计算机科学
COMPUTER SCIENCE

面向多样化数据清洗任务的证据集智能选择方法

钱泽凯, 丁小欧, 孙哲, 王宏志, 张岩

引用本文

钱泽凯, 丁小欧, 孙哲, 王宏志, 张岩. 面向多样化数据清洗任务的证据集智能选择方法[J]. 计算机科学, 2024, 51(8): 124-132.

QIAN Zekai, DING Xiaoou, SUN Zhe, WANG Hongzhi, ZHANG Yan. Intelligent Evidence Set Selection Method for Diverse Data Cleaning Tasks [J]. Computer Science, 2024, 51(8): 124-132.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230900003

面向多样化数据清洗任务的证据集智能选择方法

钱泽凯　丁小欧　孙　哲　王宏志　张　岩
哈尔滨工业大学计算机科学与技术学院　哈尔滨 150006
(qzk010728@gmail.com)

摘　要　由于针对单一特定数据质量问题而设计的数据清洗算法并不总能有效地适用于多种清洗需求共存的数据质量提升技术,因此可采用多种清洗方法互相配合的方式来解决各种数据清洗需求。将数据清洗问题转换为证据集的生成和选择问题,基于聚合查询的增量式质量评估方案和基于中间算子证据集的算子结果选择方案,在多种清洗任务下实现了多种清洗方法配合的高效数据清洗。在所提清洗模型中,算子库提供数据清洗结果并将其转换为中间算子;中游的采样器将中间算子集分流和剪枝,给搜索器提供优质的候选证据集;下游的搜索器在质量评估器的指导下进行证据集的选择,搜索完毕后向上游算子库更新数据和必要的参数,使算子库重新迭代生成中间算子。最后,基于3个不同规模的真实数据集进行了大量实验,通过不同数据清洗任务下的性能验证在任意种类的数据清洗需求下算子编排的可行性,并将所提方法和现有的智能数据清洗系统进行性能对比。结果表明,在多种清洗任务中,所提方法在多种数据质量约束、动态和大规模的数据清洗方面具有稳定的准确率和召回率,且同一清洗时间下异常值、规则违反和混合错误的清洗任务性能优于其他智能数据清洗系统15%以上。

关键词:数据清洗;数据质量评估;流水线系统设计;算子选择;证据集
中图分类号　TP311

Intelligent Evidence Set Selection Method for Diverse Data Cleaning Tasks

QIAN Zekai,DING Xiaoou,SUN Zhe,WANG Hongzhi and ZHANG Yan
College of Computer Science and Technology,Harbin Institute of Technology,Harbin 150006,China

Abstract　Due to the limitations of data cleaning algorithms designed specifically for individual data quality issues and their inability to effectively address multiple coexisting data quality enhancement requirements,a collaborative approach employing multiple data cleaning methods can be adopted to fulfill various data cleaning needs. This paper formulates the data cleaning problem as a task of evidence set generation and selection. By utilizing an incremental quality assessment scheme based on aggregate queries and an operator result selection scheme based on intermediate operator evidence sets,efficient data cleaning involving a combination of diverse cleaning methods is achieved across various cleaning tasks. In the proposed cleaning model,the operator repository yields data cleaning results and transforms them into intermediate operators. The sampler in the midstream module distributes and prunes the set of intermediate operators to provide the searcher with a high-quality candidate evidence set. The downstream searcher,guided by the quality evaluator,selects evidence sets. Upon completion of the search process,the upstream operator repository updates data and necessary parameters,facilitating the reiteration of intermediate operator generation. Finally,extensive experiments are conducted on three real-world datasets of varying scales. Performance verification across different data cleaning tasks demonstrates the feasibility of operator orchestration for any type of data cleaning requirement,underpinning the proposed method's stable precision and recall in scenarios involving diverse data quality constraints,dynamics,and large-scale data cleaning. Furthermore,a performance comparison with existing intelligent data cleaning systems reveals that the proposed method outperforms these systems by over 15% in tasks related to outlier detection,rule violations,and mixed errors,all within the same cleaning time.

Keywords　Data cleaning,Data quality assessment,Pipeline system design,Operator selection,Evidence set

到稿日期:2023-09-01　返修日期:2023-12-06
基金项目:国家重点研发计划(2021YFB3300502);国家自然科学基金(62232005,62202126);中国博士后科学基金(2022M720957);黑龙江省博士后资助项目(LBH-Z21137)
This work was supported by the National Key Research and Development Program of China(2021YFB3300502),National Natural Science Foundation of China(62232005,62202126),China Postdoctoral Science Foundation(2022M720957) and Heilongjiang Postdoctoral Financial Assistance Program(LBH-Z21137).
通信作者:王宏志(wangzh@hit.edu.cn)

1 引言

数据清洗是目前数据分析工作的主要步骤。脏数据的修复工作占用了数据科学家全部工作的60%[1]，因此实现一种自动化的数据质量提升框架在当前的数据分析工作中尤为重要[2]。总的来说，数据清洗可以分为数据质量评估和错误修复两部分。为了实现这两项任务的自动化，学术界和工业界已经分别针对这些数据质量问题[3]提出了众多清洗方法[4]。但目前的清洗技术仍然存在以下问题：

1）在复杂的数据面前，一种针对特定质量问题设计的数据清洗框架在单独使用时的准确率和召回率还不够稳定[5]；

2）在大规模多种属性类型的脏数据下，并不总能迅速准确地评估任意定制规则的数据修复效果[6]；

3）不同的数据清洗信号同时修复数据时出现修复冲突和效率低下的问题[7]，很难进行深度的配合。

为了能得到一种在多样数据集下准确率和召回率稳定的清洗策略，将数据清洗问题转换为中间算子证据集的生成和选择问题。本文的贡献在于：

1）面向大规模多种属性类型的数据集，设计了一种基于聚合查询的增量式质量评估算法（AQ算法），其能够高效地评估指定规则集下的修复效果。

2）设计了一种具有通用性和可解释性的清洗方案，将清洗问题转换为证据集的选择问题。针对不同的清洗任务和数据集，生成具有稳定准确率和召回率的可解释修复方案。

3）设计了一系列证据集算法，有效解决了数据清洗算子之间配合困难以及不同算子执行结果之间的冗余和冲突问题。在清洗时间相同的情况下，相比Holoclean系统[1]，本文方法在处理空缺值、规则违反及混合错误等清洗任务上的准确率和召回率提升了15%以上。

