Journal of Sustainability, Policy, and Practice

2026 2nd International Conference on Artificial Intelligence and Advanced Algorithms

Article

Empirical Evaluation of Constraint Discovery Techniques for Business Logic Consistency Verification in Payroll Data Migration

Hao Cao [1,*] and Muyu Liu [2]

[1] Master of Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA
[2] Center for Applied Statistics and School of Statistics, Renmin University of China, Beijing, China
* Correspondence: Hao Cao, Master of Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA

Abstract: Enterprise payroll system migrations demand rigorous verification that business logic---salary calculations, grade-based rules, and overtime policies---is faithfully preserved in the target environment. Manual testing remains the prevailing practice, yet it scales poorly and is prone to human oversight. Automated constraint discovery offers a promising alternative by extracting latent business rules directly from source data and applying them as verification checks on migrated records. This paper presents an empirical comparative evaluation of three constraint discovery families---functional dependencies (FDs), conditional functional dependencies (CFDs), and denial constraints (DCs)---for detecting business logic inconsistencies introduced during payroll data migration. Experiments are conducted on two real-world datasets: the NYC Citywide Payroll Data (approximately 600,000 records per fiscal year, 17 attributes) and the IBM HR Analytics dataset (1,470 records, 35 attributes). A controlled error injection methodology simulating three categories of migration inconsistencies at four injection rates (1%, 3%, 5%, 10%) is employed to construct ground truth labels. Results indicate that CFD-based discovery achieves the most favorable precision-recall balance (F1 = 0.778 at a 5% error rate on NYC data), while DC discovery attains the highest recall (0.81) at the cost of reduced precision. FD discovery, though computationally efficient, exhibits limited recall due to its inability to encode context-sensitive rules. These findings provide actionable guidance for practitioners selecting automated verification strategies in enterprise data migration projects.

Keywords: constraint discovery; business logic verification; payroll data migration; data quality

Received: 17 March 2026
Revised: 25 April 2026
Accepted: 08 May 2026
Published: 13 May 2026

Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

1.1. Background and Motivation

Large-scale enterprise resource planning (ERP) migrations---particularly transitions from legacy on-premise payroll systems to cloud-based platforms---have become a recurring operational challenge across public and private sectors. During such migrations, the faithful transfer of business logic governing salary computation, tax withholding, overtime policies, and deduction hierarchies is as critical as the transfer of raw data itself. A single misconfigured rule can propagate systematic errors affecting thousands of employee records, generating compliance violations and financial losses. The scale of this challenge is not trivial: municipal payroll systems routinely manage hundreds of thousands of employee records with dozens of interdependent attributes encoding complex compensation structures, benefit eligibility criteria, and regulatory constraints.

Relational data profiling techniques, as surveyed by Abedjan et al. encompass a rich family of methods capable of extracting structural patterns---including functional dependencies, unique column combinations, and inclusion dependencies---directly from data instances [1]. These patterns encode implicit business rules that can serve as verification oracles during migration. Automated data quality verification at industrial scale, as demonstrated by Schelter et al. further motivates the need for declarative constraint-based approaches that eliminate reliance on manual test case authoring [2]. The convergence of these two research threads---automated constraint discovery and scalable data validation---creates an opportunity to transform payroll migration verification from a labor-intensive manual process into a data-driven automated procedure [3].

Among the available constraint formalisms, three families are particularly relevant to payroll domains. Conditional functional dependencies (CFDs), introduced by Bohannon et al. extend classical FDs with value-binding conditions, enabling context-sensitive rules such as department-specific salary thresholds. Denial constraints (DCs), formalized for automated discovery by Chu et al. generalize FDs, order dependencies, and unique constraints into a single expressive language capable of encoding cross-tuple comparisons [4]. Classical FDs, while less expressive, remain the most computationally tractable option and benefit from mature discovery algorithms.

1.2. Research Questions and Contributions

1.2.1. Research Questions

This study investigates three research questions grounded in the practical needs of payroll migration verification. RQ1 asks how effectively FD, CFD, and DC discovery methods can automatically identify business logic rules embedded in enterprise payroll data [5]. RQ2 examines which constraint family achieves the most favorable precision-recall tradeoff for detecting inconsistencies introduced during data migration. RQ3 evaluates how these methods scale with increasing data volume and attribute complexity in payroll migration scenarios. Each question targets a distinct facet of the verification problem: rule expressiveness, detection accuracy, and operational feasibility.

1.2.2. Scope and Contributions

The scope of this work is strictly empirical: no new discovery algorithm or detection technique is proposed. The contributions are threefold. The study provides the first systematic comparison of constraint discovery techniques applied specifically to payroll data migration verification, extending a well-established comparative evaluation tradition in the constraint discovery literature [6,7]. It introduces a payroll-domain error injection methodology that generates ground-truth migration inconsistencies spanning value perturbation, cross-field dependency violation, and semantic logic corruption. It produces actionable practitioner guidelines mapping data characteristics and verification objectives to recommended constraint families.

