Automatic Consistency Assessment through Partial Functional Dependency Mining

Marcian Seeger
Marburg University
Marburg, Germany
marcian.seeger@uni-marburg.de

Thorsten Papenbrock
Marburg University
Marburg, Germany
thorsten.papenbrock@uni-marburg.de

Felix Naumann
Hasso Plattner Institute, University of Potsdam
Potsdam, Germany
felix.naumann@hpi.de

Lisa Ehrlinger
Hasso Plattner Institute, University of Potsdam
Potsdam, Germany
lisa.ehrlinger@hpi.de

ABSTRACT

High data quality is critical for many data-driven applications, ranging from business analytics to medical diagnostics to industrial automation. Because structural integrity violations are a key indicator of poor data quality, *consistency* is widely considered a core dimension of data quality assessment. Functional dependencies in relational databases are a particularly interesting type of consistency indicator, because they capture semantic relationships and are usually not enforced as integrity constraints by database management systems; hence, their violations often mark data quality issues. Existing consistency assessment approaches, however, either assume manually specified dependencies or rely on semi-automatic discovery and selection. This results in high costs, limited scalability, and concerns regarding reproducibility and objectivity.

We propose a fully automated, deterministic consistency assessment technique that requires no human involvement: It (i) discovers partial functional dependencies (pFDs) directly in the data, (ii) weights each pFD with a statistical genuineness measure to distinguish true semantic rules from coincidental patterns, and (iii) aggregates the weighted pFDs into an overall consistency score. We demonstrate that our metric satisfies established requirements for data quality metrics and outperforms manually defined rule sets in coverage, objectivity, and robustness on real-world datasets for data cleaning. We implemented the metric in the data quality assessment framework Metis.

VLDB Workshop Reference Format:
Marcian Seeger, Thorsten Papenbrock, Felix Naumann, and Lisa Ehrlinger. Automatic Consistency Assessment through Partial Functional Dependency Mining. VLDB 2026 Workshop: 15th International Workshop on Quality in Databases (QDB'26).

VLDB Workshop Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/SeegerM/cpfd-reproducibility.

Proceedings of the VLDB Endowment. ISSN 2150-8097.

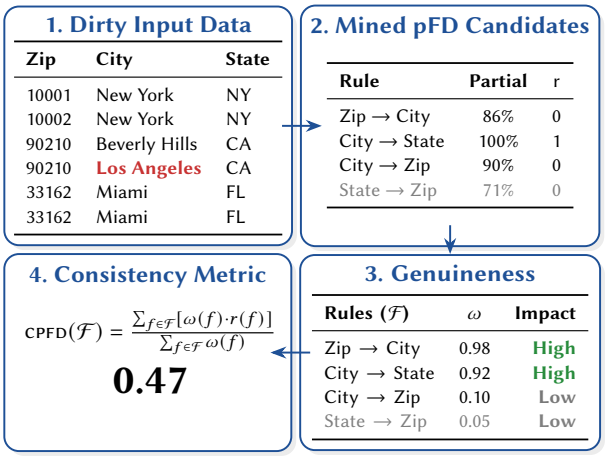

Figure 1: Our automated consistency assessment technique: To handle minor data errors in (1), we mine partial dependencies in (2) and weight them by their statistical genuineness ω in (3) to calculate a robust overall consistency score CPFD in (4). Gray dependencies are not considered in further steps.

1 INTRODUCTION

Many data-driven applications depend on high-quality data [18, 19]. If the input data is of poor quality, the resulting predictions, decisions, and actions may become unreliable or even harmful. Consequences range from increasing expenses (e.g., due to failed deliveries or incorrect invoices) to mistrust in organizations and their systems. *Data quality* (DQ) is commonly described as the degree to which the data of an information system accurately represents the real world [19, 24]. It is typically organized into dimensions, such as *accuracy*, *completeness*, and *consistency* [3]. Each dimension can be quantified through DQ metrics [8, 24], which are measurement functions that map a high-level DQ dimension to a numerical value. Our DQ tool Metis[1] allows automatic assessment of tabular data quality across multiple DQ dimensions.

Consistency, the focus of this paper, captures the violation of semantic rules defined over data [3]. Such semantic rules are often

[1] https://github.com/HPI-Information-Systems/Metis

expressed as data dependencies, most notably functional dependencies (FDs). However, existing consistency metrics typically assume a given set of rules, which are often provided by domain experts or are already known [1, 2, 12, 14]. In practice, this assumption often does not hold: Limited human expertise and large data volumes usually render the manual definition of consistency rules infeasible. And even if manually defined rules exist, they are, as we show in our evaluation, often outdated, overly specific, too abstract, or even incorrect. In addition, these rules are inherently subjective because different human experts tend to make different judgments, especially when data is shared across teams. To date, no fully automated, deterministic consistency assessment approach exists.

Modern data profiling algorithms offer a potential solution: They automatically discover all syntactically valid consistency rules, and by relaxing the scope of the to-be-discovered data dependencies, they are able to extract partial (functional) dependencies that tolerate a certain number of errors in the data. The automatic and deterministic selection of semantically meaningful, i.e., genuine data dependencies is, however, still an open research question.