本文第2章介绍和评析了本文的相关工作；第3章给出了研究问题的定义并总述了研究方法；第4章介绍了基于聚合查询的增量式质量评估器策略；第5章介绍了基于中间算子证据集的算子选择策略；第6章在实际的3个数据集上对已有方法进行了测试，并介绍了实验结果；最后总结全文。

2 相关工作

2.1 数据质量评估

数据清洗是将低质量数据修正为高质量数据的过程。数据清洗系统的数据修复效果由质量评估函数表示，该函数通过对修复后数据的质量进行评估来表征修复的合理性[8]。

在实际的数据清洗任务中，数据集往往是动态、大规模且具有多种属性类型的[5]。一个可靠的数据质量评估方案，需要保证在不同约束条件和不同数据类型下没有倾向性地评估修复策略是否合理[9]。

在基于规则的评估方面，Alphaclean数据清洗系统[10]是基于条件依赖这一种约束类型，来评估数据对条件依赖违反的情况。然而，它并没有考虑到在其他约束类型（例如外部字典匹配约束、否定函数依赖等）同时存在时，如何平等地对不同规则的违反情况进行评估。

在基于属性种类的评估方面，Boostclean数据清洗系统[11]主要针对阈值完整性错误对数值型的数据进行错误修复。但是，当数据集中多种属性都有错误（如数值型和字符型的数据同时存在错误）时，其并不能很好地处理字符型错误。

在评估大数据和动态数据的质量时，需要增量式地评估策略。例如，Ilyas等提出的一个迭代式数据质量评估框架，可根据实时的用户标注增量式地评估数据质量[5]。同时，Alphaclean系统[10]也可以增量地在流水线中实时评估修复的质量。

2.2 数据清洗方法

目前已经开发出多种自动化的数据清洗框架，且其可以进行自动的数据错误修复。但是，由于数据清洗任务具有复杂性，这些框架往往是在某一个特定的角度考虑修复的策略与设计相应的框架。

有些框架针对某一种常见错误类型进行修复设计。Abedjan等最近在5个专有的真实世界数据集上评估了一系列的错误检测技术[12]。他们发现，很大一部分数据错误可以被归类为阈值完整性错误。Krishnan等针对这样的错误，设计实现了Boostclean清洗系统[11]，其对阈值完整性错误的修复有很好的效果。

还有一些框架主要是对多错误混合下的数据进行清洗，但是侧重点和优先级有所不同，包括针对连续数据错误的一致性约束[13]和针对语义信息错误的模式函数依赖[14]等。这一类的框架在很多实际的数据集上都有着比较不错的效果。例如，Krishnan等针对更有可能出错的数据优先采样[15]，利用交互式的数据清洗让人参与数据清洗的关键节点，根据数据对模型结果的影响程度来优先清洗那些记录。

事实上，仅采用一种专门针对特定质量问题而设计的数据清洗框架来处理任意清洗任务，往往会导致准确率和召回率的不稳定性[5]。因此，研究人员的重点逐渐转变为恰当地集成现有清洗方法产生的结果，以便更全面地清洗数据。这类集成方法的清洗框架通常依赖于多种自动化数据清洗方法，通过灵活地配置、选择和排序算子库中的不同方法，对目标数据进行有效的清洗。

在这方面，Alphaclean数据清洗系统[10]采用了一种“先生成后搜索”的策略，并设计了一个通用的数据清洗管道。在这个管道中，多种清洗算子按照特定的框架进行编排。这种方法通过结合各种面向任务的清洗技术来实现数据清洗，并进一步优化了整个清洗流程。然而，这些面向任务的清洗方法往往较为复杂，不易于进行更精细化的优化。此外，EasyDR[16]采用了相似的思想，并在可视化工作上做出了大量优化，为用户提供不同清洗任务的先后顺序以及可选择的清洗任务类型。

同时，Rekatsinas等从带有噪声数据集的概率因子图模型的角度出发[1]，设计并实现了Holoclean数据清洗系统[1]。Holoclean是一个弱监督的机器学习系统，它利用质量规则、值的相关性、参考数据以及其他多种修复信号来构建一个能够精确描述数据生成过程的概率模型。通过这个概率因子图，系统可以对各种修复信号进行有效的编排。然而，Holoclean在构建概率因子图时未能优化清洗效率，在大数据下清洗效率不够高。在条件函数依赖修复相关的工作上，Horizoh

数据清洗系统[18]致力于在数据集上构建模式图，以解决修复的迭代冲突问题，并在大型数据集上取得了最高的效率。但是，Horizon 的研究仅限于针对普通函数依赖错误的修复。Baran[19]和 Garf[20]系统利用神经网络模型对清洗操作进行学习和预测。其中，Baran 系统首先将元学习应用于整体数据清洗中，但其仍然依赖于用户进行大量手动标注，且强调无规则配置状态下的数据清洗。Garf 系统使用序列生成对抗网络进行自我监督和可解释的数据清理，但其对个体单元格的概率推理使得模型训练过程过于耗时。

综上，以往的研究中，质量评估参考的因素过于单一。可靠的评估方案应当是任意属性类型上的数据违反任意种类的约束时都需要记录相同的代价，从而保证对修复策略的评估是没有倾向性的。在数据清洗算法编排方面，不同的数据清洗方法同时修复数据时还是会出现修复冲突和效率低下的问题[7]，很难进行深度的配合。

3 本文研究内容

本文旨在研究实现一种大规模、动态且具有多种属性类型的数据中的数据清洗方法。通过将数据清洗问题转换为证据集的生成和选择问题，将多种清洗信号的修复结果进行选择和决策。下面通过实例来说明本文的研究。

表 1 列出了示例数据的初始信息；表 2 列出了该数据对应的条件依赖约束，包括 $c1$，$c2$，$c3$ 和 $c4$ 在内的 3 项条件依赖规则。通过表中信息可以发现实例数据包括 4 类错误：元组 $t4$ 的 DBAName 属性不满足名称数据的格式，存在异常值的错误；元组 $t4$ 的 City 属性由表 2 的条件依赖发现存在违反规则 $c3$ 的错误；元组 $t1$ 和元组 $t2$ 数据由条件依赖发现存在违反规则 $c1$ 的错误，且元组 $t1$ 和元组 $t3$ 也同理存在违反 $c1$ 规则的错误；最后 $t3$ 元组的 AKAName 存在违反 c4 规则的错误。表 1 中，相关错误的单元格用下划线表示。