2. Related Work

2.1. Integrity Constraint Discovery

2.1.1. Functional and Denial Constraint Discovery

FD discovery has matured through two decades of algorithmic development, yielding algorithms spanning lattice-traversal, difference-set, and hybrid strategies. The first comprehensive experimental comparison of seven major FD discovery algorithms---classifying them into lattice-traversal, difference/agree-set, and dependency-inference approaches---was conducted by Papenbrock et al. providing practical guidance for algorithm selection based on dataset characteristics [8,9]. On the more expressive end, Pena et al. accelerated DC enumeration through parallel evidence-set computation and inverted-index pruning, consistently outperforming all prior DC discovery algorithms on both real and synthetic benchmarks [10]. For real-world data that inevitably contains noise, Kruse and Naumann introduced Pyro, an algorithm for discovering approximate functional dependencies and approximate unique column combinations that tolerates a

specified error threshold, making it suitable for enterprise data where strict dependencies rarely hold perfectly [11].

2.1.2. Conditional Dependencies and Context-Sensitive Rules

CFDs bridge the gap between unconditional FDs and the context-dependent rules prevalent in enterprise domains. Chiang and Miller addressed the practical challenge of automatically discovering CFDs from data, proposing algorithms that identify both the dependency structure and the conditional patterns simultaneously [8]. This capability is essential for payroll verification, where rules differ across departments, job grades, and bargaining units. More recently, Fariha et al. introduced conformance constraints---arithmetic relationships across numerical attributes---that capture quantitative business rules (such as salary-to-grade ratios) not expressible through traditional FDs or CFDs [12]. The conformance constraint paradigm aligns closely with payroll verification needs, where numerical relationships between compensation fields often follow domain-specific formulas.

2.2. Data Quality Verification and Error Detection

The operationalization of discovered constraints for data quality purposes has been addressed by several end-to-end cleaning and detection platforms. Dallachiesa et al. developed NADEEF, an extensible platform that handles heterogeneous quality rules---including FDs, CFDs, and matching dependencies---through a unified programming interface for both detection and repair [13]. Rekatsinas et al.proposed HoloClean, which unifies integrity constraints with external data and quantitative statistics via probabilistic inference, achieving substantial F1 improvements over prior cleaning methods while scaling to millions of tuples [14]. Mahdavi et al. took a different approach with Raha, a configuration-free error detection method that generates feature vectors from diverse detection strategies and requires only approximately 20 labeled tuples for semi-supervised classification, eliminating the need for user-specified rules [15]. These platforms demonstrate the practical viability of constraint-based quality enforcement, though none has been systematically evaluated in the specific context of payroll data migration [16].

Anomaly Detection and Automated Testing: Tabular anomaly detection offers a rule-free alternative to constraint-based verification. Han et al. established ADBench, the most comprehensive anomaly detection benchmark to date, evaluating 30 algorithms across 57 datasets and revealing that the relative effectiveness of methods varies significantly with anomaly type---a finding directly relevant to the heterogeneous nature of migration errors [17]. From the software engineering perspective, formal verification of business process models and automated test oracle generation represent complementary paradigms that approach the verification challenge from process-level and code-level perspectives, respectively [18]. While these methods address related problems, their application to data-level consistency checking during enterprise migrations remains underexplored, motivating the empirical investigation presented here.

3. Experimental Setup

3.1. Datasets and Preprocessing

Two publicly available datasets with payroll and human resource attributes are used in this study. The NYC Citywide Payroll Data, published by the NYC Office of Payroll Administration under a public domain license, contains municipal employee compensation records spanning fiscal years 2014 through 2024. Each fiscal year comprises approximately 600,000 records across 17 attributes, including Agency Name, Title Description, Base Salary, Pay Basis, Regular Hours, Regular Gross Paid, OT Hours, Total OT Paid, Total Other Pay, and Leave Status [19]. The dataset captures the full complexity of municipal payroll operations, with employees spanning more than 80 agencies and hundreds of distinct title descriptions. The IBM HR Analytics Employee Attrition and Performance dataset, released by IBM under a CC0 public domain license and hosted on

Kaggle, contains 1,470 synthetic employee records with 35 attributes encompassing compensation (MonthlyIncome, DailyRate, HourlyRate, MonthlyRate), job characteristics (JobLevel, JobRole, Department), and career trajectory (YearsAtCompany, YearsInCurrentRole, TotalWorkingYears).

Preprocessing for the NYC dataset involves selecting fiscal year 2023 records, removing name fields to retain 12 analysis-relevant attributes, and converting Pay Basis values to a standardized enumeration. Records with missing values in key compensation fields (1.3% of total records) are excluded to ensure that observed violations reflect injected errors rather than data completeness issues. Preprocessing for the IBM dataset involves removing constant-value columns (StandardHours, EmployeeCount, Over18) and encoding categorical attributes [20]. Table 1 summarizes the characteristics of both datasets after preprocessing.

Table 1. Dataset Characteristics After Preprocessing

Property	NYC Payroll (FY2023)	IBM HR Analytics
Records	604,182	1,470
Attributes	12	32
Numerical attributes	7	21
Categorical attributes	5	11
Missing value rate	1.3%	0.0%
Source	NYC Open Data (data.cityofnewyork.us)	Kaggle (IBM)
License	Public Domain	CC0

Data source: NYC Office of Payroll Administration; IBM via Kaggle.