Contributions. In this paper, we propose CPFD, a fully automated technique to assess consistency through partial functional dependency mining, as shown in Figure 1. Given a (potentially dirty) input dataset (Step 1), CPFD first discovers all partial functional dependencies (pFDs) with their *scope of validity* (Step 2). Then, it calculates a statistical *genuineness weight* for each discovered pFD to distinguish meaningful from spurious dependencies; technically, we measure how much a relationship deviates from random chance (Step 3). Finally, CPFD aggregates the weighted pFDs into a consistency score (Step 4) using a metric formula that fulfills all five requirements for DQ metrics [8]. In detail, our contributions are:

(1) We extend the HyFD data profiling algorithm to discover *partial functional dependencies* and use the gpdep measure to weight the *genuineness* of the pFDs.
(2) We introduce a new DQ metric to fully automatically assess the *consistency* of a relational dataset.
(3) We complement the metric result with *explanations* that allow to trace and identify violations in the data as well as inconsistent rules.
(4) We evaluate our consistency metric on seven different datasets to demonstrate its effectiveness and robustness.

2 RELATED WORK AND BACKGROUND

Table 1 summarizes related work along two groups of design dimensions. The first group covers rule type and discovery requirements for automated consistency assessment: automatic rule discovery (D1), robustness of rule discovery to dirty data (D2), that rules can be weighted, e.g., by uncertainty (D3), and independence of human input (D4). The second group covers the metric used and its compliance with the five metric requirements by Heinrich et al. [8] (M1–M5), which we discuss in detail in Section 4.3. The metrics follow the notation from [12]: the to-be-assessed data is either value g, a record t, or a database D_B; N denotes the number of relevant rules, $r_n(\cdot)$ a binary violation indicator, and w_n an optional weight. As Table 1 shows, no existing approach satisfies all dimensions. In the following, we review existing consistency assessment approaches in Section 2.1 and partial FD discovery approaches in Section 2.2.

2.1 Consistency Assessment Approaches

Consistency is typically measured by determining whether a set of semantic rules holds for the data [3]. A major limitation of related work in our setting is that the relevant rules are typically assumed to be given: many existing approaches rely on predefined semantic rules (cf. Table 1) [10, 13, 22]. Association rule-based approaches introduce some automation by discovering implications between frequently co-occurring attribute value patterns. However, they remain use-case specific and often require human judgment to separate meaningful from coincidental rules [2, 14]. Approaches that are based on conditional functional dependencies (CFDs), i.e., FDs with pattern tableaux that restrict the tuples where a dependency holds, support consistency analysis with expressive constraints, but still have two limitations for our setting: they depend on domain knowledge and meaningful CFDs are difficult to discover automatically from dirty data since errors can invalidate CFDs that should hold otherwise [1, 25]. Heinrich et al. [12] incorporate uncertainty through probabilistic weighting, yet they continue to rely on externally provided rules, which are obtained through analyzing a reference dataset, conducting a study, or surveying domain experts. As shown in Table 1, no existing approach jointly offers automatic rule discovery (D1), rule discovery from dirty data (D2), automated weighting of rules (D3), and independence of human input (D4).

2.2 Functional Dependency Discovery

While the research community has invested a lot of work into the discovery of FDs in clean datasets (e.g., [20]), real-world data often contains errors, such as missing values or typos that prevent the discovery of semantically meaningful but slightly violated FDs. For this reason, several *relaxations* have been proposed for different types of data dependencies to tolerate dependency (a few) violations [6, 21]. A *partial* dependency is true only for a certain percentage of records; there is no quantification or description for the severity of the individual violations so that every violation counts the same. A *conditional* dependency is true for only those records that meet a dependency-specific condition, which is usually formulated as a pattern tableaux; the conditions provide more details about the semantics of the included/excluded records, but not every error can effectively be masked by a condition.

Approximating data dependencies, such as *matching*, *metric*, *similarity*, and *differential* dependencies, also ignore (minor) errors in values by considering also similar values as equal, i.e., matches; this makes them robust against certain types of errors and systematic noise, but their discovery is particularly expensive [23]. In this paper, we focus on partial dependencies, as these are the most versatile and generic form of relaxation.

To formalize pFDs, we define a relation instance T over a schema R as a set of tuples. For a tuple $t \in T$ and a set of attributes $X \subseteq R$, we denote the projection of t onto X as $t[X]$. We write attribute sets in capital letters from the end of the alphabet (e.g., X, Y) and single attributes in letters from the beginning of the alphabet (e.g., A, B). A functional dependency $X \rightarrow A$ holds iff for all tuples

Table 1: Comparison of related work along design dimensions relevant for automated rule discovery (D1–D4) and compliance with metric requirements (M1–M5). D1 = automatic rule discovery; D2 = rule discovery from dirty data; D3 = weighted rules; D4 = independence of human input; M1 = existence of minimum and maximum; M2 = interval-scaled metric; M3 = quality of configuration parameters and determination of metric values; M4 = sound aggregation; M5 = economic efficiency.