表 1 示例数据初始信息

Table 1 Initial information of sample data

	DBAName	AKAName	City	Zip
t1	John Veliotis Sr.	Johnnyo's	Chicago	60607
t2	John Veliotis Sr.	Johnnyo's	Chicago	60609
t3	John Veliotis Sr.	Helen's	Chicago	60609
t4	Johnnyo's	Johnnyo's	Cicago	60608

表 2 条件依赖示例

Table 2 Conditional dependency example

	Functional Dependencies
$c1$	DBAName→Zip
$c2$	Zip→City
$c3$	City→Zip
$c4$	DBAName, City→AKAName

利用表 2 的规则信息，可以设计多种最小规则违反的清洗信号，即修复尽量少的数据单元格，使得数据不违法条件依赖的规则，满足一致性。这种修复信号可以快速修复一些不满足条件依赖的元组。

另一种修复信号是通过匹配依赖规则进行修复，它包括依赖的规则和对应的外部字典信息，示例内容如表 3 所列。匹配依赖的规则部分是条件依赖的一个子集，与条件依赖不同的是它有其对应的外部信息进行匹配。这样在进行基于匹配依赖的修复时，可以参考外部信息进行基于匹配函数依赖的清洗。相比最小规则违反的修复方式，这种修复策略更加准确，但是要求更多的外部信息和更多的时间进行匹配。

表 3 匹配依赖示例

Table 3 Match dependency example

id	Matching Dependencies
m1	Zip=Ext. Zip→ City=Ext. City [Ext_City=Chicago &Ext_Zip=60608]

还有一种常用的清洗信号是基于数据统计特征的清洗，例如元组 t4 的 DBAName 属性不满足与其他元组一样的格式，属于一种异常值错误。这种情况就可以通过定义一些正则规则进行检测，并参考其他的数据统计信息进行修复。

分别采用以上基于最小规则违反的修复信号、基于匹配依赖的修复信号和基于统计特征的修复信号对表 1 中的数据进行数据清洗，得到的结果如表 4 所列。其中，正体且加粗的部分是用最小规则违反的清洗方法修复的数据；斜体不加粗的部分是用匹配依赖修复的数据；斜体且加粗的部分是统计特征修复的数据；同时用下划线的部分是仍然存在不一致的信息。

可以看出，最小规则违反的修复方法可以修复 t3 元组的 AKAName 属性，但是对 Zip 属性添加了错误的修复；基于匹配依赖的修复方法可以修复好属性 Zip 上的数据；基于统计特征的修复方法则可以修复好 DBAName 上不符合统计特征的数据。总体而言，独立来看 3 种修复信号在修复一部分不一致数据的同时，也可能会添加少量错误。对于 t1 元组的 Zip 属性，基于最小规则违反的修复和基于匹配依赖的修复存在冲突，则可以通过质量评估选择更加合理的策略。

表 4 基于 3 种算子结果编排的修复

Table 4 Repair based on three operator result arrangements

	DBAName	AKAName	City	Zip
t1	John Veliotis Sr.	Johnnyo's	Chicago	**60609**或 *60608*
t2	John Veliotis Sr.	Johnnyo's	Chicago	*60608*
t3	John Veliotis Sr.	**Johnnyo's**	Chicago	*60608*
t4	***John Veliotis Sr***	Johnnyo's	Chicago	*60608*

综上所述，可以通过将多种清洗信号不冲突的修复结果进行修复，将冲突的内容在基于质量评估的决策下进行选择，设计出基于算子结果证据集选择的数据清洗系统来统一各种清洗信号的结果，最终将数据清洗问题转换为清洗证据集的生成和选择问题。

3.1 基本定义

定义 1(中间算子) 设 D 为原始数据集，D' 为经过中间算子修复后的数据集。一个中间算子 M 被定义为一个函数 $M: D \to D'$，其中 M 接受原始数据集 D 作为输入，并返回修复后的数据集 D'。M 可以形式化为三元组参数的形式：$M=\{(c,p,v) \mid c$ 是属性列；p 是谓词，v 是属性值$\}$。其中，$(column, predicate, value)=(c,p,v)$ 表示将列 c 和谓词 p 匹配的单元格替换为属性值 v 的修复操作；$(column, predicate)=(c,p,\_)$ 表示删除属性列 c 和谓词 p 匹配的单元格中的内容的修复操作；$Insert(column, value)=(c,\_,v)$ 表示在属性列 c

上添加内容为属性值 v 的单元格的修复操作。中间算子作为系统最基本的通用修复语言在清洗系统中实现，可方便地在各个模块之间传递并在搜索器中进行优化。

定义 2（算子库与数据清洗信号） 算子库 O 是一个集合，表示为 $O=\{O_1,O_2,\cdots,O_m\}$，其中 $O_1,O_2,\cdots,O_m$ 是数据清洗信号。每个数据清洗信号 O_i 都是一个函数，它接受原始数据集 D 作为输入，并产生对数据 D 的清洗结果。具体地，对于每个数据清洗信号 O_i，可以定义如下函数：$O_i:D\rightarrow D_i$。其中，D_i 是通过算子 O_i 对数据集 D 进行清洗后得到的数据集。

定义 3（采样器） 采样器 $Sample$ 是一个函数，用于将产生的修复情况统一包装成中间算子的形式，并抽取合适的中间算子作为证据集。可以定义采样器为以下函数：$Sample:(D'-D)\rightarrow\{M_1,M_2,\cdots,M_m\}$，其中 $(D'-D)$ 是原始数据集 D 和修复后的数据集 D' 之间的差异，$\{M_1,M_2,\cdots,M_m\}$ 是采样器产生的中间算子证据集。

定义 4（质量评估器） 设 Q 为质量评估器。对于已知的约束信号 C，当前数据 D 偏离清洁数据的程度定义为 $Distance(D,C)$；当前数据 D 偏离初始数据 A 的程度定义为 $Distance(D,A)$；质量评估器 Q 可表示为 $Q(D)=w_1 Distance(D,C)+w_2 Distance(D,A))$，其中 w_1 和 w_2 是权重，用于平衡对清洁数据和初始数据的重视程度。质量评估器在系统的每个环节起着关键作用，可以通过设计质量评估函数系统，根据其计算结果来评估数据的质量，并在流水线的各个模块中进行优化以提升数据质量。

