/febrl/）生成的。“febrl](https://sourceforge.net/projects/febrl/）生成的。)”的数据源是澳大利亚某卫生部门的数据库。生成数据集中记录包含的字段及字段含义如下表所示：

表3.6 记录字段说明

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段描述 | 举例1 | 举例2 |
| rec\_id | 记录ID | rec-454-org | rec-454-dup-0 |
| culture | 文化 | pak | pak |
| sex | 性别 | f | f |
| age | 年龄 | 30 | 30 |
| date\_of\_birth | 出生日期 | 19870221 | 19870221 |
| title | 头衔 | hon | hon |
| given\_name | 名字 | sophie | sophie |
| surname | 姓氏 | bozdar | bozdaa |
| state | 州 |  |  |
| suburb | 郊区 | holsworthy | holsworthy |
| postcode | 邮编 |  |  |
| street\_number | 街道号码 | 46 | 46 |
| address\_1 | 地址1 | thurgood court | thurgood court |
| address\_2 | 地址2 |  |  |
| phone\_number | 电话号码 | 08 42167414 | 08 42167414 |
| soc\_sec\_id | 社保ID | 3920942 |  |

采用数据生成器的好处在于：生成器公开的接口中提供了多个参数，这些参数能够方便用户自定义数据集的大小、重复比例、字段特征、错误类型、重复记录的概率分布等，由该生成器得到的数据集非常接近现实数据；相较于真实数据，生成的数据记录拥有唯一的标识符，更方便后期对算法的查准率、查全率等进行计算和评估。

“febrl”的公开参数列表及其说明如下表所示：

表3.7 febrl公开接口的参数说明

|  |  |
| --- | --- |
| 参数名称 | 参数说明 |
| outputFileName | 输出文件名（.CSV格式） |
| numberOfOriginalRecords | 原始数据集大小 |
| numberOfDupRecords | 由原始数据集生成的重复数据集大小 |
| maxNumOfDupPerRec | 一条原始记录能够最多生成的重复记录个数 |
| maxNumOfModPerRec | 一个字段最多可修改数目 |
| maxNumOfModPerRec | 一条记录最多可修改字段数目 |
| probabilityOfDup | 重复记录在数据集中的概率分布  （均匀分布、泊松分布或齐夫分布） |
| typeOfModification | 字段可能发生的错误类型（typo：印刷错误、ocr：  扫描错误、phonetic：发音错误或者以上所有） |

### 3.5.2 算法的评价指标

衡量相似重复记录检测算法效果的常用标准主要有查全率（Recall），查准率（Precision）以及F-measure。

将算法的重复记录检测结果与实际数据集重复记录进行比较时，会出现以下四种可能的情况：

（1）True Positive（TP）：算法判定为重复记录，实际上也是重复记录；

（2）False Positive（FP）：算法判定为重复记录，但实际不是重复记录；

（3）True Negative（TN）：算法判定为非重复记录，实际也不是重复记录；

（4）False Negative（FN）：算法判定为非重复记录，但实际上却是重复记录。

查全率（Recall）代表了算法检测重复记录是否完备的能力，它的计算方法是算法检测出的正确重复数除以实际数据集中的重复记录总数，如公式(3-6)：



查准率（Precision）代表了算法正确识别重复记录的能力，它的计算方法是算法检测出的正确重复数除以被算法识别为重复记录的总数：



F-measure是查全率和查准率的加权调和平均，F-measure的计算公式如(3-8)：



其中参数取值为1时，F-measure即为最常见的F1-measure：



假设有8条记录、、、、、、、，其中{，，}和{，，}互为相似重复记录，若通过算法检测出的重复结果为{，，，}和{，，，}互为相似重复记录，则TP=4，FP=2，TN=2，FN=0，所以算法的查全率为4/(4+0)=100.00%，查准率为4/(4+2)=66.67%，F1-measure=2\*1\*0.6667/(1+0.6667)=80.00%。

### 3.5.3 实验结果分析

在不同规模的数据集上进行实验，对IMPN算法、MPN算法和SNM算法的查全率和查准率进行综合对比。这三种算法的实验参数为：

IMPN算法的实验参数：最大滑动窗口最大为=20，最小为=3，进行单趟SNM的次数=3；MPN算法的实验参数：固定滑动窗口大小为=5，进行单趟SNM的次数=3；SNM算法的实验参数：固定滑动窗口大小为=5。

在字段相似度检测过程中用到的两个常数VERY\_CLOSE\_CONSTANT（字符串非常接近）、CLOSE\_CONSTANT（字符串比较接近），对它们分别赋值：VERY\_CLOSE\_CONSTANT=0.8，CLOSE\_CONSTANT=0.6。

数据集的规模与生成数据的参数取值如表3.8所示：

表3.8 测试数据集参数取值说明

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 数据集 |  |  |  |  |  |  |  |
| dataset1 | 5000 | 1000 | 3 | 1 | 1 | uniform | phonetic |
| dataset2 | 10000 | 2000 | 3 | 1 | 1 | uniform | phonetic |
| dataset3 | 20000 | 4000 | 3 | 1 | 1 | uniform | phonetic |
| dataset4 | 50000 | 10000 | 3 | 1 | 1 | uniform | phonetic |
| dataset5 | 80000 | 16000 | 3 | 1 | 1 | uniform | phonetic |
| dataset6 | 100000 | 20000 | 3 | 1 | 1 | uniform | phonetic |
| dataset7 | 200000 | 40000 | 3 | 1 | 1 | uniform | phonetic |
| dataset8 | 500000 | 100000 | 3 | 1 | 1 | uniform | phonetic |

上表中的参数~代表的含义分别为：-原始记录数、-重复记录数、-单个记录最多重复数、-单个字段最多修改数、-单个记录最多修改字段数、**-**重复记录的概率分布、错误类型。

本实验的实验环境配置如表3.9所示：

表3.9 实验环境配置

|  |  |
| --- | --- |
| 操作系统 | Windows 7旗舰版 |
| 处理器 | Core i7（8核心） |
| 内存大小 | 8G |
| JDK版本 | JDK1.8.0 |
| JVM配置 | -Xmx4096m（堆内存最大4G） |

在表3.8所示的8种不同规模的据集上，分别使用SNM、MPN和IMPN进行实验，统计每种算法在每种数据集上的查准率、查全率和运行时间，实验结果分别如表3.10至表3.12所示。

表3.10 SNM、MPN、IMPN算法查全率对比表

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集大小（千条） | SNM算法（%） | MPN算法（%） | IMPN算法（%） |
| 5 | 80.60 | 94.80 | 98.80 |
| 10 | 78.15 | 91.80 | 98.40 |
| 20 | 78.03 | 89.28 | 98.75 |
| 50 | 78.01 | 87.39 | 97.90 |
| 80 | 76.38 | 82.08 | 98.58 |
| 100 | 75.85 | 80.76 | 98.73 |
| 200 | 73.83 | 79.72 | 99.10 |
| 500 | 72.90 | 78.91 | 96.91 |

表3.11 SNM、MPN、IMPN算法查准率对比表

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集大小（千条） | SNM算法（%） | MPN算法（%） | IMPN算法（%） |
| 5 | 100.00 | 99.89 | 99.50 |
| 10 | 99.94 | 100.00 | 99.54 |
| 20 | 99.97 | 100.00 | 98.78 |
| 50 | 99.87 | 99.98 | 96.93 |
| 80 | 99.81 | 99.92 | 95.69 |
| 100 | 99.92 | 99.87 | 94.77 |
| 200 | 99.73 | 99.89 | 89.70 |
| 500 | 99.53 | 99.70 | 88.17 |

表3.12 SNM、MPN、IMPN算法运行时间对比表

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集大小（千条） | SNM算法（s） | MPN算法（s） | IMPN算法（s） |
| 5 | 0.503 | 0.938 | 1.274 |
| 10 | 1.196 | 2.320 | 2.961 |
| 20 | 3.540 | 6.985 | 10.327 |
| 50 | 17.169 | 26.347 | 59.420 |
| 80 | 47.255 | 239.434 | 197.439 |
| 100 | 67.173 | 372.245 | 290.903 |
| 200 | 237.805 | 1668.995 | 1391.415 |
| 500 | 1493.305 | 10464.894 | 11296.066 |