Work	Rule type & discovery requirements					Metric & metric requirements by Heinrich et al. [8]						Main limitation w.r.t. our goal
	Rule type	D1	D2	D3	D4	Metric (notations from [12])	M1	M2	M3	M4	M5	
Hinrichs [13]	Consistency rules	✗	✗	✓	✗	$\text{cons}(g) = \frac{1}{\sum_{n=1}^{N} w_n r_n(g) + 1}$	✗	✗	✓	✗	✗	Manually defined rules
Alpar & Winkelsträter [2]; Hipp et al. [14]	Association rules	✓	✗	✓	✗	$\text{cons}(t) = \sum_{n=1}^{N}(w_n^- r_n(t) + w_n^+(1 - r_n(t)))$ $(1 - h_n(t)) + \sum_{l=1}^{L} h_l(t)\, w_l^0$	✗	✗	✓	✗	✓	Use-case-specific; needs human review
Cordts; Pipino et al. [22]	Semantic rules	✗	✗	✗	✗	$\text{cons}(g) = 1 - \frac{\sum_{n=1}^{N} r_n(g)}{N}$	✓	✓	✓	✓	–	Manually defined rules
Heinrich et al. [9, 11]	Semantic rules	✗	✗	✗	✗	$\text{cons}(g) = \prod_{n=1}^{N}(1 - r_n(g))$	✓	✓	✓	✓	–	Manually defined rules
Abboura et al. [1]	Conditional FDs	~	~	~	✗	$\text{cons}(a) = \prod_{n=1}^{N} \text{support}(r_n)\text{conf}(r_n)$	✓	✓	–	–	–	Needs domain knowledge; rule discovery only on clean data
Wang et al. [25]	Conditional FDs	~	~	~	✗	$\text{incons}(D_B) = \frac{\lvert C_{\min}(D_B)\rvert}{S}$	✓	✓	–	–	–	Needs domain knowledge; rule discovery only on clean data
Heinrich et al. [12]	Uncertain rules	~	✓	✓	✗	$\text{cons}(\phi(t_j, a_i), r{:}A \to C) = p\text{-value}(X(r)$ $\sim B(\lvert\text{fr}(DB, A)\rvert, p(r)), \lvert\text{fr}(DB, A \wedge C)\rvert)$ *fr = fulfilling records*	✓	✓	✓	✓	–	Manually defined rule sources
This work	**Partial FDs**	✓	✓	✓	✓	$\text{CPFD}(\mathcal{F}) = \frac{\sum_{f \in \mathcal{F}} \lceil \omega(f) \cdot r(f) \rceil}{\sum_{f \in \mathcal{F}} \omega(f)}$	✓	✓	✓	✓	✓	–

$t_i, t_j \in T$: $t_i[X] = t_j[X] \Rightarrow t_i[A] = t_j[A]$. A partial FD relaxes this by requiring the condition to hold on only a fraction ρ of the data:

*Definition 2.1 (**Partial Functional Dependency**).* Let T be an instance of schema R and $X \subseteq R$ and $A \in R$ attributes of R. A *partial functional dependency* $T[X] \to_\rho T[A]$ (short $X \to_\rho A$) with $\rho \in (0, 1]$ holds iff at least $\lceil \rho \cdot |T| \rceil$ tuples support the FD $X \to A$, i.e., $\exists T' \subseteq T : (|T'| = \lceil \rho \cdot |T| \rceil) \wedge (\forall t_i, t_j \in T' : t_i[X] = t_j[X] \Rightarrow t_i[A] = t_j[A])$.

Because partial FD discovery can return many syntactically valid but semantically weak dependencies, prior work has proposed genuineness measures to distinguish between meaningful and coincidental dependencies. Several measures have been proposed to distinguish spurious from genuine dependencies [21]. In this work, we use the gpdep measure [21], which was identified as the most effective genuineness measure that remains computationally feasible, and adapt it as a weight function in Section 3.2.

3 AUTOMATED CONSISTENCY ASSESSMENT

Our goal is to fully automatically assess the consistency of a given relation dataset and, in particular, not rely on manually specified rules. To achieve this, our technique CPFD follows three steps: (i) automatic discovery of pFDs, (ii) automatic weighting of pFDs based on their statistical genuineness, and (iii) automatic aggregation of the weighted pFDs into a single consistency score.

3.1 Automatic pFD Discovery

To discover pFDs, we modified the existing algorithm HyFD [20], a hybrid FD discovery algorithm that combines sampling-based pruning with exact validation. While HyFD is designed for non-relaxed, exact FDs, our use case requires FDs that may be violated by a (small) number of tuples. We therefore changed the validation logic from binary validity to threshold-based validity.

Given a dataset with n records, a pFD $X \to_\rho A$ expresses that the pFD holds on at least $\rho \cdot n$ records. The goal of the discovery is to find pFD candidates with $\rho \geq \rho_{min}$ for some threshold $\rho_{min} \in (0, 1]$. The final dependency set is then obtained by selecting the most genuine candidates per right-hand side (see Section 3.2). So instead of invalidating an FD candidate with a single violating tuple pair, the pFD discovery needs to collect a certain number of pFD violating tuples to constitute that the pFD does not hold on $\rho_{min} \cdot n$ records. Thus, with n tuples, the maximum number of tolerable pFD violating tuples is $v_{max} = n - \lceil \rho_{min} \cdot n \rceil$.