定义 5（搜索器与清洗问题的优化目标） 设 S 为搜索器，已知数据集 D、采样器 $Sample$ 产生的中间算子证据集 $I=\{M_{i_1},M_{i_2},\cdots,M_{i_m}\}$ 以及质量评估器 Q、搜索器 S 可以表示为 $S(I,D)=J$，其中 $J=\{M_{j_1},M_{j_2},\cdots,M_{j_n}\}\subset I$，使得 $\{j_1,j_2,\cdots,j_m\}=\underset{\{i_1,i_2,\cdots,i_m\}}{\operatorname{argmax}}Q(M_{j_m}\circ M_{j_{m-1}}\circ\cdots\circ M_1(D))$。其中，集合 $\{j_1,j_2,\cdots,j_m\}$ 表示最优的中间算子集，也是本文清洗问题的目标结果序列。

定义 6（数据清洗的分块） 将清洗管道 K 划分为 $\{k_1,k_2,\cdots,k_m\}=K$ 的若干块小清洗流程块，使得块内的修复不影响其他块的修复。这样的划分方式，可以提高清洗的效率。

3.2 方法概述

本文提出的修复方案是流水线模式的，在流水线中采用“先生成后选择”的策略：上游模块负责生成已有数据清洗信号的结果，中游模块负责将结果提取为统一的中间算子构成证据集，下游模块则负责选择合适的证据集并对上游进行反馈式的调优。所提方法的整体设计如图 1 所示。下文结合 3.1 节的相关定义进行说明。

第一步，从数据库中读入脏数据和一些必要的参数（例如数据的约束规则等），将脏数据送入算子库中，算子库中的多种清洗信号 $O=\{O_1,O_2,\cdots,O_m\}$ 分别产生对应的清洗结果。第二步，采样器在质量评估器的指导下，将清洗结果转换为中间算子构成证据集 $I=\{M_1,M_2,\cdots,M_m\}$ 并送往搜索器。第三步，搜索器将采样得到的中间算子证据集加入目前的搜索序列中，并在质量评估器的指导下选择能让数据质量提高的中间算子集 $\{M_{j_1},M_{j_2},\cdots,M_{j_n}\}\subset I$。第四步，当完成当前阶段的搜索后，向算子库更新数据和必要的参数，算子库中的清洗信号重新利用新数据和更新后的参数继续产生中间算子，重复第一步到第三步的过程。系统渐进地将数据质量优化，最终得到最优质的算子操作序列。

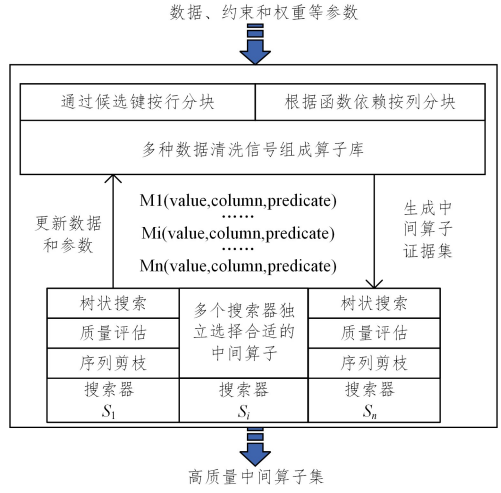


图 1 系统模型设计

Fig. 1 Overview of system model design

4 多样清洗任务下的质量评估体系

如 2.1 节所述，在质量评估器效率方面，质量评估器需要设计一个增量式的算子序列质量评估算法来满足数据和系统各个模块动态运行的特点。具体来说，当输入数据集更新时，可以仅计算新修复的元组的清洗质量，而不需要重新计算整个数据集的清洗质量。若用 Q 表示质量评估的结果，ops 表示一个中间算子序列，ops_prev 表示当前获取的中间算子序列，new_op 表示新加入的中间算子，增量式的质量评估函数可以递归表示为 $Q(data,ops)=Q'(Q(data,ops\_prev),new\_op)$。

在质量评估器准确性方面，由定义 4 可知质量评估器的结果主要涉及 $Distance$ 函数的计算。已知当前数据 D 和初始数据 A 的修复惩罚 $Distance(D,A)$ 可以由多种相似度的方法（如 Jaccard 相似度、Levenshtein 编辑距离等[21]）进行度量；但是对于已知的约束信号 C，计算当前数据 D 偏离未知的清洁数据的程度 $Distance(D,C)$，则需要在多种约束的情况下，找到一种能在多属性下统一度量距离的方式。本文使用聚合查询的方式，实现一个增量式的质量评估方法。

4.1 基于聚合查询的增量式质量评估策略

数据约束信号 C 第 i 个约束信号 C_i 下的数据 D 距离清洁数据的程度分量可以表示为聚合查询函数 $Count(D,C_i)$ 的形式。初始条件下，当前数据 D 偏离清洁数据的程度 $Distance(D,C)=\sum_{i=1}^{|C|}Count(D,C_i)$。进行修复后，$r_{pred}$ 表示修改之前 D 上匹配谓词的数据单元集合，r'_{pred} 表示修改之后 D' 上匹配的数据单元集合，则：

$$Distance(D',C)=Distance(D,C)+\sum_{i=1}^{|C|}(Count(D\bowtie r'_{pred},C_i)-Count(D\bowtie r_{pred},C_i))$$

基于聚合查询的增量式质量评估策略(AQ算法)采用算法1进行计算。

算法1 基于聚合查询的增量式质量评估(AQ算法)

输入:数据D,约束信号C,修改前后新增的元组r_{pred},r'_{pred}

输出:数据D偏离清洁数据的程度Distance

```
1. IF run in first time: # 初始化
2.    Distance←0
3. FOR i=1 TO |C| DO:
4.       Distance+=Count(D,Ci)
5. ELSE:
6.    FOR i=1 to |C| DO:
7.          delta=Count(D ⋈ r'pred,Ci)−Count(D ⋈ rpred,Ci)
8.          Distance+=delta
9. RETUTN Distance
```