根据表3.10-表3.12的实验结果，分别作出查准率、查全率和运行时间的折线图，如图3.5至图3.7所示。



图3.5 SNM、MPN、IMPN算法查全率折线图



图3.6 SNM、MPN、IMPN算法查准率折线图



图3.7 SNM、MPN、IMPN算法运行时间折线图

根据表3.10至表3.12计算得到三种算法的每种数据集上的F1值，如表3.13所示，并作出如图3.8所示的F1值折线图。

表3.13 SNM、MPN、IMPN算法F1-measure值对比表

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集大小（千条） | SNM算法（%） | MPN算法（%） | IMPN算法（%） |
| 5 | 89.258 | 97.278 | 99.149 |
| 10 | 87.712 | 95.725 | 98.967 |
| 20 | 87.648 | 94.336 | 98.765 |
| 50 | 87.597 | 93.262 | 97.413 |
| 80 | 86.537 | 90.126 | 97.114 |
| 100 | 86.237 | 89.304 | 96.709 |
| 200 | 84.847 | 88.672 | 94.166 |
| 500 | 84.159 | 88.095 | 92.334 |



图3.8 SNM、MPN、IMPN算法F1-measure折线图

由图3.5可以看出：改进的IMPN算法面对不同大小的数据集（其他条件如滑动窗口大小、数据集特征参数等均一致），查全率均高于SNM算法和MPN算法，并且IMPN算法查全率的数值稳定在96%以上。所以证明了改进的IMPN算法在查全率上相较于传统的MPN算法拥有较大的提升；随着数据集的增大，SNM算法和MPN算法的查全率均呈下降趋势，而IMPN算法的查全率较为稳定。

由图3.6可以看出：面对不同的数据集，SNM和MPN算法表现较好，这是因为算法中采用的相似记录判等方法是基于当前数据集特征的判定规则，所以需要领域专家的经验和知识；IMPN算法采用基于字段区分度的加权判等方式，在检测大小从5000到200000的数据集时，查准率均在90%以上，属于尚可接受的范围，但是当数据集大小达到500000时，查准率低于90%，所以IMPN算法的查准率还有待改进的空间。本文的第四章针对这一问题提出了新的改进思路，并取得了良好的效果。

由图3.7可以看出：SNM算法时间消耗最低，因为MPN和IMPN算法均包含多趟独立的SNM过程，并且含有传递闭包的计算；与MPN相比较，IMPN算法采用了自适应大小的滑动窗口，这一改进是可以减少时间消耗的，但是IMPN算法还包括对含有空（或者不完整）排序键值的记录进行归并的过程，所以总体的时间和MPN算法差距不大。

综合以上结果可以看出，IMPN算法相对于MPN算法较为明显的改进在于：在和MPN消耗时间差距非常小的同时实现更高的查全率；算法的缺点也很明显，即随着数据量的增大，相似重复记录判等方法表现较差，导致查准率的下降。但是由图3.8可以看出，综合考虑查全率和查准率，IMPN算法的整体表现要好于MPN。

## 3.6 本章小结

本章首先介绍了MPN算法的缺点，然后在此基础上提出了一种改进的算法IMPN，并从三个方面介绍了传统MPN的处理方法的不足以及IMPN算法的对其缺点的改进。之后介绍了IMPN算法的设计与流程步骤，最后通过采用数据生成器febrl生成的不同规模的数据集进行对比实验，实验验证了IMPN算法的查全率和F1值都较高，比传统的SNM算法和MPN算法更优，但是也反映了IMPN算法查准率不够理想的问题，因此，本文继续对IMPN算法进行改进，详细介绍在第四章。

# 第四章 基于遗传神经网络改进的IMPN算法

IMPN算法采用“排序/归并”的思想，在归并过程中，需要采用有效的方式对滑动窗口内的记录进行判等，IMPN采用基于字段区分度的加权判等方式，对不同的字段分配相应的权值，这种方式表现不稳定，随着数据量的增加算法的查准率下降。传统的MPN算法主要根据人工经验选择有效的字段对不同的记录进行判断，因此，受到人为因素的影响较大。为了避免这个弊端，同时提高IMPN算法的查准率，本章提出了基于遗传算法和神经网络相结合的、有监督地学习判等方法，并与IMPN的算法思想相结合，提出增强的多趟近邻排序算法（Advanced - Improved Multi-Pass Sorted Neighborhood，A-IMPN）。A-IMPN算法的查全率和查准率都优于现有算法，但是遗传神经网络的训练耗时较大，所以，我们在A-IMPN的基础上进行简化，只训练BP神经网络进行判等操作，由此得到基于BP网络的多趟近邻排序算法（BP-based Improved Multi-Pass Sorted Neighborhood，BP-IMPN）。

本章首先介绍了遗传神经网络在相似重复记录检测问题上的操作过程，然后分别给出A-IMPN算法和BP-IMPN算法的具体操作过程，并通过实验验证了本文提出的A-IMPN算法和BP-IMPN算法的性能。

## 4.1 遗传神经网络用于相似重复记录检测

### 4.1.1 BP神经网络的设计

BP神经网络是一种前馈传播神经网络，因采用BP（Back Propagation）算法进行网络模型的训练而得名。



图4.1 神经网络的拓扑结构

BP神经网络一般由3层或者3层以上神经元组成，对于简单的3层BP神经网络，它包括输入层、输出层、隐含层，图4.1给出了三层神经网络的结构图。引理[[[40]](#endnote-41)]指出：三层的反向传播神经网络能够以任意的精度逼近非线性函数。因为网络层数的增加在增强表达能力的同时也会带来网络复杂度的增加，所以，设计BP神经网络时，优先考虑增加隐含层的结点的个数，而不是增加网络层数。针对相似重复记录检测问题，采用简单的三层BP神经网络。

输入层的结点数由数据集中记录的字段数目确定，例如，对于表3.7所示的数据，每条记录都有16个字段，其中，第一个字段“rec\_id”是为了方便计算算法的查重效果而设计的，不属于数据集的特征，所以这个数据集共有15个字段，因此，输入层应当有15个结点（不含一个偏置结点）；输出层结点为1，输出的结果介于0到1之间，结果越大表示两条记录相似程度越高。隐含层结点数目由公式(4-1)[[[41]](#endnote-42)]计算得到。



其中代表隐含层结点的个数，代表输入层结点的个数，代表输出层结点的个数，计算过程的结果需要四舍五入进行修整。在本例中，=1，=15，=7，=24，所以计算得到隐含层结点数=24。

本文使用Sigmoid函数作为激活函数，sigmoid函数如公式(2-9)所示。

神经网络的初始权值的选择较为关键，它影响着网络的学习是否达到局部最小、最终是否能够收敛，也会影响训练时间的长短。最理想的情况是，初始化赋值之后，每个神经元的输出接近于0，这样可以在sigmoid激活函数的导数取最大值的地方进行下一轮权值的调节与变化，在本文中，初始权值取之间的随机数。

权值更新的学习速率决定了每次训练结束后权值更新的幅度，过大的学习速率会导致系统震荡，过小的学习速率会导致训练学习时间变长，收敛速度也变得很慢。一般情况下，选择较小的学习速率，以保证系统稳定性而不会导致修改幅度过大。期望误差代表着当训练后的误差结果在可接受范围之内则人为进行收敛，停止训练。

BP神经网络训练终止的条件是训练误差达到预先设定的阈值或权值更新次数达到预先设定的迭代次数，本文对神经网络的实现过程中，同时采用这两种终止条件，只要满足其中一条，则训练终止。

针对相似重复检测问题，将已知记录是否重复的数据作为训练集，用于训练神经网络，得到训练完成的网络后，用该网络对测试数据进行相似重复检测。使用BP神经网络进行相似重复记录检测的算法流程图如图4.2所示：



