Data Quality - State-of-the-Art

By

Kristóf Balázs
Stefanos Kypritidis

Professor:
Oscar Romero

Barcelona, June 2025

Introduction

High-quality data is important for reliable analysis in big data and data science, and it is often overlooked. As data sets grow in volume and variety, their veracity becomes equally important - 'big data = quantity of data + data quality' [1]. Poor data quality can weaken analytics and machine learning results. However, ensuring data quality in practice is challenging due to the nature of modern data. Data quality is a multidimensional, context-dependent concept (often defined as "fitness for use" [2]), and has dimensions such as accuracy, completeness, consistency, and timeliness. These dimensions are hard to measure and maintain, especially as real-world data continuously changes and is aggregated from heterogeneous sources. Datasets often contain a variety of errors, from typos and missing values to duplicates, outliers, and integrity violations, which makes it difficult to have consistent data quality. In fact, even defining a common set of data quality criteria has required a lot of effort (e.g., a recent taxonomy of data quality dimensions [3]), showing that what counts as 'good data' often depends on the specific use case.

Over the past decade, many automated tools and techniques have been developed to detect and fix data errors. These range from rule-based validation and constraint checking systems to statistical outlier detection and machine learning approaches for error detection [4, 5]. Important data cleaning frameworks (for instance, using integrity rules or probabilistic models) have been successful in specific domains such as healthcare, finance, or web data. However, most solutions are specialized, designed for particular data types and domains or for specific error categories. Many tools require a lot of domain expertise or custom rules to be useful, which limits their ability to generalize. Moreover, there is no single tool or method that works for all data sets and error types. There is no technique that consistently outperforms others on diverse datasets; each is limited to detecting certain kinds of errors [4]. In practice, ensuring high data quality often requires combining multiple techniques, and even then, some errors are still missed. This challenge is reflected in the large number of domain-specific tools. More than half of the tools surveyed focus on specific domains [2], and there is still no standard set of metrics used in all platforms.

In short, in the domain of data quality tools, there is no one-size-fits-all solution. By systematically reviewing state-of-the-art data error detection tools, our objective is to clarify how different solutions operate, the trade-offs they make, and where they complement each other.

Methodology

This work uses a comparative literature review methodology, specifically focusing on automated error detection tools and their strategies within data quality (DQ) management. Given the amount of research on data quality, our initial goal was to narrow the scope to perform a comparative analysis.

The research process involved a keyword-based search on platforms such as Google Scholar and DBLP. We first gathered the most influential papers in the domain to get a general idea. After this, relevant keywords like *data quality*, *error detection*, *data cleaning*, and specific strategies such as *constraint-based*, *statistical-based*, *ML-based*, and *knowledge-based* were used. Moreover, snowballing techniques were applied, both backward and forward. We narrowed our scope and selected representative tools that cover error detection approaches. Our final comparative framework uses constraint-based, ML-based, statistical-based, statistical constraints-based, knowledge-based, and transformer-based approaches.

When we had a better understanding of the available methods, we narrowed our focus and chose tools that represent different types of approaches. The decision was made based on two primary criteria: first, selecting tools that represent different error-detection strategies; second, focusing on recent publications. Initial candidates included NADEEF (constraint-based), Deequ (constraint and profiling-based), Raha (ML-based), Uni-Detect (statistical-based), CODED (statistical constraints-based), KATARA (knowledge-based), Data X-Ray (error provenance), ActiveClean (ML-loop based), and DataVinci (transformer-based).

The main research question in this study is: *Which automated error-detection strategies do current data quality tools use, and how well does each strategy cover the main data quality dimensions and error types found in data?*

Background & Terminology

In today's world, marked by the rapid advancement of artificial intelligence and big data, data quality has emerged as a central focus of technological innovation and development. The concept of data quality has been extensively explored in academic research over the past several decades, resulting in a diverse array of definitions. These definitions differ significantly. In detail, data quality is characterized as a multidimensional construct, with each dimension representing a distinct quality attribute by some fundamental works [6] [7] [8]. In that case, general frameworks are provided, applicable to a wide range of data types. Other studies focus on more specific contexts, for instance on its implications within the healthcare sector [9].

Data errors are closely linked to data quality and have a significant impact on it. These errors can be classified into several categories, including missing values, where data entries are absent, and duplicates, which occur when records are repeated unnecessarily. Inconsistencies arise when conflicting information exists within the dataset, while outliers represent data points that deviate significantly from the expected range of values. Additionally, violation of semantic rules, for example, violation of integrity constraints in relational theory, affects the consistency [10]. These diverse error types highlight the need for comprehensive methodologies to assess and improve data quality effectively.