4.2 多样清洗任务下的聚合查询函数设计

针对各种清洗任务的约束信号C下数据偏离干净数据的程度,可以采用聚合查询的方法进行计算。该方法的目标是找到不满足一致性的元组个数,用不满足一致性元组的个数来表示距离清洁数据的距离,这样设计的优势是能公平地评估不同类型属性之间的距离。下面分别针对规则违反、异常值和空值错误的约束对聚合查询函数进行设计。

4.2.1 针对规则违反错误的聚合查询方法

采用否定依赖[22]作为规则违反的约束形式,对于数据D,若M表示约束目标的属性集合,N表示约束具体的逻辑内容,则针对规则违反错误的聚合查询语句为:SELECT COUNT(*) FROM D GROUP BY M HAVING NOT(N)。

可以看出,如果数据D所有的元组满足规则的约束条件,则结果应该为0,如此计算可以找到规则违反的数据对数,从而达到评估规则违反下当前数据与清洁数据距离的目的。

4.2.2 针对异常值错误的聚合查询方法

有一种基于聚类的异常值检测方法[23],其假设正常的数据对象属于大的和密集的聚类;而异常值属于小的或稀疏的聚类,或者不属于任何聚类。对于数据D,若$\{M_1, M_2, \cdots, M_m\}$表示$m$列的属性集合,定义一个聚类规模阈值$k$,针对异常值错误的聚合查询语句为:SELECT SUM(cnt) FROM (SELECT COUNT(*) AS cnt FROM D GROUP BY M_i HAVINGCOUNT(*)<k) AS sub。

可以看出,若将聚类规模小于k的聚类视为大概率为异常值的聚类,聚合查询的结果表示属性M_i下是异常值的元组种类,以此描述该约束下数据D和清洁数据的距离。

4.2.3 针对空缺值错误的聚合查询方法

对于数据D,若$\{M_1, M_2, \cdots, M_m\}$表示$m$列的属性集合,定义一个聚类规模阈值$k$,针对空值错误的查询语句为:SELECT COUNT(*) FROM D GROUP BY M_i HAVINGM is NULL。

可以看出,聚合查询的结果表示属性M_i下是空值的元组种类,以空值的个数描述该约束下数据D和清洁数据的距离。

5 基于中间算子证据集的选择策略

如3.2节所述,系统将修复结果规范化为统一的中间算子形式,并将数据清洗问题转化为基于中间算子证据集的生成和选择问题。下面对中间算子证据集的生成与选择策略进行说明。

5.1 中间算子证据集的生成与分块策略

中间算子证据集由算子库产生,算子库中的每一个数据清洗信号通过读入脏数据,最终通过采样器生成中间算子证据集。

由于算子库在处理大规模数据时需要消耗大量时间,并且在生成第一批中间算子期间,流水线下游的搜索器无法进行操作,因此面对大数据量时应采用分块策略,将数据分为独立的块,然后对每个块应用独立的清洗规则。根据定义6,可以从规则和数据两个维度来设计分块策略。

在规则分块方面,可以将约束目标属性集合不重叠的规则分为不同的块。在数据分块方面,考虑到候选键的主要功能是唯一标识一个元组,它不会受到条件依赖违反错误的影响,可以先对候选键中的异常值和空值进行预清洗,然后选择一组清洁的候选键进行分组。采用这种方法,利用候选键进行的分块可以确保每个数据块内的修复操作是独立的。

综上所述,结合3.1节的定义,算法2给出了中间算子证据集的具体生成策略(GOS算法)。

算法2 生成中间算子证据集(GOS算法)

输入:算子库O,数据集D,采样器Sample

输出:初始中间算子证据集M

```
1. M←{}
2. D_block=Block(D) # 按照规则和候选键分块
3. FOR op IN O DO:
4.    FOR d IN D_block DO:
5. M. append(Sample(op(d))
6. RETURN M
```

5.2 采样器对证据集的分流剪枝策略

采样器在数据清洗流程中起到关键作用,它直接决定了中间算子池输出到搜索器的顺序。通过采样器,系统可以对中间算子证据集进行分流和剪枝,从而优化搜索器的搜索空间。可以看出,通过预先剔除错误的中间算子,并且分析出冲突的中间算子,可以实现搜索的预处理。结合3.1节的定义,下面通过数据实例进行说明。

以表1为例,第3章详细介绍了算子库如何结合各种修复方法生成表4的修改。基于这些清洗结果,采样器生成了如表5所列的初始中间算子证据集。需要注意的是,尽管采样器进行了分流,搜索空间的快速扩大仍然使得寻找最优中间算子集的代价大增,导致搜索问题变得难以求解。因此,当处理大数据时,进一步的剪枝变得尤为重要。

接下来,采样器可以根据属性值对证据集进行分流,将其划分为冲突和非冲突的中间算子。非冲突的中间算子可以直接用于数据清洗;而冲突的中间算子则需要送往搜索器,作为其搜索空间进行算子选择。这一分流结果如表6所列。

表 5 中间算子证据集构成的搜索空间

Table 5 Search space for intermediate operator evidence set

	中间算子	来源算子类型
o1	(‘Zip’,‘60609’,{‘Zip’,’60607’})	最小规则违反
o2	(‘AKAName’,‘Johnnyo’s’,{‘AKAName’,‘Helen’s’})	最小规则违反
o3	(‘city’,‘Chicago’,{‘city’,‘Cicago’})	匹配依赖
o4	(‘Zip’,‘ 60608’,{‘Zip’,‘60609’})	匹配依赖
o5	(‘Zip’,‘ 60608’,{‘Zip’,‘60607’})	匹配依赖
o6	(‘DBAName’,‘John Veliotis Sr.’{‘DBAName’,‘Johnnyo’s’})	基于统计特征

表 6 中间算子集的分流

Table 6 Distribution of middle operator set

修复的属性	中间算子
AKAName	o2
DBAName	o6
City	o3
Zip	o1,o4,o5

为了解决这一问题，使用外部信息进行预剪枝。具体地说，将预先提供的权威外部字典与待清洗的数据集进行匹配，利用字典中的客观事实来确定数据集中的正确值，从而剪除一些不合理的中间算子。例如，结合表 3 中的匹配依赖外部信息，可以发现初始数据中的某些属性值是错误的，这为搜索器提供了剪枝的依据。