图4.2 BP网络检测相似重复记录流程图

### 4.1.2 基于遗传算法改进的神经网络

BP神经网络通过训练能够有效地对测试数据集中的相似重复记录进行检测判断，解决了大数据量情况下传统的基于“排序、归并”思想的检测算法检测效果较差的问题，并且拥有较好的适应性。但BP神经网络在训练过程中可能会陷入局部最优。GA（Genetic Algorithm）算法是一种模拟生物进化过程的进化算法，它利用“物竞天择，适者生存”的进化思想，通过选择、交叉、变异等操作不断演化，并在演化过程中淘汰掉适应度较差的个体，经过数代进化之后筛选出适应度较高的个体，得到全局的最优解[[42]](#endnote-43)。遗传算法拥有较强的全局搜索能力，所以可以将其应用到BP神经网络中，以解决神经网络难以跳出局部最优的问题[[[43]](#endnote-44)][[[44]](#endnote-45)]。

基于遗传神经网络求解相似重复记录检测的遗传算法框架的元素主要有：

1.染色体

采用实数编码的编码方式，将BP神经网络的权值和阈值作为一个独立的染色体，设种群规模为，则共有组神经网络权值和阈值，即需要训练个BP神经网络。

以图4.1所示的神经网络为例，将训练得到的BP神经网络的权值以及阈值转化为染色体，输入层包含4个神经元，1个偏置单元，隐含层包含6个神经元，所以输入层到隐含层之间的权值共有(4+1)\*6=30个，输出层有1个神经元，所以隐含层到输出层之间的权值共有6+1=7个，因此，该神经网络共有37个权值，1个输出层阈值，所以采用实数编码时，染色体长度为38。

2.适应度函数

在重复检测问题中，适应度函数用于衡量相似重复检测结果和真实结果的之间的差距，差距越小，说明这个染色体所代表的的BP神经网络对数据的检测结果越准确，所以适应度值越大，相反，染色体的适应度就越小。本文使用的适应度函数如公式(4-2)所示。



公式(4-2)中，代表全局误差，的计算公式如公式(4-3)所示。



公式(4-3)中，代表训练数据集的记录总数，代表期望的输出结果，代表实际的输出结果。

3.选择

采用轮盘赌方法进行染色体的选择，使得适应度函数值较大的个体被选中的概率较大。轮盘赌算法的计算公式(4-4)如下：



其中，代表种群中个体的数量，代表第个染色体的适应度函数的值。

4.交叉、变异

采用实数编码，所以在交叉操作时交换的是两条染色体上相同位置的两个实数，采用两点交叉操作。

变异算子模仿的是基因突变的过程：染色体某个位置上的基因突变成为其等位基因，从而可能引发性状表现上的变异。

为了使群体能延续表现更好的个体的性状，在交叉和变异时，适应度值高的个体交叉和变异概率较低。对于适应度为的个体，在进行交叉或变异时的概率为：



其中、代表变异率（或者交叉率）的上、下限，、代表当前代的种群中适应度的最大、最小值。若用表示一个个体，令，，则染色体中的每个位置上的基因的变异过程计算公式如(4-6)所示：



其中表示变异后的值，是0到1之间随机数，表示当前是迭代过程中的第代，的取值如公式(4-7)所示：



其中是取值在0到1之间的随机数，是最大迭代次数。由公式(4-7)可知，的取值随着的增加而接近于0，这样可以使得变异概率在种群迭代初期时较大，而后逐渐减小，有利于种群的稳定性。

使用遗传算法改进的BP神经网络进行相似重复记录检测的流程图如图4.3所示。首先进行有监督的训练学习得到组BP网络，然后用这组BP网络的权值初始化含有条染色体的种群，经过选择、交叉、变异等操作获取最优的BP网络，最后对待检测数据集合进行重复性检测。使用这种方法进行相似重复记录检测时，需要对数据集中的任意两个不同的记录进行比较，判断他们是否相似或重复，存在冗余判断过程，因此时间复杂度较高，为。所以需要对其进行改进，在保留遗传神经网络所具有的优势的同时，减小算法的时间复杂度。



图4.3遗传神经网络检测相似重复记录流程图

## 4.2 基于神经网络的A-IMPN算法和BP-IMPN算法

A-IMPN算法和BP-IMPN算法的操作过程都包括两部分：（1）根据IMPN算法对原数据进行处理，包括基于字段区分度提取排序关键字、依据关键字对所有记录进行排序、选取自适应大小的滑动窗口；（2）使用某种判等方法，对滑动窗口内的数据进行判断，最终得到重复记录。与传统的基于遗传神经网络求解相似重复记录问题相比，A-IMPN算法和BP-IMPN算法首先通过IMPN算法对数据集进行预处理，这样就避免了大量冗余的判等操作，因此时间复杂度降低。A-IMPN算法使用遗传神经网络进行判等，BP-IMPN算法仅使用BP神经网络进行判等。

### 4.2.1 基于遗传BP网络的A-IMPN算法

传统SNM算法和MPN算法使用的是基于专家经验知识的规则产生式系统（OPS5），这种判等方法主要依赖人工操作以选择进行比较的字段。第三章提出了基于字段区分度的加权判等方式，对不同的字段分配相应的权值，但是算法的查准率表现不理想。传统的基于遗传神经网络进行相似重度检测的算法不存在排序操作，所以需要在数据全集对数据进行两两判断，时间复杂度为。

本文提出了A-IMPN算法，经过训练完成的神经网络判断滑动窗口内的记录是否重复，这样可以根据数据集的特点进行动态调整，减少了人工干预，同时可以提高算法对多种数据集的适应性。

算法对于给定的训练集，共训练个BP神经网络，然后用训练得到的个BP神经网络的权值作为初始种群，再经遗传算法进行操作，不断优化，最后从个BP神经网络中选择最优的结果作为最终训练得到的神经网络，然后用这个神经网络进行判等操作，算法步骤如下：

|  |
| --- |
| **算法4.1：**A-IMPN算法 |
| 1.读取数据集，得到待检测的数据以及训练数据集；  2.计算数据集字段的字段区分度并排序；  3.优先选取区分度较大的字段去生成组排序关键字；  4.随机初始化出个BP神经网络；  5.对个BP网络进行迭代训练；  6.将训练完成的个BP网络初始化种群；  7.使用遗传算法的选择、交叉、变异等操作求种群的最优化解；  8.独立地执行步骤9~12次；  9.按照排序关键字的产生方式对每条记录提取其排序关键字；  10.对数据集按照关键字排序，如果某条记录的不完整或者为空则将该记录的ID加入到缺失关键字记录集合中，完整则正常排序；  11.采用进行可伸缩大小的滑动窗口重复检测得到重复集合；  12.将与进行重复归并，然后计算此集合的传递闭包；  13.将次SNM重复检测得到的集合进行归并，然后计算传递闭包得到最终的重复记录集合。 |

使用A-IMPN算法进行相似重复记录检测的流程图如图4.4所示。



图4.4 A-IMPN算法流程图

与IMPN算法的基于字段区分度的加权判等操作相比，使用遗传神经网络进行判等可以有效地提高算法的查准率。同时，与传统的使用遗传神经网络求解该问题的算法相比，IMPN算法对数据集进行了预处理，只需要比较滑动窗口范围内的记录，减少了不必要的判断计算，时间复杂度降低。

### 4.2.2 基于BP神经网络的BP-IMPN算法

A-IMPN算法的一个缺点是网络训练时间过长。设BP神经网络的输入层节点个数为，隐含层节点个数为，则一次训练过程（包括前向计算输出和误差反向更新过程）的时间复杂度为，设BP网络终止训练过程的最大次数为，遗传算法的种群数量为，则种群初始化需要训练个BP神经网络，时间复杂度为，种群演化过程的时间复杂度为，所以A-IMPN算法训练过程的总时间复杂度如公式(4-8)：