The extension of HyFD with threshold-based validation capabilities affects only two components: sampling and validation. Both components now count the number of violating tuples per pFD candidate. If the number exceeds v_{max}, HyFD discards the candidate and, as usual, creates specialized pFD candidates with larger left-hand sides. Every new candidate carries over only a subset of violating tuples, namely those that still apply to the new candidate. The complex part in the validation is to identify the minimal number of violating tuples v to test if $v < v_{max}$. For example, if (t_1, t_2) and (t_1, t_3) are our violations, then the pFD holds on $\{t_1\}$ and $\{t_2, t_3\}$; the second constellation indicates the smaller number of violating tuples with $v = 1$ and needs to be identified. In the sampling component, our extended HyFD follows the idea of related work [16]. For a candidate $X \to_\rho A$, we sample tuple pairs from the equivalence classes of X, i.e., from groups of tuples that agree on X. For each sampled tuple pair, we compute the set of all attributes on which they agree, i.e., the values are equal. We use the sampling to estimate the error for this candidate and any candidate $Y \to A$ with $X \subseteq Y$. In the validation component, HyFD now counts the violations within each equivalence class computed

from the left-hand side of the pFD candidate. For each of these groups, the algorithm first finds the most frequent right-hand side value; then, it counts all values that are not part of the majority as violations. Notably, with increasing v_{max}, also the computation cost increases. The result is a set $\mathcal{F}$ of pFDs together with their partial scores ρ. Because partial discovery alone may leave underspecified or introduce accidental dependencies, we use $\rho \geq \rho_{min}$ only as a necessary condition and select the highest-weighted candidate for each right-hand-side attribute.

3.2 Automatic pFD Weighting by Genuineness

Because the number of pFDs grows exponentially with the size of the data and only a few pFDs actually reflect genuine semantics (e.g., *zip* $\rightarrow$ *city*), we suggest to weight each pFD by its genuineness — a task for which various solutions exist [21]. For a pFD $X \rightarrow_\rho A$, the probabilistic dependency $pdep(X, A)$ represents the conditional probability that two records agree on A if they agree on X. $E[pdep(X, A)]$ is the expected baseline if all values of A are randomly permuted while the values of X remain fixed and indicates a purely coincidental dependency. The *genuine* probabilistic dependency score $gpdep(X, A) = pdep(X, A) - E[pdep(X, A)]$ then measures how much stronger the observed dependency is compared to what would be expected by chance. Since $pdep(X, A) \leq 1$, the best possible improvement over the expected value is $1 - E[pdep(X, A)]$, which we use as a normalization factor. To weight our pFDs, we use the normalized score to $[0, 1]$:

$$\omega(X, A) = \max\left(0,\ \frac{pdep(X, A) - E[pdep(X, A)]}{1 - E[pdep(X, A)]}\right)$$

The normalization $\max(0, \cdot)$ ensures that coincidental dependencies weaker than the random baseline receive a weight of 0, while a perfectly genuine dependency receives a weight of 1.

3.3 Automatic Consistency Metric Calculation

Given the set $\mathcal{F} \neq \emptyset$ of pFDs $X \rightarrow_\rho A$ with their individual weights $\omega(X, A)$, we now aggregate these weights into a consistency score for the dataset. To this end, we first define a binary expression $r(X, A)$ that indicates whether a pFD is exact, i.e., non-violated:

$$r(X, A) = \begin{cases} 1 & \text{if } X \rightarrow_\rho A \text{ with } \rho = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Given $\omega(X, A)$ and $r(X, A)$, we define our consistency metric CPFD as a weighted linear aggregation:

$$\text{CPFD}(\mathcal{F}) = \frac{\sum_{X \rightarrow_\rho A \in \mathcal{F}} \omega(X, A) \cdot r(X, A)}{\sum_{X \rightarrow_\rho A \in \mathcal{F}} \omega(X, A)}$$

Note that CPFD is expected to capture consistency – how well does the data follow structural rules – and not accuracy – how clean and valid are the relational records. For this reason, CPFD essentially calculates the proportion of satisfied functional dependencies, i.e., satisfied FDs $\sum_{X \rightarrow_\rho A \in \mathcal{F}} r(X, A)$ divided by all FDs $\sum_{X \rightarrow_\rho A \in \mathcal{F}} 1$. But because not every (automatically discovered) FD is equally meaningful, we weight all FDs by their gpdep-based $\omega(X, A)$ scores, so that genuine FDs have a higher impact on the score than spurious FDs.

3.4 Diagnostic Consistency Analysis

While CPFD provides a single global consistency score (which is often required for data governance in practice), the underlying computation contains more information. We expose this information as *explanations* of the score. These explanations answer three questions: (i) which dependencies are violated, (ii) how severe are the violations, and (iii) where do the violations occur?

Which dependencies are violated? The violated dependencies are directly identified by the indicator $r(X, A)$, i.e., all pFDs with $r(X, A) = 0$. However, this set alone does not explain the score. On real-world data, the set of violated dependencies can be large, and most of its members are dependencies with low genuineness that barely affect CPFD. We therefore quantify the contribution of each individual pFD to the score deficit as

$$\delta(X, A) \ = \ \frac{\omega(X, A) \cdot \big(1 - r(X, A)\big)}{\sum_{Y \rightarrow_\rho B \in \mathcal{F}} \omega(Y, B)}$$

with

$$\sum_{X \rightarrow_\rho A \in \mathcal{F}} \delta(X, A) \ = \ 1 - \text{CPFD}(\mathcal{F}).$$

The global consistency score can thus be decomposed into the dependencies that are used to compute it, which makes the score itself explainable: A score of 0.73 indicates that exactly satisfied dependencies account for 73% of the total genuineness weight, and the remaining 27% can be explained as a list of violated dependencies ranked in descending order of $\delta(X, A)$.