综上所述，本文提出了一种对中间算子证据集进行分流剪枝的策略，即 OSP 算法，具体流程如算法 3 所示。

算法 3 中间算子证据集分流剪枝(OSP 算法)

输入：外部信息字典 I，初始中间算子证据集 M

输出：分流后的中间算子集 N

1. M_1←{}，M_2←{}
2. FOR m IN M DO：
3. IF m NOT IN I：
4. CONTINUE
5. IF m.attribute IN M_1.attribute：
6. M_2..append(m)
7. ELSE：
8. M_1..append(m)
9. N=[M_1, M_2]
10. RETURN N

5.3 搜索器选择和更新优化策略

Alphaclean 清洗系统的树搜索方法[10]，将数据集放入各个清洗方法中以获得所有中间算子的证据集，并在每轮迭代评估中选择质量评估器返回的质量最高的中间算子且将其他中间算子全部作废，需要更新数据集并将其输入算子库中重新生成更新的中间算子证据集。但是过于频繁的迭代往往会产生与上一轮相同的中间算子，以上这种方式需要不断调用算子库更新中间算子，造成很大的浪费。通常情况下，只需第一步的算子库输出就可以涵盖大部分高质量的修复，无需不断调用算子库来更新中间算子证据集。可以在搜索深度达到一定深度或现存的证据集被搜索完时，再通过迭代发现是否存在新的证据集。

搜索器通过选择质量最高的中间算子证据集，来完成本阶段的搜索。搜索完成后，需要向算子库更新参数进行下一轮的数据清洗。基于证据集选择的搜索算法(ES 算法)如算法 4 所示。

算法 4 基于证据集选择的搜索(ES 算法)

输入：质量评估器 Q，数据集 D，分流后的中间算子集 N

输出：清洗后的最优数据 D′

1. M←{}
2. D=N[0](D) ＃对数据执行非冲突的中间算子
3. FOR m IN N[1] do：＃对冲突的中间算子进行搜索
4. D′=m(D)
5. IF Q(D′−D)<0：＃修复后的质量更高
6. D=D′
7. RETURN D

结合算法 1—算法 4，图 2 给出了基于中间算子证据集的算子选择策略的清洗流程。更新迭代时搜索无需反复调用算子库，通过直接调用一次算子库生成大量有效的中间表示，确定搜索空间。每次搜索选择并删除冲突项后，若达到合适的搜索深度或在搜索空间内搜索完毕后，再更新参数迭代算子库产生有效的中间算子。

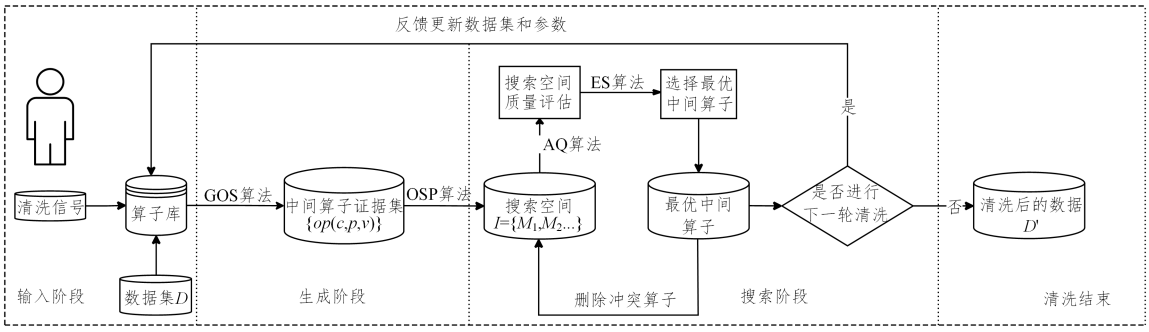


图 2 清洗方案流程

Fig. 2 Cleaning scheme flow

6 实验与结果

本实验中的算子库采用包含基于最小条件依赖违反的修复方法、基于匹配依赖的匹配修复方法和基于统计特征的修复算法作为算子库中基本的清洗信号。

6.1 数据集

为了评估系统的数据清洗能力，本文使用了 3 个真实世界的数据集：Holoclean[1] 中使用的 Hospital 数据集，以及 Flight 数据集[24] 和 Tax 数据集[25]。这些数据集覆盖了各种数据规模，最大规模包含 200 000 条数据记录、15 个属性和 32 条约束条件。最后，在 Hospital 数据集上将本系统与 Holoclean 清洗系统[1] 进行了性能对比。

1) Hospital 数据集[1]：来自美国卫生署，包含 1 000 条关于医院信息的记录，是一种数据清洗领域常见的数据集。

Hospital基本数据集具有1 000条数据、19条属性，以及20条以上的相关条件依赖和模式匹配，具有较好的代表性。

2)Flight数据集[24]：包含不同数据源在网上报告的航班起飞和到达时间的信息，是一组真实航班数据集。其包含2376条航班的起降信息，共2376条数据，且每条数据包含以航班号为主键的7条属性记录，主要用于验证模式匹配错误以及简单函数依赖错误。

3)Tax数据集[25]：一个常用于检测数据冲突的数据集，可以基于此扩大数据规模。该数据集包含200 000条数据记录，每条记录包含15个属性，用于储存个人的地址和纳税信息。该数据集上存在多个函数依赖和相关的模式匹配。

6.2 实验设计

6.2.1 实验对照策略

实验将不同的数据集拆分为不同的数据规模进行测试，分别进行5次重复清洗后取结果的平均值。一方面，使用单一错误下的数据和多错误混合的数据进行数据清洗，对比本系统在处理不同错误情况下数据的清洗能力；另一方面，对比本流水线系统和其他智能数据清洗系统(如Holoclean系统[1])在混合错误下的性能。

6.2.2 不同清洗任务的噪声注入

实验不仅覆盖了不同规模的数据集，还涵盖了多种数据清洗任务，如空缺值填补、异常错误修复、规则违反修复和混合错误修复等，以证明本文方法在多种场景下的有效性。为了向纯净数据集注入各种错误噪声来模拟不同的清洗任务，本文使用BART框架[25]在3个无缺失的清洁数据集上进行实验。本文实现了4种不同的噪声插入方式对错误分别进行不同比例的噪声注入，如表7所列。