可以看出算法训练过程的时间复杂度为平方级别。除此之外，BP神经网络优化算法属于高维优化问题，算法训练的结果往往是收敛于某个鞍点附近，而不是局部最小值[[[45]](#endnote-46)]。在鞍点和局部极小值的梯度都等于零，但是Hessian矩阵在鞍点附近有正、负特征值，而在局部极小值附近的Hessian矩阵是正定的。大量鞍点的存在才是神经网络优化困难的真正原因，而基于遗传算法改进的BP神经网络算法对于拥有较多局部极值的问题效果较好，适用较为局限，并且算法的搜索效率低下，相对于基于梯度的算法要多倍的搜素[[[46]](#endnote-47)]。所以，在A-IMPN算法的训练过程中，往往牺牲了更多的时间但是并没有达到理想的优化效果，故本节提出了简化的BP-IMPN算法，训练过程仅采用BP神经网络。BP-IMPN步骤如算法4.2所示：

|  |
| --- |
| **算法4.2：**BP-IMPN算法 |
| 1.读取数据集，得到待检测的数据以及训练数据集；  2.计算数据集字段的字段区分度并排序；  3.优先选取区分度较大的字段去生成组排序关键字；  4.随机初始化BP神经网络；  5.对BP网络进行迭代训练得到；  6.独立地执行步骤7~10次；  7.按照排序关键字的产生方式对每条记录提取其排序关键字；  8.对数据集按照关键字排序，如果某条记录的不完整或者为空则将该记录的ID加入到缺失关键字记录集合中，完整则正常排序；  9.采用进行可伸缩大小的滑动窗口重复检测得到重复集合；  10.将与进行重复归并，然后计算此集合的传递闭包；  11.将次SNM重复检测得到的集合进行归并，然后计算传递闭包得到最终的重复记录集合。 |

BP-IMPN算法流程图如图4.5所示：



图4.5 BP-IMPN算法流程图

## 4.3 IMPN、A-IMPN、BP-IMPN综合对比实验

本节对IMPN算法、A-IMPN算法和BP-IMPN算法在不同数据集上进行实验，并统计实验结果。

实验环境设置和数据集设定同第三章3.5节相同，均采用8个规模不同但生成方式相同（生成器“febrl”的参数一致）的数据集，对于神经网络的训练，训练过程采用的数据集大小为500，即含有500条记录，使用“febrl”生成，生成参数与实验数据集一致，详情见表3.7与表3.8。

对于IMPN算法，最大滑动窗口大小=20，最小滑动窗口大小=3，算法过程中进行单趟SNM的次数=3。

对于BP神经网络的训练，学习速率设置为0.05，动量系数设置为0.9[[[47]](#endnote-48)]，网络最大迭代次数设置为5000；遗传算法种群初始大小150，交叉概率=0.3，变异概率=0.05，种群最多迭代300次。

查全率、查准率和运行时间（不含训练过程）的实验结果分别如表4.1至表4.3所示：

表4.1 IMPN、A-IMPN和BP-IMPN算法查全率对比表

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集大小（千条） | IMPN算法（%） | A-IMPN算法（%） | BP-IMPN算法（%） |
| 5 | 98.80 | 98.90 | 98.80 |
| 10 | 98.40 | 98.35 | 98.15 |
| 20 | 98.75 | 98.90 | 98.90 |
| 50 | 97.90 | 97.85 | 97.65 |
| 80 | 98.58 | 98.53 | 98.39 |
| 100 | 98.73 | 98.76 | 98.78 |
| 200 | 99.10 | 99.15 | 99.50 |
| 500 | 96.91 | 97.23 | 97.43 |

表4.2 IMPN、A-IMPN和BP-IMPN算法查准率对比表

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集大小（千条） | IMPN算法（%） | A-IMPN算法（%） | BP-IMPN算法（%） |
| 5 | 99.50 | 100.00 | 100.00 |
| 10 | 99.54 | 100.00 | 100.00 |
| 20 | 98.78 | 100.00 | 99.98 |
| 50 | 96.93 | 99.87 | 98.95 |
| 80 | 95.69 | 99.85 | 98.80 |
| 100 | 94.77 | 99.88 | 98.81 |
| 200 | 89.70 | 99.68 | 97.81 |
| 500 | 88.17 | 99.21 | 97.69 |

表4.3 IMPN、A-IMPN和BP-IMPN算法运行时间对比表

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集大小（千条） | IMPN算法（s） | A-IMPN算法（s） | BP-IMPN算法（s） |
| 5 | 1.274 | 21.110 | 20.800 |
| 10 | 2.961 | 83.169 | 77.989 |
| 20 | 10.327 | 336.404 | 328.788 |
| 50 | 59.420 | 2172.766 | 1928.253 |
| 80 | 197.439 | 5510.201 | 5287.167 |
| 100 | 290.903 | 8505.129 | 8119.437 |
| 200 | 1391.415 | 32842.004 | 30901.765 |
| 500 | 11296.066 | 204281.991 | 193198.802 |

根据表4.1-表4.3所示的查全率、查准率和运行时间，分别作出IMPN算法、A-IMPN算法和BP-IMPN算法的查全率折线图、查准率折线图和运行时间折线图，如图4.6至4.8所示：

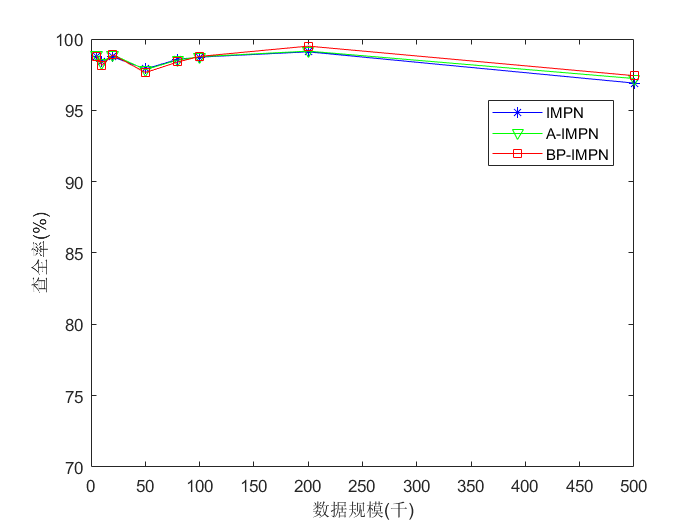


图4.6 IMPN、A-IMPN、BP-IMPN算法查全率折线图

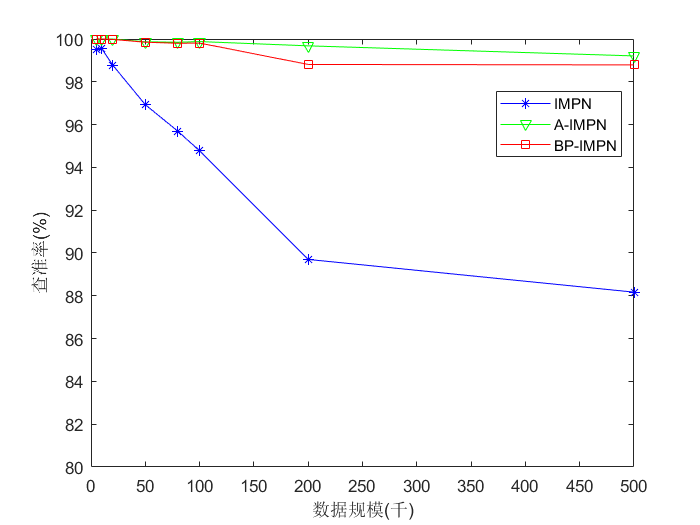


图4.7 IMPN、A-IMPN、BP-IMPN算法查准率折线图

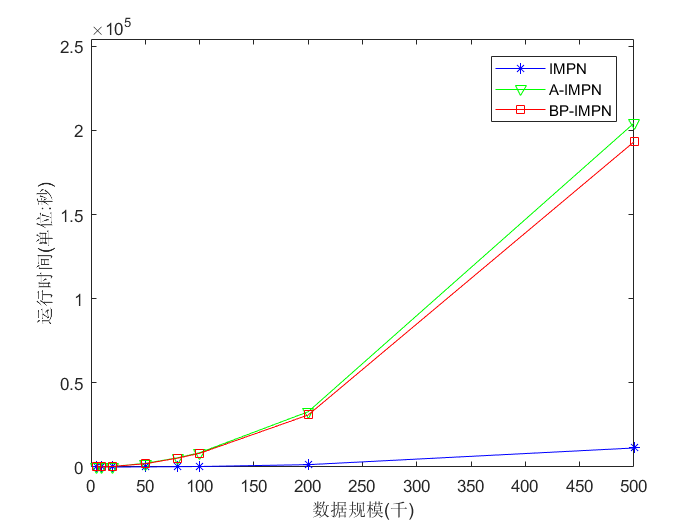


图4.8 IMPN、A-IMPN、BP-IMPN算法运行时间折线图

由图4.6查全率对比结果可以看出，A-IMPN算法、BP-IMPN算法与IMPN算法相比，查全率非常接近，证明基于IMPN改进的A-IMPN算法和BP-IMPN算法能够保证较好的查全率，这与理论上的结果一致，因为基于神经网络改进的算法和IMPN算法的主要区别在于判等过程采用了不同的方法。