3.2. Constraint Discovery Techniques under Comparison

3.2.1. Dependency-Based Constraint Discovery

Three constraint discovery methods, each representing a distinct constraint family, are evaluated. FD discovery is implemented using the HyFD algorithm, a hybrid approach combining lattice traversal with difference-set validation that adaptively switches strategies based on data characteristics. HyFD accepts a relational table as input and outputs a minimal cover of all holding functional dependencies [21,22]. On the NYC dataset, an FD such as {PayBasis} → {RegularHours} encodes the rule that salaried employees uniformly work a standard number of hours. CFD discovery follows the approach of Chiang and Miller, producing conditional rules that bind specific attribute values---such as [Department = "FIRE", PayBasis = "per Annum"] → BaseSalary ≥ 42,510---capturing department-specific pay thresholds invisible to unconditional FDs. DC discovery is implemented using Hydra, a sampling-based algorithm whose linear-time evidence-set computation overcomes the quadratic bottleneck of earlier DC discovery methods [23]. A representative DC on the IBM dataset takes the form $\neg(t_1.\text{JobLevel} > t_2.\text{JobLevel} \wedge t_1.\text{MonthlyIncome} < t_2.\text{MonthlyIncome} \wedge t_1.\text{Department} = t_2.\text{Department})$, encoding the expectation that within the same department, higher job levels correspond to higher compensation. All three methods operate in an unsupervised manner on the pre-migration source data, producing constraint sets that are subsequently applied as verification checks on the migrated dataset [24].

3.2.2. Statistical and ML-Based Baselines

Two baseline methods are included for comparative context. The statistical baseline (Deequ-Manual) employs a manually authored set of 45 domain-specific constraints codified in the Deequ declarative validation API, representing an upper-bound reference that assumes perfect domain knowledge and requires considerable expert effort to construct. The machine learning baseline (ICL) applies the internal contrastive learning method of Shenkar and Wolf for unsupervised anomaly detection on the pre-migration

data distribution, flagging migrated records that deviate from learned distributional patterns without explicit rule formulation [25]. This method represents the class of learning-based anomaly detectors that do not require dataset-specific parameter tuning.

3.3. Evaluation Protocol

3.3.1. Error Injection Strategy

A controlled error injection methodology is employed to simulate migration inconsistencies, producing labeled ground truth for quantitative evaluation. The design follows the principle that consistency and accuracy must be evaluated jointly, as migration errors can affect both structural conformance and numerical correctness [26]. Three error categories are defined. Value perturbation errors randomly modify numerical payroll fields (Base Salary, Regular Gross Paid, OT Hours, Total OT Paid) by multiplicative factors uniformly sampled from $[0.5, 0.8] \cup [1.2, 2.0]$, breaking arithmetic relationships between related fields. Cross-field dependency errors decouple related attributes---replacing Pay Basis with a randomly selected alternative while leaving Regular Hours unchanged, or altering Department without adjusting Base Salary to match the new department's pay range. Semantic logic errors introduce violations of compound business rules---setting OT Hours > 0 with Leave Status = "ON LEAVE" or assigning Base Salary below the minimum threshold for a given Title Description and Pay Basis combination [27]. Error categories are injected in equal proportions at four aggregate rates: $\varepsilon \in \{1\%, 3\%, 5\%, 10\%\}$ of total records. Each injection is repeated across five random seeds to mitigate sampling variance. Table 2 details the injection configuration.

Table 2. Error Injection Configuration

Error Category	Proportion	Target Attributes (NYC)	Target Attributes (IBM)
Value perturbation	1/3 of ε	BaseSalary, RegularGrossPaid, TotalOTPaid	MonthlyIncome, DailyRate, HourlyRate
Cross-field dependency	1/3 of ε	PayBasis ↔ RegularHours, Department ↔ BaseSalary	JobLevel ↔ MonthlyIncome, Department ↔ JobRole
Semantic logic	1/3 of ε	OTHours vs. LeaveStatus, BaseSalary vs. TitleDescription	YearsAtCompany vs. YearsInCurrentRole, PerformanceRating vs. PercentSalaryHike

Each injection rate is repeated with five random seeds; reported metrics are averaged across seeds.

3.3.2. Evaluation Metrics

Detection performance is measured through precision (fraction of flagged records that are genuinely injected errors), recall (fraction of injected errors successfully flagged), and F1-score (harmonic mean of precision and recall), computed at the record level. Rule discovery effectiveness is assessed by two quantities: the total number of raw constraints discovered and the number of meaningful constraints remaining after manual expert filtering---where meaningful is defined as encoding a genuine payroll business rule rather than a spurious statistical artifact. Computational efficiency is measured as wall-clock

runtime in seconds on a workstation equipped with an Intel Xeon W-2255 processor (10 cores, 3.7 GHz) and 64 GB RAM, running Ubuntu 22.04.