How severe are the violations? The binary function $r(X, A)$ is designed to measure the consistency of the structural rules. The partial degree ρ of a pFD, in contrast, describes the extent to which the FD holds and thus explains how far a violated dependency is from being satisfied. In our implementation, we also expose the ρ of a pFD as a diagnostic quantity. Furthermore, we can aggregate this information into a global diagnostic score:

$$\text{CPFD}_\rho(\mathcal{F}) = \frac{\sum_{X \rightarrow_\rho A \in \mathcal{F}} \omega(X, A) \cdot \rho}{\sum_{X \rightarrow_\rho A \in \mathcal{F}} \omega(X, A)}$$

The score summarizes how closely the data satisfies the genuine rules. Therefore, the two scores answer different questions. CPFD answers *whether* genuine rules hold exactly, whereas CPFD_ρ answers *to what degree* they hold.

Where do the violations occur? Because the validation of a pFD computes violations per equivalence class, we can further report, for each violated dependency, the affected equivalence classes and the violating tuples. This shows the user exactly which records cause the violations.

Together, the three questions turn the single number CPFD into an explainable DQ assessment. First, CPFD provides the user with a consistency score to assess whether the data conforms to the given rules. Second, if there are consistency problems, the user can inspect the pFDs with the largest $\delta(X, A)$ to see which dependencies are responsible, and their ρ values, summarized by CPFD_ρ, indicate how severely each is violated. Lastly, the user can identify the conflicting equivalence classes and the individual records that need repair.

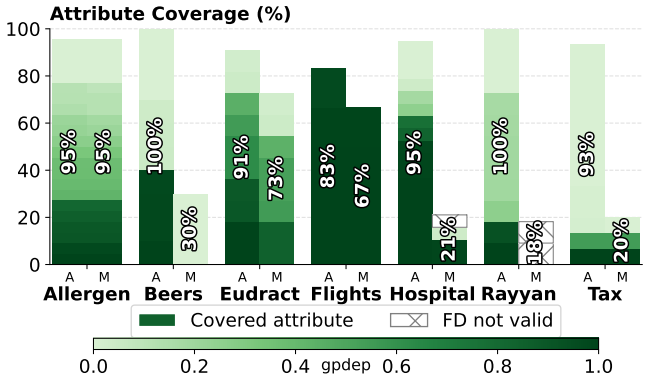

Figure 2: Attribute coverage per dataset for our automatically discovered pFDs (left bar [A]) vs. manually defined FDs by dataset owners or cleaning experts (right bar [M]). Color intensity represents the statistical genuineness (gpdep) of the FDs. Striped bars indicate FDs that do not hold on the dataset.

Dataset	Functional Dependencies	Runtime [s]
Allergen (E) [4]	code → nuts,…	11.3
Beers (O) [17]	brewery id → brewery name; brewery id → city; brewery id → state	0.04
Eudract (E) [4]	eudract_number → open, …; single_blind,open → double_blind; double_blind,open → single_blind; single_blind,double_blind → open	0.07
Flights (O) [17]	flight → act. dep. time; flight → act. arr. time; flight → sched. dep. time; flight → sched. arr. time	0.01
Hospital (O) [17]	city → zip; city → county; zip → city; zip → state; zip → county; county → state	0.91
Rayyan (O) [17]	journal abbreviation → journal title; journal abbreviation → journal issn; journal issn → journal title	0.01
Tax (O) [17]	zip → city; zip → state; first name → gender; area code → state	106.7

Table 2: FDs provided by dataset owners ("(O)" in blue) [17] and data cleaning experts ("(E)" in green) [4], and automatic profiling times for all pFDs.

4 EVALUATION

We evaluate CPFD along three aspects: the quality of our pFD discovery, the validity of the metric, and the compliance with DQ metric requirements [8]. The code, datasets, and additional experimental results are publicly available[2]. To evaluate whether automatically discovered pFDs can replace manually curated pFDs without loss of quality, Section 4.1 compares their quality against the quality of manually provided pFDs by dataset owners and data cleaning experts. Section 4.2 validates CPFD against a synthetically polluted dataset with known ground truth and compares it with rule-based and probabilistic consistency metrics. Finally, Section 4.3 demonstrates that CPFD fulfills all five requirements for DQ metrics defined by Heinrich et al. [8].

Experimental Setup. We run all our experiments on a desktop computer with an AMD Ryzen 7 9800X3D CPU (8 physical cores), 64GB 6000MHz DDR5 RAM, and a Samsung 980 PRO SSD. CPFD and the pFD discovery algorithm are implemented in Java and executed on an OpenJDK JVM 23. We perform our experiments on seven datasets from related work: Allergen, Beers, Eudract, Flights, Hospital, Rayyan, and Tax [4, 17]. The datasets cover diverse domains, are commonly used in data cleaning research, and include a gold standard with manually defined FDs. For the discovery of the FDs with our HyFD extension, we use a threshold of $\rho_{min} = 0.95$.