由图4.7查准率对比结果可以看出，A-IMPN算法和BP-IMPN算法相对于IMPN算法，查准率均有较大的提升，

由上图可以看出，改进的IMPN算法的时间消耗远高于IMPN算法。这是由于在相似记录进行重复判断时，IMPN算法的时间复杂度是常数级的，而改进的IMPN算法在判等时，若BP神经网络的输入层结点个数为，隐含层结点个数为，则时间复杂度为。

综上，基于遗传神经网络改进的算法在保证了IMPN算法较好查全率的基础上，弥补了算法查准率较低的缺陷，但是以牺牲时间为代价。取得此实验结果的理论原因在于：两者最主要的区别即是相似重复记录判等阶段，改进的算法采用遗传算法改进BP神经网络，然后通过学习训练数据集字段相似与整体记录相似的非线性关系，可以较为准确地对两条记录相似与否进行预测和判断，但同时神经网络前向传播的计算时间比IMPN算法采用基于权重的判等过程时间复杂度高，导致了算法整体时间消耗的增加。算法另外一个缺点在于，检测效果非常依赖于训练数据集的选取与训练参数的调整，令算法的应用变的更加复杂。

## 4.4 本章小结

本章首先介绍了目前较为成熟的使用遗传神经网络进行相似重复记录检测的方式：使用训练数据集对BP神经网络进行训练，然后针对其易陷入局部最小值的缺点，引入遗传算法对其进行改进。然后说明这种方式的一个很明显的缺点就是时间复杂过大，接着引出遗传神经网络应用于IMPN算法中可以互相取长补短：遗传神经网络可以提高IMPN算法的查准率，IMPN算法作为对遗传神经网络算法的预处理，降低了其时间复杂度。最后通过设计对比实验证明了改进的IMPN算法相对于IMPN算法的高查准率的优点。

# 第五章 航天情报系统中的相似重复记录检测

目前数据清洗技术在各行业的信息管理系统中取得了广泛的应用。本章首先介绍了航天情报信息管理系统的需求分析及概要设计与技术实现，然后重点介绍了作者主要负责的数据清理模块，包括数据清理模块的设计、重复记录产生的原因、IMPN算法在系统中的应用以及该算法对数据质量的提高等。

## 5.1 系统需求分析

### 5.1.1 系统建设背景与目标

北京空间科技信息研究所为了提高科技化水平，实现航天情报数据的采集、处理、分析的信息化，于2016年开展“航天情报信息管理系统”项目的研究。该研究以“知识结构化、成果产品化”为目标，立足多年的情报信息数据积累，致力于打造一款功能丰富、实用高效的情报数据信息管理系统。

### 5.1.2 需求分析

“航天情报信息管理系统”主要面向研究所内部研究人员的日常办公使用，经过项目调研与分析后将系统的总体需求概括如表5.1所示：

表5.1需求分析总结表

|  |  |  |
| --- | --- | --- |
| 运行环境需求 | Web | 操作系统为Windows XP，浏览器为Internet Explorer 8 |
| iOS | 操作系统为iOS8.0及其以上 |
| Android | 操作系统为Android4.0及其以上 |
| 功能性需求 | 1. 数据采集模块：将现有的数据采集到系统中并保持和系统中的数据格式一致，包括两种采集模式：人工在线录入以及Excel表格批量导入。 2. 数据清洗模块：对多源数据合并导致的重复数据进行检测清理。 3. 数据检索模块：方便研究人员更加快捷地检索和查询所需信息。 4. 服务支撑模块：包括用户权限管理、数据异常下载行为监视、综合营销平台建设。 5. 移动应用模块：涵盖iOS和Android操作系统在内的手机端App，是系统在手机端的简化体现，方便研究人员随时查看相关信息。 6. 数据应用模块：在已有数据集的基础上，对数据进行统计，并进行可视化展示，方便研究人员更直观地分析数据。 | |
| 非功能性需求 | 1. 性能需求：并发用户数2000，事物平均响应时间3.0s。 2. 稳定性需求：双机热备方案。 3. 安全性需求：网络/系统的安全监测与检查、反爬虫设计等。 | |

## 5.2 系统设计与实现

上一节对系统的需求进行了分析总结，本节主要介绍“航天情报信息管理系统”的设计与实现方式，主要包括系统架构、数据库设计、功能模块实现等。

### 5.2.1 系统概要设计

根据5.1节中的系统功能性需求分析，可以将本系统按照功能模块划分成6个主要的部分，如图5.1所示：



图5.1 六大功能模块示意图

“航天情报信息管理系统”的总体设计从以下三个层面展开：前端和移动端的交互设计、服务器端的逻辑功能、数据持久化的实现。

持久化层主要是负责数据的存储以及向服务器端提供增删改查的服务接口，这里存储了“航天情报信息管理系统”的核心数据信息。

服务器端是处理业务逻辑的核心层，是系统的枢纽部分。数据请求由前端发给服务器端，经用户鉴权通过之后向持久化层请求数据并进行整理发送给前端页面。

前端交互部分主要包括Web网页界面和App移动端页面，这一层是直接和用户交互的最上一层，负责接收用户的指令以及向用户呈现系统信息等。它主要包括用户的注册与登录、数据检索与查询、可视化展示、数据统计等功能页面。

系统的总体架构如下图：



图5.2 航天情报信息管理系统架构图

“航天情报信息管理系统”中的数据设计是系统设计较为核心的一个环节。数据内容主要包括航天器、轨道信息、发射场、运载火箭、航天国家与机构、航天器故障信息等。其中，航天器信息数据是系统的核心数据，按照所属类别又可以将其分成8种：通信卫星、导航卫星、遥感卫星、在轨服务与空间安全卫星、空间科学卫星、技术试验卫星、空间探测器和载人航天器。系统数据库对应的ER（实体-关系）图如图5.3所示：



图5.3实体关系结构图

由上图可知，航天器实体是系统数据库中的最关键实体，它与故障、卫星平台、航天机构、航天国家、航天发射场、运载火箭都存在直接的关系。其中，除了和航天器故障对应关系是“一对多”之外，和其他的几个实体的关系都是“多对一”。航天国家和与其相连的几个实体的关系均为“一对多”。

接下来介绍几个核心数据库表的设计：

（1）航天器表

表5.2 航天器字段设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名称 | 属性描述 | 类型 | 可否为空 | 备注 |
| spacecraft\_id | 航天器ID | int(11) | 否 | 主键 |
| spacecraft\_name\_cn | 航天器中文名称 | varchar(60) | 否 | 无 |
| spacecraft\_name\_en | 航天器英文名称 | varchar(40) | 否 | 无 |
| spacecraft\_launch\_num | 航天器发射编号 | varchar(20) | 否 | 无 |
| spacecraft\_num | 航天器编号 | varchar(20) | 否 | 无 |
| task\_property | 任务性质 | varchar(5) | 否 | 无 |
| spacecraft\_type | 航天器类型 | varchar(50) | 否 | 无 |
| country\_id | 所属国家ID | int(11) | 否 | 外键 |
| country | 所属国家 | varchar(20) | 否 | 无 |
| institution\_id | 所属机构ID | int(11) | 否 | 外键 |
| institution\_name | 所属机构 | varchar(200) | 否 | 无 |
| operator\_id | 运营单位ID | int(11) | 否 | 外键 |
| operator\_name | 所属运营单位 | varchar(200) | 否 | 无 |
| … | … | … | … | … |
| spacecraft\_image | 外形图片 | varchar(60) | 是 | 无 |

由于篇幅限制表5.2只给出了部分关键字段的设计。从上表可以看出航天器的ID是标识一条航天器记录的唯一关键字，而该表中的外键：country\_id、institution\_id、operator\_id则是航天器与“所属国家”、“所属机构”以及“所属运营单位”三张表的关联。