As described, data quality is recognized as a multidimensional concept. Various dimensions define this concept, including accessibility, amount of data, believability, completeness, concise representation, consistent representation, ease of manipulation, free of error, interpretability, objectivity, relevancy, reputation, security, timeliness, understandability, and value-added [7]. However, a lack of consensus has been mentioned on the quality dimensions [6]. Furthermore, one literature review identified the most frequently cited quality dimensions. Those were found to be accessibility, accuracy, completeness, consistency, and currency (also called timeliness) [11]. Similarly, another paper found that the most common dimensions are completeness, accuracy, timeliness, consistency, and accessibility [12].

After reviewing the mentioned studies, we selected the most commonly cited data quality dimensions to serve as the basis for our comparative analysis of data quality tools. These dimensions were chosen due to their broad acceptance and relevance. Specifically, following the definitions provided in [6] the dimensions are:

- **Completeness:** The extent to which data possess sufficient breadth, depth, and scope to support the intended task.
- **Accuracy:** The degree to which data are correct, reliable, and validated.
- **Timeliness:** The relevance of the data's age in relation to the specific requirements of the task.
- **Consistency:** The extent to which data are presented in a uniform format and remain compatible with previous datasets.
- **Accessibility:** The ease with which information can be accessed, retrieved, and used efficiently.

Identifying incorrect values within a database, known as error detection, has been a persistent challenge over time [4]. Aside from the consulting-based approach where companies rely on in-house experts or external consultants to clean high-value data sets, there is the automated error-detection approach for less valuable datasets [13].

The automated error detection methods are generally categorized into six types: statistical-based, constraint-based, cluster-based, outlier-based, ML-based, and deep-learning-based [5]. Statistical-based methods use metrics and models like Principal Component Analysis (PCA) to detect deviations from expected distributions, offering strong interpretability but limited generalization. Constraint-based methods rely on data integrity rules—such as Functional Dependencies (FDs) and Denial Constraints—to flag violations, which makes them explainable and domain-aware, though sometimes rigid. Cluster-based methods use clustering algorithms like DBSCAN or OPTICS to group similar data points and mark outliers as errors, providing intuitive insights but often being sensitive to parameters. On the other hand, outlier-based methods identify data points that significantly deviate from the norm using distance or density

metrics, making them flexible but potentially prone to false positives in high-dimensional or noisy datasets [5].

In recent years, learning-based methods have gained traction for error detection due to their adaptability and scalability. Machine learning-based error detection leverages statistical and probabilistic models—such as those used in HoloClean, GDR, and ERACER—to identify and repair data errors by learning patterns from prior data, integrating repair signals, and in some cases incorporating active learning or user feedback to improve accuracy and generalizability [14]. Deep learning-based methods use models like autoencoders or attention mechanisms to capture complex, non-linear relationships in the data, enabling detection of subtle anomalies but typically needing large datasets and sacrificing interpretability [5]. Lastly, a system detecting data errors by leveraging knowledge bases to validate facts and identify inconsistencies has been introduced, using semantic matching and entity linking to compare data entries against trusted sources [15].

Data Quality Tools

CODED

A new approach to identifying data quality issues by using statistical constraints (SCs) is introduced [16]. SCs are declarative specifications of expected statistical relationships between columns. In detail, SCs capture probabilistic relationships like, conditional independence and correlation among sets of attributes. A unified framework for error detection is presented by checking for violations of these statistical expectations in a given dataset. The system, called CODED, identifies likely mistaken records by analyzing where SCs are not satisfied, and as a result, offers a scalable and general method for error detection across diverse datasets.

CODED integrates statistical hypothesis testing and data repair techniques to not only identify potential data errors but also provide explanations for these errors. This is achieved by identifying the top-k data records that most influence the violation of the SC. The system requires a set of SCs as input, which can either be domain knowledge of domain experts or automated machine learning methods to identify dependencies, but verified by a domain expert. SCs are shown as highly effective for detecting various types of data errors, such as missing values, outliers, and sorting errors, performing much better than baseline methods in precision and recall. Those insights are produced from extensive experiments on real-world and synthetic datasets[16].