4. Results and Analysis

4.1. Rule Discovery Effectiveness

Coverage and Precision of Discovered Rules: Table 3 reports the number of raw and meaningful constraints discovered by each constraint family on both datasets. On the NYC Payroll dataset, HyFD produced 847 raw FDs, of which 23 (2.7%) survived expert filtering as genuine payroll rules. CFD-Discover generated 156 candidate CFDs with 31 (19.9%) deemed meaningful---the highest filtering precision among the three families. Hydra discovered 2,341 raw DCs, yet only 18 (0.8%) were retained, reflecting the combinatorial explosion inherent in pairwise tuple predicates. The IBM HR dataset exhibited similar patterns at a smaller scale: 312 raw FDs yielding 19 meaningful rules, 89 CFDs yielding 27 meaningful rules, and 1,156 DCs yielding 14 meaningful rules.

Table 3. Rule Discovery Statistics by Constraint Family

Method	Dataset	Raw Rules	Meaningful Rules	Filtering Precision (%)
HyFD (FD)	NYC Payroll	847	23	2.7
CFD-Discover	NYC Payroll	156	31	19.9
Hydra (DC)	NYC Payroll	2,341	18	0.8
HyFD (FD)	IBM HR	312	19	6.1
CFD-Discover	IBM HR	89	27	30.3
Hydra (DC)	IBM HR	1,156	14	1.2

Meaningful rules are those confirmed by manual expert review as encoding genuine payroll or HR business logic.

The qualitative nature of discovered rules differed substantially across families. FDs captured unconditional structural relationships---{PayBasis} → {RegularHours} and {EmployeeNumber} → {Department}---that are useful as basic sanity checks but cannot express conditional pay policies. CFDs identified context-sensitive rules invisible to FDs: the conditional dependency [Borough = "MANHATTAN", PayBasis = "per Annum"] → BaseSalary ≥ 48,372 reflects the borough-specific minimum salary schedule embedded in NYC municipal pay policy. DCs uniquely captured cross-tuple ordering constraints---the expectation that within the same agency, employees with higher title seniority codes receive higher base salaries---which neither FDs nor CFDs can express.

4.2. Computational Efficiency

Table 4. Runtime (Seconds) Across NYC Payroll Subsamples

Records	HyFD	CFD-Discover	Hydra	Deequ-Manual	ICL
10,000	0.3	1.2	4.1	0.2	8.6
50,000	1.1	5.8	21.3	0.8	22.4
100,000	2.4	12.1	48.7	1.5	38.1
300,000	6.8	28.4	112.5	2.4	62.9
604,182	12.4	47.8	186.3	3.2	94.7

All experiments run on Intel Xeon W-2255 (10 cores, 3.7 GHz), 64 GB RAM, Ubuntu 22.04.

Runtime scaling diverges markedly across methods. HyFD exhibits near-linear growth, reaching 12.4 seconds on the full 604,182-record NYC dataset---roughly 15× faster than Hydra at 186.3 seconds. CFD-Discover occupies an intermediate position at 47.8

seconds, reflecting the additional cost of enumerating value-binding conditions. Deequ-Manual is the fastest at 3.2 seconds, as it executes pre-specified SQL aggregation queries without discovery overhead; this speed advantage, of course, presumes that a domain expert has already invested the (unmetered) effort of authoring 45 constraints. ICL required 94.7 seconds for training and inference on the full dataset.

Figure 1 shows runtime (in seconds, log scale on the y-axis) as a function of dataset size for the three constraint discovery methods and two baselines. HyFD maintains near-linear scaling throughout, rising from 0.3 seconds at 10,000 records to 12.4 seconds at 604,182 records. CFD-Discover follows a moderately steeper trajectory, reaching 47.8 seconds at full scale. Hydra exhibits the steepest growth curve, consuming 186.3 seconds at 604,182 records---approximately 15× the HyFD runtime. Deequ-Manual remains nearly flat below 4 seconds due to its pre-authored constraint set, while ICL grows sub-linearly, reaching 94.7 seconds (As shown in Table 4).

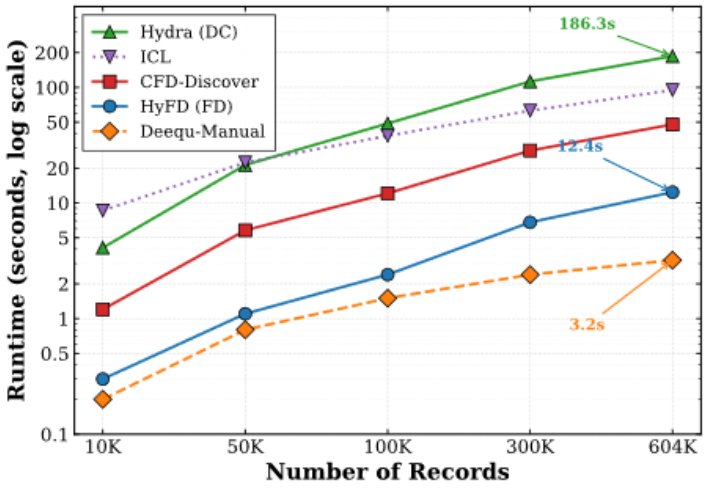

Figure 1. Scalability of Constraint Discovery Methods on NYC Payroll Subsamples

4.3. Inconsistency Detection Performance

Table 5 presents the primary detection results: F1-scores for all five methods across four error injection rates on the NYC Payroll dataset, averaged over five random seeds.