（2）航天国家表

表5.3 航天国家字段设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名称 | 属性描述 | 类型 | 可否为空 | 备注 |
| country\_id | 国家ID | int(11) | 否 | 主键 |
| country\_name\_cn | 国家中文名称 | varchar(60) | 否 | 无 |
| country\_name\_en | 国家英文名称 | varchar(60) | 否 | 无 |
| budget\_per\_year\_gov | 政府年度航天预算 | float(8,3) | 是 | 无 |
| budget\_per\_year\_civil | 民用年度航天预算 | float(8,3) | 是 | 无 |
| … | … | … | … | … |
| main\_spacecraft | 主要航天器 | text | 是 | 无 |

（3）航天器故障表

表5.4航天器故障字段设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名称 | 属性描述 | 类型 | 可否为空 | 备注 |
| malfunction\_id | 故障ID | int(11) | 否 | 主键 |
| malfunction\_spacecraft\_id | 故障航天器ID | int(11) | 否 | 外键 |
| malfunction\_level | 故障等级 | varchar(10) | 否 | 无 |
| malfunction\_date | 故障发生时间 | date | 否 | 无 |
| malfunction\_in\_designlife | 是否发生  在寿命期 | tinyint(1) | 否 | 无 |
| … | … | … | … | … |
| malfunction\_consequence | 故障后果 | text | 是 | 无 |

（4）卫星平台表

表5.4 卫星平台字段设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名称 | 属性描述 | 类型 | 可否为空 | 备注 |
| satellite\_platform\_id | 平台ID | int(11) | 否 | 主键 |
| platform\_dev\_org\_id | 平台研制  单位ID | int(11) | 否 | 外键 |
| platform\_dev\_data | 研制时间 | date | 否 | 无 |
| platform\_descrip | 平台描述 | text | 否 | 无 |
| … | … | … | … | … |
| platform\_image | 平台图片 | varchar(60) | 是 | 无 |

### 5.2.2 系统实现

（1）系统开发与运行环境

系统的开发与运行环境如表5.5所示：

表5.5 航天情报信息系统开发与运行环境

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 服务器端 | | Web端 | App | |
| iOS | Android |
| 开发平台 | Windows 7 | | Windows 7 | macOS Yosemite | Windows 7 |
| 开发工具 | Intellij idea  JDK 1.8.0 | | Sublime | Xcode 8 | Android Studio  JDK 1.8.0 |
| 数据库 | MySQL | |  | SQLite | SQLite |
| 运行环境 | 硬件 | 操作系统：  Windows server  2008以上  内存大小：4G  磁盘空间：300G | Internet Explorer8及以上 | iOS 8.0及以上 | Android 4.0  及以上 | |
| 软件 | JDK 7.0.71  Tomcat 7.0.54  MySQL 5.6  Navicat 11 |

（2）技术路线

本系统采用B/S加移动端C/S的综合技术方案进行实现。其中服务端采用较为成熟的SSM（Spring、SpringMVC、MyBatis）三层技术框架，使用Maven添加依赖。数据持久化层由MyBatis实现，它起到了对JDBC的封装作用。服务端的业务逻辑由Spring控制，Spring框架起衔接SpringMVC和MyBatis框架作用。移动端采用经典的MVC技术路线，Model层负责沙盒内数据的封装与维护，View层负责“空间瞭望”App的页面展示与用户交互、包括搜索、统计、航天器信息分类浏览等，ViewController负责处理逻辑业务，如更新航天器列表、收藏航天器、统计信息提取等，并调用Model的接口更新数据库内容。

（3）接口设计

服务端与前端以及移动端的数据传输采用JSON的数据格式，JSON数据更加简便易读易操作，前后端交互采用Http通信协议。核心的接口设计如表5.6所示：

表5.6 数据接口设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 接口名称 | 请求方式 | 接口说明 |
| 1 | getHasLaunchedSpacecraft | GET | 请求已发射航天器列表 |
| 2 | getSpacecraftByCountry | GET | 按照国家分类返回  航天器列表 |
| 3 | getCountBySpacecraftType | GET | 请求某类型所有航天器数量 |
| 4 | getCountByCountry | GET | 请求某国家所有航天器数量 |
| 5 | getSearchResult | GET | 按照关键字返回检索结果 |
| 6 | getMyCollection | GET | 返回当前登录账户  收藏的航天器列表 |
| 7 | getSpacecraftDetailByID | GET | 返回某个航天器的详细信息 |
| 8 | addToCollectionByID | POST | 收藏某颗航天器 |
| … | … | … | … |
|  | login | POST | 登录 |

## 5.3 数据清洗模块

### 5.3.1 “脏数据”产生原因

“航天情报信息管理系统”的数据采集方式有两种，包括人工在线填报数据以及从现存的Excel表格数据批量导入到系统中。人工操作的出错是难以避免的，再加上现存Excel数据来自于不同的子部门由不同的研究人员维护，没有统一的标准，以上即是现存数据集中的“脏数据”产生的主要原因。再加上数据库中的航天器信息多来自于不同的渠道，这就使得数据集中并不存在一个能唯一标识航天器的字段。

表5.7 航天器重复记录举例

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 航天器  名称 | 发射场 | 发射  结果 | 发射时间 | 国家 | 研制单位 | …… | 运载火箭 |
| A | Cape Canaveral | 成功 | 2010/8/14 11:07 | 美国 | 洛克希德-马丁 | …… | 宇宙神-5 |
| A1 | 卡纳维  拉尔角 | 成功 | 2010.08.14  11:07 | US | 洛马 | …… | Atlas-5 |
| B | 卡纳维拉尔  角发射场 | 失败 |  | United.States | Lockhead Martin | …… | 猎鹰-9 |

表5.7展示了三条航天器记录A、A1、B（因真实数据涉及商业机密故数据略有修改）的部分信息，其中A和A1对应着同一颗航天器，B对应另外的一颗航天器。由上表可以看出待处理的数据主要有以下特征：

（1）中英文格式不统一，如“洛克希德-马丁”和“Lockhead Martin”、“宇宙神-5”和“Atlas-5”等。

（2）存在缺失数据，如B的发射时间信息缺失。

（3）中英文缩写与全拼格式不统一，如“United States”和“US”、“洛克希德-马丁”和“洛马”等。

（4）时间格式不统一，如A的“2010/8/14”和A1的“2010.08.14”。

### 5.3.2 重复记录检测算法的应用

数据清洗模块的调用发生在数据录入模块的过程中。用户首先将数据由Web端录入，由浏览器发送到服务器端进行处理。服务器端首先进行预处理，包括时间格式的统一等操作，然后调用重复记录监测模块。该模块的业务流程如图5.4所示：



图5.4 数据清洗业务流程图

由于“航天情报信息管理系统”中总的数据量不超过8000条，所以数据重复检测模块可以采用IMPN算法实现。在两条记录进行相似重复判断时，考虑到两条航天器同一时刻发射的概率非常小，所以在判重过程中可以人为设定“发射时间”占有较高的权重。数据预处理过程主要是按照固定的规则对数据进行检查，如表5.7所示：

表5.7 数据预处理规则表

|  |  |
| --- | --- |
| 规则名称 | 规则描述 |
| 主键判空 | 如果记录的主键为空，则忽略该记录 |
| 日期格式检查 | 标准格式为yyyy-MM-dd HH:mm:ss，将非标准格式的日期统一成标准格式 |
| 日期非法内容检查 | 年份超过当前年份或者当月天数超过31天，对该记录进行标记，由人工检查处理 |
| 国家字段检查 | 国家字段的标准格式为英文全称，将中文内容和英文缩写内容统一成标准格式 |

### 5.3.3 实验设计与结果分析

本节主要目的是通过实验设计说明IMPN算法对“航天情报信息管理系统”数据质量的提升。实验采用3个不同大小的航天器数据集进行测试，数据集均来自于航天情报信息管理系统，如下表所示：

表5.8 航天器测试数据集

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集名称 | dataset1 | dataset2 | dataset3 |
| 数据量（条） | 500 | 1000 | 2000 |
| 重复记录数（条） | 100 | 200 | 400 |
| 字段数目（个） | 24 | 24 | 24 |
| 单条记录最多重复数（个） | 5 | 5 | 5 |