4.1 Quality of Discovered pFDs

In this section, we evaluate the quality of our automatic pFD discovery technique compared to manually defined FDs along three aspects: (i) efficiency and scalability, (ii) coverage and semantic quality, and (iii) objectivity and robustness.

Efficiency and scalability: To assess scalability, we compare the effort required for manual rule definition against the runtime of our automated approach across all seven datasets. Table 2 contains the FDs for each dataset, which were manually defined by dataset owners and data cleaning experts [4, 17]. Note that the search space for possible FDs grows exponentially with the number of attributes, which makes exhaustive manual discovery infeasible even for moderately sized schemas. In contrast, our approach discovers pFDs and computes their genuineness with an average runtime of 17 s per dataset (maximum: 106.7 s for Tax, cf. Table 2). Compared to the original HyFD, our discovery was faster on 5 datasets (~1.2×–21×) and slower on 2 (~4.8× and 62×). This enables frequent re-evaluation and makes the approach suitable for dynamic datasets where the schema and data evolve over time, or where consistency is remeasured periodically.

Attribute coverage and FD quality: To assess coverage and quality, we compare automatically discovered and manually defined FDs in Figure 2, which shows the fraction of attributes covered per dataset and the statistical genuineness (gpdep) of the underlying FDs. Our approach identifies more dependencies, covers a broader set of attributes, and produces FDs with higher gpdep scores. In contrast, a substantial portion of manually defined FDs does not hold on the datasets, indicating outdated or incorrect domain knowledge. For example, in the Hospital dataset, the FD $city \rightarrow zip$ does actually not hold in the gold standard, as cities can have multiple zip codes. This is also reflected in its low gpdep score. Similarly, in the Rayyan dataset, a large portion of the manually defined FDs does not hold, making them misleading or irrelevant.

Objectivity and robustness: To assess objectivity, we compare the FD sets defined independently by dataset owners and data cleaning experts for the same datasets [4, 17]. Manual rule definition is inherently subjective, as different experts have different views on the data. For example, the dataset owners of the Hospital dataset defined only 6 FDs, while the data cleaning expert defined 15 FDs, with an overlap of only 1 FD. For the Flights dataset, there is no overlap at all. In contrast, our approach derives the dependencies directly from the data and is deterministic for fixed input data and

[2] https://github.com/SeegerM/cpfd-reproducibility

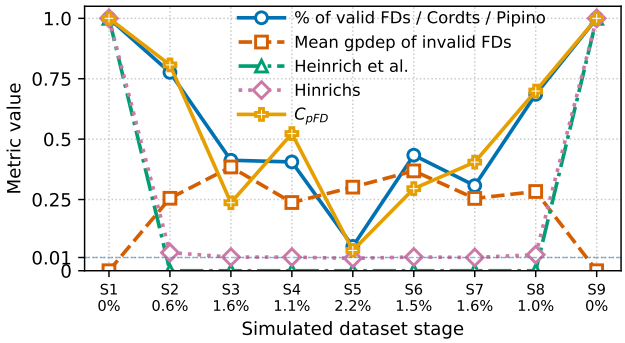

Figure 3: Metric behavior across simulated data pollution levels of the Hospital dataset.

parameter settings. This improves reproducibility and removes subjective decisions about which rules should be included. At the same time, the use of pFDs makes the discovery robust to violations up to the chosen threshold ρ. Thus, the automatically discovered pFDs provide a more objective and noise-tolerant basis for consistency assessment than manually specified strict FDs.

4.2 Comparison of CPFD to Other Metrics

To validate that CPFD accurately reflects "fitness for use," and captures violations of semantic rules, we evaluate it on systematically polluted variants of the clean datasets. To pollute the dataset, we use the *tab_err* framework by Jung et al. [15]. For each dataset, we inject errors into randomly selected attributes at rates between 0% and 5%, covering a range of realistic error types across textual, numerical, and categorical attributes. In total, we generate approximately 2,000 corrupted variants per dataset. For a fair comparison, all metrics in this section are evaluated on the same dependency set per dataset. Therefore, differences between metric values are caused by the aggregation and weighting strategy of the respective metric and not by using different input rules.

Figure 3 illustrates the behavior of different consistency metrics across a random simulated pollution stage of the Hospital dataset. The metrics by Heinrich et al. [12] and Hinrichs [13] collapse to values close to zero once dependencies are violated (S2–S8), and thus cannot differentiate intermediate pollution stages.

CPFD follows the general trend of rule satisfaction, but introduces an important distinction through the genuineness weighting. This effect can be observed between stages $S3$ and $S4$, and again between $S6$ and $S7$. In both cases, the percentage of valid FDs decreases only slightly, while the mean gpdep of invalid FDs decreases as well. This means that the additionally violated dependencies are, on average, less statistically meaningful. Consequently, these violations receive lower weights in CPFD, and the final score can increase even though the unweighted percentage of valid FDs decreases.