Deequ

Deequ is an open-source data quality validation library from AWS Labs, built on Apache Spark to enable scalable checks on large datasets [17]. It uses constraint checks and profiling as its main approach and has a declarative API to express data quality constraints (expected data

conditions) and automatically validates these assumptions as 'unit tests' for data [18]. By encoding expectations for completeness (no missing values), consistency (comply with defined rules in data), and accuracy (correct value ranges or formats), Deequ can detect records or statistics that violate these constraints and flag data errors such as null entries or values outside the range during validation.

A typical Deequ workflow first profiles the data, computes a set of quality metrics (e.g., completeness, distinctness, correlations) with Spark, then checks the data against the defined constraints, and produces a report of the violations. The library can also suggest potential constraints automatically by analysing the dataset (it infers rules from data distributions). This helps users discover quality issues without existing rules [17]. Moreover, Deequ supports continuous data quality monitoring: it can track these metrics over time and detect anomalies or distribution shifts in new data by comparing profiling results over dataset versions (early detection of data drift) [17, 19]. It was originally developed for Amazon's internal use, Deequ today validates many large production datasets [17]. Its engine is also the foundation for AWS's managed data quality services, such as AWS Glue Data Quality and Amazon SageMaker Model Monitor [20].

Raha

Raha is a machine learning-driven error detection system for structured (tabular) data, designed to catch various data errors without any manual configuration [21]. Traditional data cleaning tools often require users to specify error rules or parameters (like in the case of Deequ), but Raha it is configuration-free, which means it does not rely on rules provided by the users [21]. These include outlier detectors (for numeric out of range values), pattern checkers (for format or missing value violations), integrity/constraint validators (for duplicates or functional dependency violations), and knowledge base checks (to cross reference values against master data) [21]. Each detector flags certain cells as potential errors (e.g. marking null entries, outliers, or inconsistent values), and Raha compiles all these signals into a feature vector for every data tuple or cell [21]. This feature representation can encode hundreds of error indicators, and provides the basis for Raha's learning algorithm.

Using these features, Raha uses a semi-supervised machine learning approach to decide which data values truly contain errors [21]. It clusters data by similar feature patterns and then uses learning to minimize human intervention of effort: the system iteratively asks the user to label a small number of representative tuples as "clean" or "dirty" (around 20 labelled examples per dataset) [21]. These few labels are propagated to their clusters and used to train a classification model that generalizes across the entire dataset [22]. In other words, Raha learns an error detection classifier that combines the votes of all base detectors, rather than relying on any single rule. It is a two-step process: first it runs many different detectors to maximize recall, then it learns an optimal combination to improve precision. This allows the system to both catch most errors and remain accurate [22].

KATARA

KATARA [15] is a data cleaning system that combines knowledge bases and crowdsourcing to identify and fix errors in tabular data. Rather than relying only on predefined rules or statistical methods, KATARA relies on the semantic relationships between columns to align tables with knowledge bases like Yago, Freebase and DBpedia, which mostly use RDFS [23]. This allows it to detect inconsistencies that traditional approaches might miss. It uses a scoring mechanism to assess the confidence of entity alignments between table cells and knowledge base entries. These scores help identify potential data errors by highlighting mismatches or low-confidence mappings. Once those are found, KATARA suggests corrections based on the knowledge base information and, when needed, brings in human input through crowdsourcing to cope with uncertain cases [15].

The process starts by matching values in the table to known knowledge base entities. KATARA then checks for semantic consistency, for example, verifying that a listed city belongs to the correct country, using knowledge from the knowledge base. If a mismatch or ambiguity is found, it uses crowd workers, who are experts in the semantics of the knowledge bases, to help validate or correct the data. Their responses are combined to produce a final decision, through majority voting. This combination of automated validation and human judgment helps KATARA clean data more accurately, especially in cases where the structure is weak or the data is noisy. Lastly, it is useful for web tables or unclear naming conventions, where context and external knowledge play a key role in understanding and correcting errors [15].

Uni-Detect

Uni-Detect [13] is a unified system for automated error detection in tabular data that aims to handle uniqueness constraint violations, functional-dependency violations, numeric outliers, and spelling mistakes using an unsupervised error detection approach. Uni-Detect uses a data-driven approach, trained on a large corpus of over 100M mostly clean web tables [24] to learn statistical patterns of clean data. This enables it to generalize across diverse schemas without requiring explicit manual configuration, making it highly usable for real-world applications[13].