实验采用的IMPN算法的滑动窗口最小值=3，最大值设为=10，算法过程中进行单趟SNM的次数=3。由于系统中的数据量较小（不超过8000条记录），而且没有唯一的ID去区分不同的记录，所以若采用基于遗传神经网络改进的IMPN算法的话，需要人为提取一定量的重复记录进行训练，需要花费大量的时间而且不一定能取得良好的效果，所以这里采用第三章介绍的IMPN算法（未采用遗传神经网络进行改进）。检测结果如下表：

表5.9 IMPN算法检测结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据集名称 | 数据量（条） | 查准率（%） | 查全率（%） | 运行时间（s） |
| dataset1 | 500 | 96.08 | 98.00 | 0.118 |
| dataset2 | 1000 | 97.04 | 97.50 | 0.308 |
| dataset3 | 2000 | 97.53. | 98.75 | 0.735 |

从以上结果可以看出，IMPN算法对航天器记录的重复检测取得了良好的效果：IMPN算法的查全率和查准率均达到了95%以上，且运行时间在可接受的范围内。

## 5.4 本章小结

本章首先介绍了“航天情报信息管理系统”的需求分析以及系统设计，然后介绍了一些关键技术的核心实现。接着以数据清洗模块为重点进行展开，介绍了系统中的“脏数据”的来源，数据清洗算法在系统中的应用，并设计了对比实验验证算法对数据质量的提升。

# 第六章 总结与展望

## 6.1 总结

随着信息化技术的发展与应用，各式各样的信息管理系统支撑着企业的数据管理与维护。网络上积累了大量的数据，但这些数据的质量参差不齐。由于数据来源不同、存储于不同的操作系统以及硬件平台、人为错误等导致了包括错误数据、相似重复数据和缺失数据等“脏数据”的产生。其中相似重复记录的检测是数据清洗领域的一个研究重点。本文主要工作内容如下：

（1）研究和介绍了相似重复记录检测领域的相关技术。包括基于单字段和多字段的相似度匹配算法、常用的相似重复记录检测算法以及BP神经网络的理论基础等。

（2）基于现有MPN算法提出了IMPN算法。MPN算法主要有两个缺点：排序关键字的选择过于依赖领域专家的经验，并且当记录的排序关键字字段为空时，检测效果会大打折扣；MPN采用固定大小的滑动窗口，这种方式不够灵活，存在许多无效的比较判重。IMPN主要是从以下三点进行了改进：1.优先选择区分度较大的字段提取排序关键字，从一定程度上减少了人为的参与，减弱算法对于专家知识的依赖；2.采用根据窗口内容可伸缩大小的自适应滑动窗口，减少了部分不必要的判等比较，从而提高了算法的效率；3.对排序关键字为空或者不完整的记录进行标记，初步检测完成之后再对这些标记记录进行归并处理，提高了算法的鲁棒性。

（3）将遗传神经网络应用于IMPN算法的判重过程，提高了算法的查准率。经过实验验证，IMPN算法的优点在于实现了较高的查全率，但却随着数据量的增大查准率逐渐下降，这是算法第三个改进点带来的副作用。为了提高IMPN算法的查准率，第四章将遗传神经网络应用到IMPN算法的重复判断过程中：首先利用训练数据集训练多组三层BP神经网络，然后将其初始化为一个种群，采用遗传算法进行优化，可以克服其收敛慢和易陷入局部最优的缺点，最后将最优的BP神经网络用到IMPN算法的滑动窗口判重过程中，两条记录各字段的相似度值作为输入，输出和阈值进行大小判断以确定它们是否互为相似重复记录。经过对比实验可以验证这一改进提高了IMPN算法的查准率。

（4）最后的章节首先介绍了航天情报信息管理系统的需求分析、概要设计、技术实现，然后重点介绍了系统中的数据清理模块，包括数据清理模块的设计、重复记录产生的原因、IMPN算法在系统中的应用等，最后通过设计实验验证IMPN算法对系统中数据质量的提高。

## 6.2 展望

本文主要研究了数据清洗领域的相似重复记录检测问题，针对研究过程中发现的问题与不足之处，今后仍然需要在以下几个方面进行深入研究：

（1）算法对于中文数据库的处理能力较弱，因为在生成排序关键字以及大小排序时对字符的处理是基于值的，中文的相似重复记录检测一般先进行分词处理，所以在中文重复记录检测方面有待于更广泛和深入地研究。

（2）BP神经网络的隐含层结点个数的确定方式目前尚未有严格唯一的定论。隐层结点数过少会导致网络网络性能表现很差；隐层结点数过多，会增加网络训练时间，且可能使训练陷入局部极小值，导致“过拟合”的出现。

（3）实验部分使用的数据量大小还远未到海量数据的标准，当数据量大小超出内存所能容纳的上限时，提取数据集和排序过程均需要作出调整。可能的解决方案包括将数据集划分为多个更小的单位，采用外部排序等。数据量增大时，算法的效率问题也是亟待考察和解决的。

参考文献

致谢

三年的时间转瞬即逝，这段时间的经历让我成长了许多，也学到了很多东西。

首先我要特别感谢我的研究生导师刘志镜教授。刘老师为人和蔼可亲，科研态度严谨，做事专注，知识渊博，为学生树立了优秀的榜样。在我研究生期间，刘老师耐心指导我进行科研，引领我走进科研的大门，每当我遇到个人无法解决的问题时，老师总是耐心地与我共同分析，通过与老师进行讨论，可以帮助我分析产生问题的原因并想到切实可行的解决问题的建议，这都有助于我及时克服科研瓶颈。研究生生活即将结束，但是刘老师严谨的科研态度和认真的敬业精神一直影响着我，指导我在今后的工作和生活中永葆脚踏实地、认真求实的态度，在此我衷心感谢刘老师这几年来对我的指导与帮助。

其次我要感谢姚勇老师。姚老师在我做毕业设计的过程中给予了专业的指导和帮助，姚老师做人做事的态度令我非常欣赏，是我们学习的榜样。

再者还要感谢实验室李金洋博士、王炳华、朱群、黄辉煌、张沐杰等师兄在我研究生期间给予的指导和帮助；感谢同项目组实验室同学张亚超、严欢以及师弟毛彦淇、崔森、李天霖，大家一起为了同一个目标共同奋斗的日子令我难忘。感谢实验室同学陈仲元、王斌、陈磊、薛登峰以及其它师弟师妹对我学习和生活上的支持与帮助。

感谢我的家人对我的支持和关爱，感谢我的女朋友申星的陪伴和支持，你们是我背后最温柔和强劲的支撑力量。感谢舍友们三年来带来的欢乐和笑声。

最后，感谢所有评审老师，感谢您们抽出宝贵的时间对我的论文进行审阅！

作者简介

##### 基本情况

张攀，男，山东菏泽人，1993年7月出生，西安电子科技大学计算机学院计算机技术专业2015级硕士研究生。

##### 教育背景

2011.09～2015.07 西安电子科技大学，本科，专业：软件工程

2015.09～2018.07 西安电子科技大学，硕士研究生，专业：计算机技术

##### 攻读硕士学位期间参与的科研项目

1. 陕西省慧农科技“农掌门”项目，2015.10~2016.04，负责该项目iOS APP农民端的开发和维护。
2. 北京空间科技信息研究所航天智库平台“航天情报信息管理系统”中后台数据清洗模块的开发以及“空间瞭望”App的iOS端研发与维护。