表7 数据噪声注入方式

Table 7 Data noise injection methods

噪声种类	处理算法
异常值	通过随机插入字符、随机交换字符、随机删除字符、交换邻域值进行噪声的注入
空值	通过设置错误百分比，随机选择置空
规则违反	设置不同约束下属性的错误率
混合错误	随机采用上面3种方式

6.3 度量标准

本文设计的实验指标包括数据清洗的准确率和时间，最终实验记录中用准确率和召回率的调和平均值F1指标和时间来表征清洗系统的性能，具体的内容和计算策略如表8所列。

表8 实验设计指标

Table 8 Experimental design indexes

实验指标	计算方法
准确率(P)	正确修改的错误数与修复总数之比
召回率(R)	正确修改的错误数与错误数之比
F1指标(F)	准确率与召回率的调和平均值
时间(T)	清洗所需时间

6.4 本文方法的有效性分析

在不同的清洗任务下，验证5.1节的分块策略先对一组候选键进行清洗后再进行数据清洗的效果，实验结果如表9、表10所列。在不同规模(Data Size)的不同清洗任务(Error Type)下，数据清洗系统的性能如图3(a)和图3(b)所示。在Hospital数据集上，系统保证至少完成一轮清洗的前提下尽量控制与Holoclean系统[1]相同的时间进行清洗，对比的数据清洗性能的结果如图4所示。下文对本文方法的有效性进行分析。

表9 不进行预计清洗时不同清洗任务下的系统性能

Table 9 System performance for different cleaning tasks without pre-cleaning

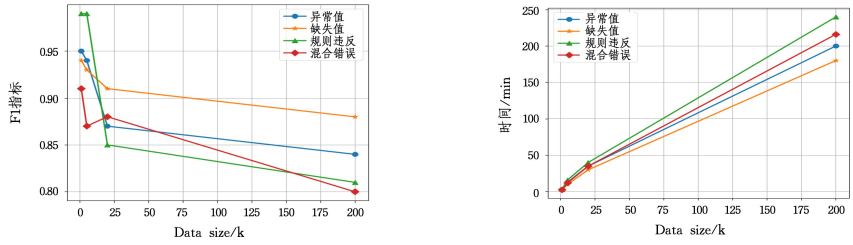
清洗任务	数据集异常值				缺失值				规则违反				混合错误			
	P	*R*	*F*	*T*/s	*P*	*R*	*F*	*T*/s	*P*	*R*	*F*	*T*/s	*P*	*R*	*F*	*T*/s
Hospital	0.83	0.68	0.75	340	0.73	0.59	0.65	382	0.35	0.11	0.17	281	0.74	0.51	0.60	692
Flight	0.85	0.72	0.78	598	0.44	0.66	0.53	957	0.25	0.05	0.08	651	0.71	0.74	0.72	988
Tax_20k	0.81	0.65	0.72	5931	0.41	0.60	0.49	9473	0.15	0.05	0.08	6831	0.68	0.61	0.64	9797

注：Tax数据集在20万规模的情况下清洗时间过长视为清洗失败，因此记录2万规模下的测试数据。

表10 对候选键预清洗后不同清洗任务下的系统性能

Table 10 System performance for different cleaning tasks after pre-cleaning candidate keys

清洗任务	数据集异常值				缺失值				规则违反				混合错误			
	P	*R*	*F*	*T*/s	*P*	*R*	*F*	*T*/s	*P*	*R*	*F*	*T*/s	*P*	*R*	*F*	*T*/s
Hospital	1.00	0.81	0.90	181	1.0	0.85	0.92	167	0.87	0.91	0.89	188	0.92	0.83	0.87	157
Flight	1.00	1.00	1.00	531	1.0	0.89	0.94	336	1.00	1.00	1.00	493	1.00	1.00	1.00	334
Tax	0.85	0.83	0.84	11839	0.91	0.85	0.88	10927	0.89	0.88	0.89	13927	0.82	0.79	0.80	12960



(a)在不同清洗任务下的F1指标 (b)在不同清洗任务下的运行时间

图3 总体实验结果

Fig. 3 Overall experimental results

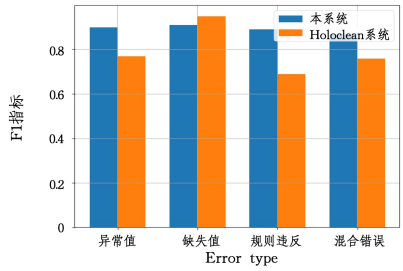


图 4 相同时间下与 HoloClean 系统在不同清洗任务下的性能对比

Fig. 4 Performance comparison with HoloClean system for different cleaning tasks over the same time period

6.4.1 预清洗与分块对清洗效果的影响

如表 9 和表 10 所列，当采用第 5.1 节描述的分块策略对候选键进行预清洗时，清洗的 F1 指标和所需时间在各种清洗任务下都得到了显著的提升，其中规则违反的清洗任务下的提升尤为突出。这一结果说明了分块策略的效果：它不仅提高了整体的清洗效率，还确保了数据的局部一致性和完整性。通过先对候选键进行预清洗再进行分块，AQ 算法中的聚合查询得以更为精确的执行，从而进一步提高了清洗的准确性。

6.4.2 清洗任务和数据规模对清洗效果的影响

如表 10、图 3(a)和图 3(b)所示，系统在处理各种清洗任务时都展现出了稳定的准确率、召回率和时间性能。无论是在异常值、缺失值、规则违反还是混合错误的清洗任务中，系统的准确率和召回率随着数据规模的增加均保持稳定。而在时间方面，随着数据规模的扩大，系统的处理时间大致呈正线性增长。

这种稳定的性能主要得益于算子库的多样性。具体来说，算子库中的基于最小条件依赖违反的修复方法、基于匹配依赖的匹配修复方法以及基于统计特征的修复都能生成多样的中间算子。通过对这些多样的中间算子进行综合选择和决策，系统能够有效地应对各种清洗任务。

6.4.3 与其他智能数据清洗系统的对比

图 4 给出了本系统采用分块策略后与领域内先进的 Holoclean 智能数据清洗系统[1]在不同清洗任务下的性能对比。为了进行公平比较，本实验通过调整搜索器的搜索深度来确保本系统与 Holoclean 的清洗时间相同，在此基础上进一步对比了两者在清洗性能上的差异。