The core methodology of Uni-Detect involves hypothesis testing at the cell level: the null hypothesis assumes no error, while the alternative hypothesis flags potential anomalies. Then, Uni-Detect employs perturbation-based testing, where subsets of rows are removed to evaluate the sensitivity of table structures and relationships. Cells that deviate significantly from expected patterns under these perturbations are flagged as errors. Empirically, Uni-Detect demonstrates state-of-the-art performance across benchmark datasets like Wikipedia and Web datasets, often surpassing specialized error detectors while minimizing manual intervention.

Its schema-agnostic design and unsupervised nature make it particularly suitable for heterogeneous data environments where labeled training data or domain expertise are limited [13].

DataVinci

DataVinci is a transformer-based data cleaning system for string data. It performs error detection and repair automatically [25]. It learns regex patterns that cover most of the values in a column and flags any outliers that deviate from these patterns as data errors [25]. For each error it spots, DataVinci computes the smallest set of edits needed to make the value match its learned pattern [25]. Unlike rule-based or semi-supervised solutions mentioned above, DataVinci does not require user-defined rules, annotated examples, or constraints [25]. It combines a pattern mining engine with a pre-trained transformer (LLM) to handle both syntactic mistakes (e.g. format inconsistencies, punctuation errors) and semantic ones (e.g. misspellings or missing entity names) in strings [25]. While learning patterns, the LLM replaces substrings with generic placeholders. When it puts them back, it uses its knowledge to fix typos and standardize them [25]. DataVinci can also use the context in which the program runs. For example, it looks at whether a spreadsheet formula works to find and fix values that cause errors [25]. This fully unsupervised, learning-based approach lets DataVinci catch and repair many types of string errors (typos, missing tokens, format deviations) and generalize to new datasets, all without manual tuning. In both real and synthetic benchmarks, DataVinci outperformed multiple prior error detection and repair methods in both precision and recall [25].

Tool Comparison

Since the tools compare different types of errors across various datasets, directly measuring overall accuracy is challenging. To address this, we briefly present the results of each paper's benchmarks. Notably, Uni-Detect demonstrates superior performance by outperforming Speller, GloVe, Word2Vec, and FuzzyCluster in terms of precision for detecting spelling errors, and also surpasses MaxSD, DBOD, LOF, and Max-MAD in precision for detecting numeric outliers [13]. On the other hand, CODED outperforms Denial Constraints and DBoost in both precision and F-score across various error levels on numerical and categorical data, and surpasses Approximate Functional Dependencies by maintaining higher F-scores when detecting errors in both dependent and independent attributes [16]. Additionally, despite KATARA's suffering from many false positives, models trained on KATARA-repaired datasets still achieve similar predictive performance to models trained on data cleaned by NADEEF, HoloClean, FAHES, and others [26]. Deequ's benchmarks focused on scalability and usability in production rather than precision/recall metrics. The tool was benchmarked on large Amazon datasets and in production pipelines, and it managed to validate data quality constraints on billions of rows using Spark (the goal was to meet Amazon's internal needs to catch data problems early) [17]. Raha was initially evaluated on eight datasets (real-world error-labelled data) against multiple

detection baselines (dBoost, NADEEF, KATARA, ActiveClean) and achieved 12–42% higher F1 scores than the best alternative, even if it only required minimal labelling [21]. Moreover, in other benchmarks involving multiple error types, Raha was consistently among the best performing tools (average F1 scores of 0.8 or higher) in many different datasets, with the only issue being limited scalability for larger datasets ($\geq$50k rows) [26]. Lastly, DataVinci's authors benchmarked it on four different types of datasets (real Wikipedia tables, public Excel spreadsheets, a synthetic corrupted data test, and an Excel formula repair task) and evaluated both error detection and error repair metrics. Across these benchmarks, DataVinci was the best, beating seven baseline methods (including prior rule-based, machine learning, and even GPT-3.5 powered approaches) both in detecting and fixing errors [25]. In particular, it had the highest precision and recall rates on tasks examined in this state-of-the-art (e.g., the best F1 in the synthetic errors dataset and the highest precision in real-world string data) [25].