We further analyze this behavior across all polluted dataset variants by computing Pearson correlations between the dataset-level metrics in Table 3. We find that CPFD correlates strongly with the related works of Hinrichs [13] (0.885) and Cordts, Pipino et al. [22], as well as the percentage of valid FDs (0.901). This is to be expected because four approaches also make a binary distinction between whether a rule holds or not. However, CPFD assigns higher impact to violations of statistically genuine dependencies and lower impact to violations of weaker or more coincidental dependencies.

Metric	Correlation with CPFD
% valid weighted FDs	0.914
Cordts / Pipino et al. [22] / % valid FDs	0.901
Hinrichs [13]	0.885
Mean gpdep	0.204
Heinrich et al. [12]	0.817

Table 3: Pearson correlation between CPFD and related consistency metrics across all polluted dataset variants.

Overall, we show that CPFD behaves consistently with established rule-based metrics while providing a more fine-grained assessment through genuineness-weighted dependency satisfaction.

Interpreting the final score of a DQ metric is not trivial (as also pointed out by [5]): Which score constitutes good data is domain-dependent, and Figure 2 shows that some datasets simply lack the structural information to discover meaningful rules. We therefore refrain from setting a universal threshold and instead recommend interpreting CPFD comparatively: Monitoring a dataset over time, comparing versions before and after cleaning, or ranking alternative data sources. Comparative interpretation is well-defined because the score is explainable. A decreasing score means that violated dependencies account for a larger share of the total genuineness weight, and the explanations from Section 3.4 identify the responsible dependencies, equivalence classes, and records. The combination of automated DQ assessment and explainable scores is increasingly demanded by regulations such as the AI Act [7]. Organizations must measure and report the quality of large, evolving datasets, which is only feasible if the assessment is done *automatically* and each reported score can be *explained*.

4.3 Compliance with Metric Requirements

We assess CPFD according to the five requirements (M1–M5) for DQ metrics by Heinrich et al. [8] (as summarized in Table 1):

(M1) Existence of minimum and maximum metric values: CPFD is strictly bounded in the interval $[0, 1]$, where 1 indicates that all considered dependencies are fully satisfied according to their weights, and 0 indicates that none of the selected weighted dependencies is exactly satisfied. Guidance on interpreting intermediate values is provided in Section 4.2.

(M2) Interval-scaled metric values: CPFD is a weighted arithmetic mean of binary rule-satisfaction values and is hence interval-scaled. Differences in the score correspond to differences in the total genuineness weight of satisfied dependencies, meaning that each dependency contributes proportionally to its normalized weight.

(M3) Quality of the configuration parameters and the determination of the metric values: CPFD minimizes subjective input by automatically discovering dependencies from the data, and determining their importance through statistical measures (gpdep) rather than manually assigned weights. The only configurable parameter is the threshold ρ_{min}, which controls the tolerance for violations

and can be set in a principled manner. This reduces user bias and ensures reproducibility across datasets and runs (see Section 4.1).

(M4) Sound aggregation of the metric values: CPFD aggregates consistency values using a weighted arithmetic mean based on the statistical genuineness of dependencies. For a fixed dependency set $\mathcal{F}$ and fixed weights ω, this aggregation is monotonic, meaning that improvements in individual dependencies cannot decrease the overall score, and the normalization of weights ensures stability regardless of the number of dependencies considered. Heinrich et al. [8] explicitly identify weighted arithmetic means as a sound aggregation function.

(M5) Economic efficiency of the metric: The aggregation of CPFD is computationally efficient ($O(n)$ in the number n of dependencies), assuming precomputed violations and weights. More importantly, the automated pFD discovery and weighting eliminates the need for manual rule specification entirely and makes the metric suitable for large-scale and evolving datasets where manual approaches are economically infeasible (see Section 4.1). Similar arguments for automated rule extraction have been discussed in [8].

Overall, CPFD fulfills all five requirements while enabling fully automated consistency assessment, in contrast to prior approaches.

5 CONCLUSION

In this paper, we presented CPFD, a fully automated and explainable technique for assessing the consistency of a relational dataset based on discovered pFDs and gpdep-based genuineness weighting. Unlike prior work that relies on expert-defined rules, external sources, or manually assigned weights, our approach obtains the rules and their genuineness directly from the data. This technique has the advantage of not depending on subjective human assessment or external input, and therefore improves the scalability, reproducibility, and objectivity of the consistency assessment, especially in scenarios where domain knowledge is incomplete, unavailable, or costly to obtain. We demonstrated that our metric satisfies all requirements for DQ metrics proposed in [8], the pFD discovery remains robust to dirty data, and outperforms manually defined rule sets in coverage and quality on real-world datasets. Overall, this work demonstrates that automated, data-driven consistency assessment is not only feasible but also effective, and takes a significant step toward more autonomous and scalable DQ management.

The main limitation of CPFD is that FDs, while widely used, do not capture all semantic relationships in complex datasets. Extending CPFD to denial constraints or multi-valued dependencies is a promising direction for future work.

ACKNOWLEDGMENTS

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) under the Project numbers 560957958 and 560743026.