Table 5. Inconsistency Detection F1-Scores on NYC Payroll Data

Method	ε = 1%	ε = 3%	ε = 5%	ε = 10%
HyFD (FD)	0.547	0.619	0.653	0.669
CFD-Discover	0.701	0.757	0.778	0.780
Hydra (DC)	0.660	0.703	0.721	0.716
Deequ-Manual	0.778	0.796	0.800	0.793
ICL	0.403	0.494	0.559	0.625

Values are means over 5 random seeds. Standard deviations range from ±0.008 to ±0.021 across all cells.

CFD-Discover achieves the highest F1 among the three automated discovery methods at every injection rate, reaching 0.778 at ε = 5%---within 0.022 of the Deequ-Manual upper bound (0.800). Hydra attains moderate F1 values (0.660--0.721) that plateau and slightly decline at ε = 10%, attributable to increased false positives when the large DC set encounters higher densities of perturbed records. HyFD consistently trails both CFD-Discover and Hydra, with its recall constrained by the inability to capture conditional and cross-tuple rules. ICL, while improving from 0.403 at ε = 1% to 0.625 at ε = 10%, remains substantially below all constraint-based methods, suggesting that sparse injected errors produce insufficient distributional shift for contrastive learning. This observation aligns

with the pipeline validation paradigm studied by Tu et al [28]. who found that historical statistical baselines outperform general-purpose anomaly detectors for structured enterprise data quality monitoring.

On the IBM HR dataset at ε = 5%, similar ranking patterns hold: CFD-Discover achieves F1 = 0.813, Hydra reaches 0.746, HyFD attains 0.692, and ICL lags at 0.529. The stronger performance of CFDs on this smaller, feature-rich dataset reflects the dense conditional structure of HR attributes (MonthlyIncome conditioned on JobLevel, Department, and JobRole jointly).

Figure 2 reports precision and recall values for all five methods at the 5% error injection rate. HyFD achieves the second-highest precision (0.85) yet the lowest recall (0.53) among constraint-based methods, confirming that unconditional FDs produce precise but narrow verification coverage. CFD-Discover balances both metrics (precision = 0.82, recall = 0.74). Hydra reaches the highest recall (0.81) among all automated methods at the cost of the lowest precision (0.65), reflecting the permissive nature of its large DC set. Deequ-Manual attains the strongest combined precision (0.87) and recall (0.74) through expert-curated rules. ICL records the weakest performance on both axes (precision = 0.58, recall = 0.54).

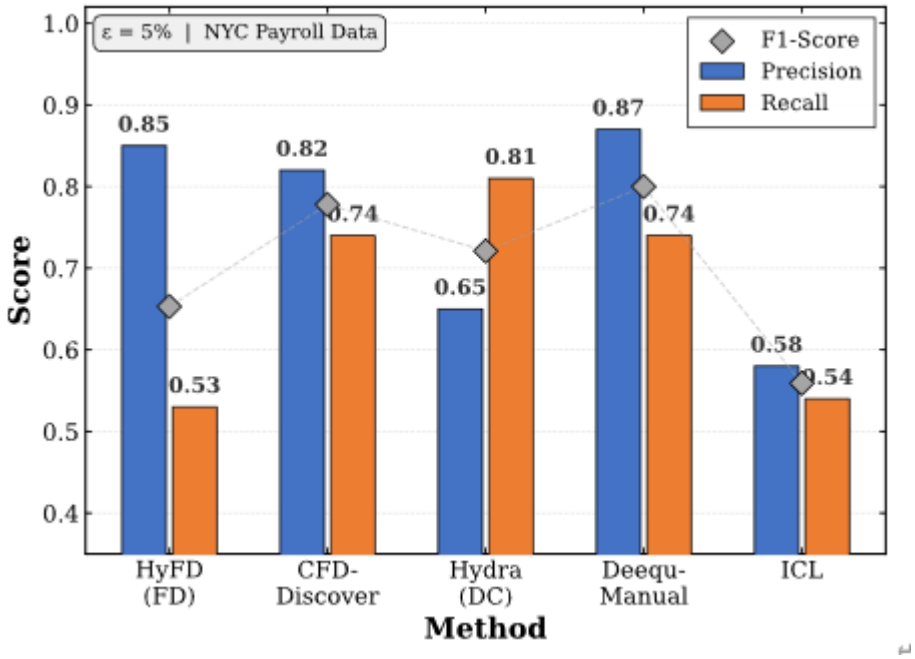

Figure 2. Precision and Recall at ε = 5% on NYC Payroll Data

Figure 3 shows F1-scores plotted against error injection rate (ε = 1%, 3%, 5%, 10%) for all five methods. CFD-Discover and Deequ-Manual follow closely parallel upward trajectories, with CFD-Discover's curve rising from 0.701 to 0.780 and Deequ-Manual's from 0.778 to 0.793---the gap narrowing from 0.077 at ε = 1% to 0.013 at ε = 10%. Hydra's curve rises from 0.660 to 0.721 before declining slightly to 0.716 at ε = 10%, exhibiting a concave shape indicative of increasing false positives at high error densities. HyFD's curve ascends gradually from 0.547 to 0.669 with diminishing returns. ICL exhibits the steepest positive slope, climbing from 0.403 to 0.625, yet remains below all constraint-based alternatives at every tested rate.