Tool	**Year**	**Method**	**Key Contribution**	**Citation**
CODED	2019	Statistical & constraint	Formalises statistical constraints for scalable error detection	[16]
DataVinci	2025	Transformer-based pattern learning	Unsupervised string-error detection and repair with regex mining plus LLM reasoning	[25]
Deequ	2018	Constraint checks & profiling	Spark library; declarative DQ tests and automatic rule suggestion	[17, 19, 20]
KATARA	2015	Knowledge-base & crowdsourcing	Combines KB alignment with crowd validation for semantic error repair	[15, 23]
Raha	2019	ML ensemble (active learning)	Config-free detector that fuses 100+ base signals with minimal labels	[21, 22]
Uni-Detect	2019	Statistical & ML	Unsupervised, schema-agnostic error detection without configuration	[13, 24]

Table 1: Reference table summarizing state-of-the-art tools

Tool	Accessibility	Accuracy	Completeness	Consistency	Timeliness
CODED	Requires input of statistical constraints	High	Focuses on detecting errors (outliers, missing data, sorting errors) rather than completeness	By enforcing statistical constraints	By updating the inputed statistical constraints in case of changes
DataVinci	No prior rules or labels required; Python prototype available	High (str)	Targets string-level issues; does not address missing rows/-columns	Learns and enforces uniform regex/semantic patterns	Batch-oriented; no built-in streaming checks
Deequ	Open-source Scala/Python Spark library; declarative API	High	Can measure / enforce null-completeness with constraints	Constraint checks for intra- and inter-column consistency	Spark jobs or scheduled runs for near real-time validation
KATARA	Requires integration with knowledge bases (KBs) & crowdsourcing	High	Improves using external knowledge	Through KBs	Up to date to the KBs with some possible crowdsourcing delays
Raha	Open-source Java/Python; needs ≈20 labelled tuples	High	Detects many types of errors but does not fill missing data	Combines binary vector of many detector signals plus learned classifier	Offline batch processing; retraining triggered after significant data drift
Uni-Detect	No input or configuration needed in current version	High	Focuses on detecting spelling mistakes, numeric outliers, functional-dependency and uniqueness violations	Being trained on a large corpus of over 100M clean web tables	Adapts to new data but no possibility to edit learning configurations

Table 2: Comparison of tools based on data quality dimensions

Final Discussion

No single of these tools can be used to automatically detect and explain all data quality error types. Each tool offers its own strengths and limitations. Different tools should be used for different error detection tasks, and this is highly dependent on the domain of the data, the error type, the structure of the data, the presence and availability of domain experts, and the operational constraints like scalability, computational resources, real-time necessity, interpretability, and privacy. Additionaly, tools differ in how well they address the various data quality dimensions, with most underperforming in timeliness. The exception there is Deequ, since it can be integrated into scheduled runs or Spark jobs.

Future research can be done to investigate further the most effective tools for real-time detection of errors, which can be significant for IoT and log monitoring. Furthermore, standard benchmarks for the data quality tools and dimensions is crucial since, as seen in the comparison section, it is very difficult to directly compare tools across specific dimensions. As a result, a unified benchmarking dataset or framework would greatly aid in assessing tool performance. Lastly, regarding the machine learning-based and deep learning-based error detection methods, enhancing their interpretability and developing interactive interfaces for domain experts will be vital for trustworthiness and adoption.

Bibliography

[1] Wenfei Fan. Data quality: From theory to practice. *Acm Sigmod Record*, 44(3):7–18, 2015.

[2] Lisa Ehrlinger and Wolfram Wöß. A survey of data quality measurement and monitoring tools. *Frontiers in big data*, 5:850611, 2022.

[3] Sedir Mohammed, Lisa Ehrlinger, Hazar Harmouch, Felix Naumann, and Divesh Srivastava. Data quality assessment: Challenges and opportunities. *arXiv preprint arXiv:2403.00526*, 2024.

[4] Ziawasch Abedjan, Xu Chu, Dong Deng, Raul Castro Fernandez, Ihab F Ilyas, Mourad Ouzzani, Paolo Papotti, Michael Stonebraker, and Nan Tang. Detecting data errors: Where are we and what needs to be done? *Proceedings of the VLDB Endowment*, 9(12):993–1004, 2016.

[5] Jingyu Zhu, Xintong Zhao, Yu Sun, Shaoxu Song, and Xiaojie Yuan. Relational data cleaning meets artificial intelligence: A survey. *Data Science and Engineering*, pages 1–28, 2024.

[6] Richard Y Wang and Diane M Strong. Beyond accuracy: What data quality means to data consumers. *Journal of management information systems*, 12(4):5–33, 1996.

[7] Leo L Pipino, Yang W Lee, and Richard Y Wang. Data quality assessment. *Communications of the ACM*, 45(4):211–218, 2002.