结果显示，在处理缺失值错误的清洗任务时，本系统的效果略逊于 Holoclean，差距约为 3%。这主要是因为算子库中的某些清洗信号与当前的数据场景不完全匹配。例如，某些基于统计特征的清洗信号倾向于使用出现频率最高的值进行填充，但在某些特定情境下，这并不是最佳选择。这导致了由这些信号生成的中间算子质量不高，使得整体算子质量不均，从而增加了筛选合适证据集的时间。

然而，在处理异常值、规则违反和混合错误等其他任务时，本系统的性能明显超越了 Holoclean，提升幅度达到了 15%以上。这一优势主要归因于我们的算子库为这些任务提供了高质量的中间算子。

结束语 本文将数据清洗过程视为一个证据集的生成与选择过程，通过将多种清洗信号相互配合成功地产生了更高质量的数据。系统采用了中间算子这一通用的数据修复语言，并在基于聚合查询的增量式质量评估器的支持下，确保数据从系统上游流向下游时其质量得到不断的优化和提升。未来工作中，我们计划在以下两个方面继续深入研究：1)针对不同的数据情境，系统能够自动筛选与之匹配的清洗信号，从而生成高质量的中间算子进行数据清洗；2)进一步丰富算子库，加入更多高质量的清洗信号，以增强系统处理各种清洗任务的能力。

参 考 文 献

[1] REKATSINAS T, CHU X, ILYAS I F, et al. HoloClean: Holistic Data Repairs with Probabilistic Inference[C]// Proceedings of the VLDB Endowment. 2017.

[2] SINGH P. Systematic review of data-centric approaches in artificial intelligence and machine learning[J]. Data Science and Management, 2023, 6(3): 144-157.

[3] PAASCHE S, GROPPE S. Enhancing data quality and process optimization for smart manufacturing lines in industry 4.0 scenarios[C]// Proceedings of The International Workshop on Big Data in Emergent Distributed Environments. 2022: 1-7.

[4] HAO S, LI G L, FENG J H, et al. Survey of structured data cleaning methods[J]. Journal of Tsinghua University (Science and Technology), 2018, 58(12): 1037-1050.

[5] ILYAS I F. Effective Data cleaning with Continuous Evaluation [J]. IEEE Data Engineering Bulletin, 2016, 39(2): 38-46.

[6] LI H, TANG B, LU H, et al. Spatial data quality in the iot era: management and exploitation[C]// Proceedings of the 2022 International Conference on Management of Data. 2022: 2474-2482.

[7] KRISHNAN S, HAAS D, FRANKLIN M J, et al. Towards reliable interactive data cleaning: A user survey and recommendations[C]// Proceedings of the Workshop on Human-In-the-Loop Data Analytics. 2016: 1-5.

[8] GUO Z, ZHOU A Y. Research on data quality and data cleaning: a survey[J]. Journal of software, 2002, 13(11): 2076-2082.

[9] DING X O, WANG H Z, ZHANG X Y, et al. Association relationships study of multi-dimensional data quality[J]. Journal of Software, 2016, 27(7): 1626-1644.

[10] KRISHNAN S, WU E. Alphaclean: Automatic generation of data cleaning pipelines[J]. arXiv: 1904.11827, 2019.

[11] KRISHNAN S, FRANKLIN M J, GOLDBERG K, et al. Boostclean: Automated error detection and repair for machine learning [J]. arXiv: 1711.01299, 2017.

[12] ABEDJAN Z, CHU X, DENG D, et al. Detecting data errors: Where are we and what needs to be done? [J]. Proceedings of the VLDB Endowment, 2016, 9(12): 993-1004.

[13] FARIHA A, TIWARI A, MELIOU A, et al. Coco: Interactive exploration of conformance constraints for data understanding and data cleaning[C]// Proceedings of the 2021 International

Conference on Management of Data. 2021:2706-2710.
[14] QAHTAN A, TANG N, OUZZANI M, et al. Pattern functional dependencies for data cleaning[C]// Proceedings of the VLDB Endowment. 2020.
[15] KRISHNAN S, WANG J, WU E, et al. Activeclean: Interactive data cleaning for statistical modeling[J]. Proceedings of the VLDB Endowment, 2016, 9(12): 948-959.
[16] XI Y, WANG N, CHEN X, et al. EasyDR: a human-in-the-loop error detection&repair platform for holistic table cleaning[J]. Proceedings of the VLDB Endowment, 2022, 15(12): 3578-3581.
[17] DE SA C, ILYAS I F, KIMELFELD B, et al. A Formal Framework for Probabilistic Unclean Databases[C]// 22nd International Conference on Database Theory. 2019.
[18] REZIG E K, OUZZANI M, AREF W G, et al. Horizon: scalable dependency-driven data cleaning[J]. Proceedings of the VLDB Endowment, 2021, 14(11): 2546-2554.
[19] MAHDAVI M, ABEDJAN Z. Baran: Effective error correction via a unified context representation and transfer learning[J]. Proceedings of the VLDB Endowment, 2020, 13(12): 1948-1961.
[20] PENG J, SHEN D, TANG N, et al. Self-supervised and Interpretable Data Cleaning with Sequence Generative Adversarial Networks[J]. Proceedings of the VLDB Endowment, 2022, 16(3): 433-446.
[21] ILYAS I F, CHU X. Data Cleaning[M]. Morgan & Claypool, 2019: 49-54.
[22] MARQUES F S L. Discovering Denial Constraints Using Boolean Patterns[C]// Companion of the 2023 International Conference on Management of Data. 2023: 281-283.
[23] RAY B, GHOSH S, AHMED S, et al. Outlier detection using an ensemble of clustering algorithms[J]. Multimedia Tools and Applications, 2022, 81(2): 2681-2709.
[24] LI X, DONG X L, LYONS K, et al. Truth Finding on the Deep Web: Is the Problem Solved? [C]// Proceedings of the VLDB Endowment. 2012.
[25] LEWIS M, LIU Y, GOYAL N, et al. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension[C]// Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020: 7871-7880.

QIAN Zekai, born in 2001, postgraduate, is a member of CCF(No. P8213G). His main research interests include data governance and so on.

WANG Hongzhi, born in 1978, Ph. D, professor, is a member of CCF(No. 07132D). His main research interests include databases, big data management and analysis, and big data governance.

(责任编辑:柯颖)