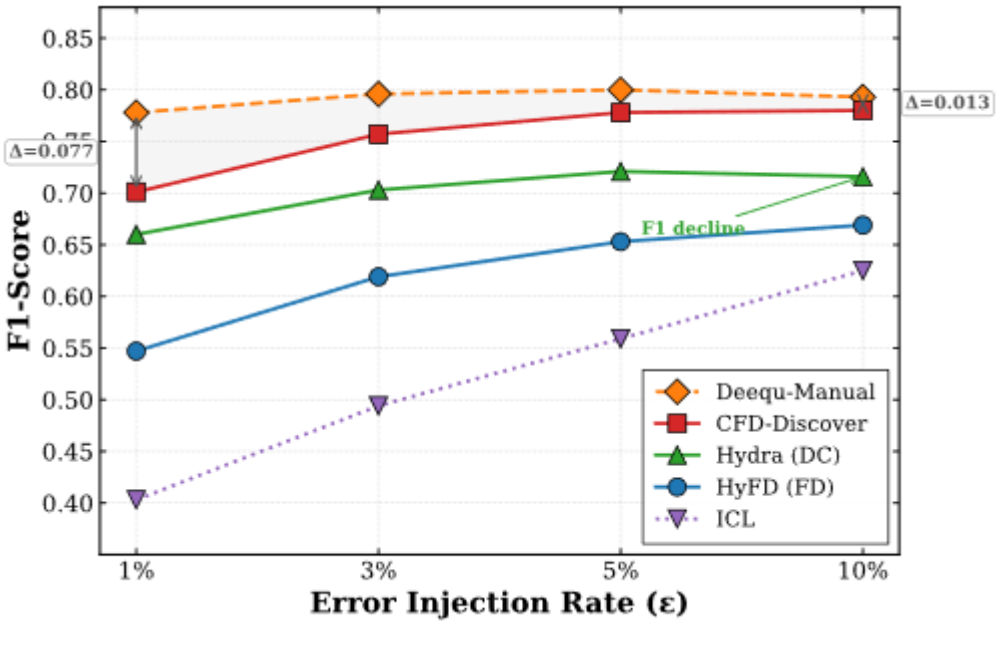

Figure 3. F1-Score Trends Across Error Injection Rates on NYC Payroll Data

4.4. Migration Scenario Analysis

4.4.1. Cross-Field Business Logic Verification

To disaggregate detection performance by error category, Table 6 reports per-category F1-scores at ε = 5% on the NYC Payroll dataset.

Table 6. Per-Category F1-Scores at ε = 5% on NYC Payroll Data

Method	Value Perturbation	Cross-Field Dependency	Semantic Logic
HyFD (FD)	0.724	0.638	0.581
CFD-Discover	0.791	0.803	0.738
Hydra (DC)	0.685	0.756	0.719
Deequ-Manual	0.832	0.811	0.756
ICL	0.612	0.537	0.521

Each cell is the mean F1 over five random seeds within the specified error category.

Value perturbation errors, which directly violate arithmetic dependencies between salary-related fields, are the easiest category for all methods. HyFD achieves 0.724 in this category---its strongest showing---because FDs such as {PayBasis, RegularHours} → {RegularGrossPaid} are directly disrupted by numerical modifications. Cross-field dependency errors reveal the advantage of context-sensitive constraints: CFD-Discover attains 0.803 in this category, outperforming HyFD (0.638) by a margin of 0.165. This gap arises because cross-field errors (altering Department without adjusting BaseSalary) violate conditional rules that unconditional FDs cannot encode. Hydra also performs well on cross-field errors (0.756), as DCs can express inter-attribute ordering expectations across tuple pairs. Semantic logic errors, the most challenging category, require compound rules spanning multiple attributes and business conditions. CFD-Discover (0.738) and Hydra (0.719) both moderately address these errors, while HyFD (0.581) and ICL (0.521) struggle with the multi-attribute nature of semantic violations.

4.4.2. Temporal Consistency Analysis

Multi-year payroll records enable evaluation of temporal consistency rules---expectations that an employee's compensation trajectory adheres to predictable patterns across fiscal years. Using NYC Payroll records from fiscal years 2021 through 2023, a temporal consistency test is constructed: for employees appearing in consecutive years, base salary should not decrease by more than 10% unless accompanied by a recorded change in Leave Status, Title Description, or Pay Basis. This challenge relates to the broader problem of validating recurring data pipelines through historical statistics, and to formal verification of business process temporal ordering, as studied by Corradini et al. Among the three constraint families, none natively supports temporal predicates over tuple sequences [29]. Temporal verification is approximated by constructing a derived relation that joins consecutive-year records for each employee and applying constraint discovery to the resulting paired tuples. Under this formulation, DC discovery achieves the strongest temporal F1 (0.683) because DCs can naturally express pairwise comparisons ($\neg(t_1.\text{FiscalYear} < t_2.\text{FiscalYear} \wedge t_1.\text{BaseSalary} > 1.1 \times t_2.\text{BaseSalary} \wedge t_1.\text{EmployeeID} = t_2.\text{EmployeeID})$). CFD-Discover attains F1 = 0.641 through year-conditioned rules, while HyFD reaches only 0.498, confirming that temporal business logic lies largely beyond the expressiveness of unconditional FDs [30]. This analysis highlights a methodological limitation: temporal verification requires pre-processing that constructs temporal features, adding an engineering step not captured in standard constraint discovery benchmarks.