[8] International Organization for Standardization. Software engineering — software product quality requirements and evaluation (square) — data quality model (iso standard no. 25012:2008). `https://www.iso.org/standard/35736.html`, 2024. Accessed: 2025-06-16.

[9] Annamaria Chiasera, Tefo James Toai, Leandro Paulo Bogoni, Giampaolo Armellin, and Juan José Jara. Federated ehr: how to improve data quality maintaining privacy. In *2011 IST-Africa Conference Proceedings*, pages 1–8. IEEE, 2011.

[10] Carlo Batini, Cinzia Cappiello, Chiara Francalanci, and Andrea Maurino. Methodologies for data quality assessment and improvement. *ACM computing surveys (CSUR)*, 41(3):1–52, 2009.

[11] Nuno Laranjeiro, Seyma Nur Soydemir, and Jorge Bernardino. A survey on data quality: classifying poor data. In *2015 IEEE 21st Pacific rim international symposium on dependable computing (PRDC)*, pages 179–188. IEEE, 2015.

[12] Corinna Cichy and Stefan Rass. An overview of data quality frameworks. *Ieee Access*, 7:24634–24648, 2019.

[13] Pei Wang and Yeye He. Uni-detect: A unified approach to automated error detection in tables. In *Proceedings of the 2019 International Conference on Management of Data*, pages 811–828, 2019.

[14] Mohammad Mahdavi, Felix Neutatz, Larysa Visengeriyeva, and Ziawasch Abedjan. Towards automated data cleaning workflows. *Machine Learning*, 15:16, 2019.

[15] Xu Chu, John Morcos, Ihab F Ilyas, Mourad Ouzzani, Paolo Papotti, Nan Tang, and Yin Ye. Katara: A data cleaning system powered by knowledge bases and crowdsourcing. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 1247–1261, 2015.

[16] Jing Nathan Yan, Oliver Schulte, Jiannan Wang, and Reynold Cheng. Detecting data errors with statistical constraints. *arXiv preprint arXiv:1902.09711*, 2019.

[17] Philipp Schmidt, Dustin Lange, Sebastian Schelter, and Tammo Rukat. Test data quality at scale with deequ. `https://aws.amazon.com/blogs/big-data/test-data-quality-at-scale-with-deequ/`, May 2019. Accessed: 2025-06-16.

[18] Sebastian Schelter, Philipp Schmidt, Tammo Rukat, Mario Kiessling, Andrey Taptunov, Felix Biessmann, and Dustin Lange. Deequ - data quality validation for machine learning pipelines. 2018.

[19] Yuhan Zhou, Fengjiao Tu, Kewei Sha, Junhua Ding, and Haihua Chen. A survey on data quality dimensions and tools for machine learning, 2024.

[20] Tammo Rukat, Dustin Lange, Sebastian Schelter, and Felix Biessmann. Towards automated data quality management for machine learning. In *ML Ops Work. Conf. Mach. Learn. Syst*, pages 1–3, 2020.

[21] Mohammad Mahdavi, Ziawasch Abedjan, Raul Castro Fernandez, Samuel Madden, Mourad Ouzzani, Michael Stonebraker, and Nan Tang. Raha: A configuration-free error detection system. In *Proceedings of the 2019 International Conference on Management of Data*, pages 865–882, 2019.

[22] BigDaMa team. Raha: Configuration-free error detection and correction systems. `https://github.com/BigDaMa/raha`, 2025. Version v1.25; accessed 2025-06-17.

[23] Omkar Deshpande, Digvijay S Lamba, Michel Tourn, Sanjib Das, Sri Subramaniam, Anand Rajaraman, Venky Harinarayan, and AnHai Doan. Building, maintaining, and using knowledge bases: a report from the trenches. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pages 1209–1220, 2013.

[24] Zhipeng Huang and Yeye He. Auto-detect: Data-driven error detection in tables. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1377–1392, 2018.

[25] Mukul Singh, José Cambronero, Sumit Gulwani, Vu Le, Carina Negreanu, Arjun Radhakrishna, and Gust Verbruggen. Datavinci: Learning syntactic and semantic string repairs. *Proceedings of the ACM on Management of Data*, 3(1):1–26, 2025.

[26] Mohamed Abdelaal, Christian Hammacher, and Harald Schoening. Rein: A comprehensive benchmark framework for data cleaning methods in ml pipelines. *arXiv preprint arXiv:2302.04702*, 2023.