1. [] Rahm E, Do H H. Data cleaning: Problems and current approaches[J]. IEEE Data Eng. Bull., 2000, 23(4): 3-13. [↑](#endnote-ref-2)
2. [] Raman V, Hellerstein J M. An interactive framework for data cleaning[M]. Computer Science Division, University of California, 2000. [↑](#endnote-ref-3)
3. [] 王曰芬, 章成志, 张蓓蓓, 等. 数据清洗研究综述[J]. 数据分析与知识发现, 2007, 2(12): 50-56. [↑](#endnote-ref-4)
4. [] Bitton D, DeWitt D J. Duplicate record elimination in large data files[J]. ACM Transactions on database systems (TODS), 1983, 8(2): 255-265. [↑](#endnote-ref-5)
5. [] Elmagarmid A K, Ipeirotis P G, Verykios V S. Duplicate record detection: A survey[J]. IEEE Transactions on knowledge and data engineering, 2007, 19(1): 1-16. [↑](#endnote-ref-6)
6. [] Jin L, Li C, Mehrotra S. Efficient record linkage in large data sets[C]//Database Systems for Advanced Applications, 2003.(DASFAA 2003). Proceedings. Eighth International Conference on. IEEE, 2003: 137-146. [↑](#endnote-ref-7)
7. [] Wang J R, Madnick S E. The inter-database instance identification problem in integrating autonomous systems[C]//Data Engineering, 1989. Proceedings. Fifth International Conference on. IEEE, 1989: 46-55. [↑](#endnote-ref-8)
8. [] Galhardas H, Florescu D, Shasha D, et al. Extensible framework for data cleaning[C]//2000 IEEE 16th International Conference on Data Engineering (ICDE'00). IEEE, 2000. [↑](#endnote-ref-9)
9. [] 陈伟. 数据清理关键技术及其软件平台的研究与应用[D]. 南京航空航天大学, 2004. [↑](#endnote-ref-10)
10. [] Jaro M A. Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida[J]. Journal of the American Statistical Association, 1989, 84(406): 414-420. [↑](#endnote-ref-11)
11. [] Monge A, Elkan C. An efficient domain-independent algorithm for detecting approximately duplicate database records[J]. 1997. [↑](#endnote-ref-12)
12. [] Hernández M A, Stolfo S J. The merge/purge problem for large databases[C]//ACM Sigmod Record. ACM, 1995, 24(2): 127-138. [↑](#endnote-ref-13)
13. [] Hernández M A, Stolfo S J. Real-world data is dirty: Data cleansing and the merge/purge problem[J]. Data mining and knowledge discovery, 1998, 2(1): 9-37. [↑](#endnote-ref-14)
14. [] Forgy C L. OPS5 user's manual[R]. CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE, 1981. [↑](#endnote-ref-15)
15. [] Qiu Y F, Tian Z, Ji W, et al. An efficient approach for detecting approximately duplicate database records[J]. CHINESE JOURNAL OF COMPUTERS-CHINESE EDITION-, 2001, 24(1): 69-77. [↑](#endnote-ref-16)
16. [] Costa G, Manco G, Ortale R. An incremental clustering scheme for data de-duplication[J]. Data Mining and Knowledge Discovery, 2010, 20(1): 152. [↑](#endnote-ref-17)
17. [] Ferro A, Giugno R, Puglisi P L, et al. An efficient duplicate record detection using q-grams array inverted index[C]//International Conference on Data Warehousing and Knowledge Discovery. Springer, Berlin, Heidelberg, 2010: 309-323. [↑](#endnote-ref-18)
18. [] 郭志懋, 周傲英. 数据质量和数据清洗研究综述[D]. , 2002. [↑](#endnote-ref-19)
19. [] 邱越峰, 田增平, 季文贇, 等. 一种高效的检测相似重复记录的方法[J]. 计算机学报, 2001, 24(1): 69-77. [↑](#endnote-ref-20)
20. [] 陈伟, 王昊, 朱文明. 一种提高相似重复记录检测精度的方法[J]. 计算机应用与软件, 2006, 23(10): 29-30. [↑](#endnote-ref-21)
21. [] Hassanien A E, Azar A T, Snasel V, et al. Big Data in Complex Systems[M]. Springer, Heidelberg, 2015. [↑](#endnote-ref-22)
22. [] 靳丹, 张磊, 王洪军,等. 基于Hadoop的大数据清洗框架设计与应用[J]. 网络新媒体技术, 2015, 4(5):33-38. [↑](#endnote-ref-23)
23. [] Lillibridge M, Eshghi K, Bhagwat D. Improving restore speed for backup systems that use inline chunk-based deduplication[C]//FAST. 2013: 183-198. [↑](#endnote-ref-24)
24. [] Levenshtein V I. Binary codes capable of correcting deletions, insertions, and reversals[C]//Soviet physics doklady. 1966, 10(8): 707-710. [↑](#endnote-ref-25)
25. [] Hassanzadeh O, Miller R J. Creating probabilistic databases from duplicated data[J]. The VLDB Journal—The International Journal on Very Large Data Bases, 2009, 18(5): 1141-1166. [↑](#endnote-ref-26)
26. [] Jaro M A. UNIMATCH, a Record Linkage System: Users Manual[M]. Bureau of the Census, 1980. [↑](#endnote-ref-27)
27. [] Cohen W W. Integration of heterogeneous databases without common domains using queries based on textual similarity[C]//ACM SIGMOD Record. ACM, 1998, 27(2): 201-212. [↑](#endnote-ref-28)
28. [] Sabhnani M, Serpen G. Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context[C]//MLMTA. 2003: 209-215. [↑](#endnote-ref-29)
29. [] Chakradhar S T, Agrawal V D, Rothweiler S G. A transitive closure algorithm for test generation[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1993, 12(7):1015-1028. [↑](#endnote-ref-30)
30. [] Hylton J A. Identifying and merging related bibliographic records[J]. 1996. [↑](#endnote-ref-31)
31. [] 韩京宇, 徐立臻, 董逸生. 一种大数据量的相似记录检测方法[J]. 计算机研究与发展, 2005, 42(12): 2206-2212. [↑](#endnote-ref-32)
32. [] Kohonen T. An introduction to neural computing[J]. Neural Networks, 1988, 1(1):3-16. [↑](#endnote-ref-33)
33. [] McCulloch W S, Pitts W. A logical calculus of the ideas immanent in nervous activity[J]. The bulletin of mathematical biophysics, 1943, 5(4): 115-133. [↑](#endnote-ref-34)
34. [] Sutskever I, Martens J, Dahl G, et al. On the importance of initialization and momentum in deep learning[C]//International conference on machine learning. 2013: 1139-1147. [↑](#endnote-ref-35)
35. [] 俞荣华, 田增平, 周傲英. 一种检测多语言文本相似重复记录的综合方法[J]. 计算机科学, 2002, 29(1):118-121. [↑](#endnote-ref-36)
36. [] Mauricio A. Hernández, Stolfo S J. The merge/purge problem for large databases[C]// Acm Sigmod International Conference on Management of Data. ACM, 1995:127-138. [↑](#endnote-ref-37)
37. [] 李军. 一种相似重复记录检测算法的改进与应用[J]. 成都工业学院学报, 2017 (2017 年 02): 17-20. [↑](#endnote-ref-38)
38. [] 胡玄子, 陈小雪, 钱叶亮,等. 数据处理中缺失数据填充方法的研究[J]. 湖北工业大学学报, 2013, 28(5):82-84. [↑](#endnote-ref-39)
39. [] Christen P, Pudjijono A. Accurate synthetic generation of realistic personal information[C]//Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, Berlin, Heidelberg, 2009: 507-514. [↑](#endnote-ref-40)
40. [] Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators[J]. Neural networks, 1989, 2(5): 359-366. [↑](#endnote-ref-41)
41. [] Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators[J]. Neural networks, 1989, 2(5): 359-366. [↑](#endnote-ref-42)
42. [] Goldberg D E. Genetic Algorithm in Search Optimization and Machine Learning[J]. Addison Wesley, 1989, xiii(7):2104–2116. [↑](#endnote-ref-43)
43. [] Zhang Y, Gao X, Katayama S. Weld appearance prediction with BP neural network improved by genetic algorithm during disk laser welding[J]. Journal of Manufacturing Systems, 2015, 34: 53-59. [↑](#endnote-ref-44)
44. [] Asadi E, da Silva M G, Antunes C H, et al. Multi-objective optimization for building retrofit: A model using genetic algorithm and artificial neural network and an application[J]. Energy and Buildings, 2014, 81: 444-456. [↑](#endnote-ref-45)
45. [] Dauphin Y N, Pascanu R, Gulcehre C, et al. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization[C]//Advances in neural information processing systems. 2014: 2933-2941. [↑](#endnote-ref-46)
46. [] Piotrowski A P. Differential evolution algorithms applied to neural network training suffer from stagnation[J]. Applied Soft Computing, 2014, 21: 382-406. [↑](#endnote-ref-47)
47. [] Sutskever I, Martens J, Dahl G, et al. On the importance of initialization and momentum in deep learning[C]//International conference on machine learning. 2013: 1139-1147. [↑](#endnote-ref-48)