5. Discussion

5.1. Practical Implications

The experimental findings yield a stratified recommendation for practitioners selecting verification approaches for payroll data migration. When computational budget is constrained and a rapid first-pass verification is needed, FD discovery via HyFD offers acceptable precision (0.85 at $\varepsilon = 5\%$) with minimal runtime (12.4 seconds on 604,182 records), making it suitable for sanity-checking basic structural relationships during iterative migration testing cycles. When detection coverage is the primary objective---particularly for department-specific pay policies and conditional business rules---CFD-Discover provides the most favorable precision-recall balance among automated methods, approaching the accuracy of expert-curated rule sets while eliminating the manual authoring burden. When comprehensive auditing is required and the organization can tolerate higher computational cost and manual post-filtering of spurious rules, DC discovery via Hydra captures the broadest class of business logic constraints, including cross-tuple ordering expectations inaccessible to both FDs and CFDs.

A pragmatic deployment strategy may combine all three families in a staged pipeline: HyFD for initial triage, CFD-Discover for targeted department-level verification, and Hydra for final comprehensive audit. The 47.8-second CFD discovery runtime on 604,182 records places this method well within the operational time budget of typical migration testing windows, which span hours to days. The narrowing gap between CFD-Discover (F1 = 0.780) and Deequ-Manual (F1 = 0.793) at the 10% error rate suggests that automated constraint discovery is approaching the accuracy of manual rule specification, at least for error densities characteristic of migration scenarios with systematic---rather than sporadic---defects.

5.2. Limitations and Future Directions

Several limitations bound the generalizability of these findings. The evaluation relies on synthetically injected errors rather than errors collected from actual system migration events; real migration errors may exhibit different distributions, correlations, and severity profiles than those simulated here. Both datasets originate from U.S. institutional contexts---municipal government and a synthetic corporate environment---and the applicability of discovered constraints to payroll systems governed by different national regulations, collective bargaining agreements, or industry-specific compensation structures has not been tested.

The temporal consistency analysis in Section 4.3 reveals a fundamental gap in current constraint formalisms: none of the three families natively supports sequential or temporal predicates without pre-processing that constructs derived relations. Future work may explore the integration of temporal logic operators into constraint discovery, enabling direct expression of rules such as monotonic salary progression and tenure-based benefit eligibility. Another promising direction involves leveraging large language models for test oracle generation---translating natural-language payroll policy documents into formal constraint specifications that seed and guide the discovery process. Combining such top-down specification with bottom-up data-driven discovery could narrow the gap between automated and expert-authored verification, particularly for complex semantic business rules where purely data-driven approaches remain weakest (F1 = 0.738 for CFDs on semantic logic errors versus 0.756 for expert rules). Extending the evaluation to real migration error logs and to payroll systems from non-U.S. jurisdictions with differing regulatory frameworks would further validate the practical relevance of these findings.