REFERENCES

[1] Asma Abboura, Soror Sahri, Latifa Baba-hamed, Mourad Ouziri, and Salima Benbernou. 2016. Quality-Based Online Data Reconciliation. *ACM Trans. Internet Techn.* 16, 1 (2016), 3:1–3:21.
[2] Paul Alpar and Sven Winkelsträter. 2014. Assessment of data quality in accounting data with association rules. *Expert Syst. Appl.* 41, 5 (2014), 2259–2268.
[3] Carlo Batini and Monica Scannapieco. 2016. *Data and Information Quality - Dimensions, Principles and Techniques.* Springer.
[4] Toon Boeckling and Antoon Bronselaer. 2025. Cleaning data with Swipe. *ACM J. Data Inf. Qual.* 17, 1 (2025), 1–29.
[5] Antoon Bronselaer, Robin De Mol, and Guy De Tré. 2018. A Measure-Theoretic Foundation for Data Quality. *IEEE Trans. Fuzzy Syst.* 26, 2 (2018), 627–639.
[6] Loredana Caruccio, Vincenzo Deufemia, and Giuseppe Polese. 2016. Relaxed Functional Dependencies - A Survey of Approaches. *IEEE Trans. Knowl. Data Eng.* 28, 1 (2016), 147–165.
[7] European Parliament. 2024. Artifical Inteligence Act. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32024R1689 Version from 2024-07-12.
[8] Bernd Heinrich, Diana Hristova, Mathias Klier, Alexander Schiller, and Michael Szubartowicz. 2018. Requirements for Data Quality Metrics. *ACM J. Data Inf. Qual.* 9, 2 (2018), 12:1–12:32.
[9] Bernd Heinrich, Marcus Kaiser, and Mathias Klier. 2007. Metrics for measuring data quality — foundations for an economic oriented management of data quality. *2nd International Conference on Software and Data Technologies (ICSOFT)* (2007).
[10] Bernd Heinrich and Mathias Klier. 2015. Datenqualitätsmetriken für ein ökonomisch orientiertes Qualitätsmanagement. In *Daten-und Informationsqualität: Auf dem Weg zur Information Excellence.* Springer, 49–67.
[11] Bernd Heinrich and Mathias Klier. 2015. Metric-based data quality assessment - Developing and evaluating a probability-based currency metric. *Decis. Support Syst.* 72 (2015), 82–96.
[12] Bernd Heinrich, Mathias Klier, Alexander Schiller, and Gerit Wagner. 2018. Assessing data quality - A probability-based metric for semantic consistency. *Decis. Support Syst.* 110 (2018), 95–106.
[13] Holger Hinrichs. 2002. *Datenqualitätsmanagement in Data-Warehouse-Systemen.* Ph.D. Dissertation. University of Oldenburg, Germany.
[14] Jochen Hipp, Markus Müller, Johannes Hohendorff, and Felix Naumann. 2007. Rule-Based Measurement Of Data Quality In Nominal Data.. In *ICIQ.* 364–378.
[15] Philipp Jung, Sebastian Jäger, Nicholas Chandler, and Felix Biessmann. 2025. Towards Realistic Error Models for Tabular Data. *ACM J. Data Inf. Qual.* 17, 4 (2025), 28:1–28:27.
[16] Sebastian Kruse and Felix Naumann. 2018. Efficient discovery of approximate dependencies. *Proceedings of the VLDB Endowment* 11, 7 (2018), 759–772.
[17] Mohammad Mahdavi and Ziawasch Abedjan. 2020. Baran: Effective Error Correction via a Unified Context Representation and Transfer Learning. *Proc. VLDB Endow.* 13, 11 (2020), 1948–1961.
[18] Sedir Mohammed, Lukas Budach, Moritz Feuerpfeil, Nina Ihde, Andrea Nathansen, Nele Noack, Hendrik Patzlaff, Felix Naumann, and Hazar Harmouch. 2025. The effects of data quality on machine learning performance on tabular data. *Information Systems* 132 (2025), 102549.
[19] Ken Orr. 1998. Data quality and systems theory. *Commun. ACM* 41, 2 (1998), 66–71.
[20] Thorsten Papenbrock and Felix Naumann. 2016. A Hybrid Approach to Functional Dependency Discovery. In *SIGMOD Conference.* ACM, 821–833.
[21] Marcel Parciak, Sebastiaan Weytjens, Niel Hens, Frank Neven, Liesbet M. Peeters, and Stijn Vansummeren. 2025. Measuring approximate functional dependencies: a comparative study. *VLDB J.* 34, 4 (2025), 56.
[22] Leo Pipino, Yang W. Lee, and Richard Y. Wang. 2002. Data quality assessment. *Commun. ACM* 45, 4 (2002), 211–218.
[23] Shaoxu Song, Fei Gao, Ruihong Huang, and Chaokun Wang. 2020. Data dependencies extended for variety and veracity: A family tree. *IEEE Transactions on Knowledge and Data Engineering* 34, 10 (2020), 4717–4736.
[24] Yair Wand and Richard Y. Wang. 1996. Anchoring Data Quality Dimensions in Ontological Foundations. *Commun. ACM* 39, 11 (November 1996), 86–95.
[25] Hao Wang, Jianzhong Li, and Hong Gao. 2016. Data Inconsistency Evaluation for Cyberphysical System. *Int. J. Distributed Sens. Networks* 12, 9 (2016), 9496878.