References

1. Z. Abedjan, L. Golab, and F. Naumann, "Profiling relational data: A survey," *The VLDB Journal*, vol. 24, no. 4, pp. 557--581, 2015.
2. S. Schelter, D. Lange, P. Schmidt, M. Celikel, F. Biessmann, and A. Grafberger, "Automating large-scale data quality verification," *Proceedings of the VLDB Endowment*, vol. 11, no. 12, pp. 1781--1794, 2018.
3. P. Bohannon, W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis, "Conditional functional dependencies for data cleaning," in *Proceedings of the IEEE International Conference on Data Engineering*, 2007, pp. 746--755.
4. X. Chu, I. F. Ilyas, and P. Papotti, "Discovering denial constraints," *Proceedings of the VLDB Endowment*, vol. 6, no. 13, pp. 1498--1509, 2013.
5. T. Papenbrock, J. Ehrlich, J. Marten, T. Neubert, J.-P. Rudolph, M. Schönberg, J. Zwiener, and F. Naumann, "Functional dependency discovery: An experimental evaluation of seven algorithms," *Proceedings of the VLDB Endowment*, vol. 8, no. 10, pp. 1082--1093, 2015.
6. Y. Zhang, "Evaluation of differential privacy and federated learning for AI-driven customer service applications," *Journal of Sustainability, Policy, and Practice*, vol. 2, no. 2, pp. 55--66, 2026.
7. E. H. M. Pena, E. C. de Almeida, and F. Naumann, "Fast algorithms for denial constraint discovery," *Proceedings of the VLDB Endowment*, vol. 16, no. 4, pp. 684--696, 2022.
8. S. Kruse and F. Naumann, "Efficient discovery of approximate dependencies," *Proceedings of the VLDB Endowment*, vol. 11, no. 7, pp. 759--772, 2018.
9. F. Chiang and R. J. Miller, "Discovering data quality rules," *Proceedings of the VLDB Endowment*, vol. 1, no. 1, pp. 1166--1177, 2008.
10. M. Zhong, "Multi-dimensional feature analysis and evaluation methods for anomalous fund flow identification in cross-border financial transactions," *Journal of Science, Innovation & Social Impact*, vol. 2, no. 2, pp. 1--13, 2026.
11. A. Fariha, A. Tiwari, A. Radhakrishna, S. Gulwani, and A. Meliou, "Conformance constraint discovery: Measuring trust in data-driven systems," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2021, pp. 499--512.
12. M. Dallachiesa, A. Ebaid, A. Eldawy, A. K. Elmagarmid, I. F. Ilyas, M. Ouzzani, and N. Tang, "NADEEF: A commodity data cleaning system," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2013, pp. 541--552.
13. P. T. Chung, "Enhancing dental polymer formulation through interpretable machine learning: A comparative analysis of feature selection and algorithm performance," in *Proceedings of the 2025 6th International Conference on Computer Science and Management Technology*, Dec. 2025, pp. 234--241.
14. Y. Li, "Comparative analysis of illumination normalization methods for autonomous driving under challenging lighting conditions," in *Proceedings of the 2025 6th International Conference on Computer Science and Management Technology*, Dec. 2025, pp. 633--639.
15. Y. Wang, "Accuracy evaluation of machine learning-based hospital resource demand forecasting during infectious disease surges: A comparative analysis," *Journal of Science, Innovation & Social Impact*, vol. 2, no. 1, pp. 314--327, 2026.
16. T. Rekatsinas, X. Chu, I. F. Ilyas, and C. Ré, "HoloClean: Holistic data repairs with probabilistic inference," *Proceedings of the VLDB Endowment*, vol. 10, no. 11, pp. 1190--1201, 2017.
17. M. Mahdavi, Z. Abedjan, R. Castro Fernandez, S. Madden, M. Ouzzani, M. Stonebraker, and N. Tang, "Raha: A configuration-free error detection system," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2019, pp. 865--882.
18. S. Han, X. Hu, H. Huang, M. Jiang, and Y. Zhao, "ADBench: Anomaly detection benchmark," in *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022.
19. T. Papenbrock and F. Naumann, "A hybrid approach to functional dependency discovery," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2016, pp. 821--833.

20. T. Bleifuß, S. Kruse, and F. Naumann, "Efficient denial constraint discovery with Hydra," *Proceedings of the VLDB Endowment*, vol. 11, no. 3, pp. 311--323, 2017.
21. T. Shenkar and L. Wolf, "Anomaly detection for tabular data with internal contrastive learning," in *Proceedings of the International Conference on Learning Representations (ICLR 2022)*, 2022.
22. Q. Zhang, "Adaptive differential privacy mechanism for federated document classification: A gradient-clipping optimization approach," in *Proceedings of the 2025 6th International Conference on Computer Science and Management Technology*, Dec. 2025, pp. 672--678.
23. Y. Wang, "Practical AI approaches for community infection early warning: From public data to actionable insights," in *Proceedings of the 2025 6th International Conference on Computer Science and Management Technology*, Dec. 2025, pp. 1545--1552.
24. M. Han, "Privacy-preserving collaborative learning across healthcare institutions: An adaptive approach with gradient compression and dynamic privacy budget allocation," in *Proceedings of the 2025 6th International Conference on Computer Science and Management Technology*, Dec. 2025, pp. 679--684.
25. D. Liang and C. Cai, "Optimizing large-scale contract review through data analytics: Practical evidence from IPO audits," in *Proceedings of the 2025 6th International Conference on Computer Science and Management Technology*, Dec. 2025, pp. 242--249.
26. G. Cong, W. Fan, F. Geerts, X. Jia, and S. Ma, "Improving data quality: Consistency and accuracy," in *Proceedings of the 33rd International Conference on Very Large Data Bases*, 2007, pp. 315--326.
27. D. Tu, Y. He, W. Cui, S. Ge, H. Zhang, S. Han, D. Zhang, and S. Chaudhuri, "Auto-Validate by-History: Auto-program data quality constraints to validate recurring data pipelines," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.
28. F. Corradini, F. Fornari, A. Polini, B. Re, F. Tiezzi, and A. Vandin, "BProVe: A formal verification framework for business process models," in *Proceedings of the IEEE/ACM 32nd International Conference on Automated Software Engineering*, 2017, pp. 217--228.
29. Y. Li, "Performance benchmarking and optimization strategies for depth estimation algorithms in unstructured environments," *Journal of Sustainability, Policy, and Practice*, vol. 2, no. 2, pp. 32--43, 2026.
30. P. T. Chung, "Comparative evaluation of machine learning algorithms for spectrophotometric dental shade classification," *Journal of Sustainability, Policy, and Practice*, vol. 2, no. 1, pp. 204--214, 2026.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of Publisher and/or the editor(s). Publisher and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.